

Handbook of Clouds and Big Data

Theory and Practice

Gregor von Laszewski

Geoffrey C. Fox

Judy Qiu

laszewski@gmail.com

Monday 8th January, 2018 19:26

Copyright © 2017
Gregor von Laszewski
Geoffrey C. Fox
Judy Qiu

laszewski@gmail.com

<https://github.com/cloudmesh/classes>

First printing by Gregor von Laszewski, October 2017



Contents

I

Preface

1	Introduction	33
1.1	Citation	34
1.2	Authors	34
1.3	Acknowledgements	35
1.4	About	35
1.5	Other Results	35
1.6	Contributing	35
1.6.1	Class Contributions	36
1.6.2	Community Contributions	36
1.6.3	Contributors	36
1.7	Conventions	37
1.7.1	Videos	37
1.7.2	Slides	37
1.7.3	Images	37
1.7.4	URLs	37
1.7.5	Boxes	37
1.8	Exercises	38

II

Syllabus

2	E516: Introduction to Cloud Computing	41
2.1	Course Description	41

2.2	Course pre-requisites	41
2.3	Registration Information	42
2.4	Teaching and learning methods	42
2.5	Covered Topics	42
2.6	Student learning outcomes	42
2.7	Grading	43
2.8	Representative bibliography	43
2.9	Lectures and Lecture Material	43
2.9.1	Communication Track	43
2.9.2	Theory Track	44
2.9.3	Programming Track	44
3	E616/I524: Advanced Cloud Computing	47
3.1	Course Description	47
3.2	Course pre-requisites	47
3.3	Course Registration	47
3.4	Teaching and learning methods	48
3.5	Covered Topics	48
3.6	Student learning outcomes	49
3.7	Grading	49
3.8	Representative bibliography	49
3.9	Lectures and Lecture Material	49
3.9.1	Communication Track	50
3.9.2	Theory Track	50
3.9.3	Programming Track	51
3.9.4	DevOps	51
3.9.5	Cloud	51
3.10	Containers	52
4	E222: Intelligent Systems Engineering II	53
4.1	Course Description	53
4.2	Course pre-requisites	53
4.3	Course Registration	53
4.4	Teaching and learning methods	53
4.5	Covered Topics	53
4.6	Student learning outcomes	54
4.7	Grading	54
4.8	Representative bibliography	54
4.9	Lectures and Lecture Material	55
4.9.1	Communication Track	55
4.9.2	Theory Track	55
4.9.3	Programming Track	55
4.10	Containers	56

5	Course Policies	59
5.1	Communication and Use of CANVAS	59
5.2	Online and Office Hours	59
5.3	Class Material	60
5.4	HID	60
5.5	Notebook	60
5.6	Blog	61
5.7	Calendar I524, E616, E516	61
5.8	Incomplete	61
5.9	Waitlist	62
5.10	Auditing the class	63
5.11	Resource restrictions	63
5.12	Office Hours	63
5.13	Plagiarism	64
5.14	Tutorials, Topic Paper, Term Paper, Project Report	65
5.14.1	Team	66
5.14.2	Common Deliverables	66
5.14.3	Project Paper	67
5.14.4	Term Paper	68
5.15	Grading	68
5.15.1	Grades on Canvas	68
5.15.2	Discussion about Grades	69
6	Piazza	71
6.1	Access to Piazza from Canvas	71
6.2	Verify you are on Piazza via a post	74
6.3	Making Piazza Work	74
6.4	Towards good questions	74
6.5	Guide on how to ask good questions	75
6.6	Piazza class Links	76
6.6.1	Current Classes	76
6.6.2	Previous Classes	76
6.7	Piazza Curation	77
6.8	Read the Originals, not just the e-mail	78
6.9	Exercises	78

7	Documenting Scientific Research	81
7.1	Overview	81

7.2	Plagiarism	82
7.2.1	Plagiarism Definition	82
7.2.2	Plagiarism Policies	82
7.2.3	Plagiarism Resources	83
7.2.4	Pattern	83
7.2.5	Tutorials	83
7.2.6	How to Recognize Plagiarism	83
7.3	Acknowledgements	84
7.4	Writing a Scientific Article or Conference Paper	85
7.4.1	Professional Paper Format	85
7.4.2	Submission Requirements	86
7.4.3	Microsoft Word vs. \LaTeX	86
7.4.4	Working in a Team	87
7.4.5	Time Management	87
7.4.6	Paper and Report Checklist	87
7.4.7	Content	88
7.4.8	Submission	88
7.4.9	Bibliography	88
7.4.10	Writing	89
7.4.11	Citation Issues and Plagiarism	89
7.4.12	Character Errors	89
7.4.13	Structural Issues	90
7.4.14	Figures and Tables	90
7.4.15	Example Paper	91
7.4.16	Creating the PDF from \LaTeX on your Computer	91
7.4.17	Class Specific README.md	91
7.4.18	Exercises	92
8	Introduction to \LaTeX	93
8.1	Installation	93
8.1.1	Local Install	93
8.1.2	Online Services	94
8.2	Basic \LaTeX Elements	95
8.2.1	Characters	96
8.2.2	Highlighting Text	96
8.2.3	Sections	96
8.2.4	Empty Lines	97
8.2.5	Itemize	97
8.2.6	Enumerate	97
8.2.7	Descriptions	97
8.2.8	Images	97
8.2.9	Tables	99
8.2.10	Labels	99
8.2.11	Mathematics	100
8.3	Advanced topics	101
8.3.1	ACM and IEEE Proceedings Format	101
8.3.2	Generating and Managing Images	101
8.3.3	Colored Boxes	102
8.3.4	Slides	102

8.3.5	LaTeX vs. X	103
8.4	Editing	104
8.4.1	Emacs	104
8.4.2	Vi/Vim	104
8.4.3	TeXshop	105
8.4.4	LyX	105
8.4.5	WYSIWYG locally	105
8.4.6	Markdown and L ^A T _E X	106
8.4.7	Including RST into LaTeX	106
8.4.8	pyCharm	107
8.4.9	MSWord	107
8.5	The LaTeX Cycle	107
8.6	Tips	107
8.6.1	Colorful Output	108
8.6.2	References	108
9	Managing Bibliographies	111
9.0.1	Integrating Bibliographies	111
9.1	Entry types	112
9.1.1	Source code References	113
9.1.2	Researching proper bibtex entries	114
9.1.3	Article in a conference proceedings	116
9.1.4	What are the differnt entry types and fields	119
9.1.5	InProceedings	119
9.1.6	TechReport	120
9.1.7	Article	120
9.1.8	Proceedings	121
9.1.9	Wikipedia Entry	121
9.1.10	Blogs	122
9.1.11	Web Page	122
9.1.12	Book	123
9.2	Integrating Bibtex entries into Other Systems	125
9.2.1	Bibtex import to MSWord	125
9.3	Other Reference Managers	125
9.3.1	Endnote	126
9.3.2	Mendeley	126
9.3.3	Zotero	126
9.3.4	Paperpile	126
10	Editors	127
10.1	Basic Emacs	127
10.1.1	Org Mode	129
10.1.2	Programming Python with Emacs	130
10.1.3	Emacs Keys in a Terminal	130
10.1.4	L ^A T _E X and Emacs	130

11	Other Formats	131
11.1	reStructuredText	131
11.1.1	Links	131
11.1.2	Source	131
11.1.3	Sections	132
11.1.4	ListTable	132
11.1.5	Exceltable	132
11.1.6	Boxes	132
11.1.7	Sidebar directive	133
11.1.8	Sphinx Prompt	133
11.1.9	Programm examples	133
11.1.10	Hyperlinks	133
11.1.11	Todo	134
11.2	Markdown	134
11.2.1	Tools	135
11.3	Communicating Research in Other Ways	135
11.3.1	Blogs	135
11.3.2	Sphinx	136
11.3.3	Notebooks	136
12	Book Format	139
12.1	Paragraphs of Text	139
12.2	Citation	140
12.3	Lists	140
12.3.1	Numbered List	140
12.3.2	Bullet Points	140
12.3.3	Descriptions and Definitions	140
12.4	Theorems	140
12.4.1	Several equations	141
12.4.2	Single Line	141
12.5	Definitions	141
12.6	Notations	142
12.7	Remarks	142
12.8	Corollaries	142
12.9	Propositions	143
12.9.1	Several equations	143
12.9.2	Single Line	143
12.10	Examples	143
12.10.1	Equation and Text	143
12.10.2	Paragraph of Text	144
12.11	Exercises	144
12.12	Problems	145
12.13	Vocabulary	145
12.14	Tables	145

12.15	Figures	145
12.15.1	Colored Boxes	145
12.15.2	Section References	146
12.15.3	Partial Compile	146

13	Linux	151
13.1	History	151
13.2	Shell	151
13.3	Multi-command execution	153
13.4	Keyboard Shortcuts	154
13.5	.bashrc and .bash_profile	154
13.6	Exercises	154
14	SSH	157
14.1	Generate a SSH key	157
14.2	Add or Replace Passphrase	158
14.3	SSH Add and Agent	159
14.4	SSH Config	159
14.5	SSH Port Forwarding	159
14.6	Upload the key to gitlab	159
14.7	Using SSH on Mac OS X	160
14.8	Using SSH on Linux	160
14.9	Using SSH on Raspberry Pi 3	160
14.10	SSH on Windows	160
14.10.1	OpenSSH Client (Beta)	160
14.10.2	Using SSH from Cygwin	161
14.10.3	SSH from putty	161
14.10.4	Chocolatey	162
14.11	Tips	163
14.12	SSH to FutureSystems Resources	164
14.13	Testing your ssh key	165
14.14	References	165
14.15	Exercises	165
15	Github	167
15.1	Obview	167
15.2	Upload Key	168
15.3	Fork	168
15.4	Rebase	168
15.5	Remote	168

15.6	Pull Request	169
15.7	Branch	169
15.8	Checkout	169
15.9	Merge	169
15.10	GUI	169
15.11	Windows	170
15.12	Git from the Commandline	170
15.13	Configuration	170
15.14	Upload your public key	171
15.15	Working with a directory that was provided for you	171
15.16	README.yml and notebook.md	172
15.17	Contributing to the Document	172
15.17.1	Clone	172
15.17.2	Merge	173
15.17.3	Resources	173
15.18	Exercises	174
16	Virtual Box	175
16.1	Instalation	176
16.2	Guest additions	176
16.3	Exercises	177

VI

Cloud Resources

17	FutureSystems	181
17.1	FutureSystems evolved from FutureGrid	181
17.2	Creating Portal Account	181
17.3	SSH Key Generation using ssh-keygen command	182
17.4	Shell Access via SSH	182
17.5	Advanced SSH	182
17.6	SSH Key Generation via putty	182
18	Chameleon Cloud	183
18.1	Overview	183
18.2	Resources	184
18.3	Hardware	185
18.3.1	Standard Cloud Units	185
18.3.2	Network	186
18.3.3	Shared Storage	186
18.3.4	Heterogeneous Compute Hardware	187
18.3.5	Live updates	187

18.4	Getting Started	187
18.4.1	Step 1: Create a Chameleon account	188
18.4.2	Step 2: Create or join a project	188
18.4.3	Step 3: Start using Chameleon!	188
18.5	Charge Rates	189
18.5.1	Service Units	189
18.5.2	Project Allocation Size	190
18.6	OpenStack Virtual Machines	190
18.6.1	Web Interface	191
18.6.2	OpenStack REST Interfaces	196
18.6.3	Downloading and uploading data	197
18.7	Horizon Graphical User Interface	197
18.7.1	Configure resources	197
18.7.2	Interact with resources	202
18.7.3	Use FPGAs	204
18.7.4	Next Step	204
18.8	HEAT	204
18.8.1	Supporting Complex Appliances	204
18.8.2	Chameleon Appliance Catalog	205
18.8.3	Deployment	205
18.8.4	Heat Template	208
18.8.5	Customizing an existing template	211
18.8.6	Writing a new template	215
18.8.7	Sharing new complex appliances	216
18.8.8	Advanced topics	217
18.9	Bare Metal	220
18.10	Frequently Asked Questions	220
18.10.1	Appliances	221
18.10.2	Bare Metal Troubleshooting	222
18.10.3	OpenStack KVM Troubleshooting	223

VII

Python

19	Introduction	227
19.1	Introduction to Python	227
19.2	References	228
20	Install	231
20.1	Python Installation	231
20.1.1	Managing custom Python installs	231
20.1.2	Installation without pyenv	234
20.1.3	Anaconda and Miniconda	235
20.2	Interactive Python	237
20.3	REPL (Read Eval Print Loop)	237
20.4	Python 3 Features in Python 2	238

21	Language	239
21.1	Statements and Strings	239
21.2	Variables	239
21.3	Data Types	240
21.3.1	Booleans	240
21.3.2	Numbers	240
21.4	Module Management	241
21.4.1	Import Statement	241
21.4.2	The from ... import Statement	241
21.5	Date Time in Python	242
21.6	Control Statements	243
21.6.1	Comparision	243
21.6.2	Iteration	244
21.7	Datatypes	244
21.7.1	Lists	244
21.7.2	Sets	245
21.7.3	Removal and Testing for Membership in Sets	246
21.7.4	Dictionaries	247
21.7.5	Dictionary Keys and Values	247
21.7.6	Counting with Dictionaries	248
21.8	Functions	248
21.9	Classes	249
21.10	Modules	250
21.11	Lambda Expressions	251
21.12	Generators	251
21.13	Non Blocking Threads	251
22	Data Management	253
22.1	Formats	253
22.1.1	Pickle	253
22.1.2	Text Files	254
22.1.3	CSV Files	254
22.1.4	Excel spread sheets	254
22.1.5	YAML	254
22.1.6	JSON	254
22.1.7	XML	254
22.1.8	RDF	254
22.1.9	PDF	255
22.1.10	HTML	255
22.1.11	ConfigParser	255
22.1.12	ConfigDict	255
22.2	Encryption	255
22.3	Database Access	256
22.3.1	Exercises	256

23	Libraries	257
23.1	Installing Libraries	257
23.2	Using pip to Install Packages	257
23.3	GUI	258
23.3.1	GUIZero	258
23.3.2	Kivy	258
23.4	Formatting and Checking Python Code	258
23.5	Using autopep8	258
23.6	Writing Python 3 Compatible Code	259
23.7	Using Python on FutureSystems	259
23.8	Ecosystem	259
23.8.1	pypi	259
23.8.2	Alternative Installations	259
23.9	Resources	261
23.9.1	Jupyter Notebook Tutorials	261
23.10	Exercises	261
23.11	Python for Big Data	262
23.11.1	An Example with Pandas, NumPy and Matplotlib	262
23.11.2	Summary of Useful Libraries	267
23.11.3	Big Data Libraries	267
23.12	Parsing Data	269
23.12.1	notebook.md Parser	269
23.12.2	Video Length	270
23.12.3	Dask	270
24	Cloudmesh Command Shell	271
24.1	CMD5	271
24.1.1	Resources	271
24.1.2	Creating a Python Development Environment	271
24.1.3	Installation from source	271
24.1.4	Execution	272
24.1.5	Create your own Extension	272
24.1.6	Excercises	274

25	Numpy	277
25.1	Float Range	277
25.2	Arrays	278
25.3	Array Operations	278
25.4	Linear Algebra	279
25.5	Resources	279

26	Scipy	281
26.1	Introduction	281
26.2	References	285
27	OpenCV	287
27.1	Overview	287
27.2	Installation	287
27.3	A Simple Example	288
27.3.1	Loading an image	288
27.3.2	Displaying the image	288
27.3.3	Scaling and Rotation	288
27.3.4	Gray-scaling	289
27.3.5	Image Thresholding	289
27.3.6	Edge Detection	289
27.4	Additional Features	290
27.5	Secchi Disk	291
27.5.1	Overview	291
27.6	Setup for OSX	291
27.6.1	Step 1: Record the video	291
27.6.2	Step 2: Analyse the images from the Video	291
27.7	Black and white	296
28	NIST Pedestrian and Face Detection	297
28.0.1	Introduction	298
28.0.2	Deployment by Ansible	299
28.0.3	Cloudmesh for Provisioning	300
28.0.4	Roles Explained for Installation	300
28.0.5	Instructions for Deployment	301
28.0.6	OpenCV in Python	302
28.0.7	Human and Face Detection in OpenCV	304
28.0.8	Pedestrian Detection using HOG Descriptor	305
28.0.9	Processing by Apache Spark	306
28.0.10	Results for 100+ images by Spark Cluster	307
29	Python Fingerprint Example	309
29.1	Overview	309
29.2	Objectives	310
29.3	Prerequisites	310
29.4	Implementation	310
29.5	Utility functions	311
29.6	Dataset	312
29.7	Data Model	313
29.7.1	Utilities	313
29.7.2	Checksum	313
29.7.3	Path	313
29.7.4	Image	314

29.7.5	Mindtct	314
29.7.6	Bozorth3	314
29.7.7	Running Bozorth3	315
29.8	Plotting	316
29.9	Putting it all Together	317

IX

Python Cloudmesh

30	Cloudmesh	321
30.1	Configuration	321
30.2	Storage	323
30.2.1	sqlite3	323
30.2.2	Context	324

X

Cloud Computing

31	Cloud Computing Fundamentals	327
31.1	Introduction	327
31.2	Data Center Model	327
31.3	Data Intensive Sciences	328
31.4	IaaS, PaaS and SaaS	328
31.5	Challenges	328
32	IaaS	329
32.1	Growth of Virtual Machines	329
32.2	Implementation Levels	330
32.3	Tools and Mechanisms	330
32.4	CPU, Memory & I/O Devices	330
32.5	Clusters and Resource Management	330
32.6	Data Center Automation	331
32.7	Clouds in the Workplace	331
32.8	Checklists and Challenges	331
32.9	Data Center Setup	332
32.10	Cultivating Clouds	332

XI

Data Management

33	NoSQL	335
33.1	RDBMS vs. NoSQL	335
33.2	NoSQL Characteristics	335
33.3	BigTable	336

33.4	HBase	336
33.5	HBase Coding	336
33.6	Indexing Applications	336
33.7	Related Work	337
33.8	Indexamples	337
33.9	Indexing 101	337
33.10	Social Media Searches	337
33.11	Analysis Algorithms	338

XII

SaaS

34	Search Engine	341
34.1	Google Components	341
34.2	Google Architecture	341
34.3	Google History	341

XIII

MapReduce

35	MapReduce	345
35.1	Apache Data Analysis OpenStack	345
35.2	MapReduce	345
35.3	Hadoop Framework	346
35.4	Hadoop Tasks	346
35.5	Fault Tolerance	346
35.6	Hadoop WordCount on VMs	346
35.7	Programming on a Compute Cluster	347
35.8	How Hadoop Runs on a MapReduceJob	347
35.9	Literature Review	347
35.10	Introduction to BLAST	347
35.11	BLAST Parallelization	348
35.12	SIMD vs MIMD;SPMD vs MPMD	348
35.13	Data Locality	348
35.14	Optimal Data Locality	348
35.15	Task Granularity	349
35.16	Resource Utilization and Speculative Execution	349
36	Iterative Map Reduce	351
36.1	Introduction to MapReduce	351
36.2	Google Search Engine 1	351
36.3	Google Search Engine 2	352

36.4	Hadoop PageRank	352
36.5	Discussions and ParallelThinking	352
36.6	Hadoop Extensions	352
36.7	Iterative MapReduce Models	353
36.8	Parallel Processes	353
36.9	Static and Variable Data	353
36.10	MapReduce Model Comparison	354
36.11	Twister K-means	354
36.12	Coding and Iterative Alternatives	354

XIV

Internet of Things

37	IoT	357
37.1	Everyday Data	357
37.2	Streaming the Data Ocean	357
37.3	Streams of Events	357
37.4	Faults & Frameworks	358
37.5	Spouts to Bolts	358

XV

Big Data Applications

37.6	Introduction	361
37.6.1	Motivation	362
37.7	Overview of Data Science	367
37.7.1	Data Science generics and Commercial Data Deluge	367
37.7.2	Data Deluge and Scientific Applications and Methodology	369
37.7.3	Clouds and Big Data Processing; Data Science Process and Analytics	370
37.7.4	Clouds	370
37.8	Health Informatics Case Study	371
37.8.1	X-Informatics Case Study: Health Informatics	372
37.9	e-Commerce and LifeStyle Case Study	376
37.9.1	Recommender Systems: Introduction	376
37.9.2	Recommender Systems: Examples and Algorithms	378
37.9.3	Item-based Collaborative Filtering and its Technologies	379
37.10	Physics Case Studies	380
37.10.1	Looking for Higgs Particles, Bumps in Histograms, Experiments and Accelerators (Part 1)	380
37.10.2	Looking for Higgs Particles: Random Variables, Physics and Normal Distributions	382
37.10.3	Looking for Higgs Particles: Random Numbers, Distributions and Central Limit Theorem (Part 3)	383
37.10.4	SKA Square Kilometer Array	384
37.11	Radar Case Study	385
37.11.1	Introduction	385

37.11.2	Remote Sensing	385
37.11.3	Ice Sheet Science	385
37.11.4	Global Climate Change	385
37.11.5	Radio Overview	386
37.11.6	Radio Informatics	386
37.12	Sensors Case Study	386
37.12.1	Internet of Things	386
37.12.2	Robotics and IOT Expectations	386
37.12.3	Industrial Internet of Things	386
37.12.4	Sensor Clouds	387
37.12.5	Earth/Environment/Polar Science data gathered by Sensors	387
37.12.6	Ubiquitous/Smart Cities	387
37.12.7	U-Korea (U=Ubiquitous)	387
37.12.8	Smart Grid	387
37.12.9	Resources	387
37.13	Sports Case Study	388
37.13.1	Sports Informatics I : Sabermetrics (Basic)	388
37.13.2	Sports Informatics II : Sabermetrics (Advanced)	389
37.13.3	PITCHf/X	390
37.13.4	Sports Informatics III : Other Sports	390
37.14	Big Data Use Cases Survey	391
37.14.1	Overview of NIST Big Data Public Working Group (NBD-PWG) Process and Results	391
37.14.2	51 Big Data Use Cases	394
37.14.3	Features of 51 Big Data Use Cases	396
37.15	Web Search and Text Mining	399
37.15.1	Web Search and Text Mining I	400
37.15.2	Web and Document/Text Search: The Problem	400
37.15.3	Information Retrieval leading to Web Search	400
37.15.4	History behind Web Search	400
37.15.5	Key Fundamental Principles behind Web Search	400
37.15.6	Information Retrieval (Web Search) Components	401
37.15.7	Search Engines	401
37.15.8	Boolean and Vector Space Models	401
37.15.9	Web crawling and Document Preparation	401
37.15.10	Indices	401
37.15.11	TF-IDF and Probabilistic Models	401
37.15.12	Resources	401
37.15.13	Web Search and Text Mining II	402
37.15.14	Data Analytics for Web Search	402
37.15.15	Link Structure Analysis including PageRank	402
37.15.16	Web Advertising and Search	402
37.15.17	Clustering and Topic Models	402
37.15.18	Resources	403
37.16	Technology Training - kNN & Clustering	403
37.16.1	Recommender Systems - K-Nearest Neighbors	403
37.16.2	Clustering and heuristic methods	404

38	Big Data Applications and Software	409
38.1	Overview	409
38.2	Introduction	409
38.3	Application Structure	410
38.4	Application Aspects	410
38.5	Applications	410
38.6	Other	411
39	Technology Overview	413
39.1	Workflow-Orchestration	413
39.1.1	ODE	413
39.1.2	ActiveBPEL	413
39.1.3	Airavata	414
39.1.4	Pegasus	414
39.1.5	Kepler	414
39.1.6	Swift	414
39.1.7	Taverna	415
39.1.8	Triana	415
39.1.9	Trident	415
39.1.10	BioKepler	415
39.1.11	Galaxy	416
39.1.12	Jupyter and IPython	416
39.1.13	Dryad	416
39.1.14	Naiad	417
39.1.15	Oozie	417
39.1.16	Tez	417
39.1.17	Google FlumeJava	418
39.1.18	Crunch	418
39.1.19	Cascading	418
39.1.20	Askalon	419
39.1.21	Scalding	419
39.1.22	e-Science Central	419
39.1.23	Azure Data Factory	419
39.1.24	Google Cloud Dataflow	420
39.1.25	NiFi (NSA)	420
39.1.26	Jitterbit	420
39.1.27	Talend	421
39.1.28	Pentaho	421
39.1.29	Apatar	421
39.1.30	Docker Compose	421
39.1.31	KeystoneML	422
39.2	Application and Analytics	422
39.2.1	Mahout (384)	422
39.2.2	MLlib	422
39.2.3	MLbase	422
39.2.4	DataFu	423

39.2.5	R	423
39.2.6	pbdR	423
39.2.7	Bioconductor	423
39.2.8	ImageJ	424
39.2.9	OpenCV	424
39.2.10	Scalapack	424
39.2.11	PetSc	424
39.2.12	PLASMA MAGMA	424
39.2.13	Azure Machine Learning	425
39.2.14	Google Prediction API & Translation API	425
39.2.15	scikit-learn	426
39.2.16	PyBrain (531)	426
39.2.17	CompLearn	426
39.2.18	DAAL(Intel)	427
39.2.19	Caffe	427
39.2.20	Torch	427
39.2.21	Theano	427
39.2.22	DL4j	428
39.2.23	H2O	428
39.2.24	IBM Watson	428
39.2.25	Oracle PGX	428
39.2.26	GraphLab	429
39.2.27	GraphX	429
39.2.28	IBM System G	429
39.2.29	GraphBuilder(Intel)	430
39.2.30	TinkerPop	430
39.2.31	Parasol	430
39.2.32	Dream:Lab	430
39.2.33	Google Fusion Tables	431
39.2.34	CINET	431
39.2.35	NWB	431
39.2.36	Elasticsearch	431
39.2.37	Kibana	432
39.2.38	Logstash	432
39.2.39	Graylog	432
39.2.40	Splunk	433
39.2.41	Tableau	433
39.2.42	D3.js	433
39.2.43	three.js	433
39.2.44	Potree	434
39.2.45	DC.js	434
39.2.46	TensorFlow	434
39.2.47	CNTK	434
39.3	Application Hosting Frameworks	435
39.3.1	Google App Engine	435
39.3.2	AppScale	435
39.3.3	Red Hat OpenShift	435
39.3.4	Heroku	436
39.3.5	Aerobatic	436
39.3.6	AWS Elastic Beanstalk	436
39.3.7	Azure	436

39.3.8	Cloud Foundry	437
39.3.9	Pivotal	437
39.3.10	IBM BlueMix	437
39.3.11	(Ninefold)	437
39.3.12	Jelastic	437
39.3.13	Stackato	438
39.3.14	appfog	438
39.3.15	CloudBees	438
39.3.16	Engine Yard (577)	438
39.3.17	(CloudControl)	439
39.3.18	dotCloud (429)	439
39.3.19	Dokku	439
39.3.20	OSGi	439
39.3.21	HUBzero	439
39.3.22	OODT	439
39.3.23	Agave	439
39.3.24	Atmosphere	440
39.4	High level Programming	440
39.4.1	Kite	440
39.4.2	Hive	440
39.4.3	HCatalog	441
39.4.4	Tajo	441
39.4.5	Shark	441
39.4.6	Phoenix	442
39.4.7	Impala	442
39.4.8	MRQL	443
39.4.9	SAP HANA	443
39.4.10	HadoopDB	443
39.4.11	PolyBase	443
39.4.12	Pivotal HD/Hawq	444
39.4.13	Presto	444
39.4.14	Google Dremel	444
39.4.15	Google BigQuery	444
39.4.16	Amazon Redshift	445
39.4.17	Drill	445
39.4.18	Kyoto Cabinet	445
39.4.19	Pig	446
39.4.20	Sawzall	446
39.4.21	Google Cloud DataFlow	446
39.4.22	Summingbird	446
39.4.23	Lumberyard	447
39.5	Streams	447
39.5.1	Storm	447
39.5.2	S4	447
39.5.3	Samza	448
39.5.4	Granules	448
39.5.5	Neptune	448
39.5.6	Google MillWheel	448
39.5.7	Amazon Kinesis	449
39.5.8	LinkedIn	449
39.5.9	Twitter Heron	449

39.5.10	Databus	450
39.5.11	Facebook Puma/Ptail/Scribe/ODS	450
39.5.12	Azure Stream Analytics	450
39.5.13	Floe	451
39.5.14	Spark Streaming (567)	451
39.5.15	Flink Streaming	451
39.5.16	DataTurbine	451
39.6	Basic Programming Model and Runtime, SPMD, MapReduce	451
39.6.1	Hadoop	451
39.6.2	Spark (543)	452
39.6.3	Twister	452
39.6.4	MR-MPI	452
39.6.5	Stratosphere (Apache Flink)	452
39.6.6	Reef	452
39.6.7	Disco	453
39.6.8	Hama	453
39.6.9	Giraph	453
39.6.10	Pregel	453
39.6.11	Pegasus	453
39.6.12	Ligra	454
39.6.13	GraphChi	454
39.6.14	Galois	454
39.6.15	Medusa-GPU	454
39.6.16	MapGraph	455
39.6.17	Totem	455
39.7	Inter process communication Collectives	455
39.7.1	point-to-point	455
39.7.2	(a) publish-subscribe: MPI	455
39.7.3	(b) publish-subscribe: Big Data	455
39.7.4	HPX-5	455
39.7.5	Argo BEAST HPX-5 BEAST PULSAR	456
39.7.6	Harp	456
39.7.7	Netty	456
39.7.8	ZeroMQ	456
39.7.9	ActiveMQ	456
39.7.10	RabbitMQ	457
39.7.11	NaradaBrokering	457
39.7.12	QPid	458
39.7.13	Kafka	458
39.7.14	Kestrel	458
39.7.15	JMS	458
39.7.16	AMQP	459
39.7.17	Stomp	459
39.7.18	MQTT	459
39.7.19	Marionette Collective	459
39.7.20	Public Cloud: Amazon SNS	460
39.7.21	Lambda	460
39.7.22	Google Pub Sub	460
39.7.23	Azure Queues	461
39.7.24	Event Hubs	461

39.8	In-memory databases/caches	461
39.8.1	Gora (general object from NoSQL)	461
39.8.2	Memcached	461
39.8.3	Redis	462
39.8.4	LMDB (key value)	462
39.8.5	Hazelcast	462
39.8.6	Ehcache	463
39.8.7	Infinispan	463
39.8.8	VoltDB	463
39.8.9	H-Store	463
39.9	Object-relational mapping	464
39.9.1	Hibernate	464
39.9.2	OpenJPA	464
39.9.3	EclipseLink	464
39.9.4	DataNucleus	465
39.9.5	ODBC/JDBC	465
39.10	Extraction Tools	465
39.10.1	UIMA	465
39.10.2	Tika	466
39.11	SQL and SQL Services	466
39.11.1	Oracle	466
39.11.2	DB2	466
39.11.3	SQL Server	466
39.11.4	SQLite	467
39.11.5	MySQL	467
39.11.6	PostgreSQL	467
39.11.7	CUBRID	468
39.11.8	Galera Cluster	468
39.11.9	SciDB	468
39.11.10	Rasdaman	468
39.11.11	Apache Derby	469
39.11.12	Pivotal Greenplum	469
39.11.13	Google Cloud SQL	469
39.11.14	Azure SQL	469
39.11.15	Amazon RDS	469
39.11.16	Google F1	470
39.11.17	IBM dashDB	470
39.11.18	N1QL	470
39.11.19	BlinkDB	470
39.11.20	Spark SQL	470
39.12	NoSQL	471
39.12.1	Lucene	471
39.12.2	Solr	471
39.12.3	Solandra	471
39.12.4	Voldemort	471
39.12.5	Riak	472
39.12.6	ZHT	472
39.12.7	Berkeley DB	472
39.12.8	Kyoto/Tokyo Cabinet	472
39.12.9	Tycoon	473

39.12.10	Tyrant	473
39.12.11	MongoDB	473
39.12.12	Espresso	473
39.12.13	CouchDB	474
39.12.14	Couchbase Server	474
39.12.15	IBM Cloudant	475
39.12.16	Pivotal Gemfire (6)	475
39.12.17	HBase	475
39.12.18	Google Bigtable	475
39.12.19	LevelDB	476
39.12.20	Megastore and Spanner	476
39.12.21	Accumulo	476
39.12.22	Cassandra	477
39.12.23	RYA	477
39.12.24	Sqrrl	477
39.12.25	Neo4J	477
39.12.26	graphdb	477
39.12.27	Yarcdata	478
39.12.28	AllegroGraph	478
39.12.29	Blazegraph	478
39.12.30	Facebook Tao	479
39.12.31	Titan:db	479
39.12.32	Jena	479
39.12.33	Sesame	479
39.12.34	Public Cloud: Azure Table	480
39.12.35	Amazon Dynamo	480
39.12.36	Google DataStore	480
39.13	File management	481
39.13.1	iRODS	481
39.13.2	NetCDF	481
39.13.3	CDF	481
39.13.4	HDF	482
39.13.5	OPeNDAP	482
39.13.6	FITS	482
39.13.7	RCFile	482
39.13.8	ORC	482
39.13.9	Parquet	483
39.14	Data Transport	483
39.14.1	BitTorrent	483
39.14.2	HTTP	483
39.14.3	FTP	483
39.14.4	SSH	483
39.14.5	Globus Online (GridFTP)	484
39.14.6	Flume	484
39.14.7	Sqoop	484
39.14.8	Pivotal GLOAD/GPFDIST	484
39.15	Cluster Resource Management	485
39.15.1	Mesos	485
39.15.2	Yarn	485
39.15.3	Helix	485

39.15.4	Llama	486
39.15.5	Google Omega	486
39.15.6	Facebook Corona	486
39.15.7	Celery	486
39.15.8	HTCondor	487
39.15.9	SGE	487
39.15.10	OpenPBS	487
39.15.11	Moab	488
39.15.12	Slurm (553)	488
39.15.13	Torque	488
39.15.14	Globus Tools	488
39.15.15	Pilot Jobs	488
39.16	File systems	489
39.16.1	HDFS	489
39.16.2	Swift	489
39.16.3	Haystack	489
39.16.4	f4	489
39.16.5	Cinder	489
39.16.6	Ceph	490
39.16.7	FUSE	490
39.16.8	Gluster	491
39.16.9	Lustre	491
39.16.10	IBM Spectrum Scale, formerly GPFS	491
39.16.11	GFFS	491
39.16.12	Public Cloud: Amazon S3	491
39.16.13	Azure Blob	492
39.16.14	Google Cloud Storage	492
39.17	Interoperability	493
39.17.1	Cloudmesh	493
39.17.2	Libvirt	493
39.17.3	Libcloud	493
39.17.4	JClouds	493
39.17.5	TOSCA	494
39.17.6	OCCI	494
39.17.7	CDMI	494
39.17.8	Whirr	495
39.17.9	Saga	495
39.17.10	Genesis	496
39.18	DevOps	496
39.18.1	Docker (Machine, Swarm)	496
39.18.2	Puppet	496
39.18.3	Chef	496
39.18.4	Ansible	497
39.18.5	SaltStack	497
39.18.6	Boto	497
39.18.7	Cobbler	497
39.18.8	Xcat	498
39.18.9	Razor	498
39.18.10	Juju	498
39.18.11	Foreman	498

39.18.12	OpenStack Heat	498
39.18.13	Sahara	499
39.18.14	Rocks	499
39.18.15	Cisco Intelligent Automation for Cloud	499
39.18.16	Ubuntu Maas	499
39.18.17	Facebook Tupperware	499
39.18.18	AWS OpsWorks	500
39.18.19	OpenStack Ironic	500
39.18.20	Google Kubernetes	500
39.18.21	Buildstep	500
39.18.22	Gitreceive	501
39.18.23	OpenTOSCA	501
39.18.24	Winery	501
39.18.25	CloudML	501
39.18.26	Blueprints	502
39.18.27	Terraform	502
39.18.28	DevOpSlang	502
39.18.29	Any2Api	503
39.19	IaaS Management from HPC to hypervisors	503
39.19.1	Xen	503
39.19.2	KVM	503
39.19.3	QEMU	504
39.19.4	Hyper-V	504
39.19.5	VirtualBox	504
39.19.6	OpenVZ	504
39.19.7	LXC	505
39.19.8	Linux-Vserver	505
39.19.9	OpenStack	505
39.19.10	OpenNebula	506
39.19.11	Eucalyptus	506
39.19.12	Nimbus	506
39.19.13	CloudStack	506
39.19.14	CoreOS	507
39.19.15	rkt	507
39.19.16	VMware ESXi	507
39.19.17	vSphere and vCloud	507
39.19.18	Amazon	508
39.19.19	Azure	508
39.19.20	Google and other public Clouds	508
39.19.21	Networking: Google Cloud DNS	508
39.19.22	Amazon Route 53	509
39.20	Cross-Cutting Functions	509
39.20.1	Monitoring	509
39.20.2	Security and Privacy	510
39.20.3	Distributed Coordination	512
39.21	Message and Data Protocols	513
39.21.1	MQTT	513
39.21.2	Avro	513
39.21.3	Thrift	513
39.21.4	Protobuf	514

39.22	Technologies To Be Integrated	514
39.22.1	Snort	514
39.22.2	Fiddler	514
39.22.3	Zeppelin	514
39.22.4	Open MPI	515
39.22.5	Apache Tomcat	515
39.22.6	Apache Beam	515
39.22.7	Cloudability	515
39.22.8	CUDA	515
39.22.9	Blaze	516
39.22.10	CDAP	516
39.22.11	Apache Arrow	516
39.22.12	OpenRefine	516
39.22.13	Apache OODT	517
39.22.14	Omid	517
39.22.15	Apache Ant	517
39.22.16	LXD	518
39.22.17	Wink	518
39.22.18	Apache Apex	518
39.22.19	Apache Knox	518
39.22.20	Apache Apex	519
39.22.21	Robot Operating System (ROS)	519
39.22.22	Apache Flex	519
39.22.23	Apache Ranger	520
39.22.24	Google Cloud Machine Learning	520
39.22.25	Karajan	520
39.23	Exercise	521

XVII TBD

XVIII Container

40	REST	527
40.1	Swagger	528
40.1.1	Tools	528
40.2	FlaskRESTful	528
40.3	Django REST Framework	529
40.4	Eve	529
40.4.1	Creating your own objects	530
40.4.2	Towards cmd5 extensions to manage eve and mongo	531
41	Container	533
41.1	Kubernetes	533

42	Run Docker Locally on your Machine	535
42.1	Installing Docker Community Edition	535
42.1.1	Installation for OSX	535
42.2	Testing if the install works	536
43	Running Docker on FutureSystems	539
43.1	Overview	539
43.2	Getting Access	539
43.3	Creating a service and deploy to the swarm cluster	539
43.3.1	Create your own service	540
	Bibliography	543
	Index	583

Todo List

TODO: An example todo	38
TODO: TA: set up doodle for meetings. Must be completed in first week by next week Monday.	64
TODO: TA: set up initial times. mark they are temporary.	64
TODO: include link to Results section	65
TODO: Integrate the format from the class web page into the LaTeX section More details about the format can be found at https://cloudmesh.github.io/classes/lesson/doc/report.html	66
TODO: describe ssh config	159
TODO: describe port forwarding	159
TODO: provide us with screenshots of the windows.	160
TODO: lessons-ssh-generate-key	168
TODO: This section has to be redone as we use class specific clones and not the master	172
TODO: TA: some of the python examples assume REPL, but its better to use a print statement instead as more general, please fix	239
TODO: contribute	251
TODO: contribute	251
TODO: contribute	251
TODO: Add mor conventional database access	256
TODO: image missing	309
TODO: citation	310
TODO: Information after after 4:56 is outdated.	327
TODO: HiggsClassIII.py: /files/python/physics/calculated_dice_roll/higgs_classIV_seeds.py .	383
TODO: integrate physics-references.bib	384
TODO: on box but not available, move to google drive	409
TODO: slides are on box and not google drive Aspects of Big Data Applications,Slides https://iu.box.com/s/atgkxucop1lzftkunf8a12fe74x65na6	410
TODO: slides are on box and not google drive Big Data Applications and Generalizing their Structure,Slides https://iu.box.com/s/01dndtucmynekgehur00vktdfgcpgc1r	410

Preface

1	Introduction	33
1.1	Citation	
1.2	Authors	
1.3	Acknowledgements	
1.4	About	
1.5	Other Results	
1.6	Contributing	
1.7	Conventions	
1.8	Exercises	



1. Introduction

This document is the first one in a series of documents that are providing a comprehensive overview of Clouds, Big Data, and their technologies and applications. Material from these volumes are used selectively as part of this class.

Indiana University

They are developed within the Intelligent Systems Engineering department at Indiana University. This courses are for students who are interested in any component of the Masters or Ph.D. in Intelligent Systems Engineering. It is an advanced elective class.

The Series includes:

1. (This document) **Handbook of Clouds and Big Data**, Gregor von Laszewski, Geoffrey C. Fox, and Judy Qiu, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
2. **Use Cases in Big Data Software and Analytics Vol. 1**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
3. **Use Cases in Big Data Software and Analytics Vol. 2**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v2.pdf>
4. **Use Cases in Big Data Software and Analytics Vol. 3**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v3.pdf>
5. (Draft) **Big Data Software Vol 1.**, Gregor von Laszewski, Spring 2017, <https://github.com/cloudmesh/sp17-i524/blob/master/paper1/proceedings.pdf>
6. (Draft) **Big Data Software Vol 2.**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/paper2/proceedings.pdf>
7. (Draft) **Big Data Projects**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/project/projects.pdf>

Our plan for this semester also includes the reorganization of some of the volumes while sorting the contributions into chapters with papers from the same topic.

1.1 Citation

The citation for this document is

Gregor von Laszewski, Geoffrey C. Fox, and Judy Qiu, Handbook of Clouds and Big Data – Theory and Practice, Indiana University, Jan., 2018 Smith Research Center, Bloomington, IN 47408 <https://tinyurl.com/cloudmesh/vonLaszewski-bigdata.pdf>

The bibtex entry for this document is

```
@TechReport{las17handbook,
  author = {Gregor von Laszewski and Geoffrey C. Fox and Judy Qiu},
  title = {Handbook of Clouds and Big Data -- Theory and Practice},
  institution = {Indiana University},
  year = {2018},
  OPTtype = {Draft},
  address = {Smith Research Center, Bloomington, IN 47408},
  month = jan,
  url={https://tinyurl.com/cloudmesh/vonLaszewski-bigdata.pdf}
}
```

1.2 Authors



part/preface.tex

Gregor von Laszewski Gregor von Laszewski is an Assistant Director DSC in the School of Informatics and Computing and Engineering at Indiana University. He holds also a position as Adjunct Professor in the Intelligent Systems Engineering Department. Previously he held Adjunct Professor positions at the Computer Science Department at Indiana University and University of North Texas. He received a Ph.D. from Syracuse University in computer science.

At IU He served as the architect of the FutureGrid project. His current interest and projects include cloud computing, big data, and scientific impact metrics, and edge computing. He initiated the Cloudmesh project which is a toolkit to enable cloud computing across various Cloud and Container IaaS such as OpenStack, AWS, Azure, docker, docker swarm, and kubernetes.

Geoffrey C. Fox Fox received a Ph.D. in Theoretical Physics from Cambridge University and is now distinguished professor of Informatics and Computing, and Physics at Indiana University where he is director of the Digital Science Center, Chair of Department of Intelligent Systems Engineering and Director of the Data Science program at the School of Informatics, Computing, and Engineering.

He currently works in applying computer science from infrastructure to analytics in Biology, Pathology, Sensor Clouds, Earthquake and Ice-sheet Science, Image processing, Deep Learning, Manufacturing, Network Science and Particle Physics. The infrastructure work is built around Software Defined Systems on Clouds and Clusters. The analytics focuses on scalable parallelism.

Dr. Judy Qiu is an Associate Professor in the School of Informatics and Computing at Indiana University. Her research interests focus on data-intensive computing at the intersection of cloud and multicore technologies, with an emphasis on life science applications using MapReduce as well as traditional parallel and distributed computing approaches. Her contributions are focused on Hadoop and Introduction into Cloud Computing.

1.3 Acknowledgements

Chapter 18 focussing on Chameleon Cloud was copied with permission from the chameleon-cloud.org website on Jan 1st, 2018 and represents the contribution of the Chameleon project team, developed under the NSF grant 1743358, Collaborative Research: Chameleon: A Large-Scale, Reconfigurable Experimental Environment for Cloud Research. Some content has been modified and added by Gregor von Laszewski to specifically target classes and education material targeted for Indiana University.

1.4 About

The material in this document is covering material that is or has been used in the following classes at Indiana University.

- Undergraduate: E222 Intelligent systems II
- Graduate: E516 Introduction to Cloud Computing
- Graduate: E616 Advanced Cloud Computing
- Graduate: E534 Big Data Applications
- Graduate: I524 Big Data Applications and Open Source Software
- Graduate: I523 Big Data Applications and Analytics

The format has been influenced by experiences gained from teaching I523 and I524 by Gregor von Laszewski. The collection of this material is updated continuously and new versions will be made available throughout the semester. A timestamp on the front page indicates the last time the document was updated.

1.5 Other Results

Besides the documents posted in Section 1 a small set of additional results exist in form of a video presentation. However, these results are older and we are no longer using such presentations. To guarantee completeness, we include them however in this section

- | | |
|---|--|
| Student Work 1 (8:48) | |
| Student Work 1 (Page 7) | |
| Student Work 1 - pptx (Page 7) | |
| Student Work 2 (10:03) | |
| Student Work 2 (Page 12) | |
| Student Work 2 - pptx (Page 12) | |

1.6 Contributing

We encourage students of the classes to contribute to this document or other volumes, provide corrections, and additions. This document is managed on [github.com](https://github.com/cloudmesh/book) at

- <https://github.com/cloudmesh/book>

1.6.1 Class Contributions

The current release version is held in the master branch. Development versions will be held under a number of branches:

- e222** Branch with contributions from students of the e222 class. Merges to and from the *latex* branch will be conducted on a daily bases by TA's.
- e516** Branch with contributions from students of the e616 class. Merges to and from the *latex* branch will be conducted on a daily bases by TA's.
- e616** Branch with contributions from students of the e616 class. Merges to and from the *latex* branch will be conducted on a daily bases by TA's.
- dev** Branch managed by Gregor and the TA's
- master** Branch that contains the current released version. This version is updated once a week from the branch *latex*.

Contributions are to be conducted as pull requests. It is important to keep the pull requests small and on a section or even paragraph base. This helps avoiding conflicts at time of checkin and is a common practice in large communities. It is not a good practice to work for weeks on improvements and than issue the pull request. For sure things will have changed and it will take you a long time to catch up.

The original document is based on selected material published and edited by Gregor von Laszewski at the following Web page

- <https://cloudmesh.github.io/classes/>

Based on feedback from students the desire to have the material in book format available was raised and we have implemented it.

1.6.2 Community Contributions

It is easy to contribute to this document and we invite everyone to improve the material. To do so you need to fork the repository from

- <https://github.com/cloudmesh/book/>

and clone it. Make sure you check out the *dev* branch. Then you can modify information in the different files or add new sections. It is important that you make changes based on sections and than for them create a new pull request. This simplifies the review process. We will typically want only one file to be changed. Aslo before you issue your pull request make sure that no one else has already made changes. In that case we ask you to integrate them into your document.

1.6.3 Contributors

We like to acknowledge the following contributors that helped on this document. Please notify us with your name and a brief commend on what you contributed:

Descriptions provided in Section 39 were contributed by the following people that are either listed by full name or their github.com id:

Abhijit Thakre, Abhishek Gupta, Abhishek Naik, Ajit Balaga, Anurag Kumar Jain, Avadhoot Agasti, Badi' Abdul-Wahid, Cmbays, DIKSHA, Dimitar Nikolov, Govind, Govind Mishra, Grace Li, Gregor von Laszewski, Harshit Krishnakumar, Hyungro Lee, Jerome Mitchell, Jimmy Ardiansyah, Jon, Jon Montgomery, Jordan Simmons, Juliette Zerick, Karthik, Kumar Satyam, Mark McCombe, Matthew Lawson, Methku-

palli Vasantha, Miao Jiang, Miao Zhang, Milind Suryawanshi, MilindSuryawanshi, Nandita Sathe, Naveen, Niteesh01, Piyush Rai, Piyush Shinde, Prashanth, Pratik Jain, Rahul Raghatare, Rahul Singh, Ribka Rufael, Ronak Parekh, Saber Sheybani, Sabyasachi Roy Choudhury, Sagar Vora, Sahiti Korrapati, Scott McClary, Sean Shiverick, SilviaKarim14, Sivaprasad Sushmita, Snehal Chemburkar, Sowmya Ravi, Srikanth Ramanam, Sunanda Unni, SushmitaSivaprasad, Tony Liu, Vasantha Methkupalli, Veera Marni, Vibhatha Abeykoon, Vibhatha Lakmal Abeykoon, Vishwanath Kodre, William H Knapp III, acastrob, ak.15, alyez, anveling, argetlam115, athakre, bhavesh37, coultes, cglmoocs, elenadesigner, eunosm3, harkrish1, jemitchell, justbbusy, jzerick, kartanba, karthick, karthick venkatesan, karthik-anba, kpvenkat, ksrivatsav, lmundia, miaozhan, michaelsmith1983, mmcccombe, nsathe, piyurai, pratik11jain, ronak1182, sabyasachi087, shah0112, sriramsitharaman, suunni, tifabi, tonythomascn, vasantha, vibhatha, vkodre, vlabecko, xl41, yatinsharma7

1.7 Conventions

1.7.1 Videos

Videos to the class are referred to with embedded links into the PDF document as follows:

[Test Video \(25:36\)](#) 

An index will also be available in the index page that lists on which page the video has been added.

1.7.2 Slides

Sides

[Test slides \(10\)](#) 

1.7.3 Images

The video icon was copied from <http://www.freeiconspng.com/img/8039>.

1.7.4 URLs

The online version of this document contains a significant number of links. The links are either embedded or are directly visible. The color of the links is blue.

Direct URL: This is an example for a <https://github.com/cloudmesh/book/>

Embedded URL: This is an example for an embedded URL that points to the [source on github](#)

1.7.5 Boxes

Note

Notes are written in blue boxes and indicate a clarification or some important information that you do not want to overlook.

Warning

Warnings are written in red boxes and indicate a piece of information that you must not ignore.

Indiana University

Red boxes with Indiana University are information that relate to students that use this material as part of the courses offered at Indiana University.

To dos are highlighted in boxes with the keyword TODO. They offer opportunities for student to gather points for the discussion grade.

TODO: An example todo

1.8 Exercises

Exercise 1.1 Inspect the PDF documents produced by previous classes. Note the differences between technology and application reviews and projects. ■

Exercise 1.2 Contribute to this document while finding a single spelling error. Before you do the pull request, make sure the document compiles. ■

Syllabus

2 E516: Introduction to Cloud Computing 41

- 2.1 Course Description
- 2.2 Course pre-requisites
- 2.3 Registration Information
- 2.4 Teaching and learning methods
- 2.5 Covered Topics
- 2.6 Student learning outcomes
- 2.7 Grading
- 2.8 Representative bibliography
- 2.9 Lectures and Lecture Material

3 E616/I524: Advanced Cloud Computing

47

- 3.1 Course Description
- 3.2 Course pre-requisites
- 3.3 Course Registration
- 3.4 Teaching and learning methods
- 3.5 Covered Topics
- 3.6 Student learning outcomes
- 3.7 Grading
- 3.8 Representative bibliography
- 3.9 Lectures and Lecture Material
- 3.10 Containers

4 E222: Intelligent Systems Engineering II 53

- 4.1 Course Description
- 4.2 Course pre-requisites
- 4.3 Course Registration
- 4.4 Teaching and learning methods
- 4.5 Covered Topics
- 4.6 Student learning outcomes
- 4.7 Grading
- 4.8 Representative bibliography
- 4.9 Lectures and Lecture Material
- 4.10 Containers



2. E516: Introduction to Cloud Computing

We present the syllabus for the introduction to Cloud Computing course taught at Indiana University that uses in part the material presented in this document. The information includes the section or chapter number, the name of the chapter or section, and the timeframe when it is recommended to work through the material and the page number where to find the material in this document. Please note that we will update the document throughout the semester thus, pagenumbers will change.

2.1 Course Description

This course describes the emerging cloud and big data technologies within the world of distributed intelligent systems where each system component has internal, and external sources of intelligence that are subject to collective control. Examples are given from a wide variety of applications. A project will allow students to dive into practical issues after they have obtained a theoretical background.

2.2 Course pre-requisites

Python will be used as a Programming Languages. In some cases Java may also be useful however it's use in class will be marginal. The class will be using Linux commandline tools. Prior knowledge of Linux is of advantage but not required. Studnets are expected to have access to a computer on which they can execute Linux easily. As the OS requirements have recently increased we recommend a computer with 8GB main memory and the ability to run virtualbox and/or containers. If it turns out your machine is not capable enough we attempt to provide access to IU linux machines.

2.3 Registration Information

ENGR-E 516 ENGINEERING CLOUD COMPUTING (3 CR)
33699 RSTR 11:15A-12:30P F I2 150 Von Laszewski G
Above class taught online
Above class open to graduates only
Discussion (DIS)

ENGR-E 516 ENGINEERING CLOUD COMPUTING (3 CR)
33909 RSTR ARR ARR WB WEB Von Laszewski G
This is a 100% online class taught by IU Bloomington. No
on-campus class meetings are required. A distance education
fee may apply; check your campus bursar website for more
information
Above class open to graduates only

2.4 Teaching and learning methods

- Lectures
- Assignments including specific lab activities
- Final project

2.5 Covered Topics

The class will cover a number of topics as part of tracks that are executed in parallel throughout the class. Although we assume that at a graduate course level the communication track has already been covered elsewhere, we made sure we also include it here in case you did not yet have such experiences.

- **Introduction:** Overview and status of cloud computing
- **Cloud Computing:** Basics of Cloud Computing
- **Virtual Machines:** Introduction to OpenStack
- **Cloud 3.0:** Microservices, Events, Functions
- **Hadoop:** Introduction to Hadoop
- **Project:** Delivery of a reproducible substantial student project

2.6 Student learning outcomes

When students complete this course, they should be able to:

- Have an elementary understanding of issues involved in designing and Software Defined Systems.
- Understand the concepts of Cloud Computing
- Gain hands-on laboratory experience with several examples.
- Apply knowledge of mathematics, science, and engineering
- Have advanced skills in teamwork with peers.

2.7 Grading

Grade Item	Percentage
Assignments	40%
Final Project	50%
Participation	10%

2.8 Representative bibliography

1. Cloud Computing for Science and Engineering By Ian Foster and Dennis B. Gannon
 - <https://mitpress.mit.edu/books/cloud-computing-science-and-engineering>
2. Ansible Tutorials
3. <http://bigdataopensourceprojects.soic.indiana.edu/>
4. The backdrop for course is the 350 software subsystems illustrated at <http://hpc-abds.org/kaleidoscope/>
5. http://cloudmesh.github.io/introduction_to_cloud_computing/
6. There are a huge number of web resources
7. (This document) **Handbook of Clouds and Big Data**, Gregor von Laszewski, Geoffrey C. Fox, and Judy Qiu, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
8. **Use Cases in Big Data Software and Analytics Vol. 1**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
9. **Use Cases in Big Data Software and Analytics Vol. 2**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v2.pdf>
10. **Use Cases in Big Data Software and Analytics Vol. 3**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v3.pdf>
11. (Draft) **Big Data Software Vol 1.**, Gregor von Laszewski, Spring 2017, <https://github.com/cloudmesh/sp17-i524/blob/master/paper1/proceedings.pdf>
12. (Draft) **Big Data Software Vol 2.**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/paper2/proceedings.pdf>
13. (Draft) **Big Data Projects**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/project/projects.pdf>

2.9 Lectures and Lecture Material

Warning

This section will change and be updated throughout the semester

2.9.1 Communication Track

<input type="checkbox"/>	<input checked="" type="checkbox"/> 7. Documenting Scientific Research	Month 1	81
<input type="checkbox"/>	<input checked="" type="checkbox"/> 7.2. Plagiarism	Week 1	82
<input type="checkbox"/>	<input checked="" type="checkbox"/> 1.5. Other Results	Week 1	35
<input type="checkbox"/>	<input checked="" type="checkbox"/> 11.2. Markdown	Week 1	134
<input type="checkbox"/>	<input checked="" type="checkbox"/> 10.1. Basic Emacs	Week 2	127
<input type="checkbox"/>	<input checked="" type="checkbox"/> 7.4. Writing a Scientific Article or Conference Paper	Week 3	85

<input type="checkbox"/> ✓ 8. Introduction to L ^A T _E X	Week 3 93
<input type="checkbox"/> ✓ 9. Managing Bibliographies	Week 4 111

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarized you will receive an ‘‘F’’ and it is reported based on University policies.

2.9.2 Theory Track

- IaaS - OpenStack
- PaaS - Hadoop
- SaaS - SaaS with REST

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarized you will receive an ‘‘F’’ and it is reported based on University policies. The paper is in a directory called paper1. All images are in the directory paper1/images, the report is in report.tex, the content is in content.tex. It follows the template we provided. Submission of report.pdf is not allowed. We will create the report and check completeness that way.

2.9.3 Programming Track

Development Environment

<input type="checkbox"/> X 13. Linux	Week 2 151
<input type="checkbox"/> X 14. SSH	Week 2 157
<input type="checkbox"/> X 15. Github	Week 3 167
<input type="checkbox"/> X 16. Virtual Box	Week 3 175

Evaluation Experiment 1: Create a virtual machine and take a photo with your laptop or computer and the virtual box running on the screen. Showcase the virtual box interface and in non full screen mode at the same time the operating system you run. We recommend yo use Ubuntu.

Python

<input type="checkbox"/> ✓ 19. Introduction	Week 3 - 4 227
<input type="checkbox"/> ✓ 20. Install	Week ? 231
<input type="checkbox"/> ✓ 21. Language	Week ? 239
<input type="checkbox"/> ✓ 23. Libraries	Week ? 257
<input type="checkbox"/> X 24. Cloudmesh Command Shell	Week ? 271

Cloud

Assignments

Develop a Big Data related REST service with swagger.

Chose two of them so you can see the difference between the APIs:

- Build a Multi cloud interface to OpenStack
- Build a Multi cloud interface to AWS
- Build a Multi cloud interface to Azure

- Build a Multi cloud interface to Docker/Kubernetes

Evaluation Experiment 1: Create a program in python that identifies a termination criteria for the Secchi disk problem. E.g. at what image can we no longer see the disk? Describe your solution in md and submit to the git repository in a directory called *secchi*. The program is called *secchi.py*, the description is in *README.md*. It uses cmd5 for creating a command shell that can load the data and analyze it.

Evaluation Assignment 1: Create two a REST service for Big Data with swagger and provide a python implementation from the exported python code to provide real functionality.

Evaluation Assignment 2: Develop cloudmesh extensions to access cloud services from 2 different providers.

Chameleon Cloud and OpenStack

<input type="checkbox"/> ✓ 18. Chameleon Cloud	Week 4	183
<input type="checkbox"/> ✓ 18.5. Charge Rates	Week 4	189
<input type="checkbox"/> ✓ 18.3. Hardware	Week 4	185
<input type="checkbox"/> ✓ 18.4. Getting Started	Week 4	187
<input type="checkbox"/> ✓ 18.7. Horizon Graphical User Interface	Week 5	197
<input type="checkbox"/> ✓ 18.6. OpenStack Virtual Machines	Week 4	190
<input type="checkbox"/> X 18.8. HEAT	Optional	204
<input type="checkbox"/> X 18.9. Bare Metal	Optional	220



3. E616/I524: Advanced Cloud Computing

This class is also known under the name *I524 Big Data Applications and Open Source Software*.

We present the syllabus for the E616 course taught at Indiana University that uses in part the material presented in this document. The information includes the section or chapter number, the name of the chapter or section, and the timeframe when it is recommended to work through the material and the page number where to find the material in this document. Please note that we will update the document throughout the semester thus, page numbers will change.

3.1 Course Description

This course describes Cloud 3.0 in which DevOps, Microservices, and Function as a Service is added to basic cloud computing. The discussion is centered around the Apache Big Data Stack and a major student project aimed at demonstrating integration of cloud capabilities.

3.2 Course pre-requisites

Python will be used as a Programming Languages. It is expected that you know a programming language. ENGR-E516 or an introduction to cloud computing is recommended. Students are expected to have access to a computer on which they can execute Linux easily. As the OS requirements have recently increased we recommend a computer with 8GB main memory and the ability to run virtualbox and/or containers. If it turns out your machine is not capable enough we attempt to provide access to IU linux machines.

3.3 Course Registration

I524 and E616 are identical. In this courses we will be focussing on Advanced Cloud Computing and Big Data Applications and Analytics.

Intelligent Systems Engineering Residential:

ENGR-E 616 ADVANCED CLOUD COMPUTING (3 CR)					
***** RSTR ARR	ARR	ARR	Von Laszewski G		
Above class taught online					
Above class open to graduates only					
Discussion (DIS)					
33697 RSTR 09:30A-10:45A F	I2 150	Von Laszewski G			

Info Residential:

INFO-I 524 BIG DATA SOFTWARE AND PROJECTS (3 CR)					
***** RSTR ARR	ARR	ARR	Von Laszewski G		
Above class open to graduates only					
Above class taught online					
Discussion (DIS)					
13053 RSTR 09:30A-10:45A F	I2 150	Von Laszewski G			

Info Online:

INFO-I 524 BIG DATA SOFTWARE AND PROJECTS (3 CR)					
13054 RSTR ARR	ARR	ARR	Von Laszewski G		
Above class open to graduates only					
This is a 100% online class taught by IU Bloomington. No on-campus class meetings are required. A distance education fee may apply; check your campus bursar website for more information					

3.4 Teaching and learning methods

- Lectures
- Assignments including specific lab activities
- Final project
- Class will use software mainly written in Python and Linux Shell.

3.5 Covered Topics

The class will cover a number of topics as part of tracks that are executed in parallel throughout the class. Although we assume that at a graduate course level the communication track has already been covered elsewhere, we made sure we also include it here in case you did not have such experiences.

- **Introduction** Overview and status of cloud computing
- **Cloud and Big Data Applications** Introduction to the Apache Big Data Stack. Selective presentation of the members of the Big Data set
- **New Cloud Big Data Application Technologies** Students will explore the Apache Web page and report on them.
- **DevOps** Introduction to DevOps with Dockerfile and ansible
- **Virtual Machines** Introduction to OpenStack
- **Containers** Introduction to Container technology
- **Advanced Containers** Building clusters with containers.
- **Cloud 3.0** Microservices, Events, Functions
- **Project** Delivery of a reproducible substantial student project (you are allowed to substantially enhance a project that you started in E516)

3.6 Student learning outcomes

When students complete this course, they should be able to:

- Have an advanced understanding of issues involved in designing and applying modern cloud technologies using the latest developments.
- Gain hands-on laboratory experience.
- Understand the Apache Big Data Software Stack.
- Apply knowledge of mathematics, science, and engineering.
- Understand research challenges and important application areas of clouds
- Have advanced skills in teamwork with peers.
- Be able to use DevOps technologies.

3.7 Grading

Grade Item	Percentage
Assignments	40%
Final Project	50%
Participation	10%

3.8 Representative bibliography

1. Cloud Computing for Science and Engineering By Ian Foster and Dennis B. Gannon
 - <https://mitpress.mit.edu/books/cloud-computing-science-and-engineering>
2. Machine to machine protocols <https://en.wikipedia.org/wiki/MQTT>
3. Cloud software systems <http://hpc-abds.org/kaleidoscope/>
4. Software Defined Networks https://en.wikipedia.org/wiki/Software-defined_networking
5. There are a huge number of other web resources
6. (This document) **Handbook of Clouds and Big Data**, Gregor von Laszewski, Geoffrey C. Fox, and Judy Qiu, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
7. **Use Cases in Big Data Software and Analytics Vol. 1**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
8. **Use Cases in Big Data Software and Analytics Vol. 2**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v2.pdf>
9. **Use Cases in Big Data Software and Analytics Vol. 3**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v3.pdf>
10. (Draft) **Big Data Software Vol 1.**, Gregor von Laszewski, Spring 2017, <https://github.com/cloudmesh/sp17-i524/blob/master/paper1/proceedings.pdf>
11. (Draft) **Big Data Software Vol 2.**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/paper2/proceedings.pdf>
12. (Draft) **Big Data Projects**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/project/projects.pdf>

3.9 Lectures and Lecture Material

Warning

This section will change and be updated throughout the semester

3.9.1 Communication Track

<input type="checkbox"/> ✓ 7. Documenting Scientific Research	Month 1 81
<input type="checkbox"/> ✓ 7.2. Plagiarism	Week 1 82
<input type="checkbox"/> ✓ 1.5. Other Results	Week 1 35
<input type="checkbox"/> ✓ 11.2. Markdown	Week 1 134
<input type="checkbox"/> ✓ 10.1. Basic Emacs	Week 2 127
<input type="checkbox"/> ✓ 7.4. Writing a Scientific Article or Conference Paper	Week 3 85
<input type="checkbox"/> ✓ 8. Introduction to L ^A T _E X	Week 3 93
<input type="checkbox"/> ✓ 9. Managing Bibliographies	Week 4 111

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarised you will receive an “F” and it is reported based on University policies.

3.9.2 Theory Track

<input type="checkbox"/> X 39.1. Workflow-Orchestration	Week 1 413
<input type="checkbox"/> X 39.2. Application and Analytics	Week 2 422
<input type="checkbox"/> X 39.4. High level Programming	Week 3 440
<input type="checkbox"/> X 39.5. Streams	Week 4 447
<input type="checkbox"/> X 39.6. Basic Programming Model and Runtime, SPMD, MapReduce	Week 5 451
<input type="checkbox"/> X 39.7. Inter process communication Collectives	Week 5 455
<input type="checkbox"/> X 39.8. In-memory databases/caches	Week 6 461
<input type="checkbox"/> X 39.9. Object-relational mapping	Week 6 464
<input type="checkbox"/> X 39.10. Extraction Tools	Week 7 465
<input type="checkbox"/> X 39.11. SQL and SQL Services	Week 6 466
<input type="checkbox"/> X 39.12. NoSQL	Week 6 471
<input type="checkbox"/> X 39.13. File management	Week 8 481
<input type="checkbox"/> X 39.14. Data Transport	Week 8 483
<input type="checkbox"/> X 39.15. Cluster Resource Management	Week 9 485
<input type="checkbox"/> X 39.16. File systems	Week 8 489
<input type="checkbox"/> X 39.17. Interoperability	Week 3 493
<input type="checkbox"/> X 39.18. DevOps	Week 10 496
<input type="checkbox"/> X 39.19. IaaS Management from HPC to hypervisors	Week 11 503
<input type="checkbox"/> X 39.20. Cross-Cutting Functions	Week 12 509
<input type="checkbox"/> X 39.21. Message and Data Protocols	Week 13 513
<input type="checkbox"/> X 39.22. Technologies To Be Integrated	Week 4 514

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarized you will receive an “F” and it is reported based on University policies. The paper is in a directory called paper1. All images are in the directory paper1/images, the report is in report.tex, the content is in content.tex. It follows the template we provided. Submission of report.pdf is not allowed. We will create the report and check completeness that way.

3.9.3 Programming Track

Development Environment

<input type="checkbox"/> X 13. Linux	Week 2	151
<input type="checkbox"/> X 14. SSH	Week 2	157
<input type="checkbox"/> X 15. Github	Week 3	167
<input type="checkbox"/> X 16. Virtual Box	Week 3	175

Evaluation Experiment 1: Create a virtual machine and take a photo with your laptop or computer and the virtual box running on the screen. Showcase the virtual box interface and in non full screen mode at the same time the operating system you run. We recommend yo use Ubuntu.

Python

<input type="checkbox"/> ✓ 19. Introduction	Week 3 - 4	227
<input type="checkbox"/> ✓ 20. Install	Week ?	231
<input type="checkbox"/> ✓ 21. Language	Week ?	239
<input type="checkbox"/> ✓ 23. Libraries	Week ?	257
<input type="checkbox"/> X 24. Cloudmesh Command Shell	Week ?	271
<input type="checkbox"/> X 25. Numpy		277
<input type="checkbox"/> X 26. Scipy		281
<input type="checkbox"/> X 27. OpenCV		287
<input type="checkbox"/> X 27.5. Secchi Disk		291

Evaluation Experiment 1: Create a program in python that identifies a termination criteria for the Secchi disk problem. E.g. at what image can we no longer see the disk? Describe your solution in md and submit to the git repository in a directory called *secchi*. The program is called *secchi.py*, the description is in *README.md*. It uses cmd5 for creating a command shell that can load the data and analyze it.

3.9.4 DevOps

- ssh for DevOps
- Ansible
- Dockerfile

3.9.5 Cloud

- OpenAPI REST Service
- OpenAPI Big Data Services
- Deploy cloud services with Ansible

Chameleon Cloud and OpenStack

<input type="checkbox"/> ✓ 18. Chameleon Cloud	Week 4	183
<input type="checkbox"/> ✓ 18.5. Charge Rates	Week 4	189
<input type="checkbox"/> ✓ 18.3. Hardware	Week 4	185
<input type="checkbox"/> ✓ 18.4. Getting Started	Week 4	187

<input type="checkbox"/> ✓ 18.7. Horizon Graphical User Interface	Week 5	197
<input type="checkbox"/> ✓ 18.6. OpenStack Virtual Machines	Week 4	190
<input type="checkbox"/> ✗ 18.8. HEAT	Optional	204
<input type="checkbox"/> ✗ 18.9. Bare Metal	Optional	220

3.10 Containers

- Introduction to Docker - Docker File
- Advanced Docker - Docker Swarm - Kubernetes
- Deploy cloud services with Docker/Kubernetes
- Build a Raspberry Pi based Kubernetes cluster with 5 nodes (residential student can work on this in teams up to 5 students, we have 50 Raspberry Pi's)
- Benchmark a 144 node Raspberry PI kubernetes cluster after deploying it (open class work)



4. E222: Intelligent Systems Engineering II

4.1 Course Description

In this undergraduate course students will be familiarized with different specific applications and implementations of intelligent systems and their use in desktop and cloud solutions.

4.2 Course pre-requisites

One programming language, Intelligent Systems Engineering I or equivalent

4.3 Course Registration

ENGR-E 222 INTELLIGENT SYSTEMS II (3 CR)					
***** RSTR	02:30P-03:45P	TR		GY 436	Fox
Laboratory (LAB)					
E 222 : P - ENGR-E 221					
31434 RSTR	05:45P-06:35P	R		GY 447	Fox
Above class for Intelligent Systems Engineering students					

4.4 Teaching and learning methods

- Lectures
- Assignments including specific lab activities
- Final project

4.5 Covered Topics

The topics covered in this class include.

- Introduction to REST: Theory and Practice - develop a REST service
- Introduction to Clouds: Theory and Practice - create via a program virtual machines and start on them the REST service
- Introduction to Kubernetes: Theory and Practice - create a container that runs a REST service
- Introduction to Advanced AI: Integrate your AI engine into a REST service and run on a cloud and in Kubernetes
- Introduction to Hadoop: Theory and Practice - Run Hadoop in a container; run hadoop on a futuresystems cluster
- Edge Computing: Theory and Practice - Integrate Sensordata into Cloud Services via REST and MQTT

4.6 Student learning outcomes

When students complete this course, they should be able to:

- Have an elementary understanding of issues involved in Cloud Computing as part of the intelligent systems effort.
- Gain hands-on laboratory experience with several examples.
- Apply knowledge of mathematics, science, and engineering
- Understand research challenges and important issues with Software Defined Systems.
- Have advanced skills in teamwork with peers.
- Have theoretical and practical knowledge about REST, Clouds, Containers, and Edge Computing.

4.7 Grading

Grade Item	Percentage
Assignments	40%
Final Project	50%
Participation	10%

4.8 Representative bibliography

1. Cloud Computing for Science and Engineering By Ian Foster and Dennis B. Gannon
 - <https://mitpress.mit.edu/books/cloud-computing-science-and-engineering>
2. (This document) **Handbook of Clouds and Big Data**, Gregor von Laszewski, Geoffrey C. Fox, and Judy Qiu, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
3. **Use Cases in Big Data Software and Analytics Vol. 1**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v1.pdf>
4. **Use Cases in Big Data Software and Analytics Vol. 2**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v2.pdf>
5. **Use Cases in Big Data Software and Analytics Vol. 3**, Gregor von Laszewski, Fall 2017, <https://tinyurl.com/cloudmesh/vonLaszewski-i523-v3.pdf>
6. (Draft) **Big Data Software Vol 1.**, Gregor von Laszewski, Spring 2017, <https://github.com/cloudmesh/sp17-i524/blob/master/paper1/proceedings.pdf>
7. (Draft) **Big Data Software Vol 2.**, Gregor von Laszewski, Spring 2017,
 - <https://github.com/cloudmesh/sp17-i524/blob/master/paper2/proceedings.pdf>
8. (Draft) **Big Data Projects**, Gregor von Laszewski, Spring 2017,

- <https://github.com/cloudmesh/sp17-i524/blob/master/project/projects.pdf>

4.9 Lectures and Lecture Material

4.9.1 Communication Track

<input type="checkbox"/>	<input checked="" type="checkbox"/> 7. Documenting Scientific Research	Month 1 81
<input type="checkbox"/>	<input checked="" type="checkbox"/> 7.2. Plagiarism	Week 1 82
<input type="checkbox"/>	<input checked="" type="checkbox"/> 1.5. Other Results	Week 1 35
<input type="checkbox"/>	<input checked="" type="checkbox"/> 11.2. Markdown	Week 1 134
<input type="checkbox"/>	<input checked="" type="checkbox"/> 10.1. Basic Emacs	Week 2 127
<input type="checkbox"/>	<input checked="" type="checkbox"/> 7.4. Writing a Scientific Article or Conference Paper	Week 3 85
<input type="checkbox"/>	<input checked="" type="checkbox"/> 8. Introduction to L ^A T _E X	Week 3 93
<input type="checkbox"/>	<input checked="" type="checkbox"/> 9. Managing Bibliographies	Week 4 111

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarised you will receive an “F” and it is reported based on University policies.

4.9.2 Theory Track

- IaaS - OpenStack
- PaaS - Hadoop
- SaaS - SaaS with REST

Evaluation Paper1: Create a paper about a cloud technology with our give class template in the git repository. If a paper is plagiarized you will receive an “F” and it is reported based on University policies. The paper is in a directory called paper1. All images are in the directory paper1/images, the report is in report.tex, the content is in content.tex. It follows the template we provided. Submission of report.pdf is not allowed. We will create the report and check completeness that way.

4.9.3 Programming Track

Development Environment

<input type="checkbox"/>	<input checked="" type="checkbox"/> 13. Linux	Week 2 151
<input type="checkbox"/>	<input checked="" type="checkbox"/> 14. SSH	Week 2 157
<input type="checkbox"/>	<input checked="" type="checkbox"/> 15. Github	Week 3 167
<input type="checkbox"/>	<input checked="" type="checkbox"/> 16. Virtual Box	Week 3 175

Python

<input type="checkbox"/>	<input checked="" type="checkbox"/> 19. Introduction	Week 3 - 4 227
<input type="checkbox"/>	<input checked="" type="checkbox"/> 20. Install	Week ? 231
<input type="checkbox"/>	<input checked="" type="checkbox"/> 21. Language	Week ? 239

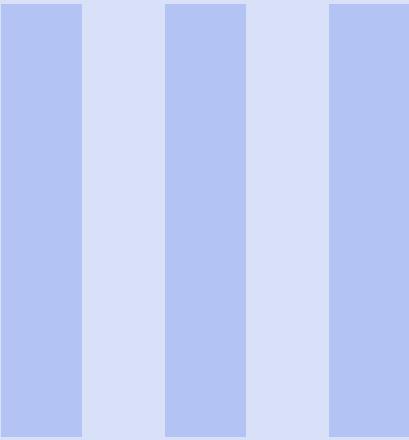
- | | |
|--|------------|
| <input type="checkbox"/> ✓ 23. Libraries | Week ? 257 |
| <input type="checkbox"/> X 24. Cloudmesh Command Shell | Week ? 271 |

Cloud

- Build a Rest Service
- Build a program to create VMs on an OpenStack cloud.

4.10 Containers

- Docker
- Docker File
- Doker Swarm
- Kubernetes



Class Policies and Communication

5	Course Policies	59
5.1	Communication and Use of CANVAS	
5.2	Online and Office Hours	
5.3	Class Material	
5.4	HID	
5.5	Notebook	
5.6	Blog	
5.7	Calendar I524, E616, E516	
5.8	Incomplete	
5.9	Waitlist	
5.10	Auditing the class	
5.11	Resource restrictions	
5.12	Office Hours	
5.13	Plagiarism	
5.14	Tutorials, Topic Paper, Term Paper, Project Report	
5.15	Grading	
6	Piazza	71
6.1	Access to Piazza from Canvas	
6.2	Verify you are on Piazza via a post	
6.3	Making Piazza Work	
6.4	Towards good questions	
6.5	Guide on how to ask good questions	
6.6	Piazza class Links	
6.7	Piazza Curation	
6.8	Read the Originals, not just the e-mail	
6.9	Exercises	



5. Course Policies

We describe briefly how we manage different classes and how to interact with us.

5.1 Communication and Use of CANVAS

In the past we have found numerous limitations to CANVAS that makes it not possible for us to use it effectively for our classes. Such limitations were not overcome even after consulting our professional course support team at IU although we spend days and weeks trying to fix the issues we encountered. From the professional staff we have been given the advice to avoid using CANVAS and use a better approach that we outline as follows:

- we use CANVAS notifications only for the initial class setup and notify you how to gain access to piazza where all class discussions take place;
- any discussion about grades is however only done in CANVAS;
- we will DELETE all e-mail send to us on personal or IU e-mail addresses. All discussions need to take place either in private or in public posts on piazza (other than grades which need to be send via CANVAS as the previous point explained);
- TA's are strictly forbidden to answer any e-mail that is not send via piazza. Their answers will only be posted within piazza;
- any post to piazza to a TA must be not done to the individual TA but to *instructors*;
- any post from piazza by a TA must be not done to the individual student only, but also must include *instructors*;

5.2 Online and Office Hours

To support you we have established an open policy of sharing information not only as part of the class material, but also as part of how we conduct support. We establish the following principals:

- in case of doubt how to communicate address this early in class and attend online hours;

- all office hours if not of personal nature are open office hours meaning that any student in class can be joined by other students of the class and all meeting times are posted publicly.
- it is in your responsibility to attend in person classes and online hours as we found that those that do get better grades. For residential students participation in the residential classes is mandatory due to the same reason.
- instructors of this class will attempt within reason to find suitable times for you to attend an online hour in case you are an online student. Residential students can attend the class on Friday and ask any question.

5.3 Class Material

Warning

As the class material will evolve during the semester it is obvious that some content will be improved and material will be added. This benefits all classes. To stay up to date, please, revisit this document on weekly basis. This is obvious, as we will adapt content based on your feedback.

The requirement of the classes to become an expert in cloud and/or Big Data applications and technologies stays unchanged.

5.4 HID

You will be assigned an hid (Homework IDentifier) which allows us to easily communicate with you and does allow us to not use your university ID to communicate with you.

You will receive the HID within the first week of the semester by the TA's.

5.5 Notebook

All students are required to maintain a *class notebook* in github in which they summarize their weekly activities for this course in bullet form. This includes a self maintained list of which lecture material they viewed.

The notebook is maintained in the class github.com in your hid project folder. It is a file called notebook.md that uses markdown as format. While using md, you can either edit it locally and upload to github, or directly edit it via the git hub Web editor. Notebooks are expected to be set up as soon as the git repository was created.

You will be responsible to set up and maintaining the notebook.md and update it accordingly. We suggest that you prepare sections such as: Logistic, Theory, Practice, Writing and put in bullet form what you have done into these sections during the week. We can see from the github logs when you changed the notebook.md file to monitor progress. The management of the notebook will be part of your discussion grade.

The format of the notebook is very simple markdown format and must follow these rules:

- use headings with the # character and have a space after the #
- use bullets in each topic.
- each bullet **must** have an individual date that is of the form yyyy/mm/dd. Please do not lump bullet points under a single date. Have each bullet point its own date

- if you have done the activity in a period than add the second date to it yyyy/mm/dd - yyyy/mm/dd
- If you refer to section numbers in your notebook, please also add the section title as the section numbers may change in case we need to add content

Please examine carefully the sample note book is available at:

- <https://raw.githubusercontent.com/bigdata-i523/sample-hid000/master/notebook.md>

This will render in git as:

- <https://github.com/bigdata-i523/sample-hid000/blob/master/notebook.md>

The notebook.md is not a blog and should only contain a summary of what you have done.

5.6 Blog

Naturally you can maintain your own blog, but it is optional. If you like to maintain your own blog, you can create yourself also a blog.md file. However do not include sensitive information in there. A blog is not a replacement for the notebook. The blog will not be used for grading. If something does not go so well, do not focus on the negative things, but focus on how that experience can be overcome and how you turn it to a positive experience.

5.7 Calendar I524, E616, E516

This class is a full term class of 16 weeks.

Indiana University

The semester calendar is posted at

- <http://registrar.indiana.edu/official-calendar/official-calendar-spring.shtml>

The class begins Mon, Jan 8th and ends Fri, May 4th

5.8 Incomplete

Incompletes have to be explicitly requested in piazza. All incompletes have to be filed by May 1st.

Incomplete's will receive a fractional Grade reduction: A will become A-, A- will become B+, and so forth. There is enough time in the course to complete all assignments without getting an incomplete.

Why do we have such a policy? As we teach state-of-the-art software this software is subject to change, not only within the course, but also after the course. As we may offer some services and only have access to the TA's during the semester it is obvious that we like all class projects and homework assignments to be completed within a semester. Services that were offered during the semester may no longer be available after the semester is over and could adversely effect your planning. It will be in the students responsibility to identify such services and provide alternatives if they become unavailable. We try hard to avoid this but we can not guarantee it.

Furthermore, once an incomplete is requested, you will have 10 months to complete it. We will need 2 months to grade. No grading will be conducted over breaks. This may effect those that require student loans. Please plan ahead.

The incomplete request needs to be off the following format in piazza:

Subject:
INCOMPLETE REQUEST: HID000: Lastname, Firstname

Body:
Firstname: TBD
Lastname: TBD
HID: TBD
Semester: TBD
Course: TBD
Online: yes/no

URL notebook: TBD
URL assignment1:
URL assignment2: TBD
....
URL paper1: TBD
URL project: TBD
URL other1: TBD

Please make sure that the links are clickable in piazza. Also as classes have different assignments, make sure to include whatever is relevant for that class and add the appropriate artifacts.

Indiana University

Here is the process for how to deal with incomplets at IU are documented:

- <http://registrar.indiana.edu/grades/grade-values/grade-of-incomplete.shtml>

5.9 Waitlist

The waitlist contains students that are unable to enroll in a section of a course. Students choose to add themselves to the waitlist. They are not automatically added, but choose to do so intentionally based on the status of the course. There are two reasons for students to be on the waitlist. The first, and primary, reason is that the class is already at the scheduled, maximum capacity. Since there are no seats available, the student can elect to add themselves to the waitlist. The second reason is that the students' own schedule has a time conflict. This occurs when they are trying to enroll in a class that overlaps with the time of a class they are already enrolled in.

Students are moved from the waitlist to the regular section during a daily batch process, and not in real time. The process is not in realtime because the registrar receives many requests to increase capacity, decrease capacity, and change rooms. If the process were real time there would be a catastrophe of conflicts.

Students are moved from the waitlist in chronological order that they added themselves to the waitlist. If you are still on the waitlist there are no spaces free, the batch process has not run for the day, or the student in question has a schedule conflict.

Faculty are not able to selectively choose students from the waitlist.

How long does the waitlist process stay active?: The automated processing of the waitlist ends on Thursday of the first week of class. At this time the waitlist will no longer be processed. As the residential class starts on Friday, this may cause issues. Either talk to the department on Thursday or show up on Friday. Most likely there will be spaces left. Students on the waitlist at that time will remain on the waitlist, but remain there until the student decides to change their registration. Students may not do that, because they get assessed a change schedule fee.

Students tell me they still want to enroll after the first week of classes. How do they do this?

Beginning Monday, after the first week of class students begin to use the eAdd process to do a late addition of the course. The request is routed to the professor of record on an eDoc and the faculty will be notified via email. Faculty can deny or approve based on whatever criteria they wish to apply. If the faculty member approves, the eDoc is electronically forwarded to the Academic Operations office and we will approve the late add **if the room capacity** allows the addition, otherwise we must deny the addition because of fire marshal regulations. Many times, there are seats in a classroom/discussion/lab, but because other students have not *officially* dropped, enrollment is still at capacity.

After everything, a student that was unable to enroll in the class attended all year and completed all course work as if they had enrolled. Can the student get credit and can I give the student a grade?

Yes. There is a provision for a late registration - contact our office if this occurs. Students will be assessed a tuition fee at the time of late or retroactive registration.

5.10 Auditing the class

We no longer allow students to audit E222, I524, E516, and E616. The motivation to not offer these classes for auditing are:

- Seating in the lecture room is limited and we want foster students that enroll full time first.
- The best way to take the class is to conduct a project. As this can not be achieved without taking the class full time and as auditing the class does not provide the full value of the class, e.g. not more than 10% of the class. Hence, we do not think it is useful to audit the class.
- Accounts and services have to be set up and require considerable resources that are not accessible to students that audit the class.

5.11 Resource restrictions

- It is not allowed to use our services for profit (e.g. just enrolling in the class to use our clouds).
- In case of abuse of available compute time on our clouds the student is aware that we will terminate the computer account on our clouds and the student may have to conduct the project on a public cloud or his own computer under own cost. There will be no guarantee that cloud services we offer will be available after the semester is over. Projects can be conducted as part of the class that do not require access to the cloud.

5.12 Office Hours

Online Students: Online hours are prioritized for online students, residential students should attend the residential meetings.

Residential Students: Residential students participate in the official meeting times. If additional times are required, they have to be done by appointment. Office hours will be announced publicly. All technical office hours are public and can be attended by any student. Online hours are not an excuse not to come to the residential class.

However Residential students can in addition to the residential class use the online student meeting times. However, in that case online students will be served first. It is probably good to check into the zoom meeting and identify if the TA has time. They will be in zoom.

We suggest that you let the TA's know in piazza before you come, in order to make sure they are at the office.

TODO: TA: set up doodle for meetings. Must be completed in first week by next week Monday.

TODO: TA: set up initial times. mark they are temporary.

- Mon 6:00pm-7:00pm, 7:00pm-8:00pm, Gregor (online)
- Tue TBD, Smith Research Center
- Wed TBD, Smith Research Center
- Thu TBD, Smith Research Center
- Fri TBD, Smith Research Center
- Sat TBD, Smith Research Center

If a meeting is needed with Gregor, this is done upon appointment Tue-Thu 10am - 2:30pm. However, TA's will figure out if a meeting is needed. Please prepare your technical questions ahead of time, and place them in Piazza first. TA's and the class will try to answer them if possible

The link for joining the meeting on Zoom is posted in Piazza.

- [TBD](#)

For more up-to-date details, refer to Piazza.

5.13 Plagiarizm

In teh first week of class you will need to read the information about plagiarizm. If there are any questions about plagiarizm we require you to take a course offered from the IU educational department.

Warning

If we find cheating or plagiarizm, your assignment will be receiving an *F*. This especially includes copying text without proper attribution. In addition you will be receiving an *F* for the appropriate time for the discussion points an assignment was issued, e.g. If a paper duration assignment is 4 weeks, you get for these four weeks no discussion points, meaning an *F*. Furthermore, we will follow IU policy and report your case to the dean of students who may elect to expell you form the university. Please understand that it is your doing and the instructors have no choice as to follow university policies. Thus, please do not blame the instructors for your actions. Excuses such as "I missed the lecture on plagiarizm", "I forgot to include the original refrence as I ran out of time", "I did not understand what plagiarizm is" do not count obviously as we explicitly make the policies clear. This applies to all material prepared for class including assignments, excercises, code, tutorials, papers, and projects. If there is no time, do not submit and instead of an *F* ask for an incomplete. In fact if you know you have plagerized, do not even have us review your paper.

For more information on this topic please see:

- <https://studentaffairs.indiana.edu/student-conduct/misconduct-charges/academic-misconduct.shtml>

5.14 Tutorials, Topic Paper, Term Paper, Project Report

 section/preface/project.tex

Dependendt on the class you need to do different assignments. The assignments will be clearly posted in this document and updated in case clarification is needed.

We use the following terminology:

Tutorials: Tutorials are written in markdown, RST, or LaTeX and include information on a particular technical issue that is in general helpful for other students. Tutorials can be small, but some may need to be substantial. As we expect that the tutorials can be included in the Handbook, please be careful of plagiarizm and do not just copy the tutorial from elsewhere.

Topic Paper: A topic paper, or short paper is a smapp paper about a technology, application, or useful information that provides an overview of what you are trying to describe and analyses its relationship to the class topic. Be mindful about plagiarizm. The paper is written in \LaTeX and uses jabref for bibliography management.

Term Paper: A term paper is an enhanced topic paper. The difference is in length. Comparative or review papers can also be term papers. Term papers should have the quality to be publishable either in a workshop or as part of the handbook.

Project Paper: A project report is an enhanced topic paper that includes not just the analysis of a topic, but an actual code, with benchmark or demosnarted application use. Obviously it is longer than a paper and includes descriptions about reproducability of the application. Term papers should have the quality to be publishable either in a workshop or as part of the handbook.

Examples from prior classes are available in the class proceedings.

TODO: include link to Results section

Dependent on the class you have to fulfill different requirements. Please make sure you understand which requirement you will have.

E516 and E616 In these classes you will need to produce tutorials, topic papers and a project report with real code.

I524 same as E516 and E616, but you have the choice to substitute the project report with a term paper. There is a penalty though and we encourage studnets to do the project.

Please be aware that the project or term paper constitute to **60%** of your class grade. You have plenty of time to make this choice and if you find you struggle with programming you may want to consider a term paper instead of a project.

In case you chose a project paper your maximum grade for the entire class could be an A+. However, an A+ project must be truly outstanding and include an exceptional project report. Such a project and report will have the potential quality of being able to be published in a conference.

In case you chose a term Paper your maximum grade for the *entire* class will be an A-.

Please note that a project includes writing a project paper. However the length is a bit lower than for a term paper.

5.14.1 Team

Software projects and term papers can be conducted with one, two or three class members. We do not allow more than three members in a project. It will be up to you to determine a team, but we recommend that you chose wisely. Naturally if a team member does not contribute to the project you need to address this early on. Please do not come to us a week before the deadline is due and say a team member has not contributed, this is far to late to do any adjustment to the team. It is in your responsibility to manage the team.

5.14.2 Common Deliverables

Both Projects and Term paper have the following common deliverables

Work Breakdown: This is an appendix to the document that describes in detail who did what in the project. This section comes in a new page after the references. It does not count towards the page length of the document. It also includes explicit URLs to the git history that documents the statistics to demonstrate not only one student has worked on the project. If you can not provide such a statistic or all checkins have been made by a single student, the project has shown that they have not properly used git. Thus points will be deducted from the project. Furthermore, if we detect that a student has not contributed to a project we may invite the student to give a detailed presentation of the project.

Bibliography: All bibliography has to be provided in a jabref/bibtex file. This is regardless if you use LaTeX or Word. There is **NO EXCEPTION** to this rule. Please be advised doing references right takes some time so you want to do this early. Please note that exports of Endnote or other bibliography management tools do not lead to properly formatted bibtex files, despite they claiming to do so. You will have to clean them up and we recommend to do it the other way around. Manage your bibliography with jabref, and if you like to use it import them to endnote or other tools. Naturally you may have to do some cleanup to. If you use LaTeX and jabref, you have naturally much less work to do. What you chose is up to you.

Report Format: All reports will be using our common format. This format is not the same as the ACM format, so if you use systems such as overleaf or sharelatex, you need to upload it and use it there.

The format for LaTeX and Word found here:

- <https://github.com/bigdata-i523/sample-hid000/tree/master/paper1>

There will be **NO EXCEPTION** to this format. In case you are in a team, you can use either github while collaboratively developing the LaTeX document or use Microsoft One Drive which allows collaborative editing features. All bibliographical entries must be put into a bibliography manager such as jabref, endnote, or Mendeley. This will guarantee that you follow proper citation styles. You can use either ACM or IEEE reference styles. Your final submission will include the bibliography file as a separate document.

Documents that do not follow the ACM format and are not accompanied by references managed with jabref or endnote or are not spell checked will be returned without review.

TODO: Integrate the format from the class web page into the LaTeX section More details about the format can be found at <https://cloudmesh.github.io/classes/lesson/doc/report.html>

5.14.3 Project Paper

Systems Usage

Projects may be executed on your local computer, a cloud or other resources you may have access to. This may include:

- chameleoncloud.org
- furturesystems.org
- AWS (you will be responsible for charges)
- Azure (you will be responsible for charges)
- virtualbox if you have a powerful computer and like to prototype
- other clouds, please confirm with us.

Access to clouds must be scripted and a cmd5 extension must be developed as part of your project to receive full credit.

Deliverables

The following artifacts are part of the deliverables for a project

Code: You must deliver the **source code** in github. The code must be compilable and a TA may try to replicate to run your code. You MUST avoid lengthy install descriptions and everything must be installable from the command line. We will check submission. All team members must be responsible for one or all parts of the project.

Code repositories are for code, if you have additional libraries that are needed you need to develop a script or use a DevOps framework to install such software. Thus zip files and .class, .o files are not permissible in the project. Each project must be reproducible with a simple script. An example is:

```
git clone ....  
make install  
make run  
make view
```

Which would use a simple make file to install, run, and view the results. You are expected to integrate cmd5, which we teach in class. In addition you can use or are expected to use DOCKERFILES, ansible, or shell scripts. It is not permissible to use GUI based DevOps preinstalled frameworks. Everything must be installable and reproducible from the command line.

Data: Data is to be hosted on IUs google drive if needed. If you have larger data, it should be downloaded from the internet. It is in your responsibility to develop a download program,

Project Report: A report must be produced while using the format discussed in the Report Format section. The following length is required:

- 6 pages, one student in the project
- 8 pages, two students in the project
- 10 pages, three students in the project

License: All projects are developed under an open source license such as Apache 2.0 License. You will be required to add a LICENCE.txt file and if you use other software identify how it can be reused in your project. If your project uses different licenses, please add in a README.rst file which packages are used and which license these packages have.

5.14.4 Term Paper

In case you chose the term paper, you or your team will pick a topic relevant for the class. You will write a high quality scholarly paper about this topic. The following artifacts are part of the deliverables for a term paper. A report must be produced while using the format discussed in the Report Format section. The following length is required:

- 8 pages, one student in the project
- 10 pages, two student in the project
- 12 pages, three student in the project

5.15 Grading

Grading for homework will be done within reasonable time of the submission if the submission was on time. This however could still take multiple weeks. Students that miss the deadline will be graded on best effort, which could mean at the end of the semester as TAs will grade first papers that have been handed in on time. A 10% grade reduction will be given for residential students if the project is late. Some homework can not be delivered late (which will be clearly marked and 0 points will be given if late; these are mostly related to setting up your account and communicating to us your account names.)

It is the student's responsibility to upload submissions well ahead of the deadline to avoid last minute problems with network connectivity, browser crashes, cloud issues, etc. It is a very good idea to make early submissions and then upload updates as the deadline approaches; we will grade the last submission received before the deadline.

Note that the term paper or project paper will take a considerable amount of time and doing proper time management is a must for this class. Avoid starting your project late. Procrastination does not pay off. Starting a paper a day or even in the week before the deadline will allow you not to achieve your best. Late Projects or term papers will receive a 10% grade reduction.

For E516 I524, E616 the grading scene is

- 5% Tutorial 1
- 5% Tutorial 2
- 20% Paper 1
- 60% Term Paper or Project
- 10% Participation/Discussion

All other assignments are pass/fail assignments and are geared towards you being able to explore rather than you being bound by small assignments.

5.15.1 Grades on Canvas

The final grade for your class is **NOT** accurately posted in CANVAS. Please visit the registrar for your final grade. We have run many times into issues with CANVAS thus we try to stay as much as possible away from it. We know that the total grade will not be accurately reported in CANVAS.

Furthermore we are using only a letter grading scheme that distinguishes qualitatively between grades. Typically we do not engage in arguing if you get a point more or less. Instead we look at the artifact and decide if it is an A or A- and so on.

Once the due date is passed all incomplete assignments will appear as an *F* in CANVAS. Once we receive and review the submission this grade will be changed. Please, do not contact us if you

submitted late and you see an F in CANVAS temporarily.

5.15.2 Discussion about Grades

Should it be necessary a discussion about a grade must not be taking place via e-mail nor via piazza. YOu must use the CANVAS message feature as we want to make sure that by accident you post information to others that you do not intend to. FOr this reason we will not read such messages on pizza and in our e-mail and deleted them without reading.



6. Piazza

F

[section/doc/piazza.tex](#)

We use Piazza (<https://piazza.com>) because questions and answers on Piazza are community-edited and provides the opportunity not only for instructors, but also for students to contribute. Each question has a single answer edited by the students of the class and if needed an instructors' answer that is collaboratively edited by the instructors.

Due to this wiki-style Q&A, when a student has a question, one does not have to look through long e-mail threads but instead can look at the answer. For details that lead up to the answer you are highly encouraged to also look at some comments that lead up to the answer.

An advertisement video from Piazza summarizes the features:

- <https://www.youtube.com/watch?v=2jLSiN8E18w>

Piazza Support with a lot of information is available at:

- <http://support.piazza.com>

A good document about piazza is available at

- https://piazza.com/pdfs/piazza_product_introduction.pdf

6.1 Access to Piazza from Canvas

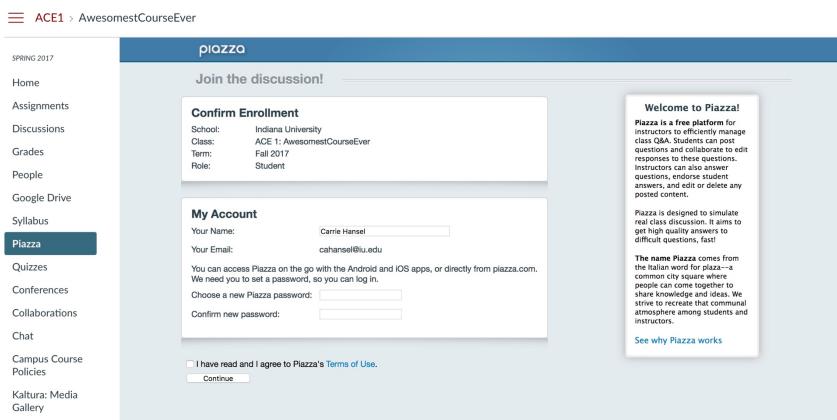
Piazza is one of the recommended IU supported technologies within CANVAS. It replaces the CANVAS discussion groups with superior technology targeted to support large student classes while also focussing on student engagement.

To access piazza you can have the following situations provided in the next four subsections. Please read *ALL** of them **CAREFULLY**, decide which applies to you and follow the instructions. If you have improvements to this instructions, please let us know.

[section/doc/piazza.tex](#)

Situation: You have never logged into piazza

First, Click the Piazza link on the left navigation of your Canvas course.



Second, create password and accept terms.

The email address shown on this screen is your default IU email address. It is the address Canvas sends to all integrated tools like Piazza. You can not edit it, so do not try.

The password you create here is for accessing Piazza from a mobile device. You must use the default IU email address from this screen to access this account on another device, so make a note of it.

This screenshot shows the 'My Account' setup page. It includes fields for 'Your Name' (Carrie Hansel) and 'Your Email' (cahansel@iu.edu). Below these are instructions for mobile access and two password input fields. At the bottom, there is a checked checkbox for accepting terms of use and a 'Continue' button.

Choose current degree program (only important if you want to opt into their recruiting program on the next screen; choose whatever you want here)

Third, associate your IU account

Set Up Your Piazza account:

Academic Information (required)

What degree are you currently pursuing?

Graduate Program: PhD

Major: Enter current major... (Input field: Instructional Systems Technology)

Anticipated Completion: May 2021

Contact us at team@piazza.com with any questions.

I'm not pursuing a degree

This information will be used for collaborative features on Piazza. We will never share your information without your permission.

Continue

Learn more about how Piazza complies with FERPA

Forth, if all goes well you see the Success screen

Success! Before heading to your classes, take a moment to register for Piazza Careers...

Find your next job or internship at:

Piazza Careers
It's free and always will be!

I'm open to hearing from and connecting with companies and alumni (employers can view your Careers profile)

I don't need any help getting the most fulfilling and rewarding career opportunities at this time

Continue

+ 100s more

Situation: You have logged into piazza and used your default IU e-mail

1. Click the Piazza link on the left navigation of your Canvas course.
2. You will be automatically enrolled in the course Piazza site and logged in.
3. Start using Piazza.

Situation: You have logged into piazza and you used another non IU e-mail

1. Click the Piazza link on the left navigation of your Canvas course.
2. Proceed as in #1 above. This will create your new Piazza account that is linked to your courses in Canvas. This is the account you should always use in your IU courses.
3. If you wish to merge other accounts that you own, please see [Add an email address or merge two accounts](#).

Situation: You have multiple accounts in piazza

1. If one of your multiple accounts corresponds with your default IU email address, you will be automatically enrolled in the course Piazza site and logged in.
2. If none of your accounts corresponds to your default IU email address, follow the instructions in #3 above.
3. If you wish to merge other accounts that you own, please see [Add an email address or merge two accounts](#).

I post the official response form the CANVAS team here: "When a student clicks the Piazza link in

your course navigation, they will be authenticated through to Piazza. If the student already has a Piazza account that matches their default Canvas email, they will simply be enrolled in the Piazza course. If the student doesn't have an account, Canvas sends the pertinent information (default email address primarily) to Piazza, Piazza creates the student's account and enrolls the student in the Piazza course. There is nothing you need to do to."

If you have any questions regarding accessing piazza, please send them to
“Ricci, Margaret P” <mricci@iu.edu>

6.2 Verify you are on Piazza via a post

Post on the **bio** folder a short introduction about yourself. One that you could include in a paper.
An example is provided at <https://laszewski.github.io/bio.html> with an image at https://laszewski.github.io/_images/gregor.jpg

Use the subject line *Biography: Firstname Lastname* and post it into the bio folder.

6.3 Making Piazza Work

In order for Piazza to work students and instructors need to participate

Students participate: Students must collaboratively work on an answer to a question. Students must not post irrelevant followups to a question. If you notice your comment was irrelevant, please delete it. Students must **search** prior to asking a new question if the question has already been asked. Duplicated questions can be merged.

Instructors guide: The instructor guides the students in order to obtain an answer to a question. In some cases the instructor may be the only one knowing the answer in which case he tries to provide it.

Not using e-mail: Instructors will and must not use e-mail to communicate with a student. All communication will be done via piazza. There, are only very few situations where e-mail is allowed, ask on piazza first if you should engage in e-mail conversations.

Not using CANVAS discussions: We will not engage in any CANVAS message exchange. Any communication is to be done on Piazza. It is in your responsibility to enroll in Piazza to make it work for you. Instructions are posted in this document. (Grade discussion will be done in CANVAS)

6.4 Towards good questions

Naturally when you ask a question you need to do it in a reasonable form and provide sufficient information so that the question can be answered. It is in the responsibility of the student to update the question to provide enough information.

Thus information may include: Firstname Lastname, HID, and URL to document in question

To give you an example of a **bad** question consider:

send from Xi Lee

Hi Professor:

I read a nice article about apples and potato's and updated my paper. Please give me feedback

Thank you

Kevin

Here the reasons why this can be improved:

1. As professors and instructors may review your document it is unnecessary to start with "Hi Professor:", just leave it away. If you want a particular instructor use the name explicitly, such as "Gregor:", e.g. multiple professors may be teaching your course.
2. You have not specified which article you read, you need to include the URL to the article so we can follow your argument.
3. You have not included the link to your document so we do not know what you are talking about. Remember there are many others students in the class
4. You are using a different name from the one that you are registered with. This can lead to confusion when we look up your name. We prefer that you use only one name that is associated with your e-mail.

The above question will simply be commented on (if at all):

"Missing information" or "?" indicating that information is missing.

It is in your responsibility to figure out which information is missing. You need to modify the original post and.

6.5 Guide on how to ask good questions

This guide is adapted from

- <http://www.techsupportalert.com/content/how-ask-question-when-you-want-technical-help.htm>

Ten steps to getting your question answered on piazza

1. Before you even go to ask a question, think through what your problem is. Write down how you are going to describe it. Think about it from the other side - what would you need to know if a student came to you and asked the question? Gather all the system information that seems to bear on the problem (see how at this link). Sometimes it even happens that by thinking through the problem, you come up with the answer yourself.
2. Verify that your question has not yet been answered with a search on the Web, Class Web page, or class piazza, this may require multiple searches.
3. In case it is a technical question, write down any error codes that appear on your screen. **do not use screenshots** if the text is characters. This is because a reply my need to paste and copy from the original. Also screenshots are not searchable. We will not answer any questions that post screenshots if they are not necessary. It is far easier to copy and paste and use terminal type in the formatting. Also if the text is posted it is searchable. (Any unnecessary screenshot will receive a point deduction. Based on experience we have to do this as previous students in other classes ignored this policy).
4. Place your question or problem in a forum that is relevant to its subject. That may seem obvious but anyone who has experience with forums knows that a lot of questions show up in the wrong place. You will need to identify one or more a fitting piazza "folders" (folders sort the posts by topics).

5. Select a title that briefly and accurately describes your problem. A title like “Help!” or “Computer won’t work” will often get ignored. Almost any problem can be titled with a few key words that will raise interest in somebody who is familiar with the subject. A corollary to this is to avoid using all caps or a lot of exclamation points. Something like “HELP!!!” turns many people off.
6. In the post, briefly describe the problem in a paragraph. Leave out unnecessary details. Save everybody time by listing any solutions that you have tried but didn’t work. Avoid using screenshots if they are not needed. (I mention this again).
7. IN case of a technical issue describe relevant system details. For example, it is essential to designate your operating system and type of computer and any components that might be involved in your problem. List any error code that has been displayed. Be prepared to provide more details if asked.
8. Tell what you were doing when you encountered the problem. If it is a reproducible problem, list the steps or computer operations that cause the problem.
9. If applicable, List any recent software you have installed or hardware changes you have made. If you have updated any drivers recently, also list that.
10. Formulate your questions and answers in a courteous manner. Respect the answers from others. Somebody is giving you their time and expertise for free. You may want to come back to the forum and it pays to be friendly.
11. If a suggested solution works, be sure to return to piazza and report your success. It is the least you can do to return something for the help you have been given. It will make you welcome in the forum the next time you go there for help.

6.6 Piazza class Links

Using the following direct links can lead to you not getting proper access via Canvas. If you click on these links **before they create** the account via the link in your current Canvas course, you will create an account that is not matched up with Canvas.

To avoid issues make sure you integrate to piazza via Canvas first.

If you have questions bout this contact Margaret Ricci.

Classes hosted on Piazza

6.6.1 Current Classes

- E516 Spring 2018: <https://piazza.com/iu/spring2018/e516spring18/home>
- E616 and I524 Spring 2018: <https://piazza.com/iu/spring2018/e616spring18/home>

6.6.2 Previous Classes

- I523 Fall 2017: <https://piazza.com/iu/fall2017/i523/home>
- I524 Spring 2017: <https://piazza.com/class/ix39m27czn5uw>
- I523 Fall 2016: <https://piazza.com/class/irqfvh1ctr2vt>

6.7 Piazza Curation

We are using Piazza in a curated fashion and we like that all students participate in this. This will allow Piazza to become a superior tool for all in the class. In general we only allow **exactly one folder** for a message. If a message is wrongly filed it will be corrected, either by students or TAs.

As part of this we are introducing a number of folders. Some of which must not be used by students. We list the following folders and their purpose:

Folder	Description
logistics	Any question and discussion related to the logistics of the course
lectures	Any question and discussion related to the lectures.
p1	Any question and discussion related to paper 1.
p2	Any question and discussion related to paper 2.
t1	Any question and discussion related to paper 1.
t2	Any question and discussion related to paper 2.
project	Any question and discussion related to iot projects.
term-paper	Any question and discussion related to the term project.
python	Any question and discussion related to python.
pi	Any question and discussion related to the Raspberry Pi 3. We are not using older Raspberry Pi's and therefore can not comment to them.
8266	Any question and discussion related to the esp8266.
bio	A homework folder in which you only publish your bio. The bio needs to be published as a <i>note</i> . This assignment also serves us to see if you are in piazza. Please do this assignment ASSAP. You need to post a formal bio. See the many great examples in the folder.
help	If you need help and none of the other folders fits, please use this folder. If information from here will result into new Web page content it will be added and marked into the folder <i>resolved</i> . See the <i>resolved</i> folder for more detail.
resolved	Sometimes we move some general help messages to the resolved folder in case the help message results into information that is posted on our class Web page. We than will add a link to where in the class Web page this question was answered. The TAs will aggressively try to put information into the Web page.
discussion	Any content that deserves its separate discussion and is not covered in the above folder.

In addition to these general folders we also have two folders which **MUST NOT BE USED BY ANY STUDENT TO POST CONTENT**. These folders serve to communicate your assignments and are used internally between Grgeor and the TA's.

assignments: This folder only lists the assignments. At any time in the class you can click on the assignment folder and list the assignments given to the class. Thus there is no confusion which assignments have been given. In case students have questions about assignments they should not use the *assignments* folder, but the *help* folder. TAs are instructed to correct wrongly filed messages in folders.

ta: Any question and discussion you have for the ta's. Typically you should however use the folder *help*. Gregor use most often the *ta* folder for internal coordination with the tas.

It may be necessary to create new folders for the class. Their meaning will be updated here once this occurs.

In case you decide to post privately and the information is useful for others also, the message will be published to the class.

6.8 Read the Originals, not just the e-mail

Piazza provides a convenient mechanism to update you through e-mail when an answer is changed or when someone posts.

However, this is just a reminder that something happened. In some cases I always recommend that instead of only reading the mail to use the <click here> feature in the mail to get not only to the update, but to the actual post. Then you can get reminded about the information that is part of the post and potentially answers your question in full. It is not sufficient to participate in this class while only reading email, you should participate while visiting piazza and actively contribute to it.

6.9 Exercises

Exercise 6.1 Enroll in piazza



Exercise 6.2 Post a short formal bio in the bio folder and optionally include a professional portrait of yourself. Make sure you understand what a formal bio and portrait is. Research this in the internet. Look at IEEE papers for examples.



Exercise 6.3 How do you find out within Piazza which assignments have been posted?



Exercise 6.4 Please watch the Video about Piazza



IV Documenting Scientific Research

7	Documenting Scientific Research	81
7.1	Overview	
7.2	Plagiarism	
7.3	Acknowledgements	
7.4	Writing a Scientific Article or Conference Paper	
8	Introduction to \LaTeX	93
8.1	Installation	
8.2	Basic \LaTeX Elements	
8.3	Advanced topics	
8.4	Editing	
8.5	The \LaTeX Cycle	
8.6	Tips	
9	Managing Bibliographies	111
9.1	Entry types	
9.2	Integrating Bibtex entries into Other Systems	
9.3	Other Reference Managers	
10	Editors	127
10.1	Basic Emacs	
11	Other Formats	131
11.1	reStructuredText	
11.2	Markdown	
11.3	Communicating Research in Other Ways	
12	Book Format	139
12.1	Paragraphs of Text	
12.2	Citation	
12.3	Lists	
12.4	Theorems	
12.5	Definitions	
12.6	Notations	
12.7	Remarks	
12.8	Corollaries	
12.9	Propositions	
12.10	Examples	
12.11	Exercises	
12.12	Problems	
12.13	Vocabulary	
12.14	Tables	
12.15	Figures	



7. Documenting Scientific Research

 part/doc.tex

7.1 Overview

 section/doc/overview.tex

Using a paper as part of your project planning is an important learning outcome. Instead of starting with a project we recommend that you start with a paper to direct your research.

This argument is made also by the following presentation.

How to write a paper by Simon Peyton Jones (57:39) 

We do recommend that you read the sections in this part carefully as they will introduce you to important tools that make writing a paper relatively simple while allowing professional paper format and bibliography management tools.

To get a first impression we have also prepared a number of videos that may help you. However, note that the format for papers used in these videos is different from the class and you must use the written documentation instead and use that format. Papers not using our format will be returned without review. I suggest you start right from the beginning.

Warning

The videos that show you the ACM paper template that we do not use

ShareLaTeX (8:49) 

jabref (14:41) 

Exercise 7.1 Watch the three lectures about How to write a paper, ShareLaTeX, and jabref. ■

 section/doc/plagiarism.tex

7.2 Plagiarism

We start with the review of a most important topic.

7.2.1 Plagiarism Definition

In academic life it is important to understand and avoid plagiarism. The dictionary defines plagiarism as follows [dictionary.com](#):

pla·gia·rism “the practice of taking someone else’s work or ideas and passing them off as one’s own.”

7.2.2 Plagiarism Policies

Organizations and universities will have policies in place do address plagiarism. An example is provided for Indiana University [635]. We quote:

“Honesty requires that any ideas or materials taken from another source for either written or oral use must be fully acknowledged. Offering the work of someone else as one’s own is plagiarism. The language or ideas thus taken from another may range from isolated formulas, sentences, or paragraphs to entire articles copied from books, periodicals, speeches, or the writings of other students. The offering of materials assembled or collected by others in the form of projects or collections without acknowledgment also is considered plagiarism. Any student who fails to give credit for ideas or materials taken from another source is guilty of plagiarism.

(Faculty Council, May 2, 1961; University Faculty Council, March 11, 1975; Board of Trustees, July 11, 1975)”

Faculty members at Universitys are also bound by policies that mandate reporting. At Indiana University the following policy applies (for a complete policy see the Web page):

“Should the faculty member detect signs of plagiarism or cheating, it is his or her most serious obligation to investigate these thoroughly, to take appropriate action with respect to the grades of students, and *in any event* to report the matter to the Dean for Student Services [or equivalent administrator]. The necessity to report every case of cheating, whether or not further action is desirable, arises particularly because of the possibility that this is not the student’s first offense, or that other offenses may follow it. Equity also demands that a uniform reporting practice be enforced; otherwise, some students will be penalized while others guilty of the same actions will go free.

(Faculty Council, May 2, 1961)”

Naturally if a student has any questions about understanding plagiarism the University can provide assistance. If a student is in doubt and asks for help this is not considered at that time plagiarism.

As you can see from the previous policies, the faculty do not have any choice but reporting real cases of plagiarism to the university administration. Thus you must not hold them personally

responsible as this is part of the tasks they are required to do if they like it or not. Instead, it is **the responsibility of the authors of the document** to assure no plagiarism occurs. If you are a student of a class that writes a paper or project report this naturally also all applies to you. In addition, if you work in a team you need to assure the entire team addresses plagiarism appropriately.

In practice this means that the teachers of a course expect you to know plagiarism and you need to be informed about it. This is typically done in other courses. However, as it is often overlooked by the student we are pointing it out here so we can make sure you contribute to courses that require you to write papers and reports. This also means you can not claim you did not know what plagiarism is. You are required to know what it is, know how to detect it and know how to avoid it. The resources provided next will give you the necessary tools and background.

7.2.3 Plagiarism Resources

The [School of Education at Indiana University](https://www.indiana.edu/~istd/patterns.html) has a significant set of resources to get educated about plagiarism. These resources are intended to “preparing educators, advancing knowledge, and improving education” [] <https://www.indiana.edu/~istd/patterns.html>

The content here is copied from the Web Page

- <https://www.indiana.edu/~istd/patterns.html>

As such we have not included quotes but refer to their Web page for the original source which may also include updates. Naturally we do not want to be accused of plagiarize in a chapter about plagiarism. Thus assume the content for the rest of this chapter are copied from that Web page.

7.2.4 Pattern

- [IU Definition](#) of Plagiarism from Student Code of Conduct
- [Overview](#) How to give proper credit, steps.
- [Cases](#) of Plagiarism in the US, in the news, and elsewhere
- [Examples](#) Word for word, paraphrasing
- [Practice](#) with feedback on word-for-word and paraphrasing plagiarism
- [Test](#) 10 questions on recognizing plagiarism
- [Tutorial Site Map](#) Expanded table of contents
- [Resources](#) Websites, books, dictionary links, references for learning more about plagiarism

7.2.5 Tutorials

A number of tutorials are offered by Indiana University [Instructional Systems Technology Department](#) Web pages dealing with plagiarism. These include:

- [Plagiarism Tutorial](#)
- [Understanding Plagiarism](#)

7.2.6 How to Recognize Plagiarism

We are listing fifteen patterns of plagiarism that are defined on the Web pages identified in Section 7.2.5:

Name	Plagiarism Type	Reason
Clueless Quote	word-for-word	no quotes, no citation, no reference
Crafty Cover-up	proper paraphrase but word-for-word	also present
Cunning Cover-up	paraphrasing	no citation, no reference
Deceptive Dupe	word-for-word	no quotes, no citation, but has reference
Delinked Dupe	word-for-word	no reference, even though quotes and citation
Devious Dupe	correct quote but word-for-word	also present
Dippy Dupe	word-for-word	quotes missing, even though full citation and reference
Disguised Dupe	looks like proper para, but actually word-for-word	no quotes, no locator
Double Trouble	word-for-word and paraphrasing	although has reference
Linkless Loser	word-for-word	citation and reference lacking, although has quotes and locator
Lost Locator	word-for-word	missing locator, although has quotes, citation, and reference
Placeless Paraphrase	paraphrasing	no reference, although citation present
Severed Cite	paraphrasing	reference but no citation
Shirking Cite	word-for-word	lacks locator and reference, although quotes and citation present
Triple D–Disguised Disconnected Dupe	word-for-word	looks like proper para, but no quotes, no reference, no locator

In addition they do specify three patterns of non-plagiarism:

Name	Type	Description
Correct Quote	non-plagiarism	takes another's words verbatim and acknowledges with quotation marks, full in-text citation with locator, and reference
Proper Paraphrase	non-plagiarism	summarizes another's words and acknowledges with in-text citation and reference
Parroted Paraphrase	non-plagiarism	appears to be paraphrasing, and technically may not be plagiarism, but . . . ???



section/doc/acknowledgement.tex

7.3 Acknowledgements

In many cases you want to include an acknowledgement section. In some cases you may be tempted to eliminate this section as you think you are out of space and the acknowledgement section may give you some additional space. This however is the wrong strategy and you should not do this. Instead you should shorten your paper elsewhere and leave enough space for acknowledgements.

In some cases where you get financial support from a university for a project such as from NIH or NSF specific information **must** be included. The best way is to verify with your coauthors.

Additional acknowledgements may have to be added and you need to evaluate if for example significant help on the paper warrants co-authorship.

An issue that we have seen often is for example when a professor has helped significantly on the paper but is not properly acknowledged. This can even lead to the professor asking you to remove him from the acknowledgement. A bad acknowledgement example is the following:

We like to thank Professor Zweistein for his help in compiling the \LaTeX paper.

We do not think that the professor will be happy with this acknowledgement as it sounds like the only thing that was provided was the help on \LaTeX that you should have done anyways without the help of the professor. Ask yourself, if he introduced you to the field, has helped you with preparing the text, has given you insights, has corrected things in your paper, made suggestions. So instead of the above maybe a more general term such as *helped with the paper* would be more appropriate. If not leaving it off is more appropriate. In some cases you may want to invite your professor to become a co-author.



section/doc/report-book.tex

7.4 Writing a Scientific Article or Conference Paper

An important part of any scientific research is to document it. This is often done through scientific conferences or journal articles. Hence it is important to learn how to prepare and submit such papers. Most conferences accept typically the papers in PDF format but require the papers to be prepared on MSWord or in \LaTeX . While working with many students in the past we noticed however that those students using Word often spend unnecessarily countless hours on trying to make their papers beautiful while actually violating the template provided by the conference. Furthermore, we noticed that the same students had issues with bibliography management. Instead of Word helping the student it provided the illusion to be easier than \LaTeX but when adding up the time spent on the paper we found that \LaTeX actually saved time. This has been especially true with the advent of collaborative editing services such as sharelatex [539] and overleaf [459].

In this section we provide you with a professional template that is used for either system based on the ACM standard that you can use to write papers. Naturally this will be extremely useful if the quality of your research is strong enough to be submitted to a conference. We structure this section as follows. Although we do not recommend that you use MSWord for your editing of a scientific paper, we have included a short section about it and outline some of its pitfalls that initially you may not think is problematic, but has proven to be an issue with students. Next we will focus on introducing you to \LaTeX and showcasing you the advantages and disadvantages. We will dedicate an entire section on bibliography management and teach you how to use jabref which clearly has advantages for us.

Having a uniform report format not only helps the students but also allows the comparison of paper length and effort as part of teaching a course. We have added an entire section to this chapter that discusses how we can manage a *Class Proceedings* form papers that are contributed by teams in the class.

7.4.1 Professional Paper Format

The report format we suggest here is based on the standard ACM proceedings format. It is of very high quality and can be adapted for your own activities. Moreover, it is possible to use most of

teh text to adapt to other formats in case the conference you intend to submit your paper to has a different format. The ACM format is always a good start.

Important is that you do not need to change the template but you can change some parameters in case you are not submitting the paper to a conference but use it for class papers. Certainly you should not change the spacing or the layout and instead focus on writing content. As for bibliography management we recommend you use jabref which we will introduce in Section 9.0.1.

We recommend that you carefully study the requirements for the report format. We would nat want that your paper gets rejected by a journal, conference or the class just because you try to modify the format or do not follow the established publication guidlines. The template we are providing is available from:

- <https://github.com/bigdata-i523/sample-hid000/tree/master/paper1>

You will find in it a modified ACM proceedings templates that you must use.

7.4.2 Submission Requirements

Although the initial requirement for some conferences or journals is the document PDF, in many cases you must be prepared to provide the source when submitting to the conference. This includes the submission of the original images in an images foder. You may be asked to package the document into a folder with all of its sources and submit to the conference for professional publication.

7.4.3 Microsoft Word vs. L^AT_EX

Microsoft word will provide you with the initial impression that you will safe lots of time writing in it while you see the layout of the document. This will be initially true, but once you progress to the more challenging parts and later pages such as image menagement and bibliography management you will see some issues. Thes include that figure placement in Word need sto be done just right in order for images to be where they need. We have seen students spending hours with the placement of figures in a paper but when they did additional changes the images jumped around and were not at the place where teh students expected them to be. So if you work with images, make sure you understand how to place them. Also always use relative caption counters so that if an image gets placed elsewhere the counter stays consistent. So nefer use just the number, but a reference to the figure when referring to it. Recently a new bibliography management system was added to Word. However, however it is not well documented and the references are placed in the system bibliography rather than a local managed bibliography. This mah have severe consequences when working with many authors on a paper. The same is true when using Endnote. We have heard in many occasions that the combination of endnote and Word destroyed documents. You certainly do not want that to happen the day before your deadline. Also in classes we observed that those using LaTe_X deliver better structured and written papers as the focus is on text and not beautiful layout.

For all these reasons we do not recommend that you use Word.

In LaTe_X where we have an easier time with this as we can just ignore all of these issues due to relative good image placement and excellent support for academic reference management. Hence, it is in your best interest to use LaTe_X. The information we provide here will make it easy for you to get started and write a paper in no time as it is just like filling out a form.

7.4.4 Working in a Team

Today research is done in potentially large research teams. This also include the writing of a document. There are multiple ways this is done these days and depends on the system you chose.

In MSWord you can use skydrive, while for LaTeX you can use sharelatex and overleaf. However, in many cases the use of github is possible as the same groups that develop the code are also familiar with github. Thus we provide you here also with the introduction on how to write a document in github while group members can contribute.

Here are the options:

LaTeX and git: This option will likely save you time as you can use jabref also for managing collaborative bibliographies and

sharelatex: an online tool to write latex documents

overleaf: an online tool to write latex documents

MS onedrive: It allows you to edit a word document in collaboration. We recommend that you use a local installed version of Word and do the editing with that, rather than using the online version. The online editor has some bugs. See also (untested): <http://www.paulkiddie.com/2009/07/jabref-exports-to-word-2007-xml/>, <http://usefulcodes.blogspot.com/2015/01/using-jabref-to-import-bib-to-microsoft.html>

Google Drive: google drive could be used to collaborate on text that is than pasted into document. However it is just a starting point as it does not support typically the format required by the publisher. Hence at one point you need to switch to one of the other systems.

7.4.5 Time Management

Obviously writing a paper takes time and you need to carefully make sure you devote enough time to it. The important part is that the paper should not be an after thought but should be the initial activity to conduct and execute your research. Remember that

1. It takes time to read the information
2. It takes time understand the information
3. It takes time to do the research

For deadlines the following will get you in trouble:

1. *There are still 10 weeks left till the deadline, so let me start in 4 week* Procrastination is your worst enemy.
2. If you work in a team that has time management issues address them immediately
3. Do not underestimate the time it takes to prepare the final submission into the submission system. Prepare automated scripts that can deliver the package for submission in minutes rather than hours by hand.

7.4.6 Paper and Report Checklist

In this section we summarize a number of checks that you may perform to make sure your paper is properly formatted and in excellent shape. Naturally this list is just a partial list and if you find things we should add here, let us know.

A checklist with a subset of these issues that you can add to your draft is available at

- <https://github.com/bigdata-i523/sample-hid000/blob/master/paper1/issues.tex>

7.4.7 Content

- Is the paper formal paper and not an experience report?
- Do not include phrases such as “In week 1 we did this”
- When writing the *proposal* do not use the word “proposal” write the document as if it would be teh final paper. We see too many reports at the end forgetting to remove the word proposal in the final paper, so we can not tell if you did it or if it is still a proposal. As the final paper is not a proposal we reject such papers and you get a 1/3 grade reduction. To avoid this, just do not use the word proposal.
- When writing the abstract do not make it a proposal. Abstracts are no proposals, e.g We propose to do, We intend to show If the paper intends to show you are still in the draft phase of the paper. However, if you say We show ... That would be good. Let us just assume you intended to show something but did not achieve then you can say “We intended to show this but we it was not possible to verify. We have provided reasosn for this in the paper”. As you can see not only the intention is communicated, but the result. If you just focuss on the intent that is just a proposal and is not a proper abstract.
- Add keywords to the paper, where the first two are your HID, and your class number.
- If your paper is an introduction or overview paper, please do not assume the reader to be an expert. Provide enough material for the paper to be useful for an introduction into the topic.
- If your paper limit is x number of pages but you want to hand in x plus 100 pages. If however you page limit is 2 pages and you hand in 4 or 6 pages that is no issue.

7.4.8 Submission

- Do not make changes to your paper during grading, when your repository should be frozen.
- Do not use filenames and directory names that have spaces in them only use [a-z0-9]*
- Make all file names lower case other than Makefile and README.yml
- You are required to run yamllint README.yml on all team members README.yml including your own. All of them must pass. Do this on the first day you start writing the paper. Only push and commit the files when they pass this test. If you do not have yamllint you can write one in python. Its 3 lines of code.
- Have you included the paper in the submission system (In our case git). This includes all images, bibliography files and other material that is needed to build the paper from scratch?
- Have you made sure your paper compiles with *make* and the provided Makefile before you committed?
- Are all images checked in?
- Did you submit the rebort.bib file?

7.4.9 Bibliography

- Are you managing your references in jabref and endnote (we need both)
- In the author field, authors are separated with an *and* and not a comma.
- The filename for the bibliography is report.bib.
- Bibtex labels must have any spaces, _ or & in it
- Fix citations in text that show as [?]. This means either your report.bib is not up-to-date or there is a spelling error in the label of the item you want to cite, either in report.bib or in report.tex
- Urls in citations are never placed in howpublished, instead we use url = { }. howpublished is just used for a text sting such as Web Page, Blog, ...
- Do not use the \url={ } in teh text, instead use a citation.

- Are you references correct? References to a paper are no afterthought, they should be properly cited. Use jabref and make sure the citation type of the reference is correct and fill out as many fields as you can. Some journals and conferences have for example special requirements that go beyond the requirements of for example jabref. One example is that many conferences require you that when you cite papers from another conference to augment the conference not only with the location where the conference took place, but also with the dates the conference took place. Unfortunately, this is information that is often only available through additional google queries and many reference entries you find in the internet do not have this information readily available.

7.4.10 Writing

- Have you spellchecked the paper?
- Have you grammar checked the paper?
- Use proper capitalization in the title, see: <https://capitalizemytitle.com/>
- Are you using *a* and *the* properly?
- Short form of verbs is for spoken language. Do not use them in scientific writing. Example: can't is incorrect, cannot is correct.
- Do not use phrases such as *shown in the Figure below*. Instead, use *as shown in Figure 3*, when referring to the 3rd figure, but use the *ref label* macros.
- Do not use the word *I* instead use *we* even if you are the sole author. In many cases you may want to avoid using the word *we* also.
- Do not use the phrase *In this paper/report we show* instead use *We show*. It is not important if this is a paper or a report and does not need to be mentioned.
- If you want to say *and* do not use & but use the word *and*.
- Use a space after . , :
- When using a section command, the section title is not written in all-caps as the L^AT_EX template will do this automatically for you. Thus it is \section{Introduction} and NOT \section{INTRODUCTION}.

7.4.11 Citation Issues and Plagiarism

- It is your responsibility to make sure no plagiarism occurs.
- When stating claims you added the proper citations.
- Do avoid paraphrasing long quotations (whole sentences or longer) from other papers.
- Double check your paper if you have quote from other papers and included the citation.
- The \cite{} command must not be in the beginning of the sentence or paragraph, but in the end, before the period mark. example: ... a library called Message Passing Interface(MPI) [7].
- Put a space between the citation mark and the previous word or better use ~.
- There must not be any citation in the abstract or conclusion.

7.4.12 Character Errors

The following errors are very often found and must be avoided.

- To emphasize a word, use *emphasize* and not “quote”. Quotes are reserved for quotes from other papers and must not be used to emphasize words or phrases. to put around a word that you like to emphasize.
- Generally we do not use **bold fett** text. Instead use *em*.
- Erroneous use of quotation marks, i.e. use ‘‘quotes’’, but not the double quote that you

find on your keyboard such as " ".

- When using the characters & # % _ put a backslash before them so that they show up correctly.
- Pasting and copying from the Web often results in non-ASCII characters to be used in your text, please remove them and replace accordingly. This is the case for quotes, dashes and all the other special characters.
- If you see a figure and not a figure in text you copied from a text that has the fi combined as a single character. It happens often with combinations of f such as fi fl ff

7.4.13 Structural Issues

- Does your paper include an Acknowledgement section.
- Is the acknowledgment including all the people appropriately that helped you in your activity.
- In case of a class and if you do a multi-author paper, you need to add an appendix called *Workbreak Down* describing who did what in the paper, after the bibliography
- Do you fulfill the minimum page length such as defined in the submission guideline. Remember that images, tables and figures do not count towards the page length.
- Do not artificially inflate your paper if you are below the page limit.
- In case you have an appendix it is included after the bibliography

7.4.14 Figures and Tables

- Images must be at least 300dpi if they are not in a scalable format such as PDF which you can generate from Powerpoint and other drawing programs.
- If you use Microsoft products, use ppt 4:3 ratio for drawing decent images. In case there is a powerpoint in the submission, the image must be exported as PDF.
- If you have OSX, you are allowed to use omnigraffle.
- Make sure you capitalize Figure 1, Table 2 when used in a sentence.
- Do use \label{} and \ref{} to automatically create figure numbers.
- Figure caption must be below the image.
- Table captions must be above the image.
- Do not include the titles of the figures in the figure itself but instead use the caption or that information.
- All images must be in native format, e.g. .graffle, .pptx, .png, .jpg in the images directory
- Do not submit eps images. Instead, convert them to PDF
- The image files must be in a single directory named *images*
- Make the figures large enough so we can read the details. If needed make the figure over two columns
- Do not worry about the figure placement if they are at a different location than you think. Figures are allowed to float. To illustrate this case we force all images to be placed at the end of the paper, although you may have included it at a special location in the paper. This forces you to avoid the phrases as seen in the following image, but you need to use the ref and label features in LaTeX.
- In case you copied a figure from another paper you need to ask for copyright permission. In case of a class paper, you must include a reference to the original in the caption. In general we like to avoid this for the reports and like that you produce original pictures.
- Remove any figure that is not referred to explicitly in the text with a ref command. Again just putting in the number will not be good enough. This allows you to place the figure in the final submission at a location without needing to fix the numbers.
- Do not use textwidth as a parameter for includegraphics, but instead use \columnwidth as

demonstrated in our template.

- Figures should be reasonably sized and often you just need to add columnwidth e.g.

```
/includegraphics[width=1.0\columnwidth]{images/myimage.pdf}
```

Do not play with the size, just leave it with 1.0.

If you observe something missing let us know.

7.4.15 Example Paper

An example report in PDF format is available:

- [report.pdf](#)

7.4.16 Creating the PDF from LaTeX on your Computer

Latex can be easily installed on any computer as long as you have enough space. Furthermore if your machine can execute the make command we have provided in the standard report format a simple [Makefile](#) that allows you to do editing with immediate preview as documented in the LaTeX lesson.

7.4.17 Class Specific README.md

For the class we will manage all papers via [github.com](#). You will be added to our github at

- <https://github.com/bigdata-i523>

and assigned an hid (homework index directory) directory with a unique hid number for you. In addition, once you decide for a project, you will also get a project id (pid) and a directory in which you place the projects. Projects must not be placed in hid directories as they are treated differently and a class proceedings is automatically created based on your submission.

As part of the hid directory, you will need to create a README.md file in it, that **must** follow a specific format. The good news is that we have developed an easy template that with common sense you can modify easily. The template is located at

- <https://raw.githubusercontent.com/bigdata-i523/sample-hid000/master/README.md>

As the format may have been updated over time it does not hurt to revisit it and compare with your README.md and make corrections. It is important that you follow the format and not eliminate the lines with the three quotes. The text in the quotes is actually yaml. yaml is a data format the any data scientist must know. If you do not, you can look it up. However, if you follow our rules you should be good. If you find a rule missing for our purpose, let us know. We like to keep it simple and want you to fill out the *template* with your information.

Simple rules:

- replace the hid number with your hid number.
- naturally if you see sample- in the directory name you need to delete that as your directory name does not have sample- in it.
- do not ignore where the author is to be placed, it is in a list starting with a -
- there is always a space after a -
- do not introduce empty lines
- do not use TAB and make sure your editor does not bay accident automatically creates tabs. This is probably the most frequent error we see.

- do not use any : & _ in the attribute text including titles
- an object defined in the README.md must have on a single type field. for example in the project section. Make sure you select only one type and delete the other
- in case you have long paragraphs you can use the > after the abstract
- Once you understood how the README.md works, please delete the comment section.
- Add a chapter topic that your paper belongs to

7.4.18 Exercises

Exercise 7.2 Install latex and jabref on your system □

Exercise 7.3 Check out the report example directory. Create a PDF and view it. Modify and recompile. □

Exercise 7.4 Learn about the different bibliographic entry formats in bibtex □

Exercise 7.5 What is an article in a magazine? Is it really an Article or a Misc? □

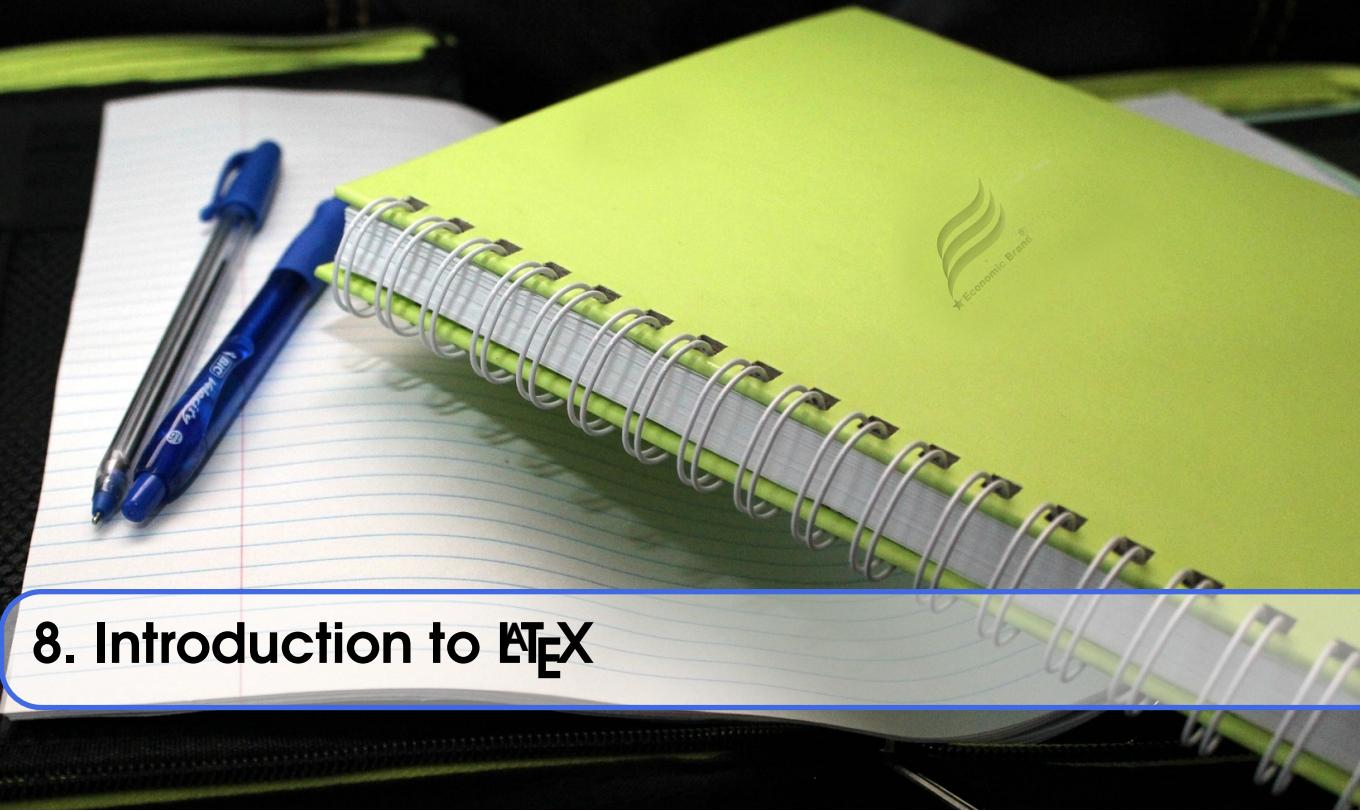
Exercise 7.6 What is an InProceedings and how does it differ from Conference? □

Exercise 7.7 What is a Misc? □

Exercise 7.8 Why are spaces, underscores in directory names problematic and why should you avoid using them for your projects □

Exercise 7.9 Write an objective report about the advantages and disadvantages of programs to write reports. □

Exercise 7.10 Why is it advantageous that directories have no underscore or space in the name? □



8. Introduction to \LaTeX

F

section/doc/latex.tex

Mastering a text processing system is an essential part of a researcher's life. Not knowing how to use a text processing system can slow down the productivity of research drastically.

8.1 Installation

LaTeX is available on all modern computer systems. A very good installation for OSX is available at:

- <https://tug.org/mactex/>

However, if you have older versions on your systems you may have to first completely uninstall them.

8.1.1 Local Install

Installing LaTeX is trivial, and is documented on the internet very well. However, it requires sufficient space and time as it is a large environment. A system such as TeX Live takes in full install about 5.5 GB. In addition to LaTeX we recommend that you install jabref and use it for bibliography management.

Thus you will have the most of them on your system.

- pdflatex: the latex program producing pdf
- bibtex: to create bibliographies
- jabref: GUI application to bibtex files (<http://www.jabref.org/>)

Make sure you check that these programs are there, for example with the Linux commands:

section/doc/latex.tex

```
which pdflatex
which bibtex
which jabref (on OSX you may have an icon for it)
```

If these commands are missing, please install them. For the newest documentation on installation of LaTeX we recommend you look up the installation for your specific OS.

Install on Ubuntu 16.04

The easiest way to install it on ubuntu is to use the terminal and type in (make sure you have enough space):

```
sudo apt-get install texlive-full
```

One of the best editors for LaTeX is emacs as you can also do bibliography management with it and not just LaTeX. However, other editors are available including:

- Kile, TeXworks, JLatexEditor, Gedit LaTeX Plugin, TeXMaker

Please look up how to install them if you like to use them. TeXMaker is popular, However I find the combination of emacs and latexmk superior. TeXmaker is installed with:

```
sudo apt-get install texmaker
```

Other installations:

- kile is installed by default
- <https://www.tug.org/texworks/> (Works on ubuntu, Windows, OSX)

LaTeX for OSX

- <https://www.latex-project.org/get/>

LaTeX for Windows

- <https://www.latex-project.org/get/>

8.1.2 Online Services

ShareLaTeX

ShareLaTeX is an online, collaborative LaTeX editor that makes the creation, preview, and sharing of LaTeX documents easy through a web-based interface. Those that like to use latex, but do not have it installed on their computers may want to look at the following video:

Video: <https://youtu.be/PfhS0juQk8Y>

Video with cc: <https://www.youtube.com/watch?v=8IDCGTFXoBs>

ShareLaTeX not only allows you to edit online, but allows you to share your documents in a group of up to three. Licenses are available if you need more than three people in a team.

IU Licensed ShareLaTeX

At IU we has a license for the ShareLaTeX service available to School of Informatics and Computing and Engineering students, faculty, and staff only on the Bloomington campus.

You can create a free ShareLaTeX account but the free accounts have limitations. Adding your account to the IU license will give you access to advanced features, including unlimited sharing. It

will also allow GitHub integration. This however only works with the commercial github.com and not the IU Enterprise GitHub at github.iu.edu. As we require in our courses github.com you will be able to use it.

Please note that this license is only available to School of Informatics and Computing students, faculty, and staff on the Bloomington campus. Students must be enrolled in one of the SoIC degree programs on the Bloomington campus to be eligible. Students in other degree programs (even those taking SoIC classes) are not eligible.

If you want to use this service, please do and be aware of the following:

1. Go to the ShareLaTeX site and register. Please note that you **must** use either an @indiana.edu or @iu.edu email address when you register. If you use any other email address, we will not be able to add you to our site license. You are also required to use your IU passphrase as your ShareLaTeX password. Once you have registered, send an email to soichelp@indiana.edu asking to have your sharelatex account added to the IU license.
2. In your request, you must include the following: The IU email address you used when you registered (which must be in either the @indiana.edu or @iu.edu domain) A statement indicating that you understand that the ShareLaTeX service cannot be used for any sensitive data
3. Note that the ShareLaTeX service is **not** qualified for any sensitive data. This includes all data in the Critical, Restricted, and University-Internal categories as defined in the Data Classifications Page.

Overleaf

Overleaf.com is a collaborative latex editor. In its free version it has a very limited disk space. However it comes with a Rich text mode that allows you to edit the document in a preview mode. The free templates provided do not include ACM template, but you are allowed to use the OSA template.

Features of overleaf are documented at: <https://www.overleaf.com/benefits>

Paperia

We do not know where this service is located. However it offers similar services as ShareLatex and Overleaf.

- <https://papeeria.com/>

8.2 Basic LaTeX Elements

Often researchers may be initially overwhelmed with all the features that \LaTeX provides. However, it is much simpler than you initially believe. In Chapter ?? we introduced you towards using an article template. As a template is provided you can just look at the elements in that article and modify or copy them while adapting the content. Thus, it is more like filling out a form. You do not have to learn much and you can learn as you go. We are providing in this chapter some basic \LaTeX elements that will help you getting started quickly while serving you as a reminder what how to do certain things in \LaTeX .

8.2.1 Characters

\LaTeX is a command language and as such uses some special characters as part of the language. Thus if you want to use these characters either in your text or bibliography you need to be especially careful about. These characters include % \$ # _

Other than in hyperref links and urls you need to put a backslash in front of them. For example to print a % in the text you need to use:

```
\%
```

Furthermore the character " is not at all used as discussed in the next section.

8.2.2 Highlighting Text

Quotes are not written with the " character, but are embedded in two left single quotes and two right single quotes:

quote

<code>1 ``This is a quote''</code>	<code>“This is a quote”</code>
------------------------------------	--------------------------------

In many papers we see that the quote is misused while putting quotes around a word. However quotes are often just used to quote a text from another paper. Instead of using quotes authors may actually emphasize a word. \LaTeX has a special command for that using:

emphasize

<code>1 {\em this is emphasized}</code>	<code>this is emphasized</code>
---	---------------------------------

To write a text as bold (which should also be avoided as bold is typically used in section headers), you can use:

bold fett

<code>1 {\bf this is bold fett}</code>	<code>this is bold fett</code>
--	--------------------------------

8.2.3 Sections

\LaTeX provides a convenient mechanism to structure a paper with sections and subsections. This is achieved with the following commands:

```
\section{This is a Section}
\subsection{This is a Subsection}
\subsubsection{This is a Subsubsection}
```

Once you use one of these commands the next paragraph will start below the section command.

In addition you have the command:

```
\paragraph{This is a paragraph.}
```

The line is behind the paragraph heading

The command is special as it does not introduce a new line between the Heading and the next line even if you include empty lines

8.2.4 Empty Lines

Multiple empty lines will be reduced to a single empty line.

8.2.5 Itemize

Itemized lists can be written as follows:

itemize

```

1 \begin{itemize}
2   \item First item
3   \item Second item
4 \end{itemize}
```

- First item
- Second item

8.2.6 Enumerate

Enumerations can be written as:

enumerate

```

1 \begin{enumerate}
2   \item First item
3   \item Second item
4 \end{enumerate}
```

1. First item
2. Second item

8.2.7 Descriptions

Description lists can be written as:

description

```

1 \begin{description}
2   \item[Cloud:] My
      definition of a Cloud is
      over more than one line
      so we show the
      indentation.
3   \item[Big Data:] My
      definition of Big Data is
      also a long description.
5 \end{description}
```

Cloud: My definition of a Cloud is over more than one line so we show the indentation.

Big Data: My definition of Big Data is also a long description.

8.2.8 Images

Figures are extremely easy to handle by including them from source. We never worry about the placement as LaTeX does typically a very good job of doing this.:

In Figure \ref{F:flow} we show a black and white graph about

```
\begin{figure}[htb]
\includegraphics[width=1.0\columnwidth]{images/gregor.png}
\caption{A demonstration in the scalability of PDF images.}
```

```
\label{F:flow}
\end{figure}
```

Which results in the following:

In Figure 8.1 we show a black and white graph about

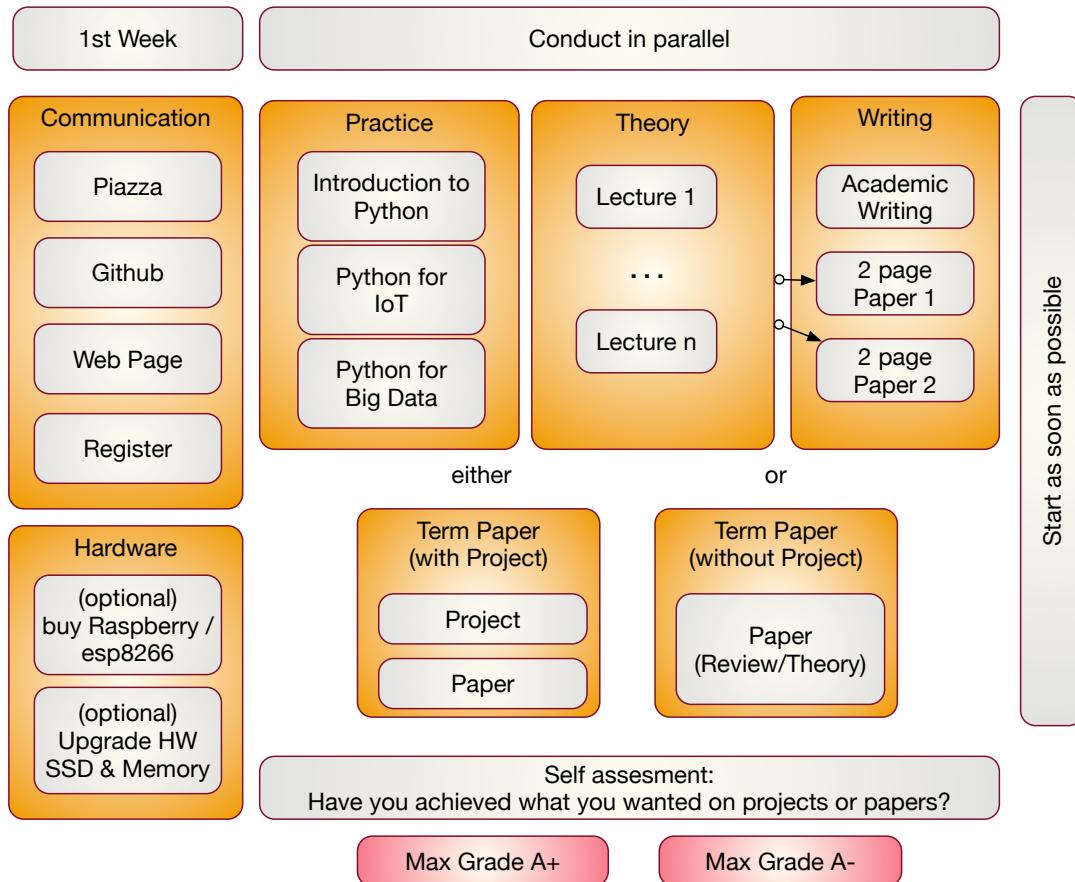


Figure 8.1: A demonstration in the scalability of PDF images.

Note that `las17graph` must be a label of a valid bibtex entry. This is needed if you have copied the image from elsewhere to avoid plagiarism. However, if you came up with the graph yourself than you do not need a citation.

We recommend that you place in your paper drafts all images at the which can be done with the `endfloat` package

This can be enabled if you include the following lines before `begin document` command:

```
\usepackage{endfloat}
\renewcommand{\efloatseparator}{\mbox{}}

\begin{document}
```

8.2.9 Tables

Latex tables are very similar to csv table. Thus you could with appropriate manipulation create tables from csv tables and use tools such as spreadsheet editors to manage your table. There are even packages that allow you to import the contents of csv tables directly into L^AT_EX.

In many cases you simply can start with an online table generator such as

- <https://www.tablesgenerator.com/>

For some tables you may want to rescale the width with

```
\resizebox{\textwidth}{!}{%
    ... PUT YOUR TABLE HERE ....
}
```

In other cases you may want to rotate the table which you can easily google for. In all cases use as figures, tables need to be in the `table` float environment. In contrast to figures captions are on the top. All tables must be referred to by `ref`. For more information in directly including csv tables see

- <http://mirror.utexas.edu/ctan/macros/latex/contrib/csvsimple/csvsimple.pdf>

However the format is very easy and can in most cases directly be included in latex as shown in Table 8.1.

```
\begin{table}[htb]
\caption{Table with elements}\label{T:elements}
\bigskip
\begin{center}
\begin{tabular}{ c c c }
column1 & column2 & column3 \\
\hline
\hline
element1 & element2 & element3 \\
element4 & element5 & element6 \\
element7 & element8 & element9 \\
\hline
\end{tabular}
\end{center}
\end{table}
```

Table 8.1: Table with elements

column1	column2	column3
element1	element2	element3
element4	element5	element6
element7	element8	element9

8.2.10 Labels

As we saw already for figures and tables it is recommended to use the `label` and `ref` commands to refer to figure or table numbers. This applies also to sections. Thus I can place a label after a section:

```
\section{Introduction}\label{S:introduction}
```

and write elsewhere in the paper:

```
As we showcased in Section \ref{S:introduction}
```

Furthermore to conveniently distinguish sections tables and figures, we use the prefix S T F followed by a colon for the label. This helps organizing your paper in case you have many labels.

8.2.11 Mathematics

One of the strength of LaTeX is the ability to write easily sophisticated mathematical expressions on paper with high quality. A good online resource is provided by the following online resource from which we have copied some examples:

- <https://en.wikibooks.org/wiki/LaTeX/Mathematics>

To activate them use

```
\usepackage{amsmath}
```

at the beginning of the document after the document class

Exponents are using the \wedge character:

exponents

```
1 $(a+b)^2 = a^2 + 2ab + b^{c+2}$
```

$$(a+b)^2 = a^2 + 2ab + b^{c+2}$$

Greek letters are referred to by their name proceeded by the slash:

greek

```
1 $ \alpha \beta \gamma \Gamma \pi \Pi \phi
```

$$\alpha\beta\gamma\Gamma\pi\Pi\phi$$

Limits can be written as follows:

limits

```
1 $ \lim_{x \rightarrow \infty} \exp(-x) = 0 $
```

$$\lim_{x \rightarrow \infty} \exp(-x) = 0$$

Fractions are indicated by the `\frac` command, and binomials by `\binom`:

fraction

```
1 $ \frac{n!}{k!(n-k)!} = \binom{n}{k}
```

$$\frac{n!}{k!(n-k)!} = \binom{n}{k}$$

Matrices can be created as follows:

matrix

```

1 $ A_{m,n} =
2 \begin{pmatrix}
3 a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\
4 a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\
5 \vdots & \vdots & \ddots & \vdots \\
6 a_{m,1} & a_{m,2} & \cdots & a_{m,n} \\
7 \end{pmatrix}

```

$$A_{m,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

8.3 Advanced topics

8.3.1 ACM and IEEE Proceedings Format



Figure 8.2: The look of the ACM and IEEE format templates

- <http://www.acm.org/publications/proceedings-template>
- https://www.ieee.org/conferences_events/conferences/publishing/templates.html

8.3.2 Generating and Managing Images

To produce high quality images the programs PowerPoint and omnigraffle on OSX are recommended. When using powerpoint please keep the image ratio to 4x3 as they produce nice size graphics which you also can use in your presentations. When using other ratios they may not fit in presentations and thus you may increase unnecessarily your work. We do not recommend vizio as it is not universally available and produces images that in case you have to present them in a slide presentation does not easily reformat if you do not use 4x3 aspect ratio.

Naturally, graphics should be provided in SVG or PDF format so they can scale well when we look at the final PDF. Including PNG, gif, or jpeg files often do not result in the necessary resolution or the files become real big. For this reason we for example can also not recommend tools such as tablaeu as they do not provide proper exports to high quality publication formats. For interactive display such tool may be good, but for publications it produces inferior formatted images.

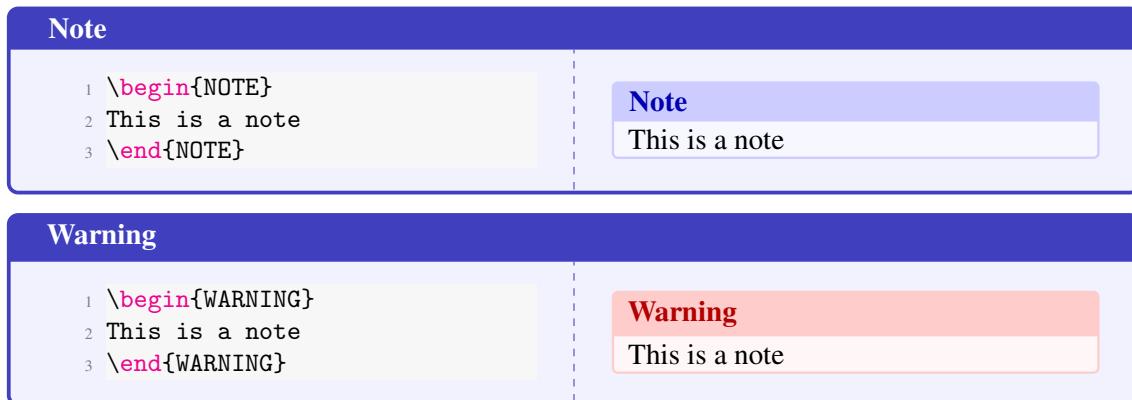
We recommend that all images be stored into a folder called images in the same directory where your \LaTeX main document resides.

8.3.3 Colored Boxes

The package `tcolorbox` provides sophisticated support to include color boxes. Together with the new environment we can create nice add ons to for example include notes.

- <http://osl.ugr.es/CTAN/macros/latex/contrib/tcolorbox/tcolorbox.pdf>

We have provided in this document notes as follows



8.3.4 Slides

Slides are best produced with the seminar package:

```

\documentclass{seminar}

\begin{slide}

Hello World on slide 1

\end{slide}

The text between slides is ignored

\begin{slide}

Hello World on slide 2

\end{slide}

```

However, in case you need to have a slide presentation we recommend you use ppt. Just paste and copy content from your PDF or your \LaTeX source file into the ppt.

8.3.5 LaTeX vs. X

We will refrain from providing a detailed analysis on why we use LaTeX in many cases versus other technologies. In general, we find that LaTeX:

- is incredibly stable
- produces high-quality output
- is platform independent
- has lots of templates
- has been around for many years so it works well
- removes you from the pain of figure placements
- focusses you on content rather than the appearance of the paper
- integrates well with code repositories such as git to write collaborative papers.
- has superior bibliography integration
- has a rich set of tools that make using LaTeX easier
- authors do not play with layouts much so papers in a format are uniform

In case you need a graphical view to edit LaTeX or LateX exportable files you also find AucTeX and Lyx.

Word

Word is arguably available to many, but if you work on Linux you may be out of luck. Also Word often focusses not on structure of the text but on its appearance. Many students abuse Word and the documents in Word become a pain to edit with multiple users. Recently Microsoft has offered online services to collaborate on writing documents in groups which work well. Integration with bibliography managers such as endnote or Mendeley is possible.

However, we ran into issues whenever we use word:

- Word tends sometimes to crash for unknown reasons and we lost a lot of work
- Word has some issues with the bibliography managers and tends to crash sometimes for unknown reasons.
- Word is slow with integration to large bibliographies.
- Figure placement in Word in some formats is a disaster and you will spend many hours to correct things just to find out that if you make small changes you have to spend additional many hours to get used to the new placement. We have not yet experienced a word version where we have not lost images. Maybe that has changed, so let us know

However, we highly recommend the collaborative editing features of Word that work on a paragraph and not letter level. Thus saving is essential so you do not block other people from editing the paragraph.

Google Docs

Unfortunately, many useful features got lost in the new google docs. However, it is great to collaborate quickly online, share thoughts and even write your latex documents together if you like (just copy your work in a file offline and use latex to compile it ;-)

The biggest issue we have with Google Docs is that it does not allow the support of 2 column formats, that the bibliography integration is non-existent and that paste and copy from web pages and images encourages unintended plagiarism when collecting information without annotations (LaTeX and Word are prone to this too, but we found from experience that it tends to happen more with Google docs users).

A Place for Each

When looking at the tools we find a place for each:

Google docs: Short meeting notes, small documents, quick online collaborations to develop documents collaboratively at the same time.

Word: Available to many, supports 2 column format, supports paragraph based collaborative editing, Integrates with bibliography managers.

LaTeX: Reduces failures, great offline editing, superior bibliography management, superior image placement, runs everywhere. Great collaborative editing with sharelatex, allows easy generation of proceedings written by hundreds of people with shared index.

The best choice for your class: LaTeX

8.4 Editing

8.4.1 Emacs

The text editor emacs provides a great basis for editing TeX and LaTeX documents. Both modes are supported. In addition there exists a color highlight module enabling the color display of LaTeX and TeX commands. On OSX aquemacs and carbon emacs have build in support for LaTeX. Spell checking is done with flyspell in emacs.

Aquamacs

Aquamacs is an editor based on GNU Emacs that runs on OSX and integrates with the OSX desktop. This is for many the preferred editor on OSX for \LaTeX .

<http://aquamacs.org>

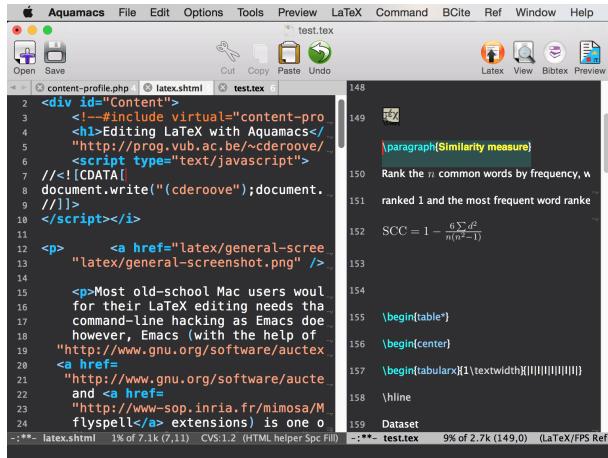


Figure 8.3: Aquamacs

8.4.2 Vi/Vim

Another popular editor is vi or vim. It is less feature rich but many programmers are using it. As it can edit ASCII text you can edit LaTeX. With the LaTeX add-ons to vim, vim becomes similar powerful while offering help and syntax highlighting for LaTeX as emacs does. (The authors still prefer emacs)

8.4.3 TeXshop

Other editors such as TeXshop are available which provide a more integrated experience. However, we find them at times to stringent and prefer editors such as emacs.

8.4.4 LyX

We have made very good experiences with Lyx. You must assure that the team you work with uses it consistently and that you all use the same version.

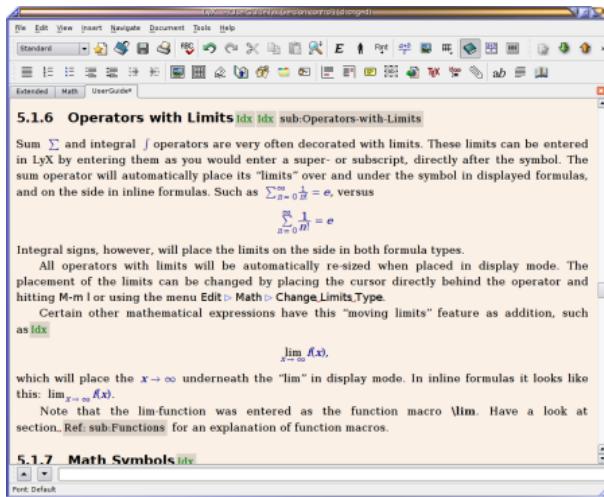


Figure 8.4: Lyx

Using the ACM templates is documented here:

- <https://wiki.lyx.org/Examples/AcmSiggraph>

On OSX it is important that you have a new version of LaTeX and Lyx installed. As it takes up quite some space, you may want to delete older versions. The new version of LyX comes with the acmsigplan template included. However on OSX and other platforms the .cls file is not included by default. However the above link clearly documents how to fix this.

8.4.5 WYSIWYG locally

We have found that editors such as Lyx and Auctex provide very good WYSIWYG alike features. However, we found an even easier way while using skim, a pdf previewer, in conjunction with emacs and latexmk. This can be achieved while using the following command assuming your latex file is called ‘report.tex’:

```
latexmk -pvc -view=pdf report
```

This command will update your pdf previewer (make sure to use skim) whenever you edit the file report.tex and save it. It will maintain via skim the current position, thus you have a real great way of editing in one window, while seeing the results in the other.

Skim can be found at: <http://skim-app.sourceforge.net/>

8.4.6 Markdown and \LaTeX

It may come as a surprise to many that one can actually write simple \LaTeX documents also in markdown Syntax or mix section written in markdown while others are written in \LaTeX . To do so all you ahve to do is place the markdown text in a separate file. Let us call the file content.md which has the following lines included in it:

```
# Section

* item a
* item b
```

Obviously, we would have to convert this to \LaTeX . Luckily there is a very useful program called *pandoc* that does this for you. YOu could make the translation in the shell, but you could also make the translation locally on your computer while allowing \LaTeX to start up external programs. This is achieved with the *write18* command and allowing \LaTeX explicitly to call external programs. Please inspect the following latex file that includes a template on how to do this. We assume the file is called markdown.tex for our example.

```
\documentclass{article}

\usepackage{graphicx}
\newcommand{\tightlist}{}{}

\begin{document}
\immediate\write18{pandoc content.md -o content.tex}

\input{content}

\end{document}
```

Now to generate the PDF we simply have to call the following command that include the *-shell-escape* flag to allow the execution of *write18* embedded commands:

```
pdflatex -shell-escape markdown-test
```

The output will be *markdown.pdf* with the content from the markdown file translated. Doing this naturally allows you to write large portions in markdown and automatically include them in your \LaTeX document. Hence, you can use editors such as Macdown to initially work in semi WYSIWYG mode and do fairly straight forward edition. Naturally the same can be done in RST. Naturally the most elementary features are supported. For more sophisticated features, please use \LaTeX directly.

8.4.7 Including RST into \LaTeX

content.rst:

```
Section
-----
* item a
* item b
```

sample.tex:

```
\documentclass{article}

\usepackage{graphicx}
\newcommand{\tightlist}{}{}
```

```
\begin{document}
\immediate\write18{pandoc content.rst -o content.tex}

\input{content}

\end{document}
```

8.4.8 pyCharm

TODO: comment on how we can use pycharm for editing and what the limitations are.

8.4.9 MSWord

it is possible to use Word.

be careful with

8.5 The LaTeX Cycle

To create a PDF file from latex yo need to generate it following a simple development and improvement cycle.

First, Create/edit ASCII source file with `file.tex` file:

```
emacs file.tex
```

Create/edit bibliography file:

```
jabref refs.bib
```

Create the PDF:

```
pdflatex file
bibtex file
pdflatex file
pdflatex file
```

View the PDF:

```
open file
```

It not only showcases you an example file in ACM 2 column format, but also integrates with a bibliography. Furthermore, it provides a sample Makefile that you can use to generate view and recompile, or even autogenerate. A compilation would look like:

```
make
make view
```

If however you want to do things on change in the tex file you can do this automatically simply with:

```
make watch
```

for make watch its best to use skim as pdf previewer

8.6 Tips

Including figures over two columns:

- <http://tex.stackexchange.com/questions/30985/displaying-a-wide-figure-in-a-two-column-document>
- positioning figures with textwidth and columnwidth https://www.sharelatex.com/learn/Positioning_images_and_tables
- An organization as the author. Assume the author is National Institute of Health and want to have the author show up, please do:


```
key= {National Institute of Health},
author= {{National Institute of Health}},
```

 Please note the {{ }}}
- words containing ‘fi’ or ‘ffi’ showing blank places like below after recompiling it: find as nd efficiency as e ciency
 You copied from word or PDF ff which is actually not an ff, but a condensed character, change it to ff and ffi, you may find other such examples such as any non ASCII character. A degree is for example another common issue in data science.
- do not use | & and other latex characters in bibtex references, instead use , and the word and
- If you need to use _ it is _ but if you use urls leave them as is
- We do recommend that you use sharelatex and jabref for writing papers. This is the easiest solution and beats in many cases MSWord as you can focus on writing and not on formatting.

8.6.1 Colorful Output

Instead of using pdflatex, you can also install pydflatex that provides a convenient wrapper and colorizes the output while eliminating a lot of warnings that you may initially not want to deal with. To install it please use:

```
pip install blessings
pip install -e "git+https://github.com/olivierverdier/pydflatex#egg=pydflatex"
```

You can see the manual page with

```
pydflatex --help

usage: usage: pydflatex [options] texfile1

Compile a tex file with pdflatex and make the auxiliary files invisible. Note
that the '.tex' extension may be omitted

positional arguments:
  tex path          path to tex file

optional arguments:
  -h, --help        show this help message and exit
  -o, --open        view the pdf file(s) in a pdf viewer.
  -k, --continue   continue on error
  -w, --with-warning  do not suppress common warnings
  -v, --verbose    Verbose output for debugging
  -p, --plain      No coloured output
  -x, --xetex      Use XeLaTeX engine
  -l, --log-parsing Only parse log
  -t, --typesetting Only typeset
```

8.6.2 References

Latex Sheet: <https://wch.github.io/latexsheet/latexsheet.pdf>

Latex Short: <http://tug.ctan.org/info/lshort/english/lshort.pdf>

Wikibook: <https://en.wikibooks.org/wiki/LaTeX>

Wikibook (PDF) : <https://upload.wikimedia.org/wikipedia/commons/2/2d/LaTeX.pdf>

Links to books: <https://latexforhumans.wordpress.com/2008/10/11/the-best-guides-to-latex/>

Links to books: <https://www.latex-project.org/help/books/>

LaTeX2e: The [LaTeX Reference Manual](#) provides a good introduction to Latex.

- LaTeX Users and Reference Guide, by Leslie Lamport https://www.amazon.com/LaTeX-Document-Preparation-System-2nd/dp/0201529831/ref=sr_1_2?s=books&ie=UTF8&qid=1507114870&sr=1-2&keywords=lamport
- LaTeX an Introduction, by Helmut Kopka https://www.amazon.com/Guide-LaTeX-4th-Helmut-Kopka/dp/0321173856/ref=pd_lpo_sbs_14_t_0?_encoding=UTF8&psc=1&refRID=2BB4APDFEX34A4JM65ZB
- The LaTeX Companion, by Frank Mittelbach <https://www.amazon.com/LaTeX-Companion-Techniques-Computer-Typesetting/dp/0201362996>



9. Managing Bibliographies

F section/doc/bibtex.tex

9.0.1 Integrating Bibliographies

Bibtex integrates very well with L^AT_EX. There are numerous pre-formatted bibliography styles available. It includes also styles for ACM and IEEE bibliographies. For the ACM style we recommend that you replace abrv.bst with abrvurl.bst, add hyperref to your usepackages so you can also display URLs in your citations:

```
\bibliographystyle{IEEEtran}
\bibliography{references.bib}
```

Then you have to run latex and bibtex in the following order:

```
latex file
bibtex file
latex file
latex file
```

or simply call make from our makefile.

Indiana University

At IU you are required to use our template.

The reason for the multiple execution of the latex program is to update all cross-references correctly. In case you are not interested in updating the library every time in the writing progress just postpone it till the end. Missing citations are viewed as [?].

Two programs stand out when managing bibliographies: emacs and jabref:

- <http://www.jabref.org/>

Other programs such as Mendeley, Zotero, and even endnote integrate with bibtex. However their

support is limited, so we recommend that you just use jabref. Furthermore its free and runs on all platforms.

jabref

Jabref is a very simple to use bibliography manager for LaTeX and other systems. It can create a multitude of bibliography file formats and allows upload in other online bibliography managers.

- Installation: Go to <http://www.jabref.org/> and click download
- Video: <https://youtu.be/cMtYOHCHZ3k>
- Video with cc: <https://www.youtube.com/watch?v=QVbifcLgMic>

jabref and MSWord

According to others it is possible to integrate jabref references directly into MSWord. This has been conducted so far however only on a Windows computer.

We have not tried this ourselves, but give it as a potential option.

Here are the steps the need to be done:

1. Create the Jabref bibliography just like in presented in the Jabref video
2. After finishing adding your sources in Jabref, click File -> export
3. Name your bibliography and choose MS Office 2007 (*.xml) as the file format. Remember the location of where you saved your file.
4. Open up your word document. If you are using the ACM template, go ahead and remove the template references listed under Section 7. References
5. In the MS Word ribbon choose ‘References’
6. Choose ‘Manage Sources’
7. Click ‘Browse’ and locate/select your Jabref xml file
8. You should now see your references appear in the left side window. Select the references you want to add to your document and click the ‘copy’ button to move them from the left side window to the right window.
9. Click the ‘Close’ button
10. In the MS Word Ribbon, select ‘Bibliography’ under the References tab
11. Click ‘Insert Bibliography’ and your references should appear in the document
12. Ensure references are of Style: IEEE. Styles are located in the References tab under ‘Manage Sources’

As you can see there is significant effort involve, so we do recommend you use LaTeX as you can focus there on content rather than dealing with complex layout decisions. This is especially true, if your papers have figures or tables, or you need to add references.

9.1 Entry types

In this section we will explain how to find and properly generate bibliographic entries. We are using bibtex for this as it is easy to use and generates reasonable entries that can be included in papers. What we like to achieve in this section is not to just show you a final entry, but to document the process on how that entry was derived. This will allow you to replicate or learn from the process to apply to your own entries.

We will address a number of important entry types which includes:

- wikipedia entries

- github entries
- books
- articles in a scientific journal
- articles in a conference
- articles in magazines (non scientific)
- blogs

9.1.1 Source code References

We will learn how to cite a source code from a publicly hosted repository. Such repositories are frequently used and include, for example github, bitbucket, sourcefore, or your Universities code repository as long as it is publicly reachable. As changes can occur on these repositories, it is important that the date of access is listed in the entry or even the release version of the source code.

Let us without bias chose a random source code entry that has been contributed by a student as follows:

```
@Misc{gonzalez_2015,
  Title = {Buildstep},
  Author = {Gonzalez, Jose and Lindsay, Jeff},
  HowPublished = {Web Page},
  Month = {Jul},
  Note = {Accessed: 2017-1-24},
  Year = 2015,
  Key = {www-buildstep},
  Url = {https://github.com/progium/buildstep}
}
```

Is this entry correct? Let us analyse.

Entry type Misc

First, it seems appropriate to use a `@misc` entry. We correctly identify this is a misc entry as it is online available. More recent version of bibtex include also the type `@online` for it. However, in order to maintain compatibility to older formats we chose simply Misc here and if we really would need to we could replace it easily

Label

Typically the Label should contain 3 letters from an author name, short year and the short name of the publication to provide maximum information regarding the publication. Underscores need to be replaced by dashes or removed. However as this is a github repository it is better to integrate this into the label. Hence, we simply use the github-projectname (in our case github-buildstep, out of convention we only use lower case letters).

Author

Unless the last name contains spaces, it should be first name followed by the last name with multiple authors separated with “and”.

Key

In this case the key field can be removed as the entry has an author field entry. If there was no author field, we could use key to specify the alphabetical ordering based on the specified key. Note that a key is not the label. In fact in our original entry the key field was wrongly used and the

student did not understand that the key is used for sorting.

Howpublished

Since the source is a github project repository, the howpublished field shall hold the value {Code Repository} rather than a web page. If the url specified was a normal webpage, the {Web Page} entry would be valid.

Month

The lowercase month is, used for international notation since months are not capitalized in some other languages.

Owner

In class we introduced the convention to put the student HID in it. If multiple students contributed, add them with space separation.

Accessed

As we do not yet typically an accessed field, we simply include it in the note field. This is absolutely essential as code can change and when we read the code we looked at a particular snapshot in time. In addition it is often necessary to record the actual version of the code. Typically this can also be done with the month and year field while relying on a release date

Final Entry

Filling out as many fields as possible with information for this entry we get:

```
@Misc{github-buildstep,
    Title = {Buildstep},
    Author = {Jose Gonzalez and Jeff Lindsay},
    HowPublished = {Code Repository},
    Year = {2015},
    Month = jul,
    Note = {Accessed: 2017-1-24},
    Url = {https://github.com/progium/buildstep},
    Owner = {S17-I0-3025},
}
```

We are using the release date in the year and month field as this project uses this for organizing releases. However, other project may have release versions so you would have in addition to using the data also to include the version in the note field such as:

```
Note = {Version: 1.2.3, Accessed: 2017-1-24},
```

All those that helped should add your HID to this entry with a space separated from each other

9.1.2 Researching proper bibtex entries

Article in a journal

Many online bibtex entries are wrong or incomplete. Often you may find via google a bibtex entry that may need some more research. Lets assume your first google query returns a publication and you cite it such as this:

```
@Unpublished{unpublished-google-sawzall,
    Title = {{Interpreting the Data: Parallel Analysis with Sawzall}},
```

```

Author = {{Rob Pike, Sean Dorward, Robert Griesemer, Sean Quinlan}},
Note = {accessed 2017-01-28},
Month = {October},
Year = {2005},
Owner = {for the purpose of this discussion removed},
Timestamp = {2017.01.31}
}

```

Could we improve this entry to achieve your best? We observe:

1. The author field has a wrong entry as the , is to be replaced by an and.
2. The author feild has authors and thus must not have a {{ }}
3. The url is missing, as the simple google search actually finds a PDF document.

Let us investigate a bit more while searching for the title. We find

- A) https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwj_ytSA-PDRAhUH8IMKHaomC-oQFggaMAA&url=https%3A%2F%2Fresearch.google.com%2Farchive%2Fsawzall-sciprog.pdf&usg=AFQjCNHSSfKBwbxVAVPQ0td4rTjitKucpA&sig2=vbiVzi36B3gGFjIzlUKBDA&bvm=bv-1
- B) <https://research.google.com/pubs/pub61.html>
- C) <http://dl.acm.org/citation.cfm?id=1239658>

Let us look at A)

As you can see from the url this is actually some redirection to a google web page which probably is replaced by B as its from google research. So let us look at B)

Now when you look at the link we find the url <https://research.google.com/archive/sawzall-sciprog.pdf> which redirects you to the PDF paper.

When we go to B) we find surprisingly a bibtex entry as follows:

```

@article{61,
    title = {Interpreting the Data: Parallel Analysis with Sawzall},
    author = {Rob Pike and Sean Dorward and Robert Griesemer and Sean Quinlan},
    year = 2005,
    URL = {https://research.google.com/archive/sawzall.html},
    journal = {Scientific Programming Journal},
    pages = {277--298},
    volume = {13}
}

```

Now we could say lets be satisfied, but C) seems to be even more interesting as its from a major publisher. So lats just make sure we look at C)

If you go to C, you find under the colored box entitled Tools and Resources a link called **bibtex**. Thus it seems a good idea to click on it. This will give you:

```

@article{Pike:2005:IDP:1239655.1239658,
    author = {Pike, Rob and Dorward, Sean and Griesemer, Robert and Quinlan, Sean},
    title = {Interpreting the Data: Parallel Analysis with Sawzall},
    journal = {Sci. Program.},
    issue_date = {October 2005},
    volume = {13},
    number = {4},
    month = oct,
    year = {2005},
    issn = {1058-9244},
    pages = {277--298},
    numpages = {22},
    url = {http://dx.doi.org/10.1155/2005/962135},
    doi = {10.1155/2005/962135},
}

```

```

acmid = {1239658},
publisher = {IOS Press},
address = {Amsterdam, The Netherlands, The Netherlands},
}

```

Now we seem to be at a position to combine our search result as neither entry is sufficient. As the doi number properly specifies a paper (look up what a doi is) we can replace the url with one that we find online, such as the one we found in A) Next we see that all field sin B are already covered in C, so we take C) and add the url. Now as the label is great and uniform for ACM, but for us a bit less convenient as its difficult to remember, we just change it while for example using authors, title, and year information. lets also make sure to do mostly lowercase in the label just as a convention. Thus our entry looks like:

```

@article{pike05swazall,
author = {Pike, Rob and Dorward, Sean and Griesemer, Robert and Quinlan, Sean},
title = {Interpreting the Data: Parallel Analysis with Sawzall},
journal = {Sci. Program.},
issue_date = {October 2005},
volume = {13},
number = {4},
month = oct,
year = {2005},
issn = {1058-9244},
pages = {277--298},
numpages = {22},
url = {https://research.google.com/archive/sawzall-sciprog.pdf},
doi = {10.1155/2005/962135},
acmid = {1239658},
publisher = {IOS Press},
address = {Amsterdam, The Netherlands, The Netherlands},
}

```

As you can see properly specifying a reference takes multiple google queries and merging of the results you find from various returns. As you still have time to correct things I advise that you check your references and correct them. If the original reference would have been graded it would have been graded with a “fail” instead of a “pass”.

9.1.3 Article in a conference proceedings

Lets look at a second obvious example that needs improvement:

```

@InProceedings{wettinger-any2api,
Title          = {Any2API - Automated APIfication},
Author         = {Wettinger, Johannes and
                  Uwe Breitenb\"ucher
                  and Frank Leymann},
Booktitle      = {Proceedings of the 5th International
                  Conference on Cloud Computing and
                  Services Science},
Year           = {2015},
Pages          = {475486},
Publisher      = {SciTePress},
ISSN           = {2326-7550},
Owner          = {S17-I0-3005},
Url            = {https://pdfs.semanticscholar.org/1cd4/4b87be8cf68ea5c4c642d38678a7b40a8}
}

```

As you can see this entry seems to define all required fields, so we could be tempted to stop here.

But its good to double check. Lets do some queries against ACM, . and google scholar, so we just type in the title, and if this is in a proceedings they should return hopefully a predefined bibtex record for us.

Lets query:

```
google: googlescholar Any2API Automated APIfication
```

We get:

- https://scholar.google.de/citations?view_op=view_citation&hl=en&user=j6lIXt0AAAAJ&citation_for_view=j6lIXt0AAAAJ:8k81kl-MbHgC

On that page we see [Cite](#)

So we find a PDF at <https://pdfs.semanticscholar.org/1cd4/4b87be8cf68ea5c4c642d38678a7b40a86de.pdf>

Lets click on this and the document includes a bibtex entry such as:

```
@inproceedings{Wettinger2015,
    author= {Johannes Wettinger and Uwe Breitenb\{\"u\}cher and Frank
             Leymann},
    title = {Any2API - Automated APIfication},
    booktitle = {Proceedings of the 5th International Conference on Cloud
                Computing and Service Science (CLOSER)},
    year = {2015},
    pages = {475--486},
    publisher = {SciTePress}
}
```

Now lets add the URL and owner:

```
@inproceedings{Wettinger2015,
    author= {Johannes Wettinger and Uwe Breitenb\{\"u\}cher and Frank
             Leymann},
    title = {Any2API - Automated APIfication},
    booktitle = {Proceedings of the 5th International Conference on Cloud
                Computing and Service Science (CLOSER)},
    year = {2015},
    pages = {475--486},
    publisher = {SciTePress},
    url ={https://pdfs.semanticscholar.org/1cd4/4b87be8cf68ea5c4c642d38678a7b40a86de.pdf},
    owner = {S17-I0-3005},
}
```

Should we be satisfied? No, even our original information we gather provided more information. So lets continue. Lets googlesearch different queries with ACM or IEEE and the title. When doing the IEEE in the example we find an entry called

[dlp: Frank Leyman](#)

Lets look at it and we find two entries:

```
@inproceedings{DBLP:conf/closer/WettingerBL15,
    author = {Johannes Wettinger and
              Uwe Breitenb\{\"u\}cher and
              Frank Leymann},
    title = {{\{ANY2API\}} - Automated APIfication - Generating APIs for Executables
             to Ease their Integration and Orchestration for Cloud Application
             Deployment Automation},
    booktitle = {{\{CLOSER\}} 2015 - Proceedings of the 5th International Conference on
                Cloud Computing and Services Science, Lisbon, Portugal, 20-22 May,
```

```

    2015.},
  pages      = {475--486},
  year       = {2015},
  crossref   = {DBLP:conf/closer/2015},
  url        = {http://dx.doi.org/10.5220/0005472704750486},
  doi         = {10.5220/0005472704750486},
  timestamp  = {Tue, 04 Aug 2015 09:28:21 +0200},
  biburl     = {http://dblp.uni-trier.de/rec/bib/conf/closer/WettingerBL15},
  bibsource   = {dblp computer science bibliography, http://dblp.org}
}

@proceedings{DBLP:conf/closer/2015,
  editor      = {Markus Helfert and
                 Donald Ferguson and
                 V{\'e}ctor M{\'a}ndez Mu{\~n}oz},
  title       = {{\{}CLOSER 2015 - Proceedings of the 5th International Conference on
                 Cloud Computing and Services Science, Lisbon, Portugal, 20-22 May,
                 2015{\}}},
  publisher   = {SciTePress},
  year        = {2015},
  isbn        = {978-989-758-104-5},
  timestamp  = {Tue, 04 Aug 2015 09:17:34 +0200},
  biburl     = {http://dblp.uni-trier.de/rec/bib/conf/closer/2015},
  bibsource   = {dblp computer science bibliography, http://dblp.org}
}

```

So lets look at the entry and see how to get a better one for our purpose to combine them. When using jabref, you see optional and required fields, we want to add as many as possible, regardless if optional or required, so Lets do that (I write here in ASCII as easier to document:

```

@InProceedings{,
  author = {},
  title = {},
  OPTcrossref = {},
  OPTkey = {},
  OPTbooktitle = {},
  OPTyear = {},
  OPTeditor = {},
  OPTvolume = {},
  OPTnumber = {},
  OPTseries = {},
  OPTpages = {},
  OPTmonth = {},
  OPTaddress = {},
  OPTorganization = {},
  OPTpublisher = {},
  OPTnote = {},
  OPTannote = {},
  url = {}
}

```

So lets copy and fill out the **form** from our various searches:

```

@InProceedings{Wettinger2015any2api,
  author      = {Johannes Wettinger and
                 Uwe Breitenb\"ucher and
                 Frank Leymann},
  title       = {{\{}ANY2API - Automated APIfication - Generating APIs for Executables
                 to Ease their Integration and Orchestration for Cloud Application
                 Deployment Automation{\}}},
  booktitle   = {{\{}CLOSER 2015 - Proceedings of the 5th International Conference on
                 Cloud Computing and Services Science{\}}},

```

```

year = {2015},
editor = {Markus Helfert and
          Donald Ferguson and
          V{\'{i}}ctor M{\'{e}}ndez Mu{\'{n}}oz},
publisher = {SciTePress},
isbn = {978-989-758-104-5},
pages = {475--486},
month = {20-22 May},
address = {Lisbon, Portugal},
doi = {10.5220/0005472704750486},
url ={https://pdfs.semanticscholar.org/1cd4/4b87be8cf68ea5c4c642d38678a7b40a86de.pdf},
owner = {S17-I0-3005},
}

```

9.1.4 What are the differnt entry types and fields

We were asked what are the different entry types and fields, so we did a google query and found the following useful information. please remember that we also have fields such as doi, owner, we will add status ={pass/fail} at time of grading to indicate if the reference passes or fails. We may assign this to you so you get familiar with the identification if a refernce is ok or not.

Please see <https://en.wikipedia.org/wiki/BibTeX>

9.1.5 InProceedings

Please fill out

```

@InProceedings{,
    author = {},
    title = {},
    OPTcrossref = {},
    OPTkey = {},
    OPTbooktitle = {},
    OPTyear = {},
    OPTeditor = {},
    OPTvolume = {},
    OPTnumber = {},
    OPTseries = {},
    OPTpages = {},
    OPTmonth = {},
    OPTaddress = {},
    OPTorganization = {},
    OPTpublisher = {},
    OPTnote = {},
    OPTannote = {},
    url = {}
}

@inproceedings{vonLaszewski15tas,
    author = {DeLeon, Robert L. and Furlani, Thomas R. and Gallo,
              Steven M. and White, Joseph P. and Jones, Matthew
              D. and Patra, Abani and Innus, Martins and Yearke,
              Thomas and Palmer, Jeffrey T. and Sperhac, Jeanette
              M. and Rathsam, Ryan and Simakov, Nikolay and von
              Laszewski, Gregor and Wang, Fugang},
    title = {{TAS View of XSEDE Users and Usage}},
    booktitle = {Proceedings of the 2015 XSEDE Conference: Scientific
                Advancements Enabled by Enhanced
                Cyberinfrastructure},
}

```

```

series = {XSEDE '15},
year = 2015,
isbn = {978-1-4503-3720-5},
location = {St. Louis, Missouri},
pages = {21:1--21:8},
articleno = 21,
numpages = 8,
url = {http://doi.acm.org/10.1145/2792745.2792766},
doi = {10.1145/2792745.2792766},
acmid = 2792766,
publisher = {ACM},
address = {New York, NY, USA},
keywords = {HPC, SUPReMM, TAS, XDMoD, XSEDE usage, XSEDE users},
}

```

9.1.6 TechReport

Please fill out

```

@TechReport{,
author = {},
title = {},
institution = {},
year = {},
OPTkey = {},
OPTtype = {},
OPTnumber = {},
OPTaddress = {},
OPTmonth = {},
OPTnote = {},
OPTannote = {},
url = {}
}

@TechReport{las05exp,
title = {{The Java CoG Kit Experiment Manager}},
Author = {von Laszewski, Gregor},
Institution = {Argonne National Laboratory},
Year = 2005,
Month = jun,
Number = {P1259},
url = {https://laszewski.github.io/papers/vonLaszewski-exp.pdf}
}

```

9.1.7 Article

Please fill out

```

@Article{,
author = {},
title = {},
journal = {},
year = {},
OPTkey = {},
OPTvolume = {},
OPTnumber = {},
OPTpages = {},
OPTmonth = {},
OPTnote = {},
OPTannote = {}
}

```

```

    url = {}
}

@Article{las05gridhistory,
  title = {{The Grid-Idea and Its Evolution}},
  author = {von Laszewski, Gregor},
  journal = {Journal of Information Technology},
  year = 2005,
  month = jun,
  number = 6,
  pages = {319-329},
  volume = 47,
  doi = {10.1524/jitit.2005.47.6.319},
  url = {https://laszewski.github.io/papers/vonLaszewski-grid-idea.pdf}
}

```

9.1.8 Proceedings

Please fill out

```

@Proceedings{,
  title = {},
  year = {},
  OPTkey = {},
  OPTbooktitle = {},
  OPTeditor = {},
  OPTvolume = {},
  OPTnumber = {},
  OPTseries = {},
  OPTaddress = {},
  OPTmonth = {},
  OPTorganization = {},
  OPTpublisher = {},
  OPTnote = {},
  OPTannote = {},
  url = {}
}

@Proceedings{las12fedcloud-proc,
  title = {{FederatedClouds '12: Proceedings of the 2012
            Workshop on Cloud Services, Federation, and the 8th
            Open Cirrus Summit}},
  year = 2012,
  address = {New York, NY, USA},
  editor = {vonLaszewski, Gregor and Robert Grossman and Michael
            Kozuch and Rick McGeer and Dejan Milojevic},
  publisher = {ACM},
  ISBN = {978-1-4503-1754-2},
  location = {San Jose, California, USA},
  url =
    {http://dl.acm.org/citation.cfm?id=2378975&picked=prox&cfid=389635474&cftoken=32712}
}

```

9.1.9 Wikipedia Entry

Please fill out

```

@Misc{,
  OPTkey = {},
  OPTauthor = {},
  OPTtitle = {},

```

```

OPThowpublished = {},
OPTmonth = {},
OPTyear = {},
OPTnote = {},
OPTannote = {},
url = {}
}

@Misc{www-ode-wikipedia,
    Title = {Apache ODE},
    HowPublished = {Web Page},
    Note = {Accessed: 2017-2-11},
    Key = {Apache ODE},
    Url = {https://en.wikipedia.org/wiki/Apache_ODE}
}

```

9.1.10 Blogs

Please fill out

```

@Misc{},
    OPTkey = {},
    OPTauthor = {},
    OPTtitle = {},
    OPThowpublished = {},
    OPTmonth = {},
    OPTyear = {},
    OPTnote = {},
    OPTannote = {},
    OPTurl = {}
}

@Misc{www-clarridge-discoproject-blog,
    title = {Disco - A Powerful Erlang and Python Map/Reduce
              Framework},
    author = {Clarridge, Tait},
    howpublished = {Blog},
    month = may,
    note = {Accessed: 25-feb-2017},
    year = 2014,
    url = {http://www.taitclarridge.com/techlog/2014/05/disco-a-powerful-erlang-and-python-mapreduce-f
}

```

9.1.11 Web Page

Please fill out

```

@Misc{},
    OPTkey = {},
    OPTauthor = {},
    OPTtitle = {},
    OPThowpublished = {},
    OPTmonth = {},
    OPTyear = {},
    OPTnote = {},
    OPTannote = {},
    url = {}
}

@Misc{www-cloudmesh-classes,
    OPTkey = {},

```

```

author = {von Laszewski, Gregor},
title = {Cloudmesh Classes},
howpublished = {Web Page},
OPTmonth = {},
OPTyear = {},
OPTnote = {},
OPTannote = {},
url = {https://cloudmesh.github.io/classes/}
}

@Misc{www-awslambda,
title = {AWS Lambda},
author = {{Amazon}},
key = {AWS Lambda},
howpublished = {Web Page},
url = {https://aws.amazon.com/lambda/faqs/}
}

```

9.1.12 Book

Given the following entry. What is the proper entry for this book. Provide rationale:

```

@Book{netty-book,
Title = {Netty in Action},
Author = {Maurer, Norman and Wolfthal, Marvin},
Publisher = {Manning Publications},
Year = {2016},
}

```

To obtain the record of a book you can look at many information sources. The can include:

- <https://www.manning.com/books/netty-in-action>
- <https://www.amazon.com/Netty-Action-Norman-Maurer/dp/1617291471>
- <http://www.barnesandnoble.com/w/netty-in-action-norman-maurer/1117342155?ean=9781617291470#productInfoTabs>
- <http://www.powells.com/book/netty-in-action-9781617291470/1-0>

Furthermore, we need to consider the entry of a book, we simply look it up in emacs where we find the following but add the owner and the url field:

```

@Book{,
ALTAuthor = {},
ALTEditor = {},
title = {},
publisher = {},
year = {},
OPTkey = {},
OPTvolume = {},
OPTnumber = {},
OPTseries = {},
OPTaddress = {},
OPTedition = {},
OPTmonth = {},
OPTnote = {},
OPTannote = {},
ownwer = {},
url = {}
}

```

In summary we find the following fields:

Required fields: author/editor, title, publisher, year

Optional fields: volume/number, series, address, edition, month, note, key

We apply the following to fill out the fields.

address: The address is the Publisher's address. Usually just the city, but can be the full address for lesser-known publishers.

author: The name(s) of the author(s) (in the case of more than one author, separated by and). Names can be written in one of two forms: Donald E. Knuth or Knuth, Donald E. or van Halen, Eddie. Please note that Eddie van Halen would result in a wrong name. For our purpose we keep nobelity titles part of the last name.

edition: The edition of a book, long form (such as "First" or "Second")

editor: The name(s) of the editor(s)

key: A hidden field used for specifying or overriding the alphabetical order of entries (when the "author" and "editor" fields are missing). Note that this is very different from the key that is used to cite or cross-reference the entry.

label: The label field should contain three letters from the auth field, a short year reference and a short name of the publication to provide the maximum information regarding the publication. Underscores should be replaced with dashes or removed completely.

month: The month of publication or, if unpublished, the month of creation. Use three-letter abbreviations for this field in order to account for languages that do not capitalize month names. Additional information for the day can be included as follows: aug #“~10,”

publisher: The publisher's name

series: The series of books the book was published in (e.g. "The Hardy Boys" or "Lecture Notes in Computer Science")

title: The title of the work. As the capitalization depends on the bibliography style and the language used we typically use camel case. To force capitalization of a word or its first letter you can use the curly braces, '{ }'. To keep the title in camel case simple use title = { {My Title} }

type: The field overriding the default type of publication (e.g. "Research Note" for techreport, "{PhD} dissertation" for phdthesis, "Section" for inbook/incollection) volume The volume of a journal or multi-volume book year The year of publication (or, if unpublished, the year of creation)

While applying the above rules and tips we summarize what we have done for this entry:

1. Search for the book by title/Author on ACM (<http://dl.acm.org/>) or Amazon or barnesandnoble or upcitemdb (<http://upcitemdb.com>). These services return bibtex entrie that you can improve.
2. Hence one option is to get the ISBN of the book. For "Mesos in action" from upcitemdb we got the ISBN as "9781617 292927". This is the 13 digit ISBN. The first 3 digits (GS1 code) can be skipped. Using the rest of 10 digits "1617 292927", Add in JabRef in Optional Fields->ISBN.

However it is fine to just specify the full number.

We can also return a bibtex entry generated while using Click on the "Get BibTex from ISBN".

Now we get more information on this book entry from ISBN. We can opt either the original or newly searched entry for the below bibtex fields or merge as appropriate. URL may not match from where we initially read the book, however there is option to put your original url or newly searched url. EAN, Edition, Pages,url,published date etc. Do a search on amazon for "ASIN". Can skip if not available. Sometime we get ASIN for a different publication, maybe a paperback ASIN={B01MT311CU} We can add it as it becomes easier to search

doi: If you can find a doi numer you should also add it. IN this case we could not locate one.

As a result we obtain the entry:

```
@Book{netty-book,
  title = {Netty in Action},
  publisher = {Manning Publications Co.},
  year = {2015},
  author = {Maurer, Norman and Wolfthal, Marvin Allen},
  address = {Greenwich, CT, USA},
  edition = {1st},
  isbn = {1617291471},
  asin = {1617291471},
  date = {2015-12-23},
  ean = {9781617291470},
  owner = {S17-I0-3022 S17-I0-3010 S17-I0-3012},
  pages = {296},
  url = {http://www.ebook.de/de/product/21687528/norman_maurer_netty_in_action.html},
}
```

9.2 Integrating Bibtex entries into Other Systems

We have not tested any of this

9.2.1 Bibtex import to MSWord

XML import

Please respond back to us if you have used this and give feedback.

1. In JabRef, export the bibliography in MS Word 2008 xml format
2. Name the file Sources.xml (case sensitive)
3. In OSX with MS Word 2015: Go to /Library/Containers/com.microsoft.word/Data/Library/Application Support/Microsoft/Word/
4. Rename the original Sources.xml file to Sources.xml.bak
5. Copy the generated Sources.xml in this folder
6. Restart MS Word.

We do not know what needs to be done in case you need to make changes to the references. Please report back your experiences. To avoid issues we recommend that you use LaTeX. and not MSWord.

BibTex4Word

We have not tried this:

- <http://www.ee.ic.ac.uk/hp/staff/dmb/perl/index.html>

You are highly recommended to use Jabref for bibliography management in this class. Here is an introductory video on Jabref: <https://youtu.be/roi7vezNmfo?t=8m6s>

9.3 Other Reference Managers

Please note that you should first decide which reference manager you like to use. In case you for example install zotero and mendeley, that may not work with word or other programs.

9.3.1 Endnote

Endnote os a reference manager that works with Windows. Many people use Endnote. However, in the past, Endnote has caused complications when dealing with collaborative management of references. Its price is considerable. We have lost many hours of work because of instability of Endnote in some cases. As a student, you may be able to use Endnote for free at Indiana University.

- <http://endnote.com/>

9.3.2 Mendeley

Mendeley is a free reference manager compatible with Windows Word 2013, Mac Word 2011, LibreOffice, BibTeX. Videos on how to use it are available at:

- <https://community.mendeley.com/guides/videos>

Installation instructions are available at

- <https://www.mendeley.com/features/reference-manager/>

When dealing with large databases, we found the integration of Mendeley into word slow.

9.3.3 Zotero

Zotero is a free tool to help you collect, organize, cite, and share your research sources. Documentation is available at

- <https://www.zotero.org/support/>

The download link is available from

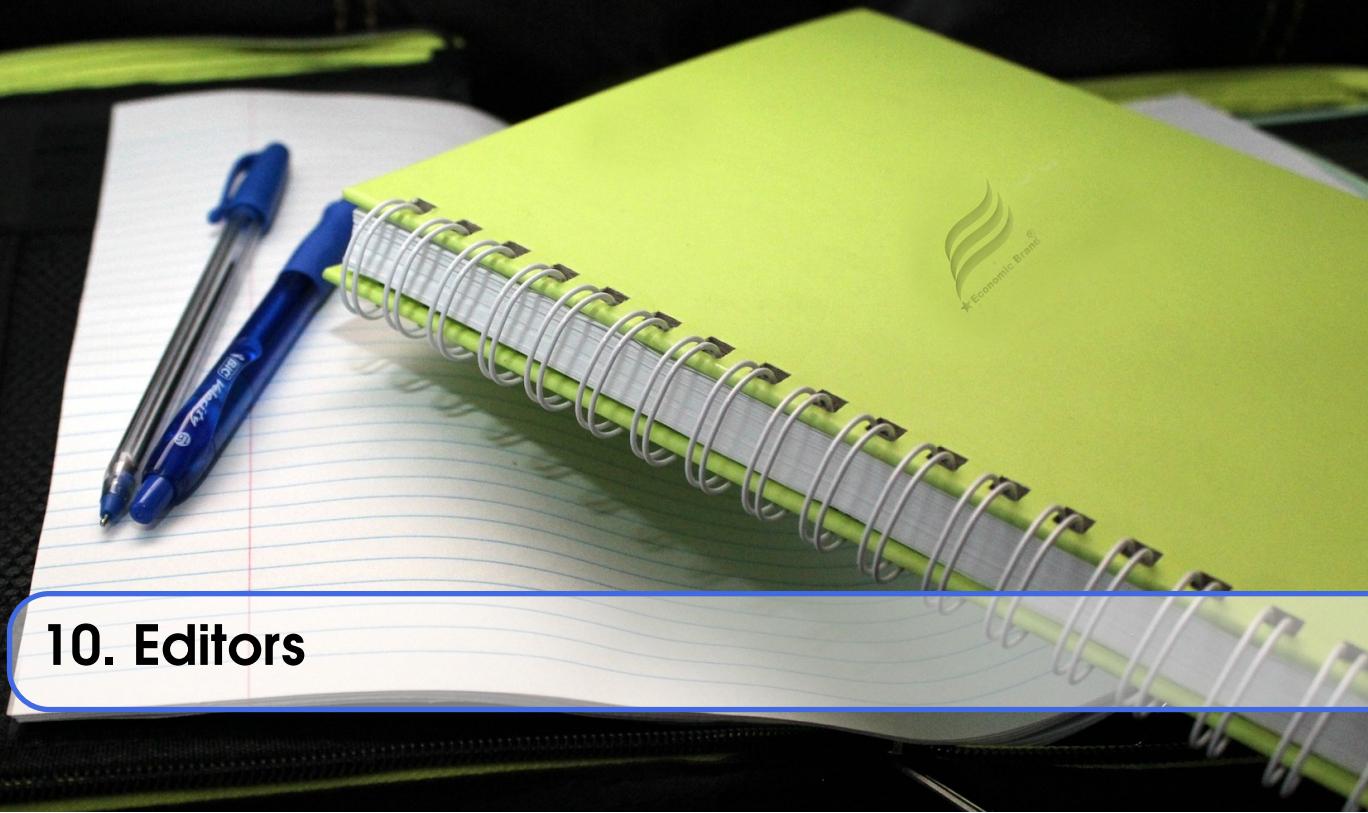
- <https://www.zotero.org/>

We have limited experience with Zotero

9.3.4 Paperpile

Paper pile is a Web based refernce management tool that integrates with Google docs. It can export the database as bibtex. Paperpile is a commercial tool cosing about \$36 a year for academic users. For others it is about 3 times as expensive.

- <https://paperpile.com>



10. Editors

F section/doc/emacs.tex

10.1 Basic Emacs

One of the most useful short manuals for emacs is the following reference card. It takes some time to use this card efficiently, but the most important commands are written on it. Generations of students have literally been just presented with this card and they learned emacs from it.

- <https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf>

There is naturally also additional material available and a great manual. You could also look at

- <https://www.gnu.org/software/emacs/tour/>

From the last page we have summarized the most useful and **simple** features. And present them here. One of the hidden gems of emacs is the ability to recreate replayable macros which we include here also. You ought to try it and you will find that for data science and the cleanup of data emacs (applied to smaller datasets) is a gem.

Notation

Key	Description
C	Control
M	Esc (meta character)

Here are some other ways on what to do if you have accidentally pressed a wrong key:

- C-g If you pressed a prefix key (e.g. C-x) or you invoked a command which is now prompting you for input (e.g. Find file: ...), type C-g, repeatedly if necessary, to cancel. C-g also

cancels a long-running operation if it appears that Emacs has frozen.

- C-/ If you executed a command and Emacs has modified your buffer, use C-/ to undo that change.

To save the current file say

Key	Description
C-x C-w	Write the buffer to file
C-x C-s	Write the buffer to file and quit Emacs

Moving around in buffers can be done with cursor keys, or with the following key combinations:

Key	Description
C-f	Forward one character
C-n	Next line
C-b	Back one character
C-p	Previous line

Here are some ways to move around in larger increments:

Key	Description
C-a	Beginning of line
M-f	Forward one word
M-a	Previous sentence
M-v	Previous screen
M-<	Beginning of buffer
C-e	End of line
M-b	Back one word
M-e	Next sentence
C-v	Next screen
M->	End of buffer

You can jump directly to a particular line number in a buffer:

Key	Description
M-g g	Jump to specified line

Searching is easy with the following commands

Key	Description
C-s	Incremental search forward
C-r	Incremental search backward

Replace

Key	Description
M-%	Query replace

Killing (“cutting”) text

Key	Description
C-k	Kill line

Yanking

Key	Description
C-y	Yanks last killed text

Macros

Keyboard Macros

Keyboard macros are a way to remember a fixed sequence of keys for later repetition. They’re handy for automating some boring editing tasks.

Key	Description
M-x (Start recording macro
M-x)	Stop recording macro
M-x e	Play back macro once
M-5 C-x-e	Play back macro 5 times

Modes

“Every buffer has an associated major mode, which alters certain behaviors, key bindings, and text display in that buffer. The idea is to customize the appearance and features available based on the contents of the buffer.” modes are typically activated by ending such as .py, .java, .rst, ...

Key	Description
M-x python-mode	Mode for editing Python files
M-x auto-fill-mode	Wraps your lines automatically when they get longer than 70 characters.
M-x flyspell-mode	Highlights misspelled words as you type.

10.1.1 Org Mode

Emacs has some very advanced fetures that you can activate via a mode. One such feature is to organize a TODO list via org-mode.

Instead of us designing our own video, we point to a community tutorial such as

[Emacs org-mode \(18:04\)](#)  Youtube

10.1.2 Programming Python with Emacs

Emacs comes by default with syntax hughl lighting for python when you edit a .py file. This is realy all you need. It also comes with a python ide that you can use and customize.

Python auto-completion for Emacs:

- <https://github.com/tkf/emacs-jedi>

Some more information is available at

- <https://realpython.com/blog/python/emacs-the-best-python-editor/>
- <https://www.emacswiki.org/emacs/PythonProgrammingInEmacs>

10.1.3 Emacs Keys in a Terminal

One of the real great features of knowing emacs is that you can set all your editors to emacs shortcuts. This includes pyCharm, but also bash. IN bash you simply say

```
set -o emacs
```

in your bash prompt. Additionally, if you do not have a window systems configured, you can run emacs directly in the terminal with

```
emacs -nw
```

This you can log in to a remote computer and if it has emacs installed. use it in the terminal. This would replace editors such as vi, vim, nano, pico or others that work in a terminal.

10.1.4 L^AT_EX and Emacs

LaTeX is directly supported by emacs and nothinghas to be changed. However, a collection of information about additional L^AT_EX features for emacs is available at

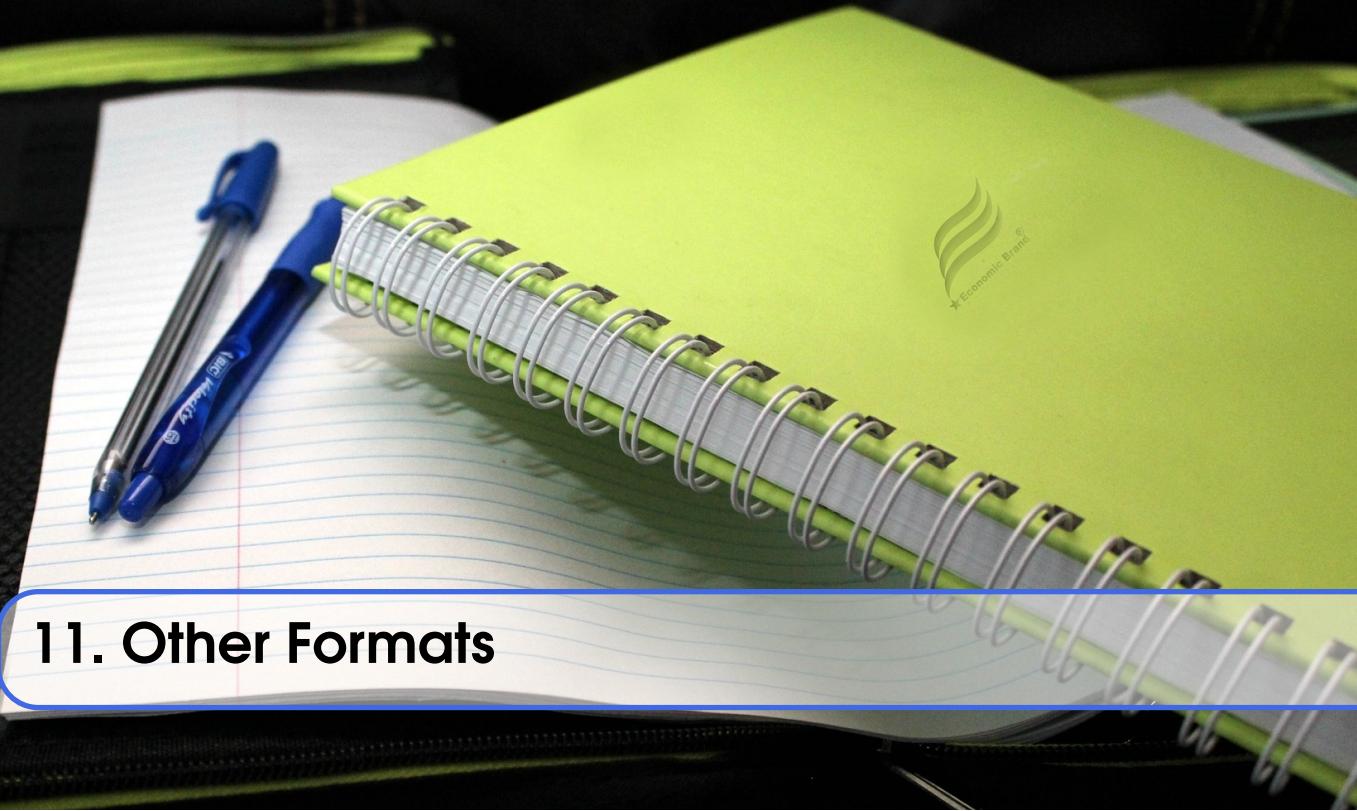
- <https://www.emacswiki.org/emacs/LaTeX>

Of interest are for example also

* M-x flyspell-mode: allowing to do spell checking in the window * predictive mode: <https://www.emacswiki.org/emacs/>
 * preview latex * whizzy tex

However instead of previes and whizzy tex we recommend to use <https://www.emacswiki.org/emacs/LatexMk> which comes preinstalled and allows you to do editing in one terminal, while previewing the update on change in another window.

LatexMk is all integrated in our report Makefiles and the Book format, so you will be able to use this immediately. This is similar to share latex, but much faster and without collaborators editing teh same file.



11. Other Formats

F section/doc/rst.tex

11.1 reStructuredText

reStructuredText (RST) purpose is to provide an easy-to-read, what-you-see-is-what-you-get plaintext markup syntax and parser system. With its help you can develop documentation not only for stand alone documentation, simple web pages, an in-line program documentation (such as Python). RST is extensible and new features can be added. It is used in sphinx as one of its supported formats.

11.1.1 Links

- RST Sphinx documentation: <http://www.sphinx-doc.org/en/stable/rest.html>
- RST Syntax: <http://docutils.sourceforge.net/rst.html>
- Important extensions: <http://sphinx-doc.org/ext/todo.html>

Cheatheat:

- <http://github.com/ralsina/rst-cheatsheet/raw/master/rst-cheatsheet.pdf>
- <http://docutils.sourceforge.net/docs/ref/rst/directives.html>

11.1.2 Source

The source for this page is located at

- <https://raw.githubusercontent.com/cloudmesh/classes/master/docs/source/lesson/doc/rst.rst>

This way you can look at the source on how we create this page.

11.1.3 Sections

with overline, for parts * with overline, for chapters =, for sections -, for subsections ^, for subsubsections ", for paragraphs

RST allows to specify a number of sections. You can do this with the various underlines:

```
*****
Chapter
*****
Section
=====
Subsection
-----
Subsubsection
~~~~~
Paragraph
~~~~~
```

11.1.4 Listtable

```
.. csv-table:: Eye colors
  :header: "Name", "Firstname", "eyes"
  :widths: 20, 20, 10

  "von Laszewski", "Gregor", "gray"
```

11.1.5 Exceltable

we have integrated Excel table from <http://pythonhosted.org//sphinxcontrib-exceltable/> into our sphinx allowing the definition of more elaborate tables specified in excel. However the most convenient way may be to use list-tables. The documentation to list tables can be found at <http://docutils.sourceforge.net/docs/ref/rst/directives.html#list-table>

11.1.6 Boxes

Seealso

```
.. seealso:: This is a simple **seealso** note.
```

Note

This is a **note** box.

```
.. note:: This is a **note** box.
```

Warning

note the space between the directive and the text

```
.. warning:: note the space between the directive and the text
```

Others

This is an **attention** box.

```
.. attention:: This is an **attention** box.
```

This is a **caution** box.

```

.. caution:: This is a **caution** box.

This is a danger box.

.. danger:: This is a **danger** box.

This is a error box.

.. error:: This is a **error** box.

This is a hint box.

.. hint:: This is a **hint** box.

This is an important box.

.. important:: This is an **important** box.

This is a tip box.

.. tip:: This is a **tip** box.

```

11.1.7 Sidebar directive

It is possible to create sidebar using the following code:

```

.. sidebar:: Sidebar Title
    :subtitle: Optional Sidebar Subtitle

    Subsequent indented lines comprise
    the body of the sidebar, and are
    interpreted as body elements.

```

Sidebar Title: Optional Sidebar Subtitle

Subsequent indented lines comprise the body of the sidebar, and are interpreted as body elements.

11.1.8 Sphinx Prompt

```

.. prompt:: bash, cloudmesh$

    wget -O cm-setup.sh http://bit.ly/cloudmesh-client-xenial
    sh cm-setup.sh

```

11.1.9 Programm examples

You can include code examples and bash commands with two colons.

This is an example for python:

```
print ("Hello World")
```

This is an example for a shell command:

```
$ ls -lisa
```

11.1.10 Hyperlinks

Direct links to html pages can be done with:

```
'This is a link to an html page <hadoop.html>'_
```

Note that this page could be generated from an rst page

Links to the FG portal need to be formulated with the portal tag:

```
:portal:'List to FG projects </projects/all>'
```

In case a subsection has a link declared you can use :ref: (this is the prefered way as it can be used to point even to subsections:

```
:ref:'Connecting private network VMs clusters <_s_vpn>'
```

A html link can be created anywhere in the document but must be unique. for example if you place:

```
.. _s_vpn:
```

in the text it will create a target to which the above link points when you click on it

11.1.11 Todo

```
.. todo:: an example
```

```
Todo  
an example
```



section/doc/markdown.tex

11.2 Markdown

The content form this section originates from see: <https://en.wikipedia.org/wiki/Markdown>.

Markdown is a simple markup language, however there is no precise standard defined for it and implementations may have features not supported by other implementations. Nevertheless, it provides as simple and easy way to quickly develop clean looking documents.

There are several tools that make markdown attractive allowing to write the text in one window while at the same time seeing the rendered output in another.

This includes

Macdown An editor for markdown targeted on OSX

To convert the markdown to other formats with pandoc

```
# Heading  
  
## Sub-heading  
  
### Another deeper heading  
  
Paragraphs are separated  
by a blank line.  
  
Two spaces at the end of a line leave a  
line break.  
  
Text attributes italic, *italic*, bold, **bold**, ‘monospace’.  
  
Horizontal rule:
```

Bullet list:

- * apples
- * oranges
- * pears

Numbered list:

1. apples
2. oranges
3. pears

A [link] (<http://example.com>) .

11.2.1 Tools

Dilinger • <https://dillinger.io/>

. A HTML5 based cloud enabled editor. It allows to download the created Markdown.

Macdown • <https://macdown.uranusrjr.com/>

“MacDown is an open source Markdown editor for macOS”



section/doc/type.tex

11.3 Communicating Research in Other Ways

Naturally, writing papers is not the only way to communicate your research with others. We find that today we see additional pathways for communication including blogs, twitter, facebook, e-mail, Web pages, and electronic notebooks. Let us revisit some of them and identify when they are helpful.

11.3.1 Blogs

blog: noun, a regularly updated website or web page, typically one run by an individual or small group, that is written in an informal or conversational style.

Advantages:

- encourages spontaneous posts
- encourages small short contributions
- chronologically ordered
- standard software exists to set up blogs
- online services exist to set up blogs

Disadvantages:

- structuring data is difficult (some blog software support it)
- not suitable for formal development of a paper
- often lack of sophisticated track change features
- no collaborative editing features

11.3.2 Sphinx

Sphinx (<http://www.sphinx-doc.org/>) is a tool that creates integrated documentation from a markup language while.

Advantages:

- output formats: html, LaTeX, PDF, ePub
- integrates well with directory structure
- powerful markup language (reStructuredText)
- can be hosted on GitHub via GitHub pages
- can integrate other renderers such as Markdown
- automatic table of contents, table of index
- code documentation integration
- search
- written in Python and using bash, so extensions and custom automation are possible

Disadvantage:

- requires compile step
- When using markdown GitHub can render individual page

Others:

- Read the Docs (<https://readthedocs.org/>)
- Doxygen (<http://www.stack.nl/~dimitri/doxygen/>)
- MkDocs (<http://www.mkdocs.org/>)

11.3.3 Notebooks

Jupyter

The Jupyter Notebook (<http://jupyter.org/>) is an open-source web application allowing users to create and share documents that contain live code, equations, visualizations and explanatory text. Use cases include data cleaning and transformation, numerical simulation, statistical modeling, machine learning.

Advantages:

- Integrates with Python
- Recently other programming languages have been integrated
- Allows experimenting with settings
- Allows a form of literate programming while mixing documentation with code
- automatically renders on GitHub
- comes with web service that allows hosting

Disadvantage:

- mostly encourages short documents
- mark up language is limited
- editing in ASCII is complex and Web editing is preferred

Apache Zeppelin

A Web-based notebook that enables data-driven, interactive data analytics and collaborative documents with SQL, Scala and Hadoop. It integrates a web-based notebook with data ingestion, data

exploration, visualization, sharing and collaboration features to Hadoop and Spark.

Advantages:

- integration to various framework
- Web framework
- integration with spark, hadoop

Disadvantages:

- larger framework
- must leverages existing deployments of spark, hadoop



12. Book Format

F section/doc/book-format.tex

This document is prepared using a special enhanced L^AT_EXbook format to allow easy integration for section and index references, as well as citations and other straight forward enhancements that makes it easy for students to obtain a document with much of the Lecture material included. We provide some examples so that if you like to contribute you can do so easily and can use the correct way of doing so.

12.1 Paragraphs of Text

Lore ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

12.2 Citation

Numbered List

```

1 This statement requires
  citation
  \cite{article_key};
2 this one is more specific
  \cite[162]{book_key}.

```

This statement requires citation [554]; this one is more specific [555, page 162].

12.3 Lists

Lists are useful to present information in a concise and/or ordered way¹.

12.3.1 Numbered List

Numbered List

```

1 \begin{enumerate}
2 \item The first item
3 \item The second item
4 \item The third item
5 \end{enumerate}

```

- 1. The first item
- 2. The second item
- 3. The third item

12.3.2 Bullet Points

Bullet Points

```

1 \begin{itemize}
2 \item The first item
3 \item The second item
4 \item The third item
5 \end{itemize}

```

- The first item
- The second item
- The third item

12.3.3 Descriptions and Definitions

Enumerations

```

1 \begin{description}
2 \item[Name] Description
3 \item[Word] Definition
4 \item[Comment] Elaboration
5 \end{description}

```

Name Description
Word Definition
Comment Elaboration

12.4 Theorems

This is an example of theorems.

¹Footnote example...

12.4.1 Several equations

This is a theorem consisting of several equations.

```

1 \begin{theorem}[Name of the theorem]
2 In $E=\mathbb{R}^n$ all norms are equivalent. It has the properties:
3 \begin{align}
4 & \|\mathbf{x}\| - \|\mathbf{y}\| \leq \|\mathbf{x} - \mathbf{y}\| \\
5 & \|\sum_{i=1}^n \mathbf{x}_i\| \leq \sum_{i=1}^n \|\mathbf{x}_i\| \quad \text{where } n \text{ is a finite integer}
6 \end{align}
7 \end{theorem}

```

Theorem 12.4.1 — Name of the theorem. In $E = \mathbb{R}^n$ all norms are equivalent. It has the properties:

$$\|\mathbf{x}\| - \|\mathbf{y}\| \leq \|\mathbf{x} - \mathbf{y}\| \quad (12.1)$$

$$\left\| \sum_{i=1}^n \mathbf{x}_i \right\| \leq \sum_{i=1}^n \|\mathbf{x}_i\| \quad \text{where } n \text{ is a finite integer} \quad (12.2)$$

12.4.2 Single Line

This is a theorem consisting of just one line.

```
\begin{theorem}
A set $\mathcal{D}(G)$ in dense in $L^2(G)$, $\|\cdot\|_0$.
\end{theorem}
```

Theorem 12.4.2 A set $\mathcal{D}(G)$ in dense in $L^2(G)$, $\|\cdot\|_0$.

12.5 Definitions

This is an example of a definition. A definition could be mathematical or it could define a concept.

Definition 12.5.1 — Definition name. Given a vector space E , a norm on E is an application, denoted $\|\cdot\|$, E in $\mathbb{R}^+ = [0, +\infty[$ such that:

$$\|\mathbf{x}\| = 0 \Rightarrow \mathbf{x} = \mathbf{0} \quad (12.3)$$

$$\|\lambda \mathbf{x}\| = |\lambda| \cdot \|\mathbf{x}\| \quad (12.4)$$

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\| \quad (12.5)$$

12.6 Notations

Exercise

```

1 \begin{notation}
2 Given an open subset $G$ of
    $\mathbb{R}^n$, the set of
    functions $\varphi$ are:
3 \begin{enumerate}
4 \item Bounded support $G$;
5 \item Infinitely
        differentiable;
6 \end{enumerate}
7 a vector space is denoted by
    $\mathcal{D}(G)$.
8 \end{notation}

```

Notation 12.1. Given an open subset G of \mathbb{R}^n , the set of functions φ are:

1. Bounded support G ;
2. Infinitely differentiable;

a vector space is denoted by $\mathcal{D}(G)$.

12.7 Remarks

This is an example of a remark.

Exercise

```

1 \begin{remark}
2 The concepts presented here
    are now in conventional
    employment in mathematics.
    Vector spaces are taken
    over the field
    $\mathbb{K}=\mathbb{R}$,
    however, established
    properties are easily
    extended to
    $\mathbb{K}=\mathbb{C}$.
3 \end{remark}

```



The concepts presented here are now in conventional employment in mathematics. Vector spaces are taken over the field $\mathbb{K} = \mathbb{R}$, however, established properties are easily extended to $\mathbb{K} = \mathbb{C}$.

12.8 Corollaries

This is an example of a corollary.

Exercise

```

1 \begin{corollary}[Corollary
    name]
2 The concepts presented here
    are now in conventional
    employment in mathematics.
    Vector spaces are taken
    over the field
     $\mathbb{K} = \mathbb{R}$ ,
    however, established
    properties are easily
    extended to
     $\mathbb{K} = \mathbb{C}$ .
3 \end{corollary}

```

Corollary 12.8.1 — Corollary name.

The concepts presented here are now in conventional employment in mathematics. Vector spaces are taken over the field $\mathbb{K} = \mathbb{R}$, however, established properties are easily extended to $\mathbb{K} = \mathbb{C}$.

12.9 Propositions

These are examples of propositions.

12.9.1 Several equations

Proposition 12.9.1 — Proposition name. It has the properties:

$$|||\mathbf{x}|| - ||\mathbf{y}||| \leq ||\mathbf{x} - \mathbf{y}|| \quad (12.6)$$

$$||\sum_{i=1}^n \mathbf{x}_i|| \leq \sum_{i=1}^n ||\mathbf{x}_i|| \quad \text{where } n \text{ is a finite integer} \quad (12.7)$$

12.9.2 Single Line

Proposition 12.9.2 Let $f, g \in L^2(G)$; if $\forall \varphi \in \mathcal{D}(G)$, $(f, \varphi)_0 = (g, \varphi)_0$ then $f = g$.

12.10 Examples

This is an example of examples.

12.10.1 Equation and Text

■ **Example 12.1** Let $G = \{x \in \mathbb{R}^2 : |x| < 3\}$ and denoted by: $x^0 = (1, 1)$; consider the function:

$$f(x) = \begin{cases} e^{|x|} & \text{si } |x - x^0| \leq 1/2 \\ 0 & \text{si } |x - x^0| > 1/2 \end{cases} \quad (12.8)$$

The function f has bounded support, we can take $A = \{x \in \mathbb{R}^2 : |x - x^0| \leq 1/2 + \varepsilon\}$ for all $\varepsilon \in]0; 5/2 - \sqrt{2}[$. ■

12.10.2 Paragraph of Text

Example

```

1 \begin{example}[Example name]
2 \lipsum[1]
3 \end{example}

```

■ **Example 12.2 — Example name.** Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

12.11 Exercises

This is an example of an exercise.

Exercise

```

1 \begin{exercise}
2 This is a good place to ask a
     question to test learning
     progress or further cement
     ideas into students' minds.
3 \end{exercise}

```

Exercise 12.1 This is a good place to ask a question to test learning progress or further cement ideas into students' minds.

12.12 Problems

Exercise

```

1 \begin{problem}
2 What is the average airspeed
   velocity of an unladen
   swallow?
3 \end{problem}

```

Problem 12.1 What is the average air-speed velocity of an unladen swallow?

12.13 Vocabulary

Define a word to improve a students' vocabulary.

Exercise

```

1 \begin{vocabulary}[Word]
2 Definition of word.
3 \end{vocabulary}

```

Vocabulary 12.1 — Word. Definition of word.

12.14 Tables

Treatments	Response 1	Response 2
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table 12.1: Table caption

12.15 Figures



Figure 12.1: Figure caption

12.15.1 Colored Boxes

The package `tcolorbox` provides sophisticated support to include color boxes. Together with the new environment we can create nice add ons to for example include notes.

- <http://osl.ugr.es/CTAN/macros/latex/contrib/tcolorbox/tcolorbox.pdf>

We have provided in this document notes as follows

Note	<pre>1 \begin{NOTE} 2 This is a note 3 \end{NOTE}</pre>	Note This is a note
Warning	<pre>1 \begin{WARNING} 2 This is a note 3 \end{WARNING}</pre>	Warning This is a note
Warning	<pre>1 \begin{IU} 2 This is a note 3 \end{IU}</pre>	Indiana University This is a note

12.15.2 Section References

Section references with the Where command are used to refer to a section by label, but be presented with the section number, the title, and the page number. It can be augmented with a text, such as in which week the section is best to be studied. This is useful for creating a customized Syllabus

Warning	<pre>1 \WHERE{\YES}{C:linux}{Week 2}</pre>	<input type="checkbox"/> ✓ 13. Linux Week 2 151
Warning	<pre>1 \WHERE{\NO}{C:linux}{Week 2}</pre>	<input type="checkbox"/> ✗ 13. Linux Week 2 151

The checkmark indicate if the lecture has been released for the ongoing semester.

12.15.3 Partial Compile

As we use include in the different parts, changes will only effect the appropriate part. However, we can also explicitly set in the preamble of the document which includes we compile. This is helpful, as the document is rather large and we may want to focus on just one or more parts without creating a large document that includes all the parts. For this we can use the convenient include only command and add to it with comma separated the parts that we want to compile and are regularly included within the main document. For example to just work on the container part, we simply add the following to the preamble.

```
\documentclass{format/laszewski}
```

```
\includeonly{part/container}
```

Once we are satisfied, we either remove the `includeonly` line, or we comment it out.



Development Tools

13	Linux	151
13.1	History	
13.2	Shell	
13.3	Multi-command execution	
13.4	Keyboard Shortcuts	
13.5	.bashrc and .bash_profile	
13.6	Exercises	
14	SSH	157
14.1	Generate a SSH key	
14.2	Add or Replace Passphrase	
14.3	SSH Add and Agent	
14.4	SSH Config	
14.5	SSH Port Forwarding	
14.6	Upload the key to gitlab	
14.7	Using SSH on Mac OS X	
14.8	Using SSH on Linux	
14.9	Using SSH on Raspberry Pi 3	
14.10	SSH on Windows	
14.11	Tips	
14.12	SSH to FutureSystems Resources	
14.13	Testing your ssh key	
14.14	References	
14.15	Exercises	
15	Github	167
15.1	Overview	
15.2	Upload Key	
15.3	Fork	
15.4	Rebase	
15.5	Remote	
15.6	Pull Request	
15.7	Branch	
15.8	Checkout	
15.9	Merge	
15.10	GUI	
15.11	Windows	
15.12	Git from the Commandline	
15.13	Configuration	
15.14	Upload your public key	
15.15	Working with a directory that was provided for you	
15.16	README.yml and notebook.md	
15.17	Contributing to the Document	
15.18	Exercises	
16	Virtual Box	175
16.1	Installation	
16.2	Guest additions	
16.3	Exercises	

13. Linux

F section/linux.tex

13.1 History

LINUX is a reimplementation by the community of UNIX which was developed in 1969 by Ken Thompson and Dennis Ritchie of Bell Laboratories and rewritten in C. An important part of UNIX is what is called the *kernel* which allows the software to talk to the hardware and utilize it.

In 1991 Linus Trovalds started developing a Linux Kernel that was initially targeted for PC's. This made it possible to run it on Laptops and was later on further developed by making it a full Operating system replacement for UNIX.

13.2 Shell

One of the most important features for us will be to access the computer with the help of a *shell*. The shell is typically run in what is called a terminal and allows interaction to the computer with commandline programs.

There are many good tutorials out there that explain why one needs a linux shell and not just a GUI. Randomly we picked the first one that came up with a google query (This is not an endorsement for the material we point to, but could be a worth while read for someone that has no experience in Shell programming:

- http://linuxcommand.org/lc3_learning_the_shell.php

Certainly you are welcome to use other resources that may suite you best. We will however summarize in table form a number of useful commands that you may also find even as a RefCard.

- <http://www.cheat-sheets.org/#Linux>

We provide in Table 13.1 a number of useful commands that you want to explore. For more information simply type man and the name of the command.

Table 13.1: Common commands

Example	Description
Help Commands	
man <i>command</i> apropos <i>text</i>	manual page for the <i>command</i> list all commands that have text in it
File Commands	
ls ls -lisa tree cd <i>dirname</i> mkdir <i>dirname</i> pwd rm <i>file</i> cp <i>a b</i> mv <i>a b</i> cat <i>a</i> cat - n less <i>a</i> head -5 <i>a</i> tail -5 <i>a</i> du -hs . wc <i>filename</i> sort <i>filename</i> tar -xvf <i>dir</i> rsync gzip bzip2 chmod go-rwx <i>file</i> chown <i>username file</i> chgrp <i>group file</i>	Directory listing list details list the directories in graphical form Change directory to <i>dirname</i> create the directory print working directory remove the file copy file <i>a</i> to <i>b</i> move/rename file <i>a</i> to <i>b</i> print content of file <i>a</i> (assignment) print paged content of file <i>a</i> Display first 5 lines of file <i>a</i> Display last 5 lines of file <i>a</i> show in human readable form the space used by the current directory counts the word in a file sorts the file tars up a compressed version of the directory (assignment) (assignment) (assignment) changes the permission of the file changes the ownership of the file changes the group of a file
Search Commands	
fgrep “text” <i>filename</i> grep -R “xyz” . find . -name “*.py” ps kill -9 1234 at cron crontab	searches the text in the given file recursively searches for xyz in all files find all files with .py at the end list the running processes kill the process with the id 1234 (assignment) (assignment) (assignment)

Continued on next page

Table 13.1 – continued from previous page

Example	Description
Device Commands	
mount /dev/cdrom /mnt/cdrom	mount a filesystem from a cd rom to /mnt/cdrom
System Commands	
users	(assignment)
who	(assignment)
whoami	(assignment)
dmesg	(assignment)
last	(assignment)
free -tm	(assignment)
uname	(assignment)
date	prints the current date and time
time <i>command</i>	prints the system, real and user time
shutdown -h “shut down”	(assignment)
Networking Commands	
ping	(assignment)
netstat	(assignment)
hostname	(assignment)
traceroute	(assignment)
ifconfig	(assignment)
Internet Commands	
host	(assignment)
whois	(assignment)
dig	(assignment)
wget	(assignment)
curl	(assignment)
Remote Access Commands	
ssh	(assignment)
scp	(assignment)
sftp	(assignment)

13.3 Multi-command execution

One of the important features is that one can execute multiple commands in the shell.

To execute command 2 once command 1 has finished use

```
command1; command2
```

To execute command 2 as soon as command 1 forwards output to stdout use

```
command1; command2
```

To execute command 1 in the background use

```
command1 &
```

13.4 Keyboard Shortcuts

These shortcuts will come in handy. Note that many overlap with emacs short cuts.

Keys	Description
Up Arrow	Show the previous command
Ctrl + z	Stops the current command
	Resume with fg in the foreground
	Resume with bg in the background
Ctrl + c	Halts the current command
Ctrl + l	Clear the screen
Ctrl + a	Return to the start of the line
Ctrl + e	Go to the end of the line
Ctrl + k	Cut everything after the cursor to a special clipboard
Ctrl + y	Paste from the special clipboard
Ctrl + d	Logout of current session, similar to exit

13.5 .bashrc and .bash_profile

Usage of a particular command and all the attributes associated with it, use ‘man’ command. Avoid using `rm -r` command to delete files recursively. A good way to avoid accidental deletion is to include the following in your `.bash_profile` file:

```
alias e=open_emacs
alias rm='rm -i'
alias mv='mv -i'
alias h='history'
```

More Information

<https://cloudmesh.github.io/classes/lesson/linux/refcards.html>

13.6 Exercises

Exercise 13.1 Familiarize yourself with the commands

Exercise 13.2 Find more commands that you find useful and add them to this page.

Exercise 13.3 Use the sort command to sort all lines of a file while removing duplicates.

Exercise 13.4 In Table 13.1 you will find a number of commands with (assignment). Develop descriptions that you will contribute and add to the manual with a pull request. Work in a team so that only one pull request is issued. Do not only provide the description, but also a real example as showcased within the table.

Exercise 13.5 Should there be other commands listed in the table. If so which? Create a pull request for them. ■

14. SSH



If you do not know what ssh is we recommend that you [read up on it](#). However, the simple material presented here will help you getting started quickly. It can however not replace the more comprehensive documentation. We also recommend that you read this entire section first before you start past and copy style tutorials as this will not allow you to understand the way ssh is used. For this reason we start with the explanation of ssh and not the setup on your machine. We want you to obtain first a solid overview of what you need to do.

To access remote resources this is often achieved via SSH. You need to provide a public ssh key to FutureSystem. We explain how to generate a ssh key, upload it to the FutureSystem portal and log onto the resources. This manual covers UNIX, Mac OS X, as well as Windows 10.

14.1 Generate a SSH key

First we must generate a ssh key with the tool [ssh-keygen](#). This program is commonly available on most UNIX systems (this includes Cygwin if you installed the ssh module or use our pre-generated cygwin executable). It will ask you for the location and name of the new key. It will also ask you for a passphrase, which you **MUST** provide. Please use a strong passphrase to protect it appropriately.

Warning

Some teachers and teaching assistants advice you to not use passphrases. This is **WRONG** as it allows someone that gains access to your computer to also gain access to all resources that have the public key.

In case you already have a ssh key in your machine, you can reuse it and skip this whole section.

To generate the key, please type:

Example:

```
ssh-keygen -t rsa -C localname@indiana.edu
```

This command requires the interaction of the user. The first question is:

```
Enter file in which to save the key (/home/localname/.ssh/id_rsa):
```

We recommend using the default location `~/.ssh/` and the default name `id_rsa`. To do so, just press the enter key.

Your `localname` is the username on your computer.

The second and third question is to protect your ssh key with a passphrase. This passphrase will protect your key because you need to type it when you want to use it. Thus, you can either type a passphrase or press enter to leave it without passphrase. To avoid security problems, you **MUST** choose a passphrase. Make sure to not just type return for an empty passphrase:

```
Enter passphrase (empty for no passphrase):
```

and:

```
Enter same passphrase again:
```

If executed correctly, you will see some output similar to:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/localname/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/localname/.ssh/id_rsa.
Your public key has been saved in /home/localname/.ssh/id_rsa.pub.
The key fingerprint is:
34:87:67:ea:c2:49:ee:c2:81:d2:10:84:b1:3e:05:59 localname@indiana.edu

+--[ RSA 2048]----+
|.+.Eo= . |
| ..=.o + o +o |
|0. = ..... |
|= . . . |
+-----+
```

Once, you have generated your key, you should have them in the `.ssh` directory. You can check it by :

```
$ cat ~/.ssh/id_rsa.pub
```

If everything is normal, you will see something like:

```
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCXJH2iG2FMqC6T/U7uB8kt
6K1Rh4kU0jgw9sc4Uu+Uwe/kshuispauhfsjhf,anf6787sjgdkjsgl+EwD0
thkoamyi0VvhTVZhj61pTdhy1t8hlkoL19JVnVBPP5kIN3wVyNAJjYBrAUNW
4dXXXtmfkXp98T30W4mxAtTH434MaT+QcPTcxims/hwsUeDAVKZY7UgZhEbIE
xxkejtnRBHTipi0W03W05TOUGRW7EuKf/4ftNVPilC04DpfY44NFG1xPwHeim
Uk+t9h48pBQj16FrUCp0rS02Pj+4/9dNeS1kmNJu5ZYS8HVRhvuotXuAY/UVc
ynEPUEgkp+qYnR user@myemail.edu
```

14.2 Add or Replace Passphrase

In case you need to change your change passphrase, you can simply run `ssh-keygen -p` command. Then specify the location of your current key, and input (old and) new passphrases. There is no need to re-generate keys:

```
ssh-keygen -p
```

You will see the following output once you have completed that step:

```
Enter file in which the key is (/home/localname/.ssh/id_rsa):  
Enter old passphrase:  
Key has comment '/home/localname/.ssh/id_rsa'  
Enter new passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved with the new passphrase.
```

14.3 SSH Add and Agent

To not always type in your password, you can use `ssh-add`.

It prompts the user for a private key passphrase and add it to a list of keys managed by the ssh-agent. Once it is in this list, you will not be asked for the passphrase as long as the agent is running with your public key.

To start the agent please use the following command:

```
eval `ssh-agent`
```

It is important that you use the backquote (`), located under the tilde (~), rather than the single quote ('). Once the agent is started it will print a PID that you can use to interact with later

To add the key use the command

```
ssh-add
```

To remove the agent use the command

```
kill $SSH_AGENT_PID
```

To execute the command upon logout, place it in your `.bash_logout` (assuming you use bash).

14.4 SSH Config

TODO: describe ssh config

14.5 SSH Port Forwarding

TODO: describe port forwarding

14.6 Upload the key to gitlab

Indiana University

We do not use gitlab.com for most classes but use github.com

In case you use gitlab Follow the instructions provided here to upload the public key

- <http://docs.gitlab.com/ce/ssh/README.html>

14.7 Using SSH on Mac OS X

Mac OS X comes with an ssh client. In order to use it you need to open the Terminal.app application. Go to Finder, then click Go in the menu bar at the top of the screen. Now click Utilities and then open the Terminal application.

14.8 Using SSH on Linux

All Linux versions come with ssh and can be used right from the terminal.

14.9 Using SSH on Raspberry Pi 3

Install Raspbian on an SD card, and boot up your system. Before putting it on the network, reset the default user password with the passwd command. Now you can use the terminal and use ssh just like on any Linux computer.

14.10 SSH on Windows

Warning

For this class we recommend that you use a virtual machine via virtual box and use the Linux ssh instructions. The information here is just provided for completeness and no support will be offered for native windows support.

Windows users need to have some special software to be able to use the SSH commands. If you have one that you are comfortable with and know how to setup key pairs and access the contents of your public key, please feel free to use it.

On Windows you have a couple of options on running Linux commands such as ssh. At this time it may be worth while to try the OpenSSH Client available for Windows, although it is in beta. If you like to use other methods we have included alternatives.

14.10.1 OpenSSH Client (Beta)

TODO: provide us with screenshots of the windows.

In case you need access to ssh Microsoft has fortunately updated their software to be able to run it directly from the Windows commandline including PowerShell.

However it is as far as we know not activated by default so you need to follow some setup scripts. Also this software is considered beta and its development and issues can be found at

- <https://github.com/PowerShell/Win32-OpenSSH>
- <https://github.com/PowerShell/Win32-OpenSSH/issues>

What you have to do is to install it by going to Settings > Apps and click Manage optional features under Apps & features.

Next, Click on the Add feature. You will be presented with a list in which you scroll down, till you find OpenSSH Client (Beta). Click on it and invoke Install.

After the install has completed, you can use the ssh command. Just type it in the commandshell or PowerShell

```
PS C:\Users\gregor> ssh
```

Naturally you can now use it just as on Linux or OSX. and use it to login to other resources

```
PS C:\Users\gregor> ssh myname@computer.example.com
```

14.10.2 Using SSH from Cygwin

One established way of using ssh is from using cygwin.

- <http://cygwin.com/install.html>

Cygwin contains a collection of GNU and Open Source tools providing Linux like functionality on Windows. A DLL is available that exposes the POSIX API functionality.

A list of supported commands is available at

- https://cygwin.com/packages/package_list.html

Please be minded that in order for cygwin to function easily the Windows user name should not include spaces. However, as the setup in windows encourages to use the full name when you buy and setup a machine it may not be convenient to use. However, we just recommend that you create yourself a new username and use this if you like to use cygwin.

You can selectively install from the cygwin setup terminal which software you like to use, obviously you may want to use ssh

14.10.3 SSH from putty

As you will see the process is somewhat cumbersome and when you compare it with the command-line tools available, we do recommend using them instead.

PuTTY allows you to access the SSH, Telnet and Rlogin network protocols from windows.

- <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

Although PuTTY has been out there for many years and served the community well, it is not following the standard ssh command line syntax when invoked from a command shell.

```
putty -ssh user@host.name
```

In addition to using ssh, it also provides a copy command.

```
pscp user@host.name:"\"remote filename with spaces\\"" local_filename
```

Putty is best known for its GUI configuration application to manage several machines as demonstrated next. Once you have downloaded it and opened PuTTYgen, you will be presented with a key generator window (images provided by chameleon cloud) (see Figure 14.1).

To generate a key you click the *Generate* button which is blue. The PuTTY Key Generator (see Figure 14.2) will then ask you to move your mouse around the program's blank space to generate "randomness" for your key. You must enter a "Key passphrase" and then confirm the passphrase.

Next you need to save both the public and private keys into a file of your choice using the "Save public key" and "Save private key" buttons. We suggest you name something obvious like "public_key.pub" and "private_key" so that you can distinguish between the two.

Before closing this window, select the entire public key and copy it with "Control-C". Please note

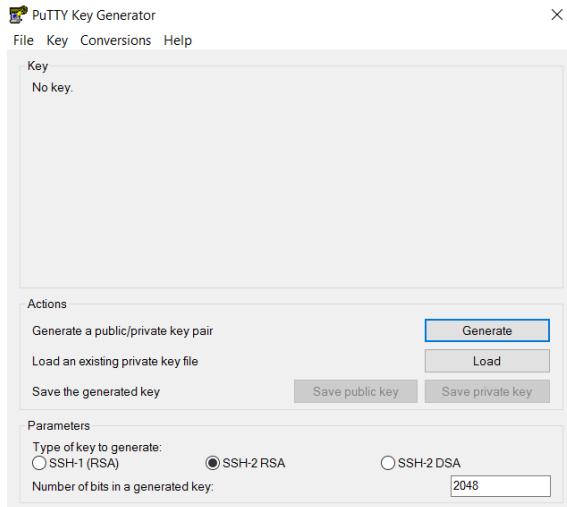


Figure 14.1: Key generation window

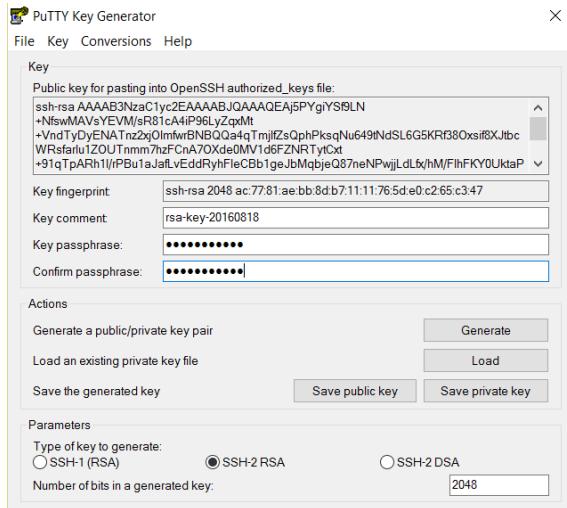


Figure 14.2: Key generation window

that everything should be copied, including “ssh-rsa”. This will be used when importing the key pair to Openstack.

At this time, the public key has been created and copied. Now you can use the public key and upload it to systems you like to login to.

14.10.4 Chocolatey

Another approach is to use it in Powershell with the help of chocolatey. Other options may be better suited for you and we leave it up to you to make this decision.

Chocolatey is a software management tool that mimics the install experience that you have on Linux and OSX. It has a repository with many packages. The packages are maintained by the community and you need to evaluate security implications when installing packages hosted on chocolatey just as you have to do if you install software on Linux and OSX from their repositories. Please be aware that there could be malicious code offered in the chocolatey repository although

the distributors try to remove them.

The installation is sufficiently explained at

- <https://chocolatey.org/install>

Once installed you have a command choco and you should make sure you have the newest version with :

```
choco upgrade chocolatey
```

Now you can browse packages at

- <https://chocolatey.org/packages>

Search for openssh and see the results. You may find different versions. Select the one that most suits you and satisfies your security requirements as well as your architecture. Lets assume you chose the Microsoft port, than you can install it with:

```
choco install openssh
```

Naturally, you can also install cygwin and pty over chocolatey. A list of packages can be found at

- <https://chocolatey.org/packages>

Packages of interest include

- emacs: choco install emacs
 - <https://chocolatey.org/packages/Emacs>
- pandoc: choco install pandoc
 - <https://chocolatey.org/packages/pandoc>
- LaTeX: choco install miktex
 - <https://chocolatey.org/packages/miktex>
- jabref: choco install jabref
 - <https://chocolatey.org/packages/JabRef>
- pycharm: choco install pycharm-community
 - <https://chocolatey.org/packages/PyCharm-community>
- lyx: choco install lyx
 - <https://chocolatey.org/packages/lyx>
- python 2: choco install python2
 - <https://chocolatey.org/packages/python2>
- python 3: choco install python
 - <https://chocolatey.org/packages/python/3.6.4>
- pip: choco install pip
 - <https://chocolatey.org/packages/pip>
- virtualbox: choco install virtualbox
 - <https://chocolatey.org/packages/virtualbox>
- vagrant: choco install vagrant
 - <https://chocolatey.org/packages/vagrant>

Before installing any of them evaluate if you need them and identify security risks.

14.11 Tips

Use SSH keys You will need to use ssh keys to access remote machines

No blank passphrases In most cases you must use a passphrase with your key. In fact if we find that you use passwordless keys to futuresystems and to chameleon cloud resources, we may elect to give you an *F* for the assignment in question. There are some exceptions, but they will be clearly communicated to you in class. You will as part of your cloud drivers license test explain how you gain access to futuresystmes and chameleon to explicitly explain this point and provide us with reasons what you can not do.

A key for each server Under no circumstances copy the same private key on multiple servers.

This violates security best practices. Create for each server a new private key and use their public keys to gain access to the appropriate server.

Use SSH agent So as to not to type in all the time the passphrase for a key, we recommend using ssh-agent to manage the login. This will be part of your cloud drivers license.

But shut down the ssh-agent if not in use

keep an offline backup, put encrypt the drive You may for some of our projects need to make backups of private keys on other servers you set up. If you like to make a backup you can do so on a USB stick, but make sure that access to the stick is encrypted. Do not store anything else on that key and look it in a safe place. If you lose the stick, recreate all keys on all machines.

14.12 SSH to FutureSystems Resources

Next, you need to upload the key to the portal. You must be logged into the portal to do so.

Step 1: Log into the portal

Step 2: Click in the “ssh key” button. or go directly to <https://portal.futuresystems.org/my/ssh-keys>

Step 3: Click in the “add a public key” link.

Title	Fingerprint	Operations
javinew	71:6e:5a:3a:47:7d:4e:5b:55:e2:3e:b0:43:f7:c5:ed	Edit Delete
jdiaz-india	fe:8e:8c:98:f3:49:02:56:f1:30:7e:21:46:b9:49:7b	Edit Delete
jdiaz@javi-OptiPlex-960	d9:96:06:b4:cf:05:5d:79:63:fb:38:60:61:15:30:7c	Edit Delete
jdiaz@localhost	42:af:52:fa:01:dc:4c:14:82:ea:00:8b:02:eb:dc	Edit Delete
mincluster	c5:3f:01:c9:3e:1e:0e:6ff5:a6:7f:04:3d:1e:af:45	Edit Delete
rsa-key-20101004	c6:f5:05:0b:bf:09:4a:31:fd:d3:65:4c:ca:68:83	Edit Delete
scl1-key	46:8b:8f:44:75:a3:16:21:61:63:96:01:62:e3:36:73	Edit Delete
sierra	23:b2:97:39:26:4c:da:c7:38:9a:75:3b:52:c6:c3:d8	Edit Delete

Step 4: Paste your ssh key into the box marked Key. Use a text editor to open the “id_rsa.pub”. Copy the entire contents of this file into the ssh key field as part of your profile information. Many errors are introduced by users in this step as they do not paste and copy correctly.

The screenshot shows a web page titled "FutureGrid Portal". At the top, there are links for "Staff", "Welcome, jdiaz!", "Logout", and a search bar. Below the header, there are navigation links for "Summer School", "About", "News", "Support", "Community", and "Projects". The main content area is titled "Add a SSH key". It includes a note about GitHub help for public keys. A "Title" field is present, with a note that it will be automatically generated if left blank. A large "Key" field contains a long, complex SSH key string. At the bottom, there are "Submit" and "Cancel" buttons.

Step 5: Click the submit button. **IMPORTANT:** Leave the Title field blank. Make sure that when you paste your key, it does not contain newlines or carriage returns that may have been introduced by incorrect pasting and copying. If so, please remove them.

At this point, you have uploaded your key. However, you will still need to wait till all accounts have been set up to use the key, or if you did not have an account till it has been created by an administrator. Please, check your email for further updates. You can also refresh this page and see if the boxes in your account status information are all green. Then you can continue.

14.13 Testing your ssh key

If you have had no FutureSystem account before, you need to wait for up to two business days so we can verify your identity and create the account. So please wait. Otherwise, testing your new key is almost instantaneous on india. For other clusters like it can take around 30 minutes to update the ssh keys.

To log into india simply type the usual ssh command such as:

```
$ ssh portalname@india.futuresystems.org
```

The first time you ssh into a machine you will see a message like this:

```
The authenticity of host 'india.futuresystems.org (192.165.148.5)' can't be established.
RSA key fingerprint is 11:96:de:b7:21:eb:64:92:ab:de:e0:79:f3:fb:86:dd.
Are you sure you want to continue connecting (yes/no)? yes
```

You have to type yes and press enter. Then you will be logging into india. Other FutureSystem machines can be reached in the same fashion. Just replace the name india, with the appropriate FutureSystems resource name.

14.14 References

- [The Secure Shell: The Definitive Guide, 2 Ed \(O'Reilly and Associates\)](#)
- [putty](#)

14.15 Exercises

Exercise 14.1 create an SSH key pair

Exercise 14.2 upload the public key to git repository you use. Create a fork in git and use your ssh key to clone and commit to it

Exercise 14.3 Get an account on futuresystems.org (if you are authorized to do so). Upload your key to futuresystems.org. Login to india.futuresystems.org Note. that this could take some time as administrators need to approve you. Be patient. ■



15. Github

F section/git/github.tex

In some classes the material may be openly shared in code repositories. This includes class material, papers and project. Hence, we need some mechanism to share content with a large number of students. For this reason we use

- github.com

First, we like to introduce you to git and github.com (Section 15.1). Next, we provide you with the basic commands to interact with git from the commandline (Section 15.12). Than we will introduce you how you can contribute to this set of documentations with pull requests.

15.1 Overview

Github is a code repository that allows the development of code and documents with many contributors in a distributed fashion. There are many good tutorials about github. Some of them can be found on the github Web page. An interactive tutorial is for example available at

- <https://try.github.io/>

However, although these tutorials are helpful in many cases they do not address some cases. For example, you have already a repository set up by your organization and you do not have to completely initialize it. Thus do not just replicate the commands in the tutorial, or the once we present here before not evaluating their consequences. In general make sure you verify if the command does what you expect **before** you execute it.

A more extensive list of tutorials can be found at

- <https://help.github.com/articles/what-are-other-good-resources-for-learning-git-and-github>

The github foundation has a number of excellent videos about git. If you are unfamiliar with git and you like to watch videos in addition to reading the documentation we recommend these videos

- <https://www.youtube.com/user/GitHubGuides/videos>

Next, we introduce some important concepts used in github.

15.2 Upload Key

Before you can work with a repository in an easy fashion you need to upload a public key in order to access your repository. Naturally, you need to generate a key first which is explained in

TODO: lessons-ssh-generate-key

before you upload one. Copy the contents of your `.ssh/id_rsa.pub` file and add them to [your github keys](#).

More information on this topic can be found on the [github Web page](#).

15.3 Fork

Forking is the first step to contributing to projects on GitHub. Forking allows you to copy a repository and work on it under your own account. Next, creating a branch, making some changes, and offering a pull request to the original repository, rounds out your contribution to the open source project.

Fork (1:41) 

15.4 Rebase

When you start editing your project, you diverge from the original version. During your developing, the original version may be updated, or other developers may have some of their branches implementing good features that you would like to include in your current work. That is when *Rebase* becomes useful. When you *Rebase* to certain points, could be a newer Master or other custom branch, consider you graft all your on-going work right to that point.

Rebase may fail, because some times it is impossible to achieve what we just described as conflicts may exist. For example, you and the to-be-rebased copy both edited some common text section. Once this happens, human intervention needs to take place to resolve the conflict.

Rebase (4:20) 

15.5 Remote

Collaborating with others involves managing the remote repositories and pushing and pulling data to and from them when you need to share work. Managing remote repositories includes knowing how to add remote repositories, remove remotes that are no longer valid, manage various remote branches and define them as being tracked or not, and more.

Though out this semester, you will typically work on two *remote* repos. One is the office class repo, and another is the repo you forked from the class repo. The class repo is used as the centralized,

authority and final version of all student submissions. The repo under your own Github account is for your personal storage. To show progress on a weekly basis you need to commit your changes on a weekly basis. However make sure that things in the master branch are working. If not, just use another branch to conduct your changes and merge at a later time. we like you to call your development branch dev.

- <https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes>

15.6 Pull Request

Pull requests are a means of starting a conversation about a proposed change back into a project. We will be taking a look at the strength of conversation, integration options for fuller information about a change, and cleanup strategy for when a pull request is finished.

Pull Request (4:26) 

15.7 Branch

Branches are an excellent way to not only work safely on features or experiments, but they are also the key element in creating Pull Requests on GitHub. Lets take a look at why we want branches, how to create and delete branches, and how to switch branches in this episode.

Branch (2:25) 

15.8 Checkout

Change where and what you are working on with the checkout command. Whether we are switching branches, wanting to look at the working tree at a specific commit in history, or discarding edits we want to throw away, all of these can be done with the checkout command.

Checkout (3:11) 

15.9 Merge

Once you know branches, merging that work into master is the natural next step. Find out how to merge branches, identify and clean up merge conflicts or avoid conflicts until a later date. Lastly, we will look at combining the merged feature branch into a single commit and cleaning up your feature branch after merges.

Merge (3:11) 

15.10 GUI

Using Graphical User Interfaces can supplement your use of the command line to get the best of both worlds. GitHub for Windows and GitHub for Mac allow for switching to command line, ease of grabbing repositories from GitHub, and participating in a particular pull request. We will also see the auto-updating functionality helps us stay up to date with stable versions of Git on the command line.

[GUI \(3:47\)](#) 

There are many other git GUI tools available that directly integrate into your operating system finders, windows, . . . , or PyCharm. It is up to you to identify such tools and see if they are useful for you. Most of the people we work with us git from the command line, even if they use PyCharm, eclipse, or other tools that have build in git support. You can identify a tool that works best for you.

15.11 Windows

This is a quick tour of GitHub for Windows. It offers GitHub newcomers a brief overview of what this feature-loaded version control tool and an equally powerful web application can do for developers, designers, and managers using Windows in both the open source and commercial software worlds. More: <http://windows.github.com>

[Windows \(1:25\)](#) 

15.12 Git from the Commandline

Although github.com provides a powerful GUI and other GUI tools are available to interface with github.com, the use of git from the commandline can often be faster and in many cases may be simpler.

Git commandline tools can be easily installed on a variety of operating systems including Linux, OSX, and Windows. Many great tutorials exist that will allow you to complete this task easily. We found the following two tutorials sufficient to get the task accomplished:

- <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
- <https://www.atlassian.com/git/tutorials/install-git>

Although the later is provided by an alternate repository to github. The installation instructions are very nice and are not impacted by it. Once you have installed git you need to configure it.

15.13 Configuration

Once you installed Git, you can need to configure it properly. This includes setting up your username, email address, line endings, and color, along with the settings' associated configuration scopes.

[Configuration \(2:47\)](#) 

It is important that make sure that use the `git config` command to initialize git for the first time on each new computer system or virtual machine you use. This will ensure that you use on all resources the same name and e-mail so that git history and log will show consistently your checkins across all devices and computers you use. If you do not do this, your checkins in git do not show up in a consistent fashion as a single user. Thus on each computer execute the following commands:

```
$ git config --global user.name "Albert Zweistein"
$ git config --global user.email albert.zweistein@gmail.com
```

where you replace the information with the information related to you. You can set the editor to emacs with:

```
$ git config --global core.editor emacs
```

Naturally if you happen to want to use other editors you can configure them by specifying the command that starts them up. You will also need to decide if you want to push branches individually or all branches at the same time. It will be up to you to make what will work for you best. We found that the following seems to work best:

```
git config --global push.default matching
```

More information about a first time setup is documented at:

```
* http://git-scm.com/book/en/Getting-Started-First-Time-Git-Setup
```

To check your setup you can say:

```
$ git config --list
```

One problem we observed is that students often simply copy and paste instructions, but do not read carefully the error that is reported back and do not fix it. Overlooking the proper set of the push.default is often overlooked. Thus we remind you: **Please read the information on the screen when you set up.**

15.14 Upload your public key

Please upload your public key to the repository as documented in github, while going to your account and find it in settings. There you will find a panel SSH key that you can click on which brings you to the window allowing you to add a new key. If you have difficulties with this find a video from the github foundation that explains this.

15.15 Working with a directory that was provided for you

In case your course provided you with a github directory, starting and working in it is going to be real simple. If you are the only student working on this you still need to make sure that papers or programs you manage in the repository work and do not interfere with scripts that instructors may use to check your assignments. Thus it is good to still create a branch, work in the branch and then merge the branch into the master once you verified things work. After you merged you can push the content to the github repository.

Tip: Please use only **lowercase** characters in the directory names and no special characters such as @ ; / _ and spaces. In general we recommend that you avoid using directory names with capital letters spaces and _ in them. This will simplify your documentation efforts and make the URLs from git more readable. Also while on some OS's the directories *MyDirectory* is different from *mydirectory* on OSX it is considered the same and thus renaming from capital to lower case can not be done without first renaming it to another directory.

Your homework for submission should be organized according to folders in your clone repository. To submit a particular assignment, you must first add it using:

```
git add <name of the file you are adding>
```

Afterwards, commit it using:

```
git commit -m "message describing your submission"
```

Then push it to your remote repository using:

```
git push
```

If you want to modify your submission, you only need to:

```
git commit -m "message relating to updated file"
```

afterwards:

```
git push
```

If you lose any documents locally, you can retrieve them from your remote repository using:

```
git pull
```

15.16 README.yaml and notebook.md

In case you take classes e516 and e616 with us you will have to create a README.yaml and notebook.md file in the top most directory of your repository. It serves the purpose of identifying your submission for homework and information about yourself.

It is important to follow the format precisely. As it is yaml it is an easy homework to write a 4 line python script that validates if the README.yaml file is valid. In addition you can use programs such as yamllint which is documented at

- <https://yamllint.readthedocs.io/en/latest/>

This file is used to integrate your assignments into a proceedings. An example is provided at

- <https://github.com/bigdata-i523/sample-hid000/blob/master/README.yml>

Any derivation from this format will not allow us to see your homework as our automated scripts will use the README.yaml to detect them. Make sure the file does not contain any TABs. Please also mind that all filenames of all homework and the main directory must be **lowercase** and do not include spaces. This will simplify your task of managing the files across different operating systems.

In case you work in a team, on a submission, the document will only be submitted in the author and hid that is listed first. All other readmes, will have for that particular artifact a **duplicate: yes** entry to indicate that this submission is managed elsewhere. The team will be responsible to manage their own pull requests, but if the team desires we can grant access for all members to a repository by a user. Please be aware that you must make sure you coordinate with your team.

We will not accept submission of homework as pdf documents or tar files. All assignments must be submitted as code and the reports in native latex and in github. We have a script that will automatically create the PDF and include it in a proceedings. There is no exception from this rule and all reports not compilable will be returned without review and if not submitted within the deadline receive a penalty.

Please check with your instructor on the format of the README.yaml file as it could be different for your class.

15.17 Contributing to the Document

TODO: This section has to be redone as we use class specific clones and not the master

15.17.1 Clone

```
$ git remote add upstream https://github.com/cloudmesh/book
```

15.17.2 Merge

As we are allowing contribution by the community, they are best managed through a merge with our upstream repository so you can update to the newest status before you issue a pull request.

Make sure you have upstream repo defined:

```
$ git remote add upstream https://github.com/cloudmesh/book
```

Now Get latest from upstream:

```
$ git rebase upstream/master
```

In this step, the conflicting file shows up (in my case it was refs.bib):

```
$ git status
```

should show the name of the conflicting file:

```
$ git diff <file name>
```

should show the actual differences. May be in some cases, It is easy to simply take latest version from upstream and reapply your changes.

So you can decide to checkout one version earlier of the specific file. At this stage, the re-base should be complete. So, you need to commit and push the changes to your fork:

```
$ git commit  
$ git rebase origin/master  
$ git push
```

Then reapply your changes to refs.bib - simply use the backedup version and use the editor to redo the changes.

At this stage, only refs.bib is changed:

```
$ git status
```

should show the changes only in refs.bib. Commit this change using:

```
$ git commit -a -m "new:usr: <message>"
```

And finally push the last committed change:

```
$ git push
```

The changes in the file to resolve merge conflict automatically goes to the original pull request and the pull request can be merged automatically.

You still have to issue the pull request from the Github Web page so it is registered with the upstream repository.

15.17.3 Resources

- [Pro Git book](#)
- [Official tutorial](#)
- [Official documentation](#)
- [TutorialsPoint on git](#)
- [Try git online](#)
- [GitHub resources for learning git](#) Note: this is for github and not for gitlab. However as it is for gt the only thing you have to do is replace hithub, for gitlab.
- [Atlassian tutorials for git](#)

In addition the tutorials from atlassian are a good source. However remember that you may not use bitbucket as the repository, so ignore those tutorials. We found the following useful

- What is git: <https://www.atlassian.com/git/tutorials/what-is-git>
- Installing git: <https://www.atlassian.com/git/tutorials/install-git>
- git config: <https://www.atlassian.com/git/tutorials/setting-up-a-repository#git-config>
- git clone: <https://www.atlassian.com/git/tutorials/setting-up-a-repository#git-clone>
- saving changes: <https://www.atlassian.com/git/tutorials/saving-changes>
- collaborating with git: <https://www.atlassian.com/git/tutorials syncing>

15.18 Exercises

Exercise 15.1 How do you set your favorite editor as a default with github config



Exercise 15.2 What is the difference between merge and rebase?



Exercise 15.3 Assume you have made a change in your local fork, however other users have since committed to the master branch, how can you make sure your commit works off from the latest information in the master branch?



Exercise 15.4 Find a spelling error in the Web page or a contribution and create a pull request for it.



Exercise 15.5 Create a README.yml in your github account directory provided for you for class. information





16. Virtual Box

F section/virtualbox.tex

For development purposes we recommend that you use for this class an Ubuntu virtual machine that you set up with the help of virtualbox. We recommend that you use the current version of ubuntu and do not install or reuse a version that you have set up years ago.

As access to cloud resources requires some basic knowledge of linux and security we will restrict access to our cloud services to those that have demonstrated responsible use on their own computers. Naturally as it is your own computer you must make sure you follow proper security. We have seen in the past students carelessly working with virtual machines and introducing security vulnerabilities on our clouds just because "it was not their computer." Hence, we will allow using of cloud resources only if you have demonstrated that you responsibly use a linux virtual machine on your own computer. Only after you have successfully used ubuntu in a virtual machine you will be allowed to use virtual machines on clouds.

A "cloud drivers license test" will be conducted. Only after you pass it we will let you gain access to the cloud infrastructure. We will announce this test. Before you have not passed the test, you will not be able to use the clouds. Furthermore, you do not have to ask us for join requests to cloud projects before you have not passed the test. Please be patient. Only students enrolled in the class can get access to the cloud.

If you however have access to other clouds yourself you are welcome to use them. However, be reminded that projects need to be reproducible, on our cloud. This will require you to make sure a TA can replicate it.

Let us now focus on using virtual box.

16.1 Installation

First you will need to install virtualbox. It is easy to install and details can be found at

- <https://www.virtualbox.org/wiki/Downloads>

After you have installed virtualbox you also need to use an image. For this class we will be using ubuntu Desktop 16.04 which you can find at:

- <http://www.ubuntu.com/download/desktop>

Please note some hardware you may have may be too old or has too little resources to be useful. We have heard from students that the following is a minimal setup for the desktop machine:

- multi core processor or better allowing to run hypervisors
- 8 GB system memory
- 50 GB of free hard drive space

For virtual machines you may need multiple, while the minimal configuration may not work for all cases.

As configuration we often use

minimal 1 core, 2GB Memory, 5 GB disk

latex 2 core, 4GB Memory, 25 GB disk

A video to showcase such an install is available at:

[Video \(seconds\)](#) 

Note

If you specify your machine too small you will not be able to install the development environment. Gregor used on his machine 8GB RAM and 25GB diskspace.

Please let us know the smalest configuration that works.

16.2 Guest additions

The virtual guest additions allow you to easily do the following tasks:

- Resize the windows of the vm
- Copy and paste content between the Guest operating system and the host operating system windows.

This way you can use many native programs on you host and copy contents easily into for example a terminal or an editor that you run in the Vm.

A video is located at

[Video \(4:46\)](#) 

Please reboot the machine after installation and configuration.

On OSX you can once you have enabled bidirectional copying in the Device tab with

OSX to Vbox: command c shift CONTROL v

Vbox to OSX: shift CONTROL v shift CONTROL v

Note

On Windows the key combination is naturally different. Please consult your windows manual. If you let us know TAs will add the information here.

16.3 Exercises

Exercise 16.1 Install ubuntu desktop on your computer with guest additions. ■

Exercise 16.2 Make sure you know how to paste and copy between your host and guest operating system. ■

Exercise 16.3 Install the programs defined by the development configuration. ■

Exercise 16.4 Provide us with the key combination to copy and paste between Windows and Vbox. ■

VI

Cloud Resources

17	FutureSystems	181
17.1	FutureSystems evolved from FutureGrid	
17.2	Creating Portal Account	
17.3	SSH Key Generation using ssh-keygen command	
17.4	Shell Access via SSH	
17.5	Advanced SSH	
17.6	SSH Key Generation via putty	
18	Chameleon Cloud	183
18.1	Overview	
18.2	Resources	
18.3	Hardware	
18.4	Getting Started	
18.5	Charge Rates	
18.6	OpenStack Virtual Machines	
18.7	Horizon Graphical User Interface	
18.8	HEAT	
18.9	Bare Metal	
18.10	Frequently Asked Questions	



17. FutureSystems

F section/futuresystems.tex

This section gives an overview of the FutureSystems that are available as part of the DSC infrastructure. We cover the creation of FutureSystems Account, Uploading SSH Key and how to instantiate and log into Virtual Machine and accessing Ipython are covered. In the end we discuss about running Python and Java on Virtual Machine.

17.1 FutureSystems evolved from FutureGrid

In this video we introduce FutureGrid a precursor to FutureSystems.

[FutureGrid \(12:12\)](#) 

At this time we are replacing several of the older systems. To use these new systems you need to ask for access through them via our portal.

17.2 Creating Portal Account

This lesson explains how to create a portal account, which is the first step in gaining access to FutureSystems.

See Lesson 4 and 7 for SSH key generation on Linux, OSX or Windows.

[FutureGrid Introduction \(11:50\)](#) 

section/futuresystems.tex

17.3 SSH Key Generation using ssh-keygen command

SSH keys are used to identify user accounts in most systems including FutureSystems. This lesson walks you through generating an SSH key via ssh-keygen command line tool.

[ssh-key gen \(4:06\)](#) 

17.4 Shell Access via SSH

This lesson explains how to get access FutureSystems resources via SSH terminal with your registered SSH key.

[Shell Access via SSH \(2:34\)](#) 

17.5 Advanced SSH

This lesson shows you how to write SSH ‘config’ file in advanced settings.

[Advanced SSH \(2:47\)](#) 

17.6 SSH Key Generation via putty

This lesson is for Windows users. You will learn how to create an SSH key using PuTTYgen, add the public key to your FutureSystems portal, and then login using the PuTTY SSH client.

[Windows users \(3:51\)](#) 



18. Chameleon Cloud

18.1 Overview

 part/chameleon.tex

This chapter is copied from the Chameleon Web page. However we have included where appropriate some updates.

Chameleon is an experimental testbed for Computer Science funded by the NSF FutureCloud program. Chameleon is built over two sites, University of Chicago and TACC, offering a total of over 550 nodes and 5 PB of space in twelve [Standard Cloud Unit \(SCU\) racks](#). To effectively support Computer Science experiments Chameleon offers bare metal reconfigurability on most of the hardware. To provide easy access to educational users, three SCUs at TACC (a quarter of the testbed) are configured with OpenStack KVM. You can read more about Chameleon [here](#).

Chameleon is broadly available to members of the US Computer Science research community and its international collaborators working in the open community on cloud research. The expectation is that any research performed on Chameleon will result in publication in a broadly available journal or conference.

In order to promote fairness to all users, we have the following set of Best Practices for using Chameleon bare metal partitions:

Think Small for Development and class use: If you are just developing or prototyping a system, and not yet running experiments at scale, use only as many nodes as you actually need (e.g., many projects can be developed and tested on 3-4 nodes), and try to take short reservations. Start with hours first and do not let your VMs unnecessarily run for long unused periods.

Automate deployments: You can always snapshot your work/images between sessions using the [snapshotting instructions](#) to simplify the redeployment of your environment during the next

work session. You can also use scripting and environment customization to make it easier to redeploy images. An additional benefit of automation is that it makes it easier for you to reproduce your work and eventually share it with colleagues within your lab and other collaborators.

Think Big for Experimentation: Once you are ready to experiment you will want to test your experimental setup on increasingly larger scales. This is possible by taking an advance reservation for many resources for a relatively short time. The more resources you need, the more likely it is that you will need to run experiments at a less attractive time (e.g., during the weekend) — here's where automation will also help. In justified cases, we will support reserving even the whole bare metal testbed.

 section/cloud/chameleon/resources.tex

18.2 Resources

In any case please visit the Chameleon Web page as there is also more information about other topics that we may not care about. Furthermore we do not use Advanced Appliances, but use ansible instead as it is independent from OpenStack and can be used with other frameworks.

If you prefer you can also go to the Chameleon Web site using the following links. However we have improved some of the documentation found in this document. We would like to get your feedback in case you find errors or like to contribute to this documentation.

Warning

Chameleon cloud promotes insecure use of ssh while suggesting passphrase less keys. This is **very dangerous** due to the fact that someone could gain access to your computer and if a password less key is stolen easy access to other systems can be achieved. Instead you must use whenever possible passphrases and use ssh agent and ssh add!

Hence do not use their advise that is mentioned multiple times in their documentation. Follow ours!

The links to Chameleons Documentation

- [Home](#)
- [Documentation](#)
 - [Getting Started](#)
 - [Bare Metal](#)
 - [Complex Appliances](#)
 - [OpenStack KVM Cloud](#)
 - [User FAQ](#)
 - [Community](#)
- [Appliances](#)
- [Hardware](#)
- [News](#)
- [About](#)
 - [About Chameleon](#)
 - [Hardware Description](#)
 - [Talks](#)
 - [Stay in Touch](#)
 - [Media Resources](#)

- [Log in](#)
- [Users](#)
 - [Register](#)
 - [Experiment](#)
 - [Help](#)

18.3 Hardware



[section/cloud/chameleon/hardware.tex](#)

The Chameleon architecture consists of a set of standard cloud units (SCUs), each of which is a single rack with 42 compute nodes, 4 storage nodes attached to 128TB of local storage in configurable arrays, and an OpenFlow compliant network switch. In addition to the homogeneous SCUs, a variety of heterogeneous hardware types is available to experiment with alternative technologies. The testbed also includes a shared infrastructure with a persistent storage system accessible across the testbed, a top-level network gateway to allow access to public networks, and a set of management and provisioning servers to support user access, control, monitoring and configuration of the testbed. Chameleon is physically distributed between the Texas Advanced Computing Center (TACC) and the University of Chicago (UC) through 100Gbps Internet2 links, to allow users to examine the effects of a distributed cloud.

Hardware Summary

Standard Cloud Units (SCUs)	Homogeneous Hardware Types
Number of Nodes per Rack:	42 Compute Nodes
	4 Storage Nodes
Local Storage per homogeneous SCU:	128TB (configurable)
Network Switch:	OpenFlow Compliant
TACC/UC Distributed Cloud	100Gbps Internet2 links

[Detailed information in our Resource Discovery Portal](#)

18.3.1 Standard Cloud Units

The homogeneous standard cloud unit is a self-contained rack with all the components necessary to run a complete cloud infrastructure, and the capability to combine with other units to form a larger experiment. The rack consists of 42 Dell R630 servers; each with 24 cores delivered in dual socket Intel Xeon E5-2670 v3 “Haswell” processors (each with 12 cores @ 2.3GHz) and 128 GiB of RAM. In addition to the compute servers, each unit contains storage hosted in two FX2 chassis, each containing two Dell FC430 servers attached to two Dell PowerEdge FD332 storage blocks containing 16 2TB hard drives, for a total of 128TB of raw disk storage per unit. These FC430 storage nodes contain dual socket Intel Xeon E5-2650 v3 “Haswell” processors (each with 10 cores @ 2.3 GHz), 64 GiB of RAM, and can be combined across SCUs to create a Big Data infrastructure with more than a PB of storage. Each node in the SCU connects to a Dell switch at 10Gbps, with 40Gbps of bandwidth to the core network from each SCU. The total system contains 12 SCUs (10 at TACC and 2 at UC) for a total of 13,056 cores, 66 TiB of RAM, and 1.5PB of configurable storage in the SCU subsystem.



Figure 18.1: Chameleon Cloud Racks

18.3.2 Network

Networking is changing rapidly, and the network fabric is as much a part of the research focus of Chameleon as the compute or storage. For the Chameleon network, every switch in the research network is a fully OpenFlow compliant programmable Dell S6000-ON switch. Each node connects to this network at 10 Gbps, and each unit uplinks with 40Gbps per rack to the Chameleon core network. The core switches (Dell S6000-ON) are connected by 40 Gbps Ethernet links, which connect to the backbone 100Gbps services at both UC and TACC. A Fourteen Data Rate (FDR) Infiniband network (56Gbps) is also deployed on one SCU to allow exploration of alternate networks.

18.3.3 Shared Storage

While storage is dynamically provisioned to researchers to be used as an experiment needs within the SCUs, Chameleon also provides a shared storage system. The shared storage provides more than 3.6PB of raw disk in the initial configuration, which is partitioned between a file system and an object store that is persistent between experiments. The shared storage is comprised of four Dell R630 servers with 128 GiB of RAM, four MD3260 external drive arrays, and six MD3060e drive expansion chassis, populated by 600 6TB near line SAS drives. The system also includes a dozen PowerEdge R630 servers as management nodes to provide for login access to the resource, data staging, system monitoring, and hosting various OpenStack services.

18.3.4 Heterogeneous Compute Hardware

The heterogeneous hardware includes various technologies: GPU and FPGA accelerators, SSD and NVMe storage, low-power ARM, Atom, and Xeon systems-on-a-chip. With the exception of the low-power systems-on-a-chip, each of the additional nodes is a Dell PowerEdge R730 server with the same CPUs as the R630 servers in our SCUs.

The two storage hierarchy nodes have been designed to enable experiments using multiple layers of caching: they are configured with 512 GiB of memory, two Intel P3700 NVMe of 2 TB each, four Intel S3610 SSDs of 1.6 TB each, and four 15K SAS HDDs of 600 GB each.

The GPU offering consists of two K80 GPU nodes, two M40 GPU nodes, sixteen P100 GPU nodes. These nodes target experiments using accelerators to improve the performance of some algorithms, experiments with new visualization systems, and deep machine learning. Each K80 GPU node is upgraded with an NVIDIA Tesla K80 accelerator, consisting of two GK210 chips with 2496 cores each (4992 cores in total) and 24 GiB of GDDR5 memory. Each M40 node is upgraded with an NVIDIA Tesla M40 accelerator, consisting of a GM200 chip with 3072 cores and 12 GiB of GDDR5 memory. The P100 nodes have two GPU cards installed each, providing 32 P100 GPUs in total. The P100 GPUs utilize GP100 chips providing 3584 cores, with 16 GiB GDDR5 RAM in each card. In order to make it easy for users to get started with the GPU nodes, we have developed a [CUDA appliance](#) that includes NVIDIA drivers as well as the CUDA framework.

GPU	Chip	Cores per GPU	RAM per GPU	GPU per node	# of nodes
Tesla K80	GK 210	2496 x 2	24 GiB GDDR5	1	2
Tesla M40	GM 200	3072	12 GiB GDDR5	1	2
Tesla P100	GP100	3584	16 GiB GDDR5	2	16

The four FPGA nodes have a Nallatech 385A board with an Altera Arria 10 1150 GX FPGA (up to 1.5 TFlops), 8 GiB DDR3 on-card memory, and dual QSFP 10/40 GbE support. The Chameleon [FPGA User Guide](#) provides details for conducting experiments on this hardware.

The low-power systems are comprised of 8 low power Xeon servers (HP ProLiant m710p with one 4-core Intel Xeon E3-1284L v4 processor), 8 Atom servers (HP ProLiant m300 with one 8-core Intel Avoton-based System on a Chip), and 24 ARM servers (HP ProLiant m400 with one 8-core AppliedMicro X-gene System on a Chip). These are all delivered in a single HP Moonshot 1500 chassis.

For more information on how you can reserve these nodes, see the [heterogeneous hardware section](#) of the bare metal user's guide.

18.3.5 Live updates

You can browse detailed information about the resources offered for bare metal reconfiguration in our [Resource Discovery Portal](#).

18.4 Getting Started



[section/cloud/chameleon/start.tex](#)

We describe how you can get access to chameleon cloud under the assumption that you are a student

[section/cloud/chameleon/start.tex](#)

or a researcher that joins an existing project on Chameleon cloud. You will need to follow the following steps:

18.4.1 Step 1: Create a Chameleon account

To get started using Chameleon you will need to [create a user account](#).

You will be asked to agree to the [Chameleon terms and conditions](#) which, among others, ask you to acknowledge the use of Chameleon in your publications.

Acknowledgement of support from the Chameleon project and the National Science Foundation should appear in any publication of material, whether copyrighted or not, that describes work which benefited from access to Chameleon cyberinfrastructure resources. The suggested acknowledgement is as follows: “Results presented in this paper were obtained using the Chameleon testbed supported by the National Science Foundation”.

Indiana University

As part of creating an account you may request PI status. However you are not a PI as you will be joining a project.

18.4.2 Step 2: Create or join a project

To use Chameleon, you will need to be associated with a [project](#) that is assigned an [allocation](#). This means that you either need to

1. [apply for a new project](#) or
2. [ask the PI of an existing Chameleon project to add you](#).

A project is headed by a project PI, typically [a faculty member or researcher scientist at a scientific institution](#). If you are a student we recommend that you ask your professor to work with you on creating a project. Please note that you must not create a project by yourself and that you indeed need to work with your professor.

In case you need to do a project application typically consists of about one paragraph description of the intended research and takes one business day to process.

Enrolling you into an existing research or class project depends on the time availability of the project lead or professor of your class. It is important that you communicate your chameleon cloud account name to the project lead so they can easily add you. Make sure you really give them only your chameleon account name and potentially your organizational e-mail, Firstname, and Lastname so they can check you are eligible to get access.

Indiana University

Indiana University students that take the e516 and e616 classes will have to fill out a google form in which they communicate the chameleon cloud name. You can already apply for an account name, but do not apply for a project. If you nevertheless apply for a project, we will hear from the administrators and you will receive a point deduction.

18.4.3 Step 3: Start using Chameleon!

Now that you have enrolled and once you are added to the project by your project lead you can start using chameleon cloud. However be reminded that you ought to shut down the resources/VMs

whenever they are not in use to avoid unnecessary charging. Remember the project has limited time on chameleon and any unused time will be charged against the project.

Chameleon provides two types of resources with links to their respective users guides below:

Bare Metal User Guide will tell you how to use Chameleon bare metal resources which provide strong isolation and allow you maximum control (reboot to new operating system, reboot the kernel, etc.)

OpenStack KVM User Guide will tell you how to get started with Chameleon's OpenStack KVM cloud which is a multi-tenant environment providing weak performance isolation.

If you have any questions or encounter any problems, you can check out our [User FAQ](#), or [submit a ticket](#).

Indiana University

As part of the classes you will need to first pass a cloud *security* drivers licence test. The test is designed so that you think about gaining access to a VM securely and how to properly secure the VM. Once passed, access is typically provided by midterm time. You are not allowed to constantly run VM's and must shut them down if not in use. You will get point deductions if we detect you do not obey by this rule. We have access to log files about your VM usage.

18.5 Charge Rates

It is important to fully understand the charge rates of your VM and storage use.

Chameleon has two types of limitations, introduced to promote fair resource usage to all:

Allocation: Chameleon projects are limited to a per-project allocation currently set to 20,000 service units for 6 months. Allocations can be renewed or extended (see [Project and Allocation Management](#) section for more details on Chameleon allocations.)

Lease: To ensure fairness to all users, resource reservations (leases) are limited to a duration of 7 days. However, an active lease within 48 hours of its end time can be prolonged by up to 7 days from the moment of request if resources are available. To prolong a lease, click on the "Update Lease" button in the Reservations panel of the CHI OpenStack dashboard, and enter the additional duration requested in the "Prolong for" box including the unit suffix, e.g. "5d" for 5 days or "30m" for 30 minutes. If there is an advance reservation blocking your lease prolongation that could potentially be moved, you can interact through the users mailing list to coordinate with others users. Additionally, if you know from the start that your lease will require longer than a week and can justify it, you can [contact Chameleon staff via the ticketing system](#) to request a one-time exception to create a longer lease. The lease must be requested by the PI.

18.5.1 Service Units

Chameleon allocations can consist of several components of the system. Users can request allocation of individual compute nodes, storage servers, or complete Scalable Compute Units (SCUs) which contain compute servers, storage nodes, and an open flow switch.

Compute servers are allocated in Service Units (SUs), which equates to one hour of wall clock time on a single server (for virtual machines, an SU is 24 cores with up to 128GB of RAM). Note this unit differs from traditional HPC or cloud service units that are charged in core-hours; a Chameleon SU is a full server, as the type of experiments and performance measurements users may wish to do

may be contaminated by sharing nodes.

Storage servers are also charged in SUs, at 2x the rate of compute servers (i.e., 1 hour allocation of 1 storage server == 2 SUs). SCUs are charged at the rate of 50 SUs per wall clock hour (42 compute servers, 4 storage nodes, plus one OpenFlow switch).

An allocation may make use of multiple SCUs, up to the size of the full testbed.

For example, a user wishing to provision a 10 node cluster +1 storage server for a 1 week experiment should budget $[(10 + 2) \text{ SUs per hour}] * [7 \text{ days} * 24 \text{ hours/day}] = 2,016 \text{ SUs}$ for that experiment.

SUs are charged the same regardless of use case. Hence, whether asking for bare metal access, virtual machine access, or use of default images, the charge is the same — you are charged for the fraction of the resource your experiment occupies, regardless of the type of the experiment.

The basic principle for charging service units for Chameleon resources is to evaluate the amount of time a fraction of the resource is unavailable to other users. If a reservation is made through the portal for a particular date/time in the future, the user will be charged for this time regardless of whether the reservation is actually used, as the Chameleon scheduling system will have to drain the appropriate part of the system to satisfy the reservation, even if the nodes requested are not actually used. A reservation request may be cancelled in which case no charges will apply.

18.5.2 Project Allocation Size

Currently Chameleon is operating on a “soft allocation model” where each project, if approved, will receive a startup allocation of 20,000 SUs for six months that can be both recharged (i.e., more SUs can be added) and renewed (i.e., the duration can be extended) via submitting a renew/recharge request. This startup allocation value has been designed to respond to both PI needs (i.e., cover an amount of experimentation needed to obtain a significant result) and balance fairness to other users (it represents roughly 1% of testbed six months’ capacity). Requests for these startup projects will receive a fast track internal review (i.e., users can expect them to be approved within a few days).

A PI can apply for multiple projects/allocations; however, the number of held allocations will be taken into account during review.

As our understanding of user need grows we expect the Chameleon allocation model to evolve towards closer reflection of those needs in the form of more differentiated allocations that will allow us to give larger allocations to users for longer time.

Indiana University

Please be mindful to shutting down your VMS when not in use as even VMs that do not do any calculations get charged. In past classes we had students that did not shut down their VMs and within 2 weeks used up all SUs for the entire class of 70 students. We like to avoid this. In future cases we will assign the grade “F” to such students, as is customary also in other universities.

18.6 OpenStack Virtual Machines



section/cloud/chameleon/user-guide.tex

OpenStack is an Infrastructure as a Service (IaaS) platform that allows you to create and manage virtual environments. Chameleon provides an installation of OpenStack version 2015.1 (Kilo) using

the KVM virtualization technology.

Since the KVM hypervisor is used on this cloud, any virtual machines you upload must be compatible with KVM.

This tutorial provide basic information about how to use the OpenStack web interface and provides some information specific to using OpenStack KVM on Chameleon.

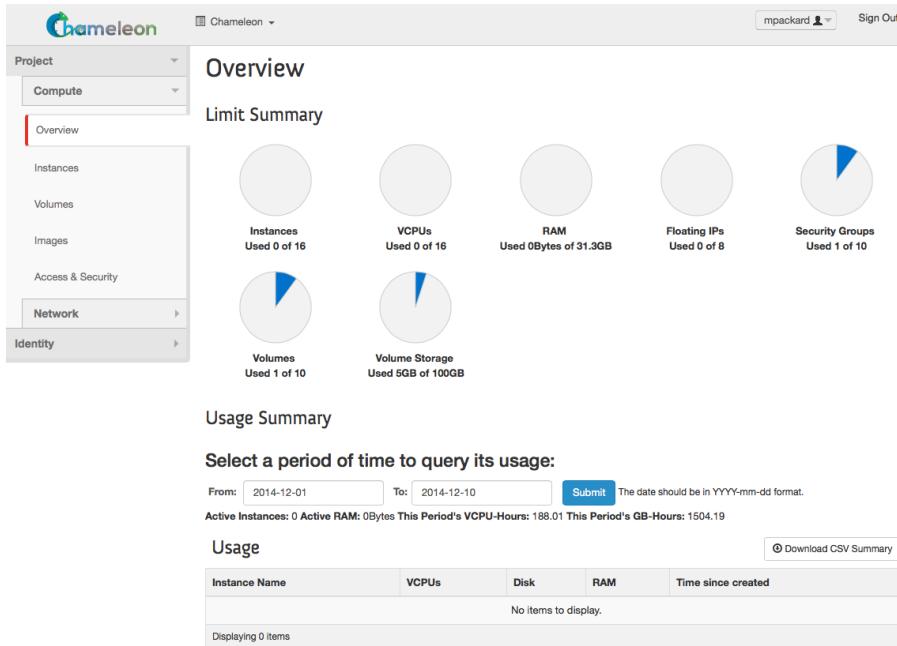
18.6.1 Web Interface

An easy way to use OpenStack KVM on Chameleon is via the [OpenStack web interface](#) also known as Horizon. You log into the web interface using your Chameleon username and password. If you change your Chameleon password in the portal, that change will propagate to the OpenStack KVM interface in about 5 minutes.

The initial log in page appears as:

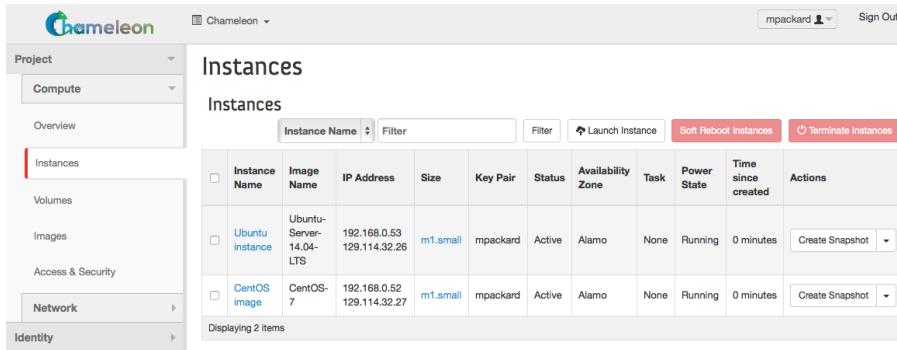


After a successful log in, you will see the Overview page as shown below. This page provides a summary of your current and recent usage and provides links to various other pages. Most of the tasks you will perform are done via the menu on the lower left and will be described below. One thing to note is that on the left, your current project is displayed. If you have multiple Chameleon projects, you can change which of them is your current project. All of the information displayed and actions that you take apply to your current project. So in the screen shot below, the quota and usage apply to the current project you have selected and no information about your other projects is shown.



Managing Virtual Machine Instances

One of the main activities you'll be performing in this web interface is the management of virtual machines, or instances. You do this via the Instances page that is reachable from the menu in the lower left of the Overview page. An example Instances page is shown below. For instances that you have running, you can click on the name of the instance to get more information about it and to access the VNC interface to the console. The dropdown menu to the left of the instance lets you perform a variety of tasks such as suspending, terminating, or rebooting the instance.



The Instances page also lets you create new virtual machines by using the ‘Launch Instance’ button in the upper-right. When you click this button, a dialog window pops up. In the first ‘Details’ tab, you select the ‘Instance Boot Source’ of the instance, which is either an ‘Image’, a ‘Snapshot’ (an image created from a running virtual machine), or a ‘Volume’ (a persistent virtual disk that can be attached to a virtual machine). If you select ‘Boot from image’, the Image Name dropdown presents a list of virtual machine images that we have provided, that other Chameleon users have uploaded and made public, or images that you have uploaded for yourself. If you select ‘Boot from snapshot’, the Instance Snapshot dropdown presents a list of virtual machine images that you have created from your running virtual machines.

On the Details tab, you also provide a name for this instance (to help you identify instances that you are running), and select the amount of resources (Flavor) to allocate to the instance. If you

select different flavors from the Flavor dropdown, their characteristics are displayed on the right.

Launch Instance

Details * Access & Security * Networking * Post-Creation * Advanced Options

Availability Zone
Alamo

Instance Name *
CentOS image

Flavor * m1.small
Some flavors not meeting minimum image requirements have been disabled.

Instance Count * 1

Instance Boot Source * Boot from image

Image Name CentOS-7 (329.2 MB)

Flavor Details

Name	m1.small
VCPUs	1
Root Disk	20 GB
Ephemeral Disk	0 GB
Total Disk	20 GB
RAM	2,048 MB

Project Limits

Number of Instances	2 of 16 Used
Number of VCPUs	2 of 16 Used
Total RAM	4,096 of 32,000 MB Used

Cancel Launch

The next tab is ‘Access & Security’, where you select an SSH keypair that will be inserted into your virtual machine. These keypairs can be uploaded via the main ‘Access & Security’ section. You will need to select a keypair here to be able to access an instance created from one of the public images Chameleon provides. These images are not configured with a default root password and you will not be able to log in to them without configuring an SSH key.

Launch Instance

Details * Access & Security * Networking * Post-Creation * Advanced Options

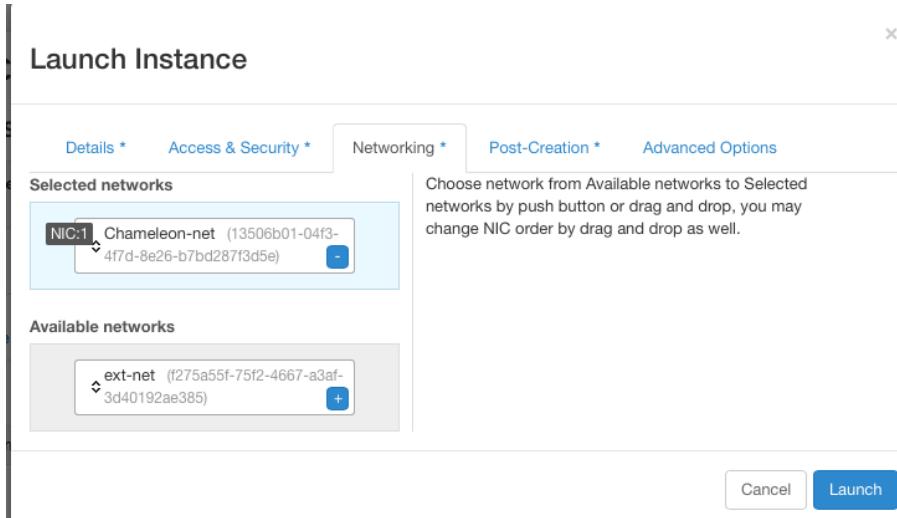
Key Pair mpackard

Security Groups default

Control access to your instance via key pairs, security groups, and other mechanisms.

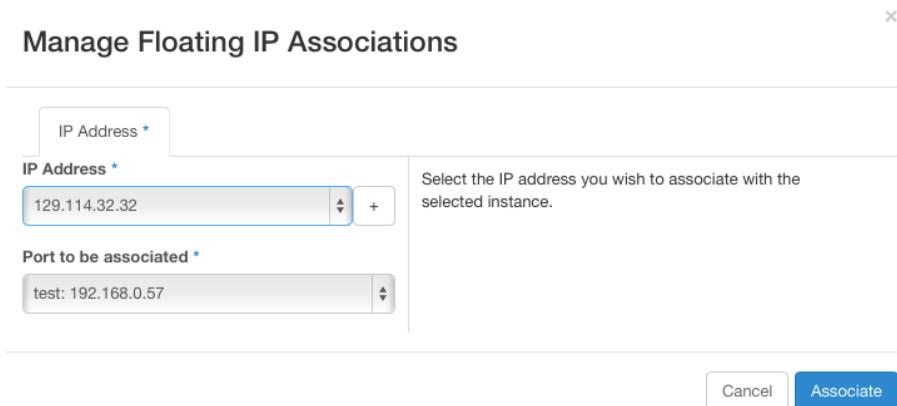
Cancel Launch

Next is ‘Networking’, where you select which network should be associated with the instance. Click the + next to your project’s private network (PROJECT_NAME-net), not ext-net.



Once you do this, you can Launch your instance and the Instances page will show progress as it starts.

If you would like to assign a public IP address to your VM, you can do that while it is booting up. Click on the dropdown under *Actions* and choose *Associate Floating IP*. Choose an IP from the *IP Address* menu and click *Associate*. If there are no addresses available, click the + and follow the prompts to add one.



OpenStack injects your SSH key into the VM and you can use the corresponding private SSH key to log in to the VM. You will need to use the public IP assigned to your VM to connect from outside of Chameleon, or connect through an existing instance that both a public and private IP.

Note that the images we provide do not allow SSH into the root account. For root access, SSH into the instance as user 'cc' and then use the sudo command to become root.

We have enabled auto-login for the cc user on the console of our supported images. This should aid in debugging if you are unable to reach the instance via ssh for some reason.

If console is not responding to keyboard input: click the grey status bar below. [Click here to show only console](#)
To exit the fullscreen mode, click the browser's back button.

```
Connected (encrypted) to: QEMU (instance-0000026d)

CentOS Linux 7 (Core)
Kernel 3.10.0-123.6.3.el7.x86_64 on an x86_64
cc-demo login: cc (automatic login)
Last login: Thu Feb 19 22:25:51 on ttys0
[cc@cc-demo ~]$ [cc@cc-demo ~]$ uname -a
Linux cc-demo 3.10.0-123.6.3.el7.x86_64 #1 SMP Wed Aug 6 21:12:36 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
[cc@cc-demo ~]$ _
```

Snapshots

The instance list page shown above has an option ‘Create Snapshot’ that allows you to save a copy of the disk contents of a running virtual machine. This allows you to start new virtual machines in the future that are identical to this one and is an easy way to save any changes you make to a running virtual machine.

Firewall (Access Security)

Each project has control over their own firewall settings for their instances. At minimum you’ll probably want to allow SSH access so you can reach your instances.

To enable this traffic, you need to configure the security group used by your virtual machine. You can see a list of your security groups using the “Access & Security” link on the left.

Access & Security

<input type="checkbox"/>	Name	Description	Actions
<input type="checkbox"/>	default	default	Manage Rules

Displaying 1 item

To edit a security group, click on “Edit Rules”. This opens a page showing the existing rules in the security group.

Manage Security Group Rules: default

Security Group Rules

<input type="checkbox"/>	Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
<input type="checkbox"/>	Ingress	IPv4	Any	-	default	<button>Delete Rule</button>
<input type="checkbox"/>	Egress	IPv6	Any	-	::/0 (CIDR)	<button>Delete Rule</button>
<input type="checkbox"/>	Egress	IPv4	Any	-	0.0.0.0/0 (CIDR)	<button>Delete Rule</button>
<input type="checkbox"/>	Ingress	IPv6	Any	-	default	<button>Delete Rule</button>
<input type="checkbox"/>	Ingress	IPv4	ICMP	-	0.0.0.0/0 (CIDR)	<button>Delete Rule</button>
<input type="checkbox"/>	Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0 (CIDR)	<button>Delete Rule</button>

Displaying 6 items

Click on “Add Rule” and choose the *SSH* rule from the list, and click *Add*. Modifications are automatically propagated to the OpenStack cloud. Feel free to add other rules as necessary.

Add Rule

Rule *

Remote * ⓘ

CIDR ⓘ

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the “Port Range” option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel
Add

18.6.2 OpenStack REST Interfaces

The OpenStack REST Interfaces are supported on Chameleon over secure HTTP connections. You can download your OpenStack credentials file from the web interface via the “Access & Security” link in the left of any page and then click on the “API Access” link on the top.

You can then install the OpenStack command line clients following [these instructions](#). If using pip, we recommend setting up a virtualenv.

The SSL certificate used by Chameleon is trusted by most operating systems, so you shouldn’t have to provide any extra options to OpenStack commands, i.e. “nova list” should work. If your command-line tool complains about the certificate, [download the Mozilla CA bundle from the CURL website](#) and run the OpenStack client tools with the –os-cacert cacert.pem arguments.

18.6.3 Downloading and uploading data

You can use the OpenStack command line clients to download data from and upload data to Chameleon clouds. Configure your environment by following the “OpenStack REST Interfaces” section above, then use the following commands:

- `glance image-download` to download images and snapshots from Glance
- `glance\ image-create` to upload images and snapshots to Glance
- `cinder upload-to-image` to convert a Cinder volume to a Glance image
- `cinder create [--image-id <image-id>] [--image <image>]` to create a Cinder volume from a Glance image

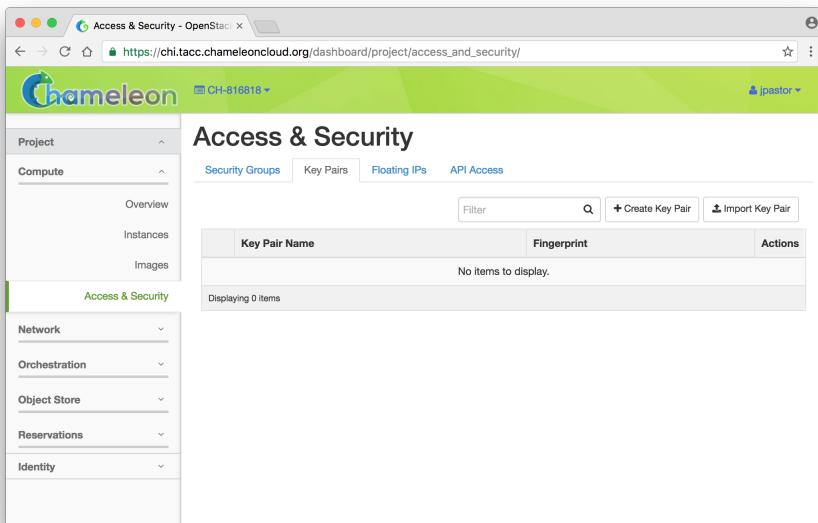
 section/cloud/chameleon/horizon.tex

18.7 Horizon Graphical User Interface

18.7.1 Configure resources

Once your lease is started, you are almost ready to start an instance. But first, you need to make sure that you will be able to connect to it by setting up a key pair. This only has to be done once per user per project.

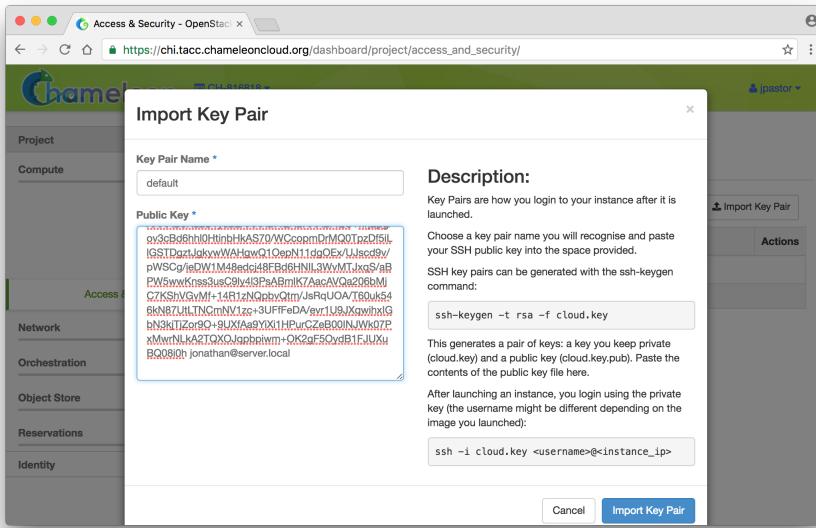
Go to Project > Compute > Access & Security, then select the Key Pairs tab.



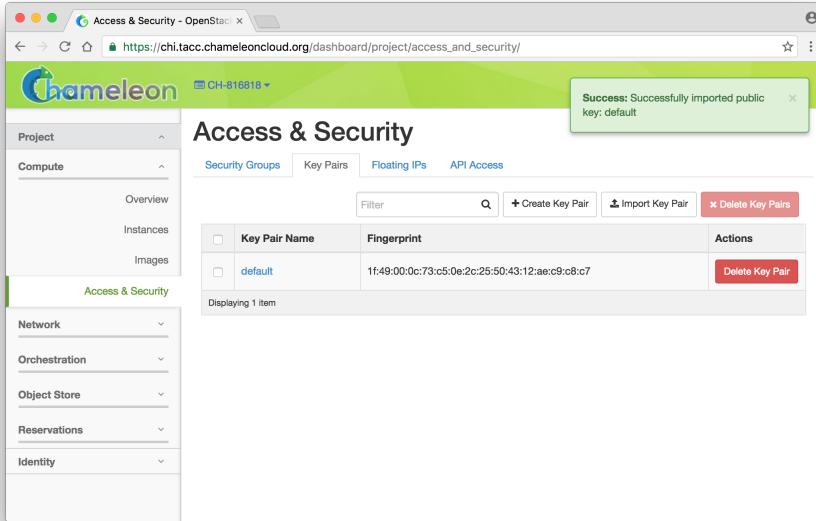
Here you can either ask OpenStack to create an SSH key pair for you (via the “Create Key” Pair button), or, if you already have an SSH key pair on your machine and are happy to use it, click on “Import Key Pair”.

If you chose to import a key pair, you will be asked to enter a name for the key pair, for example `laptop`. In the “Public Key” box, copy the content of your SSH public key. Typically it will be at `~/.ssh/id_rsa.pub`. On Mac OS X, you can run in a terminal: `cat ~/ssh/id_rsa.pub | pbcopy`

It copies the content of the public key to your copy/paste buffer. Then you can simply paste in the “Public Key” box.



Then, click on the blue “Import Key Pair” button. This should show you the list of key pairs, with the one you just added.



For those already familiar with OpenStack, note that Security Groups are not functional on bare-metal. All instances ports are open to the Internet and any security group rule you add will not be respected.

Now, go to the “Instances” panel.

Click on the “Launch Instance” button in the top right corner. Select a reservation in the Reservation box, pick an instance name (in this example my-first-instance) and in the Image Name list select our default environment named CC-CentOS7. If you have multiple key pairs registered, you need to select one in the “Access & Security” tab. Finally, click on the blue “Launch” button.

Name	baremetal
VCPUs	24
Root Disk	200 GB
Ephemeral Disk	0 GB
Total Disk	200 GB
RAM	128,000 MB

The instance will show up in the instance list, at first in Build status. It takes a few minutes to deploy the instance on bare-metal hardware and reboot the machine.

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
my-first-instance	CC-CentOS7		baremetal	default	Build		Scheduling	No State	0 minutes	<button>Associate Floating IP</button>

After a few minutes the instance should become in Active status and the Power State should be Running.

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions	
my-first-instance	CC-CentOS7	10.40.0.164	baremetal	default	Active	climate:fd0b6542-4851-4d2e-a11-0441732cecc0		None	Running	4 minutes	<button>Associate Floating IP</button>

At this point the instance might still be booting: it might take a minute or two to actually be accessible on the network and accept SSH connections. In the meantime, you can attach a floating IP to the instance. Click on the “Associate Floating IP” button. You should get a screen like the one below:

The screenshot shows a modal dialog titled "Manage Floating IP Associations". It contains three input fields: "IP Address *", "IP Address *", and "Port to be associated *". The "IP Address *" field has a dropdown menu open, showing "Select an IP address" and a "+" button. The "Port to be associated *" field contains "my-first-instance: 10.40.0.164". A tooltip next to the "IP Address *" field says "Select the IP address you wish to associate with the selected instance or port". At the bottom right are "Cancel" and "Associate" buttons.

If there are no unused floating IP already allocated to your project, click on the + button. In the window that opens, select the ext-net pool if not already selected by default and click on the blue Allocate IP button.



You will be returned to the previous window. The correct value for “Port to be associated” should already be selected, so you only have to click on “Associate”.

The screenshot shows the same "Manage Floating IP Associations" dialog as before, but with the "IP Address *" field now containing "129.114.108.90". The "Associate" button is highlighted in blue at the bottom right.

This should send you back to the instance list, where you can see the floating IP attached to the instance (you may need to refresh your browser to see the floating IP).

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions	
my-first-instance	CC-CentOS7	10.40.0.164 Floating IP: 129.114.108.90		baremetal	default	Active	climatedfd0b6542-4851-4c2e-a11-0441732cecc0	None	Running	31 minutes	<button>Dissociate Floating IP</button>

18.7.2 Interact with resources

Now you should be able to connect to the instance via SSH using the cc account. In a terminal, type `ssh cc@<floating_ip>`, in our example this would be `ssh cc@130.202.88.241`

SSH will probably tell you:

```
The authenticity of host \textquotesingle{}130.202.88.241
(130.202.88.241) can't be established. RSA key fingerprint
is 5b:ca:f0:63:6f:22:c6:96:9f:c0:4a:d8:5e:dd:fd:eb.
Are you sure you want to continue connecting (yes/no)?
```

Type yes and press Enter. You should arrive to a prompt like this one:

```
[cc@my-first-instance ~]$
```

If you notice SSH errors such as connection refused, password requests, or failures to accept your key, it is likely that the physical node is still going through the boot process. In that case, please wait before retrying. Also make sure that you use the **cc** account. If after 10 minutes you still cannot connect to the machine, please [open a ticket with our help desk](#).

You can now check whether the resource matches its known description in the resource registry. For this, simply run: `sudo cc-checks -v`

```

priteau — cc@test-new-image:~ — ssh — 55x56
cc@test-new-image:~ [cc@test-new-image ~]$ cc-checks
Chassis
OK should have the correct serial number
OK should have the correct manufacturer
OK should have the correct product name

Disk
OK should have the correct name
OK should have the correct size
OK should have the correct model
OK should have the correct revision
OK should have the correct vendor

Virtual Hardware
OK should have the good driver

Memory
OK should have the correct size

Network
OK should be the correct interface name
OK should have the correct Driver
OK should have the correct Mac Address
OK should have the correct Rate
OK should have the correct version
OK should have the correct mounted mode
OK should not be a management card
OK should be the correct interface name
OK should have the correct Driver
OK should have the correct Mac Address
OK should have the correct Rate
OK should have the correct version
OK should have the correct mounted mode
OK should not be a management card

OS
OK should be the correct name
OK should be the correct kernel version
OK should be the correct version

Processor
OK should have the correct frequency
OK should be of the correct instruction set
OK should be of the correct model
OK should be of the correct version
OK should have the correct vendor
OK should have the correct description
OK should have the correct L1i
OK should have the correct L1d
OK should have the correct L2
OK should have the correct L3

Rights on /tmp
OK should have mode 41777
[cc@test-new-image ~]$ echo $?
0
[cc@test-new-image ~]$ 

```

The cc-checks program prints the result of each check in green if it is successful and red if it failed.

You can now run your experiment directly on the machine via SSH. You can run commands with root privileges by prefixing them with sudo. To completely switch user and become root, use the sudo su - root command.

Snapshot an instance

All instances in Chameleon, whether KVM or bare-metal, are running off disk images. The content of these disk images can be snapshotted at any point in time, which allows you to save your work and launch new instances from updated images later.

While OpenStack KVM has built-in support for snapshotting in the Horizon web interface and via the command line, bare-metal instances require a more complex process. To make this process easier, we developed the [cc-snapshot](#) tool, which implements snapshotting a bare-metal instance from command line and uploads it to Glance, so that it can be immediately used to boot a new bare-metal instance. The snapshot images created with this tool are whole disk images.

For ease of use, [cc-snapshot](#) has been installed in all the appliances supported by the Chameleon project. If you would like to use it in a different setting, it can be downloaded and installed from

the [github repository](#).

Once cc-snapshot is installed, to make a snapshot of a bare-metal instance, run the following command from inside the instance:

```
sudo cc-snapshot <snapshot_name>
```

You can verify that it has been uploaded to Glance by running the following command:

```
glance image-list
```

If you prefer to use a series of standard Unix commands, or are generally interested in more detail about image management, please refer to our [image management guide](#).

18.7.3 Use FPGAs

Consult the [dedicated page](#) if you would like to use the FPGAs available on Chameleon.

18.7.4 Next Step

Now that you have created some resources, it is time to interact with them! You will find instructions to the next step by visiting the following link:

- [Monitor resources and collect results](#)

18.8 HEAT

 [section/cloud/chameleon/heat.tex](#)

Deploying an MPI cluster, an OpenStack installation, or any other type of cluster in which nodes can take on multiple roles can be complex: you have to provision potentially hundreds of nodes, configure them to take on various roles, and make them share information that is generated or assigned only at deployment time, such as hostnames, IP addresses, or security keys. When you want to run a different experiment later you have to redo all this work. When you want to reproduce the experiment, or allow somebody else to reproduce it, you have to take very precise notes and pay great attention to their execution.

To help solve this problem and facilitate reproducibility and sharing, the Chameleon team configured a tool that allows you to deploy complex clusters with “one click”. This tool requires not just a simple image (i.e., appliance) but also a document, called a template, that contains the information needed to orchestrate the deployment and configuration of such clusters. We call this image + template combination complex appliance because it consists of more than just the image (i.e., appliance).

18.8.1 Supporting Complex Appliances

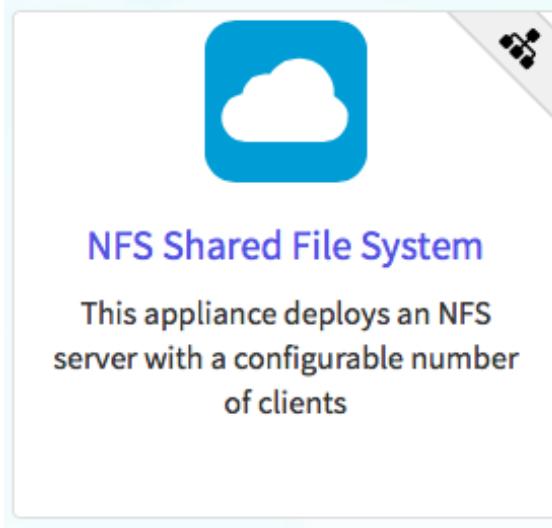
In a nutshell, complex appliances allow you to specify not only what image you want to deploy but also on how many nodes you want to deploy that image, what roles the deployed instances should boot into (such as e.g., head node and worker node in a cluster), what information from a specific instance should be passed to another instance in that complex appliance, and what scripts should be executed on boot so that this information is properly used for configuring the “one click” cluster. For example, a Network File System (NFS) appliance that we will use as an example in this guide,

might specify deployment on three nodes, out of which one will be configured as head node and others as worker nodes, the information passed between the images will be hostname of the head node, and the scripts executed on the worker nodes on boot will put that hostname in the fstab file. As you can tell from this description, images used for complex appliances are typically configured such that they can be booted into any role required on the one-click cluster we are booting; in this case the image will have both the software for NFS server node and client node.

Since complex appliances in Chameleon are currently implemented using the [OpenStack Heat](#) orchestration service, we will be using OpenStack terminology and features to work with them. The templates described above are YAML files using the [Heat Orchestration Template \(HOT\)](#) format (Heat also supports the AWS CloudFormation template format, but this is not covered here). A deployed complex appliance is referred to as a “stack” – just as a deployed single appliance is typically referred to as an “instance”. This guide will tell you all you need to know in order to use and configure complex appliances on Chameleon; if you would like to know more about Heat, please refer to its [official documentation](#).

18.8.2 Chameleon Appliance Catalog

Our [Appliance Catalog](#) has several complex appliances for popular technologies that people want to deploy such as OpenStack or MPI or even more advanced deployments such as efficient SR-IOV enabled MPI in KVM virtual machines. We also provide common building blocks for cluster architectures, such as an NFS share. Complex appliances are identified by a badge in their top-right corner representing a group of machines, as shown in the screenshot:



18.8.3 Deployment

We will explain how to launch a complex appliance based on our [NFS share appliance](#). To launch a complex appliance, you only need to follow these steps:

1. Create a lease: use the OpenStack web interface (choose between CHI@UC or CHI@TACC) to create a lease. To launch our NFS appliance, reserve at least three compute nodes (the strict minimum is two nodes but we will use three in this example and later ones).
2. Go to the [Appliance Catalog](#) and identify the appliance you want to launch. In our case you can go straight to the [NFS share appliance](#); click on it to open its details page. You will see a “Launch” button and a “Get Template” button. Follow the “Get Template” link and copy its

- url to the clipboard – you will need it in the following steps.
3. Click on the “Launch Complex Appliance at CHI@TACC” or “Launch Complex Appliance at CHI@UC” button depending on where your reservation was created.

This will take you to the Stacks page within the Orchestration menu. This page will show the current list of stacks, with controls to manage them and create new ones. Since we haven’t launched any yet, this list will be empty for now.

We will now create a new stack, which corresponds to the launch of a template. Click on Launch Stack on the top right. A window will pop up like below:

Select Template

Template Source *

File

Description:
Use one of the available template source options to specify the template to be used in creating this stack.

Template File ?
 no file selected

Environment Source

File

Environment File ?
 no file selected

We will deploy the NFS appliance described earlier; it will consist of a server node and two client nodes. Change the template source field to URL, and paste the URL of the [NFS share template](#) (if you don’t have it in your clipboard anymore you will need to go back to the appliance and get it by clicking on “Get template” again).

Don’t change the environment source settings, and click “Next”.

The next screen allows you to enter input values to your Heat template. Choose a name for your stack (e.g. my-nfs-cluster). Ignore the “Creation Timeout” and “Rollback On Failure” settings. You also need to enter your Chameleon password. Then, you need to select a value for the three parameters of the template: for key_name, choose your SSH key pair (this key pair will authorize access on each deployed instances, both server and client). For nfs_client_count, change the default value of 1 to 2. For reservation_id, choose your reservation created earlier. Finally, click “Launch”.

Launch Stack

Stack Name * [?](#)

Description:
Create a new stack with the provided values.

Creation Timeout (minutes) * [?](#)

Rollback On Failure [?](#)

Password for user "priteau" * [?](#)

key_name [?](#)

nfs_client_count [?](#)

reservation_id * [?](#)

Your stack should be in status “Create In Progress” for several minutes while it first launches the NFS server instance, followed by the NFS client instances.

<input type="checkbox"/>	Stack Name	Created	Updated	Status	Actions
<input type="checkbox"/>	my-nfs-cluster	0 minutes	Never	Create In Progress	<input type="button" value="Check Stack"/> ▼

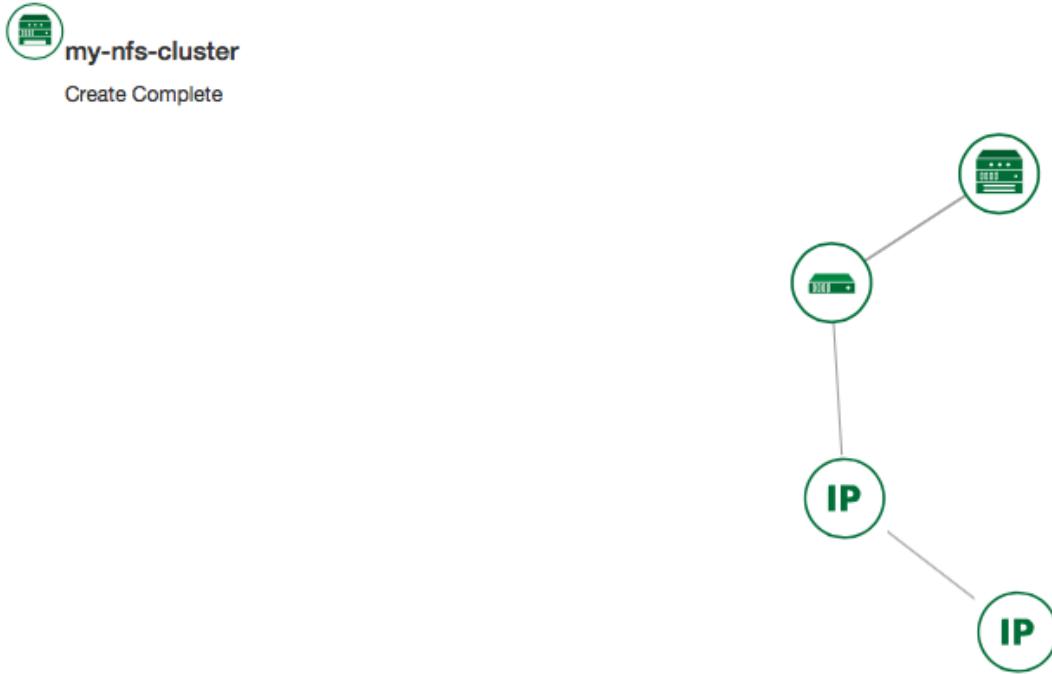
Displaying 1 item

It will then move to the status “Create Complete”.

<input type="checkbox"/>	Stack Name	Created	Updated	Status	Actions
<input type="checkbox"/>	my-nfs-cluster	15 minutes	Never	Create Complete	<input type="button" value="Check Stack"/> ▼

Displaying 1 item

You can click on the stack name to get more details, including a visualization of the deployed resources, as pictured below. The single machine inside a circle represents the NFS server instance. The rack of machine represents the group of NFS client instances (in this case, a group composed of two instances). The server’s floating IP (the public IP assigned to a resource) is represented by an IP in a circle; an IP in a circle is also used to represent the association of the IP with the NFS server instance (not the greatest idea to use the same symbol for both the IP and the association – we agree but can’t do much about it at the moment). Blow off some steam by dragging the visualization across the screen, it can be rather fun!



You can now ssh to the server using the floating IP just as you do with regular instances (use the cc account). The client does not have a floating IP attached to it (as per the visualization above) but you can connect to it via the server node with the client's private IP (connect to the server with `ssh -A` to enable the SSH agent forwarding after loading your key to your SSH agent with `ssh-add <path-to-your-key>`).

You can find out the information about the IPs and other things if you click the “Overview” tab and look in the “Outputs” section. Under the “Resources” tab you will see the resources described above (the server, clients, server’s public/floating IP, and its the association) and information about them. In the “Events” tab you will see information about the history of the deployment so far. In Template you will see the template that was used to deploy this stack.

18.8.4 Heat Template

The NFS share appliance deploys:

- an NFS server instance, that exports the directory `/exports/example` to any instance running on Chameleon bare-metal,
- one or several NFS client instances, which configure `/etc/fstab` to mount this NFS share to `/mnt` (and can subsequently read from and write to it).

This template is reproduced further below, and includes inline comments starting with the `#` character. There are three main sections:

- resources,
- parameters,
- outputs.

The resources section is the most important part of the template: it defines which OpenStack resources to create and configure. Inside this section you can see four resources defined:

- `nfs_server_floating_ip`

- nfs_server
- nfs_server_ip_association
- nfs_clients

The first resource, nfs_server_floating_ip, creates a floating IP on the ext-net public network. It is not attached to any instance yet.

The second resource, nfs_server, creates the NFS server instance (an instance is defined with the type OS::Nova::Server in Heat). It is a bare-metal instance (flavor: baremetal) using the CC-CentOS7 image and connected to the private network named sharednet1. We set the keypair to use the value of the parameter defined earlier, using the get_param function. Similarly, the reservation to use is passed to the scheduler. Finally, a user-data script is given to the instance, which configures it as an NFS server exporting /exports/example to Chameleon instances.

The nfs_server_ip_association resource associates the floating IP created earlier with the NFS server instance.

Finally, the nfs_clients resource defines a resource group containing instance configured to be NFS clients and mount the directory exported by the NFS server defined earlier. The IP of the NFS server is gathered using the get_attr function, and placed into user-data using the str_replace function.

Parameters all have the same data structure: each one has a name (key_name or reservation_id in this case), a data type (number or string), a comment field called description, optionally a default value, and a list of constraints (in this case only one per parameter). Constraints tell Heat to match a parameter to a specific type of OpenStack resource. Complex appliances on Chameleon require users to customize at least the key pair name and reservation ID, and will generally provide additional parameters to customize other properties of the cluster, such as its size, as in this example.

Outputs are declared similarly to parameters: they each have a name, an optional description, and a value. They allow to return information from the stack to the user.

```
# This describes what is deployed by this template.
description: NFS server and clients deployed with Heat on Chameleon

# This defines the minimum Heat version required by this template.
heat_template_version: 2015-10-15

# The resources section defines what OpenStack resources are to be deployed and
# how they should be configured.
resources:
    nfs_server_floating_ip:
        type: OS::Nova::FloatingIP
        properties:
            pool: ext-net

    nfs_server:
        type: OS::Nova::Server
        properties:
            flavor: baremetal
            image: CC-CentOS7
            key_name: { get_param: key_name }
            networks:
                - network: sharednet1
        scheduler_hints: { reservation: { get_param: reservation_id } }
        user_data: |
            #!/bin/bash
            yum install -y nfs-utils
```

```

mkdir -p /exports/example
chown -R cc:cc /exports
echo '/exports/example 10.140.80.0/22(rw,async) 10.40.0.0/23(rw,async)' >> /etc/exports
systemctl enable rpcbind && systemctl start rpcbind
systemctl enable nfs-server && systemctl start nfs-server

nfs_server_ip_association:
type: OS::Nova::FloatingIPAssociation
properties:
floating_ip: { get_resource: nfs_server_floating_ip }
server_id: { get_resource: nfs_server }

nfs_clients:
type: OS::Heat::ResourceGroup
properties:
count: { get_param: nfs_client_count }
resource_def:
type: OS::Nova::Server
properties:
flavor: baremetal
image: CC-CentOS7
key_name: { get_param: key_name }
networks:
- network: sharednet1
scheduler_hints: { reservation: { get_param: reservation_id } }
user_data:
str_replace:
template: |
#!/bin/bash
yum install -y nfs-utils
echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
mount -a
params:
$nfs_server_ip: { get_attr: [nfs_server, first_address] }

# The parameters section gathers configuration from the user.
parameters:
nfs_client_count:
type: number
description: Number of NFS client instances
default: 1
constraints:
- range: { min: 1 }
description: There must be at least one client.
key_name:
type: string
description: Name of a KeyPair to enable SSH access to the instance
default: default
constraints:
- custom_constraint: nova.keypair
reservation_id:
type: string
description: ID of the Blazar reservation to use for launching instances.
constraints:
- custom_constraint: blazar.reservation

outputs:
server_ip:
description: Public IP address of the NFS server
value: { get_attr: [nfs_server_floating_ip, ip] }
client_ips:

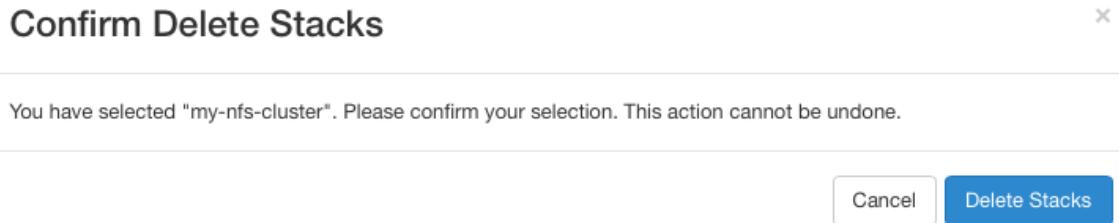
```

```
description: Private IP addresses of the NFS clients
value: { get_attr: [nfs_clients, first_address] }
```

18.8.5 Customizing an existing template

Customizing an existing template is a good way to start developing your own. We will use a simpler template than the previous example to start with: it is the [Hello World complex appliance](#).

First, delete the stack you launched, because we will need all three nodes to be free. To do this, go back to the Project > Orchestration > Stacks page, select your stack, and then click on the red “Delete Stacks” button. You will be asked to confirm, so click on the blue “Delete Stacks” button.



The template for the [Hello World complex appliance](#) is reproduced below. It is similar to the NFS share appliance, except that it deploys only a single client. You can see that it has four resources defined:

- nfs_server_floating_ip
- nfs_server
- nfs_server_ip_association
- nfs_client

The nfs_client instance mounts the NFS directory shared by the nfs_server instance, just like in our earlier example.

```
# This describes what is deployed by this template.
description: NFS server and client deployed with Heat on Chameleon

# This defines the minimum Heat version required by this template.
heat_template_version: 2015-10-15

# The resources section defines what OpenStack resources are to be deployed and
# how they should be configured.
resources:
  nfs_server_floating_ip:
    type: OS::Nova::FloatingIP
    properties:
      pool: ext-net

  nfs_server:
    type: OS::Nova::Server
    properties:
      flavor: baremetal
      image: CC-CentOS7
      key_name: { get_param: key_name }
    networks:
      - network: sharednet1
  scheduler_hints: { reservation: { get_param: reservation_id } }
  user_data: |
    #!/bin/bash
```

```

        yum install -y nfs-utils
        mkdir -p /exports/example
        chown -R cc:cc /exports
        echo '/exports/example 10.140.80.0/22(rw,async) 10.40.0.0/23(rw,async)' >> /etc/exports
        systemctl enable rpcbind && systemctl start rpcbind
        systemctl enable nfs-server && systemctl start nfs-server

nfs_server_ip_association:
    type: OS::Nova::FloatingIPAssociation
    properties:
        floating_ip: { get_resource: nfs_server_floating_ip }
        server_id: { get_resource: nfs_server }

nfs_client:
    type: OS::Nova::Server
    properties:
        flavor: baremetal
        image: CC-CentOS7
        key_name: { get_param: key_name }
        networks:
            - network: sharednet1
    scheduler_hints: { reservation: { get_param: reservation_id } }
    user_data:
        str_replace:
            template: |
                #!/bin/bash
                yum install -y nfs-utils
                echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
                mount -a
        params:
            $nfs_server_ip: { get_attr: [nfs_server, first_address] }

# The parameters section gathers configuration from the user.
parameters:
    key_name:
        type: string
        description: Name of a KeyPair to enable SSH access to the instance
        default: default
        constraints:
            - custom_constraint: nova.keypair
    reservation_id:
        type: string
        description: ID of the Blazar reservation to use for launching instances.
        constraints:
            - custom_constraint: blazar.reservation

```

Download this template from the [Hello World complex appliance details page](#) to your local machine, and open it in your favorite text editor.

We will customize the template to add a second NFS client by creating a new resource called another_nfs_client. Add the following text to your template inside the resources section. Make sure to respect the level of indentation, which is important in YAML.

```

another_nfs_client:
    type: OS::Nova::Server
    properties:
        flavor: baremetal
        image: CC-CentOS7
        key_name: { get_param: key_name }
        networks:
            - network: sharednet1

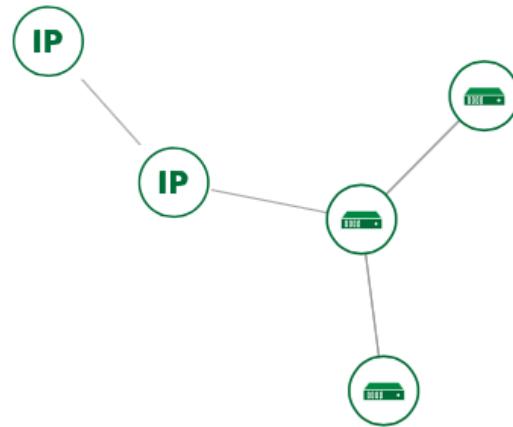
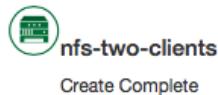
```

```

scheduler_hints: { reservation: { get_param: reservation_id } }
user_data:
  str_replace:
    template: |
      #!/bin/bash
      yum install -y nfs-utils
      echo "$nfs_server_ip:/exports/example      /mnt/      nfs" > /etc/fstab
      mount -a
params:
  $nfs_server_ip: { get_attr: [nfs_server, first_address] }

```

Now, launch a new stack with this template. Since the customized template is only on your computer and cannot be addressed by a URL, use the “Direct Input” method instead and copy/paste the content of the customized template. The resulting topology view is shown below: as you can see, the two client instances are shown separately since each one is defined as a separate resource in the template.



You may have realized already that while adding just one additional client instance was easy, launching more of them would require to copy / paste blocks of YAML many times while ensuring that the total count is correct. This would be easy to get wrong, especially when dealing with tens or hundreds of instances.

So instead, we leverage another construct from Heat: resource groups. Resource groups allow to define one kind of resource and request it to be created any number of times.

Remove the `nfs_client` and `another_client` resources from your customized template, and replace them with the following:

```

nfs_clients:
  type: OS::Heat::ResourceGroup
  properties:
    count: 2
    resource_def:
      type: OS::Nova::Server
      properties:

```

```

flavor: baremetal
image: CC-CentOS7
key_name: { get_param: key_name }
networks:
    - network: sharednet1
scheduler_hints: { reservation: { get_param: reservation_id } }
user_data:
    str_replace:
        template: |
            #!/bin/bash
            yum install -y nfs-utils
            echo "$nfs_server_ip:/exports/example    /mnt/    nfs" > /etc/fstab
            mount -a
    params:
        $nfs_server_ip: { get_attr: [nfs_server, first_address] }

```

A resource group is configured with a properties field, containing the definition of the resource to launch (`resource_def`) and the number of resources to launch (`count`). Once launched, you will notice that the topology view groups all client instances under a single Resource Group icon. We use the same `resource_def` than when defining separate instances earlier.

Another way we can customize this template is by adding outputs to the template. Outputs allow a Heat template to return data to the user. This can be useful to return values like IP addresses or credentials that the user must know to use the system.

We will create an output returning the floating IP address used by the NFS server. We define an outputs section, and one output with the name `server_ip` and a description. The value of the output is gathered using the `get_attr` function which obtains the IP address of the server instance.

```

outputs:
    server_ip:
        description: Public IP address of the NFS server
        value: { get_attr: [nfs_server_floating_ip, ip] }

```

You can get outputs in the “Overview” tab of the Stack Details page. If you want to use the command line, install `python-heatclient` and use the `heat output-list` and `heat output-show` commands, or get a full list in the information returned by `heat stack-show`.

Multiple outputs can be defined in the outputs section. Each of them needs to have a unique name. For example, we can add another output to list the private IPs assigned to client instances:

```

client_ips:
    description: Private IP addresses of the NFS clients
    value: { get_attr: [nfs_clients, first_address] }

```

The image below shows the resulting outputs as viewed from the web interface. Of course IP addresses will be specific to each deployment.

Outputs

<code>client_ips</code>	Private IP address of the NFS clients
	["10.140.82.20", "10.140.82.19"]
<code>server_ip</code>	Public IP address of the NFS server
	130.202.88.157

Finally, we can add a new parameter to replace the hardcoded number of client instances by a value passed to the template. Add the following text to the parameters section:

```
nfs_client_count:
  type: number
  description: Number of NFS client instances
  default: 1
  constraints:
    - range: { min: 1 }
  description: There must be at least one client.
```

Inside the resource group definition, change count: 2 to count: { get_param: nfs_client_count } to retrieve and use the parameter we just defined. When you launch this template, you will see that an additional parameter allows you to define the number of client instances, like in the NFS share appliance.

At this stage, we have fully recreated the NFS share appliance starting from the Hello World one! The next section will explain how to write a new template from scratch.

18.8.6 Writing a new template

You may want to write a whole new template, rather than customizing an existing one. Each template should follow the same layout and be composed of the following sections:

- Heat template version
- Description
- Resources
- Parameters
- Outputs

Heat template version

Each Heat template has to include the heat_template_version key with a valid version of HOT (Heat Orchestration Template). Chameleon bare-metal supports any HOT version up to 2015-10-15, which corresponds to OpenStack Liberty. The [Heat documentation](#) lists all available versions and their features. We recommended that you always use the latest supported version to have access to all supported features:

```
heat_template_version: 2015-10-15
```

Description

While not mandatory, it is good practice to describe what is deployed and configured by your template. It can be on a single line:

```
description: This describes what this Heat template deploys on Chameleon.
```

If a longer description is needed, you can provide multi-line text in YAML, for example:

```
description: >
  This describes what this Heat
  template deploys on Chameleon.
```

Resources

The resources section is required and must contain at least one resource definition. A [complete list of resources types known to Heat](#) is available.

However, only a subset of them are supported by Chameleon, and some are limited to administrative use. We recommend that you only use:

- OS::Glance::Image
- OS::Heat::ResourceGroup
- OS::Heat::SoftwareConfig
- OS::Heat::SoftwareDeployment
- OS::Heat::SoftwareDeploymentGroup
- OS::Neutron::FloatingIP
- OS::Neutron::FloatingIPAssociation
- OS::Neutron::Port (advanced users only)
- OS::Nova::Keypair
- OS::Nova::Server

If you know of another resource that you would like to use and think it should be supported by the OpenStack services on Chameleon bare-metal, please let us know via our help desk.

Parameters

Parameters allow users to customize the template with necessary or optional values. For example, they can customize which Chameleon appliance they want to deploy, or which key pair to install. Default values can be provided with the `default` key, as well as constraints to ensure that only valid OpenStack resources can be selected. For example, `custom_constraint: glance.image` restricts the image selection to an available OpenStack image, while providing a pre-filled selection box in the web interface. [More details about constraints](#) are available in the Heat documentation.

Outputs

Outputs allow template to give information from the deployment to users. This can include usernames, passwords, IP addresses, hostnames, paths, etc. The outputs declaration is using the following format:

```
outputs:
  first_output_name:
    description: Description of the first output
    value: first_output_value
  second_output_name:
    description: Description of the second output
    value: second_output_value
```

Generally values will be calls to `get_attr`, `get_param`, or some other function to get information from parameters or resources deployed by the template and return them in the proper format to the user.

18.8.7 Sharing new complex appliances

If you have written your own complex appliances or substantially customized an existing one, we would love if you shared them with our user community!

The process is very similar to regular appliances: log into the Chameleon portal, go to the [appliance catalog](#), and click on the button in the top-right corner: “Add an appliance” (you need to be logged in to see it).

 Add an appliance

You will be prompted to enter a name, description, and documentation. Instead of providing appliance IDs, copy your template to the dedicated field. Finally, share your contact information and assign a version string to your appliance. Once submitted, your appliance will be reviewed. We will get in touch if a change is needed, but if it's all good we will publish it right away!

18.8.8 Advanced topics

All-to-all information exchange

The previous examples have all used user-data scripts to provide instances with contextualization information. While it is easy to use, this contextualization method has a major drawback: because it is given to the instance as part of its launch request, it cannot use any context information that is not yet known at this time.

In practice, this means that in a client-server deployment, only one of these pattern will be possible:

- The server has to be deployed first, and once it is deployed, the clients can be launched and contextualized with information from the server. The server won't know about the clients unless there is a mechanism (not managed by Heat) for the client to contact the server.
- The clients have to be deployed first, and once they are deployed, the server can be launched and contextualized with information from the clients. The clients won't know about the server unless there is a mechanism (not managed by Heat) for the server to contact the clients.

This limitation was already apparent in our NFS share appliance: this is why the server instance exports the file system to all bare-metal instances on Chameleon, because it doesn't know which specific IP addresses are allocated to the clients.

This limitation is even more important if the deployment is not hierarchical, i.e. all instances need to know about all others. For example, a cluster with IP and hostnames populated in /etc/hosts required each instance to be known by every other instance.

This section presents a more advanced form of contextualization that can perform this kind of information exchange. This is implemented by Heat agents running inside instances and communicating with the Heat service to send and receive information. This means you will need to use an image bundling these agents. Currently, our CC-CentOS7 appliance and its CUDA version are the only ones supporting this mode of contextualization. If you build your own images using the [CC-CentOS7 appliance builder](#), you will automatically have these agents installed.

This contextualization is performed with several Heat resources:

- `OS::Heat::SoftwareConfig`. This resource describes code to run on an instance. It can be configured with inputs and provide outputs.
- `OS::Heat::SoftwareDeployment`. This resource applies a SoftwareConfig to a specific instance.
- `OS::Heat::SoftwareDeploymentGroup`. This resource applies a SoftwareConfig to a specific group of instances.

The template below illustrates how it works. It launches a group of instances that will automatically populates their /etc/hosts file with IP and hostnames from other instances in the deployment.

```
heat_template_version: 2015-10-15
```

```

description: >
    This template demonstrates how to exchange hostnames and IP addresses to populate /etc/hosts.

parameters:
  flavor:
    type: string
    default: baremetal
    constraints:
      - custom_constraint: nova.flavor
  image:
    type: string
    default: CC-CentOS7
    constraints:
      - custom_constraint: glance.image
  key_name:
    type: string
    default: default
    constraints:
      - custom_constraint: nova.keypair
  instance_count:
    type: number
    default: 2
  reservation_id:
    type: string
    description: ID of the Blazar reservation to use for launching instances.
    constraints:
      - custom_constraint: blazar.reservation

resources:
  export_hosts:
    type: OS::Heat::SoftwareConfig
    properties:
      outputs:
        - name: hosts
      group: script
      config: |
        #!/bin/sh
        (echo -n $(facter ipaddress); echo -n ' '; echo $(facter hostname)) > ${heat_outputs_path}.hosts

  export_hosts_sdg:
    type: OS::Heat::SoftwareDeploymentGroup
    properties:
      config: { get_resource: export_hosts }
      servers: { get_attr: [server_group, refs_map] }
      signal_transport: HEAT_SIGNAL

  populate_hosts:
    type: OS::Heat::SoftwareConfig
    properties:
      inputs:
        - name: hosts
      group: script
      config: |
        #!/usr/bin/env python
        import ast
        import os
        import string
        import subprocess
        hosts = os.getenv('hosts')
        if hosts is not None:
            hosts = ast.literal_eval(string.replace(hosts, '\n', '\\\n'))

```

```

        with open('/etc/hosts', 'a') as hosts_file:
            for ip_host in hosts.values():
                hosts_file.write(ip_host.rstrip() + '\n')

    populate_hosts_sdg:
        type: OS::Heat::SoftwareDeploymentGroup
        depends_on: export_hosts_sdg
        properties:
            config: { get_resource: populate_hosts }
            servers: { get_attr: [server_group, refs_map] }
            signal_transport: HEAT_SIGNAL
            input_values:
                hosts: { get_attr: [export_hosts_sdg, hosts] }

    server_group:
        type: OS::Heat::ResourceGroup
        properties:
            count: { get_param: instance_count }
            resource_def:
                type: OS::Nova::Server
                properties:
                    flavor: { get_param: flavor }
                    image: { get_param: image }
                    key_name: { get_param: key_name }
                    networks:
                        - network: sharednet1
            scheduler_hints: { reservation: { get_param: reservation_id } }
            user_data_format: SOFTWARE_CONFIG
            software_config_transport: POLL_SERVER_HEAT

    outputs:
        deployment_results:
            value: { get_attr: [export_hosts_sdg, hosts] }

```

There are two SoftwareConfig resources.

The first SoftwareConfig, `export_hosts`, uses the `facter` tool to extract IP address and hostname into a single line (in the format expected for `/etc/hosts`) and writes it to a special path (`${heat_outputs_path}.hosts`). This prompts Heat to assign the content of this file to the output with the name `hosts`.

The second SoftwareConfig, `populate_hosts`, takes as input a variable named `hosts`, and applies a script that reads the variable from the environment, parses it with `ast.literal_eval` (as it is formatted as a Python dict), and writes each value of the dictionary to `/etc/hosts`.

The SoftwareDeploymentGroup resources `export_hosts_sdg` and `populate_hosts_sdg` apply each SoftwareConfig to the instance ResourceGroup with the correct configuration.

Finally, the instance ResourceGroup is configured so that each instance uses the following contextualization method instead of a user-data script:

```

user_data_format: SOFTWARE_CONFIG
software_config_transport: POLL_SERVER_HEAT

```

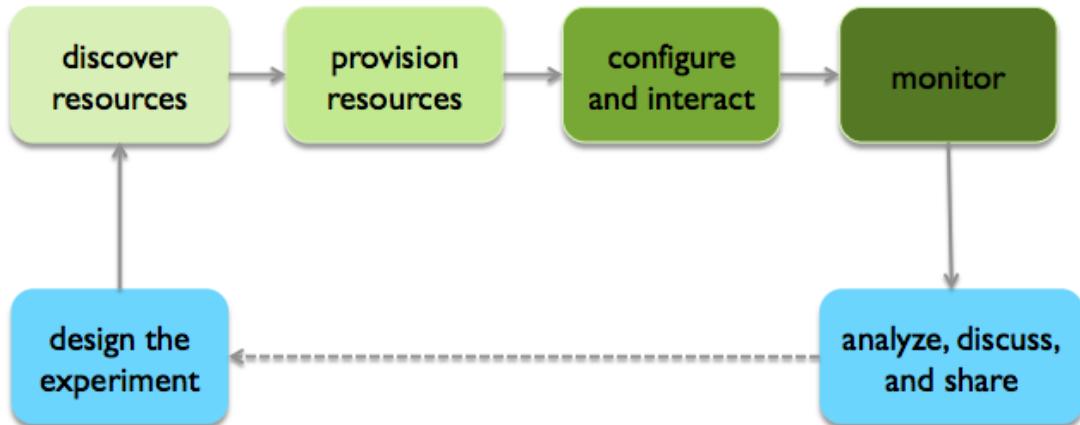
You can follow the same template pattern to configure your own deployment requiring all-to-all information exchange.



[section/cloud/chameleon/baremetal.tex](#)

18.9 Bare Metal

In this page you will find documentation guiding you through the bare-metal deployment features available in Chameleon. Chameleon gives users administrative access to bare-metal compute resources to run cloud computing experiments with a high degree of customization and repeatability. Typically, an experiment will go through several phases, as illustrated in the figure below:



The bare-metal user guide comes in two editions. The first is how to use Chameleon resources via the web interface, the recommended choice for new users to quickly learn how to use our testbed:

[Get started with Chameleon using the web interface](#)

1. [Discover Resources](#)
2. [Provision Resources](#)
3. [Configure and Interact](#)
4. [Monitor and Collect Results](#)

The second targets advanced users who are already familiar with Chameleon and would like to learn how to use Chameleon from the command line or with scripts.

[Get started with Chameleon using the command line \(advanced\)](#)

1. [Discover Resources](#)
2. [Provision Resources](#)
3. [Configure and Interact](#)
4. [Monitor and Collect Results](#)

You do not need to strictly follow the documentation sequentially. However, note that some steps assume that previous ones have been successfully performed.

You can also consult documentation describing how to use advanced features of Chameleon not covered by the guides above:

- the [Chameleon Object Store](#),
- network isolation for bare metal.

18.10 Frequently Asked Questions



[section/cloud/chameleon/faq.tex](#)

section/cloud/chameleon/faq.tex

18.10.1 Appliances

What is an appliance?

An appliance is an application packaged together with the environment that this application requires. For example, an appliance can consist of the operating system, libraries and tools used by the application, configuration features such as environment variable settings, and the installation of the application itself. Examples of appliances might include a KVM virtual machine image, a Docker image, or a bare metal image. Chameleon appliance refers to bare metal images that can be deployed on the Chameleon testbed. Since an appliance captures the experimental environment exactly, it is a key element of reproducibility; publishing an appliance used to obtain experimental results will go a long way to allowing others to reproduce and build on your research easily.

To deploy distributed applications on several Chameleon instances, complex appliances combine an image and a template describing how the cluster should be configured and contextualized. You can read more about them in the [Complex Appliances documentation](#).

What is the Appliance Catalog?

The Chameleon Appliance Catalog is a repository that allows users to discover, publish, and share appliances. The appliance catalog contains useful images of both bare metal and virtual machine appliances supported by the Chameleon team as well as appliances contributed by users.

How do I publish an appliance in the Appliance Catalog?

The new Appliance Catalog allows you to easily publish and share your own appliances so that others can discover them and use them either to reproduce the research of others or as a basis for their own research. Before creating your own appliance it is advisable to review other appliances on the [Chameleon Appliance Catalog](#) in order to get an idea of the categories you will want to contribute and what others have done.

Once you are ready to proceed, an appliance can be contributed to Chameleon in the following steps:

1. Create the appliance itself. You may want to test it as well as give some thought to what support you are willing to provide for the appliance (e.g., if your group developed and supports a software package, the appliance may be just a new way of packaging the software and making it available, in which case your standard support channels may be appropriate for the appliance as well).
2. Upload the appliance to the Chameleon Image Repository (Glance) and make the image public. In order to enter the appliance into the Catalog you will be asked to provide the Glance ID for the image. These IDs are per-cloud, so that there are three options right now: bare metal/CHI at University of Chicago, bare metal/CHI at TACC, and OpenStack/KVM at TACC. You will need to provide at least one appliance, but may want to provide all three.
3. Go to the [Appliance Catalog Create Appliance web form](#), fill out, and submit the form. Be prepared to provide the following information: a descriptive name (this sometimes requires some thought!), author and support contact, version, and an informative description. The description is a very important part of the appliance record; others will use it to evaluate if the appliance contains tools they need for their research so it makes sense to prepare it carefully. To make your description effective you may want to think of the following questions: what does the appliance contain? what are the specific packages and their versions? what is it useful for? where can it be deployed and/or what restrictions/limitations does it have? how should users connect to it / what accounts are enabled?

If you are adding a complex appliance, skip the image ID fields and enter your template instead in the dedicated text box.

As always, if you encounter any problems or want to share with us additional improvements we should do to the process, please don't hesitate to [submit a ticket](#).

How can I manage an appliance on Appliance Catalog?

If you are the owner of the appliance, you can edit the appliance data, such as the description or the support information. Browse to the appliance that you want to edit and view its Details page. At the top right of the page is an Edit button. You will be presented with the same web form as when creating the appliance, pre-filled with the appliances current information. Make changes as necessary and click Save at the bottom of the page.

And finally, you can delete appliances you had made available. Browse to the appliance that you want to delete and click Edit on the Appliance Details page. At the bottom of the page is a Delete button. You will be asked to confirm once more that you do want to delete this appliance. After confirming, the appliance will be removed and no longer listed on the Appliance Catalog.

Why are there different image IDs for the same appliance?

The three clouds forming the Chameleon testbed are fully separated, each having its own Glance image repository. The same appliance image uploaded to the three clouds will produce three different image IDs.

In addition, it is sometimes needed to customize an appliance image for each site, resulting in slightly different image files.

Can I use another operating system on bare-metal?

The recommended appliance for Chameleon is CentOS 7 (supported by Chameleon staff), or appliances built on top of it.

These appliances provide Chameleon-specific customizations, such as login using the cc account, the cc-checks utility to verify hardware against our resource registry, gathering of metrics, etc.

Since 2016, we also provide and support Ubuntu 14.04 and 16.04 appliances with the same functionality.

18.10.2 Bare Metal Troubleshooting

Why are my Bare Metal instances failing to launch?

The Chameleon Bare Metal clouds require users to reserve resources before allowing them to launch instances. Please follow the [documentation](#) and make sure that:

1. You have created a lease and it has started (the associated reservation is shown as **Active**)
2. You have selected your reservation in the **Launch Instance** panel

If you still cannot start instances, please [open a ticket with our help desk](#).

18.10.3 OpenStack KVM Troubleshooting

Why are my OpenStack KVM instances failing to launch?

If you get an error stating that **No valid host was found**, it might be caused by a lack of resources in the cloud. The Chameleon staff continuously monitors the utilization of the testbed, but there might be times when no more resources are available. If the error persists, please [open a ticket with our help desk](#).

Why can't I ping or SSH to my instance?

While the possibility that the system is being taking over by nanites should not be discounted too easily, it is always prudent to first check for the following issues:

- Do you have a floating IP associated with your instance? By default, instances do not have publicly-accessible IP addresses assigned. See the **Managing Virtual Machine Instances** section in the [User Guide](#).
- Does your security group allow incoming ICMP (e.g. ping) traffic? By default, firewall rules do not allow ping to your instances. If you wish to enable it, see the **Firewall (Access Security)** section in the [User Guide](#).
- Does your security group allow incoming SSH (TCP port 22) traffic? By default, firewall rules do not allow SSH to your instances. If you wish to enable it, see the **Firewall (Access Security)** section in the [User Guide](#).

If none of these solve your problem, please [open a ticket with our help desk](#), and send us the results of the above (and any evidence of nanites you find as well).

Python

19	Introduction	227
19.1	Introduction to Python	
19.2	References	
20	Install	231
20.1	Python Installation	
20.2	Interactive Python	
20.3	REPL (Read Eval Print Loop)	
20.4	Python 3 Features in Python 2	
21	Language	239
21.1	Statements and Strings	
21.2	Variables	
21.3	Data Types	
21.4	Module Management	
21.5	Date Time in Python	
21.6	Control Statements	
21.7	Datatypes	
21.8	Functions	
21.9	Classes	
21.10	Modules	
21.11	Lambda Expressions	
21.12	Generators	
21.13	Non Blocking Threads	
22	Data Management	253
22.1	Formats	
22.2	Encryption	
22.3	Database Access	
23	Libraries	257
23.1	Installing Libraries	
23.2	Using pip to Install Packages	
23.3	GUI	
23.4	Formatting and Checking Python Code	
23.5	Using autopep8	
23.6	Writing Python 3 Compatible Code	
23.7	Using Python on FutureSystems	
23.8	Ecosystem	
23.9	Resources	
23.10	Exercises	
23.11	Python for Big Data	
23.12	Parsing Data	
24	Cloudmesh Command Shell	271
24.1	CMD5	



19. Introduction

F part/python.tex

F section/prg/python-intro.tex

19.1 Introduction to Python

Portions of this lesson have been adapted from the [official Python Tutorial](#) copyright [Python Software Foundation](#).

Python is an easy to learn programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's simple syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation. The Python interpreter can be extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

Python is an interpreted, dynamic, high-level programming language suitable for a wide range of applications.

The philosophy of python is summarized in [The Zen of Python](#) as follows:

- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated

- Readability counts

The main features of Python are:

- Use of indentation whitespace to indicate blocks
- Object orient paradigm
- Dynamic typing
- Interpreted runtime
- Garbage collected memory management
- a large standard library
- a large repository of third-party libraries

Python is used by many companies (such as Google, Yahoo!, CERN, NASA) and is applied for web development, scientific computing, embedded applications, artificial intelligence, software development, and information security, to name a few.

The material collected here introduces the reader to the basic concepts and features of the Python language and system. After you have worked through the material you will be able to:

- use Python
- use the interactive Python interface
- understand the basic syntax of Python
- write and run Python programs stored in a file
- have an overview of the standard library
- install Python libraries using pyenv or if it is not available virtualenv

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules.

In order to conduct this lesson you need

- A computer with Python 2.7.13 or 3.6.2
- Familiarity with command line usage
- A text editor such as [PyCharm](#), emacs, vi or others. You should identify which works best for you and set it up.

19.2 References

Some important additional information can be found on the following Web pages.

- [Python](#)
- [Pip](#)
- [Virtualenv](#)
- [NumPy](#)
- [SciPy](#)
- [Matplotlib](#)
- [Pandas](#)
- [pyenv](#)
- [PyCharm](#)

Python module of the week is a Web site that provides a number of short examples on how to use

some elementary python modules. Not all modules are equally useful and you should decide if there are better alternatives. However for beginners this site provides a number of good examples

- Python 2: <https://pymotw.com/2/>
- Python 3: <https://pymotw.com/3/>



20. Install

F section/prg/python-install.tex

20.1 Python Installation

Python is easy to install and very good instructions for most platforms can be found on the python.org Web page. We will be using Python 2.7.13 and/or Python 3 in our activities.

To manage python modules, it is useful to have [pip](#) package installation tool on your system.

In the tutorial, we assume that you have a computer with python installed. However, we also recommend that for the class you use Python's virtualenv (see below) to isolate your development Python from the system installed Python.

20.1.1 Managing custom Python installs

Often you have your own computer and you do not like to change its environment to keep it in pristine condition. Python comes with many libraries that could for example conflict with libraries that you have installed. To avoid this it is best to work in an isolated python we can use tools such as virtualenv, pyenv or pyvenv for 3.6.2. Which you use depends on you, but we highly recommend pyenv if you can.

Managing Multiple Python Versions with Pyenv

Python has several versions that are used by the community. This includes Python 2 and Python 3, but all different management of the python libraries. As each OS may have their own version of python installed. It is not recommended that you modify that version. Instead you may want to create a localized python installation that you as a user can modify. To do that we recommend [pyenv](#). Pyenv allows users to switch between multiple versions of Python (<https://github.com/yyuu/pyenv>).

To summarize:

- users to change the global Python version on a per-user basis;
- users to enable support for per-project Python versions;
- easy version changes without complex environment variable management;
- to search installed commands across different python versions;
- integrate with tox (<https://tox.readthedocs.io/>).

Instalation without pyenv

If you need to have more than one python version installed and do not want or can use pyenv, we recommend you download and install python 2.7.13 and 3.6.2 from python.org (<https://www.python.org/downloads/>)

Disabeling wrong python installs on OSX

While working with students we have seen at times that they take other classes either at universities or online that teach them how to program in python. Unfortuanatley, although they seem to do that they often ignore to teach you how to properly install python. I just reacenthal had a students that had installed python 7 times on his OSX machine, while another student had 3 different instalations, all of which confliced with each other as they were not set up properly.

We recommend that you inspect if you have a files such as `~/.bashrc` or `~/.bashrc_profile` in your ehome directory and identify if it activates various versions of python on your computer. If so you could try to deactivate them while outcommenting the various versions with the `#` character at the beginning of the line, start a new terminal and see if the terminal shell still works. Than you can follow our instructions here while using an install on pyenv.

Install pyenv on OSX from git

This is our recommended way to install pyenv on OSX:

```
$ git clone https://github.com/pyenv/pyenv.git ~/.pyenv
$ git clone https://github.com/pyenv/pyenv-virtualenv.git ~/.pyenv/plugins/pyenv-virtualenv
$ git clone https://github.com/yyuu/pyenv-virtualenvwrapper.git ~/.pyenv/plugins/pyenv-virtualenvwrap
$ echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bash_profile
$ echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bash_profile
```

Instalation of Homebrew

Before installing anything on your computer make sure you have enough space. Use in the terminal the command:

```
$ df -h
```

which gives your an overview of your file system. If you do not have enough space, please make sure you free up unused files from your drive.

In many occasions it is beneficial to use readline as it provides nice editing features for the terminal and xz for completion. First, make sure you have xcode installed:

```
$ xcode-select --install
```

Next install homebrew, pyenv, pyenv-virtualenv and pyenv-virtualwrapper. Additionally install readline and some compression tools:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
brew update
```

```
brew install readline xz
```

Install pyenv on OSX with Homebrew

We describe here a mechanism of installing pyenv with homebrew. Other mechanisms can be found on the pyenv documentation page (<https://github.com/yyuu/pyenv-installer>). You must have homebrew installed as discussed in the previous section.

To install pyenv with homebrew execute in the terminal:

```
brew install pyenv pyenv-virtualenv pyenv-virtualenvwrapper
```

Install pyenv on Ubuntu

The following steps will install pyenv in a new ubuntu 16.04 distribution.

Start up a terminal and execute in the terminal the following commands. We recommend that you do it one command at a time so you can observe if the command succeeds:

```
$ sudo apt-get update
$ sudo apt-get install git python-pip make build-essential libssl-dev
$ sudo apt-get install zlib1g-dev libbz2-dev libreadline-dev libsdlite3-dev
$ sudo pip install virtualenvwrapper

$ git clone https://github.com/yyuu/pyenv.git ~/.pyenv
$ git clone https://github.com/pyenv/pyenv-virtualenv.git ~/.pyenv/plugins/pyenv-virtualenv
$ git clone https://github.com/yyuu/pyenv-virtualenvwrapper.git ~/.pyenv/plugins/pyenv-virtualenvwrap
```

Now that you have installed pyenv it is not yet activated in your current terminal. The easiest thing to do is to start a new terminal and type in:

```
which pyenv
```

If you see a response pyenv is installed and you can proceed with the next steps.

Please remember whenever you modify .bashrc or .bash_profile you need to start a new terminal.

Install Different Python Versions

Pyenv provides a large list of different python versions. To see the entire list please use the command:

```
$ pyenv install -l
```

However, for us we only need to worry about python 2.7.13 and python 3.6.2 (once 3.6.2 becomes available we will use that). You can now install different versions of python into your local environment with the following commands:

```
$ pyenv install 2.7.13
$ pyenv install 3.6.2
```

You can set the global python default version with:

```
$ pyenv global 2.7.13
```

Type the following to determine which version you activated:

```
$ pyenv version
```

Type the following to determine which versions you have available:

```
$ pyenv versions
```

Associate a specific environment name with a certain python version, use the following commands:

```
$ pyenv virtualenv 2.7.13 ENV2
$ pyenv virtualenv 3.6.2 ENV3
```

In the example above, ENV2 would represent python 2.7.13 while ENV3 would represent python 3.6.2. Often it is easier to type the alias rather than the explicit version.

Set up the Shell

To make all work smoothly from your terminal, you can include the following in your .bashrc files:

```
export PYENV_VIRTUALENV_DISABLE_PROMPT=1
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"

__pyenv_version_ps1() {
    local ret=$?
    output=$(pyenv version-name)
    if [[ ! -z $output ]]; then
        echo -n "($output)"
    fi
    return $ret;
}

PS1="\$(__pyenv_version_ps1) ${PS1}"
```

We recommend that you do this towards the end of your file.

Switching Environments

After setting up the different environments, switching between them is now easy. Simply use the following commands:

```
(2.7.13) $ pyenv activate ENV2
(ENV2) $ pyenv activate ENV3
(ENV3) $ pyenv activate ENV2
(ENV2) $ pyenv deactivate ENV2
(2.7.13) $
```

To make it even easier, you can add the following lines to your .bash_profile file:

```
alias ENV2="pyenv activate ENV2"
alias ENV3="pyenv activate ENV3"
```

If you start a new terminal, you can switch between the different versions of python simply by typing:

```
$ ENV2
$ ENV3
```

20.1.2 Instalation without pyenv

If you need to have more than one python version installed and do not want or can use pyenv, we recommend you download and install python 2.7.13 and 3.6.2 from python.org (<https://www.python.org/downloads/>)

Make sure pip is up to date

As you will want to install other packages, make sure pip is up to date:

```
pip install pip -U
pyenv virtualenv anaconda3-4.3.1 ANA3
pyenv activate ANA3
```

20.1.3 Anaconda and Miniconda

We do not recommend that you use anaconda or miniconda as it may interfere with your default python interpreters and setup.

Please note that beginners to python should always use anaconda or miniconda only after they have installed pyenv and use it. For this class neither anaconda nor miniconda is required. In fact we do not recommend it. We keep this section as we know that other classes at IU may use anaconda. We are not aware if these classes teach you the right way to install it, with *pyenv*.

Miniconda

This section about miniconda is experimental and has not been tested. We are looking for contributors that help completing it. If you use anaconda or miniconda we recommend to manage it via pyenv.

To install mini conda you can use the following commands:

```
$ mkdir ana
$ cd ana
$ pyenv install miniconda3-latest
$ pyenv local miniconda3-latest
$ pyenv activate miniconda3-latest
$ conda create -n ana anaconda
```

To activate use:

```
$ source activate ana
```

To deactivate use:

```
$ source deactivate
```

To install cloudmesh cmd5 please use:

```
$ pip install cloudmesh.cmd5
$ pip install cloudmesh.sys
```

Anaconda

This section about anaconda is experimental and has not been tested. We are looking for contributors that help completing it.

You can add anaconda to your pyenv with the following commands:

```
pyenv install anaconda3-4.3.1
```

To switch more easily we recommend that you use the following in your .bash_profile file:

```
alias ANA="pyenv activate anaconda3-4.3.1"
```

Once you have done this you can easily switch to anaconda with the command:

```
$ ANA
```

Terminology in anaconda could lead to confusion. Thus we like to point out that the version

number of anaconda is unrelated to the python version. Furthermore, anaconda uses the term root not for the root user, but for the originating directory in which the anaconda program is installed.

In case you like to build your own conda packages at a later time we recommend that you install the conda-build package:

```
$ conda install conda-build
```

When executing:

```
pyenv versions
```

you will see after the install completed the anaconda versions installed:

```
pyenv versions
system
2.7.13
2.7.13/envs/ENV2
3.6.2
3.6.2/envs/ENV3
ENV2
ENV3
* anaconda3-4.3.1 (set by PYENV_VERSION environment variable)
```

Let us now create virtualenv for anaconda:

```
$ pyenv virtualenv anaconda3-4.3.1 ANA
```

To activate it you can now use:

```
$ pyenv ANA
```

However, anaconda may modify your .bashrc or .bash_profile files and , may result in incompatibilities with other python versions. For this reason we recommend not to use it. If you find ways to get it to work reliably with other versions, please let us know and we update this tutorial.

To install cloudmesh cmd5 please use:

```
$ pip install cloudmesh.cmd5
$ pip install cloudmesh.sys
```

Exercise

Epyenv.1: Write installation instructions for an operating system of your choice and add to this documentation.

Epyenv.2: Replicate the steps above, so you can type in ENV2 and ENV3 in your terminals to switch between python 2 and 3.

virtualenv

environment while using virtualenv,. Documentation about it can be found at:

```
* https://virtualenv.pypa.io
```

The installation is simple once you have pip installed. If it is not installed you can say:

```
$ easy_install pip
```

After that you can install the virtual env with:

```
$ pip install virtualenv
```

To setup an isolated environment for example in the directory ~/ENV please use:

```
$ virtualenv ~/ENV
```

To activate it you can use the command:

```
$ source ~/ENV/bin/activate
```

you can put this command in your .bashrc or .bash_profile files so you do not forget to activate it. Instructions for this can be found in our lesson on Linux <bashrc>.

 section/prg/python-interactive.tex

20.2 Interactive Python

Python can be used interactively. You can enter the interactive mode by entering the interactive loop by executing the command:

```
1 $ python
```

You will see something like the following:

```
1 Python 2.7.13 (default, Nov 19 2016, 06:48:10)
2 [GCC 5.4.0 20160609] on linux2
3 Type "help", "copyright", "credits" or "license" for more information.
4 >>>
```

The >>> is the prompt used by the interpreter. This is similar to bash where commonly \$ is used.

Sometimes it is convenient to show the prompt when illustrating an example. This is to provide some context for what we are doing. If you are following along you will not need to type in the prompt.

This interactive python process does the following:

- *read* your input commands
- *evaluate* your command
- *print* the result of evaluation
- *loop* back to the beginning.

This is why you may see the interactive loop referred to as a **REPL: Read-Evaluate-Print-Loop**.

20.3 REPL (Read Eval Print Loop)

There are many different types beyond what we have seen so far, such as **dictionaries**, **lists**, **sets**. One handy way of using the interactive python is to get the type of a value using type():

```
1 >>> type(42)
2 <type 'int'>
3 >>> type('hello')
4 <type 'str'>
5 >>> type(3.14)
6 <type 'float'>
```

You can also ask for help about something using help():

```
1 >>> help(int)
2 >>> help(list)
3 >>> help(str)
```

Using help() opens up a help message within a pager. To navigate you can use the spacebar to go down a page w to go up a page, the arrow keys to go up/down line-by-line, or q to exit.

20.4 Python 3 Features in Python 2

In this course we want to be able to seamlesly switch between python 2 and python 3. Thus it is convenient from the start to use python 3 syntax when it is supported also in python 2. One of the most used functions is the print statement that has in python 3 parantheses. To enable it in python 2 you just need to import this function:

```
1 >>> from __future__ import print_function, division
```

The first of these imports allows us to use the print function to output text to the screen, instead of the print statement, which Python 2 uses. This is simply a [design decision](#) that better reflects Python's underlying philosophy.

Other functions such as the division also behave differently. Thus we use

```
1 >>> from __future__ import division
```

This import makes sure that the [division operator](#) behaves in a way a newcomer to the language might find more intuitive. In Python 2, division / is *floor division* when the arguments are integers, meaning that the following

```
1 (5 / 2 == 2) is True
```

In Python 3, division / is a flaoting point division, thus

```
1 (5 / 2 == 2.5) is True
```



21. Language

F section/prg/python.tex

TODO: TA: some of the python examples assume REPL, but its better to use a print statement instead as more general, please fix

21.1 Statements and Strings

Let us explore the syntax of Python. Type into the interactive loop and press Enter:

```
1 print("Hello world from Python!")
2 # Hello world from Python!
```

What happened: the print function was given a **string** to process. A string is a sequence of characters. A **character** can be a alphabetic (A through Z, lower and upper case), numeric (any of the digits), white space (spaces, tabs, newlines, etc), syntactic directives (comma, colon, quotation, exclamation, etc), and so forth. A string is just a sequence of the character and typically indicated by surrounding the characters in double quotes.

Standard output is discussed in the `../lesson/linux/shell` lesson.

So, what happened when you pressed Enter? The interactive Python program read the line `print "Hello world from Python!"`, split it into the `print` statement and the `"Hello world from Python!"` string, and then executed the line, showing you the output.

21.2 Variables

You can store data into a **variable** to access it later. For instance, instead of:

```
3 print('Hello world from Python!')
```

section/prg/python.tex

which is a lot to type if you need to do it multiple times, you can store the string in a variable for convenient access:

```
4 hello = 'Hello world from Python!'
5 print(hello)
6 # Hello world from Python!
```

21.3 Data Types

21.3.1 Booleans

A **boolean** is a value that indicates *truthiness* of something. You can think of it as a toggle: either “on” or “off”, “one” or “zero”, “true” or “false”. In fact, the only possible values of the **boolean** (or `bool`) type in Python are:

- True
- False

You can combine booleans with **boolean operators**:

- and
- or

```
7 print(True and True)
8 # True
9
10 print(True and False)
11 # False
12
13 print(False and False)
14 # False
15
16 print(True or True)
17 # True
18
19 print(True or False)
20 # True
21
22 print(False or False)
23 # False
```

21.3.2 Numbers

The interactive interpreter can also be used as a calculator. For instance, say we wanted to compute a multiple of 21:

```
24 print(21 * 2)
25 # 42
```

We saw here the `print` statement again. We passed in the result of the operation `21 * 2`. An **integer** (or **int**) in Python is a numeric value without a fractional component (those are called **floating point** numbers, or **float** for short).

The mathematical operators compute the related mathematical operation to the provided numbers. Some operators are:

* — multiplication

/ — division
 + — addition
 - — subtraction
 $\star\star$ — exponent

Exponentiation x^y is written as $x^{\star\star}y$ is x to the yth power.

You can combine **floats** and **ints**:

```
26 print(3.14 * 42 / 11 + 4 - 2)
27 # 13.9890909091
28
29 print(2**3)
30 # 8
```

Note that **operator precedence** is important. Using parenthesis to indicate affect the order of operations gives a difference results, as expected:

```
31 print(3.14 * (42 / 11) + 4 - 2)
32 # 11.42
33
34 print(1 + 2 * 3 - 4 / 5.0)
35 # 6.2
36
37 print((1 + 2) * (3 - 4) / 5.0 )
38 # -0.6
```

21.4 Module Management

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference. A module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.

21.4.1 Import Statement

When the interpreter encounters an import statement, it imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module. The from...import Statement Python's from statement lets you import specific attributes from a module into the current namespace. It is prefered to use for each import its own line such as:

```
39 import numpy
40 import matplotlib
```

When the interpreter encounters an import statement, it imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module.

21.4.2 The from ... import Statement

Python's from statement lets you import specific attributes from a module into the current namespace. The from ... import has the following syntax:

```
41 from datetime import datetime
```

21.5 Date Time in Python

The `datetime` module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. For related functionality, see also the `time` and `calendar` modules.

The import Statement You can use any Python source file as a module by executing an `import` statement in some other Python source file.

```
42 from datetime import datetime
```

This module offers a generic date/time string parser which is able to parse most known formats to represent a date and/or time.

```
43 from dateutil.parser import parse
```

`pandas` is an open source Python library for data analysis that needs to be imported.

```
44 import pandas as pd
```

Create a string variable with the class start time

```
45 fall_start = '08-21-2017'
```

Convert the string to datetime format

```
46 datetime.strptime(fall_start, '%m-%d-%Y')
47 # datetime.datetime(2017, 8, 21, 0, 0)
```

Creating a list of strings as dates

```
48 class_dates = ['8/25/2017', '9/1/2017', '9/8/2017', '9/15/2017', '9/22/2017',
                 '9/29/2017']
```

Convert `Class_dates` strings into datetime format and save the list into variable `a`

```
49 a = [datetime.strptime(x, '%m/%d/%Y') for x in class_dates]
```

Use `parse()` to attempt to auto-convert common string formats. Parser must be a string or character stream, not list.

```
50 parse(fall_start)
51 # datetime.datetime(2017, 8, 21, 0, 0)
```

Use `parse()` on every element of the `Class_dates` string.

```
52 [parse(x) for x in class_dates]
53 # [datetime.datetime(2017, 8, 25, 0, 0),
54 #  datetime.datetime(2017, 9, 1, 0, 0),
55 #  datetime.datetime(2017, 9, 8, 0, 0),
56 #  datetime.datetime(2017, 9, 15, 0, 0),
57 #  datetime.datetime(2017, 9, 22, 0, 0),
58 #  datetime.datetime(2017, 9, 29, 0, 0)]
```

Use `parse`, but designate that the day is first.

```
59 parse(fall_start, dayfirst=True)
60 # datetime.datetime(2017, 8, 21, 0, 0)
```

Create a `dataframe`.A `DataFrame` is a tabular data structure comprised of rows and columns, akin to a spreadsheet, database table. `DataFrame` as a group of `Series` objects that share an index (the column names).

```

61 import pandas as pd
62 data = { 'class_dates': [ '8/25/2017 18:47:05.069722' ,
63                           '9/1/2017 18:47:05.119994' ,
64                           '9/8/2017 18:47:05.178768' ,
65                           '9/15/2017 18:47:05.230071' ,
66                           '9/22/2017 18:47:05.230071' ,
67                           '9/29/2017 18:47:05.280592' ],
68   'complete': [1, 0, 1, 1, 0, 1]}
69 df = pd.DataFrame(data , columns = [ 'class_dates' , 'complete' ])
70 print(df)
71 #          class_dates    complete
72 # 0  8/25/2017 18:47:05.069722      1
73 # 1  9/1/2017 18:47:05.119994      0
74 # 2  9/8/2017 18:47:05.178768      1
75 # 3  9/15/2017 18:47:05.230071      1
76 # 4  9/22/2017 18:47:05.230071      0
77 # 5  9/29/2017 18:47:05.280592      1

```

Convert df[‘date’] from string to datetime

```

78 import pandas as pd
79 pd.to_datetime(df[ 'class_dates' ])
80 # 0 2017-08-25 18:47:05.069722
81 # 1 2017-09-01 18:47:05.119994
82 # 2 2017-09-08 18:47:05.178768
83 # 3 2017-09-15 18:47:05.230071
84 # 4 2017-09-22 18:47:05.230071
85 # 5 2017-09-29 18:47:05.280592
86 # Name: class_dates , dtype: datetime64[ns]

```

21.6 Control Statements

21.6.1 Comparision

Computer programs do not only execute instructions. Occasionally, a choice needs to be made. Such as a choice is based on a condition. Python has several conditional operators:

- > greater than
- < smaller than
- ==** equals
- !=** is not

Conditions are always combined with variables. A program can make a choice using the if keyword. For example:

```

87 x = int(input("Guess x:"))
88 if x == 4:
89     print('You guessed correctly!')

```

In this example, *You guessed correctly!* will only be printed if the variable x equals to four (see table above). Python can also execute multiple conditions using the elif and else keywords.

```

90 x = int(input("Guess x:"))
91 if x == 4:
92     print('You guessed correctly!')
93 elif abs(4 - x) == 1:
94     print('Wrong guess, but you are close!')
95 else:
96     print('Wrong guess')

```

21.6.2 Iteration

To repeat code, the `for` keyword can be used. For example, to display the numbers from 1 to 10, we could write something like this:

```
97 for i in range(1, 11):
98     print('Hello!')
```

The second argument to `range`, `11`, is not inclusive, meaning that the loop will only get to `10` before it finishes. Python itself starts counting from 0, so this code will also work:

```
99 for i in range(0, 10):
100     print(i + 1)
```

In fact, the `range` function defaults to starting value of `0`, so the above is equivalent to:

```
101 for i in range(10):
102     print(i + 1)
```

We can also nest loops inside each other:

```
103 for i in range(0,10):
104     for j in range(0,10):
105         print(i, ', ', j)
```

In this case we have two nested loops. The code will iterate over the entire coordinate range (0,0) to (9,9)

21.7 Datatypes

21.7.1 Lists

see: https://www.tutorialspoint.com/python/python_lists.htm

Lists in Python are ordered sequences of elements, where each element can be accessed using a 0-based index.

To define a list, you simply list its elements between square brackets ‘`[]`’:

```
106 names = [ 'Albert', 'Jane', 'Liz', 'John', 'Abby' ]
107 names[0] # access the first element of the list
108 # 'Albert'
109 names[2] # access the third element of the list
110 # 'Liz'
```

You can also use a negative index if you want to start counting elements from the end of the list. Thus, the last element has index `-1`, the second before last element has index `-2` and so on:

```
111 names[-1] # access the last element of the list
112 # 'Abby'
113 names[-2] # access the second last element of the list
114 # 'John'
```

Python also allows you to take whole slices of the list by specifying a beginning and end of the slice separated by a colon

```
115 names[1:-1] # the middle elements, excluding first and last
116 # ['Jane', 'Liz', 'John']
```

As you can see from the example above, the starting index in the slice is inclusive and the ending one, exclusive.

Python provides a variety of methods for manipulating the members of a list.

You can add elements with `append`:

```
117 names.append('Liz')
118 names
119 # ['Albert', 'Jane', 'Liz', 'John', 'Abby', 'Liz']
```

As you can see, the elements in a list need not be unique.

Merge two lists with `'extend'`:

```
120 names.extend(['Lindsay', 'Connor'])
121 names
122 # ['Albert', 'Jane', 'Liz', 'John', 'Abby', 'Liz', 'Lindsay', 'Connor']
```

Find the index of the first occurrence of an element with `'index'`:

```
123 names.index('Liz')
124 # 2
```

Remove elements by value with `'remove'`:

```
125 names.remove('Abby')
126 names
127 # ['Albert', 'Jane', 'Liz', 'John', 'Liz', 'Lindsay', 'Connor']
```

Remove elements by index with `'pop'`:

```
128 names.pop(1)
129 # 'Jane'
130 names
131 # ['Albert', 'Liz', 'John', 'Liz', 'Lindsay', 'Connor']
```

Notice that `pop` returns the element being removed, while `remove` does not.

If you are familiar with stacks from other programming languages, you can use `insert` and `'pop'`:

```
132 names.insert(0, 'Lincoln')
133 names
134 # ['Lincoln', 'Albert', 'Liz', 'John', 'Liz', 'Lindsay', 'Connor']
135 names.pop()
136 # 'Connor'
137 names
138 # ['Lincoln', 'Albert', 'Liz', 'John', 'Liz', 'Lindsay']
```

The Python documentation contains a full list of list operations.

To go back to the `range` function you used earlier, it simply creates a list of numbers:

```
139 range(10)
140 # [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
141 range(2, 10, 2)
142 # [2, 4, 6, 8]
```

21.7.2 Sets

Python lists can contain duplicates as you saw above:

```
143 names = ['Albert', 'Jane', 'Liz', 'John', 'Abby', 'Liz']
```

When we don't want this to be the case, we can use a `set`:

```
144 unique_names = set(names)
145 unique_names
146 # set(['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay'])
```

Keep in mind that the `set` is an unordered collection of objects, thus we can not access them by index:

```
147 unique_names[0]
148 # Traceback (most recent call last):
149 #   File "<stdin>", line 1, in <module>
150 #     TypeError: 'set' object does not support indexing
```

However, we can convert a set to a list easily:

```
151 unique_names = list(unique_names)
152 unique_names ['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay']
153 unique_names[0]
154 # 'Lincoln'
```

Notice that in this case, the order of elements in the new list matches the order in which the elements were displayed when we create the set (we had `set(['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay'])`) and now we have `['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay']`). You should not assume this is the case in general. That is, don't make any assumptions about the order of elements in a set when it is converted to any type of sequential data structure.

You can change a set's contents using the `add`, `remove` and `update` methods which correspond to the `append`, `remove` and `extend` methods in a list. In addition to these, `set` objects support the operations you may be familiar with from mathematical sets: *union*, *intersection*, *difference*, as well as operations to check containment. You can read about this in the [Python documentation for sets](#).

21.7.3 Removal and Testing for Membership in Sets

One important advantage of a `set` over a `list` is that **access to elements is fast**. If you are familiar with different data structures from a Computer Science class, the Python list is implemented by an array, while the set is implemented by a hash table.

We will demonstrate this with an example. Let's say we have a list and a set of the same number of elements (approximately 100 thousand):

```
155 import sys, random, timeit
156 nums_set = set([random.randint(0, sys.maxint) for _ in range(10**5)])
157 nums_list = list(nums_set)
158 len(nums_set)
159 # 100000
```

We will use the `timeit` Python module to time 100 operations that test for the existence of a member in either the list or set:

```
160 timeit.timeit('random.randint(0, sys.maxint) in nums',
161                 setup='import random; nums=%s' % str(nums_set), number=100)
162 # 0.0004038810729980469
163 timeit.timeit('random.randint(0, sys.maxint) in nums',
164                 setup='import random; nums=%s' % str(nums_list), number=100)
165 # 0.3980541229248047
```

The exact duration of the operations on your system will be different, but the take away will be the same: searching for an element in a set is orders of magnitude faster than in a list. This is important to keep in mind when you work with large amounts of data.

21.7.4 Dictionaries

One of the very important data structures in python is a dictionary also referred to as *dict*.

A dictionary represents a key value store:

```
166 person = { 'Name': 'Albert', 'Age': 100, 'Class': 'Scientist'}
167 print("person['Name']: ", person['Name'])
168 # person['Name']: Albert
169 print("person['Age']: ", person['Age'])
170 # person['Age']: 100
```

You can delete elements with the following commands:

```
171 del person['Name'] # remove entry with key 'Name'
172 person
173 # {'Age': 100, 'Class': 'Scientist'}
174 person.clear()      # remove all entries in dict
175 # person
176 # {}
177 del person          # delete entire dictionary
178 person
179 # Traceback (most recent call last):
180 #   File "<stdin>", line 1, in <module>
181 #     NameError: name 'person' is not defined
```

You can iterate over a dict:

```
182 person = { 'Name': 'Albert', 'Age': 100, 'Class': 'Scientist'}
183 for item in person:
184     print(item, person[item])
185
186 # Age 100
187 # Name Albert
188 # Class Scientist
```

21.7.5 Dictionary Keys and Values

You can retrieve both the keys and values of a dictionary using the `keys()` and `values()` methods of the dictionary, respectively:

```
189 person.keys()
190 # ['Age', 'Name', 'Class']
191 person.values()
192 # [100, 'Albert', 'Scientist']
```

Both methods return lists. Notice, however, that the order in which the elements appear in the returned lists (Age, Name, Class) is different from the order in which we listed the elements when we declared the dictionary initially (Name, Age, Class). It is important to keep this in mind: **you can't make any assumptions about the order in which the elements of a dictionary will be returned by the `keys()` and `values()` methods.**

However, you can assume that if you call `keys()` and `values()` in sequence, the order of elements will at least correspond in both methods. In the above example Age corresponds to 100, Name to

'Albert, and Class to Scientist, and you will observe the same correspondence in general as long as **keys() and values() are called one right after the other.**

21.7.6 Counting with Dictionaries

One application of dictionaries that frequently comes up is counting the elements in a sequence. For example, say we have a sequence of coin flips:

```
193 import random
194 die_rolls = [random.choice(['heads', 'tails']) for _ in range(10)]
195 # die_rolls
196 # ['heads', 'tails', 'heads', 'tails', 'heads', 'heads',
197     'tails', 'heads', 'heads', 'heads']
```

The actual list `die_rolls` will likely be different when you execute this on your computer since the outcomes of the die rolls are random.

To compute the probabilities of heads and tails, we could count how many heads and tails we have in the list:

```
198 counts = {'heads': 0, 'tails': 0}
199 for outcome in coin_flips:
200     assert outcome in counts
201     counts[outcome] += 1
202 print('Probability of heads: %.2f' % (counts['heads'] / len(coin_flips)))
203 # Probability of heads: 0.70
204
205 print('Probability of tails: %.2f' % (counts['tails'] / sum(counts.values())))
206 # Probability of tails: 0.30
```

In addition to how we use the dictionary `counts` to count the elements of `coin_flips`, notice a couple things about this example:

1. We used the `assert outcome in counts` statement. The `assert` statement in Python allows you to easily insert debugging statements in your code to help you discover errors more quickly. `assert` statements are executed whenever the internal Python `__debug__` variable is set to True, which is always the case unless you start Python with the `-O` option which allows you to run *optimized* Python.
2. When we computed the probability of tails, we used the built-in `sum` function, which allowed us to quickly find the total number of coin flips. `sum` is one of many built-in function you can [read about here](#).

21.8 Functions

You can reuse code by putting it inside a function that you can call in other parts of your programs. Functions are also a good way of grouping code that logically belongs together in one coherent whole. A function has a unique name in the program. Once you call a function, it will execute its body which consists of one or more lines of code:

```
207 def check_triangle(a, b, c):
208     return \
209         a < b + c and a > abs(b - c) and \
210         b < a + c and b > abs(a - c) and \
211         c < a + b and c > abs(a - b)
```

```
213 print(check_triangle(4, 5, 6))
```

The `def` keyword tells Python we are defining a function. As part of the definition, we have the function name, `check_triangle`, and the parameters of the function – variables that will be populated when the function is called.

We call the function with arguments 4, 5 and 6, which are passed in order into the parameters `a`, `b` and `c`. A function can be called several times with varying parameters. There is no limit to the number of function calls.

It is also possible to store the output of a function in a variable, so it can be reused.

```
214 def check_triangle(a, b, c):
215     return \
216         a < b + c and a > abs(b - c) and \
217         b < a + c and b > abs(a - c) and \
218         c < a + b and c > abs(a - b)
219
220 result = check_triangle(4, 5, 6)
221 print(result)
```

21.9 Classes

A class is an encapsulation of data and the processes that work on them. The data is represented in member variables, and the processes are defined in the methods of the class (methods are functions inside the class). For example, let's see how to define a `Triangle` class:

```
222 class Triangle(object):
223
224     def __init__(self, length, width, height, angle1, angle2, angle3):
225         if not self._sides_ok(length, width, height):
226             print('The sides of the triangle are invalid.')
227         elif not self._angles_ok(angle1, angle2, angle3):
228             print('The angles of the triangle are invalid.')
229
230         self._length = length
231         self._width = width
232         self._height = height
233
234         self._angle1 = angle1
235         self._angle2 = angle2
236         self._angle3 = angle3
237
238     def _sides_ok(self, a, b, c):
239         return \
240             a < b + c and a > abs(b - c) and \
241             b < a + c and b > abs(a - c) and \
242             c < a + b and c > abs(a - b)
243
244     def _angles_ok(self, a, b, c):
245         return a + b + c == 180
246
247 triangle = Triangle(4, 5, 6, 35, 65, 80)
```

Python has full object-oriented programming (OOP) capabilities, however we can not cover all of them in a quick tutorial, so please refer to the [Python docs on classes and OOP](#).

21.10 Modules

Now write this simple program and save it:

```
1 from __future__ import print_statement, division
2 print("Hello world!")
```

As a check, make sure the file contains the expected contents on the command line:

```
1 $ cat hello.py
2 from __future__ import print_statement, division
3 print("Hello world!")
```

To execute your program pass the file as a parameter to the python command:

```
1 $ python hello.py
2 Hello world!
```

Files in which Python code is stored are called **modules**. You can execute a Python module from the command line like you just did, or you can import it in other Python code using the `import` statement.

Let's write a more involved Python program that will receive as input the lengths of the three sides of a triangle, and will output whether they define a valid triangle. A triangle is valid if the length of each side is less than the sum of the lengths of the other two sides and greater than the difference of the lengths of the other two sides.:.

```
1 """Usage: check_triangle.py [-h] LENGTH WIDTH HEIGHT
2
3 Check if a triangle is valid.
4
5 Arguments:
6     LENGTH      The length of the triangle.
7     WIDTH       The width of the traingle.
8     HEIGHT      The height of the triangle.
9
10 Options:
11 -h --help
12 """
13 from __future__ import print_function, division
14 from docopt import docopt
15
16 if __name__ == '__main__':
17     args = docopt(__doc__)
18     a, b, c = int(args['LENGTH']), int(args['WIDTH']), int(args['HEIGHT'])
19     valid_triangle = \
20         a < b + c and a > abs(b - c) and \
21         b < a + c and b > abs(a - c) and \
22         c < a + b and c > abs(a - b)
23     print('Triangle with sides %d, %d and %d is valid: %r' %
24           a, b, c, valid_triangle
25     )
```

Assuming we save the program in a file called `check_triangle.py`, we can run it like so:

```
1 $ python check_triangle.py 4 5 6
2 Triangle with sides 4, 5 and 6 is valid: True
```

Let us break this down a bit.

1. We are importing the `print_function` and `division` modules from Python 3 like we did earlier in this tutorial. It's a good idea to always include these in your programs.

2. We've defined a boolean expression that tells us if the sides that were input define a valid triangle. The result of the expression is stored in the `valid_triangle` variable. `inside` are `true`, and `False` otherwise.
3. We've used the backslash symbol `\` to format the code nicely. The backslash simply indicates that the current line is being continued on the next line.
4. When we run the program, we do the check if `__name__ == '__main__'`. `__name__` is an internal Python variable that allows us to tell whether the current file is being run from the command line (value `__name__`), or is being imported by a module (the value will be the name of the module). Thus, with this statement we're just making sure the program is being run by the command line.
5. We are using the `docopt` module to handle command line arguments. The advantage of using this module is that it generates a usage help statement for the program and enforces command line arguments automatically. All of this is done by parsing the docstring at the top of the file.
6. In the `print` function, we are using [Python's string formatting capabilities](#) to insert values into the string we are displaying.

21.11 Lambda Expressions

TODO: contribute

21.12 Generators

TODO: contribute

21.13 Non Blocking Threads

TODO: contribute



22. Data Management

F section/prg/python-data.tex

Obviously when dealing with big data we may not only be dealing with data in one format but in many different formats. It is important that you will be able to master such formats and simlessly integerat in your analysis. Thus we provide some simple examples on which different data formats exist and how to use them.

22.1 Formats

22.1.1 Pickle

Python pickle allows you to save data in a python native format into a file that can later be read in by other programs. However, the data format may not be portable among different python versions thus the format is often not suitable to store information. INstead we recommend for standrad data to use either json or yaml.

```
import pickle

flavor = { "small": 100,
           "medium": 1000,
           "large": 10000 }

pickle.dump( flavor, open( "data.p", "wb" ) )
```

To read it back in use

```
flavor = pickle.load( open( "data.p", "rb" ) )
```

22.1.2 Text Files

```
content = open('filename.txt', 'r').read()

with open('filename.txt', 'r') as file:
    output = file.read()
```

To split up the files into an array you can do

```
with open('filename.txt', 'r') as file:
    lines = file.read().splitlines()
```

In case the file is too big you will want to read the file line by line:

```
lines = open('filename.txt', 'r').readlines()
```

22.1.3 CSV Files

```
import csv
with open('data.csv', 'rb') as f:
    contents = csv.reader(f)
for row in content:
    print row
```

using pandas

```
import pandas as pd
df = pd.read_csv("example.csv")
```

22.1.4 Excel spread sheets

```
import pandas as pd
filename = 'data.xlsx'
data = pd.ExcelFile(file)
df = data.parse('Sheet1')
```

22.1.5 YAML

```
import yaml
with open('data.yaml', 'r') as f:
    content = yaml.load(f)
```

22.1.6 JSON

```
import json
with open('strings.json') as f:
    content = json.load(f)
```

22.1.7 XML

Please contribute a section.

22.1.8 RDF

```
from rdflib.graph import Graph
g = Graph()
g.parse("filename.rdf", format="format")
for entry in g:
    print(entry)
```

22.1.9 PDF

The Portable Document Format (PDF) has been made available by Adobe Inc. royalty free. This has enabled PDF to become a world wide adopted format that also has been standardized in 2008 (ISO/IEC 32000-1:2008, <https://www.iso.org/standard/51502.html>). A lot of research is published in papers making PDF one of the defacto standards for publishing. However, PDF is difficult to parse and is focused on high quality print output instead of data representation. Nevertheless, tools to manipulate PDF exist:

PDFMiner <https://pypi.python.org/pypi/pdfminer/> allows the simple translation of PDF into text that can be further mined. The manual page helps to demonstrate some examples <http://euske.github.io/pdfminer/index.html>.

pdf-parser.py <https://blog.didierstevens.com/programs/pdf-tools/> parses pdf documents and identifies some structural elements that can then be further processed.

If you know about other tools, let us know.

22.1.10 HTML

Beautiful soup

please contribute a section

22.1.11 ConfigParser

- <https://pymotw.com/2/ConfigParser/>

22.1.12 ConfigDict

- <https://github.com/cloudmesh/cloudmesh.common/blob/master/cloudmesh/common/ConfigDict.py>

22.2 Encryption

Often we need to protect the information stored in a file. This is achieved with encryption. There are many methods of supporting encryption and even if a file is encrypted it may be target to attacks. Thus it is not only important to encrypt data that you do not want others to see but also to make sure that the system on which the data is hosted is secure. This is especially important if we talk about big data having a potential large effect if it gets into the wrong hands.

To illustrate one type of encryption that is non trivial we have chosen to demonstrate how to encrypt a file with an ssh key. In case you have openssl installed on your system, this can be achieved as follows.

```
#!/bin/sh

# Step 1. Creating a file with data
echo "Big Data is the future." > file.txt

# Step 2. Create the pem
openssl rsa -in ~/.ssh/id_rsa -pubout > ~/.ssh/id_rsa.pub.pem

# Step 3. look at the pem file to illustrate how it looks like (optional)
cat ~/.ssh/id_rsa.pub.pem
```

```
# Step 4. encrypt the file into secret.txt
openssl rsa -encrypt -pubin -inkey ~/.ssh/id_rsa.pub.pem -in file.txt -out secret.txt

# Step 5. decrypt the file and print the contents to stdout
openssl rsa -decrypt -inkey ~/.ssh/id_rsa -in secret.txt
```

Most important here are Step 4 that encrypts the file and Step 5 that decrypts the file. Using the Python os module it is straight forward to implement this. However, we are providing in cloudmesh a convenient class that makes the use in python very simple.

```
from cloudmesh.common.ssh.encrypt import EncryptFile

e = EncryptFile('file.txt', 'secret.txt')
e.encrypt()
e.decrypt()
```

In our class we initialize it with the locations of the file that is to be encrypted and decrypted. To initiate that action just call the methods `encrypt` and `decrypt`.

22.3 Database Access

TODO: Add more conventional database access

see: https://www.tutorialspoint.com/python/python_database_access.htm

22.3.1 Exercises

Exercise 22.1 Test out the shell script to replicate how this example works □

Exercise 22.2 Test out the cloudmesh encryption class □

Exercise 22.3 What other encryption methods exist. Can you provide an example and contribute to the section? □

Exercise 22.4 What is the issue of encryption that make it challenging for Big Data □

Exercise 22.5 Given a test dataset with many files text files, how long will it take to encrypt and decrypt them on various machines. Write a benchmark that you test. Develop this benchmark as a group, test out the time it takes to execute it on a variety of platforms. □



23. Libraries

23.1 Installing Libraries

Often you may need functionality that is not present in Python's standard library. In this case you have two option:

- implement the features yourself
- use a third-party library that has the desired features.

Often you can find a previous implementation of what you need. Since this is a common situation, there is a service supporting it: the [Python Package Index](#) (or PyPi for short).

Our task here is to install the `autopep8` tool from PyPi. This will allow us to illustrate the use if virtual environments using the `pyenv` or `virtualenv` command, and installing and uninstalling PyPi packages using `pip`.

23.2 Using pip to Install Packages

Let's now look at another important tool for Python development: the Python Package Index, or PyPI for short. PyPI provides a large set of third-party python packages. If you want to do something in python, first check pypi, as odd are someone already ran into the problem and created a package solving it.

In order to install package from PyPI, use the `pip` command. We can search for PyPI for packages:

```
$ pip search --trusted-host pypi.python.org autopep8 pylint
```

It appears that the top two results are what we want so install them:

```
$ pip install --trusted-host pypi.python.org autopep8 pylint
```

This will cause `pip` to download the packages from PyPI, extract them, check their dependencies and install those as needed, then install the requested packages.

You can skip ‘–trusted-host pypi.python.org’ option if you have patched urllib3 on Python 2.7.9.

23.3 GUI

23.3.1 GUIZero

Install guizero with the following command:

```
sudo pip3 install guizero
```

For a comprehensive tutorial on guizero, [click here](#).

23.3.2 Kivy

You can install Kivy on OSX as follows:

```
brew install pkg-config sdl2 sdl2_image sdl2_ttf sdl2_mixer gstreamer
pip install -U Cython
pip install kivy
pip install pygame
```

A hello world program for kivy is included in the cloudmesh.robot repository. Which you can find here

- <https://github.com/cloudmesh/cloudmesh.robot/tree/master/projects/kivy>

To run the program, please download it or execute it in cloudmesh.robot as follows:

```
cd cloudmesh.robot/projects/kivy
python swim.py
```

To create stand alone packages with kivy, please see:

- <https://kivy.org/docs/guide/packaging-osx.html>

23.4 Formatting and Checking Python Code

First, get the bad code:

```
$ wget --no-check-certificate http://git.io/pXqb -O bad_code_example.py
```

Examine the code:

```
$ emacs bad_code_example.py
```

As you can see, this is very dense and hard to read. Cleaning it up by hand would be a time-consuming and error-prone process. Luckily, this is a common problem so there exist a couple packages to help in this situation.

23.5 Using autopep8

We can now run the bad code through autopep8 to fix formatting problems:

```
$ autopep8 bad_code_example.py >code_example_autopep8.py
```

Let us look at the result. This is considerably better than before. It is easy to tell what the example1 and example2 functions are doing.

It is a good idea to develop a habit of using autopep8 in your python-development workflow. For instance: use autopep8 to check a file, and if it passes, make any changes in place using the -i flag:

```
$ autopep8 file.py      # check output to see if passes  
$ autopep8 -i file.py # update in place
```

If you use pyCharm you have the ability to use a similar function while pressing on Inspect Code.

23.6 Writing Python 3 Compatible Code

To write python 2 and 3 compatible code we recommend that you take a look at: http://python-future.org/compatible_idioms.html

23.7 Using Python on FutureSystems

This is only important if you use FutureSystems resources.

In order to use Python you must log into your FutureSystems account. Then at the shell prompt execute the following command:

```
$ module load python
```

This will make the python and virtualenv commands available to you.

The details of what the module load command does are described in the future lesson modules.

23.8 Ecosystem

23.8.1 pypi

Link: [pypi](#)

The Python Package Index is a large repository of software for the Python programming language containing a large number of packages [link]. The nice thing about pypi is that many packages can be installed with the program ‘pip’.

To do so you have to locate the <package_name> for example with the search function in pypi and say on the commandline:

```
pip install <package_name>
```

where package_name is the string name of the package. an example would be the package called cloudmesh_client which you can install with:

```
pip install cloudmesh_client
```

If all goes well the package will be installed.

23.8.2 Alternative Installations

The basic installation of python is provided by python.org. However others claim to have alternative environments that allow you to install python. This includes

- [Canopy](#)
- [Anaconda](#)
- [IronPython](#)

Typically they include not only the python compiler but also several useful packages. It is fine to use such environments for the class, but it should be noted that in both cases not every python library may be available for install in the given environment. For example if you need to use cloudmesh client, it may not be available as conda or Canopy package. This is also the case for many other cloud related and useful python libraries. Hence, we do recommend that if you are new to python to use the distribution from python.org, and use pip and virtualenv.

Additionally some python version have platform specific libraries or dependencies. For example coca libraries, .NET or other frameworks are examples. For the assignments and the projects such platform dependent libraries are not to be used.

If however you can write a platform independent code that works on Linux, OSX and Windows while using the python.org version but develop it with any of the other tools that is just fine. However it is up to you to guarantee that this independence is maintained and implemented. You do have to write requirements.txt files that will install the necessary python libraries in a platform independent fashion. The homework assignment PRG1 has even a requirement to do so.

In order to provide platform independence we have given in the class a “minimal” python version that we have tested with hundreds of students: python.org. If you use any other version, that is your decision. Additionally some students not only use python.org but have used iPython which is fine too. However this class is not only about python, but also about how to have your code run on any platform. The homework is designed so that you can identify a setup that works for you.

However we have concerns if you for example wanted to use chameleon cloud which we require you to access with cloudmesh. cloudmesh is not available as conda, canopy, or other framework package. Cloudmesh client is available form pypi which is standard and should be supported by the frameworks. We have not tested cloudmesh on any other python version than python.org which is the open source community standard. None of the other versions are standard.

In fact we had students over the summer using canopy on their machines and they got confused as they now had multiple python versions and did not know how to switch between them and activate the correct version. Certainly if you know how to do that, than feel free to use canopy, and if you want to use canopy all this is up to you. However the homework and project requires you to make your program portable to python.org. If you know how to do that even if you use canopy, anaconda, or any other python version that is fine. Graders will test your programs on a python.org installation and not canopy, anaconda, ironpython while using virtualenv. It is obvious why. If you do not know that answer you may want to think about that every time they test a program they need to do a new virtualenv and run vanilla python in it. If we were to run two installs in the same system, this will not work as we do not know if one student will cause a side effect for another. Thus we as instructors do not just have to look at your code but code of hundreds of students with different setups. This is a non scalable solution as every time we test out code from a student we would have to wipe out the OS, install it new, install a new version of whatever python you have elected, become familiar with that version and so on and on. This is the reason why the open source community is using python.org. We follow best practices. Using other versions is not a community best practice, but may work for an individual.

We have however in regards to using other python version additional bonus projects such as

- deploy run and document cloudmesh on ironpython
- deploy run and document cloudmesh on anaconda, develop script to generate a conda package from github
- deploy run and document cloudmesh on canopy, develop script to generate a conda package from github

- deploy run and document cloudmesh on ironpython
- other documentation that would be useful

23.9 Resources

If you are unfamiliar with programming in Python, we also refer you to some of the numerous online resources. You may wish to start with [Learn Python](#) or the book [Learn Python the Hard Way](#). Other options include [Tutorials Point](#) or [Code Academy](#), and the Python wiki page contains a long list of [references for learning](#) as well. Additional resources include:

- <https://virtualenvwrapper.readthedocs.io>
- <https://github.com/yyuu/pyenv>
- <https://amaral.northwestern.edu/resources/guides/pyenv-tutorial>
- <https://godjango.com/96-django-and-python-3-how-to-setup-pyenv-for-multiple-pythons/>
- <https://www.accelebrate.com/blog/the-many-faces-of-python-and-how-to-manage-them/>
- <http://ivory.idyll.org/articles/advanced-swc/>
- <http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html>
- <http://www.youtube.com/watch?v=0vJJ1VBVTFg>
- <http://www.korokithakis.net/tutorials/python/>
- <http://www.afterhoursprogramming.com/tutorial/Python/Introduction/>
- <http://www.greenteapress.com/thinkpython/thinkCSPy.pdf>
- <https://docs.python.org/3.3/tutorial/modules.html>
- https://www.learnpython.org/en/Modules/_and_/_Packages
- <https://docs.python.org/2/library/datetime.html>
- https://chrisalbon.com/python/strings/_to/_datetime.html

A very long list of useful information are also available from

- <https://github.com/vinta/awesome-python>
- https://github.com/rasbt/python_reference

This list may be useful as it also contains links to data visualization and manipulation libraries, and AI tools and libraries. Please note that for this class you can reuse such libraries if not otherwise stated.

23.9.1 Jupyter Notebook Tutorials

A Short Introduction to Jupyter Notebooks and NumPy To view the notebook, open this link in a background tab <https://nbviewer.jupyter.org/> and copy and paste the following link in the URL input area <https://cloudmesh.github.io/classes/lesson/prg/Jupyter-NumPy-tutorial-I523-F2017.ipynb> Then hit Go.

23.10 Exercises

Exercise 23.1 Write a python program called iterate.py that accepts an integer n from the command line. Pass this integer to a function called iterate.

The iterate function should then iterate from 1 to n. If the ith number is a multiple of three, print

“multiple of 3”, if a multiple of 5 print “multiple of 5”, if a multiple of both print “multiple of 3 and 5”, else print the value.

- Exercise 23.2**
1. Create a pyenv or virtualenv ~/ENV
 2. Modify your ~/.bashrc shell file to activate your environment upon login.
 3. Install the docopt python package using pip
 4. Write a program that uses docopt to define a commandline program. Hint: modify the iterate program.
 5. Demonstrate the program works and submit the code and output.

23.11 Python for Big Data

23.11.1 An Example with Pandas, NumPy and Matplotlib

In this example, we will download some traffic citation data for the city of Bloomington, IN, load it into Python and generate a histogram. In doing so, you will be exposed to important Python libraries for working with big data such as [numpy](#), [pandas](#) and [matplotlib](#).

Set Up Directories and Get Test Data

Data.gov is a government portal for open data and the [city of Bloomington, Indiana makes available a number of datasets there](#).

We will use traffic citations data for 2016.

To start, let's create a separate directory for this project and download the CSV data:

```
1 $ cd ~/projects/i524
2 $ mkdir btown-citations
3 $ cd btown-citations
4 $ wget https://data.bloomington.in.gov/dataset/c543f0c1-1e37-46ce-a0ba-e0a949bd248a/resource/24841976-fd35-4483-a2b4-573bd1e77cfb/download/2016-first-quarter-citations.csv
```

Depending on your directory organization, the above might be slightly different for you.

If you go to the link to data.gov for Bloomington above, you will see that the citations data is organized per quarter, so there are a total of four files. Above, we downloaded the data for the first quarter. Go ahead and download the remaining three files with wget.

In this example, we will use three modules, [numpy](#), [pandas](#) and [matplotlib](#). If you set up [virtualenv](#) as described in the Python tutorial <python_intro>, the first two of these are already installed for you. To install [matplotlib](#), make sure you've activated your [virtualenv](#) and use [pip](#):

```
5 $ source ~/ENV/bin/activate
6 $ pip install matplotlib
```

If you are using a different distribution of Python, you will need to make sure that all three of these modules are installed.

Load Data in Pandas

From the same directory where you saved the citations data, let's start the Python interpreter and load the citations data for Q1 2016

```

1 $ python
2 >>> from __future__ import division, print_function
3 >>> import numpy as np
4 >>> import pandas as pd
5 >>> import matplotlib.pyplot as plt
6 >>> data = pd.read_csv('2016-first-quarter-citations.csv')

```

If the first `import` statement seems confusing, take a look at the Python tutorial `<python_intro>`. The next three `import` statements load each of the modules we will use in this example. The final line uses Pandas' `read_csv` function to load the data into a Pandas DataFrame data structure.

Working with DataFrames

You can verify that you are working with a DataFrame and use some of its methods to take a look at the structure of the data as follows:

```

1 >>> type(data)
2 <class 'pandas.core.frame.DataFrame'>
3 >>> data.index
4 Int64Index([ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
5 ...
6 197, 198, 199, 200, 201, 202, 203, 204, 205, 206],
7 dtype='int64', length=200)
8 >>> data.columns
9 Index([u'Citation Number', u'Date Issued', u'Time Issued', u'Location',
10 u'District', u'Cited Person Age', u'Cited Person Sex',
11 u'Cited Person Race', u'Offense Code', u'Offense Description',
12 u'Officer Age', u'Officer Sex', u'Officer Race'],
13 dtype='object')
14 >>> data.dtypes
15 Citation Number          object
16 Date Issued              object
17 Time Issued              object
18 Location                  object
19 District                  object
20 Cited Person Age         float64
21 Cited Person Sex          object
22 Cited Person Race         object
23 Offense Code              object
24 Offense Description       object
25 Officer Age                float64
26 Officer Sex                object
27 Officer Race                object
28 dtype: object
29 >>> data.shape
30 (200, 15)

```

As you can see from the `columns` field, when the CSV file was read, the header line was used to populate the name of the columns in the DataFrame. In addition, you will notice that `read_csv` correctly inferred the data type of some columns like `Age`, but not of others like `Date Issued` and `Time Issued`. `read_csv` is a very customizable function and in general, you can correct issues like this using the `dtype` and `converters` parameters. In this specific case, it makes more sense to combine the `Date Issued` and `Time Issued` columns into a new column containing a time stamp. We will see how to do this shortly.

You can also look at the data itself with the `DataFrame`'s `head()` and `tail()` methods:

```

1 >>> data.head()
2 <Output omitted for brevity>
3 >>> data.tail()
4 <Output omitted for brevity>
```

In addition to letting you examine your data easily, `DataFrames` have methods that help you deal with missing values:

```

1 >>> data = data.dropna(how='any')
2 >>> data.shape
```

Adding columns to the data is also easy. Here, we add two columns. First, a `datetime` column that is a combination of the `Date Issued` and `Time Issued` columns originally in the data. Second, a column identifying what day of the week each citation was given. To understand this example better, take a look at the Python docs for the `strptime` and `strftime` functions in the `datetime` module linked above.

```

1 >>> from datetime import datetime
2 >>> data['DateTime Issued'] = data.apply(
3 ...     lambda row: datetime.strptime(row['Date Issued'] + ':' + row['Time
4 ...     Issued'], '%m/%d/%y:%I:%M %p'), axis=1
5 ... )
6 >>> data.columns
7 >>> data['Day of Week Issued'] = data.apply(
8 ...     lambda row: datetime.strftime(row['DateTime Issued'], '%A'), axis=1
... )
```

Plotting with Matplotlib and NumPy

Let's say we want to see how many citations were given each day of the week. We gather the data first:

```

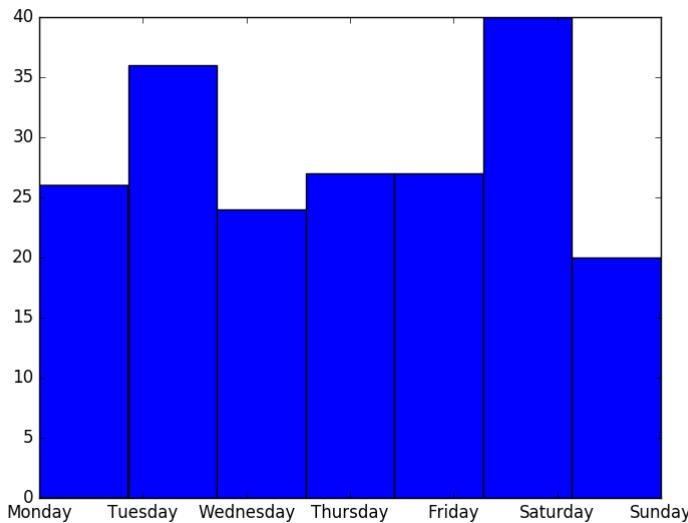
1 >>> days = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', '
2     Saturday', 'Sunday']
3 >>> dow_data = [days.index(dow) for dow in data['Day of Week Issued']]
4 <Output omitted for brevity>
```

Then we use `matplotlib` to plot it:

```

1 >>> fig = plt.figure()
2 >>> ax = fig.add_subplot(1, 1, 1)
3 >>> plt.hist(dow_data, bins=len(days))
4 >>> plt.xticks(range(len(days)), days)
5 >>> plt.show()
```

You should see something like this on your screen:

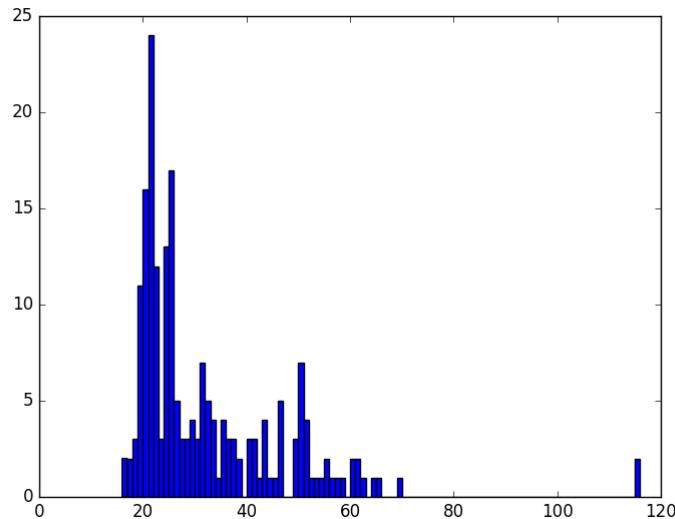


More DataFrame Manipulation and Plotting

DataFrames and numpy give us other ways to manipulate data. For example, we can plot a histogram of the ages of violators like this:

```

1 >>> ages = data[ 'Cited Person Age' ].astype( int )
2 >>> fig = plt.figure()
3 >>> ax = fig.add_subplot(1, 1, 1)
4 >>> plt.hist(ages, bins=np.max(ages) - np.min(ages))
5 >>> plt.show()
```

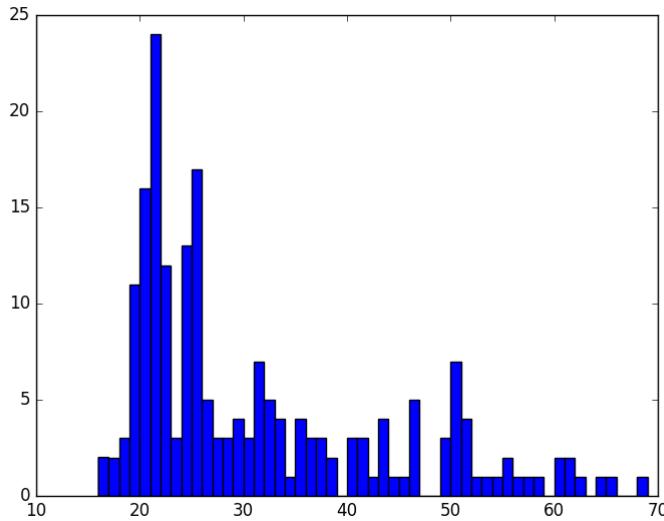


Surprisingly, we see some 116 year-old violators! This is probably an error in the data, so we can remove these data points easily and plot the histogram again:

```

1 >>> ages = ages[ages < 100]
2 >>> fig = plt.figure()
3 >>> ax = fig.add_subplot(1, 1, 1)
4 >>> plt.hist(ages, bins=np.max(ages) - np.min(ages))
```

```
5 >>> plt.show()
```



Saving Plots to PDF

Oftentimes, you will want to save your `matplotlib` graph as a PDF or an SVG file instead of just viewing it on your screen. For both, we need to create a `figure` and plot the histogram as before:

```
1 >>> fig = plt.figure()
2 >>> ax = fig.add_subplot(1, 1, 1)
3 >>> plt.hist(ages, bins=np.max(ages) - np.min(ages))
```

Then, instead of calling `plt.show()` we can invoke `plt.savefig()` to save as SVG:

```
1 >>> plt.savefig('hist.svg')
```

If we want to save the figure as PDF instead, we need to use the `PdfPages` module together with `savefig()`:

```
1 >>> import matplotlib.patches as mpatches
2 >>> from matplotlib.backends.backend_pdf import PdfPages
3 >>> pp = PdfPages('hist.pdf')
4 >>> fig.savefig(pp, format='pdf')
5 >>> pp.close()
```

Next Steps and Exercises

There is a lot more to working with `pandas`, `numpy` and `matplotlib` than we can show you here, but hopefully this example has piqued your curiosity.

Don't worry if you don't understand everything in this example. For a more detailed explanation on these modules and the examples we did, please take a look at the tutorials below. The `numpy` and `pandas` tutorials are mandatory if you want to be able to use these modules, and the `matplotlib` gallery has many useful code examples.

23.11.2 Summary of Useful Libraries

Numpy

- <http://www.numpy.org/>

According to the Numpy Web page "NumPy is a package for scientific computing with Python. It contains a powerful N-dimensional array object, sophisticated (broadcasting) functions, tools for integrating C/C++ and Fortran code, useful linear algebra, Fourier transform, and random number capabilities

Tutorial: <https://docs.scipy.org/doc/numpy-dev/user/quickstart.html>

Matplotlib

- <http://matplotlib.org/>

According to the Matplotlib Web page, “matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB®*) or Mathematica®†), web application servers, and six graphical user interface toolkits.”

Matplotlib Gallery: <http://matplotlib.org/gallery.html>

Pandas

- <http://pandas.pydata.org/>

According to the Pandas Web page, “Pandas is a library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.”

In addition to access to charts via matplotlib it has elementary functionality for conducting data analysis. Pandas may be very suitable for your projects.

Tutorial: <http://pandas.pydata.org/pandas-docs/stable/10min.html>

Pandas Cheat Sheet: https://github.com/pandas-dev/pandas/blob/master/doc/cheatsheet/Pandas_Cheat_Sheet.pdf

23.11.3 Big Data Libraries

Scipy

- <https://www.scipy.org/>

According to the SciPy Web page, “SciPy (pronounced “Sigh Pie”) is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

- NumPy
- IPython
- Pandas
- Matplotlib
- Sympy
- SciPy library

It is thus an agglomeration of useful packages and will probably suffice for your projects in case you use Python.

ggplot

- <http://ggplot.yhathq.com/>

According to the ggplot python Web page ggplot is a plotting system for Python based on R's ggplot2. It allows to quickly generate some plots quickly with little effort. Often it may be easier to use than matplotlib directly.

seaborn

- http://www.data-analysis-in-python.org/t_seaborn.html

The good library for plotting is called seaborn which is build on top of matplotlib. It provides high level templates for common statistical plots.

- Gallery: <http://stanford.edu/~mwaskom/software/seaborn/examples/index.html>
- Original Tutorial: <http://stanford.edu/~mwaskom/software/seaborn/tutorial.html>
- Additional Tutorial: <https://stanford.edu/~mwaskom/software/seaborn/tutorial/distributions.html>

Here are some examples from a previous class:

- <https://github.com/bigdata-i523/hid231/blob/master/experiment/seaborn/seaborn-exercises.ipynb>
- https://github.com/bigdata-i523/hid231/blob/master/experiment/learning-jupyter/learning-jupyter_notebook.ipynb

Exercise 23.3 Take these examples and create sections in latex that can be added to the book. Describe the process.

1. export the ipynb as rst 2. use pandoc to export it to tex 3. do some cleanup on the tex files

Can this be automated with a cmd5 script such as

```
7 cms ipynb [ url=URL | file=FILE ] --output FILENAME
```

Bokeh

Bokeh is an interactive visualization library with focus on web browsers for display. Its goal is to provide a similar experience as D3.js

- URL: <http://bokeh.pydata.org/>
- Gallery: <http://bokeh.pydata.org/en/latest/docs/gallery.html>

pygal

Pygal is a simple API to produce graphs that can be easily embedded into your Web pages. It contains annotations when you hover over data points. It also allows to present the data in a table.

- <http://pygal.org/>

Network and Graphs

- igraph: http://www.pythonforsocialscientists.org/t_igraph.html
- networkx: <https://networkx.github.io/>

REST

- django REST FFramework <http://www.django-rest-framework.org/>
- flask <https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask>
- requests <https://realpython.com/blog/python/api-integration-in-python/>
- urllib2 <http://rest.elkstein.org/2008/02/using-rest-in-python.html> (not recommended)
- web <http://www.dreamsyssoft.com/python-scripting-tutorial/create-simple-rest-web-service-with-python.php> (not recommended)
- bottle <http://bottlepy.org/docs/dev/index.html>
- falcon <https://falconframework.org/>
- eve <http://python-eve.org/>
- <https://code.tutsplus.com/tutorials/building-rest-apis-using-eve--cms-22961>

23.12 Parsing Data

Being able to parse data is an important activity in the data analysis process. Not all data may be following aspecific format and the data may need to be extracted.

23.12.1 notebook.md Parser

We are using a notebook.md to communicate what students have done throughout the semester. We like to make a simble cmd5 command that parses the notebook.md file and check it upon correctness.

An example for a notebook.md file is lokated here

- <https://raw.githubusercontent.com/bigdata-i523/sample-hid000/master/notebook.md>

The following code may inspire you

- <https://github.com/bigdata-i523/hid203/tree/master/experiment>

We like to implement the follwoing functionality and use docopts to document the command.

```

1 cms class notebook [--git=GITREPONAME] --verify hid
2
3     verifies the correctness of the notebook.md file
4
5 cms class notebook [--git=GITREPONAME] --log
6
7     displays the log of the notebook.md
8
9 cms class notebook [--git=GITREPONAME] --history
10
11    displays a true or false for each week since the first occurance
12        of the notebook.md file in the git repository

```

Exercise 23.4 Write a notebook.md parser

Exercise 23.5 How can this command be generalized to provide not only information for a student but to provide information for the class. Example: can we identify prefered days of when the notebooks are checked in. Can we identify a list of students that have not updated the

notebook for a week? Can we identify the list of student s that have updated the notebook for a week?

23.12.2 Video Length

The Latex source of this class contains a macro to include videos.

Given a LateX file, can you create a table that includes the names of all videos in that file and sums up the total viewing time. Previously the document was stored in RST and the code from a previous student may inspire you. Can you reprise it in for LaTeX?

- <https://github.com/bigdata-i523/hid107/blob/master/cloudmesh/bar/command/mycommand.py>

```

1 cms class video list FILENAME --output=[tabular|longtable|csv|txt]
2
3     prints the videolist in the given format. txt means it is just ASCII

```

Exercise 23.6 Write a tool that extracts the information for video length.

Exercise 23.7 Write a tool that finds all youtube urls that are not in a video latex macro.

23.12.3 Dask

Many times operations need to be done on data in parallel to utilize modern processor architectures.

Dask provides a *dynamic task scheduling* which is optimized for computation. It is similar to other frameworks such as Airflow, Luigi, Celery, or Make. However it is specializing in optimized interactive computational workloads.

Furthermore, Dask targets Big Data *collections* such as parallel arrays, dataframes, and lists. These collections are commonly found in NumPy, Pandas, or Python iterators to larger-than-memory or distributed environments. While using the Dask implementation we can replace the original imports from the appropriate framework, replace them with Dask imports and implicitly use parallel collections that utilize internally the dynamic task schedulers.

More information can be found at:

- <https://dask.pydata.org>

Exercise 23.8 Conduct a performance study that showcases the difference of doing parallel calculations in Dask, calculations in a framework such as SciPy, and regular unthreaded python code.



24. Cloudmesh Command Shell

F section/prg/python-cmd5.tex

24.1 CMD5

Python's CMD (<https://docs.python.org/2/library/cmd.html>) is a very useful package to create command line shells. However it does not allow the dynamic integration of newly defined commands. Furthermore, additions to CMD need to be done within the same source tree. To simplify developing commands by a number of people and to have a dynamic plugin mechanism, we developed cmd5. It is a rewrite on our earlier efforts in cloudmesh client and cmd3.

24.1.1 Resources

The source code for cmd5 is located in github:

- <https://github.com/cloudmesh/cmd5>

24.1.2 Creating a Python Development Environment

We recommend that you use a virtualenv either with virtualenv or pyenv. This is in detail documented in the Section `section_pyenv`.

24.1.3 Installation from source

Cmd5 can easily be deployed with pip:

```
pip install cloudmesh.cmd5
```

In case you like to generate easily new cmd 5 commands we also recommend you install the

cloudmesh sys command with:

```
pip install cloudmesh.sys
```

In case you like to work with the source please clone the following directories from github:

```
mkdir -p ~/github
cd ~/github

git clone https://github.com/cloudmesh/cloudmesh.common.git
git clone https://github.com/cloudmesh/cloudmesh.cmd5.git
git clone https://github.com/cloudmesh/cloudmesh.sys.git

cd ~/github/cloudmesh.common
python setup.py install
pip install .

cd ~/github/cloudmesh.cmd5
python setup.py install
pip install .

cd ~/github/cloudmesh.sys
python setup.py install
pip install .
```

The common directory contains some useful libraries, the cmd5 repository contains the shell, while the sys directory contains a command to generate extensions to cloudmesh.

24.1.4 Execution

To run the shell you can activate it with the cms command. cms stands for cloudmesh shell:

```
(ENV2) $ cms
```

It will print the banner and enter the shell:

```
+-----+
|   /---\ | --- - - - | /---\ | --- - - - | /---\ | --- - - - | | | | | | | | | | | | | | | | | | |
|   | | | | | / \ | | | | / \ | | | | / \ | | | | / \ | | | |
|   | | | | | ( ) | | | | | ( ) | | | | | | | | | | | | | |
|   \---\ | \---/ \---\ | \---\ | \---\ | \---\ | \---\ | \---\ |
+-----+
|           Cloudmesh CMD5 Shell           |
+-----+
```

cms>

To see the list of commands you can say:

```
cms> help
```

To see the manula page for a specific command, please use:

```
help COMMANDNAME
```

24.1.5 Create your own Extension

One of the most important features of CMD5 is its ability to extend it with new commands. This is done via packaged name spaces. We recommend you name is cloudmesh.mycommand, where mycommand is the name of the command that you like to create. This can easily be done while using the sys command:

```
cms sys command generate mycommand
```

It will download a template form cloudmesh called cloudmesh.bar and generate a new directory cloudmesh.mycommand with all the needed files to create your own command and register it dynamically with cloudmesh. All you have to do is to cd into the directory and install the code:

```
cd cloudmesh.mycommand
python setup.py install
pip install .
```

Adding your own command is easy. It is important that all objects are defined in the command itself and that no global variables be use in order to allow each shell command to stand alone. Naturally you should develop API libraries outside of the cloudmesh shell command and reuse them in order to keep the command code as small as possible. We place the command in:

```
cloudmsesh/mycommand/command/mycommand.py
```

An example for the bar command is presented at:

- <https://github.com/cloudmesh/cloudmesh/blob/master/cloudmesh/bar/command/bar.py>

It shows how simple the command definition is (bar.py):

```
from __future__ import print_function
from cloudmesh.shell.command import command
from cloudmesh.shell.command import PluginCommand

class BarCommand(PluginCommand):

    @command
    def do_bar(self, args, arguments):
        """
        ::

        Usage:
            command -f FILE
            command FILE
            command list
        This command does some useful things.
        Arguments:
            FILE    a file name
        Options:
            -f      specify the file
        """
        print(arguments)
```

An important difference to other CMD solutions is that our commands can leverage (besides the standrad definition), docopts as a way to define the manual page. This allows us to use arguments as dict and use simple if conditions to interpret the command. Using docopts has the advantage that contributors are forced to think about the command and its options and document them from the start. Previously we did not use but argparse and click. However we noticed that for our contributors both systems lead to commands that were either not properly documented or the developers delivered ambiguous commands that resulted in confusion and wrong ussage by subsequent users. Hence, we do recommend that you use docopts for documenting cmd5 commands. The transformation is enabled by the @command decorator that generates a manual page and creates a proper help message for the shell automatically. Thus there is no need to introduce a sepaarte help method as would normally be needed in CMD while reducing the effort it takes to contribute new commands in a dynamic fashion.

24.1.6 Exercises

Exercise 24.1 Install cmd5 on your computer. ■

Exercise 24.2 Write a new command with your firstname as the command name. ■

Exercise 24.3 Write a new command and experiment with docopt syntax and argument interpretation of the dict with if conditions. ■

Exercise 24.4 If you have useful extensions that you like us to add by default, please work with us. ■

Python - Advanced

25	Numpy	277
25.1	Float Range	
25.2	Arrays	
25.3	Array Operations	
25.4	Linear Algebra	
25.5	Resources	
26	Scipy	281
26.1	Introduction	
26.2	References	
27	OpenCV	287
27.1	Overview	
27.2	Installation	
27.3	A Simple Example	
27.4	Additional Features	
27.5	Secchi Disk	
27.6	Setup for OSX	
27.7	Black and white	
28	NIST Pedestrian and Face Detection	297
29	Python Fingerprint Example	309
29.1	Overview	
29.2	Objectives	
29.3	Prerequisites	
29.4	Implementation	
29.5	Utility functions	
29.6	Dataset	
29.7	Data Model	
29.8	Plotting	
29.9	Putting it all Together	



25. Numpy

NumPy is a popular library on that is used by many other python librairiessuch as pandas, and SciPy. It provides simple to use array operations for data. This helps to accass arrays in a more intuitive fashion and introduces various matrix operations.

We provide a short introduction to Numpy.

First we import the modules needed for this introduction and abreviate them with the as feature of the import statement

```
1 import numpy as np
2 import matplotlib as mpl
3 import matplotlib.pyplot as plt
```

Now we showcase some features of Numpy.

25.1 Float Range

`arange()` is like `range()`, but for floating-point numbers.

```
1 X = np.arange(0.2, 1, .1)
1 print (X)
```

We use this function to generate parameter space that can then be iterated on.

```
1 P = 10.0 ** np.arange(-7, 1, 1)
2
3 print (P)
1 for x,p in zip(X,P):
2     print ('%f, %f' % (x, p))
```

25.2 Arrays

To create one dimensional arrays you use

```
1 a = np.array([1, 2, 3])
```

To check some properties you can use the following

```
1 print(type(a))          # Prints <class 'numpy.ndarray'>
2 print(a.shape)           # Prints (3,)
```

The shape indicates that in the first dimension, there are 3 elements. To print the actual values you can use

```
1 print(a)                # Prints the values of the array
2 print(a[0], a[1], a[2])  # Prints "1 2 3"
```

To change values you can use the index of the element or use any other python method to do so. IN our example we change the first element to 42

```
1 a[0] = 42
2 print(a)
```

To create more dimensional arrays you use

```
1 b = np.array([[1,2,3],[4,5,6]])    # Create a 2 dimensional array
2 print(b.shape)                      # Prints (2, 3)
3 print(b[0, 0], b[0, 1], b[1, 0])   # Prints "1 2 4"
```

25.3 Array Operations

Let us first create some arrays with a predefined datatype

```
1 x = np.array([[1,2],[3,4]], dtype=np.float64)
2 y = np.array([[5,6],[7,8]], dtype=np.float64)

1 print(x)

1 print(y)
```

To add the numbers use

```
1 print(x+y)
```

Other functions such as `-`, `*`, `/` behave as expected using elementwise operations:

```
1 print(x-y)
1 print(x*y)
1 print(x/y)
```

To apply functions such as ‘sin’ make sure you use the function provided by the numpy package such as `np.sin`. The list of functions is included in the manual at * <https://docs.scipy.org/doc/numpy/reference/routines.math.html>

```
1 print(np.sin(x))
```

```
1 print (np.sum(x))
```

Computations can also be applied to columns and rows

```
1 print(np.sum(x, axis=0)) # sum of each column
2 print(np.sum(x, axis=1)) # sum of each row
```

25.4 Linear Algebra

Linear algebra methods are also provided.

```
1 from numpy import linalg
2
3 A = np.diag((1,2,3))
4
5 w,v = linalg.eig(A)
6
7 print ('w =', w)
8 print ('v =', v)
```

25.5 Resources

- <https://docs.scipy.org/doc/numpy/>
- <http://cs231n.github.io/python-numpy-tutorial/#numpy>



26. Scipy

SciPy is a library built above numpy and has a number of off the shelf algorithms and operations implemented. These include algorithms from calculus (such as integration), statistics, linear algebra, image-processing, signal processing, machine learning.

To achieve this, SciPy bundles a number of useful open-source software for mathematics, science, and engineering. It includes the following packages:

- NumPy**, for managing N-dimensional arrays
- SciPy library**, to access fundamental scientific computing capabilities
- Matplotlib**, to conduct 2D plotting
- IPython**, for an Interactive console (see jupyter)
- Sympy**, for symbolic mathematics
- pandas**, for providing data structures and analysis

26.1 Introduction

First we add the usual scientific computing modules with the typical abbreviations, including sp for scipy. We could invoke scipy's statistical package as sp.stats, but for the sake of laziness we abbreviate that too.

```
import numpy as np # import numpy
import scipy as sp # import scipy
from scipy import stats # refer directly to stats rather than sp.stats
import matplotlib as mpl # for visualization
from matplotlib import pyplot as plt # refer directly to pyplot
# rather than mpl.pyplot
```

Now we create some random data to play with. We generate 100 samples from a Gaussian distribution centered at zero.

```
s = sp.randn(100)
```

How many elements are in the set?

```
print ('There are',len(s),'elements in the set')
```

What is the mean (average) of the set?

```
print ('The mean of the set is',s.mean())
```

What is the minimum of the set?

```
print ('The minimum of the set is',s.min())
```

What is the maximum of the set?

```
print ('The maximum of the set is',s.max())
```

We can use the scipy functions too. What's the median?

```
print ('The median of the set is',sp.median(s))
```

What about the standard deviation and variance?

```
print ('The standard deviation is',sp.std(s),
      'and the variance is',sp.var(s))
```

Isn't the variance the square of the standard deviation?

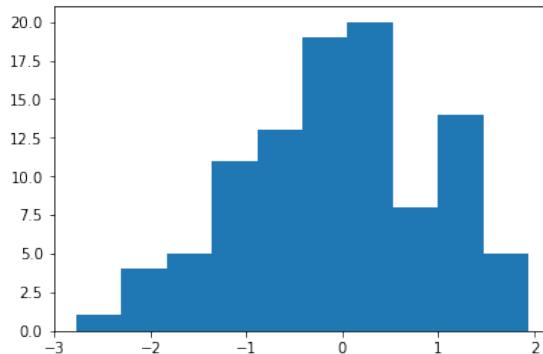
```
print ('The square of the standard deviation is',sp.std(s)**2)
```

How close are the measures? The differences are close as the following calculation shows

```
print ('The difference is',abs(sp.std(s)**2 - sp.var(s)))
print ('And in decimal form, the difference is %0.16f' %
      (abs(sp.std(s)**2 - sp.var(s))))
```

How does this look as a histogram?

```
plt.hist(s) # yes, one line of code for a histogram
plt.show()
```

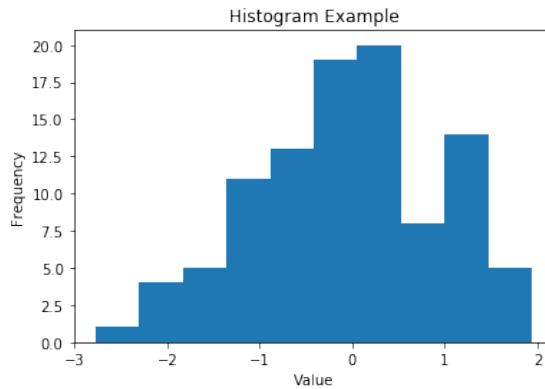


Let's add some titles.

```
plt.clf() # clear out the previous plot

plt.hist(s)
plt.title("Histogram Example")
plt.xlabel("Value")
plt.ylabel("Frequency")

plt.show()
```

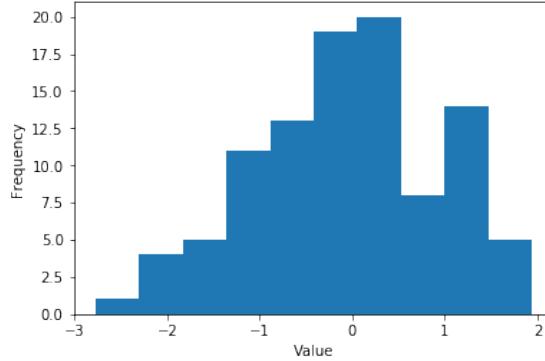


Typically we do not include titles when we prepare images for inclusion in LaTeX. There we use the caption to describe what the figure is about.

```
plt.clf() # clear out the previous plot
```

```
plt.hist(s)
plt.xlabel("Value")
plt.ylabel("Frequency")

plt.show()
```



Let's try out some linear regression, or curve fitting.

```
import random

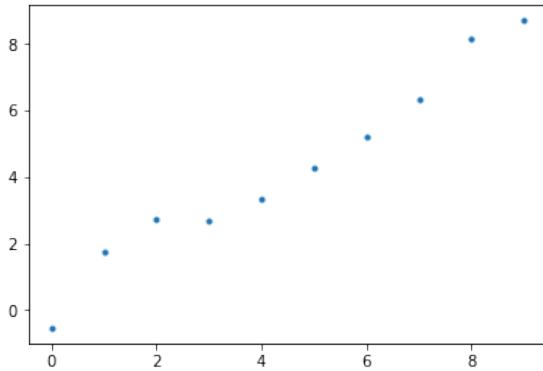
def F(x):
    return 2*x - 2

def add_noise(x):
    return x + random.uniform(-1,1)

X = range(0,10,1)

Y = []
for i in range(len(X)):
    Y.append(add_noise(X[i]))

plt.clf() # clear out the old figure
plt.plot(X,Y,'.')
plt.show()
```



Now let's try linear regression to fit the curve.

```
m, b, r, p, est_std_err = stats.linregress(X,Y)
```

What is the slope and y-intercept of the fitted curve?

```
print ('The slope is',m,'and the y-intercept is', b)
def Fprime(x): # the fitted curve
    return m*x + b
```

Now let's see how well the curve fits the data. We'll call the fitted curve F' .

```
X = range(0,10,1)

Yprime = []
for i in range(len(X)):
    Yprime.append(Fprime(X[i]))

plt.clf() # clear out the old figure

# the observed points, blue dots
plt.plot(X, Y, '.', label='observed points')

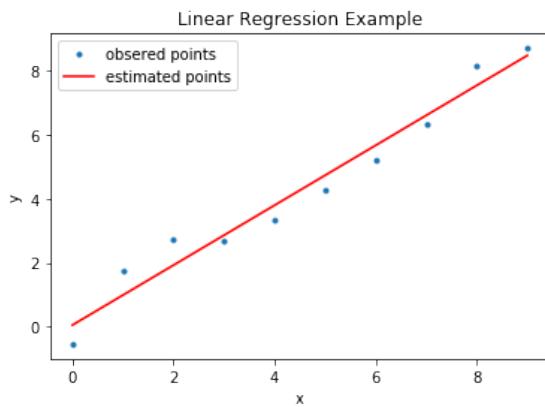
# the interpolated curve, connected red line
plt.plot(X, Yprime, 'r-', label='estimated points')

plt.title("Linear Regression Example") # title
plt.xlabel("x") # horizontal axis title
plt.ylabel("y") # vertical axis title
# legend labels to plot
plt.legend(['observed points', 'estimated points'])

# comment out so that you can save the figure
#plt.show()
```

To save images into a PDF file for inclusion into L^AT_EX documents you can save the images as follows. Other formats such as png are also possible, but the quality is naturally not sufficient for inclusion in papers and documents. For that you certainly want to use PDF. The save of the figure has to occur before you use the show() command.

```
plt.savefig("regression.pdf", bbox_inches='tight')
plt.savefig('regression.png')
plt.show()
```



26.2 References

For more information about SciPy we recommend that you visit the following link

<https://www.scipy.org/getting-started.html#learning-to-work-with-scipy>

Additional material and inspiration for this section are from

- “Getting Started guide” <https://www.scipy.org/getting-started.html>
- Prasanth. “Simple statistics with SciPy.” Comfort at 1 AU. February 28, 2011. <https://oneau.wordpress.com/2011/02/28/simple-statistics-with-scipy/>.
- SciPy Cookbook. Lasted updated: 2015. <http://scipy-cookbook.readthedocs.io/>.



27. OpenCV

OpenCV (Open Source Computer Vision Library) is a library of thousands of algorithms for various applications in computer vision and machine learning. It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. In this tutorial, we will explain basic features of this library, including the implementation of a simple example.

27.1 Overview

OpenCV has countless functions for image and videos processing. The pipeline starts with reading the images, low-level operations on pixel values, preprocessing e.g. denoising, and then multiple steps of higher-level operations which vary depending on the application. OpenCV covers the whole pipeline, especially providing a large set of library functions for high-level operations. A simpler library for image processing in Python is Scipy's multi-dimensional image processing package (`scipy.ndimage`).

27.2 Installation

OpenCV for Python can be installed on Linux in multiple ways, namely PyPI(Python Package Index), Linux package manager (`apt-get` for Ubuntu), Conda package manager, and also building from source. You are recommended to use PyPI. Here's the command that you need to run:

```
pip install opencv-python
```

This was tested on Ubuntu 16.04 with a fresh Python 3.6 virtual environment. In order to test, import the module in Python command line:

```
>>> import cv2
```

If it does not raise an error, it is installed correctly. Otherwise, try to solve the error.

For installation on Windows, see:

https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_setup/py_setup_in_windows/py_setup_in_windows.html#install-opencv-python-in-windows

Note that building from source can take a long time and may not be feasible for deploying to limited platforms such as Raspberry Pi.

27.3 A Simple Example

In this example, an image is loaded. A simple processing is performed, and the result is written to a new image.

27.3.1 Loading an image

```
%matplotlib inline
import cv2

img = cv2.imread('opencv_files/4.2.01.tiff')
```

The image was downloaded from USC standard database:

- <http://sipi.usc.edu/database/database.php?volume=misc&image=9>

27.3.2 Displaying the image

The image is saved in a numpy array. Each pixel is represented with 3 values (R,G,B). This provides you with access to manipulate the image at the level of single pixels. You can display the image using imshow function as well as Matplotlib's imshow function.

You can display the image using imshow function:

```
cv2.imshow('Original',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

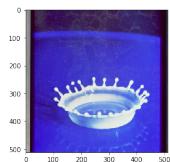
or you can use Matplotlib. If you have not installed Matplotlib before, install it using:

```
pip install matplotlib
```

Now you can use:

```
import matplotlib.pyplot as plt
plt.imshow(img)
```

which results in



27.3.3 Scaling and Rotation

Scaling (resizing) the image relative to different axis

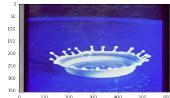
```
res = cv2.resize(img,
                 None,
                 fx=1.2,
```

```

        fy=0.7,
        interpolation=cv2.INTER_CUBIC)
plt.imshow(res)

```

which results in



Rotation of the image for an angle of t

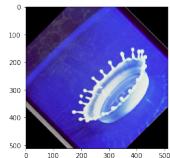
```

rows,cols,_ = img.shape
t = 45
M = cv2.getRotationMatrix2D((cols/2,rows/2),t,1)
dst = cv2.warpAffine(img,M,(cols,rows))

plt.imshow(dst)

```

which results in



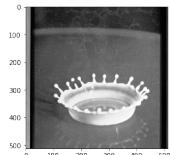
27.3.4 Gray-scaling

```

img2 = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(img2, cmap='gray')

```

which results in



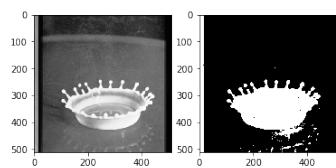
27.3.5 Image Thresholding

```

ret,thresh = cv2.threshold(img2,127,255,cv2.THRESH_BINARY)
plt.subplot(1,2,1), plt.imshow(img2, cmap='gray')
plt.subplot(1,2,2), plt.imshow(thresh, cmap='gray')

```

which results in



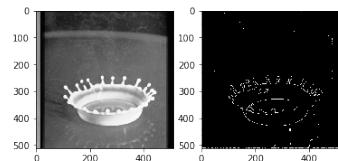
27.3.6 Edge Detection

Edge detection using Canny edge detection algorithm

```
edges = cv2.Canny(img2,100,200)

plt.subplot(121),plt.imshow(img2,cmap = 'gray')
plt.subplot(122),plt.imshow(edges,cmap = 'gray')
```

which results in



27.4 Additional Features

OpenCV has implementations of many machine learning techniques such as KMeans and Support Vector Machines, that can be put into use with only a few lines of code. It also has functions especially for video analysis, feature detection, object recognition and many more. You can find out more about them in their website

- <https://docs.opencv.org/3.0-beta/index.html>

OpenCV was initially developed for C++ and still has a focus on that language, but it is still one of the most valuable image processing libraries in Python.

27.5 Secchi Disk

27.5.1 Overview

More information about Secchi Disk can be found at:

- [https://en.wikipedia.org/wiki/Secchi/_disk](https://en.wikipedia.org/wiki/Secchi_disk)

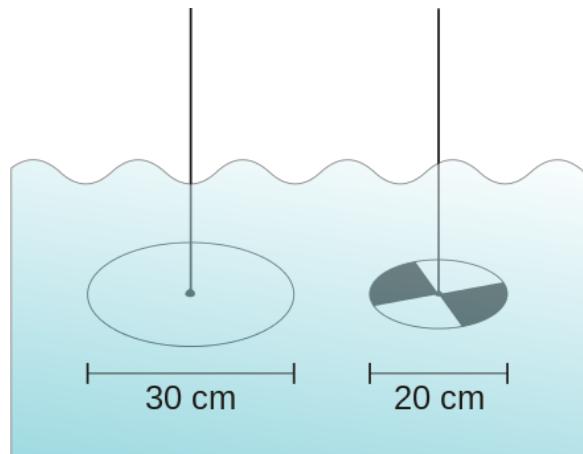


Figure 27.1: secchi

Figure: Different kinds of Secchi disks. A marine style on the left and the freshwater version on the right [wikipedia]

27.6 Setup for OSX

First let's setup the environment for OSX

```
import os, sys
from os.path import expanduser
os.path
home = expanduser("~")
sys.path.append('/usr/local/Cellar/opencv/3.3.1_1/lib/python3.6/site-packages/')
sys.path.append(home + '/.pyenv/versions/OPENCV/lib/python3.6/site-packages/')
import cv2
cv2.__version__
! pip install numpy > tmp.log
! pip install matplotlib >> tmp.log
%matplotlib inline
```

27.6.1 Step 1: Record the video

Record the video on the robot

27.6.2 Step 2: Analyse the images from the Video

For now we just selected 4 images from the video

```
import cv2
import matplotlib.pyplot as plt

img1 = cv2.imread('secchi/secchi1.png')
```

```
img2 = cv2.imread('secchi/secchi2.png')
img3 = cv2.imread('secchi/secchi3.png')
img4 = cv2.imread('secchi/secchi4.png')

figures = []
fig = plt.figure(figsize=(18, 16))
for i in range(1,13):
    figures.append(fig.add_subplot(4,3,i))
count = 0
for img in [img1,img2,img3,img4]:
    figures[count].imshow(img)

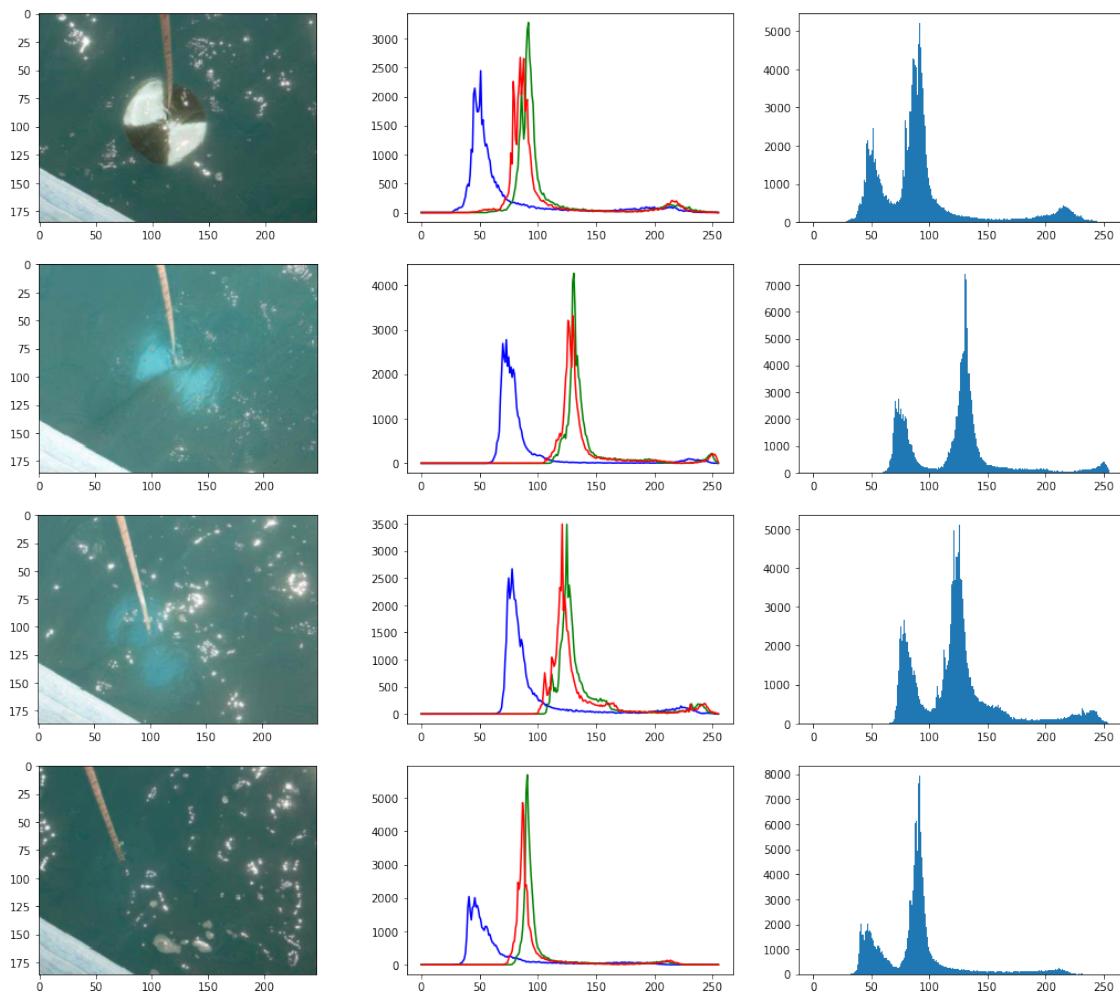
color = ('b','g','r')
for i,col in enumerate(color):
    histr = cv2.calcHist([img],[i],None,[256],[0,256])
    figures[count+1].plot(histr,color = col)

figures[count+2].hist(img.ravel(),256,[0,256])

count += 3

print("Legend")
print("First column = image of Secchi disk")
print("Second column = histogram of colors in image")
print("Third column = histogram of all values")

plt.show()
```

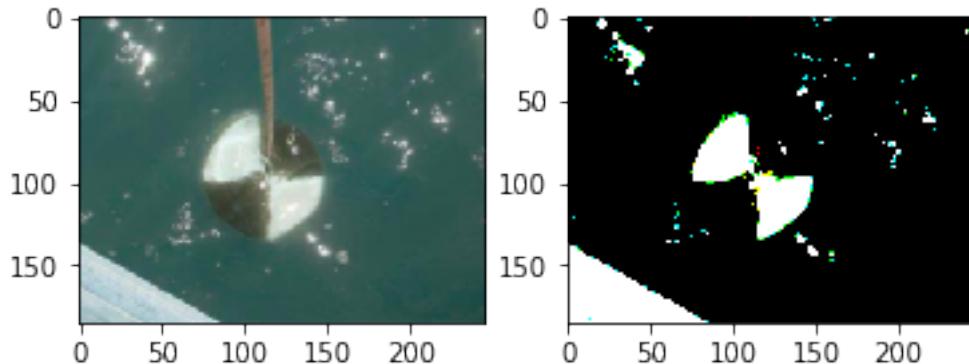


Rotation of the image for an angle of t

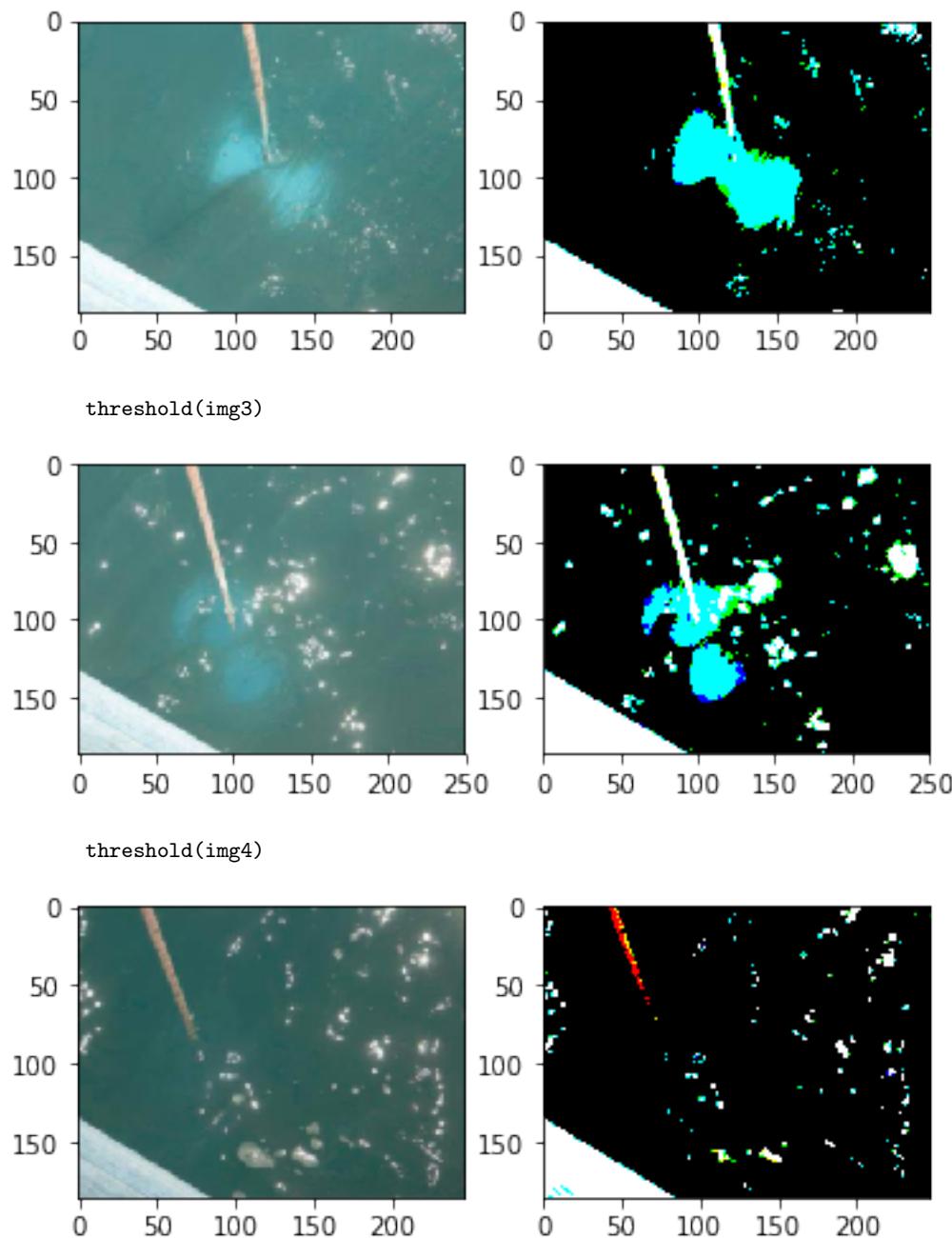
Image Thresholding

```
def threshold(img):
    ret,thresh = cv2.threshold(img,150,255,cv2.THRESH_BINARY)
    plt.subplot(1,2,1), plt.imshow(img, cmap='gray')
    plt.subplot(1,2,2), plt.imshow(thresh, cmap='gray')

threshold(img1)
```



```
threshold(img2)
```

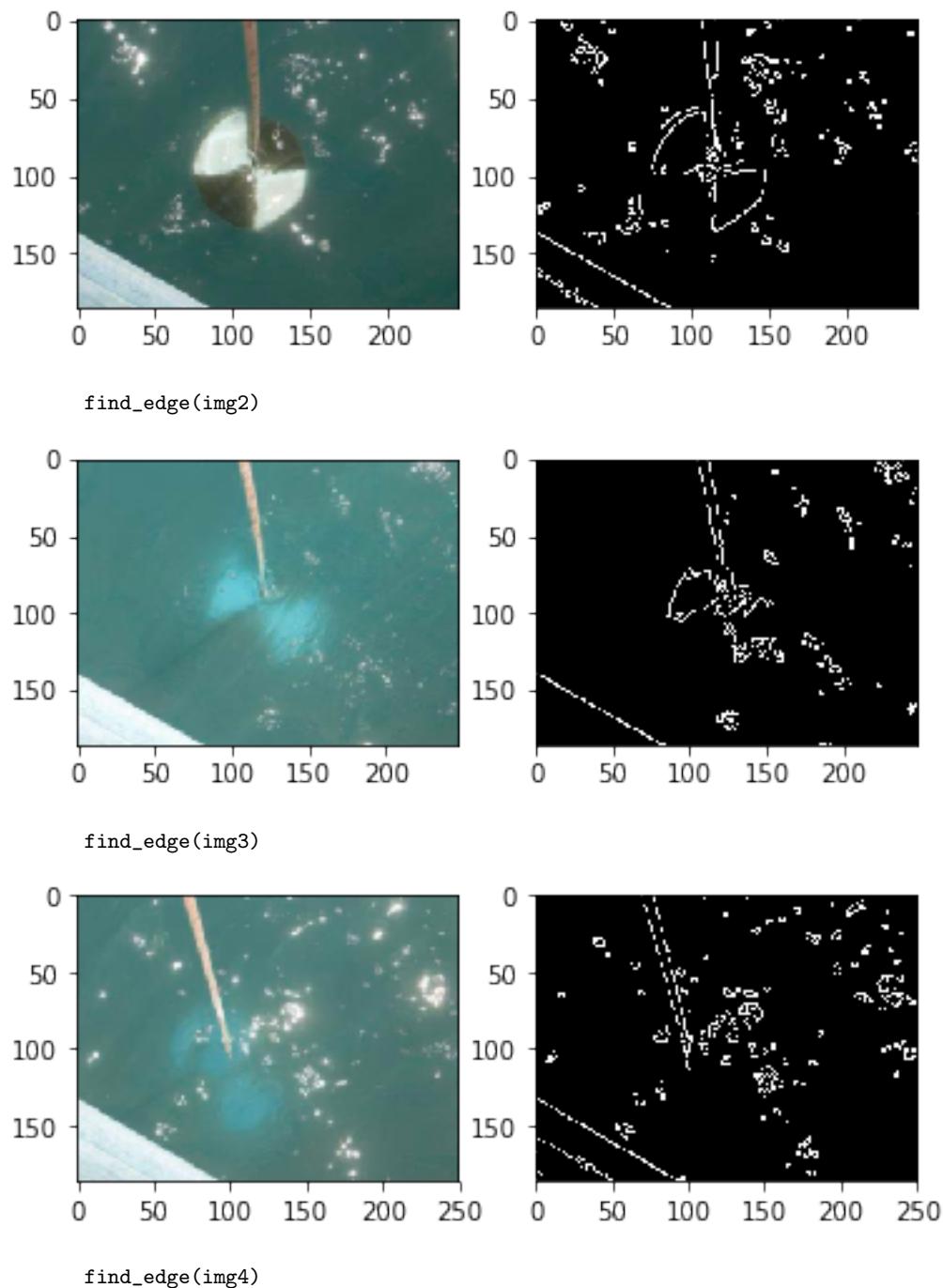


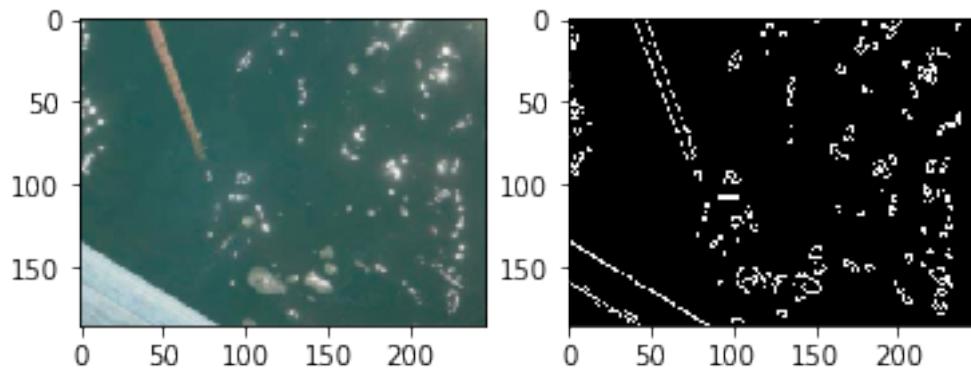
Edge Detection

Edge detection using Canny edge detection algorithm

```
def find_edge(img):
    edges = cv2.Canny(img,50,200)
    plt.subplot(121),plt.imshow(img,cmap = 'gray')
    plt.subplot(122),plt.imshow(edges,cmap = 'gray')

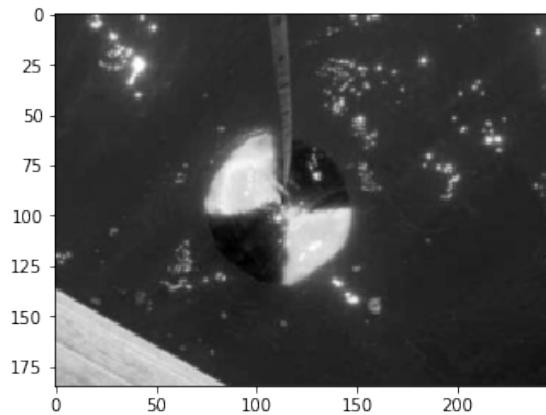
find_edge(img1)
```





27.7 Black and white

```
bw1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
plt.imshow(bw1, cmap='gray')
```





28. NIST Pedestrian and Face Detection

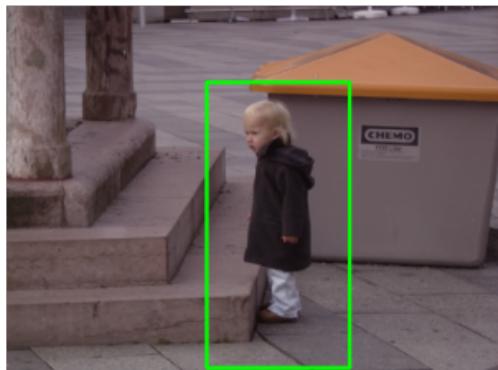
Pedestrian and Face Detection uses OpenCV to identify people standing in a picture or a video and NIST use case in this document is built with Apache Spark and Mesos clusters on multiple compute nodes.

The example in this tutorial deploys software packages on OpenStack using Ansible with its roles.

Original:



Pedestrian Detected:



Original



Pedestrian and Face/eyes Detected



28.0.1 Introduction

Human (pedestrian) detection and face detection have been studied during the last several years and models for them have improved along with Histograms of Oriented Gradients (HOG) for Human Detection [1]. OpenCV is a Computer Vision library including the SVM classifier and the HOG object detector for pedestrian detection and INRIA Person Dataset [2] is one of popular samples for both training and testing purposes. In this document, we deploy Apache Spark on Mesos clusters to train and apply detection models from OpenCV using Python API.

INRIA Person Dataset

This dataset contains positive and negative images for training and test purposes with annotation files for upright persons in each image. 288 positive test images, 453 negative test images, 614 positive training images and 1218 negative training images are included along with normalized 64x128 pixel formats. 970MB dataset is available to download [3].

HOG with SVM model

Histogram of Oriented Gradient (HOG) and Support Vector Machine (SVM) are used as object detectors and classifiers and built-in python libraries from OpenCV provide these models for human detection.

Ansible Automation Tool

Ansible is a python tool to install/configure/manage software on multiple machines with JSON files where system descriptions are defined. There are reasons why we use Ansible:

- Expandable: Leverages Python (default) but modules can be written in any language
- Agentless: no setup required on managed node
- Security: Allows deployment from user space; uses ssh for authentication
- Flexibility: only requires ssh access to privileged user
- Transparency: YAML Based script files express the steps of installing and configuring software
- Modularity: Single Ansible Role (should) contain all required commands and variables to deploy software package independently
- Sharing and portability: roles are available from source (github, bitbucket, gitlab, etc) or the Ansible Galaxy portal

We use Ansible roles to install software packages for Human and Face Detection which requires to run OpenCV Python libraries on Apache Mesos with a cluster configuration. Dataset is also downloaded from the web using an ansible role.

28.0.2 Deployment by Ansible

Ansible is to deploy applications and build clusters for batch-processing large datasets towards target machines e.g. VM instances on OpenStack and we use ansible roles with *include* directive to organize layers of big data software stacks (BDSS). Ansible provides abstractions by Playbook Roles and reusability by Include statements. We define X application in X Ansible Role, for example, and use include statements to combine with other applications e.g. Y or Z. The layers exist in sub directories (see below) to add modularity to your Ansible deployment. For example, there are five roles used in this example that are Apache Mesos in a scheduler layer, Apache Spark in a processing layer, a OpenCV library in an application layer, INRIA Person Dataset in a dataset layer and a python script for human and face detection in an analytics layer. If you have an additional software package to add, you can simply add a new role in a main ansible playbook with *include* directive. With this, your Ansible playbook maintains simple but flexible to add more roles without having a large single file which is getting difficult to read when it deploys more applications on multiple layers. The main Ansible playbook runs Ansible roles in order which look like:

```

1 -----
2 include : sched/00-mesos .yml
3 include : proc/01-spark .yml
4 include : apps/02-opencv .yml

```

```

5 include: data/03-inria-dataset.yml
6 Include: anlys/04-human-face-detection.yml

```

Directory names e.g. sched, proc, data, or anlys indicate BDSS layers like: - sched: scheduler layer - proc: data processing layer - apps: application layer - data: dataset layer - anlys: analytics layer and two digits in the filename indicate an order of roles to be run.

28.0.3 Cloudmesh for Provisioning

It is assumed that virtual machines are created by cloudmesh, the cloud management software. For example on OpenStack,

```
cm cluster create -N=6
```

command starts a set of virtual machine instances. The number of machines and groups for clusters e.g. namenodes and datanodes are defined in the Ansible inventory file, a list of target machines with groups, which will be generated once machines are ready to use by cloudmesh. Ansible roles install software and dataset on virtual clusters after that stage.

28.0.4 Roles Explained for Installation

Mesos role is installed first as a scheduler layer for masters and slaves where mesos-master runs on the masters group and mesos-slave runs on the slaves group. Apache Zookeeper is included in the mesos role therefore mesos slaves find an elected mesos leader for the coordination. Spark, as a data processing layer, provides two options for distributed job processing, batch job processing via a cluster mode and real-time processing via a client mode. The Mesos dispatcher runs on a masters group to accept a batch job submission and Spark interactive shell, which is the client mode, provides real-time processing on any node in the cluster. Either way, Spark is installed later to detect a master (leader) host for a job submission. Other roles for OpenCV, INRIA Person Dataset and Human and Face Detection Python applications are followed by.

The following software are expected in the stacks according to the [github](#):

- mesos cluster (master, worker)
- spark (with dispatcher for mesos cluster mode)
- openCV
- zookeeper
- INRIA Person Dataset
- Detection Analytics in Python
- [1] Dalal, Navneet, and Bill Triggs. “Histograms of oriented gradients for human detection.” 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05). Vol. 1. IEEE, 2005. [pdf]
- [2] <http://pascal.inrialpes.fr/data/human/>
- [3] <ftp://ftp.inrialpes.fr/pub/lear/douze/data/INRIAPerson.tar>
- [4] <https://docs.python.org/2/library/configparser.html>

Server groups for Masters/Slaves by Ansible inventory

We may separate compute nodes in two groups: masters and workers therefore Mesos masters and zookeeper quorums manage job requests and leaders and workers run actual tasks. Ansible needs group definitions in their inventory therefore software installation associated with a proper part can

be completed.

Example of Ansible Inventory file (inventory.txt)

```

1 [ masters ]
2 10.0.5.67
3 10.0.5.68
4 10.0.5.69
5 [ slaves ]
6 10.0.5.70
7 10.0.5.71
8 10.0.5.72

```

28.0.5 Instructions for Deployment

The following commands complete NIST Pedestrian and Face Detection deployment on OpenStack.

Cloning Pedestrian Detection Repository from Github

Roles are included as submodules which require --recursive option to checkout them all.

```

1 !git clone --recursive https://github.com/futuresystems/pedestrian-and-face
      -detection.git

```

Change the following variable with actual ip addresses:

```

1 sample_inventory="""
2 10.0.5.67
3 10.0.5.68
4 10.0.5.69
5 [ slaves ]
6 10.0.5.70
7 10.0.5.71
8 10.0.5.72"""

```

Create a inventory.txt file with the variable in your local directory.

```

1 !printf "$sample_inventory" > inventory.txt
2 !cat inventory.txt

```

Add ansible.cfg file with options for ssh host key checking and login name.

```

1 ansible_config="""
2 host_key_checking=false
3 remote_user=ubuntu"""
4 !printf "$ansible_config" > ansible.cfg
5 !cat ansible.cfg

```

Check accessibility by ansible ping like:

```

1 !ansible -m ping -i inventory.txt all

```

Make sure that you have a correct ssh key in your account otherwise you may encounter ‘FAILURE’ in the ping test above.

Ansible Playbook

We use a main ansible playbook to deploy software packages for NIST Pedestrian and Face detection which includes: - mesos - spark -zookeeper - opencv - INRIA Person dataset - Python script for the detection

```
1 !cd pedestrian-and-face-detection/ && ansible-playbook -i ./inventory.txt site.yml
```

The installation may take 30 minutes or an hour to complete.

28.0.6 OpenCV in Python

Before we run our code for this project, let's try OpenCV first to see how it works.

Import cv2

Let's import opencv python module and we will use images from the online database image-net.org to test OpenCV image recognition.

```
1 import cv2
```

Let's download a mailbox image with a red color to see if opencv identifies the shape with a color. The example file in this tutorial is:

```
!curl http://farm4.static.flickr.com/3061/2739199963_ee78af76ef.jpg > mailbox.jpg
```

```
1 %matplotlib inline
```

```
1 from IPython.display import Image  
2 mailbox_image = "mailbox.jpg"  
3 Image(filename=mailbox_image)
```



You can try other images. Check out the image-net.org for mailbox images: <http://image-net.org/synset?wnid=n03710193>

Image Detection

Just for a test, let's try to detect a red color shaped mailbox using opencv python functions.

There are key functions that we use:
* cvtColor: to convert a color space of an image
* inRange: to detect a mailbox based on the range of red color pixel values
* np.array: to define the range of red color using a Numpy library for better calculation
* findContours: to find a outline of the object
* bitwise_and: to black-out the area of contours found

```
1 import numpy as np  
2 import matplotlib.pyplot as plt  
3
```

```

4 # imread for loading an image
5 img = cv2.imread(mailbox_image)
6 # cvtColor for color conversion
7 hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
8
9 # define range of red color in hsv
10 lower_red1 = np.array([0, 50, 50])
11 upper_red1 = np.array([10, 255, 255])
12 lower_red2 = np.array([170, 50, 50])
13 upper_red2 = np.array([180, 255, 255])
14
15 # threshold the hsv image to get only red colors
16 mask1 = cv2.inRange(hsv, lower_red1, upper_red1)
17 mask2 = cv2.inRange(hsv, lower_red2, upper_red2)
18 mask = mask1 + mask2
19
20 # find a red color mailbox from the image
21 im2, contours, hierarchy = cv2.findContours(mask, cv2.RETR_TREE, cv2.
22     CHAIN_APPROX_SIMPLE)
23
24 # bitwise_and to remove other areas in the image except the detected object
25 res = cv2.bitwise_and(img, img, mask = mask)
26
27 # turn off - x, y axis bar
28 plt.axis("off")
29 # text for the masked image
30 cv2.putText(res, "masked image", (20,300), cv2.FONT_HERSHEY_SIMPLEX, 2,
31             (255,255,255))
32 # display
33 plt.imshow(cv2.cvtColor(res, cv2.COLOR_BGR2RGB))
34 plt.show()

```



The red color mailbox is left alone in the image which we wanted to find in this example by opencv functions. You can try other images with different colors to detect the different shape of objects using findContours and inRange from opencv.

For more information, see the useful links below.

- contours features: http://docs.opencv.org/3.1.0/dd/d49/tutorial/_py/_contour_features.html
- contours: http://docs.opencv.org/3.1.0/d4/d73/tutorial/_py/_contours/_begin.html
- red color in hsv: <http://stackoverflow.com/questions/30331944/finding-red-color-using-python-opencv>
- inrange: http://docs.opencv.org/master/da/d97/tutorial/_threshold/_inRange.

- html
- inrange: http://docs.opencv.org/3.0-beta/doc/py/_tutorials/py/_imgproc/py_colorspaces/py_colorspaces.html
- numpy: http://docs.opencv.org/3.0-beta/doc/py/_tutorials/py/_core/py_basic_ops/py_basic_ops.html

28.0.7 Human and Face Detection in OpenCV

INRIA Person Dataset

We use INRIA Person dataset to detect upright people and faces in images in this example. Let's download it first.

```
1 !curl ftp://ftp.inrialpes.fr/pub/lear/douze/data/INRIAPerson.tar >
    INRIAPerson.tar
```

```
100 969M 100 969M 0 0 8480k 0 0:01:57 0:01:57 -:-:- 12.4M
```

```
1 !tar xvf INRIAPerson.tar > logfile && tail logfile
```

Face Detection using Haar Cascades

This section is prepared based on the opencv-python tutorial: http://docs.opencv.org/3.1.0/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0

There is a pre-trained classifier for face detection, download it from here:

```
1 !curl https://raw.githubusercontent.com/opencv/opencv/master/data/
    haarcascades/haarcascade_frontalface_default.xml >
    haarcascade_frontalface_default.xml
```

```
100 908k 100 908k 0 0 2225k 0 -:-:-:-:-:- 2259k
```

This classifier XML file will be used to detect faces in images. If you like to create a new classifier, find out more information about training from here: http://docs.opencv.org/3.1.0/dc/d88/tutorial_traincascade.html

Face Detection Python Code Snippet

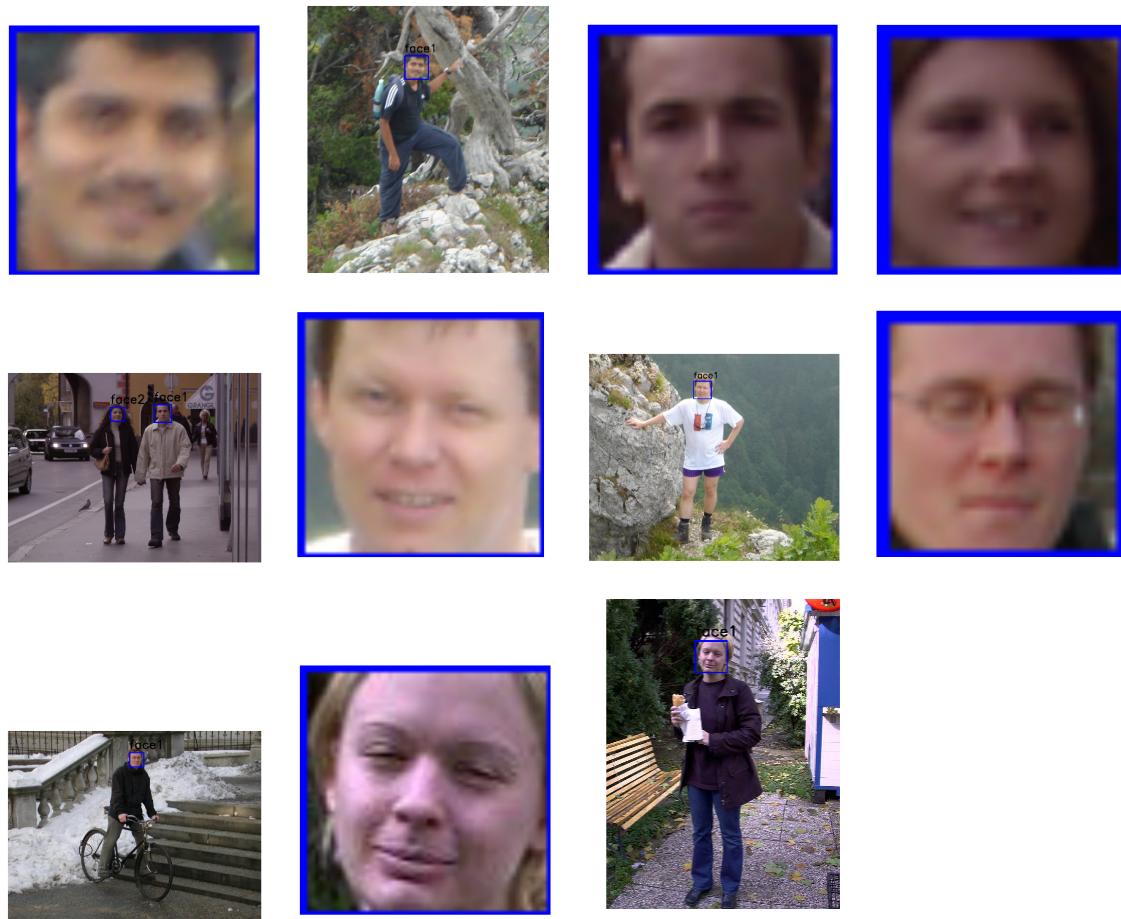
Now, we detect faces from the first five images using the classifier.

```
1 # import the necessary packages
2 from __future__ import print_function
3 import numpy as np
4 import cv2
5 from os import listdir
6 from os.path import isfile, join
7 import matplotlib.pyplot as plt
8
9 mypath = "INRIAPerson/Test/pos/"
10 face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
11
12 onlyfiles = [join(mypath, f) for f in listdir(mypath) if isfile(join(mypath,
13 , f))]
14 cnt = 0
15 for filename in onlyfiles:
16     image = cv2.imread(filename)
17     image_grayscale = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

```

18 faces = face_cascade.detectMultiScale(image_grayscale, 1.3, 5)
19 if len(faces) == 0:
20     continue
21
22 cnt_faces = 1
23 for (x,y,w,h) in faces:
24     cv2.rectangle(image,(x,y),(x+w,y+h),(255,0,0),2)
25     cv2.putText(image, "face" + str(cnt_faces), (x,y-10), cv2.
26 FONT_HERSHEY_SIMPLEX, 1, (0,0,0), 2)
27 plt.figure()
28 plt.axis("off")
29 plt.imshow(cv2.cvtColor(image[y:y+h, x:x+w], cv2.COLOR_BGR2RGB))
30 cnt_faces += 1
31 plt.figure()
32 plt.axis("off")
33 plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
34 cnt = cnt + 1
35 if cnt == 5:
36     break

```



28.0.8 Pedestrian Detection using HOG Descriptor

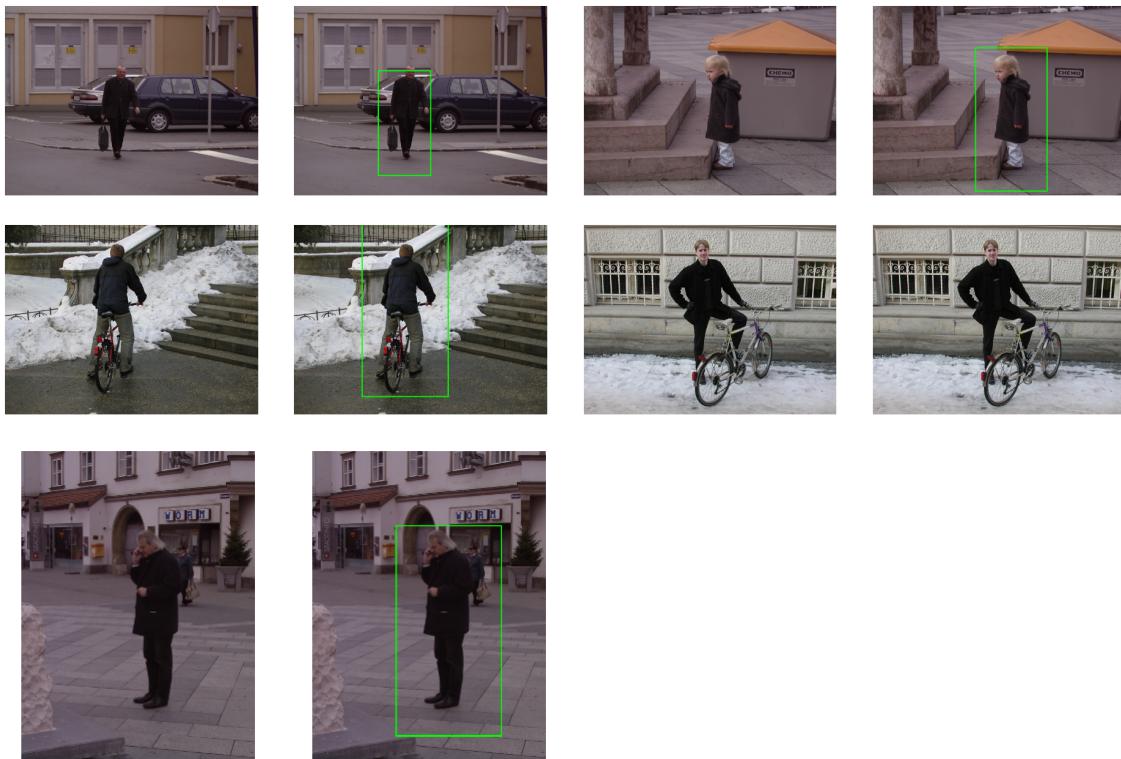
We will use Histogram of Oriented Gradients (HOG) to detect a upright person from images.

Python Code Snippet

```

1 # initialize the HOG descriptor/person detector
2 hog = cv2.HOGDescriptor()
3 hog.setSVMClassifier(cv2.HOGDescriptor_getDefaultPeopleDetector())
4
5 cnt = 0
6 for filename in onlyfiles:
7     img = cv2.imread(filename)
8     orig = img.copy()
9     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
10
11     # detect people in the image
12     (rects, weights) = hog.detectMultiScale(img, winStride=(8, 8),
13     padding=(16, 16), scale=1.05)
14
15     # draw the final bounding boxes
16     for (x, y, w, h) in rects:
17         cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
18
19     plt.figure()
20     plt.axis("off")
21     plt.imshow(cv2.cvtColor(orig, cv2.COLOR_BGR2RGB))
22     plt.figure()
23     plt.axis("off")
24     plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
25     cnt = cnt + 1
26     if cnt == 5:
27         break

```



28.0.9 Processing by Apache Spark

INRIA Person dataset provides 100+ images and Spark can be used for image processing in parallel. We load 288 images from “Test/pos” directory.

Spark provides a special object ‘sc’ to connect between a spark cluster and functions in python code. Therefore, we can run python functions in parallel to detect objects in this example.

- *map* function is used to process pedestrian and face detection per image from the `parallelize()` function of ‘sc’ spark context.
- *collect* function merges results in an array.

```

1 def apply_batch(imagePath):
2     import cv2
3     import numpy as np
4     # initialize the HOG descriptor/person detector
5     hog = cv2.HOGDescriptor()
6     hog.setSVMClassifier(cv2.HOGDescriptor_getDefaultPeopleDetector())
7     image = cv2.imread(imagePath)
8     # detect people in the image
9     (rects, weights) = hog.detectMultiScale(image, winStride=(8, 8),
10                                             padding=(16, 16), scale=1.05)
11     # draw the final bounding boxes
12     for (x, y, w, h) in rects:
13         cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
14
15 return image

```

Parallelize in Spark Context

The list of image files is given to parallelize.

```
1 pd = sc.parallelize(onlyfiles)
```

Map Function (apply_batch)

The ‘`apply_batch`’ function that we created above is given to map function to process in a spark cluster.

```
1 pdc = pd.map(apply_batch)
```

Collect Function

The result of each map process is merged to an array.

```
1 result = pdc.collect()
```

28.0.10 Results for 100+ images by Spark Cluster

```

1 for image in result:
2     plt.figure()
3     plt.axis("off")
4     plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

```


29. Python Fingerprint Example

Python is a flexible and popular language for running data analysis pipelines. In this tutorial we will implement a solution for a fingerprint matching.

29.1 Overview

Fingerprint recognition refers to the automated method for verifying a match between two fingerprints and that is used to identify individuals and verify their identity. Fingerprints (Figure 29.1) are the most widely used form of biometric used to identify individuals.

TODO: image missing

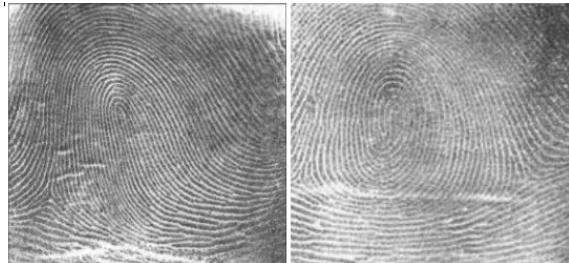


Figure 29.1: Fingerprints

The automated fingerprint matching generally required the detection of different fingerprint features (aggregate characteristics of ridges, and minutia points) and then the use of fingerprint matching algorithm, which can do both one-to- one and one-to- many matching operations. Based on the number of matches a proximity score (distance or similarity) can be calculated.

We use the following NIST dataset for the study:

Special Database 14 - NIST Mated Fingerprint Card Pairs 2. (<http://www.nist.gov/itl/iad/>

```
ig/special/_dbases.cfm)
```

TODO: citation

29.2 Objectives

Match the fingerprint images from a probe set to a gallery set and report the match scores.

29.3 Prerequisites

For this work we will use the following algorithms:

- MINDTCT: The NIST minutiae detector, which automatically locates and records ridge ending and bifurcations in a fingerprint image. (<http://www.nist.gov/itl/iad/ig/nbis.cfm>)
- BOZORTH3: A NIST fingerprint matching algorithm, which is a minutiae based fingerprint-matching algorithm. It can do both one-to-one and one-to-many matching operations. (<http://www.nist.gov/itl/iad/ig/nbis.cfm>)

In order to follow along, you must have the NBIS tools which provide `mindtct` and `bozorth3` installed. If you are on Ubuntu 16.04 Xenial, the following steps will accomplish this:

```
1 $ sudo apt-get update -qq
2 $ sudo apt-get install -y build-essential cmake unzip
3 $ wget "http://nigos.nist.gov:8080/nist/nbis/nbis_v5_0_0.zip"
4 $ unzip -d nbis nbis_v5_0_0.zip
5 $ cd nbis/Rel_5.0.0
6 $ ./setup.sh /usr/local --without-X11
7 $ sudo make
```

29.4 Implementation

1. Fetch the fingerprint images from the web
2. Call out to external programs to prepare and compute the match scores
3. Store the results in a database
4. Generate a plot to identify likely matches.

```
1 from __future__ import print_function
2 import urllib
3 import zipfile
4 import hashlib
```

We'll be interacting with the operating system and manipulating files and their pathnames.

```
1 import os.path
2 import os
3 import sys
4 import shutil
5 import tempfile
```

Some general useful utilities

```
1 import itertools
2 import functools
```

```

3 import types
4 from pprint import pprint

```

Using the `attrs` library provides some nice shortcuts to defining objects

```

1 import attr
1 import sys

```

We'll be randomly dividing the entire dataset, based on user input, into the probe and gallery sets

```

1 import random

```

We'll need to call out to the NBIS software. We'll also be using multiple processes to take advantage of all the cores on our machine

```

1 import subprocess
2 import multiprocessing

```

As for plotting, we'll use `matplotlib`, though there are many alternatives.

```

1 import matplotlib.pyplot as plt
2 import pandas as pd
3 import numpy as np

```

Finally, we'll write the results to a database.

```

1 import sqlite3

```

29.5 Utility functions

Next, we'll define some utility functions:

```

1 def take(n, iterable):
2     "Returns a generator of the first ***n*** elements of an iterable"
3     return itertools.islice(iterable, n)
4
5
6 def zipWith(function, *iterables):
7     "Zip a set of **iterables** together and apply **function** to each
8     tuple"
9     for group in itertools.izip(*iterables):
10        yield function(*group)
11
12 def uncurry(function):
13     "Transforms an N-arry **function** so that it accepts a single
14     parameter of an N-tuple"
15     @functools.wraps(function)
16     def wrapper(args):
17         return function(*args)
18     return wrapper
19
20 def fetch_url(url, sha256, prefix='.', checksum_blocksize=2**20, dryRun=False):
21     """Download a url.
22
23     :param url: the url to the file on the web

```

```

24     :param sha256: the SHA-256 checksum. Used to determine if the file was
25     previously downloaded.
26     :param prefix: directory to save the file
27     :param checksum_blocksize: blocksize to used when computing the
28     checksum
29     :param dryRun: boolean indicating that calling this function should do
30     nothing
31     :returns: the local path to the downloaded file
32     :rtype:
33
34     """
35
36     if not os.path.exists(prefix):
37         os.makedirs(prefix)
38
39     local = os.path.join(prefix, os.path.basename(url))
40
41     if dryRun: return local
42
43     if os.path.exists(local):
44         print ('Verifying checksum')
45         chk = hashlib.sha256()
46         with open(local, 'rb') as fd:
47             while True:
48                 bits = fd.read(checksum_blocksize)
49                 if not bits: break
50                 chk.update(bits)
51             if sha256 == chk.hexdigest():
52                 return local
53
54     print ('Downloading', url)
55
56     def report(sofar, blocksize, totalsize):
57         msg = '{}%\r'.format(100 * sofar * blocksize / totalsize, 100)
58         sys.stderr.write(msg)
59
60     urllib.urlretrieve(url, local, report)
61
62     return local

```

29.6 Dataset

We'll now define some global parameters

First, the fingerprint dataset

```

1 DATASET_URL = 'https://s3.amazonaws.com/nist-srd/SD4/
2   NISTSpecialDatabase4GrayScaleImageofFIGS.zip'
3 DATASET_SHA256 = '4
4   db6a8f3f9dc14c504180cbf67cdf35167a109280f121c901be37a80ac13c449'

```

We'll define how to download the dataset. This function is general enough that it could be used to retrieve most files, but we'll default it to use the values from above.

```

1 def prepare_dataset(url=None, sha256=None, prefix='.', skip=False):
2     url = url or DATASET_URL
3     sha256 = sha256 or DATASET_SHA256
4     local = fetch_url(url, sha256=sha256, prefix=prefix, dryRun=skip)
5

```

```

6     if not skip:
7         print ('Extracting', local, 'to', prefix)
8         with zipfile.ZipFile(local, 'r') as zip:
9             zip.extractall(prefix)
10
11    name, _ = os.path.splitext(local)
12    return name
13
14
15 def locate_paths(path_md5list, prefix):
16     with open(path_md5list) as fd:
17         for line in itertools imap(str.strip, fd):
18             parts = line.split()
19             if not len(parts) == 2: continue
20             md5sum, path = parts
21             checksum = Checksum(value=md5sum, kind='md5')
22             filepath = os.path.join(prefix, path)
23             yield Path(checksum=checksum, filepath=filepath)
24
25
26 def locate_images(paths):
27
28     def predicate(path):
29         _, ext = os.path.splitext(path.filepath)
30         return ext in ['.png']
31
32     for path in itertools ifilter(predicate, paths):
33         yield image(id=path.checksum.value, path=path)

```

29.7 Data Model

We'll define some classes so we have a nice API for working with the dataflow. We set `slots=True` so that the resulting objects will be more space-efficient.

29.7.1 Utilities

29.7.2 Checksum

The checksum consists of the actual hash value (`value`) as well as a string representing the hashing algorithm. The validator enforces that the algorithm can only be one of the listed acceptable methods

```

1 @attr.s(slots=True)
2 class Checksum(object):
3     value = attr.ib()
4     kind = attr.ib(validator=lambda o, a, v: v in 'md5 sha1 sha224 sha256
      sha384 sha512'.split())

```

29.7.3 Path

Paths refer to an image's filepath and associated `Checksum`. We get the checksum "for"free" since the MD5 hash is provided for each image in the dataset.

```

1 @attr.s(slots=True)
2 class Path(object):
3     checksum = attr.ib()
4     filepath = attr.ib()

```

29.7.4 Image

The start of the data pipeline is the image. An `image` has an `id` (the md5 hash) and the path to the image.

```
1 @attr.s(slots=True)
2 class image(object):
3     id = attr.ib()
4     path = attr.ib()
```

29.7.5 Mindtct

The next step in the pipeline is to apply the `mindtct` program from NBIS. A `mindtct` object therefore represents the results of applying `mindtct` on an `image`. The `xyt` output is needed for the next step, and the `image` attribute represents the image id.

```
1 @attr.s(slots=True)
2 class mindtct(object):
3     image = attr.ib()
4     xyt = attr.ib()
5
6     def pretty(self):
7         d = dict(id=self.image.id, path=self.image.path)
8         return pprint(d)
```

We need a way to construct a `mindtct` object from an `image` object. A straightforward way of doing this would be to have a `from_image` `@staticmethod` or `@classmethod`, but that doesn't work well with `multiprocessing` as top-level functions work best as they need to be serialized.

```
1 def mindtct_from_image(image):
2     imgpath = os.path.abspath(image.path_filepath)
3     tempdir = tempfile.mkdtemp()
4     oroot = os.path.join(tempdir, 'result')
5
6     cmd = ['mindtct', imgpath, oroot]
7
8     try:
9         subprocess.check_call(cmd)
10
11     with open(rooot + '.xyt') as fd:
12         xyt = fd.read()
13
14     result = mindtct(image=image.id, xyt=xyt)
15     return result
16
17 finally:
18     shutil.rmtree(tempdir)
```

29.7.6 Bozorth3

The final step in the pipeline is running the `bozorth3` from NBIS. The `bozorth3` class represents the match being done: tracking the ids of the probe and gallery images as well as the match score.

Since we'll be writing these instance out to a database, we provide some static methods for SQL statements. While there are many Object-Relational-Model (ORM) libraries available for Python, this approach keeps the current implementation simple.

```
1 @attr.s(slots=True)
```

```

2 class bozorth3(object):
3     probe = attr.ib()
4     gallery = attr.ib()
5     score = attr.ib()
6
7     @staticmethod
8     def sql_stmt_create_table():
9         return 'CREATE TABLE IF NOT EXISTS bozorth3 ' \
10            + '(probe TEXT, gallery TEXT, score NUMERIC)'
11
12     @staticmethod
13     def sql_prepared_stmt_insert():
14         return 'INSERT INTO bozorth3 VALUES (?, ?, ?)'
15
16     def sql_prepared_stmt_insert_values(self):
17         return self.probe, self.gallery, self.score

```

In order to work well with `multiprocessing`, we define a class representing the input parameters to `bozorth3` and a helper function to run `bozorth3`. This way the pipeline definition can be kept simple to a `map` to create the input and then a `map` to run the program.

As NBIS `bozorth3` can be called to compare one-to-one or one-to-many, we'll also dynamically choose between these approaches depending on if the `gallery` attribute is a list or a single object.

```

1 @attr.s(slots=True)
2 class bozorth3_input(object):
3     probe = attr.ib()
4     gallery = attr.ib()
5
6     def run(self):
7         if isinstance(self.gallery, mindtct):
8             return bozorth3_from_one_to_one(self.probe, self.gallery)
9         elif isinstance(self.gallery, types.ListType):
10             return bozorth3_from_one_to_many(self.probe, self.gallery)
11         else:
12             raise ValueError('Unhandled type for gallery: {}'.format(type(
13                 self.gallery)))

```

The next is the top-level function to running `bozorth3`. It accepts an instance of `bozorth3_input`. This is implemented as a simple top-level wrapper so that it can be easily passed to the `multiprocessing` library.

```

1 def run_bozorth3(input):
2     return input.run()

```

29.7.7 Running Bozorth3

There are two cases to handle: 1. One-to-one probe to gallery sets 1. One-to-many probe to gallery sets

Both approaches are implemented below. The implementations follow the same pattern: 1. Create a temporary directory within with to work 1. Write the probe and gallery images to files in the temporary directory 1. Call the `bozorth3` executable 1. The match score is written to `stdout` which is captured and then parsed. 1. Return a `bozorth3` instance for each match 1. Make sure to clean up the temporary directory

One-to-one

```

1 def bozorth3_from_one_to_one(probe, gallery):
2     tempdir = tempfile.mkdtemp()
3     probeFile = os.path.join(tempdir, 'probe.xyt')
4     galleryFile = os.path.join(tempdir, 'gallery.xyt')
5
6     with open(probeFile, 'wb') as fd: fd.write(probe.xyt)
7     with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)
8
9     cmd = ['bozorth3', probeFile, galleryFile]
10
11    try:
12        result = subprocess.check_output(cmd)
13        score = int(result.strip())
14        return bozorth3(probe=probe.image, gallery=gallery.image, score=
15                      score)
16    finally:
17        shutil.rmtree(tempdir)

```

One-to-many

```

1 def bozorth3_from_one_to_many(probe, galleryset):
2     tempdir = tempfile.mkdtemp()
3     probeFile = os.path.join(tempdir, 'probe.xyt')
4     galleryFiles = [os.path.join(tempdir, 'gallery%d.xyt' % i)
5                      for i, _ in enumerate(galleryset)]
6
7     with open(probeFile, 'wb') as fd: fd.write(probe.xyt)
8     for galleryFile, gallery in itertools.izip(galleryFiles, galleryset):
9         with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)
10
11    cmd = ['bozorth3', '-p', probeFile] + galleryFiles
12
13    try:
14        result = subprocess.check_output(cmd).strip()
15        scores = map(int, result.split('\n'))
16        return [bozorth3(probe=probe.image, gallery=gallery.image, score=
17                         score)
18                for score, gallery in zip(scores, galleryset)]
19    finally:
20        shutil.rmtree(tempdir)

```

29.8 Plotting

For plotting we'll operate only on the database. We'll select a small number of probe images and plot the score between them and the rest of the gallery images.

The `mk_short_labels` helper function will be defined below.

```

1 def plot(dbfile, nprobes=10):
2     conn = sqlite3.connect(dbfile)
3     results = pd.read_sql(
4         "SELECT DISTINCT probe FROM bozorth3 ORDER BY score LIMIT '%s'" %
5         nprobes,
6         con=conn
7     )
8     shortlabels = mk_short_labels(results.probe)
9     plt.figure()
10
11    for i, probe in results.probe.iteritems():
12        stmt = 'SELECT gallery, score FROM bozorth3 WHERE probe = ? ORDER
13        BY gallery DESC'

```

```

12     matches = pd.read_sql(stmt, params=(probe,), con=conn)
13     xs = np.arange(len(matches), dtype=np.int)
14     plt.plot(xs, matches.score, label='probe %s' % shortlabels[i])
15
16     plt.ylabel('Score')
17     plt.xlabel('Gallery')
18     plt.legend(bbox_to_anchor=(0, 0, 1, -0.2))
19     plt.show()

```

The image ids are long hash strings. In order to minimize the amount of space on the figure the labels occupy, we provide a helper function to create a short label that still uniquely identifies each probe image in the selected sample

```

1 def mk_short_labels(series, start=7):
2     for size in xrange(start, len(series[0])):
3         if len(series) == len(set(map(lambda s: s[:size], series))):
4             break
5     return map(lambda s: s[:size], series)

```

29.9 Putting it all Together

First, set up a temporary directory in which to work:

```

1 pool = multiprocessing.Pool()
2 prefix = '/tmp/fingerprint_example/'
3 if not os.path.exists(prefix):
4     os.makedirs(prefix)

```

Next we download and extract the fingerprint images from NIST:

```
1 dataprefix = prepare_dataset(prefix=prefix)
```

Next we'll configure the location of the MD5 checksum file that comes with the download

```
1 md5listpath = os.path.join(prefix, 'NISTSpecialDatabase4GrayScaleImagesofFIGS/sd04/sd04_md5.lst')
```

Load the images from the downloaded files to start the analysis pipeline

```

1 print('Loading images')
2 paths = locate_paths(md5listpath, dataprefix)
3 images = locate_images(paths)
4 mindtcts = pool.map(mindtct_from_image, images)
5 print('Done')

```

We can examine one of the loaded image. Note that `image` is refers to the MD5 checksum that came with the image and the `xyt` attribute represents the raw image data.

```

1 print(mindtcts[0].image)
2 print(mindtcts[0].xyt[:50])

```

For example purposes we'll only use a small percentage of the database, randomly selected, for our probe and gallery datasets.

```

1 perc_probe = 0.001
2 perc_gallery = 0.1

```

```

1 print('Generating samples')
2 probes = random.sample(mindtcts, int(perc_probe * len(mindtcts)))
3 gallery = random.sample(mindtcts, int(perc_gallery * len(mindtcts)))
4 print('|Probes|=', len(probes))
5 print('|Gallery|=', len(gallery))

```

We can now compute the matching scores between the probe and gallery sets. This will use all cores available on this workstation.

```

1 print('Matching')
2 input = [bozorth3_input(probe=probe, gallery=gallery)
3           for probe in probes]
4 bozorth3s = pool.map(run_bozorth3, input)

```

bozorth3s is now a list of lists of bozorth3 instances.

```

1 print('|Probes|=', len(bozorth3s))
2 print('|Gallery|=', len(bozorth3s[0]))
3 print('Result:', bozorth3s[0][0])

```

Now add the results to the database

```

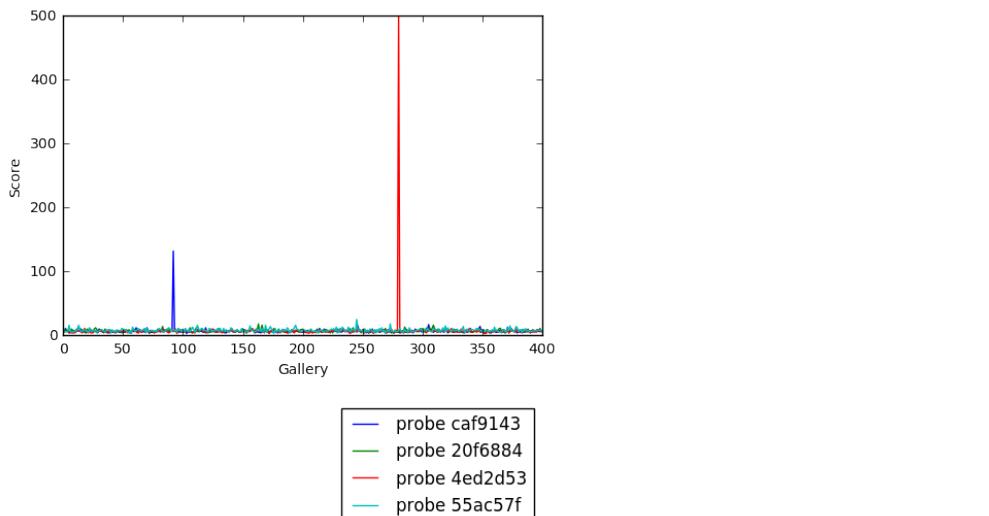
1 dbfile = os.path.join(prefix, 'scores.db')
2 conn = sqlite3.connect(dbfile)
3 cursor = conn.cursor()
4 cursor.execute(bozorth3.sql_stmt_create_table())

1 for group in bozorth3s:
2     vals = map(bozorth3.sql_prepared_stmt_insert_values, group)
3     cursor.executemany(bozorth3.sql_prepared_stmt_insert(), vals)
4     conn.commit()
5     print('Inserted results for probe', group[0].probe)

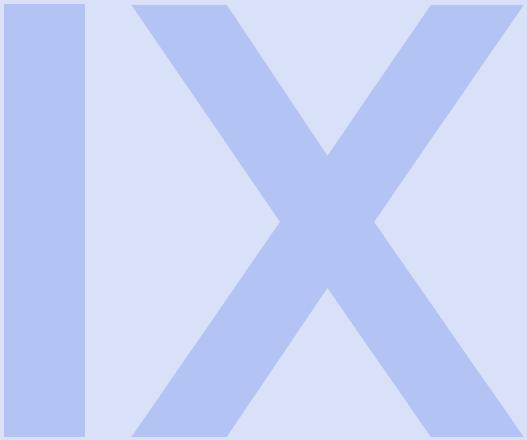
```

We now plot the results.

```
1 plot(dbfile, nprobes=len(probes))
```



```
1 cursor.close()
```



Python Cloudmesh



30. Cloudmesh

In this chapter we will be using some advanced Python features to enhance Cloudmesh that is supposed to easily manage multiple clouds. We will be explicitly using python 3 and do not worry about backwards compatibility.

We will be developing as community such new features and integrate them at the end in an extensible package management system allowing to load in new features with pip.

We are trying to develop the following:

Configuration so that we can easily configure and add various clouds to our multi-cloud environment.

Database of virtual machines and clouds so that they can be managed across different clouds in a multi cloud environment

API Classes so that we can use python as a convenient programming environment.

Context libraries so taht in python we can easily apply context for clouds and virtual machines on a block of statements

Command Shell so that we can similar to matlab and other shells execute multiple commands

REST Services so that we can access the features form other programming environments and different programming languages.

Parallel Services so that we can issue commands in parallel and manage virtual machines in a multi cloud environment.

30.1 Configuration

As we are developing a multi-cloud environment, we need some mechanism to define the clouds easily. To make our development effort simpler, we like to point out that the configuration file must be stored in a particular location relative to the home directory. We store the file in `~/.cloudmesh/class.yaml`. Additionally we store our cloud passwords in this file in cleartext and thus we must make sure our machine is not compromized and that we properly protect the file.

On a unix system you do this with:

```
mkdir ~/.cloudmesh
touch ~/.cloudmesh/class.yaml
chmod go-rw ~/.cloudmesh
chmod go-rw ~/.cloudmesh/class.yaml
```

In that directory we store a file similar to the following file:

```
version: 5.0
profile:
    firstname: Gregor
    lastname: von Laszewski
    email: laszewski@gmail.com
cloudmesh:
    default:
        - chameleon
    active:
        - chameleon
clouds:
    uc:
        name: Chameleon UC
        host: chameleoncloud.org
        type: openstack
        version: liberty
        credentials:
            OS_AUTH_URL: https://openstack.uc.chameleoncloud.org:5000/v3
            OS_PASSWORD: TBD
            OS_TENANT_NAME: CH-818664
            OS_TENANT_ID: CH-818664
            OS_PROJECT_NAME: CH-818664
            OS_PROJECT_DOMAIN_ID: default
            OS_USER_DOMAIN_ID: default
            OS_USERNAME: TBD
            OS_VERSION: liberty
            OS_REGION_NAME: RegionOne
    default:
        flavor: m1.small
        image: Ubuntu-Server-14.04-LTS
tacc:
    name: Chameleon TACC
    host: chameleoncloud.org
    type: openstack
    version: liberty
    credentials:
        OS_AUTH_URL: https://openstack.tacc.chameleoncloud.org:5000/v3
        OS_PASSWORD: TBD
        OS_TENANT_NAME: CH-818664
        OS_TENANT_ID: CH-818664
        OS_PROJECT_NAME: CH-818664
        OS_PROJECT_DOMAIN_ID: default
        OS_USER_DOMAIN_ID: default
        OS_USERNAME: TBD
        OS_VERSION: liberty
        OS_REGION_NAME: RegionOne
    default:
        flavor: m1.small
        image: Ubuntu-Server-14.04-LTS
```

Important to note is that this file defines multiple clouds and uses TBD for password and username which you may want to change. However, we also would like to support a mode that when the password is defined to be TBD that it is asked from the terminal. This way we do not necessarily

have to store the password here. In future we will enhance this file to be encrypted and decrypted with a password protected ssh key.

The file is ayaml file as the typical configuration in python is not suitable to easily store hierarchical data. YAML is also more readable than json so it provides a really good way of defining the configuration data. Problematic with yaml readers however are that they typically do not preserve the read order. Your task will be to write a *short* yaml configuration reader that preserves the order. You are encouraged to reuse methods. What you are not supposed to do is to reimplement yaml.

There are some special properties of this file that we need to discuss.

- clouds are listed in the clouds section
- the credentials section to each cloud defines how to connect to the cloud with python libraries such as libcloud. Each cloud type will have different parameters.
-

30.2 Storage

As we need to store some of the data we must identify a suitable database for storing information about virtual machines and other information related to the clouds. Although shelve comes in mind, we found out that it is not compatible between python 2 and 3 which may be an issue in future. Also when considering services such as mongodb they have to be started and properly secured. This naturally can be done with containers. We also do not want to use large frameworks such as django which come with build in object models as they are not lightweight. Hence, we start we just use a file based sql database as provided with sqlite3.

30.2.1 sqlite3

While we keep the configuration in the configuration yaml file we intend to create a database entry for virtual machines we start in the cloud. In order to store hierarchical information that we may obtain in dict format from a virtual machine we can easily create flattened out data structures, by simply connecting the attribute names and separate them by _.

Let us assume we want to store an object of the following form:

```
element = {
    'id': 1,
    'cloud': 'chameleon',
    'name': 'vm1',
    'data': {
        "image": 'ubuntu',
        "flavor": "small"
    }
}
```

A table that could store such an object could be

```
create table element (
    id      integer primary key,
    name   text,
    cloud  text,
    data_image: text,
    data_flavor: text
);
```

Obviously, we could create the table automatically from recursively iterating through the dict to

make our approach generalized for any dict. As for the primary key, we simply assume it is always the id which is an integer that always increases and is stored in the database. as a separate element.

Thus we probably want a table generator such as

```
248 class Database (object):
249     @staticmethod
250     def generate (dictionary):
251         # implement me
```

Additionally we want to create convenience methods for adding, deleting, and searching information

30.2.2 Context

Python provides the feature of a context that we are well familiar with from file management. An example is:

```
252 with open ('/tmp/gregor.txt', 'wt') as f:
253     f.write('Hello Gregor')
254 # after this, the file is automatically closed
```

If we look at this example it is desirable to develop at least two context for multicloud environments. The first is to manage virtual machines on named clouds and issue action on it, such as *start*, *stop*, *suspend*, *resume*, *delete*. In the other context we like to issue such asction on named virtual machines.

To illustrate what we have in mind, please take a look at our initial examples.

Cloud Context

When defining the following cloud context

```
255 class Cloud:
256
257     def __init__(self, name):
258         self.name = name
259
260     def __enter__(self):
261         print ('Running on:', self.name)
262         return self
263
264     def __exit__(self, exc_type, exc_val, exc_tb):
265         print('__exit__()')
266
267     def machine(self, name, action):
268         print (name, action)
```

we can issue conveniently commands such as the following

```
269 cloud = 'chameleon'
270 with Cloud(cloud) as c:
271     vm = c.machine('vml', 'start')
```

It is obvious that through this abstraction we can formulate a tempalited bahavior such as starting a virtual machine and through the switch of a ingle variable (`cloud`) issue the command on other clouds.



Cloud Computing

31	Cloud Computing Fundamentals	327
31.1	Introduction	
31.2	Data Center Model	
31.3	Data Intensive Sciences	
31.4	IaaS, PaaS and SaaS	
31.5	Challenges	
32	IaaS	329
32.1	Growth of Virtual Machines	
32.2	Implementation Levels	
32.3	Tools and Mechanisms	
32.4	CPU, Memory & I/O Devices	
32.5	Clusters and Resource Management	
32.6	Data Center Automation	
32.7	Clouds in the Workplace	
32.8	Checklists and Challenges	
32.9	Data Center Setup	
32.10	Cultivating Clouds	



31. Cloud Computing Fundamentals



[section/icloud/course/fundamentals.tex](#)

31.1 Introduction

Changes in computing technology for the past five decades are discussed. The rise of Big Data is shown in terms of its growth and significance. A prediction is made that the paradigm which has held 'til now of individual researchers with personal computers will give way to communities of researchers organizing through clouds. A more in-depth look at Unit 1 follows, focusing on the chapters from Distributed and Cloud Computing: From Parallel Processing to the Internet of Things.

[Introduction \(8:31\)](#)

[Introduction \(Page 1\)](#)

TODO: Information after after 4:56 is outdated.

31.2 Data Center Model

A look at what truly defines a ‘cloud’. Advantages like scalability and cost-effectiveness have promulgated commercial cloud offerings such as Amazon EC2. Cloud architecture as divided into three layers: Infrastructure as a Service, Platform as a Service, and Software as a Service. AzureBlast is used as an example of how to utilize the cloud setup. Certain misconceptions about clouds are then presented for further discussion.

[Data Center Model \(8:08\)](#)

[section/icloud/course/fundamentals.tex](#)

Data Center Model (Page 9) 

31.3 Data Intensive Sciences

Some time is spent analyzing the current age of vast data growth, where business, science, and consumer activity has seen an explosion of stored data measured in exabytes. In response to this, the way we conduct scientific research has also undergone an upgrade. However, the average scientist would rather focus on their own research rather than spend time trying to learn different methods of cloud and supercomputing.

Data Intensive Sciences (2:44) 

Data Intensive Sciences (Page 19) 

31.4 IaaS, PaaS and SaaS

Definitions and examples are given for Infrastructure as a Service, Platform as a Service, and Software as a Service. A chart is shown illustrating how use of clouds trades cost and control for efficiency. Following this is an exploration of the MapReduce program, and an illustration of its concepts through WordCount. Finally, four distinct approaches to MapReduce are compared.

IaaS/PaaS/SaaS (10:17) 

IaaS, PaaS and SaaS (Page 25) 

31.5 Challenges

The demands of Big Data calls for advances in areas like distributed computing, systems management, internet technology, and hardware. Clouds have become more prominent in the last few decades, so much so that many people today take advantage of them without even knowing it. Of course, this has also led to increased concerns about security, price, tech support, etc. In spite of this, clouds still have clear advantages over traditional computing models. A quiz is offered at the end asking students to correctly place software in a hierarchy of computing.

Challenges (5:27) 

Challenges (Page 42) 



32. IaaS



[section/icloud/course/iaas.tex](#)

Examples and definitions are given for SaaS, PaaS, and IaaS. Computational models must be designed with the problems and effective resources in mind. A demonstration of cloud use for Bioinformatics shows how clouds offer advantages of provisioning and virtual cluster support. Overhead and performance issues are touched upon through charts showing the use of three different virtual clusters.

32.1 Growth of Virtual Machines

Importance of virtualization is explored, including cross-platform applications. Virtualization has seen rapid growth in recent years in terms of use and services offered. Virtual machines differ from traditional computers in that software virtualization layer (hypervisor) runs on hardware, allowing guest OS to run on top of host OS. VMs can run independent of hardware specifications. Four different types of VM architecture, defined by the layer which the virtual machine monitor (VMM) runs on. VM is identical to physical machines and can be saved and stored, as well as migrated across hardware.

[*Growth of Virtual Machines \(10:16\)*](#)

[*Growth of Virtual Machines \(Page 28\)*](#)

[*Growth of Virtual Machines - pptx \(Page 28\)*](#)

32.2 Implementation Levels

Virtualization can be implemented on five levels: application, library, OS, hardware, and instruction. Their benefits are compared in terms of performance, flexibility, complexity, and isolation. A layout is provided for the Linux virtualization layer, OpenVZ (OS level), which creates virtual private servers. CUDA is a high performance computing library, not designed for VMs; vCUDA is a virtual layer that allows interaction between CUDA and VMs, creating a virtual CUDA library.

- [*Implementation Levels \(7:57\)*](#)
- [*Implementation Levels \(Page 41\)*](#)
- [*Implementation Levels - pptx \(Page 41\)*](#)

32.3 Tools and Mechanisms

A list of major hypervisors is given. Type 1 hypervisor resides on the bare metal computer, while Type 2 runs over the host OS. XEN is an open source hardware level hypervisor: consists of hypervisor, kernel, and application. Domain0 in XEN is a VM that manages other VMs. Two types of hardware virtualization: full virtualization and host-based virtualization. Para-virtualization does not need to modify the guest OS like full virtualization and works through hypercalls. An example is the ESX server from VMware.

- [*Tools and Mechanisms \(7:32\)*](#)
- [*Tools and Mechanisms \(Page 47\)*](#)
- [*Tools and Mechanisms - pptx \(Page 47\)*](#)

32.4 CPU, Memory & I/O Devices

A hybrid approach to virtualization involves offloading some tasks to the hardware to reduce overhead. This can be combined with para-virtualization for even greater effects. In a guest OS, the VMM provides shadow page tables to transfer virtual memory to machine memory. An example is shown in the Intel Extended Page Table. A virtualization layer for an I/O device is possible, allowing it to act like a physical device and manage host and guest addresses, shown in a detailed VMware example.

- [*CPU, Memory & I/O Devices \(6:41\)*](#)
- [*CPU, Memory & I/O Devices \(Page 58\)*](#)
- [*CPU, Memory & I/O Devices - pptx \(Page 58\)*](#)

32.5 Clusters and Resource Management

Characteristics of VM clusters are listed, including the ability to run multiple VMs on the same node and size alteration. Physical clusters are linked through nodes, while virtual clusters can be linked through physical or virtual nodes and can be replicated in virtual servers. Prepackaged OS can be installed in a virtual cluster. Should a VM fail for any reason, its image can be migrated to a new host so work is not lost. An example of this is demonstrated with XEN.

[*Clusters and Resource Management \(5:07\)*](#) [*Clusters and Resource Management \(Page 66\)*](#) [*Clusters and Resource Management - pptx \(Page 66\)*](#)

32.6 Data Center Automation

Whole data centers can be virtualized, enabling for the construction of private clouds. Some tools for Infrastructure as a Service clouds are Nimbus, Eucalyptus, OpenNebula, and vSphere. Eucalyptus is shown in greater detail. Trust issues in cloud security are answered in virtual machines. Suggested reading material is provided at the end.

[*Data Center Automation \(3:30\)*](#) [*Data Center Automation \(Page 74\)*](#) [*Data Center Automation - pptx \(Page 74\)*](#)

32.7 Clouds in the Workplace

Clouds run as servers for data storage and sharing on the Internet in an on-demand capacity. Cloud services are scalable depending on the client's needs, allowing for a seemingly limitless source of computing power that can expand or shrink to meet financial demands. Some examples of cloud services are LinkedIn, Amazon S3, and Google App Engine. Different variations of clouds like IaaS and PaaS are offered by both open source and commercial providers. Cloud systems are composed of separate elements like Eucalyptus, Xen and VMWare.

[*Clouds in the Workplace \(7:13\)*](#) [*Clouds in the Workplace \(Page 1\)*](#)

32.8 Checklists and Challenges

The capabilities of several IaaS cloud structures like Amazon EC2 or PaaS like Microsoft Azure are listed. Public and private clouds share certain features; the main difference is public clouds are owned by service providers while private clouds are offered by individual corporations. Certain enabling technologies are required for clouds to provide quick and scalable computing. These include virtual cluster provisioning and multi-tenant environments. PaaS demands the capability to process huge amounts of data as in the case of web searches. Some challenges faced by cloud computing include vendor lock-in owing to lack of standard APIs and metrics; for scientists, there is uncertainty about whether experiments can be reproduced effectively in different cloud environments. However there are distinct advantages clouds potentially have to offer: standardized APIs can eliminate lock-in, and encryption offers data confidentiality.

[*Checklists and Challenges \(9:08\)*](#) [*Checklists and Challenges \(Page 11\)*](#)

32.9 Data Center Setup

Huge data centers enable cloud computing, containing up to a million servers. Large data centers charge less for their services than small ones. A diagram illustrates the typical setup of a cloud; rack space on the bottom, on top of which are load balancers, then excess routers and border routers. The next figure compares cost effectiveness in a traditional IT model to a cloud. Other figures display small server clusters and a typical data center arrangement, including emergency power supply and cooling system. A chart shows the power consumption based on CPU, disk, etc. Disks in warehouse servers may be onsite or attached to outside connections like InfiniBand. Switches can form an array of racks. The distribution of memory across a local, rack, or array server in warehouse server setup is listed.

[Data Center Setup \(7:49\)](#)

[Data Center Setup \(Page 16\)](#)

32.10 Cultivating Clouds

Power utilization effectiveness (PUE) for a warehouse is determined by comparing it to IT power usage. Racks can contain 40 servers, shipping containers can have up to 1,000 servers; a data center could take 2 years to construct. Warehouse scale computing has greater economy of scale than data centers by reducing network and administrative costs. Individual users can interact with clouds in the SaaS model, while organizations use PaaS. Clouds generally use VMs to recover from system failures. It is predicted that the cloud job market and demand for clouds will experience great growth in the future. Clouds have become ubiquitous in all aspects of the private and public sector. In the future clouds must take into account user privacy, data security and copyright protection.

[Cultivating Clouds \(5:10\)](#)

[Cultivating Clouds \(Page 15\)](#)

[Cultivating Clouds - Conclusions \(Page 1\)](#)

Data Management

33	NoSQL	335
33.1	RDBMS vs. NoSQL	
33.2	NoSQL Characteristics	
33.3	BigTable	
33.4	HBase	
33.5	HBase Coding	
33.6	Indexing Applications	
33.7	Related Work	
33.8	Indexamples	
33.9	Indexing 101	
33.10	Social Media Searches	
33.11	Analysis Algorithms	



33. NoSQL

F section/icloud/course/nosql.tex

33.1 RDBMS vs. NoSQL

- [RDBMS vs. NoSQL \(9:22\)](#)
- [RDBMS vs. NoSQL \(Page 1\)](#)
- [RDBMS vs. NoSQL - pptx \(Page 1\)](#)

33.2 NoSQL Characteristics

Clouds have arisen as an answer to the data demands of social media. Three major programs for NoSQL are BigTable, Dynamo, and CAP theory. NoSQL is not meant to replace SQL, but to tackle the large-data problems SQL is not well equipped to handle. SQL ACID transactions are Atomic, Consistent, Isolated, and Durable. Consistency can be either strong (ACID) or weak (BASE). CAP theorem offers Consistency, Availability, and Partition tolerance, only two of which can coexist for a shared-data system. NoSQL comes in two varieties, each with pros and cons: Key-Value or schema-less. Common advantages of NoSQL include their being open source and fault tolerant.

- [NoSQL Characteristics \(10:31\)](#)
- [NoSQL Characteristics \(Page 11\)](#)
- [NoSQL Characteristics - pptx \(Page 11\)](#)

section/icloud/course/nosql.tex

33.3 BigTable

Big Table is a key-value NoSQL model with data arranged in rows and columns. It is composed of Data File System, Chubby, and SSTable. A tablet is a range of rows in BigTable. The master node assigns tablets to tablet servers and manages these servers. Memory is conserved by making SSTables and memtables compact. BigTable is used in features of Google like their search engine and Google Earth.

[BigTable \(6:55\)](#)

[BigTable \(Page 28\)](#)

[BigTable - pptx \(Page 28\)](#)

33.4 HBase

HBase is a NoSQL core component of the Hadoop Distributed File System. It is a scalable distributed data store. A timeline of HBase and Hadoop is shown. BigTable still has its uses but does not scale well to large amounts of analytic processing. HBase has a row-column structure similar to BigTable as well as master and slave nodes. Its place in the architecture of HDFS is shown in a diagram.

[HBase \(7:37\)](#)

[HBase \(Page 44\)](#)

[HBase - pptx \(Page 44\)](#)

33.5 HBase Coding

This video gives an overview of the code used in the installation of HBase and connecting to it.

[4:30 \(HBase Coding\)](#)

[HBase Coding \(Page 60\)](#)

[HBase Coding - pptx \(Page 60\)](#)

33.6 Indexing Applications

A brief summary of the course up to this point is given, followed by a diagram showing the setup of a search engine. Google's search engine contains three key technologies: Google File System, BigTable, and MapReduce. However, research into big data remains difficult owing to the scope of its size. Social media data in particular is a huge source of data with numerous subsets, all of which demands specific approaches in terms of search queries. There are three stages to this approach: query, analysis, and visualization.

[Indexing Applications \(9:33\)](#)

[Indexing Applications \(Page 1\)](#)

[Indexing Applications - pptx \(Page 1\)](#)

33.7 Related Work

Indexing improves efficiency in querying data subsets and analysis. Indices can be single (B+, Hash) or multi-dimensional (R, Quad). Four databases which utilize indexing are HBase, Cassandra, Riak, and MongoDB. Current indexing strategies have limits; for instance, they cannot support range queries or only retrieve Top ‘n’ most relevant topics. Customizability of indexing among NoSQL databases is desirable.

[Related Work \(5:56\)](#)

[Related Work \(Page 11\)](#)

[Related Work - pptx \(Page 11\)](#)

33.8 Indexamples

Mapping between metadata and raw index data is the essential issue with indexing. Examples are shown for HBase, Riak, and MongoDB. An abstract index structure contains index keys, entry IDs among multiple entries, and additional fields. Index configuration allows for customizability through choice of fields, which can be anything from timestamps, text, or retweet status.

[Indexamples \(8:35\)](#)

[Indexamples \(Page 15\)](#)

[Indexamples - pptx \(Page 15\)](#)

33.9 Indexing 101

User-defined index allows a user to select the fields used in their search. Data records are indexed or un-indexed. Index structure is made up of key, entry ID, and entry fields. A walk-through customized index creation is shown on HBase, called IndexedHBase. HBase is suited to accommodate the creation of index tables. A performance test of IndexedHBase is done on the Truthy Twitter repository, displaying the various tables that can be created with different criteria. Loading time for large-scale historical data can be reduced by adding nodes. Streaming data can be handled by increasing loaders. A comparison of query evaluation is made between IndexedHBase and Riak, with Riak being more efficient with small data loads but IndexedHBase proving superior for large-scale data.

[Indexing 101 \(9:53\)](#)

[Indexing 101 \(Page 20\)](#)

[Indexing 101 - pptx \(Page 20\)](#)

33.10 Social Media Searches

The Truthy Project archives social media data by way of metadata memes. Some problems faced in analyzing this data include its large volume, sparsity of information in tweets, and attempting to arrange streaming tweets. Apache Open Stack upgrades Hadoop 2.0 with YARN and a new HDFS. A diagram displays an indexing setup for social media data with YARN.

[*Social Media Searches \(6:19\)*](#) [*Social Media Searches \(Page 28\)*](#) [*Social Media Searches - pptx \(Page 28\)*](#) 

33.11 Analysis Algorithms

Another method of use for inverted indices is in analysis algorithms. The mathematics involved in this is explored, as well as how it relates to index data, mapping, and reducing. Rather than scanning all raw data present, indices allow for searching only the relevant data. An example is given illustrating how this decreases the time needed to search hashtags in Twitter.

[*Analysis Algorithms \(6:57\)*](#) [*Analysis Algorithms \(Page 35\)*](#) [*Analysis Algorithms - pptx \(Page 35\)*](#) 



SaaS

34	Search Engine	341
34.1	Google Components	
34.2	Google Architecture	
34.3	Google History	



34. Search Engine

F section/icloud/course/saas.tex

34.1 Google Components

- [*Google Components \(7:02\)*](#)
- [*Google Components \(Page 1\)*](#)
- [*Google Components - pptx \(Page 1\)*](#)

34.2 Google Architecture

- [*Google Architecture \(8:40\)*](#)
- [*Google Architecture \(Page 6\)*](#)
- [*Google Architecture - pptx \(Page 6\)*](#)

34.3 Google History

Google History: <https://youtu.be/Kg0NK0XUkHw?t=175> (starting 2:55)

Google Search Engine 1: <https://www.youtube.com/watch?v=S2oT7uMw5Yg>

Google Search Engine 2: <https://www.youtube.com/watch?v=pxos3Yt6y6I>

[*Google History \(10:36\)*](#)

Google History (Page 14) 

Google History - ptx (Page 14) 

MapReduce

35	MapReduce	345
35.1	Apache Data Analysis OpenStack	
35.2	MapReduce	
35.3	Hadoop Framework	
35.4	Hadoop Tasks	
35.5	Fault Tolerance	
35.6	Hadoop WordCount on VMs	
35.7	Programming on a Compute Cluster	
35.8	How Hadoop Runs on a MapReduceJob	
35.9	Literature Review	
35.10	Introduction to BLAST	
35.11	BLAST Parallelization	
35.12	SIMD vs MIMD;SPMD vs MPMD	
35.13	Data Locality	
35.14	Optimal Data Locality	
35.15	Task Granularity	
35.16	Resource Utilization and Speculative Execution	
36	Iterative Map Reduce	351
36.1	Introduction to MapReduce	
36.2	Google Search Engine 1	
36.3	Google Search Engine 2	
36.4	Hadoop PageRank	
36.5	Discussions and ParallelThinking	
36.6	Hadoop Extensions	
36.7	Iterative MapReduce Models	
36.8	Parallel Processes	
36.9	Static and Variable Data	
36.10	MapReduce Model Comparison	
36.11	Twister K-means	
36.12	Coding and Iterative Alternatives	



35. MapReduce

F section/icloud/course/mapreduce.tex

35.1 Apache Data Analysis OpenStack

The buildup of Big Data has seen the development of new data storage systems like MapReduce and Hadoop. Apache's Big Data Stack houses a host of programs designed around Google's offerings like MapReduce. The architecture of Hadoop 1.0 and 2.0 are compared, along with an examination of the MapReduce concept. A demo video of Twister-MDS includes a 3-dimensional representation of data cluster sorting through the PlotViz program. Data analysis tool Twister boasts features like in-memory support of tasks, data flow separation, and portability.

[Apache Data Analysis OpenStack \(12:01\)](#)

[Apache Data Analysis OpenStack \(Page 1\)](#)

[Apache Data Analysis OpenStack - pptx \(Page 1\)](#)

35.2 MapReduce

MapReduce was designed by Google to address the problem of large-scale data processing. A breakdown of basic MapReduce terms and functions follows. Use of MapReduce has flourished since its premier, as illustrated by an in-depth example of its use in WordCount. Finally the basic process of MapReduce is shown.

[MapReduce \(9:07\)](#)

[MapReduce \(Page 6\)](#)

MapReduce - pptx (Page 6) 

35.3 Hadoop Framework

Hadoop is an open source version of MapReduce designed for broad application in terms of code and settings. Storage is done in the Hadoop Distributed File System through master and slave nodes. Compute is handled by JobTracker and TaskTracker; the duties of these two intertwined programs are then explored more fully.

Hadoop Framework (8:32) 

Hadoop Framework (Page 15) 

Hadoop Framework - pptx (Page 15) 

35.4 Hadoop Tasks

The Map stage of MapReduce is shown in greater detail. This process starts with Hadoop Distributed File System, which handles the input data. Key value pairs are assigned to the data blocks. Combiner reduces data size and Partitioner determines distribution of keys among reducers. Intermediate data is stored in a circular buffer before being sent to reduce tasks. Shuffle and Merge are used to order and reduce size of intermediate data. Reduce tasks take over then to determine the output data format. A final chart illustrates the concept of parallelism in MapReduce.

Hadoop Tasks (11:01) 

Hadoop Tasks (Page 24) 

Hadoop Tasks - pptx (Page 24) 

35.5 Fault Tolerance

Fault tolerance is a natural benefit of MapReduce. The master node pings worker nodes regularly to verify they are working, and acts accordingly if they do not respond. A diagram illustrates the files which are in charge of things like number of map and reduce tasks, and what to do when the limit is reached on the buffer. The lecture ends with a discussion of class assignments.

Fault Tolerance (2:45) 

Fault Tolerance (Page 36) 

Fault Tolerance - pptx (Page 36) 

35.6 Hadoop WordCount on VMs

Hadoop WordCount on VMs (7:30) 

Hadoop WordCount on VMs (Page 17) 

Hadoop WordCount on VMs - pptx (Page 17) 

35.7 Programming on a Compute Cluster

Hadoop is now a large part of Yahoo!'s system setup, as well as handling a tremendous variety of data in other areas like medicine and business. A list of time spans for actions in system requirements is given. The original MapReduce was designed to resolve problems like load balancing and machine failures.

- [*Programming on a Compute Cluster \(6:01\)*](#)
- [*Programming on a Compute Cluster \(Page 1\)*](#)
- [*Programming on a Compute Cluster - pptx \(Page 1\)*](#)

35.8 How Hadoop Runs on a MapReduceJob

A detailed diagram of the MapReduce job framework is given. This includes task status updates, shuffling, and writing data to nodes. MapReduce is a C++ framework, while Hadoop is written in Java. Shuffling and sorting occurs in the map phase. Reduce reads and writes files to HDFS, and the merger generates the final result. The second Quiz is given at the end.

- [*How Hadoop Runs on a MapReduceJob \(9:25\)*](#)
- [*How Hadoop Runs on a MapReduceJob \(Page 8\)*](#)
- [*How Hadoop Runs on a MapReduceJob - pptx \(Page 8\)*](#)

35.9 Literature Review

This video deals primarily with scientific papers written on the topic of MapReduce and related programs. There is a certain criteria for judging scientific submissions. The first paper highlights Google File System, covering topics like data chunks, metadata, and replicas. This is followed by MapReduce and BigTable.

- [*Literature Review \(9:43\)*](#)
- [*Literature Review \(Page 16\)*](#)
- [*Literature Review - pptx \(Page 16\)*](#)

35.10 Introduction to BLAST

There are four types of programming model communication patterns: embarrassingly parallel (only map), classic map/reduce, iterative map/reduce, and loosely synchronous. The basic bioinformatics BLAST (Basic Local Alignment Sequence Tool) program data flow is illustrated. An example of database creation comes from the Seattle Children's Hospital. BLAST uses scores to find similar sequences in databases.

- [*Introduction to BLAST \(8:27\)*](#)
- [*Introduction to BLAST \(Page 1\)*](#)
- [*Introduction to BLAST - pptx \(Page 1\)*](#)

35.11 BLAST Parallelization

The role of master and worker nodes in BLAST multi-thread usage is discussed. BLAST can be parallelized in several ways: multi-thread, query segmentation, and database segmentation. BLAST is pleasingly parallel in application, but many programs are not. Further information about articles featuring BLAST is provided at the end.

- [*BLAST Parallelization \(4:44\)*](#)
- [*BLAST Parallelization \(Page 13\)*](#)
- [*BLAST Parallelization - pptx \(Page 13\)*](#)

35.12 SIMD vs MIMD;SPMD vs MPMD

Four types of parallel models: SISD (traditional PCs), SIMD (GPUs), MISD (shuttle flight control computer), MIMD (distributed systems). Point-to-point (P2P) communication in MPI is used as an example of parallelization. Each successive process adds its own stamp to the data before passing it on to the next. Matrix multiplication for scientific applications differs from the norm in that data is sent in a matrix, not a string. WordCount functions in a map/reduce pattern. These are all types of SIMD. SPMD and MPMD are two other types of model.

- [*SIMD vs MIMD;SPMD vs MPMD \(9:42\)*](#)
- [*SIMD vs MIMD;SPMD vs MPMD \(Page 1\)*](#)
- [*SIMD vs MIMD;SPMD vs MPMD - pptx \(Page 1\)*](#)

35.13 Data Locality

A brief review is given of previous topics. As opposed to MPI and HPC, MapReduce brings the computation to the data, rather than vice-versa. This is done to limit energy usage and network congestion. Several factors such as number of nodes and tasks can impact data locality. An equation to improve data locality is tested in an experiment, whose results are given. By default, Hadoop determines scheduling of tasks to available slots in terms of best local composition, not global.

- [*Data Locality \(8:36\)*](#)
- [*Data Locality \(Page 10\)*](#)
- [*Data Locality - pptx \(Page 10\)*](#)

35.14 Optimal Data Locality

Global data optimization can be achieved through a proposed algorithm given here. Task, slot, and cost are factors in this algorithm. Network bandwidth must also be taken into consideration when assigning tasks to slots. Linear Sum Assignment Problems require greater time to finish when matrix size is increased. Two different scheduling algorithms were designed to improve the original one in Hadoop. An experiment was run comparing all three, with the network topology-aware algorithm clearly outperforming the others.

[*Optimal Data Locality \(4:17\)*](#) [*Optimal Data Locality \(Page 17\)*](#) [*Optimal Data Locality - pptx \(Page 17\)*](#)

35.15 Task Granularity

Size of data blocks affects load balancing and overhead. Using Bag of Divisible Tasks method, tasks can be split into sub-tasks and distributed amongst slots to maximize efficiency. When splitting tasks, one must take into account when and which tasks to split, as well as how and how many. In our current proposed algorithm, tasks are split until each slot is occupied. It also uses ASPK (Aggressive Scheduling with Prior Knowledge) to split larger tasks first and when the performance gain is deemed optimal. Optimal and Expected Remaining Job Execution Time can help determine task splitting. Several examples are offered with either single or multiple jobs.

[*Task Granularity \(9:51\)*](#) [*Task Granularity \(Page 29\)*](#) [*Task Granularity - pptx \(Page 29\)*](#)

35.16 Resource Utilization and Speculative Execution

Resource stealing involves appropriating cores that are kept in reserve on separate nodes and returning them when the computation is over. Speculative execution addresses fault tolerance; when the master node notices a task is running slowly, it will start a speculative task which can take over if it is determined the original task will not finish in time. Overuse of speculative tasks can lead to poor data locality and higher energy demands.

[*Resource Utilization and Speculative Execution \(3:52\)*](#) [*Resource Utilization and Speculative Execution \(Page 46\)*](#) [*Resource Utilization and Speculative Execution - pptx \(Page 46\)*](#)



36. Iterative Map Reduce

F section/icloud/course/iterative-mapreduce.tex

36.1 Introduction to MapReduce

A review covers cloud computing levels, MapReduce, the course structure, etc. This is followed by a look at Google and their initial offering, Google search engine. Amount of tasks performed on this engine increased considerably over the course of a single decade.

[MapReduce Refresher \(9:00\)](#)

[MapReduce Refresher \(Page 1\)](#)

36.2 Google Search Engine 1

The Google web server relies on index and doc servers. Index servers allow the search engine to not have to depend on manually checking every document, reducing computing power demands. Index partitioning can be accomplished either through subsets of documents or words. Basic differences between index and doc servers are discussed. Cache servers save previous query results and can bypass index/doc servers for repeat queries.

[Google Search Engine 1 \(8:04\)](#)

[Google Search Engine 1 \(Page 15\)](#)

[Google Search Engine 1 - pptx \(Page 15\)](#)

36.3 Google Search Engine 2

Cache servers greatly enhance the performance of search engines. However, this duplication of queries can lead to higher latency. Crawling in a search engine handles subsets of websites. Batch indexing is the simplest way to create indexes, although it lacks advanced features like checkpointing, which could lead to issues down the line. In-memory index added to Google over a decade ago; increases throughput and decreases latency. Image-based and video-based searches were added in 2007, among others. Google File System, MapReduce and BigTable are key components of Google's current search structure. A discussion of the initial Google proposal paper follows.

[Google Search Engine 2 \(8:32\)](#)

[Google Search Engine 2 \(Page 21\)](#)

[Google Search Engine 2 - pptx \(Page 21\)](#)

36.4 Hadoop PageRank

PageRank algorithm in Google ranks a webpage's popularity and relevance. The PageRank calculation formula is examined. After this comes an example of its performance and further mathematical formulae involved in its application.

[Hadoop PageRank \(7:58\)](#)

[Hadoop PageRank \(Page 1\)](#)

[Hadoop PageRank - pptx \(Page 1\)](#)

36.5 Discussions and ParallelThinking

Four types of MapReduce: pleasingly parallel, classic, iterative, and loosely synchronous. A diagram shows the flow of data in MapReduce. Specific formulae for PageRank are shown with and without the damping factor. Key-value pairs can be written in matrix form by defining the keys as nodes. Map tasks must make sure to handle dangling nodes (isolated from neighbors), distributed page-rank contribution, and reducer output being the same format as map input. Reduce input is key-value pairs. Ideas behind parallel thinking are analyzed, along with a list of related reading. Seven important questions are asked concerning parallel computing. 13 'Dwarves' are different methods of parallel computing, including MapReduce.

[Discussions and ParallelThinking \(11:12\)](#)

[Discussions and ParallelThinking \(Page 10\)](#)

[Discussions and ParallelThinking - pptx \(Page 10\)](#)

36.6 Hadoop Extensions

A model of MapReduce shows its structure. Dryad is Microsoft's version of parallel processing. Twister is an iterative map-reduce framework, as are Haloop, Spark and Pregel. A comparison of their features and capabilities is included.

[Hadoop Extensions \(5:37\)](#) [Hadoop Extensions \(Page 50\)](#)

36.7 Iterative MapReduce Models

An introduction to the idea of iterative MapReduce. An overview of other MapReduce models follows. Map Only model has parallel map tasks with no communication between them. Classic MapReduce involves parallel map tasks and reduce tasks which aggregate output and allow legacy code. Loosely Synchronous is an MPI model used in computation and communication of scientific applications.

[Iterative MapReduce Models \(6:46\)](#) [Iterative MapReduce Models \(Page 1\)](#) [Iterative MapReduce Models - pptx \(Page 1\)](#)

36.8 Parallel Processes

CPU performance increases according to Moore's Law can no longer keep up with the high volume of data being generated. Multi-core architecture is a response to this issue. It requires runtime approaches supporting parallelism, either data-centric for higher throughput (MapReduce) or the traditional HPC approach for optimized computation performance (MPI). MapReduce allows for moving computation to the data. A diagram illustrates the base MapReduce process. MapReduce is designed to improve I/O and handle intermediate data, task scheduling, and fault tolerance. Versions of MapReduce like Hadoop, Dryad and MPI boast different features and programming languages.

[Parallel Processes \(9:44\)](#) [Parallel Processes \(Page 4\)](#) [Parallel Processes - pptx \(Page 4\)](#)

36.9 Static and Variable Data

Iterative MapReduce was introduced to support high performance systems. It runs iterations of the map/reduce cycles. Data mining algorithms like K-means run numerous iterations. Static data such as data points in K-means does not change, while variable data can alter between each iteration. A naïve iterative MapReduce model can generate huge overhead owing to constantly referencing static data. This can be overcome with long-running map/reduce tasks that distinguish between static and variable data. You can also accelerate the intermediate data transfer or combine the output of all reduce tasks. Iterative MapReduce is shown in the Twister program, which uses the combine output method and determines at the end of every iteration whether to stop or continue with further iterations. The master node in Twister is the Twister Driver, and the slave nodes are Twister Daemons. Twister stores I/O data in partition files. Three MapReduce patterns in Twister: 1) Large input data, reduced in the end; 2) Data size is constant; 3) Data volume increases after MapReduce execution. Data Manipulation Tool handles data loading and uses metadata to keep track of data in partitions. Twister employs static scheduling. Fault tolerance is reserved for failures

that terminate running tasks. Static data can then be used to reassign the failed iterations. A list of Twister APIs is given.

[Static and Variable Data \(11:01\)](#)

[Static and Variable Data \(Page 10\)](#)

[Static and Variable Data - pptx \(Page 10\)](#)

36.10 MapReduce Model Comparison

This video showcases examples of work done comparing Twister results with Hadoop, MPI and DryadLINQ. The first is Map Only with CAP3 DNA Sequence Assembly, followed by Classic MapReduce with Pair-wise Sequences and High-Energy Physics, Iterative with K-means clustering, PageRank and Multi-dimensional Scaling, and finally Loosely Synchronous with Matrix Multiplication Algorithms. In all cases, Twister outperforms or is close to the competition.

[MapReduce Model Comparison \(6:56\)](#)

[MapReduce Model Comparison \(Page 24\)](#)

[MapReduce Model Comparison - pptx \(Page 24\)](#)

36.11 Twister K-means

Twister is applied to K-means Clustering. K-means develops a set number of clusters by creating cluster centers (centroids) that encompass the data points after successive proximity calculations. Parallelization of K-means is accomplished in the partitions, and the final centroids are determined in the Reduce step. A sample of K-means Clustering code follows, after which Twister is shown being used to determine centroids on K-means. Several questions are posed pertaining to the features of Twister. The results of a Twister K-means run are compared with those from a sequential run. Shown here, as the number of data points increases, Twister's runtimes get progressively faster. In a final set of runs against Hadoop, DryadLINQ, and MPI, Twister outperforms all but MPI.

[Twister K-means \(7:28\)](#)

[Twister K-means \(Page 34\)](#)

[Twister K-means - pptx \(Page 34\)](#)

36.12 Coding and Iterative Alternatives

A more detailed look is taken at the code used to run Twister K-means. MapReduce has many programs designed around its setup, including other iterative versions like Haloop, Pregel, and Spark. Twister can extend the use of traditional MapReduce to more complex applications.

[Coding and Iterative Alternatives \(5:14\)](#)

[Coding and Iterative Alternatives \(Page 43\)](#)

[Coding and Iterative Alternatives - pptx \(Page 43\)](#)

XIV Internet of Things

37	IoT	357
37.1	Everyday Data	
37.2	Streaming the Data Ocean	
37.3	Streams of Events	
37.4	Faults & Frameworks	
37.5	Spouts to Bolts	



37. IoT



[section/icloud/course/iot.tex](#)

37.1 Everyday Data

Ph.D. candidate Supun Kamburugamuva goes over the so-called Internet of Things as well as strategies and tools developed for Distributed Stream Processing.

[Everyday Data \(9:31\)](#)

[Everyday Data \(Page 4\)](#)

37.2 Streaming the Data Ocean

Ph.D. candidate Supun Kamburugamuva goes over the so-called Internet of Things as well as strategies and tools developed for Distributed Stream Processing.

[Streaming the Data Ocean \(9:38\)](#)

[Streaming the Data Ocean \(Page 6\)](#)

37.3 Streams of Events

Ph.D. candidate Supun Kamburugamuva goes over the so-called Internet of Things as well as strategies and tools developed for Distributed Stream Processing.

[Streams of Events \(10:44\)](#)

Streams of Events (Page 1) 

37.4 Faults & Frameworks

Ph.D. candidate Supun Kamburugamuva goes over the so-called Internet of Things as well as strategies and tools developed for Distributed Stream Processing.

Faults & Frameworks (7:46) 

Faults & Frameworks (Page 9) 

37.5 Spouts to Bolts

Ph.D. candidate Supun Kamburugamuva goes over the so-called Internet of Things as well as strategies and tools developed for Distributed Stream Processing.

Spouts to Bolts (8:42) 

Spouts to Bolts (Page 15) 



Big Data Applications

- 37.6 [Introduction](#)
- 37.7 [Overview of Data Science](#)
- 37.8 [Health Informatics Case Study](#)
- 37.9 [e-Commerce and LifeStyle Case Study](#)
- 37.10 [Physics Case Studies](#)
- 37.11 [Radar Case Study](#)
- 37.12 [Sensors Case Study](#)
- 37.13 [Sports Case Study](#)
- 37.14 [Big Data Use Cases Survey](#)
- 37.15 [Web Search and Text Mining](#)
- 37.16 [Technology Training - kNN & Clustering](#)

F part/applications.tex

37.6 Introduction

F section/theory/introduction.tex

Indiana University

You may find that some videos may have a different lesson, section or unit numbers. Please ignore this as we have significantly restructured the material and. In case the content does not correspond to the title, please let us know.

This Part of the Handbook includes broad overview for the motivation on why we study Big Data Applications. The information has been consolidated from the Web page hosted at

- <https://cloudmesh.github.io/classes/>

The overview covers it's content and structure. It presents an introduction to general field of Big Data and Analytics. We are especially analysing the many different application areas in which Big Data can be applied. As Big Data is typically not just used in isolation but is part of a larger Informatics issue for a particular field we also use the term X-Informatics, where X defines a usecase or area of specialization in which Big Data is applied to. As such we organize the material around the the *Rallying Cry*: Use Clouds running Data Analytics Collaboratively processing Big Data to solve problems in X-Informatics.

This part is set up as a number of lessons that are typically between 20 minutes to an hour. The lessons are either provided as written documents or as video lectures. They are enhanced by an in person meeting that takes place either in a lecture room for residential students or as online meeting for online students.

The part covers a mix of applications (the X in X-Informatics) and technologies needed to support the field electronically i.e. to process the application data. The overview ends with a discussion of the content at highest level. The material starts with a motivation summarizing clouds and data science, then units describing applications in areas such as Physics, e-Commerce, Web Search and Text mining, Health, Sensors and Remote Sensing). These are interspersed with discussions of infrastructure (clouds) and data analytics (algorithms like clustering and collaborative filtering used in applications). We use Python as primary programming language. We will be introducing practical use of cloud resources so that you have the oportunity to explore example analytics applications on smaller data sets that you define.

We start with striking examples of the data deluge with examples from research, business and the consumer. The growing number of jobs in data science is highlighted. He describes industry trend in both clouds and big data. Then the cloud computing model developed at amazing speed by industry is introduced. The 4 paradigms of scientific research are described with growing importance of data oriented version. He covers 3 major X-informatics areas: Physics, e-Commerce and Web Search followed by a broad discussion of cloud applications. Parallel computing in general and particular features of MapReduce are described.

We discuss the following topics which may be adapted as we see fit.

Writing Track:

- Writing a short review article
- Writing a project or term report

Theory Track:

- Motivation: Big Data and the Cloud; Centerpieces of the Future Economy
- Introduction: What is Big Data, Data Analytics
- Use Cases: Big Data Use Cases Survey
 - Use Case, Physics Discovery of Higgs Particle
 - Use Case: e-Commerce and Lifestyle with recommender systems
 - Use Case: Web Search and Text Mining and their technologies
 - Use Case: Sports
 - Use Case: Health
 - Use Case: Sensors
 - Use Case: Radar for Remote Sensing.
- Parallel Computing Overview and familiar examples
- Cloud Technology for Big Data Applications & Analytics

Practice Track:

- Python for Big Data Applications and Analytics: NumPy, SciPy, Matplotlib
- Using FutureGrid for Big Data Applications and Analytics
- Using Chameleon Cloud for Big Data Applications and Analytics
- (optional) Using Plotviz Software for Displaying Point Distributions in 3D
- Recommender Systems - K-Nearest Neighbors, Clustering and heuristic methods
- PageRank
- Kmeans
- MapReduce
- Kmeans and MapReduce Parallelism

37.6.1 Motivation

We motivate the study of X-informatics by describing data science and clouds. He starts with striking examples of the data deluge with examples from research, business and the consumer. The growing number of jobs in data science is highlighted. He describes industry trend in both clouds and big data.

He introduces the cloud computing model developed at amazing speed by industry. The 4 paradigms of scientific research are described with growing importance of data oriented version. He covers 3 major X-informatics areas: Physics, e-Commerce and Web Search followed by a broad discussion of cloud applications. Parallel computing in general and particular features of MapReduce are described. He comments on a data science education and the benefits of using MOOC's.

Emerging Technologies

This presents the overview of talk, some trends in computing and data and jobs. Gartner's emerging technology hype cycle shows many areas of Clouds and Big Data. We highlight 6 issues of importance: economic imperative, computing model, research model, Opportunities in advancing computing, Opportunities in X-Informatics, Data Science Education

[Motivation \(40:14\)](#)

[Motivation \(30\)](#)

Data Deluge

We give some amazing statistics for total storage; uploaded video and uploaded photos; the social media interactions every minute; aspects of the business big data tidal wave; monitors of aircraft engines; the science research data sizes from particle physics to astronomy and earth science; genes sequenced; and finally the long tail of science. The next slide emphasizes applications using algorithms on clouds. This leads to the rallying cry “Use Clouds running Data Analytics Collaboratively processing Big Data to solve problems in X-Informatics educated in data science” “with a catalog of the many values of X” Astronomy, Biology, Biomedicine, Business, Chemistry, Climate, Crisis, Earth Science, Energy, Environment, Finance, Health, Intelligence, Lifestyle, Marketing, Medicine, Pathology, Policy, Radar, Security, Sensor, Social, Sustainability, Wealth and Wellness”

[*Data Deluge \(30:38\)*](#)

[*Data Deluge \(20\)*](#)

Jobs

Jobs abound in clouds and data science. There are documented shortages in data science, computer science and the major tech companies advertise for new talent.

[*Jobs \(9:39\)*](#)

[*Jobs \(8\)*](#)

Industrial Trends

Trends include the growing importance of mobile devices and comparative decrease in desktop access, the export of internet content, the change in dominant client operating systems, use of social media, thriving Chinese internet companies.

[*Industrial Trends \(19:25\)*](#)

[*Industrial Trends \(16\)*](#)

[*Industrial Trends II \(16:54\)*](#)

[*Indusrial Trends II \(16\)*](#)

[*Indusrial Trends III \(30:13\)*](#)

[*Industrial Trends III \(21\)*](#)

Digital Disruption of Old Favorites

Not everything goes up. The rise of the Internet has led to declines in some traditional areas including Shopping malls and Postal Services.

[*Digital Distruption and transformation \(32:54\)*](#)

[*Digital Distruption and transformation \(28\)*](#)

Computing Model

Industry adopted clouds which are attractive for data analytics

Clouds and Big Data are transformational on a 2-5 year time scale. Already Amazon AWS is a lucrative business with almost a \$4B revenue. We describe the nature of cloud centers with economies of scale and gives examples of importance of virtualization in server consolidation. Then key characteristics of clouds are reviewed with expected high growth in Infrastructure, Platform and Software as a Service.

[Computing Model I \(24:03\)](#)

[Computing Model I \(14\)](#)

[Computing Model II \(28:18\)](#)

[Computing Model II \(27\)](#)

Research Model

4th Paradigm; From Theory to Data driven science?

We introduce the 4 paradigms of scientific research with the focus on the new fourth data driven methodology.

[Research Model \(7:33\)](#)

[Research Model \(4\)](#)

Data Science Process

We introduce the DIKW data to information to knowledge to wisdom paradigm. Data flows through cloud services transforming itself and emerging as new information to input into other transformations.

[Data Science Process \(15:42\)](#)

[Data Science Process \(10\)](#)

Physics-Informatics

Looking for Higgs Particle with Large Hadron Collider LHC

We look at important particle physics example where the Large hadron Collider has observed the Higgs Boson. He shows this discovery as a bump in a histogram; something that so amazed him 50 years ago that he got a PhD in this field. He left field partly due to the incredible size of author lists on papers.

[Physics-informatics \(13:27\)](#)

[Physics-inforamtics \(6\)](#)

Recommender Systems

Many important applications involve matching users, web pages, jobs, movies, books, events etc. These are all optimization problems with recommender systems one important way of performing this optimization. We go through the example of Netflix ~~ everything is a recommendation and muses about the power of viewing all sorts of things as items in a bag or more abstractly some space with funny properties.

-
- Recommender Systems I (12:21)* 
- Recommender Systems I (9)* 
- Recommender Systems II (9:44)* 
- Recommender Systems II (6)* 

Web Search and Information Retrieval

We look at Web Search and here we give an overview of the data analytics for web search, Pagerank as a method of ranking web pages returned and uses material from Yahoo on the subtle algorithms for dynamic personalized choice of material for web pages.

- Web Search and Information Retrieval (12:05)* 
- Web Search and Information Retrieval (6)* 

Cloud Application in Research

We describe scientific applications and how they map onto clouds, supercomputers, grids and high throughput systems. He likes the cloud use of the Internet of Things and gives examples.

- Cloud Applications in Research (33:51)* 
- Cloud Applications in Research (20)* 

Parallel Computing and MapReduce

We define MapReduce and gives a homely example from fruit blending.

- Computing and MapReduce (14:02)* 
- Computing and MapReduce (9)* 

Data Science Education

We discuss one reasons for ~~ Data Science as an educational initiative and aspects of its Indiana University implementation. Then general; features of online education are discussed with clear growth spearheaded by MOOC's where we use this material and others as an example. He stresses the choice between one class to 100,000 students or 2,000 classes to 50 students and an online library of MOOC lessons. In olden days he suggested "hermit's cage virtual university" ~~ gurus in isolated caves putting together exciting curricula outside the traditional university model. Grading and mentoring models and important online tools are discussed. Clouds have MOOC's describing them and MOOC's are stored in clouds; a pleasing symmetry.

- Data Science Education (28:08)* 
- Data Science Education (19)* 

Conclusions

The conclusions highlight clouds, data-intensive methodology, employment, data science, MOOC's and never forget the Big Data ecosystem in one sentence "Use Clouds running Data Analytics Collaboratively processing Big Data to solve problems in X-Informatics educated in data science"

[*Conclusions \(4:59\)*](#) [*Conclusions \(4\)*](#) 

Resources

- <http://www.gartner.com/technology/home.jsp> and many web links
- Meeker/Wu May 29 2013 Internet Trends D11 Conference <http://www.slideshare.net/kleinerperkins/kpcb-internet-trends-2013>
- <http://cs.metrostate.edu/~sbd/slides/Sun.pdf>
- Taming The Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics, Bill Franks Wiley ISBN: 978-1-118-20878-6
- Bill Ruh http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- <http://www.genome.gov/sequencingcosts/>
- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon
- <http://www.microsoft.com/en-us/news/features/2012/mar12/03-05CloudComputingJobs.aspx>
- http://www.mckinsey.com/mgi/publications/big_data/index.asp
- Tom Davenport http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- http://research.microsoft.com/en-us/people/barga/sc09_cloudcomp Tutorial.pdf
- http://research.microsoft.com/pubs/78813/AJ18_EN.pdf
- <http://www.google.com/green/pdfs/google-green-computing.pdf>
- <http://www.wired.com/wired/issue/16-07>
- <http://research.microsoft.com/en-us/collaboration/fourthparadigm/>
- Jeff Hammerbacher <http://berkeleydatascience.files.wordpress.com/2012/01/20120117berkeley1.pdf>
- <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf>
- <http://www.interactions.org/cms/?pid=1032811>
- <http://www.quantumdiaries.org/2012/09/07/why-particle-detectors-need-a-trigger/atlasmgg/>
- <http://www.sciencedirect.com/science/article/pii/S037026931200857X>
- <http://www.slideshare.net/xamat/building-largescale-realworld-recommender-systems-recsys2012-tutorial>
- http://www.ifi.uzh.ch/ce/teaching/spring2012/16-Recommender-Systems_Slides.pdf
- <http://en.wikipedia.org/wiki/PageRank>
- <http://pages.cs.wisc.edu/~beechung/icml11-tutorial/>
- <https://sites.google.com/site/opensourceiotcloud/>
- <http://datascience101.wordpress.com/2013/04/13/new-york-times-data-science-articles/>
- <http://blog.coursera.org/post/49750392396/on-the-topic-of-boredom>
- <http://x-informatics.appspot.com/course>
- <http://iucloudsummerschool.appspot.com/preview>
- https://www.youtube.com/watch?v=M3jcSCA9_hM

37.7 Overview of Data Science

 section/theory/overview.tex

What is Big Data, Data Analytics and X-Informatics?

We start with X-Informatics and its rallying cry. The growing number of jobs in data science is highlighted. The first unit offers a look at the phenomenon described as the Data Deluge starting with its broad features. Data science and the famous DIKW (Data to Information to Knowledge to Wisdom) pipeline are covered. Then more detail is given on the flood of data from Internet and Industry applications with eBay and General Electric discussed in most detail.

In the next unit, we continue the discussion of the data deluge with a focus on scientific research. He takes a first peek at data from the Large Hadron Collider considered later as physics Informatics and gives some biology examples. He discusses the implication of data for the scientific method which is changing with the data-intensive methodology joining observation, theory and simulation as basic methods. Two broad classes of data are the long tail of sciences: many users with individually modest data adding up to a lot; and a myriad of Internet connected devices – the Internet of Things.

We give an initial technical overview of cloud computing as pioneered by companies like Amazon, Google and Microsoft with new centers holding up to a million servers. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing with a comparison to supercomputing. Features of the data deluge are discussed with a salutary example where more data did better than more thought. Then comes Data science and one part of it ~ data analytics ~ the large algorithms that crunch the big data to give big wisdom. There are many ways to describe data science and several are discussed to give a good composite picture of this emerging field.

37.7.1 Data Science generics and Commercial Data Deluge

We start with X-Informatics and its rallying cry. The growing number of jobs in data science is highlighted. This unit offers a look at the phenomenon described as the Data Deluge starting with its broad features. Then he discusses data science and the famous DIKW (Data to Information to Knowledge to Wisdom) pipeline. Then more detail is given on the flood of data from Internet and Industry applications with eBay and General Electric discussed in most detail.

TBD (45) 

What is X-Informatics and its Motto

This discusses trends that are driven by and accompany Big data. We give some key terms including data, information, knowledge, wisdom, data analytics and data science. We discuss how clouds running Data Analytics Collaboratively processing Big Data can solve problems in X-Informatics. We list many values of X you can define in various activities across the world.

TBD (9:49) 

Jobs

Big data is especially important as there are some many related jobs. We illustrate this for both cloud computing and data science from reports by Microsoft and the McKinsey institute respectively. We show a plot from LinkedIn showing rapid increase in the number of data science and analytics jobs

as a function of time.

TBD (2:58) 

Data Deluge: General Structure

We look at some broad features of the data deluge starting with the size of data in various areas especially in science research. We give examples from real world of the importance of big data and illustrate how it is integrated into an enterprise IT architecture. We give some views as to what characterizes Big data and why data science is a science that is needed to interpret all the data.

TBD (13:04) 

Data Science: Process

We stress the DIKW pipeline: Data becomes information that becomes knowledge and then wisdom, policy and decisions. This pipeline is illustrated with Google maps and we show how complex the ecosystem of data, transformations (filters) and its derived forms is.

TBD (4:27) 

Data Deluge: Internet

We give examples of Big data from the Internet with Tweets, uploaded photos and an illustration of the vitality and size of many commodity applications.

TBD (3:42) 

Data Deluge: Business

We give examples including the Big data that enables wind farms, city transportation, telephone operations, machines with health monitors, the banking, manufacturing and retail industries both online and offline in shopping malls. We give examples from ebay showing how analytics allowing them to refine and improve the customer experiences.

TBD (6:00) 

TBD (7:34) 

TBD (9:37) 

Resources

- <http://www.microsoft.com/en-us/news/features/2012/mar12/03-05CloudComputingJobs.aspx>
- http://www.mckinsey.com/mgi/publications/big_data/index.asp
- Tom Davenport http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- Anjul Bhambhani http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- Jeff Hammerbacher <http://berkeleydatascience.files.wordpress.com/2012/01/20120117berkeley1.pdf>
- <http://www.economist.com/node/15579717>
- <http://cs.metrostate.edu/~sbd/slides/Sun.pdf>
- <http://jess3.com/geosocial-universe-2/>
- Bill Ruh http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html

- <http://www.hsph.harvard.edu/ncb2011/files/ncb2011-z03-rodriguez.pptx>
- Hugh Williams http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html

37.7.2 Data Deluge and Scientific Applications and Methodology

Overview

We continue the discussion of the data deluge with a focus on scientific research. He takes a first peek at data from the Large Hadron Collider considered later as physics Informatics and gives some biology examples. He discusses the implication of data for the scientific method which is changing with the data-intensive methodology joining observation, theory and simulation as basic methods. We discuss the long tail of sciences; many users with individually modest data adding up to a lot. The last lesson emphasizes how everyday devices ~~ the Internet of Things ~~ are being used to create a wealth of data.

TBD (22)  PDF

Science & Research

We look into more big data examples with a focus on science and research. We give astronomy, genomics, radiology, particle physics and discovery of Higgs particle (Covered in more detail in later lessons), European Bioinformatics Institute and contrast to Facebook and Walmart.

TBD (11:27) 

TBD (11:49) 

Implications for Scientific Method

We discuss the emergences of a new fourth methodology for scientific research based on data driven inquiry. We contrast this with third ~~ computation or simulation based discovery - methodology which emerged itself some 25 years ago.

TBD (5:07) 

Long Tail of Science

There is big science such as particle physics where a single experiment has 3000 people collaborate!. Then there are individual investigators who don't generate a lot of data each but together they add up to Big data.

TBD (2:10) 

Internet of Things

A final category of Big data comes from the Internet of Things where lots of small devices ~~ smart phones, web cams, video games collect and disseminate data and are controlled and coordinated in the cloud.

TBD (5:45) 

Resources

- <http://www.economist.com/node/15579717>

- Geoffrey Fox and Dennis Gannon Using Clouds for Technical Computing To be published in Proceedings of HPC 2012 Conference at Cetraro, Italy June 28 2012
- http://grids.ucs.indiana.edu/ptliupages/publications/Clouds_Technical_Computing_FoxGannonv2.pdf
- <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf>
- <http://www.genome.gov/sequencingcosts/>
- <http://www.quantumdiaries.org/2012/09/07/why-particle-detectors-need-a-trigger/atlasmgg>
- <http://salsahpc.indiana.edu/dlib/articles/00001935/>
- http://en.wikipedia.org/wiki/Simple_linear_regression
- <http://www.ebi.ac.uk/Information/Brochures/>
- <http://www.wired.com/wired/issue/16-07>
- <http://research.microsoft.com/en-us/collaboration/fourthparadigm/>
- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon <https://sites.google.com/site/opensourceiotcloud/>

37.7.3 Clouds and Big Data Processing; Data Science Process and Analytics

Overview

We give an initial technical overview of cloud computing as pioneered by companies like Amazon, Google and Microsoft with new centers holding up to a million servers. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing with a comparison to supercomputing.

He discusses features of the data deluge with a salutary example where more data did better than more thought. He introduces data science and one part of it ~~ data analytics ~~ the large algorithms that crunch the big data to give big wisdom. There are many ways to describe data science and several are discussed to give a good composite picture of this emerging field.

TBD (35)  PDF

37.7.4 Clouds

We describe cloud data centers with their staggering size with up to a million servers in a single data center and centers built modularly from shipping containers full of racks. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing and a comparison to supercomputing.

 TBD (16:04) MP4

Features of Data Deluge I

Data, Information, intelligence algorithms, infrastructure, data structure, semantics and knowledge are related. The semantic web and Big data are compared. We give an example where “More data usually beats better algorithms”. We discuss examples of intelligent big data and list 8 different types of data deluge

 TBD (8:02)

TBD (6:24) 

Data Science Process

We describe and critique one view of the work of a data scientist. Then we discuss and contrast 7 views of the process needed to speed data through the DIKW pipeline.

TBD (11:28) 

Data Analytics

TBD (30) 

We stress the importance of data analytics giving examples from several fields. We note that better analytics is as important as better computing and storage capability. In the second video we look at High Performance Computing in Science and Engineering: the Tree and the Fruit.

TBD (7:28) 

TBD (6:51) 

Resources

- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon
- Dan Reed Roger Barga Dennis Gannon Rich Wolski http://research.microsoft.com/en-us/people/barga/sc09_cloud.pdf
- <http://www.datacenterknowledge.com/archives/2011/05/10/uptime-institute-the-average-pue-is-1-8/>
- <http://loosebolts.wordpress.com/2008/12/02/our-vision-for-generation-4-modular-data-centers-one-way-of-getting-it-just-right/>
- <http://www.mediafire.com/file/zzqna34282frr2f/koomeydatacenterlectuse2011finalversion.pdf>
- Bina Ramamurthy <http://www.cse.buffalo.edu/~bina/cse487/fall2011/>
- Jeff Hammerbacher <http://berkeleydatascience.files.wordpress.com/2012/01/20120117berkeley1.pdf>
- Jeff Hammerbacher <http://berkeleydatascience.files.wordpress.com/2012/01/20120119berkeley.pdf>
- Anjul Bhambhani http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- <http://cs.metrostate.edu/~sbd/slides/Sun.pdf>
- Hugh Williams http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- Tom Davenport http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html
- http://www.mckinsey.com/mgi/publications/big_data/index.asp
- <http://cra.org/ccc/docs/nitrdsymposium/pdfs/keyes.pdf>

37.8 Health Informatics Case Study



section/theory/health.tex

This section starts by discussing general aspects of Big Data and Health including data sizes, different areas including genomics, EBI, radiology and the Quantified Self movement. We review

section/theory/health.tex

current state of health care and trends associated with it including increased use of Telemedicine. We summarize an industry survey by GE and Accenture and an impressive exemplar Cloud-based medicine system from Potsdam. We give some details of big data in medicine. Some remarks on Cloud computing and Health focus on security and privacy issues.

We survey an April 2013 McKinsey report on the Big Data revolution in US health care; a Microsoft report in this area and a European Union report on how Big Data will allow patient centered care in the future. Examples are given of the Internet of Things, which will have great impact on health including wearables. A study looks at 4 scenarios for healthcare in 2032. Two are positive, one middle of the road and one negative. The final topic is Genomics, Proteomics and Information Visualization.

37.8.1 X-Informatics Case Study: Health Informatics

Overview

[131 \(Health\)](#) 

This section starts by discussing general aspects of Big Data and Health including data sizes, different areas including genomics, EBI, radiology and the Quantified Self movement. We review current state of health care and trends associated with it including increased use of Telemedicine. We summarize an industry survey by GE and Accenture and an impressive exemplar Cloud-based medicine system from Potsdam. We give some details of big data in medicine. Some remarks on Cloud computing and Health focus on security and privacy issues.

We survey an April 2013 McKinsey report on the Big Data revolution in US health care; a Microsoft report in this area and a European Union report on how Big Data will allow patient centered care in the future. Examples are given of the Internet of Things, which will have great impact on health including wearables. A study looks at 4 scenarios for healthcare in 2032. Two are positive, one middle of the road and one negative. The final topic is Genomics, Proteomics and Information Visualization.

Big Data and Health

This lesson starts with general aspects of Big Data and Health including listing subareas where Big data important. Data sizes are given in radiology, genomics, personalized medicine, and the Quantified Self movement, with sizes and access to European Bioinformatics Institute.

[Big Data and Health \(10:02\)](#) 

Status of Healthcare Today

This covers trends of costs and type of healthcare with low cost genomes and an aging population. Social media and government Brain initiative.

[Status of Healthcare Today \(16:09\)](#) 

Telemedicine (Virtual Health)

This describes increasing use of telemedicine and how we tried and failed to do this in 1994.

[Telemedicine \(8:21\)](#) 

Big Data and Healthcare Industry

Summary of an industry survey by GE and Accenture.

Big Data and Healthcare Indusry (10:02) 

Medical Big Data in the Clouds

An impressive exemplar Cloud-based medicine system from Potsdam.

Medical Big Data in the Clouds (15:02) 

Medical image Big Data

Midical Image Big Data (6:33) 

Clouds and Health

Clouds and Health (4:35) 

McKinsey Report on the big-data revolution in US health care

This lesson covers 9 aspects of the McKinsey report. These are the convergence of multiple positive changes has created a tipping point for innovation; Primary data pools are at the heart of the big data revolution in healthcare; Big data is changing the paradigm: these are the value pathways; Applying early successes at scale could reduce US healthcare costs by \$300 billion to \$450 billion; Most new big-data applications target consumers and providers across pathways; Innovations are weighted towards influencing individual decision-making levers; Big data innovations use a range of public, acquired, and proprietary data types; Organizations implementing a big data transformation should provide the leadership required for the associated cultural transformation; Companies must develop a range of big data capabilities.

McKinsey Report (14:53) 

Microsoft Report on Big Data in Health

This lesson identifies data sources as Clinical Data, Pharma & Life Science Data, Patient & Consumer Data, Claims & Cost Data and Correlational Data. Three approaches are Live data feed, Advanced analytics and Social analytics.

Microsoft Report on Big Data in Health (2:26) 

EU Report on Redesigning health in Europe for 2020

This lesson summarizes an EU Report on Redesigning health in Europe for 2020. The power of data is seen as a lever for change in My Data, My decisions; Liberate the data; Connect up everything; Revolutionize health; and Include Everyone removing the current correlation between health and wealth.

EU Report on Redesigning health in Europe for 2020 (5:00) 

Medicine and the Internet of Things

The Internet of Things will have great impact on health including telemedicine and wearables. Examples are given.

Medicine and the Internet of Things (8:17) 

Extrapolating to 2032

A study looks at 4 scenarios for healthcare in 2032. Two are positive, one middle of the road and one negative.

Extrapolating to 2032 (15:13) 

Genomics, Proteomics and Information Visualization

A study of an Azure application with an Excel frontend and a cloud BLAST backend starts this lesson. This is followed by a big data analysis of personal genomics and an analysis of a typical DNA sequencing analytics pipeline. The Protein Sequence Universe is defined and used to motivate Multi dimensional Scaling MDS. Sammon's method is defined and its use illustrated by a metagenomics example. Subtleties in use of MDS include a monotonic mapping of the dissimilarity function. The application to the COG Proteomics dataset is discussed. We note that the MDS approach is related to the well known chisq method and some aspects of nonlinear minimization of chisq (Least Squares) are discussed.

Genomics, Proteomics and Information Visualization (6:56) 

CC) Genomics, Proteomics and Information Visualization (6:56) 

Next we continue the discussion of the COG Protein Universe introduced in the last lesson. It is shown how Proteomics clusters are clearly seen in the Universe browser. This motivates a side remark on different clustering methods applied to metagenomics. Then we discuss the Generative Topographic Map GTM method that can be used in dimension reduction when original data is in a metric space and is in this case faster than MDS as GTM computational complexity scales like N not N squared as seen in MDS.

Examples are given of GTM including an application to topic models in Information Retrieval. Indiana University has developed a deterministic annealing improvement of GTM. 3 separate clusterings are projected for visualization and show very different structure emphasizing the importance of visualizing results of data analytics. The final slide shows an application of MDS to generate and visualize phylogenetic trees.

Genomics, Proteomics and Information Visualization I (10:33) 

Genomics, Proteomics and Information Visualization: II (7:41) 

131 (Proteomics and Information Visualization) 

Resources

- <https://wiki.nci.nih.gov/display/CIP/CIP+Survey+of+Biomedical+Imaging+Archives>
- <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf>
- <http://www.ieee-icsc.org/ICSC2010/Tony%20Hey%20-%2020100923.pdf>
- <http://quantifiedself.com/larry-smarr/>
- <http://www.ebi.ac.uk/Information/Brochures/>
- <http://www.kpcb.com/internet-trends>
- <http://www.slideshare.net/drsteventucker/wearable-health-fitness-trackers-and-the-quantified-self>

- <http://www.siam.org/meetings/sdm13/sun.pdf>
- [http://en.wikipedia.org/wiki/Calico_\(company\)](http://en.wikipedia.org/wiki/Calico_(company))
- http://www.slideshare.net/GSW_Worldwide/2015-health-trends
- <http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Industrial-Internet-Changing-Competitive-Landscape-Industries.pdf>
- <http://www.slideshare.net/schappy/how-realtime-analysis-turns-big-medical-data-into-precision-medicine>
- <http://medcitynews.com/2013/03/the-body-in-bytes-medical-images-as-a-source-of-healthcare-big-data-infographic/>
- http://healthinformatics.wikispaces.com/file/view/cloud_computing.ppt
- <http://www.mckinsey.com/~/media/McKinsey/dotcom/Insights/Health%20care/The%20big-data%20revolution%20in%20US%20health%20care/The%20big-data%20revolution%20in%20US%20health%20care%20Accelerating%20value%20and%20innovation.ashx>
- <https://partner.microsoft.com/download/global/40193764>
- http://ec.europa.eu/information_society/activities/health/docs/policy/taskforce/redesigning_health-eu-for2020-ehtf-report2012.pdf
- <http://www.kpcb.com/internet-trends>
- <http://www.liveathos.com/apparel/app>
- <http://debategraph.org/Poster.aspx?aID=77>
- <http://www.oerc.ox.ac.uk/downloads/presentations-from-events/microsoftworkshop/gannon>
- <http://www.delsall.org>
- http://salsahpc.indiana.edu/millionseq/mina/16SrRNA_index.html
- <http://www.geatbx.com/docu/fcnindex-01.html>
- <https://wiki.nci.nih.gov/display/CIP/CIP+Survey+of+Biomedical+Imaging+Archives>
- <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf>
- <http://www.ieee-icsc.org/ICSC2010/Tony%20Hey%20-%202020100923.pdf>
- <http://quantifiedself.com/larry-smarr/>
- <http://www.ebi.ac.uk/Information/Brochures/>
- <http://www.kpcb.com/internet-trends>
- <http://www.slideshare.net/drsteventucker/wearable-health-fitness-trackers-and-the-quantified-self>
- <http://www.siam.org/meetings/sdm13/sun.pdf>
- [http://en.wikipedia.org/wiki/Calico_\(company\)](http://en.wikipedia.org/wiki/Calico_(company))
- http://www.slideshare.net/GSW_Worldwide/2015-health-trends
- <http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Industrial-Internet-Changing-Competitive-Landscape-Industries.pdf>
- <http://www.slideshare.net/schappy/how-realtime-analysis-turns-big-medical-data-into-precision-medicine>
- <http://medcitynews.com/2013/03/the-body-in-bytes-medical-images-as-a-source-of-healthcare-big-data-infographic/>
- http://healthinformatics.wikispaces.com/file/view/cloud_computing.ppt
- <http://www.mckinsey.com/~/media/McKinsey/dotcom/Insights/Health%20care/The%20big-data%20revolution%20in%20US%20health%20care/The%20big-data%20revolution%20in%20US%20health%20care%20Accelerating%20value%20and%20innovation.ashx>
- <https://partner.microsoft.com/download/global/40193764>
- http://ec.europa.eu/information_society/activities/health/docs/policy/taskforce/redesigning_health-eu-for2020-ehtf-report2012.pdf

- <http://www.kpcb.com/internet-trends>
- <http://www.liveathos.com/apparel/app>
- <http://debategraph.org/Poster.aspx?aID=77>
- <http://www.oerc.ox.ac.uk/downloads/presentations-from-events/microsoftworkshop/gannon>
- <http://www.delsall.org>
- http://salsahpc.indiana.edu/millionseq/mina/16SrRNA_index.html
- <http://www.geatbx.com/docu/fcnindex-01.html>

37.9 e-Commerce and LifeStyle Case Study

 section/theory/lifestyle.tex

Recommender systems operate under the hood of such widely recognized sites as Amazon, eBay, Monster and Netflix where everything is a recommendation. This involves a symbiotic relationship between vendor and buyer whereby the buyer provides the vendor with information about their preferences, while the vendor then offers recommendations tailored to match their needs. Kaggle competitions help improve the success of the Netflix and other recommender systems. Attention is paid to models that are used to compare how changes to the systems affect their overall performance. It is interesting that the humble ranking has become such a dominant driver of the world's economy. More examples of recommender systems are given from Google News, Retail stores and in depth Yahoo! covering the multi-faceted criteria used in deciding recommendations on web sites.

The formulation of recommendations in terms of points in a space or bag is given where bags of item properties, user properties, rankings and users are useful. Detail is given on basic principles behind recommender systems: user-based collaborative filtering, which uses similarities in user rankings to predict their interests, and the Pearson correlation, used to statistically quantify correlations between users viewed as points in a space of items. Items are viewed as points in a space of users in item-based collaborative filtering. The Cosine Similarity is introduced, the difference between implicit and explicit ratings and the k Nearest Neighbors algorithm. General features like the curse of dimensionality in high dimensions are discussed. A simple Python k Nearest Neighbor code and its application to an artificial data set in 3 dimensions is given. Results are visualized in Matplotlib in 2D and with Plotviz in 3D. The concept of a training and a testing set are introduced with training set pre labeled. Recommender system are used to discuss clustering with k-means based clustering methods used and their results examined in Plotviz. The original labelling is compared to clustering results and extension to 28 clusters given. General issues in clustering are discussed including local optima, the use of annealing to avoid this and value of heuristic algorithms.

37.9.1 Recommender Systems: Introduction

We introduce Recommender systems as an optimization technology used in a variety of applications and contexts online. They operate in the background of such widely recognized sites as Amazon, eBay, Monster and Netflix where everything is a recommendation. This involves a symbiotic relationship between vendor and buyer whereby the buyer provides the vendor with information about their preferences, while the vendor then offers recommendations tailored to match their needs, to the benefit of both.

There follows an exploration of the Kaggle competition site, other recommender systems and Netflix, as well as competitions held to improve the success of the Netflix recommender system.

Finally attention is paid to models that are used to compare how changes to the systems affect their overall performance. It is interesting how the humble ranking has become such a dominant driver of the world's economy.

[45 \(Recommender\)](#)  PDF

Recommender Systems as an Optimization Problem

We define a set of general recommender systems as matching of items to people or perhaps collections of items to collections of people where items can be other people, products in a store, movies, jobs, events, web pages etc. We present this as “yet another optimization problem”.

[Recommender Systems I \(8:06\)](#) 

Recommender Systems Introduction

We give a general discussion of recommender systems and point out that they are particularly valuable in long tail of tems (to be recommended) that aren't commonly known. We pose them as a rating system and relate them to information retrieval rating systems. We can contrast recommender systems based on user profile and context; the most familiar collaborative filtering of others ranking; item properties; knowledge and hybrid cases mixing some or all of these.

[Recommender Systems Introduction \(12:56\)](#) 

Kaggle Competitions

We look at Kaggle competitions with examples from web site. In particular we discuss an Irvine class project involving ranking jokes.

[Kaggle Competitions: \(3:36\)](#) 

Examples of Recommender Systems

We go through a list of 9 recommender systems from the same Irvine class.

[Examples of Recommender Systems \(1:00\)](#) 

Netflix on Recommender Systems

This is Part 1.

We summarize some interesting points from a tutorial from Netflix for whom “everything is a recommendation”. Rankings are given in multiple categories and categories that reflect user interests are especially important. Criteria used include explicit user preferences, implicit based on ratings and hybrid methods as well as freshness and diversity. Netflix tries to explain the rationale of its recommendations. We give some data on Netflix operations and some methods used in its recommender systems. We describe the famous Netflix Kaggle competition to improve its rating system. The analogy to maximizing click through rate is given and the objectives of optimization are given.

[Netflix on Recommender Systems \(14:20\)](#) 

Consumer Data Science

Here we go through Netflix's methodology in letting data speak for itself in optimizing the recommender engine. An example is given on choosing self produced movies. A/B testing is discussed with examples showing how testing does allow optimizing of sophisticated criteria. This lesson is concluded by comments on Netflix technology and the full spectrum of issues that are involved including user interface, data, AB testing, systems and architectures. We comment on optimizing for a household rather than optimizing for individuals in household.

Consumer Data Science (13:04) 

Resources

- <http://www.slideshare.net/xamat/building-largescale-realworld-recommender-systems-recsys2012-tutorial>
- http://www.ifi.uzh.ch/ce/teaching/spring2012/16-Recommender-Systems_Slides.pdf
- <https://www.kaggle.com/>
- http://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/CS77B_w12.html
- Jeff Hammerbacher <https://berkeleydatascience.files.wordpress.com/2012/01/20120117berkeley1.pdf>
- <http://www.techworld.com/news/apps/netflix-foretells-house-of-cards-success-with-cassandra-big-data-engine-3437514/>
- https://en.wikipedia.org/wiki/A/B_testing
- <http://www.infoq.com/presentations/Netflix-Architecture>

37.9.2 Recommender Systems: Examples and Algorithms

We continue the discussion of recommender systems and their use in e-commerce. More examples are given from Google News, Retail stores and in depth Yahoo! covering the multi-faceted criteria used in deciding recommendations on web sites. Then the formulation of recommendations in terms of points in a space or bag is given.

Here bags of item properties, user properties, rankings and users are useful. Then we go into detail on basic principles behind recommender systems: user-based collaborative filtering, which uses similarities in user rankings to predict their interests, and the Pearson correlation, used to statistically quantify correlations between users viewed as points in a space of items.

49 (Recommender)  PDF

Recap and Examples of Recommender Systems

We start with a quick recap of recommender systems from previous unit; what they are with brief examples.

Recap and Examples of Recommender Systems (5:48) 

Examples of Recommender Systems

We give 2 examples in more detail: namely Google News and Markdown in Retail.

Examples of Recommender Systems (8:34) 

Recommender Systems in Yahoo Use Case Example

We describe in greatest detail the methods used to optimize Yahoo web sites. There are two lessons discussing general approach and a third lesson examines a particular personalized Yahoo page with its different components. We point out the different criteria that must be blended in making decisions; these criteria include analysis of what user does after a particular page is clicked; is the user satisfied and cannot that we quantified by purchase decisions etc. We need to choose Articles, ads, modules, movies, users, updates, etc to optimize metrics such as relevance score, CTR, revenue, engagement. These lesson stress that if though we have big data, the recommender data is sparse. We discuss the approach that involves both batch (offline) and on-line (real time) components.

[*Recap of Recommender Systems II \(8:46\)*](#)

[*Recap of Recommender Systems III \(10:48\)*](#)

[*Case Study of Recommender systems \(3:21\)*](#)

User-based nearest-neighbor collaborative filtering

Collaborative filtering is a core approach to recommender systems. There is user-based and item-based collaborative filtering and here we discuss the user-based case. Here similarities in user rankings allow one to predict their interests, and typically this quantified by the Pearson correlation, used to statistically quantify correlations between users.

[*User-based nearest-neighbor collaborative filtering I \(7:20\)*](#)

[*User-based nearest-neighbor collaborative filtering II \(7:29\)*](#)

Vector Space Formulation of Recommender Systems

We go through recommender systems thinking of them as formulated in a funny vector space. This suggests using clustering to make recommendations.

[*Vector Space Formulation of Recommender Systems new \(9:06\)*](#)

Resources

- <http://pages.cs.wisc.edu/~beechung/icml11-tutorial/>

37.9.3 Item-based Collaborative Filtering and its Technologies

We move on to item-based collaborative filtering where items are viewed as points in a space of users. The Cosine Similarity is introduced, the difference between implicit and explicit ratings and the k Nearest Neighbors algorithm. General features like the curse of dimensionality in high dimensions are discussed.

[*18 \(Filtering\)*](#) PDF

Item-based Collaborative Filtering

We covered user-based collaborative filtering in the previous unit. Here we start by discussing memory-based real time and model based offline (batch) approaches. Now we look at item-based collaborative filtering where items are viewed in the space of users and the cosine measure is used to quantify distances. WE discuss optimizations and how batch processing can help. We discuss

different Likert ranking scales and issues with new items that do not have a significant number of rankings.

Item Based Filtering (11:18) 

k Nearest Neighbors and High Dimensional Spaces (7:16) 

k Nearest Neighbors and High Dimensional Spaces

We define the k Nearest Neighbor algorithms and present the Python software but do not use it. We give examples from Wikipedia and describe performance issues. This algorithm illustrates the curse of dimensionality. If items were real vectors in a low dimension space, there would be faster solution methods.

k Nearest Neighbors and High Dimensional Spaces (10:03) 

37.10 Physics Case Studies



section/theory/physics.tex

This section starts by describing the LHC accelerator at CERN and evidence found by the experiments suggesting existence of a Higgs Boson. The huge number of authors on a paper, remarks on histograms and Feynman diagrams is followed by an accelerator picture gallery. The next unit is devoted to Python experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals. Then random variables and some simple principles of statistics are introduced with explanation as to why they are relevant to Physics counting experiments. The unit introduces Gaussian (normal) distributions and explains why they seen so often in natural phenomena. Several Python illustrations are given. Random Numbers with their Generators and Seeds lead to a discussion of Binomial and Poisson Distribution. Monte-Carlo and accept-reject methods. The Central Limit Theorem concludes discussion.

37.10.1 Looking for Higgs Particles, Bumps in Histograms, Experiments and Accelerators (Part 1)

This unit is devoted to Python and Java experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals. The lectures use Python but use of Java is described.

20 (Higgs) 

Files: HiggsClassI-Sloping.py </files/python/physics/mr_higgs/higgs_classI_sloping.py>_

Looking for Higgs Particle and Counting Introduction

We return to particle case with slides used in introduction and stress that particles often manifested as bumps in histograms and those bumps need to be large enough to stand out from background in a statistically significant fashion.

Discovery of Higgs Particle (13:49) 

We give a few details on one LHC experiment ATLAS. Experimental physics papers have a staggering number of authors and quite big budgets. Feynman diagrams describe processes in a

fundamental fashion.

Looking for Higgs Particle and Counting Introduction II (7:38) 

Physics-Informatics Looking for Higgs Particle Experiments

We give a few details on one LHC experiment ATLAS. Experimental physics papers have a staggering number of authors and quite big budgets. Feynman diagrams describe processes in a fundamental fashion.

Looking for Higgs Particle Experiments (9:29) 

Accelerator Picture Gallery of Big Science

This lesson gives a small picture gallery of accelerators. Accelerators, detection chambers and magnets in tunnels and a large underground laboratory used for experiments where you need to be shielded from background like cosmic rays.

Accelerator Picture Gallery of Big Science (11:21) 

Resources

- <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf>
- <http://www.sciencedirect.com/science/article/pii/S037026931200857X>
- <http://www.nature.com/news/specials/lhc/interactive.html>

Looking for Higgs Particles: Python Event Counting for Signal and Background (Part 2)

This unit is devoted to Python experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals.

29 (Higgs II)  PDF

Files:

- HiggsClassI-Sloping.py </files/python/physics/mr_higgs/higgs_classI_sloping.py>
- HiggsClassIII.py </files/python/physics/number_theory/higgs_classIII.py>
- HiggsClassIIUniform.py </files/python/physics/mr_higgs/higgs_classII_uniform.py>

Physics Use Case II 1: Class Software

We discuss how this unit uses Java and Python on both a backend server (FutureGrid) or a local client. We point out useful book on Python for data analysis. This builds on technology training in Section 3.

Higgs Particle Events and Counting (9:30) 

Warning

This video contains Java information, but we are no longer using Java in this class.

Physics Use Case II 2: Event Counting

We define “event counting” data collection environments. We discuss the python and Java code to generate events according to a particular scenario (the important idea of Monte Carlo data). Here a sloping background plus either a Higgs particle generated similarly to LHC observation or one

observed with better resolution (smaller measurement error).

Event Counting (7:02) 

Physics Use Case II 3: With Python examples of Signal plus Background

This uses Monte Carlo data both to generate data like the experimental observations and explore effect of changing amount of data and changing measurement resolution for Higgs.

With Python examples of Signal plus Background (7:33) 

Physics Use Case II 4: Change shape of background & num of Higgs Particles

This lesson continues the examination of Monte Carlo data looking at effect of change in number of Higgs particles produced and in change in shape of background.

Change shape of background & num of Higgs Particles (7:01) 

Resources

- Python for Data Analysis: Agile Tools for Real World Data By Wes McKinney, Publisher: O'Reilly Media, Released: October 2012, Pages: 472.
- <http://jwork.org/scavis/api/>
- <https://en.wikipedia.org/wiki/DataMelt>

37.10.2 Looking for Higgs Particles: Random Variables, Physics and Normal Distributions

We introduce random variables and some simple principles of statistics and explains why they are relevant to Physics counting experiments. The unit introduces Gaussian (normal) distributions and explains why they seen so often in natural phenomena. Several Python illustrations are given. Java is currently not available in this unit.

39 (Higgs) 

HiggsClassIII.py </files/python/physics/number_theory/higgs_classIII.py>

Statistics Overview and Fundamental Idea: Random Variables

We go through the many different areas of statistics covered in the Physics unit. We define the statistics concept of a random variable.

Random variables and normal distributions (8:19) 

Physics and Random Variables

We describe the DIKW pipeline for the analysis of this type of physics experiment and go through details of analysis pipeline for the LHC ATLAS experiment. We give examples of event displays showing the final state particles seen in a few events. We illustrate how physicists decide what's going on with a plot of expected Higgs production experimental cross sections (probabilities) for signal and background.

Physics and Random Variables I (8:34) 

Physics and Random Variables II (5:50) 

Statistics of Events with Normal Distributions

We introduce Poisson and Binomial distributions and define independent identically distributed (IID) random variables. We give the law of large numbers defining the errors in counting and leading to Gaussian distributions for many things. We demonstrate this in Python experiments.

Statistics of Events with Normal Distributions (11:25) 

Gaussian Distributions

We introduce the Gaussian distribution and give Python examples of the fluctuations in counting Gaussian distributions.

Gaussian Distributions (9:08) 

Using Statistics

We discuss the significance of a standard deviation and role of biases and insufficient statistics with a Python example in getting incorrect answers.

Using Statistics (14:02) 

Resources

- <http://indico.cern.ch/event/20453/session/6/contribution/15?materialId=slides>
- <http://www.atlas.ch/photos/events.html>
- <https://cms.cern/>

37.10.3 Looking for Higgs Particles: Random Numbers, Distributions and Central Limit Theorem (Part 3)

We discuss Random Numbers with their Generators and Seeds. It introduces Binomial and Poisson Distribution. Monte-Carlo and accept-reject methods are discussed. The Central Limit Theorem and Bayes law concludes discussion. Python and Java (for student - not reviewed in class) examples and Physics applications are given.

44 (Higgs III)  PDF

Files:

TODO: HiggsClassIII.py: /files/python/physics/calculated_dice_roll/higgs_classIV_seeds.py

Generators and Seeds

We define random numbers and describe how to generate them on the computer giving Python examples. We define the seed used to define to specify how to start generation.

Higgs Particle Counting Errors (6:28) 

Generators and Seeds II (7:10) 

Binomial Distribution

We define binomial distribution and give LHC data as an example of where this distribution valid.

[Binomial Distribution: \(12:38\)](#) 

Accept-Reject

We introduce an advanced method **accept/reject** for generating random variables with arbitrary distributions.

[Accept-Reject \(5:54\)](#) 

Monte Carlo Method

We define Monte Carlo method which usually uses accept/reject method in typical case for distribution.

[Monte Carlo Method \(2:23\)](#) 

Poisson Distribution

We extend the Binomial to the Poisson distribution and give a set of amusing examples from Wikipedia.

[Poisson Distribution \(4:37\)](#) 

Central Limit Theorem

We introduce Central Limit Theorem and give examples from Wikipedia.

[Central Limit Theorem \(4:47\)](#) 

Interpretation of Probability: Bayes v. Frequency

This lesson describes difference between Bayes and frequency views of probability. Bayes's law of conditional probability is derived and applied to Higgs example to enable information about Higgs from multiple channels and multiple experiments to be accumulated.

[Interpretation of Probability \(12:39\)](#) 

Resources

TODO: integrate physics-references.bib

37.10.4 SKA Square Kilometer Array

Professor Diamond, accompanied by Dr. Rosie Bolton from the SKA Regional Centre Project gave a presentation at SC17 “into the deepest reaches of the observable universe as they describe the SKA’s international partnership that will map and study the entire sky in greater detail than ever before.”

- <http://sc17.supercomputing.org/presentation/?id=inspkr101&sess=sess263>

A summary article about this effort is available at:

- <https://www.hpcwire.com/2017/11/17/sc17-keynote-hpc-powers-ska-efforts-peer-deep-cosmos/>

The video is hosted at

- <http://sc17.supercomputing.org/presentation/?id=inspkr101&sess=sess263>

Start at about 1:03:00 (e.g. the one hour mark)

37.11 Radar Case Study

 section/theory/radar.tex

The changing global climate is suspected to have long-term effects on much of the world's inhabitants. Among the various effects, the rising sea level will directly affect many people living in low-lying coastal regions. While the ocean's thermal expansion has been the dominant contributor to rises in sea level, the potential contribution of discharges from the polar ice sheets in Greenland and Antarctica may provide a more significant threat due to the unpredictable response to the changing climate. The Radar-Informatics unit provides a glimpse in the processes fueling global climate change and explains what methods are used for ice data acquisitions and analysis.

58 (Radar) 

37.11.1 Introduction

This lesson motivates radar-informatics by building on previous discussions on why X-applications are growing in data size and why analytics are necessary for acquiring knowledge from large data. The lesson details three mosaics of a changing Greenland ice sheet and provides a concise overview to subsequent lessons by detailing explaining how other remote sensing technologies, such as the radar, can be used to sound the polar ice sheets and what we are doing with radar images to extract knowledge to be incorporated into numerical models.

Radar Informatics (3:31) 

37.11.2 Remote Sensing

This lesson explains the basics of remote sensing, the characteristics of remote sensors and remote sensing applications. Emphasis is on image acquisition and data collection in the electromagnetic spectrum.

Remote Sensing (6:43) 

37.11.3 Ice Sheet Science

This lesson provides a brief understanding on why melt water at the base of the ice sheet can be detrimental and why it's important for sensors to sound the bedrock.

Ice Sheet Science (1:00) 

37.11.4 Global Climate Change

This lesson provides an understanding and the processes for the greenhouse effect, how warming effects the Polar Regions, and the implications of a rise in sea level.

Global Climate Change (2:51) 

37.11.5 Radio Overview

This lesson provides an elementary introduction to radar and its importance to remote sensing, especially to acquiring information about Greenland and Antarctica.

[Radio Overview \(4:16\)](#)

37.11.6 Radio Informatics

This lesson focuses on the use of sophisticated computer vision algorithms, such as active contours and a hidden markov model to support data analysis for extracting layers, so ice sheet models can accurately forecast future changes in climate.

[Radio Informatics \(3:35\)](#)

37.12 Sensors Case Study

[section/theory/sensor.tex](#)

We start with the Internet of Things IoT giving examples like monitors of machine operation, QR codes, surveillance cameras, scientific sensors, drones and self driving cars and more generally transportation systems. We give examples of robots and drones. We introduce the Industrial Internet of Things IIoT and summarize surveys and expectations Industry wide. We give examples from General Electric. Sensor clouds control the many small distributed devices of IoT and IIoT. More detail is given for radar data gathered by sensors; ubiquitous or smart cities and homes including U-Korea; and finally the smart electric grid.

[31 \(Sensor I\)](#)

[44 \(Sensor II\)](#)

37.12.1 Internet of Things

There are predicted to be 24-50 Billion devices on the Internet by 2020; these are typically some sort of sensor defined as any source or sink of time series data. Sensors include smartphones, webcams, monitors of machine operation, barcodes, surveillance cameras, scientific sensors (especially in earth and environmental science), drones and self driving cars and more generally transportation systems. The lesson gives many examples of distributed sensors, which form a Grid that is controlled by a cloud.

[Internet of Things \(12:36\)](#)

37.12.2 Robotics and IOT Expectations

Examples of Robots and Drones.

[Robotics and IoT Expectations \(8:05\)](#)

37.12.3 Industrial Internet of Things

We summarize surveys and expectations Industry wide.

[section/theory/sensor.tex](#)

Industrial Internet of Things (1:24:02) 

37.12.4 Sensor Clouds

We describe the architecture of a Sensor Cloud control environment and gives example of interface to an older version of it. The performance of system is measured in terms of processing latency as a function of number of involved sensors with each delivering data at 1.8 Mbps rate.

Sensor Clouds (4:40) 

37.12.5 Earth/Environment/Polar Science data gathered by Sensors

This lesson gives examples of some sensors in the Earth/Environment/Polar Science field. It starts with material from the CReSIS polar remote sensing project and then looks at the NSF Ocean Observing Initiative and NASA's MODIS or Moderate Resolution Imaging Spectroradiometer instrument on a satellite.

Earth/Environment/Polar Science data gathered by Sensors (4:58) 

37.12.6 Ubiquitous/Smart Cities

For Ubiquitous/Smart cities we give two examples: Iniquitous Korea and smart electrical grids.

Ubiquitous/Smart Cities (1:44) 

37.12.7 U-Korea (U=Ubiquitous)

Korea has an interesting positioning where it is first worldwide in broadband access per capita, e-government, scientific literacy and total working hours. However it is far down in measures like quality of life and GDP. U-Korea aims to improve the latter by Pervasive computing, everywhere, anytime i.e. by spreading sensors everywhere. The example of a 'High-Tech Utopia' New Songdo is given.

U-Korea (U=Ubiquitous) (2:49) 

37.12.8 Smart Grid

The electrical Smart Grid aims to enhance USA's aging electrical infrastructure by pervasive deployment of sensors and the integration of their measurement in a cloud or equivalent server infrastructure. A variety of new instruments include smart meters, power monitors, and measures of solar irradiance, wind speed, and temperature. One goal is autonomous local power units where good use is made of waste heat.

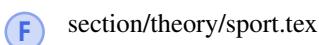
Smart Grid (6:04) 

37.12.9 Resources

- <https://www.gesoftware.com/minds-and-machines>
- <https://www.gesoftware.com/predix>
- <https://www.gesoftware.com/sites/default/files/the-industrial-internet/index.html>

- <https://developer.cisco.com/site/eiot/discover/overview/>
- <http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Industrial-Internet-Changing-Competitive-Landscape-Industries.pdf>
- <http://www.gesoftware.com/ge-predictivity-infographic>
- <http://www.getransportation.com/railconnect360/rail-landscape>
- <http://www.gesoftware.com/sites/default/files/GE-Software-Modernizing-Machine-to-Machine-Interactions.pdf>

37.13 Sports Case Study



Sports sees significant growth in analytics with pervasive statistics shifting to more sophisticated measures. We start with baseball as game is built around segments dominated by individuals where detailed (video/image) achievement measures including PITCHf/x and FIELDf/x are moving field into big data arena. There are interesting relationships between the economics of sports and big data analytics. We look at Wearables and consumer sports/recreation. The importance of spatial visualization is discussed. We look at other Sports: Soccer, Olympics, NFL Football, Basketball, Tennis and Horse Racing.

37.13.1 Sports Informatics I : Sabermetrics (Basic)

Unit Overview

This unit discusses baseball starting with the movie Moneyball and the 2002-2003 Oakland Athletics. Unlike sports like basketball and soccer, most baseball action is built around individuals often interacting in pairs. This is much easier to quantify than many player phenomena in other sports. We discuss Performance-Dollar relationship including new stadiums and media/advertising. We look at classic baseball averages and sophisticated measures like Wins Above Replacement.

[40 \(Overview\)](#)

Introduction and Sabermetrics (Baseball Informatics) Lesson

Introduction to all Sports Informatics, Moneyball The 2002-2003 Oakland Athletics, Diamond Dollars economic model of baseball, Performance - Dollar relationship, Value of a Win.

[Introduction and Sabermetrics \(Baseball Informatics\) Lesson \(31:4\)](#)

Basic Sabermetrics

Different Types of Baseball Data, Sabermetrics, Overview of all data, Details of some statistics based on basic data, OPS, wOBA, ERA, ERC, FIP, UZR.

[Basic Sabermetrics \(26:53\)](#)

Wins Above Replacement

Wins above Replacement WAR, Discussion of Calculation, Examples, Comparisons of different methods, Coefficient of Determination, Another, Sabermetrics Example, Summary of Sabermetrics.

Wins Above Replacement (30:43) 

Resources

- <http://www.slideshare.net/BrandEmotivity/sports-analytics-innovation-summit-data-powered-storytelling>
- <http://www.sloansportsconference.com/>
- <http://sabr.org/>
- <http://en.wikipedia.org/wiki/Sabermetrics>
- http://en.wikipedia.org/wiki/Baseball_statistics
- <http://www.sportvision.com/baseball>
- <http://m.mlb.com/news/article/68514514/mlbam-introduces-new-way-to-analyze-every-play>
- <http://www.fangraphs.com/library/offense/offensive-statistics-list/>
- http://en.wikipedia.org/wiki/Component_ERA
- <http://www.fangraphs.com/library/pitching/fip/>
- <http://nomaas.org/2012/05/a-look-at-the-defense-the-yankees-d-stinks-edition/>
- http://en.wikipedia.org/wiki/Wins_Above_Replacement
- <http://www.fangraphs.com/library/misc/war/>
- http://www.baseball-reference.com/about/war_explained.shtml
- http://www.baseball-reference.com/about/war_explained_comparison.shtml
- http://www.baseball-reference.com/about/war_explained_position.shtml
- http://www.baseball-reference.com/about/war_explained_pitch.shtml
- <http://www.fangraphs.com/leaders.aspx?pos=all&stats=bat&lg=all&qual=y&type=8&season=2014&month=0&season1=1871&ind=0>
- <http://battingleadoff.com/2014/01/08/comparing-the-three-war-measures-part-ii/>
- <http://battingleadoff.com/2014/01/08/comparing-the-three-war-measures-part-ii/>
- http://en.wikipedia.org/wiki/Coefficient_of_determination
- http://www.sloansportsconference.com/wp-content/uploads/2014/02/2014_SSAC_Data-driven-Method-for-In-game-Decision-Making.pdf
- <https://courses.edx.org/courses/BUx/SABR101x/2T2014/courseware/10e616fc7649469ab4457ae>

37.13.2 Sports Informatics II : Sabermetrics (Advanced)

This unit discusses ‘advanced sabermetrics’ covering advances possible from using video from PITCHf/X, FIELDf/X, HITf/X, COMMANDf/X and MLBAM.

41 (Sporta II) 

Pitching Clustering

A Big Data Pitcher Clustering method introduced by Vince Gennaro, Data from Blog and video at 2013 SABR conference.

Pitching Clustering (20:59) 

Pitcher Quality

Results of optimizing match ups, Data from video at 2013 SABR conference.

Pitcher Quality (10:02) 

37.13.3 PITCHf/X

Examples of use of PITCHf/X.

PITCHf/X (10:39) 

Other Video Data Gathering in Baseball

FIELDf/X, MLBAM, HITf/X, COMMANDf/X.

Other Video Data Gathering in Baseball (18:5) 

Resources

- <http://vincegennaro.mlblogs.com/>
- https://www.youtube.com/watch?v=H-kx-x_d0Mk
- <http://www.sportvision.com/media/pitchfx-how-it-works>
- <http://www.baseballprospectus.com/article.php?articleid=13109>
- <http://baseball.physics.illinois.edu/FastPFXGuide.pdf>
- <http://baseball.physics.illinois.edu/FieldFX-TDR-GregR.pdf>
- <http://www.sportvision.com/baseball/fieldfx>
- <http://regressing.deadspin.com/mlb-announces-revolutionary-new-fielding-tracking-syste-1534200504>
- <http://grantland.com/the-triangle/mlb-advanced-media-play-tracking-bob-bowman-interview/>
- <http://www.sportvision.com/baseball/hitfx>
- <https://www.youtube.com/watch?v=YkjtnuNmK74>

37.13.4 Sports Informatics III : Other Sports

We look at Wearables and consumer sports/recreation. The importance of spatial visualization is discussed. We look at other Sports: Soccer, Olympics, NFL Football, Basketball, Tennis and Horse Racing.

44 (Sports III) 

Wearables

Consumer Sports, Stake Holders, and Multiple Factors.

Wearables (22:2) 

Soccer and the Olympics

Soccer, Tracking Players and Balls, Olympics.

Soccer and the Olympics (8:28) 

Spatial Visualization in NFL and NBA

NFL, NBA, and Spatial Visualization.

Spatial Visualization in NFL and NBA (15:19) 

Tennis and Horse Racing

Tennis, Horse Racing, and Continued Emphasis on Spatial Visualization.

Video 8:52 Tennis and Horse Racing <https://www.youtube.com/watch?v=2P-pismFSrI>

Resources

- http://www.sloansportsconference.com/?page_id=481&sort_cate=Research%20Paper
- http://www.slideshare.net/Tricon_Infotech/big-data-for-big-sports
- <http://www.slideshare.net/BrandEmotivity/sports-analytics-innovation-summit-data-powered-storytelling>
- <http://www.liveathos.com/apparel/app>
- <http://www.slideshare.net/elew/sport-analytics-innovation>
- <http://www.wired.com/2013/02/catapult-smartball/>
- http://www.sloansportsconference.com/wp-content/uploads/2014/06/Automated_Playbook_Generation.pdf
- <http://autoscout.adsc.illinois.edu/publications/football-trajectory-dataset/>
- http://www.sloansportsconference.com/wp-content/uploads/2012/02/Goldsberry_Sloan_Submission.pdf
- <http://gamesetmap.com/>
- <http://www.trakus.com/technology.asp#tNetText>

37.14 Big Data Use Cases Survey



section/theory/usecases.tex

This section covers 51 values of X and an overall study of Big data that emerged from a NIST (National Institute for Standards and Technology) study of Big data. The section covers the NIST Big Data Public Working Group (NBD-PWG) Process and summarizes the work of five subgroups: Definitions and Taxonomies Subgroup, Reference Architecture Subgroup, Security and Privacy Subgroup, Technology Roadmap Subgroup and the Requirements and Use Case Subgroup. 51 use cases collected in this process are briefly discussed with a classification of the source of parallelism and the high and low level computational structure. We describe the key features of this classification.

37.14.1 Overview of NIST Big Data Public Working Group (NBD-PWG) Process and Results

This unit covers the NIST Big Data Public Working Group (NBD-PWG) Process and summarizes the work of five subgroups: Definitions and Taxonomies Subgroup, Reference Architecture Subgroup, Security and Privacy Subgroup, Technology Roadmap Subgroup and the Requirements and Use Case Subgroup. The work of latter is continued in next two units.

45 (Overview) 

section/theory/usecases.tex

Introduction to NIST Big Data Public Working Group (NBD-PWG) Process

The focus of the (NBD-PWG) is to form a community of interest from industry, academia, and government, with the goal of developing a consensus definitions, taxonomies, secure reference architectures, and technology roadmap. The aim is to create vendor-neutral, technology and infrastructure agnostic deliverables to enable big data stakeholders to pick-and-choose best analytics tools for their processing and visualization requirements on the most suitable computing platforms and clusters while allowing value-added from big data service providers and flow of data between the stakeholders in a cohesive and secure manner.

[Introduction \(13:02\)](#) 

Definitions and Taxonomies Subgroup

The focus is to gain a better understanding of the principles of Big Data. It is important to develop a consensus-based common language and vocabulary terms used in Big Data across stakeholders from industry, academia, and government. In addition, it is also critical to identify essential actors with roles and responsibility, and subdivide them into components and sub-components on how they interact/ relate with each other according to their similarities and differences.

For Definitions: Compile terms used from all stakeholders regarding the meaning of Big Data from various standard bodies, domain applications, and diversified operational environments.
For Taxonomies: Identify key actors with their roles and responsibilities from all stakeholders, categorize them into components and subcomponents based on their similarities and differences. In particular data Science and Big Data terms are discussed.

[Taxonomies \(7:42\)](#) 

Reference Architecture Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus-based approach to orchestrate vendor-neutral, technology and infrastructure agnostic for analytics tools and computing environments. The goal is to enable Big Data stakeholders to pick-and-choose technology-agnostic analytics tools for processing and visualization in any computing platform and cluster while allowing value-added from Big Data service providers and the flow of the data between the stakeholders in a cohesive and secure manner. Results include a reference architecture with well defined components and linkage as well as several exemplars.

[Architecture \(10:05\)](#) 

Security and Privacy Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus secure reference architecture to handle security and privacy issues across all stakeholders. This includes gaining an understanding of what standards are available or under development, as well as identifies which key organizations are working on these standards. The Top Ten Big Data Security and Privacy Challenges from the CSA (Cloud Security Alliance) BDWG are studied. Specialized use cases include Retail/Marketing, Modern Day Consumerism, Nielsen Homescan, Web Traffic Analysis, Healthcare, Health Information Exchange, Genetic Privacy, Pharma Clinical Trial Data Sharing, Cyber-security, Government, Military and Education.

[Security \(9:51\)](#) 

Technology Roadmap Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus vision with recommendations on how Big Data should move forward by performing a good gap analysis through the materials gathered from all other NBD subgroups. This includes setting standardization and adoption priorities through an understanding of what standards are available or under development as part of the recommendations. Tasks are gather input from NBD subgroups and study the taxonomies for the actors' roles and responsibility, use cases and requirements, and secure reference architecture; gain understanding of what standards are available or under development for Big Data; perform a thorough gap analysis and document the findings; identify what possible barriers may delay or prevent adoption of Big Data; and document vision and recommendations.

Technology (4:14) 

Interfaces subgroup

This subgroup is working on the following document: *NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interface*.

This document summarizes interfaces that are instrumental for the interaction with Clouds, Containers, and HPC systems to manage virtual clusters to support the NIST Big Data Reference Architecture (NBDRA). The Representational State Transfer (REST) paradigm is used to define these interfaces allowing easy integration and adoption by a wide variety of frameworks. . This volume, Volume 8, uses the work performed by the NBD-PWG to identify objects instrumental for the NIST Big Data Reference Architecture (NBDRA) which is introduced in the NBDIF: Volume 6, Reference Architecture.

This presentation was given at the *2nd NIST Big Data Public Working Group (NBD-PWG) Workshop* in Washington DC in June 2017. It explains our thoughts on deriving automatically a refernce architecture form the Refernce Architecture Interface specifications directly from the document.

The workshop Web page is located at

- <https://bigdatawg.nist.gov/workshop2.php>

The agenda of teh workshop is as follows:

- https://bigdatawg.nist.gov/2017_NIST_Big_Data_PWG_WorkshopAgenda_with_Speakers_Bio.pdf

The Web cas of the presentation is given bellow, while you need to fast forward to a particular time

- Webcast: Interface subgroup: <https://www.nist.gov/news-events/events/2017/06/2nd-nist-big-data-public-working-group-nbd-pwg-workshop>
 - see: Big Data Working Group Day 1, part 2 Time start: 21:00 min, Time end: 44:00
- Slides: <https://github.com/cloudmesh/cloudmesh.rest/blob/master/docs/NBDPWG-vol8.pptx?raw=true>
- Document: <https://github.com/cloudmesh/cloudmesh.rest/raw/master/docs/NIST.SP.1500-8-draft.pdf>

You are welcome to view other presentations if you are interested.

Requirements and Use Case Subgroup Introduction

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus list of Big Data requirements across all stakeholders. This includes gathering and understanding various use cases from diversified application domains. Tasks are to gather use case input from all stakeholders; derive Big Data requirements from each use case; analyze/prioritize a list of challenging general requirements that may delay or prevent adoption of Big Data deployment; develop a set of general patterns capturing the “essence” of use cases (not done yet) and work with Reference Architecture to validate requirements and reference architecture by explicitly implementing some patterns based on use cases. The progress of gathering use cases (discussed in next two units) and requirements systemization are discussed.

Requirements (27:28) 

37.14.2 51 Big Data Use Cases

This unit consists of one or more slides for each of the 51 use cases - typically additional (more than one) slides are associated with pictures. Each of the use cases is identified with source of parallelism and the high and low level computational structure. As each new classification topic is introduced we briefly discuss it but full discussion of topics is given in following unit.

100 (51) 

Government Use Cases

This covers Census 2010 and 2000 - Title 13 Big Data; National Archives and Records Administration Accession NARA, Search, Retrieve, Preservation; Statistical Survey Response Improvement (Adaptive Design) and Non-Traditional Data in Statistical Survey Response Improvement (Adaptive Design).

Government Use Cases (17:43) 

Commercial Use Cases

This covers Cloud Eco-System, for Financial Industries (Banking, Securities & Investments, Insurance) transacting business within the United States; Mendeley - An International Network of Research; Netflix Movie Service; Web Search; IaaS (Infrastructure as a Service) Big Data Business Continuity & Disaster Recovery (BC/DR) Within A Cloud Eco-System; Cargo Shipping; Materials Data for Manufacturing and Simulation driven Materials Genomics.

Commercial Use Cases (17:43) 

Defense Use Cases

This covers Large Scale Geospatial Analysis and Visualization; Object identification and tracking from Wide Area Large Format Imagery (WALF) Imagery or Full Motion Video (FMV) - Persistent Surveillance and Intelligence Data Processing and Analysis.

Defense Use Cases (15:43) 

Healthcare and Life Science Use Cases

This covers Electronic Medical Record (EMR) Data; Pathology Imaging/digital pathology; Computational Bioimaging; Genomic Measurements; Comparative analysis for metagenomes and genomes; Individualized Diabetes Management; Statistical Relational Artificial Intelligence for Health Care; World Population Scale Epidemiological Study; Social Contagion Modeling for Planning, Public Health and Disaster Management and Biodiversity and LifeWatch.

Healthcare and Life Science Use Cases (30:11) 

Deep Learning and Social Networks Use Cases

This covers Large-scale Deep Learning; Organizing large-scale, unstructured collections of consumer photos;Truthy: Information diffusion research from Twitter Data; Crowd Sourcing in the Humanities as Source for Bigand Dynamic Data; CINET: Cyberinfrastructure for Network (Graph) Science and Analytics and NIST Information Access Division analytic technology performance measurement, evaluations, and standards.

Deep Learning and Social Networks Use Cases (14:19) 

Research Ecosystem Use Cases

DataNet Federation Consortium DFC; The ‘Discinnet process’, metadata -big data global experiment; Semantic Graph-search on Scientific Chemical and Text-based Data and Light source beamlines.

Research Ecosystem Use Cases (9:09) 

Astronomy and Physics Use Cases

This covers Catalina Real-Time Transient Survey (CRTS): a digital, panoramic, synoptic sky survey; DOE Extreme Data from Cosmological Sky Survey and Simulations; Large Survey Data for Cosmology; Particle Physics: Analysis of LHC Large Hadron Collider Data: Discovery of Higgs particle and Belle II High Energy Physics Experiment.

Astronomy and Physics Use Cases (17:33) 

Environment, Earth and Polar Science Use Cases

EISCAT 3D incoherent scatter radar system; ENVRI, Common Operations of Environmental Research Infrastructure; Radar Data Analysis for CReSIS Remote Sensing of Ice Sheets; UAVSAR Data Processing, DataProduct Delivery, and Data Services; NASA LARC/GSFC iRODS Federation Testbed; MERRA Analytic Services MERRA/AS; Atmospheric Turbulence - Event Discovery and Predictive Analytics; Climate Studies using the Community Earth System Model at DOE’s NERSC center; DOE-BER Subsurface Biogeochemistry Scientific Focus Area and DOE-BER AmeriFlux and FLUXNET Networks.

Environment, Earth and Polar Science Use Cases (25:29) 

Energy Use Case

This covers Consumption forecasting in Smart Grids.

Energy Use Case (4:01) 

37.14.3 Features of 51 Big Data Use Cases

This unit discusses the categories used to classify the 51 use-cases. These categories include concepts used for parallelism and low and high level computational structure. The first lesson is an introduction to all categories and the further lessons give details of particular categories.

[43 \(Features\)](#)

Summary of Use Case Classification I

This discusses concepts used for parallelism and low and high level computational structure. Parallelism can be over People (users or subjects), Decision makers; Items such as Images, EMR, Sequences; observations, contents of online store; Sensors – Internet of Things; Events; (Complex) Nodes in a Graph; Simple nodes as in a learning network; Tweets, Blogs, Documents, Web Pages etc.; Files or data to be backed up, moved or assigned metadata; Particles/cells/mesh points. Low level computational types include PP (Pleasingly Parallel); MR (MapReduce); MRStat; MRIter (Iterative MapReduce); Graph; Fusion; MC (Monte Carlo) and Streaming. High level computational types include Classification; S/Q (Search and Query); Index; CF (Collaborative Filtering); ML (Machine Learning); EGO (Large Scale Optimizations); EM (Expectation maximization); GIS; HPC; Agents. Patterns include Classic Database; NoSQL; Basic processing of data as in backup or metadata; GIS; Host of Sensors processed on demand; Pleasingly parallel processing; HPC assimilated with observational data; Agent-based models; Multi-modal data fusion or Knowledge Management; Crowd Sourcing.

[Summary of Use Case Classification \(23:39\)](#)

Database(SQL) Use Case Classification

This discusses classic (SQL) database approach to data handling with Search&Query and Index features. Comparisons are made to NoSQL approaches.

[Database \(SQL\) Use Case Classification \(11:13\)](#)

NoSQL Use Case Classification

This discusses NoSQL (compared in previous lesson) with HDFS, Hadoop and Hbase. The Apache Big data stack is introduced and further details of comparison with SQL.

[NoSQL Use Case Classification \(11:20\)](#)

Use Case Classifications I

This discusses a subset of use case features: GIS, Sensors, the support of data analysis and fusion by streaming data between filters.

[Use Case Classifications I \(12:42\)](#)

Use Case Classifications II

This discusses a subset of use case features: Pleasingly parallel, MRStat, Data Assimilation, Crowd sourcing, Agents, data fusion and agents, EGO and security.

[Use Case Classifications II \(20:18\)](#)

Use Case Classifications III

This discusses a subset of use case features: Classification, Monte Carlo, Streaming, PP, MR, MRStat, MРИter and HPC(MPI), global and local analytics (machine learning), parallel computing, Expectation Maximization, graphs and Collaborative Filtering.

Use Case Classifications III (17:25) 

Resources

- NIST Big Data Public Working Group (NBD-PWG) Process <https://www.nist.gov/el/cyber-physical-systems/big-data-pwg>
- Big Data Definitions: <http://dx.doi.org/10.6028/NIST.SP.1500-1> (link is external)
- Big Data Taxonomies: <http://dx.doi.org/10.6028/NIST.SP.1500-2> (link is external)
- Big Data Use Cases and Requirements: <http://dx.doi.org/10.6028/NIST.SP.1500-3> (link is external)
- Big Data Security and Privacy: <http://dx.doi.org/10.6028/NIST.SP.1500-4> (link is external)
- Big Data Architecture White Paper Survey: <http://dx.doi.org/10.6028/NIST.SP.1500-5> (link is external)
- Big Data Reference Architecture: <http://dx.doi.org/10.6028/NIST.SP.1500-6> (link is external)
- Big Data Standards Roadmap: <http://dx.doi.org/10.6028/NIST.SP.1500-7> (link is external)

Some of the links bellow may be outdated. Please let us know the new links and notify us of the outdated links.

- DCGSA Standard Cloud: <https://www.youtube.com/watch?v=14Qi7T8zeg>
- On line 51 Use Cases <http://bigdatawg.nist.gov/usecases.php>
- Summary of Requirements Subgroup http://bigdatawg.nist.gov/_uploadfiles/M0245_v5_6066621242.docx
- Use Case 6 Mendeley <http://mendeley.com%20http://dev.mendeley.com>
- Use Case 7 Netflix <http://www.slideshare.net/xamat/building-largescale-realworld-recommender-systems-recsys2012-tutoria>
- Use Case 8 Search <http://www.slideshare.net/kleinerperkins/kpcb-internet-trends-2013>, http://webcourse.cs.technion.ac.il/236621/Winter2011-2012/en/ho_Lectures.html, <http://www.ifis.cs.tu-bs.de/teaching/ss-11/irws>, <http://www.slideshare.net/beechung/recommender-systems-tutorialpart1intro>, <http://www.worldwidewebsize.com/>
- Use Case 9 IaaS (Infrastructure as a Service) Big Data Business Continuity & Disaster Recovery (BC/DR) Within A Cloud Eco-System provided by Cloud Service Providers (CSPs) and Cloud Brokerage Service Providers (CBSPs) <http://www.disasterrecovery.org/>
- Use Case 11 and Use Case 12 Simulation driven Materials Genomics <https://www.materialsproject.org/>
- Use Case 13 Large Scale Geospatial Analysis and Visualization <http://www.opengeospatial.org/standards>, <http://geojson.org/>, <http://earth-info.nga.mil/publications/specs/printed/CADRG/cadrg.html>
- Use Case 14 Object identification and tracking from Wide Area Large Format Imagery (WALF) Imagery or Full Motion Video (FMV) - Persistent Surveillance <http://www.militaryaerospace.com/topics/m/video/79088650/persistent-surveillance-relies-on-extracting-relevant-data-points-and-connecting-the-dots.htm>, <http://>

www.defencetalk.com/wide-area-persistent-surveillance-revolutionizes-tactical-isr-45745/

- Use Case 15 Intelligence Data Processing and Analysis http://www.afcea-aberdeen.org/files/presentations/AFCEAAberdeen_DCGSA_COLWells_PS.pdf, http://stids.c4i.gmu.edu/papers/STIDSPapers/STIDS2012/_T14/_SmithEtAl/_HorizontalIntegrationOfWarfi.pdf, http://stids.c4i.gmu.edu/STIDS2011/papers/STIDS2011_CR_T1_SalmenEtAl.pdf, <https://www.youtube.com/watch?v=l4Qii7T8zeg>, <http://dcgsa.apg.army.mil/>
- Use Case 16 Electronic Medical Record (EMR) Data: Regenstrief Institute, Logical observation identifiers names and codes, Indiana Health Information Exchange, Institute of Medicine Learning Healthcare System
- Use Case 17 Pathology Imaging/digital pathology; <https://web.cci.emory.edu/confluence/display/PAIS> , <https://web.cci.emory.edu/confluence/display/HadoopGIS>
- Use Case 19 Genome in a Bottle Consortium: www.genomeinabottle.org
- Use Case 20 Comparative analysis for metagenomes and genomes <http://img.jgi.doe.gov/>
- Use Case 25 Biodiversity and LifeWatch
- Use Case 26 Deep Learning: Recent popular press coverage of deep learning technology: <http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html> , <http://www.nytimes.com/2012/06/26/technology/in-a-big-network-of-computers-evidence-of-machine-learning.html> , http://www.wired.com/2013/06/andrew_ng/,
A recent research paper on HPC for Deep Learning: http://www.stanford.edu/~acoates/papers/CoatesHuvalWangWuNgCatanzaro_icml2013.pdf, Widely-used tutorials and references for Deep Learning: http://ufldl.stanford.edu/wiki/index.php/Main_Page, <http://deeplearning.net/>
- Use Case 27 Organizing large-scale, unstructured collections of consumer photos <http://vision.soic.indiana.edu/projects/disco/>
- Use Case 28 Truthy: Information diffusion research from Twitter Data <http://truthy.indiana.edu/> , <http://cnets.indiana.edu/groups/nan/truthy/> , <http://cnets.indiana.edu/groups/nan/despic/>
- Use Case 30 CINET: Cyberinfrastructure for Network (Graph) Science and Analytics http://cinet.vbi.vt.edu/cinet_new/
- Use Case 31 NIST Information Access Division analytic technology performance measurement, evaluations, and standards <http://www.nist.gov/itl/iad/>
- Use Case 32 DataNet Federation Consortium DFC: The DataNet Federation Consortium, iRODS
- Use Case 33 The ‘Discinnet process’, metadata < - > big data global experiment <http://www.discinnet.org/>
- Use Case 34 Semantic Graph-search on Scientific Chemical and Text-based Data http://www.eurekalert.org/pub_releases/2013-07/aiop-ffm071813.php , <http://xpdb.nist.gov/chemblast/pdb.pl>
- Use Case 35 Light source beamlines <http://www-als.lbl.gov/> , <https://www1.aps.anl.gov/>
- Use Case 36 CRTS survey, CSS survey ; For an overview of the classification challenges, see, e.g., <http://arxiv.org/abs/1209.1681>
- Use Case 37 DOE Extreme Data from Cosmological Sky Survey and Simulations <http://www.lsst.org/lsst/> , <http://www.nersc.gov/> , <http://www.nersc.gov/assets/Uploads/HabibcosmosimV2.pdf>
- Use Case 38 Large Survey Data for Cosmology <http://desi.lbl.gov/> , <http://www.darkenergysurvey.org/>
- Use Case 39 Particle Physics: Analysis of LHC Large Hadron Collider Data: Discovery of Higgs particle <http://grids.ucs.indiana.edu/ptliupages/publications/Where%20the%20Higgs%20Particle%20is%20Produced.pdf>

- 20does%20all%20the%20data%20come%20from%20v7.pdf, http://www.es.net/assets/pubs_presos/High-throughput-lessons-from-the-LHC-experience.Johnston.TNC2013.pdf
- Use Case 40 Belle II High Energy Physics Experiment <http://belle2.kek.jp/>
 - Use Case 41 EISCAT 3D incoherent scatter radar system <https://www.eiscat3d.se/>
 - Use Case 42 ENVRI, Common Operations of Environmental Research Infrastructure, ENVRI Project website, ENVRI Reference Model, ENVRI deliverable D3.2 : Analysis of common requirements of Environmental Research Infrastructures, ICOS, Euro-Argo, EISCAT 3D, LifeWatch, EPOS, EMSO
 - Use Case 43 Radar Data Analysis for CReSIS Remote Sensing of Ice Sheets <https://www.cresis.ku.edu/>
 - Use Case 44 UAVSAR Data Processing, Data Product Delivery, and Data Services <http://uavasar.jpl.nasa.gov/>, <http://www.asf.alaska.edu/program/sdc>, <http://geo-gateway.org/main.html>
 - Use Case 47 Atmospheric Turbulence - Event Discovery and Predictive Analytics <http://oceanworld.tamu.edu/resources/oceanography-book/teleconnections.htm>, <http://www.forbes.com/sites/toddwoody/2012/03/21/meet-the-scientists-mining-big-data-to-predict-the-weather/>
 - Use Case 48 Climate Studies using the Community Earth System Model at DOE's NERSC center <http://www-pcmdi.llnl.gov/>, <http://www.nersc.gov/>, <http://science.energy.gov/ber/research/cesd/>, <http://www2.cisl.ucar.edu/>
 - Use Case 50 DOE-BER AmeriFlux and FLUXNET Networks <http://ameriflux.lbl.gov/>, <http://www.fluxdata.org/default.aspx>
 - Use Case 51 Consumption forecasting in Smart Grids <http://smartgrid.usc.edu/>, http://ganges.usc.edu/wiki/Smart_Grid, https://www.ladwp.com/ladwp/faces/ladwp/aboutus/a-power/a-p-smartgridla?_afrLoop=157401916661989&_afrWindowMode=0&_afrWindowId=null#%40%3F_afrWindowId%3Dnull%26_afrLoop%3D157401916661989%26_afrWindowMode%3D0%26_adf.ctrl-state%3Db7yulr4rl_17, <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6475927>

37.15 Web Search and Text Mining



section/theory/web.tex

This section starts with an overview of data mining and puts our study of classification, clustering and exploration methods in context. We examine the problem to be solved in web and text search and note the relevance of history with libraries, catalogs and concordances. An overview of web search is given describing the continued evolution of search engines and the relation to the field of Information.

The importance of recall, precision and diversity is discussed. The important Bag of Words model is introduced and both Boolean queries and the more general fuzzy indices. The important vector space model and revisiting the Cosine Similarity as a distance in this bag follows. The basic TF-IDF approach is discussed. Relevance is discussed with a probabilistic model while the distinction between Bayesian and frequency views of probability distribution completes this unit.

We start with an overview of the different steps (data analytics) in web search and then goes key steps in detail starting with document preparation. An inverted index is described and then how it is

prepared for web search. The Boolean and Vector Space approach to query processing follow. This is followed by Link Structure Analysis including Hubs, Authorities and PageRank. The application of PageRank ideas as reputation outside web search is covered. The web graph structure, crawling it and issues in web advertising and search follow. The use of clustering and topic models completes the section.

37.15.1 Web Search and Text Mining I

The unit starts with the web with its size, shape (coming from the mutual linkage of pages by URL's) and universal power laws for number of pages with particular number of URL's linking out or in to page. Information retrieval is introduced and compared to web search. A comparison is given between semantic searches as in databases and the full text search that is base of Web search. The origin of web search in libraries, catalogs and concordances is summarized. DIKW – Data Information Knowledge Wisdom – model for web search is discussed. Then features of documents, collections and the important Bag of Words representation. Queries are presented in context of an Information Retrieval architecture. The method of judging quality of results including recall, precision and diversity is described. A time line for evolution of search engines is given.

Boolean and Vector Space models for query including the cosine similarity are introduced. Web Crawlers are discussed and then the steps needed to analyze data from Web and produce a set of terms. Building and accessing an inverted index is followed by the importance of term specificity and how it is captured in TF-IDF. We note how frequencies are converted into belief and relevance.

[56 \(Web Search and Text Mining\)](#) 

37.15.2 Web and Document/Text Search: The Problem

[Text Mining \(9:56\)](#) 

This lesson starts with the web with its size, shape (coming from the mutual linkage of pages by URL's) and universal power laws for number of pages with particular number of URL's linking out or in to page.

37.15.3 Information Retrieval leading to Web Search

[Information Retrieval \(6:06\)](#) 

Information retrieval is introduced A comparison is given between semantic searches as in databases and the full text search that is base of Web search. The ACM classification illustrates potential complexity of ontologies. Some differences between web search and information retrieval are given.

37.15.4 History behind Web Search

[Web Search History \(5:48\)](#) 

The origin of web search in libraries, catalogs and concordances is summarized.

37.15.5 Key Fundamental Principles behind Web Search

[Principles \(9:30\)](#) 

This lesson describes the DIKW – Data Information Knowledge Wisdom – model for web search. Then it discusses documents, collections and the important Bag of Words representation.

37.15.6 Information Retrieval (Web Search) Components

Fundamental Principles of Web Search (5:06) 

This describes queries in context of an Information Retrieval architecture. The method of judging quality of results including recall, precision and diversity is described.

37.15.7 Search Engines

Search Engines (3:08) 

This short lesson describes a time line for evolution of search engines. The first web search approaches were directly built on Information retrieval but in 1998 the field was changed when Google was founded and showed the importance of URL structure as exemplified by PageRank.

37.15.8 Boolean and Vector Space Models

Boolean and Vector Space Model (6:17) 

This lesson describes the Boolean and Vector Space models for query including the cosine similarity.

37.15.9 Web crawling and Document Preparation

Web crawling and Document Preparation (4:55) 

This describes a Web Crawler and then the steps needed to analyze data from Web and produce a set of terms.

37.15.10 Indices

Indices (5:44) 

This lesson describes both building and accessing an inverted index. It describes how phrases are treated and gives details of query structure from some early logs.

37.15.11 TF-IDF and Probabilistic Models

TF-IDF and Probabilistic Models (3:57) 

It describes the importance of term specificity and how it is captured in TF-IDF. It notes how frequencies are converted into belief and relevance.

37.15.12 Resources

- http://saedsayad.com/data_mining_map.htm
- http://webcourse.cs.technion.ac.il/236621/Winter2011-2012/en/ho_Lectures.html

- The Web Graph: an Overview: <https://www.youtube.com/watch?v=yPFI6xFnDHE&feature=youtu.be>
- Jean-Loup Guillaume and Matthieu Latapy <https://hal.archives-ouvertes.fr/file/index/docid/54458/filename/webgraph.pdf>
- Constructing a reliable Web graph with information on browsing behavior, Yiqun Liu, Yufei Xue, Danqing Xu, Rongwei Cen, Min Zhang, Shaoping Ma, Liyun Ru <http://www.sciencedirect.com/science/article/pii/S0167923612001844>
- <http://www.ifis.cs.tu-bs.de/teaching/ss-11/irws>

37.15.13 Web Search and Text Mining II

[33 \(Text Mining\)](#)  PDF

We start with an overview of the different steps (data analytics) in web search. This is followed by Link Structure Analysis including Hubs, Authorities and PageRank. The application of PageRank ideas as reputation outside web search is covered. Issues in web advertising and search follow. This leads to emerging field of computational advertising. The use of clustering and topic models completes unit with Google News as an example.

37.15.14 Data Analytics for Web Search

[Web Search and Text Mining II \(6:11\)](#) 

This short lesson describes the different steps needed in web search including: Get the digital data (from web or from scanning); Crawl web; Preprocess data to get searchable things (words, positions); Form Inverted Index mapping words to documents; Rank relevance of documents with potentially sophisticated techniques; and integrate technology to support advertising and ways to allow or stop pages artificially enhancing relevance.

37.15.15 Link Structure Analysis including PageRank

[Realated Applications \(17:24\)](#) 

The value of links and the concepts of Hubs and Authorities are discussed. This leads to definition of PageRank with examples. Extensions of PageRank viewed as a reputation are discussed with journal rankings and university department rankings as examples. There are many extension of these ideas which are not discussed here although topic models are covered briefly in a later lesson.

37.15.16 Web Advertising and Search

[Web Advertising and Search \(9:02\)](#) 

Internet and mobile advertising is growing fast and can be personalized more than for traditional media. There are several advertising types Sponsored search, Contextual ads, Display ads and different models: Cost per viewing, cost per clicking and cost per action. This leads to emerging field of computational advertising.

37.15.17 Clustering and Topic Models

[Clustering and Topic Models \(6:21\)](#) 

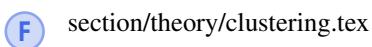
We discuss briefly approaches to defining groups of documents. We illustrate this for Google News

and give an example that this can give different answers from word-based analyses. We mention some work at Indiana University on a Latent Semantic Indexing model.

37.15.18 Resources

- <http://www.ifis.cs.tu-bs.de/teaching/ss-11/irws>
- <https://en.wikipedia.org/wiki/PageRank>
- http://webcourse.cs.technion.ac.il/236621/Winter2011-2012/en/ho_Lectures.html
- Meeker/Wu May 29 2013 Internet Trends D11 Conference <http://www.slideshare.net/kleinerperkins/kpcb-internet-trends-2013>

37.16 Technology Training - kNN & Clustering



section/theory/clustering.tex

This section is meant to provide a discussion on the kth Nearest Neighbor (kNN) algorithm and clustering using K-means. Python version for kNN is discussed in the video and instructions for both Java and Python are mentioned in the slides. Plotviz is used for generating 3D visualizations.

37.16.1 Recommender Systems - K-Nearest Neighbors

We discuss simple Python k Nearest Neighbor code and its application to an artificial data set in 3 dimensions. Results are visualized in Matplotlib in 2D and with Plotviz in 3D. The concept of training and testing sets are introduced with training set pre-labelled.

Files:

- kNN.py </files/python/knn/kNN.py>
- kNN_Driver.py </files/python/knn/kNN_Driver.py>
- DatingTesting2.txt </files/python/knn/dating_test_set2.txt>
- clusterFinal-M3-C3Dating-ReClustered.pviz </files/python/knn/clusterFinal-M3-C3Dating-ReClustered.pviz>
- DatingRating-OriginalLabels.pviz </files/python/knn/dating_rating_original_labels.pviz>
- clusterFinal-M30-C28.pviz </files/python/knn/clusterFinal-M30-C28.pviz>

Python k'th Nearest Neighbor Algorithms

This lesson considers the Python k Nearest Neighbor code found on the web associated with a book by Harrington on Machine Learning. There are two data sets. First we consider a set of 4 2D vectors divided into two categories (clusters) and use k=3 Nearest Neighbor algorithm to classify 3 test points. Second we consider a 3D dataset that has already been classified and show how to normalize. In this lesson we just use Matplotlib to give 2D plots.

3D Visualization

The lesson modifies the online code to allow it to produce files readable by PlotViz. We visualize already classified 3D set and rotate in 3D.

Testing k'th Nearest Neighbor Algorithms

The lesson goes through an example of using k NN classification algorithm by dividing dataset into 2 subsets. One is training set with initial classification; the other is test point to be classified by k=3 NN using training set. The code records fraction of points with a different classification from that input. One can experiment with different sizes of the two subsets. The Python implementation of algorithm is analyzed in detail.

37.16.2 Clustering and heuristic methods

We use example of recommender system to discuss clustering. The details of methods are not discussed but k-means based clustering methods are used and their results examined in Plotviz. The original labelling is compared to clustering results and extension to 28 clusters given. General issues in clustering are discussed including local optima, the use of annealing to avoid this and value of heuristic algorithms.

Files:

- Fungi_LSU_3_15_to_3_26_zeroidx.pviz </files/python/plotviz/fungi_lsu_3_15_to_3_26_zeroidx.pviz>
- DatingRating-OriginalLabels.pviz </files/python/plotviz/datingrating_originallabels.pviz>
- clusterFinal-M30-C28.pviz </files/python/plotviz/clusterFinal-M30-C28.pviz>
- clusterFinal-M3-C3Dating-ReClustered.pviz </files/python/plotviz/clusterfinal_m3_c3dating_reclustered.pviz>

Kmeans Clustering

We introduce the k means algorithm in a gentle fashion and describes its key features including dangers of local minima. A simple example from Wikipedia is examined.

Clustering of Recommender System Example

Plotviz is used to examine and compare the original classification with an “optimal” clustering into 3 clusters using a fancy deterministic annealing method that is similar to k means. The new clustering has centers marked.

Clustering of Recommender Example into more than 3 Clusters

The previous division into 3 clusters is compared into a clustering into 28 separate clusters that are naturally smaller in size and divide 3D space covered by 1000 points into compact geometrically local regions.

Local Optima in Clustering

This lesson introduces some general principles. First many important processes are “just” optimization problems. Most such problems are rife with local optima. The key idea behind annealing to avoid local optima is described. The pervasive greedy optimization method is described.

Clustering in General

The two different applications of clustering are described. First find geometrically distinct regions and secondly divide spaces into geometrically compact regions that may have no “thin air” between them. Generalizations such as mixture models and latent factor methods are just mentioned. The important distinction between applications in vector spaces and those where only inter-point distances are defined is described. Examples are then given using PlotViz from 2D clustering of a

mass spectrometry example and the results of clustering genomic data mapped into 3D with Multi Dimensional Scaling MDS.

Heuristics

Some remarks are given on heuristics; why are they so important why getting exact answers is often not so important?

Resources

- <https://en.wikipedia.org/wiki/Kmeans>
- http://grids.ucs.indiana.edu/ptliupages/publications/DACIDR_camera_ready_v0.3.pdf
- <http://salsahpc.indiana.edu/millionseq/>
- <http://salsafungiphy.blogspot.com/>
- <https://en.wikipedia.org/wiki/Heuristic>

Cloud and Big Data Technologies

38	Big Data Applications and Software	409
38.1	Overview	
38.2	Introduction	
38.3	Application Structure	
38.4	Application Aspects	
38.5	Applications	
38.6	Other	
39	Technology Overview	413
39.1	Workflow-Orchestration	
39.2	Application and Analytics	
39.3	Application Hosting Frameworks	
39.4	High level Programming	
39.5	Streams	
39.6	Basic Programming Model and Runtime, SPMD, MapReduce	
39.7	Inter process communication Collectives	
39.8	In-memory databases/caches	
39.9	Object-relational mapping	
39.10	Extraction Tools	
39.11	SQL and SQL Services	
39.12	NoSQL	
39.13	File management	
39.14	Data Transport	
39.15	Cluster Resource Management	
39.16	File systems	
39.17	Interoperability	
39.18	DevOps	
39.19	IaaS Management from HPC to hypervisors	
39.20	Cross-Cutting Functions	
39.21	Message and Data Protocols	
39.22	Technologies To Be Integrated	
39.23	Exercise	



38. Big Data Applications and Software

 part/i524.tex

38.1 Overview

- [Overview \(Pages 26\) !\[\]\(73380f5ef24a7ddb0890ea6276db12c0_img.jpg\)](#)
- [Part 1 \(11:29\) !\[\]\(5c9b4c94dcc060a1693f137ab727fd97_img.jpg\)](#)
- [Part 2 \(04:10\) !\[\]\(53247b0cd0e5360370aa176988731332_img.jpg\)](#)
- [Part 3 \(12:41\) !\[\]\(9415b4c516e244beb6203f9b51c91fb4_img.jpg\)](#)

38.2 Introduction

- [Course Introduction \(Pages 39\) !\[\]\(df2fdafa69700c4541c0f4a172187c84_img.jpg\)](#)
- [Introduction \(0:13:59\) !\[\]\(b9038a05021108a72d3a793155e97f4e_img.jpg\)](#)
- [Real World Big Data \(0:15:28\) !\[\]\(9ca6ab098bae5ae7a9b8fa65e02e4857_img.jpg\)](#)
- [Basic Trends and Jobs \(0:10:57\) !\[\]\(d37da12fe3acdb6de7e6f67d5b6d1c1a_img.jpg\)](#)

TODO: on box but not available, move to google drive

- [Access Patterns, Data Access Patterns and Introduction to using HPC-ABDS \(Pages ???\) !\[\]\(bd0f1c1f7fae3a740d8e071d220509fb_img.jpg\)](#)
- [A. Introduction to HPC-ABDS Software and Access Patterns \(0:27:45\) !\[\]\(1fde42e9eb94b7b853db52364fad810f_img.jpg\)](#)
- [B. Science Examples \(Data Access Patterns\) \(0:18:38\) !\[\]\(d752953617f15975e90adfb6dcd49f79_img.jpg\)](#)

- Resource 1 <http://grids.ucs.indiana.edu/ptliupages/publications/HPC-ABDSDescribedv2.pdf>
- Resource 2 <http://hpc-abds.org/kaleidoscope/>

- C. Remaining General Access Patterns (11:26)*
- D. Summary HPC-ABDS Layers 1 - 6 (14:32)*
- E. Summary HPC-ABDS Layers 7 - 13 (30:52)*
- F. Summary HPC-ABDS Layers 14 - 17 (28:02)*
- G. Summary HPC-ABDS Others (20:20)*

38.3 Application Structure

Big Data Application Structure, Slides <https://iu.box.com/s/zl71trvqfw6vv4wnc8xr6gf1qpc9a2dr>

- NIST Big Data Sub Groups (0:23:25)*
- Big Data Patterns - Sources of Parallelism (0:23:51)*
- First and Second Set of Features (0:18:26)*
- Machine Learning Aspect of Second Feature Set and the Third Set (0:18:38)*

38.4 Application Aspects

TODO: slides are on box and not google drive Aspects of Big Data Applications, Slides <https://iu.box.com/s/atgkxucop1lzftkunf8al2fe74x65na6>

- Other sources of use cases and Classical Databases/SQL Solutions (0:16:50)*
- SQL Solutions - Machine Learning Example - and MapReduce (0:18:49)*
- Clouds vs HPC - Data Intensive vs. Simulation Problems (0:20:26)*

38.5 Applications

TODO: slides are on box and not google drive Big Data Applications and Generalizing their Structure, Slides <https://iu.box.com/s/01dndtucmynekgehur00vktfgcppgc1r>

- NIST UseCases and Image Based Applications Examples I (0:25:20)*
- Image Based Applications II (0:15:23)*
- Internet of Things Based Applications (0:25:25)*
- Big Data Patterns - the Ogres and their Facets I (0:22:44)*
- Facets of the Big Data Ogres II (0:15:09)*

38.6 Other

More of Software Stack (0:24:00) 

 section/i524/technologies.tex



39. Technology Overview

In this section we find a number of technologies that are related to big data. Certainly a number of these projects are hosted as an Apache project. One important resource for a general list of all apache projects is at

Apache projects: <https://projects.apache.org/projects.html?category>

39.1 Workflow-Orchestration

39.1.1 ODE

Apache ODE (Orchestration Director Engine) is an open source implementation of the WS-BPEL 2.0 standard. WS- BPEL which stands for Web Services Business Process Execution Language, is an executable language for writing business processes with web services [102]. It includes control structures like conditions or loops as well as elements to invoke web services and receive messages from services. ODE uses WSDL (Web Services Description Language) for interfacing with web services [435]. Naming a few of its features, It supports two communication layers for interacting with the outside world, one based on Axis2 (Web Services http transport) and another one based on the JBI standard. It also supports both long and short living process executions for orchestrating services for applications [434].

39.1.2 ActiveBPEL

Business Process Execution Language for Web Services (BPEL4WS or just BPEL) is an XML-based grammar for describing the logic to coordinate and control web services that seamlessly integrate people, processes and systems, increasing the efficiency and visibility of the business. ActiveBPEL is a robust Java/J2EE runtime environment that is capable of executing process definitions created to the Business Process Execution Language for Web Services. The ActiveBPEL also provides an administration interface that is accessible via web service invocations;and it can

also be used to administer, to control and to integrate web services into a larger application [10].

39.1.3 Airavata

Apache Airavata [203] is a software framework that enables you to compose, manage, execute, and monitor large scale applications and workflows on distributed computing resources such as local clusters, supercomputers, computational grids, and computing clouds. Scientific gateway developers use Airavata as their middleware layer between job submissions and grid systems. Airavata supports long running applications and workflows on distributed computational resources. Many scientific gateways are already using Airavata to perform computations (e.g. Ultrascan [490], SEAGrid [306] and GenApp [202]).

39.1.4 Pegasus

The Pegasus [466] is workflow management system that allows to compose and execute a workflow in an application in different environment without the need for any modifications. It allows users to make high level workflow without thinking about the low level details. It locates the required input data and computational resources automatically. Pegasus also maintains information about tasks done and data produced. In case of errors Pegasus tries to recover by retrying the whole workflow and providing check pointing at workflow-level. It cleans up the storage as the workflow gets executed so that data-intensive workflows can have enough required space to execute on storage-constrained resources. Some of the other advantages of Pegasus are: scalability, reliability and high performance. Pegasus has been used in many scientific domains like astronomy, bioinformatics, earthquake science , ocean science, gravitational wave physics and others.

39.1.5 Kepler

Kepler, scientific workflow application, is designed to help scientist, analyst, and computer programmer create, execute and share models and analyses across a broad range of scientific and engineering disciplines. Kepler can operate on data stored in a variety of formats, locally and over the internet, and is an effective environment for integrating disparate software components such as merging *R* scripts with compiled *C* code, or facilitating remote, distributed execution of models. Using Kepler's GUI, users can simply select and then connect pertinent analytical components and data sources to create a *scientific workflow*. Overall, the Kepler helps users share and reuse data, workflow, and components developed by the scientific community to address common needs [347].

39.1.6 Swift

Swift is a general-purpose, multi-paradigm, compiled programming language. It has been developed by Apple Inc. for iOS, macOS, watchOS, tvOS, and Linux. This programming language is intended to be more robust and resilient to erroneous code than Objective-C, and more concise. It has been built with the LLVM compiler framework included in Xcode 6 and later and, on platforms other than Linux. C, Objective-C, C++ and Swift code can be run within one program as Swift uses the Objective-C runtime library [701].

Swift supports the core concepts that made Objective-C flexible, notably dynamic dispatch, widespread late binding, extensible programming and similar features. Swift features have well-known safety and performance trade-offs. A system that helps address common programming errors like null pointers was introduced to enhance safety. Apple has invested considerable effort in

aggressive optimization that can flatten out method calls and accessors to eliminate this overhead to handle performance issues.

39.1.7 Taverna

Taverna is workflow management system. According to [584], Taverna is transitioning to Apache Incubator as of Jan 2017. Taverna suite includes 2 products:

1. Taverna Workbench is desktop client where user can define the workflow.
2. Taverna Server is responsible for executing the remote workflows.

Taverna workflows can also be executed on command-line. Taverna supports wide range of services including WSDL-style and RESTful Web Services, BioMart, SoapLab, R, and Excel. Taverna also support mechanism to monitor the running workflows using its web browser interface. In the [621] paper, the formal syntax and operational semantics of Taverna is explained.

39.1.8 Triana

Triana is an open source problem solving software that comes with powerful data analysis tools [619]. Having been developed at Cardiff University, it has a good and easy-to-understand User Interface and is typically used for signal, text and image processing. Although it has its own set of analysis tools, it can also easily be integrated with custom tools. Some of the already available toolkits include signal-analysis toolkit, an image-manipulation toolkit, etc. Besides, it also checks the data types and reports the usage of any incompatible tools. It also reports errors, if any, as well as useful debug messages in order to resolve them. It also helps track serious bugs, so that the program does not crash. It has two modes of representing the data - a text-editor window or a graph-display window. The graph-display window has the added advantage of being able to zoom in on particular features. Triana is specially useful for automating the repetitive tasks, like finding-and-replacing a character or a string.

39.1.9 Trident

In [205], it is explained that Apache Trident is a “high-level abstraction for doing realtime computing on top of [Apache] Storm.” Similarly to Apache Storm, Apache Trident was developed by Twitter. Furthermore, [205] introduces Trident as a tool that “allows you to seamlessly intermix high throughput (millions of messages per second), stateful stream processing with low latency distributed querying.” In [204], the five kinds of operations in Trident are described as “Operations that apply locally to each partition and cause no network transfer”, “repartitioning operations that repartition a stream but otherwise don’t change the contents (involves network transfer)”, “aggregation operations that do network transfer as part of the operation”, “operations on grouped streams” and “merges and joins.” In [205], these five kinds of operations (i.e. joins, aggregations, grouping, functions, and filters) and the general concepts of Apache Trident are described as similar to “high level batch processing tools like Pig or Cascading.”

39.1.10 BioKepler

BioKepler is a Kepler module of scientific workflow components to execute a set of bioinformatics tools using distributed execution patterns [97]. It contains a specialized set of actors called “bioActors” for running bioinformatic tools, directors providing distributed data-parallel(DPP) execution on Big Data platforms such as Hadoop and Spark they are also configurable and reusable

[96]. BioKepler contains over 40 example workflows that demonstrate the actors and directors [372].

39.1.11 Galaxy

Ansible Galaxy is a website platform and command line tool that enables users to discover, create, and share community developed roles. Users' GitHub accounts are used for authentication, allowing users to import roles to share with the ansible community. [212] describes how Ansible roles are encapsulated and reusable tools for organizing automation content. Thus a role contains all tasks, variables, and handlers that are necessary to complete that role. [276] depicts roles as the most powerful part of Ansible as they keep playbooks simple and readable. "They provide reusable definitions that you can include whenever you need and customize with any variables that the role exposes." [213] provides the project documents for Ansible Galaxy on github.

39.1.12 Jupyter and IPython

The Jupyter Notebook is a language-agnostic HTML notebook web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text [489]. The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results [227]. The Jupyter notebook combines two components:

1. A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.
2. Notebook documents: a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects.

Notebooks may be exported to a range of static formats, including HTML (for example, for blog posts), reStructuredText, LaTeX, PDF, and slide shows, via the nbconvert command [228]. Notebook documents contains the inputs and outputs of a interactive session as well as additional text that accompanies the code but is not meant for execution [602]. In this way, notebook files can serve as a complete computational record of a session, interleaving executable code with explanatory text, mathematics, and rich representations of resulting objects [313]. These documents are internally JSON files and are saved with the .ipynb extension. Since JSON is a plain text format, they can be version-controlled and shared with colleagues [656].

39.1.13 Dryad

Dryad is a general-purpose distributed execution engine for coarse-grain data-parallel applications. According to [165] it was created with the objective of automatically managing scheduling, distribution, fault tolerance etc. Dryad concentrates on the throughput instead of latency and it assumes that a private data centre is used. It creates a dataflow graph by using computational 'vertices' and communication 'channels'. The computational vertices are written using C++ base classes and objects. During runtime, the dataflow graph is parallelized by distributing the vertices across multiple processor cores on the same computer or different physical computers connected by a network. The Dryad runtime handles this scheduling without any explicit intervention. The data flow from one vertex to another is realized by TCP/IP streams, shared memory, or temporary files. In the directed acyclic graph created by Dryad, each vertex is a program and the edges represent

data channels. Each graph is represented as $G = (VG, EG, IG, OG)$ in [166] where VG is a sequence of vertices with EG directed edges and two sets IG is a subset of VG and OG is a subset of VG that indicate the input and output vertices respectively. Other technologies used for the same purpose as Dryad include Map Reduce, MPI etc.

39.1.14 Naiad

Naiad [411] is a distributed system based on computational model called *Timely Dataflow* developed for execution of data-parallel, cyclic dataflow programs. It provides an in-memory distributed dataflow framework which exposes control over data partitioning and enables features like the high throughput of batch processors, the low latency of stream processors, and the ability to perform iterative and incremental computations. The Naiad architecture consists of two main components: (1) incremental processing of incoming updates and (2) low-latency real-time querying of the application state.

Compared to other systems supporting loops or streaming computation, Naiad provides support for the combination of the two, nesting loops inside streaming contexts and indeed other loops, while maintaining a clean separation between the many reasons new records may flow through the computation [609].

This model enriches dataflow computation with timestamps that represent logical points in the computation and provide the basis for an efficient, lightweight coordination mechanism. All the above capabilities in one package allows development of High-level programming models on Naiad which can perform tasks as streaming data analysis, iterative machine learning, and interactive graph mining. On the contrary, it's public reusable low-level programming abstractions leads Naiad to outperforms many other data parallel systems that enforce a single high-level programming model.

39.1.15 Oozie

Oozie is a workflow manager and scheduler. Oozie is designed to scale in a Hadoop cluster. Each job will be launched from a different datanode [317] [659]. Oozie [442] is architected from the ground up for large-scale Hadoop workflow. Scales to meet the demand, provides a multi-tenant service, is secure to protect data and processing, and can be operated cost effectively. As demand for workflow and the sophistication of applications increase, it must continue to mature in these areas [317]. Is well integrated with Hadoop security. Is the only workflow manager with built-in Hadoop actions, making workflow development, maintenance and troubleshooting easier. It's UI makes it easier to drill down to specific errors in the data nodes. Proven to scale in some of the world's largest clusters [317]. Gets callbacks from MapReduce jobs so it knows when they finish and whether they hang without expensive polling. Oozie Coordinator allows triggering actions when files arrive at HDFS. Also supported by Hadoop vendors [317].

39.1.16 Tez

Apache Tez is open source distributed execution framework build for writing native YARN application. It provides architecture which allows user to convert complex computation as dataflow graphs and the distributed engine to handle the directed acyclic graph for processing large amount of data. It is highly customizable and pluggable so that it can be used as a platform for various application. It is used by the Apache Hive, Pig as execution engine to increase the performance of map reduce functionality [590]. Tez focuses on running application efficiently on Hadoop

cluster leaving the end user to concentrate only on its business logic. Tez provides features like distributed parallel execution on hadoop cluster, horizontal scalability, resource elasticity, shared library reusable components and security features. Tez provides capability to naturally map the algorithm into the hadoop cluster execution engine and it also provides the interface for interaction with different data sources and configurations.

Tez is client side application and just needs Tez client to be pointed to Tez jar libraries path makes it easy and quick to deploy. User can have multiple tez version running concurrently. Tez provides DAG API's which lets user define structure for the computation and Runtime API's which contain the logic or code that needs to be executed in each transformation or task.

39.1.17 Google FlumeJava

FlumeJava [196] is a java library that allows users to develop and run data parallel pipelines. Its goal is to allow a programmer to express his data-parallel computations in a clear way while simultaneously executing it in the best possible optimized manner. The MapReduce function eases the task of data parallelism. However, a pipeline of MapReduce functions is desired by many real time computation systems. FlumeJava provides these abstractions of data parallel computations by providing support for pipelined execution. To provide optimized parallel execution, FlumeJava defers the execution of these pipelines and instead constructs an execution plan dataflow graph depending on the results needed by each stage of the pipeline. “When the final results of the parallel operations are eventually needed, FlumeJava first optimizes the execution plan, and then executes the optimized operations on appropriate underlying primitives” [113]. FlumeJava library is written on top of the collection framework in Java.

When developing a large pipeline, it is time consuming to find a bug in the later stages and then re-compile and re-evaluate all the operations. FlumeJava library supports a cached execution mode to aid in this scenario. In this mode, it automatically creates temporary files to hold the outputs of each operation it executes [113]. Thus, rather than recomputing all the operations once the pipeline has been rectified to fix all the bugs, it simply reads the output from these temporary files and later deletes them once they are no longer in use.

39.1.18 Crunch

Arvados Crunch [148] is a containerized workflow engine for running complex, multi-part pipelines or workflows in a way that is flexible, scalable, and supports versioning, reproducibility, and provenance while running in virtualized computing environments. The Arvados Crunch [147] framework is designed to support processing very large data batches (gigabytes to terabytes) efficiently. Arvados Crunch increases concurrency by running tasks asynchronously, using many CPUs and network interfaces at once (especially beneficial for CPU-bound and I/O-bound tasks respectively). Crunch also tracks inputs, outputs, and settings so you can verify that the inputs, settings, and sequence of programs you used to arrive at an output is really what you think it was. Crunch ensures that your programs and workflows are repeatable with different versions of your code, OS updates, etc. and allows you to interrupt and resume long-running jobs consisting of many short tasks and maintains timing statistics automatically.

39.1.19 Cascading

[109] Cascading software authored by Chris Wensel is development platform for building the application in Hadoop. It basically act as an abstraction for Apache Hadoop used for creating

complex data processing workflow using the scalability of hadoop however hiding the complexity of mapReduce jobs. User can write their program in java without having knowledge of mapReduce. Applications written on cascading are portable.

Cascading Benefits 1. With Cascading application can be scaled as per the data sets. 2. Easily Portable 3. Single jar file for application deployment.

39.1.20 Askalon

Askalan was developed at the University of Innsbruck [482]. It is application development as well as a runtime environment. It allows easy execution of distributed work flow applications in service oriented grids. It uses a Service Oriented Architecture. Also, for its Grid middleware it uses the Globus Toolkit. The work flow applications are developed using Abstract Grid Work flow Language (AGWL). The architecture has various components like the resource broker responsible for brokerage functions like management and reservation, information service for the discovery and organization of resources and data, metascheduler for mapping in the Grid, performance analysis for unification of performance monitoring and integration of the results and the Askalon scheduler.

The Metascheduler is of special significance since it consists of two major components - the workflow converter and the scheduling engine. The former is responsible for conversion of traditional workflows into directed acyclic graphs (DAGs) while the later one is responsible for the scheduling of workflows for various specific tasks. It has a conventional pluggable architecture which allows easy integration of various services. By default, the Heterogeneous Earliest Finish Time (HEFT) is used as the primary scheduling algorithm.

39.1.21 Scalding

39.1.22 e-Science Central

In [279], it is explained that e-Science Central is designed to address some of the pitfalls within current Infrastructure as a Service (e.g. Amazon EC2) and Platform as a Service (e.g. force.com) services. For instance, in [279], the “majority of potential scientific users, access to raw hardware is of little use as they lack the skills and resources needed to design, develop and maintain the robust, scalable applications they require” and furthermore “current platforms focus on services required for business applications, rather than those needed for scientific data storage and analysis.” In [75], it is explained that e-Science Central is a “cloud based platform for data analysis” which is “portable and can be run on Amazon AWS, Windows Azure or your own hardware.” In [279], e-Science Central is further described as a platform, which “provides both Software and Platform as a Service for scientific data management, analysis and collaboration.” This collaborative platform is designed to be scalable while also maintaining ease of use for scientists. In [279], “a project consisting of chemical modeling by cancer researchers” demonstrates how e-Science Central “allows scientists to upload data, edit and run workflows, and share results in the cloud.”

39.1.23 Azure Data Factory

Azure data factory is a cloud based data integration service that can ingest data from various sources, transform/ process data and publish the result data to the data stores. A data management gateway enables access to data on SQL Databases [537]. The data processing is done by It works by creating pipelines to transform the raw data into a format that can be readily used by BI Tools or applications. The services comes with rich visualization aids that aid data analysis. Data Factory supports two types of activities: data movement activities and data transformation activities. Data Movement

[540] is a Copy Activity in Data Factory that copies data from a data source to a Data sink. Data Factory supports the following data stores. Data from any source can be written to any sink. Data Transformation: Azure Data Factory supports the following transformation activities such as Map reduce, Hive transformations and Machine learning activities. Data factory is a great tool to analyze web data, sensor data and geo-spatial data.

39.1.24 Google Cloud Dataflow

Google Cloud Dataflow is a unified programming model and a managed service for developing and executing a wide variety of data processing patterns (pipelines). Dataflow includes SDKs for defining data processing workflows and a Cloud platform managed services to run those workflows on a Google cloud platform resources such as Compute Engine, BigQuery amongst others [237]. Dataflow pipelines can operate in both batch and streaming mode. The platform resources are provided on demand, allowing users to scale to meet their requirements, it's also optimized to help balance lagging work dynamically.

Being a cloud offering, Dataflow is designed to allow users to focus on devising proper analysis without worrying about the installation and maintaining [321] the underlying data piping and process infrastructure.

39.1.25 NiFi (NSA)

[64] Defines NiFi as “An Easy to use, powerful and reliable system to process and distribute data”. This tool aims at automated data flow from sources with different sizes , formats and following different protocols to the centralized location or destination [47].

This comes equipped with an easy use UI where the data flow can be controlled with a drag and a drop. NiFi was initially developed by NSA (called Niagarafiles) using the concepts of flow-based programming and latter submitted to Apache Software foundation [430].

39.1.26 Jitterbit

Jitterbit [331] is an integration tool that delivers a quick, flexible and simpler approach to design, configure, test, and deploy integration solutions. It delivers powerful, flexible, and easy to use integration solutions that connect modern on premise, cloud, social, and mobile infrastructures. Jitterbit employs high performance parallel processing algorithms to handle large data sets commonly found in ETL initiatives [330]. This allows easy synchronization of disparate computing platforms quickly. The Data Cleansing and Smart Reconstruction tools provides complete reliability in data extraction, transformation and loading.

Jitterbit employs a no-code GUI (graphical user interface) and work with diverse applications such as : ETL (extract-transform-load), SaaS (Software as a Service), SOA (service-oriented architecture).

Thus it provides centralized platform with power to control all data. It supports many document types and protocols: XML, web services, database, LDAP, text, FTP, HTTP(S), Flat and Hierarchic file structures and file shares [332]. It is available for Linux and Windows, and is also offered through Amazon EC2 (Amazon Elastic Compute Cloud). Jitterbit Data Loader for Salesforce is a free data migration tool that enables Salesforce administrators automated import and export of data between flat files, databases and Salesforce.

39.1.27 **Talend**

Talend is Apache Software Foundation sponsor Big data integration tool design to ease the development and integration and management of big data, Talend provides well optimized auto generated code to load transform, enrich and cleanse data inside Hadoop, where one don't need to learn write and maintain Hadoop and spark code. The product has 900+ build-in components feature data integration

Talend features multiple products that simplify the digital transformation tools such as Big data integration, Data integration, Data Quality, Data Preparation, Cloud Integration, Application Integration, Master Data management, Metadata Manager. Talend Integration cloud is secure and managed integration Platform-as-a-service (iPaaS), for connecting, cleansing and sharing cloud on premise data.

39.1.28 **Pentaho**

Pentaho is a business intelligence corporation that provides data mining, reporting, dashboarding and data integration capabilities. Generally, organizations tend to obtain meaningful relationships and useful information from the data present with them. Pentaho addresses the obstacles that obstruct them from doing so [467]. The platform includes a wide range of tools that analyze, explore, visualize and predict data easily which simplifies blending any data. The sole objective of pentaho is to translate data into value. Being an open and extensible source, pentaho provides big data tools to extract, prepare and blend any data [34]. Along with this, the visualizations and analytics will help in changing the path that the organizations follow to run their business. From spark and hadoop to noSQL, pentaho transforms big data into big insights.

39.1.29 **Apatar**

39.1.30 **Docker Compose**

Docker is an open-source container based technology. A container allows a developer to package up an application and all its part including the stack it runs on, dependencies it is associated with and everything the application requires to run within an isolated environment . Docker separates Application from the underlying Operating System in a similar way as Virtual Machines separates the Operating System from the underlying Hardware. Dockerizing an application is very lightweight in comparison with running the application on the Virtual Machine as all the containers share the same underlying kernel, the Host OS should be same as the container OS (eliminating guest OS) and an average machine cannot have more than few VMs running on them.

Docker Machine is a tool that lets you install Docker Engine on virtual hosts, and manage the hosts with docker-machine commands [388]. You can use Machine to create Docker hosts on your local Mac or Windows box, on your company network, in your data center, or on cloud providers like AWS or Digital Ocean. For Docker 1.12 or higher swarm mode is integrated with the Docker Engine, but on the older versions with Machine's swarm option, we can configure a swarm cluster Docker Swarm provides native clustering capabilities to turn a group of Docker engines into a single, virtual Docker Engine. With these pooled resources “you can scale out your application as if it were running on a single, huge computer” [160] as swarm can be scaled upto 1000 Nodes or upto 50,000 containers

39.1.31 KeystoneML

A framework for building and deploying large-scale machine-learning pipelines within Apache Spark. It captures and optimizes the end-to-end large-scale machine learning applications for high-throughput training in a distributed environment with a high-level API [564]. This approach increases ease of use and higher performance over existing systems for large scale learning [564]. It is designed to be a faster and more sophisticated alternative to SparkML, the machine learning framework that's a full member of the Apache Spark club. Whereas SparkML comes with a basic set of operators for processing text and numbers, KeystoneML includes a richer set of operators and algorithms designed specifically for natural language processing, computer vision, and speech processing [565]. It has enriched set of operations for complex domains:vision,NLP,Speech, plus,advanced math And is Integrated with new BDAS technologies: Velox, ml-matrix, soon Planck, TuPAQ and Sample Clean [706].

39.2 Application and Analytics

39.2.1 Mahout (384)

“Apache Mahout software provides three major features: (1) A simple and extensible programming environment and framework for building scalable algorithms (2) A wide variety of premade algorithms for Scala + Apache Spark, H2O, Apache Flink (3) Samsara, a vector math experimentation environment with R-like syntax which works at scale”

39.2.2 MLlib

MLlib is Apache Spark’s scalable machine learning library [402]. Its goal is to make machine learning scalable and easy. MLlib provides various tools such as, algorithms, feature extraction, utilities for data handling and tools for constructing, evaluating, and tuning machine learning pipelines. MLlib uses the linear algebra package Breeze, which depends on netlib-java for optimized numerical processing. MLlib is shipped with Spark and supports several languages which provides functionality for wide range of learning settings. MLlib library includes Java, Scala and Python APIs and is released as a part of Spark project under the Apache 2.0 license [394].

39.2.3 MLbase

MLBase [159] is a distributed machine learning system built with Apache Spark [543]. Machine Learning (ML) and Statistical analysis are tools for extracting insights from big data. MLbase is a tool for execute machine learning algorithms on a scalable platform. It consist of three components MLLib, MLI and ML Optimizer. MLLib was initially developed as a part of MLBase project but is now a part of Apache Spark. MLI is an experimental API for developing ML algorithm and to extract information. It provides high-level abstraction to the core ML algorithms. A prototype is currently implemented against Spark. ML optimizer on the other hand is use to automate the MLI pipeline construction. It solves for the search problem over feature extractors and ML algorithms included in MLI and ML lib. This library is its in early stage and under active development. Publications like [566], [358] and [583] are available on distributed machine learning with MLBase.

39.2.4 DataFu

The Apache DataFu project was created out of the need for stable, well-tested libraries for large scale data processing in Hadoop. As detailed in [150] Apache DatFu consists of two libraries Apache DataFu Pig and Apache DataFu Hourglass. Apache DataFu Pig is a collection of useful user-defined functions for data analysis in Apache Pig. The functions are in areas of Statistics, Bag Operations, Set Operations, Sessions, Sampling, Estimation, Hashing and Link Analysis. Apache DataFu Hourglass is a library for incrementally processing data using Hadoop MapReduce. It is designed to make computations over sliding windows more efficient. For these types of computations, the input data is partitioned in some way, usually according to time, and the range of input data to process is adjusted as new data arrives. Hourglass works with input data that is partitioned by day, as this is a common scheme for partitioning temporal data.

39.2.5 R

R, a GNU project, is a successor to S - a statistical programming language. It offers a range of capabilities – “programming language, high level graphics, interfaces to other languages and debugging”. “R is an integrated suite of software facilities for data manipulation, calculation and graphical display”. The statistical and graphical techniques provided by R make it popular in the statistical community. The statistical techniques provided include linear and nonlinear modelling, classical statistical tests, time-series analysis, classification and clustering to name a few [499]. The number of packages available in R has made it popular for use in machine learning, visualization, and data operations tasks like data extraction, cleaning, loading, transformation, analysis, modeling and visualization. Its strength lies in analyzing data using its rich library but falls short when working with very large datasets [486].

39.2.6 pbdR

Programming with Big Data in R (pbdR) [458] is an environment having series of R packages for statistical computing with Big Data using high-performance statistical computation. It uses R, a popular language between statisticians and data miners. *pbdR* focuses on distributed memory system, where data is distributed across several machines and processed in batch mode. It uses MPI for inter process communications. R focuses on single machines for data analysis using a interactive GUI. Currently there are two implementation of pbdR, one Rmpi and another being pdbMPI. Rmpi uses SPMD parallelism while pbdRMPI uses manager/worker parallelism.

39.2.7 Bioconductor

Bioconductor is an open source and open development platform used for analysis and understanding of high throughput genomic data. Bioconductor is used to analyze DNA microarray, flow, sequencing, SNP, and other biological data. All contributions to Bioconductor are under an open source license. [223] describes the goals of Bioconductor “include fostering collaborative development and widespread use of innovative software, reducing barriers to entry into interdisciplinary scientific research, and promoting the achievement of remote reproducibility of research results” [95] described that Bioconductor is primarily based on R, as most components of Bioconductor are released in R packages. Extensive documentation is provided for each Bioconductor package as vignettes, which include task-oriented descriptions for the functionalities of each package. Bioconductor has annotation functionality to associate “genomic data in real time with biological metadata from web databases such as GenBank, Entrez genes and PubMed.” Bioconductor also has tools to process genomic annotation data.

39.2.8 ImageJ

ImageJ is a Java-based image processing program developed at the National Institutes of Health (NIH). ImageJ was designed with an open architecture that provides extensibility via Java plugins and recordable macros. Using ImageJ's built-in editor and a Java compiler, it has enabled to solve many image processing and analysis problems in scientific research from three-dimensional live-cell imaging to radiological image processing. ImageJ's plugin architecture and built-in development environment has made it a popular platform for teaching image processing [303].

39.2.9 OpenCV

OpenCV stands for Open source Computer Vision. It was designed for computational efficiency and with a strong focus on real-time applications. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. It can take advantage of the hardware acceleration of the underlying heterogeneous compute platform as it is enabled with OpenCL(Open Computing Language) [1]. OpenCV 3.2 is the latest version of the software that is currently available [2].

39.2.10 Scalapack

ScaLAPACK is a library of high-performance linear algebra routines for parallel distributed memory machines. It solves dense and banded linear systems, least squares problems, eigenvalue problems, and singular value problems. It is designed for heterogeneous computing and is portable on any computer that supports Message Passing Interface or Parallel Virtual Machine [529].

ScaLAPACK is a open source software package and is available from netlib via anonymous ftp and the World Wide Web. It contains driver routines for solving standard types of problems, computational routines to perform a distinct computational task, and auxiliary routines to perform a certain subtask or common low-level computation. ScaLAPACK routines are based on block-partitioned algorithms in order to minimize the frequency of data movement between different levels of the memory hierarchy.

39.2.11 Petsc

39.2.12 PLASMA MAGMA

PLASMA is built to address the performance shortcomings of the LAPACK and ScaLAPACK libraries on multicore processors and multi-socket systems of multicore processors and their inability to efficiently utilize accelerators such as Graphics Processing Units (GPUs). Real arithmetic and complex arithmetic are supported in both single precision and double precision. PLASMA has been designed by restructuring the software to achieve much greater efficiency, where possible, on modern computers based on multicore processors. PLASMA does not support band matrices and does not solve eigenvalue and singular value problems. Also, PLASMA does not replace ScaLAPACK as software for distributed memory computers, since it only supports shared-memory machines [106] [477]. Recent activities of major chip manufacturers, such as Intel, AMD, IBM and NVIDIA, make it more evident than ever that future designs of microprocessors and large HPC systems will be hybrid/heterogeneous in nature, relying on the integration (in varying proportions) of two major types of components: [162] [268] 1. Many-cores CPU technology, where the number of cores will continue to escalate because of the desire to pack more and more components on a chip while avoiding the power wall, instruction level parallelism wall, and the memory wall; 2. Special purpose hardware and accelerators, especially Graphics Processing Units (GPUs), which are in commodity production, have outpaced standard CPUs in floating point performance in recent

years, and have become as easy, if not easier to program than multicore CPUs. While the relative balance between these component types in future designs is not clear, and will likely to vary over time, there seems to be no doubt that future generations of computer systems, ranging from laptops to supercomputers, will consist of a composition of heterogeneous components [360][617][618].

39.2.13 Azure Machine Learning

Azure Machine Learning is a cloud based service that can be used to do predictive analytics, machine learning or data mining. It has features like in-built algorithm library, machine learning studio and a web service [261]. In built algorithm library has implementation of various popular machine learning algorithms like decision tree, SVM, linear regression, neural networks etc. Machine learning studio facilitates creation of predictive models using graphical user interface by dragging, dropping and connecting of different modules that can be used by people with minimal knowledge in the machine learning field. Machine learning studio is a free service for basic version and comes with a monthly charge for advanced versions. Apart from building models, studio also has options to do preprocessing like clean, transform and normalize the data. Webservice provides option to deploy the machine learning algorithm as ready to consume APIs that can be reused in future with minimal effort and can also be published.

39.2.14 Google Prediction API & Translation API

Google Prediction API & Translation API are part of Cloud ML API family with specific roles. Below is a description of each and their use.

Google Prediction API provides pattern-matching and machine learning capabilities. Built on HTTP and JSON, the prediction API uses training data to learn and consecutively use what has been learned to predict a numeric value or choose a category that describes new pieces of data. This makes it easier for any standard HTTP client to send requests to it and parse the responses. The API can be used to predict what users might like, categorize emails as spam or non-spam, assess whether posted comments sentiments are positive or negative or how much a user may spend in a day. Prediction API has a 6 month limited free trial or a paid use for \$10 per project which offers up to 10,000 predictions a day [239].

Google Translation API is a simple programmatic interface for translating an arbitrary string into any supported language. Google Translation API is highly responsive allowing websites and applications to integrate for fast dynamic translation of source text from source language to a target language. Translation API also automatically identifies and translate languages with a high accuracy from over a hundred different languages. Google Translation API is charged at \$20 per million characters making it an affordable localization solution. Translation API is also distributed in two editions, premium edition which is tailored for users with precise long-form translation services like livestream, high volumes of emails or detailed articles and documents. There's also standard edition which is tailored for short, real-time conversations [240]. 46. `mlpy`

`mlpy` is an open source python library made for providing machine learning functionality. It is built on top of popular existing python libraries of NumPy, SciPy and GNU scientific libraries (GSL). It also makes extensive use of Cython language. These form the prerequisites for `mlpy`. [18] explains the significance of its components: NumPy, SciPy provide sophisticated N-dimensional arrays, linear algebra functionality and a variety of learning methods, GSL, which is written in C, provides complex numerical calculation functionality.

`mlpy` provides a wide range of machine learning methods for both supervised and unsupervised

learning problems. mlp is multiplatform and works both on Python 2 and 3 and is distributed under GPL3. Mlp provides both classic and new learning algorithms for classification, regression and dimensionality reduction. [403] provides a detailed list of functionality offered by mlp. Though developed for general machine learning applications, mlp has special applications in computational biology, particularly in functional genomics modeling.

39.2.15 scikit-learn

Scikit-learn is an open source library that provides simple and efficient tools for data analysis and data mining. It is accessible to everybody and reusable in various contexts. It is built on numpy, Scipy and matplotlib and is commercially usable as it is distributed under many linux distributions [105]. Through a consistent interface, scikit-learn provides a wide range of learning algorithms. Scikits are the names given to the modules for SciPy, a fundamental library for scientific computing and as these modules provide different learning algorithms, the library is named as scikit-learn [534]. It provides an in-depth focus on code quality, performance, collaboration and documentation. Most popular models provided by scikit-learn include clustering, cross-validation, dimensionality reduction, parameter tuning, feature selection and extraction.

39.2.16 PyBrain (531)

The goal of PyBrain is to provide flexible, easy-to-use algorithms that are not just simple but are also powerful for machine learning tasks. The algorithms implemented are Long Short-Term Memory (LSTM), policy gradient methods, (multidimensional) recurrent neural networks and deep belief networks. These algorithms include a variety of predefined environments and benchmarks to test and compare algorithms.

PyBrain provides a toolbox for supervised, unsupervised and reinforcement learning as well as black-box and multi-objective optimization as it is much larger than Python libraries.

PyBrain implements many recent learning algorithms and architectures while emphasizing on sequential and nonsequential data and tasks. These algorithms range from areas such as supervised learning and reinforcement learning to direct search / optimization and evolutionary methods. For application-oriented users, PyBrain contains reference implementations of a number of algorithms at the bleeding edge of research and this is in addition to standard algorithms which are not available in Python library. Besides this PyBrain sets itself apart by its versatility for composing custom neural networks architectures that range from (multi-dimensional) recurrent networks to restricted Boltzmann machines or convolutional networks.

39.2.17 ComLearn

ComLearn is a system that makes use of data compression methodologies for mining patterns in a large amount of data. So, it is basically a compression-based machine learning system. For identifying and learning different patterns, it provides a set of utilities which can be used in applying standard compression mechanisms. The most important characteristic of ComLearn is its power in mining patterns even in domains that are unrelated. It has the ability to identify and classify the language of different bodies of text [118]. This helps in reducing the work of providing background knowledge regarding a particular classification. It provides such generalization through a library that is written in ANSI C which is portable and works in many environments [118]. ComLearn provides immediate access to every core functionality in all the major languages as it is designed to be extensible.

39.2.18 DAAL(Intel)

DAAL stands for Data Analytics Acceleration Library. DAAL is software library offered by Intel which is written in C++, python, and Java which implements algorithm for doing efficient and optimized data analysis tasks to solve big-data problems [688]. The library is designed to use data platforms like Hadoop, Spark, R, and Matlab. The important algorithms which DAAL implements are 'Lower Order Moments' which is used to find out max, min standard deviation of a dataset, 'Clustering' which is used to do unsupervised learning by grouping data into unlabelled group. It also include 10-12 other important algorithms.

[308] It supports three processing modes namely batch processing, online processing and distributed processing. Intel DAAL addresses all stages of data analytics pipeline namely pre-processing, transformation, analysis, modelling, validation, and decision making.

39.2.19 Caffe

Caffe is a deep learning framework made with three terms namely expression, speed and modularity [730]. Using Expressive architecture, switching between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity cluster or mobile devices. Here the concept of configuration file will comes without hard coding the values . Switching between CPU and GPU can be done by setting a flag to train on a GPU machine then deploy to commodity clusters or mobile devices.

It can process over 60 million images per day with a single NVIIA k40 GPU. It is being used by academic research projects, startup prototypes, and even large-scale industrial applications in vision, speech, and multimedia.

39.2.20 Torch

Torch is a open source machine learning library, a scientific computing framework [673] . It implements LuaJIT programming language and implements C/CUDA. It implements N-dimensional array. It does routines of indexing, slicing, transposing etc. It has an interface to C language via scripting language LuaJIT. It supports different artificial intelligence models like neural network and energy based models. It is compatible with GPU. The core package is *torch*. It provides a flexible N dimensional array which supports basic routings. It has been used to build hardware implementation for data flows like those found in neural networks.

39.2.21 Theano

Theano is a Python library. It was written at the LISA lab. Initially it was created with the purpose to support efficient development of machine learning(ML) algorithms. Theano uses recent GPUs for higher speed. It is used to evaluate mathematical expressions and especially those mathematical expressions that include multi-dimensional arrays. Theano's working is dependent on combining aspects of a computer algebra system and an optimizing compiler. This combination of computer algebra system with optimized compilation is highly beneficial for the tasks which involves complicated mathematical expressions and that need to be evaluated repeatedly as evaluation speed is highly critical in such cases. It can also be used to generate customized C code for number of mathematical operations. For cases where many different expressions are there and each of them is evaluated just once, Theano can minimize the amount of compilation and analyses overhead [608].

39.2.22 DL4j

DL4j stands for Deeplearning4j [689]. It is a deep learning programming library written for Java and the Java virtual machine (JVM) and a computing framework with wide support for deep learning algorithms. Deeplearning4j includes implementations of the restricted Boltzmann machine, deep belief net, deep autoencoder, stacked denoising autoencoder and recursive neural tensor network, word2vec, doc2vec, and GloVe. These algorithms all include distributed parallel versions that integrate with Apache Hadoop and Spark. It is a open-source software released under Apache License 2.0.

Training with Deeplearning4j occurs in a cluster. Neural nets are trained in parallel via iterative reduce, which works on Hadoop-YARN and on Spark. Deeplearning4j also integrates with CUDA kernels to conduct pure GPU operations, and works with distributed GPUs.

39.2.23 H2O

It is an open source software for big data analysis. It was launched by the Start-up H2O in 2011. It provides an in-memory, distributed, fast and a scalable machine learning and predictive analytics platform that allows the users to build machine learning models on big data [264]. It is written in Java. It is currently implemented in 5000 companies. It provides APIs for R(3.0.0 or later), Python(2.7.x, 3.5.x), Scala(1.4-1.6) and JSON [135]. The software also allows online scoring and modeling on a single platform. It is scalable and has a wide range of OS and language support. It works perfectly on the conventional operating systems, and big data systems such as Hadoop, Cloudera, MapReduce, HortonWorks. It can be used on cloud computing environments such as Amazon and Microsoft Azure [265].

39.2.24 IBM Watson

IBM Watson [300] is a super computer built on cognitive technology that processes information like the way human brain does by understanding the data in a natural language as well as analyzing structured and unstructured data. It was initially developed as a question and answer tool more specifically to answer questions on the quiz show *Jeopardy* but now it has been seen as helping doctors and nurses in the treatment of cancer. It was developed by IBM's DeepQA research team led by David Ferrucci. [301] illustrates that with Watson you can create bots that can engage in conversation with you. You can even provide personalized recommendations to Watson by understanding a user's personality, tone and emotion. Watson uses the Apache Hadoop framework in order to process the large volume of data needed to generate an answer by creating in-memory datasets used at run-time. Watson's DeepQA UIMA (Unstructured Information Management Architecture) annotators were deployed as mappers in the Hadoop Map-Reduce framework. Watson is written in multiple programming languages like Java, C++, Prolog and it runs on the SUSE Linux Enterprise Server. [301] mentions that today Watson is available as a set of open source APIs and Software As a Service product as well.

39.2.25 Oracle PGX

Numerous information is revealed from graphs. Information like direct and indirect relations or patterns in the elements of the data, can be easily seen through graphs. The analysis of graphs can unveil significant insights. Oracle PGX (Parallel Graph AnalytiX) is a toolkit for graph analysis. “It is a fast, parallel, in-memory graph analytic framework that allows users to load up their graph data, run analytic algorithms on them, and to browse or store the result” [455]. Graphs can be loaded

from various sources like SQL and NoSQL databases, Apache Spark and Hadoop [461].

39.2.26 GraphLab

GraphLab [257] is a graph-based, distributed computation, high performance framework for machine learning written in C++. It is an open source project started by Prof. Carlos Guestrin of Carnegie Mellon University in 2009, designed considering the scale, variety and complexity of real world data. It integrates various high level algorithms such as Stochastic Gradient Descent, Gradient Descent & Locking and provides high performance experience. It includes scalable machine learning toolkits which has implementation for deep learning, factor machines, topic modeling, clustering, nearest neighbors and almost everything required to enhance machine learning models. This framework is targeted for sparse iterative graph algorithms. It helps data scientists and developers easily create and install applications at large scale.

39.2.27 GraphX

GraphX is Apache Spark's API for graph and graph-parallel computation [56].

GraphX provides:

Flexibility: It seamlessly works with both graphs and collections. GraphX unifies ETL, exploratory analysis, and iterative graph computation within a single system. You can view the same data as both graphs and collections, transform and join graphs with RDDs efficiently, and write custom iterative graph algorithms using the Pregel API.

Speed: Its performance is comparable to the fastest specialized graph processing systems while retaining Apache Spark's flexibility, fault tolerance, and ease of use.

Algorithms: GraphX comes with a variety of algorithms such as PageRank, Connected Components, Label propagations, SVD++, Strongly connected components and Triangle Count.

It combines the advantages of both data-parallel and graph-parallel systems by efficiently expressing graph computation within the Spark data-parallel framework [728].

It gets developed as a part of Apache Spark project. It thus gets tested and updated with each Spark release.

39.2.28 IBM System G

IBM System G provides a set of Cloud and Graph computing tools and solutions for Big Data [294]. In fact, the G stands for Graph and typically spans a database, visualization, analytics library, middleware and Network Science Analytics tools. It assists the easy creating of graph stores and queries and exploring them via interactive visualizations [295]. Internally, it uses the property graph model for its working. It consists of five individual components - gShell, REST API, Python interface to gShell, Gremlin and a Visualizer. Some of the typical applications wherein it can be used include Expertise Location, Commerce, Recommendation, Watson, Cybersecurity, etc [374].

However, it is to be noted that the current version does not work in a distributed environment and it is planned that future versions would support it.

39.2.29 GraphBuilder(Intel)

Intel GraphBuilder for Apache Hadoop V2 is a software that is used to build graph data models easily enabling data scientists to concentrate more on the business solution rather than preparing/formatting the data. The software automates a)Data cleaning, b)transforming data and c)creating graph models with high throughput parallel processing using hadoop, with the help of prebuilt libraries. Intel Graph Builder helps to speed up the time to insight for data scientists by automating heavy custom workflows and also by removing the complexities of cluster computing for constructing graphs from Big Data. Intel Graph Building uses Apache Pig scripting language to simplify data preparation pipeline. “Intel Graph Builder also includes a connector that parallelizes the loading of the graph output into the Aurelius Titan open source graph database—which further speeds the graph processing pipeline through the final stage”. Finally being an open source there is a possibility of adding a load of functionalities by various contributors [309].

39.2.30 TinkerPop

ThinkerPop is a graph computing framework from Apache software foundation [62]. Before coming under the Apache project, ThinkerPop was a stack of technologies like Blueprint, Pipes, Frames, Rexters, Furnace and Gremlin where each part was supporting graph-based application development. Now all parts are come under single TinkerPop project repo [168]. It uses Gremlin, a graph traversal machine and language. It allows user to write complex queries (traversal), that can use for real-time transactional (OLTP) queries, graph analytic system (OLAP) or combination of both as in hybrid. Gremlin is written in java [59]. TinkerPop has an ability to create a graph in any size or complexity. Gremlin engine allows user to write graph traversal in Gremlin language, Python, JavaScript, Scala, Go, SQL and SPARQL. It is capable to adhere with small graph which requires a single machine or massive graphs that can only be possible with large cluster of machines, without changing the code.

39.2.31 Parasol

The parasol laboratory is a multidisciplinary research program founded at Texas A&M University with a focus on next generation computing languages. The core focus is centered around algorithm and application development to find solutions to data concentrated problems [462]. The developed applications are being applied in the following areas: computational biology, geophysics, neuroscience, physics, robotics, virtual reality and computer aided drug design(CAD). The program has organized a number of workshops and conferences in the areas such as software, intelligent systems, and parallel architecture.

39.2.32 Dream:Lab

DREAM:Lab stands for “Distributed Research on Emerging Applications and Machines Lab.” [460] DREAM:Lab is centered around distributed systems research to enable expeditious utilization of distributed data and computing systems [460]. DREAM:Lab utilizes the “capabilities of hundreds of personal computers” to allow access to supercomputing resources to average individuals [504]. The DREAM:Lab pursues this goal by utilizing distributed computing [504]. Distributed computing consists of independent computing resources that communicate with each other over a network [155]. A large, complex computing problem is broken down into smaller, more manageable tasks and then these tasks are distributed to the various components of the distributed computing system [155].

39.2.33 Google Fusion Tables

Fusion Tables is a cloud based services, provided by Google for data management and integration. Fusion Tables allow users to upload the data in tabular format using data files like spreadsheet, CSV, KML, .tsv up to 250MB [242]. It used for data management, visualizing data (e.g. pie-charts, bar-charts, lineplot, scatterplot, timelines) [702], sharing of tables, filter and aggregation the data. It allows user to take the data privately, within controlled collaborative group or in public. It allows to integrate the data from different tables from different users or tables. Fusion Table uses two-layer storage, Bigtable and Magastore. The information rows are stored in bigdata table called *Rows*, user can merge the multiple table in to one, from multiple users. “Megastore is a library on top of bigtable” [241]. Data visualization is one the feature, where user can see the visual representation of their data as soon as they upload it. User can store the data along with geospatial information as well.

39.2.34 CINET

A representation of connected entities such as “physical, biological and social phenomena” [120] predictive model. Network science has grown its importance understanding these phenomena Cyberinfrastructure is middleware tool helps study Network science, [344] “by providing unparalleled computational and analytic environment for researcher”.

Network science involves study of graph a large volume which requires high power computing which usually cant be achieve by desktop. Cyberinfrastructure provides cloud based infrastructure (e.g. FutureGrid) as well as use of HPC (e.g. Shadowfax, Pecos). With use of advance intelligent Job mangers, it select the infrastructure smartly suitable for submitted job.

It provides structural and dynamic network analysis, has number of algorithms for “network analysis such as shortest path, sub path, motif counting, centrality and graph traversal”. CiNet has number of range of network visualization modules. CiNet is actively being used by several universities, researchers and analysist.

39.2.35 NWB

[433] NWB stands for Network workbench is analysis, modelling and visualization toolkit for the network scientists. It provides an environment which help scientist researchers and practitioner to get online access to the shared resource environment and network datasets for analysis, modelling and visualization of large scale networking application. User can access this network datasets and algorithms previously obtained by doing lot of research and can also add their own datasets helps in speeding up the process and saving the time for redoing the same analysis.

NWB provides advanced tools for users to understand and interact with different types of networks. NWB members are largely the computer scientist, biologist, engineers, social and behavioral scientist. The platform helps the specialist researchers to transfer the knowledge within the broader scientific and research communities.

39.2.36 Elasticsearch

Elasticsearch [173] is a real time distributed, RESTful search and analytics engine which is capable of performing full text search operations for you. It is not just limited to full text search operations but it also allows you to analyze your data, perform CRUD operations on data, do basic text analysis including tokenization and filtering [175]. For example while developing an E-commerce website,

Elasticsearch can be used to store the entire product catalog and inventory and can be used to provide search and autocomplete suggestions for the products. Elasticsearch is developed in Java and is an open source search engine which uses standard RESTful APIs and JSON on top of Apache's Lucene - which is a full text search engine library. Clinton Gormley & Zachary Tong [255] describes elastic search as "A distributed real time document store where every field is indexed and searchable". They also mention that "Elastic search is capable of scaling to hundreds of servers and petabytes of structured and unstructured data" [176]. mentions that Elastic search can be used on big data by using the Elasticsearch-Hadoop (ES-Hadoop) connector. ES-Hadoop connector lets you index the Hadoop data into the Elastic Stack to take full advantage of the Elasticsearch engine and returns output through Kibana visualizations [174]. A log parsing engine "Logstash" and analytics and visualization platform *Kibana* are also developed alongside Elasticsearch forming a single package.

39.2.37 Kibana

Kibana is an open source data visualization plugin for Elasticsearch [352]. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data [226]. The combination of Elasticsearch, Logstash, and Kibana (also known as ELK stack or Elastic stack) is available as products or service. Logstash provides an input stream to Elastic for storage and search, and Kibana accesses the data for visualizations such as dashboards [283]. Elasticsearch is a search engine based on Lucene [177]. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Kibana makes it easy to understand large volumes of data. Its simple, browser-based interface enables you to quickly create and share dynamic dashboards that display changes to Elasticsearch queries in real time [351] [311].

39.2.38 Logstash

Logstash is an open source data collection engine with real-time pipelining capabilities. Logstash can dynamically unify data from disparate sources and normalize the data into destinations of your choice [178]. Cleanse and democratize all your data for diverse advanced downstream analytics and visualization use cases.

While Logstash originally drove innovation in log collection, its capabilities extend well beyond that use case. Any type of event can be enriched and transformed with a broad array of input, filter, and output plugins, with many native codecs further simplifying the ingestion process. Logstash accelerates your insights by harnessing a greater volume and variety of data.

39.2.39 Graylog

Graylog is an open source log management tool that allows an organization to assemble, organize and analyze large amounts of data from its network activity. It collects and aggregates events from a group of sources and presents data in a streamlined, simplified interface where one can drill down to significant metrics, identify key relationships, generate powerful data visualizations and derive actionable insights [381]. Graylog allows us to centrally collect and manage log messages of an organization's complete infrastructure [214]. A user can perform search on terabytes of log data to discover number of failed logins, find application errors across all servers or monitor the activity of a suspicious user id. Graylog works on top of ElasticSearch and MongoDB to facilitate this high availability searching. Graylog provides visualization through creation of dashboards that allows a user to build pre-defined views on his data to assemble all of his important data only a

single click away [258]. Any search result or metric shall be added as a widget on the dashboard to observe trends in one single location. These dashboards can also be shared with other users in the organization. Based on a user's recent search queries, graylog also allows you to distinguish data that are not searched upon very often and thus can be archived on cost effective storage drives. Users can also add certain trigger conditions that shall alert the system about performance issues, failed logins or exceptions in the flow of the application.

39.2.40 Splunk

Splunk is a platform for big data analytics. It is a software product that enables you to search, analyze, and visualize the machine-generated data gathered from the websites, applications, sensors, devices, and so on, that comprise your IT infrastructure or business [568]. After defining the data source, Splunk indexes the data stream and parses it into a series of individual events that you can view and search. It provides distributed search and MapReduce linearly scales search and reporting. It uses a standard API to connect directly to applications and devices. It was developed in response to the demand for comprehensible and actionable data reporting for executives outside a company's IT department [568].

39.2.41 Tableau

[581] Tableau is a family of interactive data visualization products focused on business intelligence. The different products which tableau has built are: Tableau Desktop, for individual use; Tableau Server for collaboration in an organization; Tableau Online, for Business Intelligence in the Cloud; Tableau Reader, for reading files saved in Tableau Desktop; Tableau Public, for journalists or anyone to publish interactive data online. [582]. Tableau uses VizQL as a visual query language for translating drag-and-drop actions into data queries and later expressing the data visually. Tableau also benefits from an Advanced In-Memory Technology for handling large amounts of data. The strengths of Tableau are mainly the ease of use and speed. However, it has a number of limitations, which the most prominent are unfitness for broad business and technical user, being closed-source, no predictive analytical capabilities and no support for expanded analytics.

39.2.42 D3.js

D3.js is a JavaScript library responsible for manipulating documents based on data. D3 helps in making data more interactive using HTML, SVG, and CSS. D3's emphasis on web standards makes it framework independent utilizing the full capabilities of modern browsers, combining powerful visualization components and a data-driven approach to DOM manipulation [149].

It assists in binding random data to a Document Object Model (DOM), followed by applying data-driven transformations to the document. It is very fast, supports large datasets and dynamic behaviours involving interaction and animation.

39.2.43 three.js

Three.js is an API library with about 650 contributions till date , where users can create and display an animated 3D computer graphics in a web browser. It is written in javascript and uses WebGL, HTML5 or SVG. Users can animate HTML elements using CSS3 or even import models from 3D modelling apps [610]. In order to display anything using three.js we need three basic features, which are scene, camera and renderer. This will result in rendering the scene with a camera. In addition to these three features , we can add animation, lights (ambience,spot lights, shadows),

objects (lines , ribbons , particles) , geometry etc [611].

39.2.44 Potree

Potree [484] is a opensource tool powered by WebGL based viewer to visualize data from large point clouds. It started at the TU Wien, institute of Computer Graphics and Algorithms and currently begin continued under the Harvest4D project. Potree relies on reorganizing the point cloud data into an multi-resolution octree data structure which is time consuming. Its efficiency can be improved by using techniques such as divide and conquer as discussed in a conference paper Taming the beast: Free and Open Source massive cloud point cloud web visualization [387]. It has also been widely used in works involving spatio-temporal data where the changes in geographical features are across time [269].

39.2.45 DC.js

According to [154]: “DC.js is a javascript charting library with native crossfilter support, allowing exploration on large multi-dimensional datasets. It uses d3 to render charts in CSS-friendly SVG format. Charts rendered using dc.js are data driven and reactive and therefore provide instant feedback to user interaction.” DC.js library can be used to perform data analysis on both mobile devices and different browsers. Under the dc namespace the following chart classes are included: barChart, boxplot, bubbleChart, bubbleOverlay, compositeChart, dataCount, dataGrid, dataTable, geoChoroplethChart, heatMap, legend, lineChart, numberDisplay, pieChart, rowChart, scatterPlot, selectMenu and seriesChart.

39.2.46 TensorFlow

TensorFlow is a platform that provides a software library for expressing and executing machine learning algorithms. [4] states TensorFlow has a flexible architecture allowing it to be executed with minimal change to many heterogeneous systems such as CPUs and GPUs of mobile devices, desktop machines, and servers. TensorFlow can “express a wide variety of algorithms, including training and inference algorithms for deep neural network models, and it has been used for conducting research and for deploying machine learning systems into production across more than a dozen areas”. [588] describes that TensorFlow utilizes data flow graphs in which the “nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them.” TensorFlow was developed by the Google Brain Team and has a reference implementation that was released on 2015-11-09 under the Apache 2.0 open source license.

39.2.47 CNTK

The Microsoft Cognitive Toolkit - CNTK - is a unified deep-learning toolkit by Microsoft Research. It is in essence an implementation of Computational Network(CN) which supports both CPU and GPU. CNTK supports arbitrary valid computational networks and makes building DNNs, CNNs, RNNs, LSTMs, and other complicated networks as simple as describing the operations of the networks. The toolkit is implemented with efficiency in mind. It removes duplicate computations in both forward and backward passes, uses minimal memory needed and reduces memory reallocation by reusing them. It also speeds up the model training and evaluation by doing batch computation whenever possible [130]. It can be included as a library in your Python or C++ programs, or used as a standalone machine learning tool through its own model description language (BrainScript)

[280]. Latest Version:2017-02-10. V 2.0 Beta 11 Release

39.3 Application Hosting Frameworks

39.3.1 Google App Engine

Google App Engine is a cloud computing platform to host your mobile or web applications on Google managed servers. Google App Engine provides automatic scaling for web applications, i.e it automatically allocates more resources to the application upon increase in the number of requests. It gives developers the freedom to focus on developing their code and not worry about the infrastructure. Google App Engine provides built-in services and APIs such as load balancing, automated security scanning, application logging, NoSQL datastores, memcache, and a user authentication API, that are a core part to most applications [65].

An App Engine platform can be run in either the Standard or the Flexible environment. Standard environment lays restrictions on the maximum number of resources an application can use and charges a user based on the instance hours used. The flexible environment as the name suggests provides higher flexibility in terms of resources and is charged based on the CPU and disk utilization. The App Engine requires developers to use only its supported languages and frameworks. Supported languages are Java, Python, Ruby, Scala, PHP, GO, Node.js and other JVM oriented languages. The App Engine datastore uses a SQL like syntax called the GQL (Google Query Language) which works with non-relational databases when compared to SQL [66].

39.3.2 AppScale

AppScale is an application hosting platform. This platform helps to deploy and scale the unmodified Google App Engine application, which run the application on any cloud infrastructure in public, private and on premise cluster [68]. AppScale provide rapid, API development platform that can run on any cloud infrastructure. The platform separates the app logic and its service part to have control over application deployment, data storage, resource use, backup and migration. AppScale is based on Google's App Engine APIs and has support for Python, Go, PHP and Java applications. It supports single and multimode deployment, which will help with large, dataset or CPU. AppScale allows to deploy app in thee main mode i.e. dev/test, production and customize deployment [69].

39.3.3 Red Hat OpenShift

OpenShift was launched as a PaaS (Platform as a Service) by Red Hat in the Red Hat Summit, 2011 [514]. It is a cloud application development and hosting platform that envisages shifting of the developer's focus to development by automating the management and scaling of applications [511]. Thus, OpenShift [512] enables us to write our applications in any one web development language (using any framework) and it itself takes up the task of running the application on the web. This has its advantages and disadvantages - advantage being the developer doesn't have to worry about how the stuff works internally (as it is abstracted away) and the disadvantage being that he cannot control how it works, again because it is abstracted.

OpenShift is powered by Origin, which is in turn built using Docker container packaging and Kubernetes container cluster [513]. Due to this, OpenShift offers a lot of options, including online, on-premise and open source project options.

39.3.4 Heroku

Heroku [278] is a platform as a service that is used for building, delivering monitoring and scaling applications. It lets you develop and deploy application quickly without thinking about irrelevant problems such as infrastructure. Heroku also provides a secure and scalable database as a service with number of developers' tools like database followers, forking, data clips and automated health checks. It works by deploying to cedar stack [110], an online runtime environment that supports apps built in Java, Node.js, Scala, Clojure, Python and PHP. It uses Git for version controlling. It is also tightly integrated with Salesforce, providing seamless and smooth Heroku and Salesforce data synchronization enabling companies to develop and design creative apps that use both platforms.

39.3.5 Aerobatic

According to [14]: Aerobatic is a platform that allows hosting static websites. It used to be an ad-on for Bitbucket but now Aerobatic is transitioning to standalone CLI(command Line Tool) and web dashboard . Aerobatic allows automatic builds to different branches. New changes to websites can be deployed using aero deploy command which can be executed from local desktop or any of CD tools and services like Jenkins, Codeship, Travis and so on. It also allows users to configure custom error pages and offers authentication which can also be customized. Aerobatic is backed by AWS cloud. Aerobatic has free plan and pro plan options for customers.

39.3.6 AWS Elastic Beanstalk

AWS Elastic Beanstalk is an orchestration service offered from Amazon Web Services which provides user with a platform for easy and quick deployment of their WebApps and services [27]. Amazon Elastic BeanStack automatically handles the deployment details of capacity provisioning by Amazon Cloud Watch, Elastic Load Balancing, Auto-scaling, and application health monitoring of the WebApps and service [340]. AWS Management Console allows the users to configure an automatic scaling mechanism of AWS Elastic Beanstalk. Elastic Load Balancing enables a load balancer, which automatically spreads the load across all running instances in an auto-scaling group based on metrics like request count and latency tracked by Amazon CloudWatch. Amazon CloudWatch tracks and stores per-instance metrics, including request count and latency, CPU, and RAM utilization. Elastic Beanstalk supports applications developed in Java, PHP, .NET, Node.js, Python, and Ruby as well as supports different container types for each language such as Apache Tomcat for Java applications, Apache HTTP Server for PHP applications Docker, GO and much more for specific languages where the container defines the infrastructure and software stack to be used for a given environment. “AWS Elastic Beanstalk runs on the Amazon Linux AMI and the Windows Server 2012 R2 AMI. Both AMIs are supported and maintained by Amazon Web Services and are designed to provide a stable, secure, and high-performance execution environment for Amazon EC2 Cloud computing”[27].

39.3.7 Azure

Microsoft Corporation (MSFT) markets its cloud products under the *Azure* brand name. At its most basic, Azure acts as an *infrastructure- as-a-service* (IaaS) provider. IaaS virtualizes hardware components, a key differentiation from other *-as-a-service* products. IaaS “abstract[s] the user from the details of infrastructure like physical computing resources, location, data partitioning, scaling, security, backup, etc.” [686]

However, Azure offers a host of closely-related tool and products to enhance and improve the

core product, such as raw block storage, load balancers, and IP addresses [398]. For instance, Azure users can access predictive analytics, Bots and Blockchain-as-a-Service [398] as well as more-basic computing, networking, storage, database and management components [397]. The Azure website shows twelve major categories under *Products* and twenty *Solution* categories, e.g., e-commerce or Business SaaS apps.

Azure competes against Amazon's *Amazon Web Service*, [28] even though IBM (*SoftLayer* [297] and *Bluemix* [298]) and Google (*Google Cloud Platform*) [22] offer IaaS to the market. As of January 2017, Azure's datacenters span 32 Microsoft-defined *regions*, or 38 *declared regions*, throughout the world. [398]

39.3.8 Cloud Foundry

It is an open source software with multi cloud application .It is a platform for running applications and services. It was originally developed by VMware and currently owned by Pivotal . It is written in Ruby and Go .It has a commercial version called Pivotal Cloud Foundry (PFC)[705]. Cloud Foundry is available as a stand alone software package, we can also deploy it to Amazon AWS as well as host it on OpenStack server , HP's Helion or VMware's vSphere as given in the blog [123] , it delivers quick application from development to deployment and is highly scalable. It has a DevOps friendly workflow. Cloud Foundry changes the way application and services are deployed and reduces the develop to deployment cycle time.

39.3.9 Pivotal

Pivotal Software, Inc. (Pivotal) is a software and services company. It offeres multiple consulting and technology services, which includes Pivotal Web Services, which is an agile application hosting service. It has a single step upload feature *cf push*, another feature called Buildpacks lets us push applications written for any language like Java, Grails, Play, Spring, Node.js, Ruby on Rails, Sinatra or Go. Pivotal Web Services also allows developers to connect to 3rd party databases, email services, monitoring and more from the Marketplace. It also offers performance monitoring, active health monitoring, unified log streaming, web console built for team-based agile development [472].

39.3.10 IBM BlueMix

39.3.11 (Ninefold)

The Australian based cloud computing platform has shut down their services since January 30, 2016. Refer [427]

39.3.12 Jelastic

Jelastic (acronym for Java Elastic) is an unlimited PaaS and Container based IaaS within a single platform that provides high availability of applications, automatic vertical and horizontal scaling via containerization to software development clients, enterprise businesses, DevOps, System Admins, Developers, OEMs and web hosting providers [326]. Jelastic is a Platform-as-Infrastructure provider of Java and PHP hosting. It has international hosting partners and data centers. The company can add memory, CPU and disk space to meet customer needs. The main competitors of Jelastic are Google App Engine, Amazon Elastic Beanstalk, Heroku, and Cloud Foundry. Jelastic is unique in that it does not have limitations or code change requirements, and it offers automated vertical

scaling, application lifecycle management, and availability from multiple hosting providers around the world [263].

39.3.13 Stackato

Hewlett Packard Enterprise or HPE Helion Stackato is a platform as a service(PaaS) cloud computing solution. The platform facilitates deployment of the user's application in the cloud and will function on top of an Infrastructure as a service(IaaS) [286]. Multiple cloud development is supported across AWS, vSphere, and Helion Openstack. The platform supports the following programming languages: native .NET support, java, Node.js, python, and ruby. This flexibility is advantageous compared to early PaaS solutions which would force the customer into utilizing a single stack. Additionally, this solution has the capacity to support private, public and hybrid clouds. [343] This capability user has to not have to make choices of flexibility over security of sensitive data when choosing a cloud computing platform.

39.3.14 appfog

According to [647], "AppFog is a platform as a service (PaaS) provider." Platform as a service provides a platform for the development of web applications without the necessity of purchasing the software and infrastructure that supports it [346]. PaaS provides an environment for the creation of software [346]. The underlying support infrastructure that AppFog provides includes things such as runtime, middleware, o/s, virtualization, servers, storage, and networking [67]. AppFog is based on VMWare's CloudFoundry project [647]. It gets things such as MySQL, Mongo, Reddis, memCache, etc. running and then manages them [626].

39.3.15 CloudBees

Cloudbees provides Platform as a Service (PaaS) solution, which is a cloud service for Java applications [125]. It is used to build, run and manage the web applications. It was created in 2010 by Jenkins. It has a continuous delivery platform for DevOps, and adds a enterprise-grade functionality with an expert level support. Cloudbees is better than the traditional Java platform as it requires no provision of the nodes, clusters, load balancers and databases. In cloudbees the environment is constantly managed and monitored where a metering and scale updating is done on a real time basis. The platform ships with verified security and enhancements assuring less risk for sharing sensitive information. It implies the task of getting the platform accessed by every user using the feature *Jenkins Sprawl* [126].

39.3.16 Engine Yard (577)

A deployment platform with fully managed services that combines high-end clustering resources to run Ruby and Rails applications in the cloud is offered by Engine Yard. It is designed as a platform-as-a-Service for Web application developers using Ruby on Rails, PHP and Node.js who requires the advantages of cloud computing. Amazon cloud is the platform where the Engine Yard perform its operations and accomplishes application stack for its users. Amazon allows as many as eight regions to Engine Yard to deploy its CPU instances in varying capacities such as normal, high memory and high CPU. According to customer requirements multiple software components are configured and processed when an instance is started in Engine Yard.

Engine Yard builds its version on Gentoo Linux and has non-proprietary approach to its stack. The stack includes HAProxy load balancer, Ngnix and Rack Web servers, Passenger and Unicorn app

servers, as well as MySQL and PostgreSQL relational databases in addition to Ruby, PHP, and Node.js. The credibility of Engine Yard rests with orchestration and management as developers have option of performing functions in Amazon cloud. Standard operations management procedures are performed once the systems are configured and deployed. Key operations tasks such as performing backups, managing snapshots, managing clusters, administering databases and load balancing are taken care by Engine Yard.

Engine Yard users are empowered as they have more control over virtual machine instances. These instances are dedicated instances and are not shared with other users. As the instances are independent every user can exercise greater control over instances without interferences with other users.

39.3.17 (CloudControl)

No Longer active as of Feb. 2016 [[674](#)]

39.3.18 dotCloud ([429](#))

dotCloud services were shutdown on February 29,2016.

39.3.19 Dokku

39.3.20 OSGi

39.3.21 HUBzero

HUBzero is a collaborative framework which allows creation of dynamic websites for scientific research as well as educational activities. HUBzero lets scientific researchers work together online to develop simulation and modeling tools. These tools can help you connect with powerful Grid computing resources as well as rendering farms [[655](#)]. Thus allowing other researchers to access the resulting tools online using a normal web browser and launch simulation runs on the Grid infrastructure without having to download or compile any code. It is a unique framework with simulation and social networking capabilities [[391](#)].

39.3.22 OODT

The Apache Object Oriented Data Technology (OODT) is an open source data management system framework. OODT was originally developed at NASA Jet Propulsion Laboratory to support capturing, processing and sharing of data for NASA's scientific archives. OODT focuses on two canonical use cases: Big Data processing and on Information integration. It facilitates the integration of highly distributed and heterogeneous data intensive systems enabling the integration of different, distributed software systems, metadata and data. OODT is written in the Java, and through its REST API used in other languages including Python [[441](#)].

39.3.23 Agave

Agave is an open source, application hosting framework and provides a platform-as-a-service solution for hybrid computing [[646](#)]. It provides everything ranging from authentication and authorization to computational, data and collaborative services. Agave manages end to end lifecycle of an application's execution. Agave provides an execution platform, data management platform,

or an application platform through which users can execute applications, perform operations on their data or simple build their web and mobile applications [15].

Agave's API's provide a catalog with existing technologies and hence no additional appliances, servers or other software needs to be installed. To deploy an application from the catalog, the user needs to host it on a storage system registered with Agave, and submit to agave, a JSON file that shall contain the path to the executable file, the input parameters, and specify the desired output location. Agave shall read the JSON file, formalize the parameters, execute the user program and dump the output to the requested destination [646].

39.3.24 Atmosphere

Atmosphere is developed by CyVerse (previously named as iPlant Collaborative). It is a cloud-computing platform. It allows one to launch his own “isolated virtual machine (VM) image [74]. It does not require any machine specification. It can be run on any device (tablet/desktop/laptop) and any machine(Linux/Windows/Max/Unix). User should have a CyVerse account and be granted permission to access to Atmosphere before he can begin using Atmosphere. No subscription is needed. Atmosphere is designed to execute data-intense bioinformatics tasks that may include a)Infrastructure as a Service (IaaS) with advanced APIs; b)Platform as a Service (PaaS), and c)Software as a Service (SaaS). On Atmosphere one has several images of virtual machine and user can launch any image or instance according to his requirements. The images launched by users can be shared among different members as and when required [378].

39.4 High level Programming

39.4.1 Kite

Kite is a programming language designed to minimize the required experience level of the programmer. It aims to allow quick development and running time and low CPU and memory usage. Kite was designed with lightweight systems in mind. On OS X Leopard, the main Kite library is only 88KB, with each package in the standard library weighing in at 13-30KB. The main design philosophy is minimalism — only include the minimum necessary, while giving developers the power to write anything that they can write in other languages. Kite combines both object oriented and functional paradigms in the language syntax. One special feature is its use of the pipe character (!) to indicate function calls, as opposed to the period (.) or arrow (->) in other languages. Properties are still de-referenced using the period [156]. Kite also offers a digital assistant for programmers. Kite offers a product which sits as a sidebar in code editor and enables programmers to search for opensource codes to implement in their codes. It even provides relavant examples/syntax and also tries to spot errors in the programs [643].

39.4.2 Hive

The reason behind development of Hive is making it easier for end users to use Hadoop. Map reduce programs were required to be developed by users for simple to complex tasks. It lacked expressiveness like query language. So, it was a time consuming and difficult task for end users to use Hadoop. For solving this problem Hive was built in January 2007 and open sourced in August2008. Hive is an open source data warehousing solution which is built on top of Hadoop. It structures data into understandable and conventional database terms like tables, columns, rows and partitions. It supports HiveQL queries which have structure like SQL queries. HiveQL queries are compiled to map reduce jobs which are then executed by Hadoop. Hive also contains Metastore

which includes schemas and statistics which is useful in query compilation, optimization and data exploration [70]

39.4.3 HCatalog

39.4.4 Tajo

Apache Tajo [57] is a big data relational and distributed data warehouse system for Apache's Hadoop framework. It uses the Hadoop Distributed File System (HDFS) as a storage layer and has its own query execution engine instead of the MapReduce framework. Tajo is designed to provide low-latency and scalable ad-hoc queries, online aggregation, and ETL (extraction-transformation-loading process) on large-data sets which are stored on HDFS (Hadoop Distributed File System) and on other data sources [58]. Apart from HDFS, it also supports other storage formats as Amazon S3, Apache HBase, Elasticsearch etc. It provides distributed SQL query processing engine and even has query optimization techniques and provides interactive analysis on large-data sets. Tajo is compatible with ANSI/ISO SQL standard, JDBC standard. Tajo can also store data from various file formats such as CSV, JSON, RCFFile, SequenceFile, ORC and Parquet. It provides a SQL shell which allows users to submit the SQL queries. It also offers user defined functions to work with it which can be created in python. A Tajo cluster has one master node and a number of worker nodes [58]. The master node is responsible for performing the query planning and maintaining a coordination among the worker nodes. It does this by dividing a query in small task which are assigned to the workers who have a local query engine for executing the queries assigned to them.

39.4.5 Shark

Data Scientists when working on huge data sets try to extract meaning and interpret the data to enhance insight about the various patterns, opportunities, and possibilities that the dataset has to offer [179]. At a traditional EDW (Enterprise Data Warehouse), a simple data manipulation can be performed using SQL queries but we have to rely on other systems to apply the machine learning algorithms on these data sets. Apache Shark is a distributed query engine developed by the open source community whose goal is to provide a unified system for easy data manipulation using SQL and pushing sophisticated analysis towards the data.

Shark is a data Warehouse system built on top of Apache Spark which does the parallel data execution and is also capable of deep data analysis using the Resilient Distributed Datasets(RDD) memory abstraction which unifies the SQL query processing engine with analytical algorithms [179]. Based on this common abstraction, it allows running two query in the same set of workers and share intermediate data. Since RDDs are designed to scale horizontally, it is easy to add or remove nodes to accommodate more data or faster query processing. Thus, it can be scaled to the large number of nodes in a fault-tolerant manner

"Shark is built on Hive Codebase and it has the ability to execute HIVE QL queries up to 100 times faster than Hive without making any change in the existing queries [179]. "Shark can run both on the Standalone Mode and Cluster-Mode. Shark can answer the queries 40X faster than Apache Hive and can run machine learning algorithms 25X faster than MapReduce programs in Apache Hadoop on large data sets" [179]. Thus, this new data analysis system performs query processing and complex analytics (iterative Machine learning) at scale and efficiently recovers from the failures.

39.4.6 Phoenix

In the first quarter of 2013, Salesforce.com released its proprietary SQL-like interface and query engine for HBase, *Phoenix*, to the open source community. The company appears to have been motivated to develop Phoenix as a way to 1) increase accessibility to HBase by using the industry-standard query language (SQL); 2) save users time by abstracting away the complexities of coding native HBase queries; and, 3) implementing query best practices by implementing them automatically via Phoenix [348]. Although Salesforce.com initially *open-sourced* it via Github, by May of 2014 it had become a top-level Apache project [683].

Phoenix, written in Java, ”compiles [SQL queries] into a series of HBase scans, and orchestrates the running of those scans to produce regular JDBC result sets [55]. “ In addition, the program directs compute intense portions of the calls to the server. For instance, if a user queried for the top ten records across numerous regions from an HBase database consisting of a billion records, the program would first select the top ten records for each region using server-side compute resources. After that, the client would be tasked with selecting the overall top ten” [536].

Despite adding an abstraction layer, Phoenix can actually speed up queries because it optimizes the query during the translation process [348]. For example, “Phoenix beats Hive for a simple query spanning 10M-100M rows” [76].

Finally, another program can enhance HBase’s accessibility for those inclined towards graphical interfaces. SQuirell only requires the user to set up the JDBC driver and specify the appropriate connection string. [580].

39.4.7 Impala

Cloudera Impala is Cloudera’s open source massively parallel processing (MPP) SQL query engine for data stored in a computer cluster running Apache Hadoop [129]. It allows users to execute low latency SQL queries for data stored in HDFS and HBase, without any movement or transformation of data. The Apache Hive provides a powerful query mechanism for hadoop users, but the query response time are not acceptable due to Hive’s reliance on MapReduce. Impala technology by Cloudera has its MPP query engine written in C++ replacing the Java engine proves to improve the interactive Hadoop queries and interactive query response time for hadoop users [197] . Impala is faster than Hive also because it executes the SQL queries natively without translating them into Hadoop MapReduce jobs, thus taking less time. Impala uses HiveQL as programming interface and also the Impala’s Query Exec Engines are co-located with the HDFS data nodes, so that the data nodes and processing tasks are co-located, following the haddops paradigm [197]. Impala can also use Hbase as a data source. Thus, Impala can be considered as an extension to the Apache Hadoop, providing a better performance alternative to Hive-on-top-of-MapReduce model.

Hive and other frameworks built on MapReduce are best suited for long running batch jobs, such as those involving batch processing of Extract, Transform, and Load (ETL) type jobs [129]. The important applications of Impala are when the data is to be partially analyzed or when the same kind of query is to be processed several times from the dataset. When the data is to be partially analyzed, Impala uses parquet as the file format, which is developed by Twitter and Cloudera and it stores data in vertical manner [8]. When Parquet queries the dataset it only reads the column split part files rather than reading the entire dataset as compared to Hive.

39.4.8 MRQL

MapReduce Query Language (MRQL, pronounced miracle) “is a query processing and optimization system for large-scale, distributed data analysis” [52]. MRQL provides a SQL like language for use on Apache Hadoop, Hama, Spark, and Flink. MRQL allows users to perform complex data analysis using only SQL like queries, which are translated by MRQL to efficient Java code. MRQL can evaluate queries in Map-Reduce (using Hadoop), Bulk Synchronous Parallel (using Hama), Spark, and Flink modes [52].

MRQL was created in 2011 by Leleonids Fegaras [731] and is currently in the Apache Incubator. All projects accepted by the Apache Software Foundation (ASF) undergo an incubation period until a review indicates that the project meets the standards of other ASF projects [51].

39.4.9 SAP HANA

As noted in [723], SAP HANA is in-memory massively distributed platform that consists of three components: analytics, relational ACID compliant database and application. Predictive analytics and machine learning capabilities are dynamically allocated for searching and processing of spatial, graphical, and text data. SAP HANA accommodates flexible development and deployment of data on premises, cloud and hybrid configurations. In a nutshell, SAP HANA acts as a warehouse that integrates live transactional data from various data sources on a single platform [437]. It provides extensive administrative, security features and data access that ensures high data availability, data protection and data quality.

39.4.10 HadoopDB

HadoopDB is a hybrid of parallel database and MapReduce technologies. It approaches parallel databases in performance and efficiency, yet still yields the scalability, fault tolerance, and flexibility of MapReduce systems. It is a free and open source parallel DBMS. The basic idea behind it is to give Hadoop access to multiple single-node DBMS servers (eg. PostgreSQL or MySQL) deployed across the cluster. It pushes as much as possible data processing into the database engine by issuing SQL queries which results in resembling a shared-nothing cluster of machines [267].

HadoopDB is more scalable than currently available parallel database systems and DBMS/MapReduce hybrid systems. It has been demonstrated on clusters with 100 nodes and should scale as long as Hadoop scales, while achieving superior performance on structured data analysis workloads.

39.4.11 PolyBase

“PolyBase is a technology that accesses and combines both non-relational and relational data, all from within SQL Server. It allows you to run queries on external data in Hadoop or Azure Blob storage acts mediator between SQL and non SQL data store it makes the analysis of the relation data and other data that is non structure to tables (Hadoop).” [481] Unless there is a way to transfer data between the data stores it is always difficult to do so. PolyBase bridges this gap by operating on data that is external to SQL server. It don’t require additional software, querying to external can be done with same syntax as querying a database table. This happens transparently behind the scene, no knowledge of Hadoop or Azure is required.

It can query data store in Hadoop using T-SQL, polybase also makes it easy to access the Azure blob data using T-SQL. There is no need for a separate ETL or import tool while importing data from Hadoop, “Azure blob storage or Azure Data Lake into relational tables. It leverages Microsoft’s

Columnstore technology and analysis capabilities while importing” [481]. It also archives data into Hadoop Azure blob and data lake store in cost effective way.

Push computation to Hadoop. The query optimizer makes a cost-based decision to push computation to Hadoop and while doing so will improve query performance. It uses statistics on external tables to make the cost-based decision. Pushing computation creates MapReduce jobs and leverages Hadoop’s distributed computational resources. Scale compute resources. SQL Server PolyBase scale-out groups can be used to improve query performance. This enables parallel data transfer between SQL Server instances and Hadoop nodes, and it adds compute resources for operating on the external data.

39.4.12 Pivotal HD/Hawq

Pivotal HDB is the Apache Hadoop native SQL database powered by Apache HAWQ [475] for data science and machine learning workloads. It can be used to gain deeper and actionable insights into data without the need from moving data to another platform to perform advanced analytics. Few important problems that Pivot HDB address are as follows Quickly unlock business insights with exceptional performance, Integrate SQL BI tools with confidence and Iterate advanced analytics and machine learning in database support. Pivotal HDB comes with an elastic SQL query engine which combines MPP-based analytical performance, robust ANSI SQL compliance and integrated Apache MADlib for machine learning [476].

39.4.13 Presto

Presto [488] is an open-source distributed SQL query engine that supports interactive analytics on large datasets. It allows interfacing with a variety of data sources such as Hive, Cassandra, RDBMSs and proprietary data source. Presto is used at a number of big-data companies such as Facebook, Airbnb and Dropbox. Presto’s performance compares favorably to similar systems such as Hive and Stinger [117].

39.4.14 Google Dremel

Dremel is a scalable, interactive ad-hoc query system for analysis of read-only nested data. By combining multi-level execution trees and columnar data layout, Google Dremel is capable of running aggregation queries over trillion-row tables in seconds [393]. With Dremel, you can write a declarative SQL-like query against data stored in a read-only columnar format efficiently for analysis or data exploration. It’s also possible to write queries that analyze billions of rows, terabytes of data, and trillions of records in seconds. Dremel can be used for a variety of jobs including analyzing web-crawled documents, detecting e-mail spam, working through application crash reports.

39.4.15 Google BigQuery

Google BigQuery [92] is an enterprise data warehouse used for large scale data analytics [93]. A user can store and query massive datasets by storing the data in BigQuery and querying the database using fast SQL queries using the processing power of Google’s infrastructure. In Google BigQuery a user can control access to both the project and the data based on his business needs which gives the ability to others to view and even query the data [92]. BigQuery can scale the database from GigaBytes to PetaBytes. BigQuery can be accessed using a Web UI or a command-line tool or even by making calls to the BigQuery REST API using a variety of client libraries such as Java,

.NET or python. BigQuery can also be accessed using a variety of third party tools. BigQuery is fully managed to get started on its own, so there is no need to deploy any resources such as disks and virtual machines.

Projects in BigQuery [93] are top-level containers in Google Cloud Platform. They contain the BigQuery Data. Each project is referenced by a name and unique ID. Tables contain the data in BigQuery. Each table has a schema that describes field names, types, and other information. Datasets enable to organise and control access to the tables. Every table must belong to a dataset. A BigQuery data can be shared with others by defining roles and setting permissions for organizations, projects, and datasets, but not on the tables within them. BigQuery stores data in the [464] Capacitor columnar data format, and offers the standard database concepts of tables, partitions, columns, and rows.

39.4.16 Amazon Redshift

Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud. Redshift service manages all of the work of setting up, operating and scaling a data warehouse. AWS Redshift can perform these tasks including provisioning capacity, monitoring and backing up the cluster, and applying patches as well as upgrades to the Redshift's engine [26]. Redshift is built on the top of technology from the Massive Parallel Processing (MPP) data-warehouse company ParAccel which based on PostgreSQL 8.0.2 to PostgreSQL 9.x with capability to handle analytics workloads on large- scale dataset stored by a column-oriented DBMS principle [678].

39.4.17 Drill

Apache Drill [164] is an open source framework that provides schema free SQL query engine for distributed large-scale datasets. Drill has an extensible architecture at its different layers. It does not require any centralized metadata and does not have any requirement for schema specification. Drill is highly useful for short and interactive ad-hoc queries on very large scale data sets. It is scalable to several thousands of nodes. Drill is also capable to query nested data in various formats like JSON and Parquet. It can query large amount of data at very high speed. It is also capable of performing discovery of dynamic schema. A service called 'Drillbit' is at the core of Apache Drill responsible for accepting requests from the client, processing the required queries, and returning all the results to the client. Drill is primarily focused on non-relational datastores, including Hadoop and NoSQL

39.4.18 Kyoto Cabinet

Kyoto Cabinet as specified in [363] is a library of routines for managing a database which is a simple data file containing records. Each record in the database is a pair of a key and a value. Every key and value is serial bytes with variable length. Both binary data and character string can be used as a key and a value. Each key must be unique within a database. There is neither concept of data tables nor data types. Records are organized in hash table or B+ tree. Kyoto Cabinet runs very fast. The elapsed time to store one million records is 0.9 seconds for hash database, and 1.1 seconds for B+ tree database. Moreover, the size of database is very small. The overhead for a record is 16 bytes for hash database, and 4 bytes for B+ tree database. Furthermore, scalability of Kyoto Cabinet is great. The database size can be up to 8EB (9.22e18 bytes).

39.4.19 Pig

39.4.20 Sawzall

Google engineers created the domain-specific programming language (DSL) *Sawzall* as a productivity enhancement tool for Google employees. They targeted the analysis of large data sets with flat, but regular, structures spread across numerous servers. The authors designed it to handle “simple, easily distributed computations: filtering, aggregation, extraction of statistics,” etc. from the aforementioned data sets. [470]

In general terms, a Sawzall job works as follows: multiple computers each create a Sawzall instance, perform some operation on a single record out of (potentially) petabytes of data, return the result to an aggregator function on a different computer and then shut down the Sawzall instance.

The engineer’s focus on simplicity and parallelization led to unconventional design choices. For instance, in contrast to most programming languages Sawzall operates on one data record at a time; it does not even preserve state between records. [521] Additionally, the language provides just a single primitive result function, the *emit* statement. The emitter returns a value from the Sawzall program to a designated virtual receptacle, generally some type of aggregator. In another example of pursuing language simplicity and parallelization, the aggregators remain separate from the formal Sawzall language (they are written in C++) because “some of the aggregation algorithms are sophisticated and best implemented in a native language [and] [m]ore important[ly] drawing an explicit line between filtering and aggregation enables a high degree of parallelism, even though it hides the parallelism from the language itself” [470].

Important components of the Sawzall language include: *szl*, the binary containing the code compiler and byte-code interpreter that executes the program; the *libszl* library, which compiles and executes Sawzall programs “[w]hen szl is used as part of another program, e.g. in a [map-reduce] program”; the Sawzall language plugin, designated *protoc_gen_szl*, which generates Sawzall code when run in conjunction with Google’s own *protoc* protocol compiler; and libraries for intrinsic functions as well as Sawzall’s associated aggregation functionality. [21]

39.4.21 Google Cloud DataFlow

Google Cloud DataFlow [238] is a unified programming model that manages the deployment, maintenance and optimization of data processes such as batch processing, ETL etc. It creates a pipeline of tasks and dynamically allocates resources thereby maintaining high efficiency and low latency. According to [238], these capabilities make it suitable for solving challenging big data problems. Also, google DataFlow overcomes the performance issues faced by Hadoop Mapreduce while building pipelines. As stated in [392] the performance of MapReduce started deteriorating while facing multiple petabytes of data whereas Google Cloud Dataflow is apparently better at handling enormous datasets. [238]. Additionally Google Dataflow can be integrated with Cloud Storage, Cloud Pub/Sub, Cloud Datastore, Cloud Bigtable, and BigQuery. The unified programming ability is another noteworthy feature which uses Apache Beam SDKs to support powerful operations like windowing and allows correctness control to be applied to batch and stream data processes.

39.4.22 Summingbird

According to [100], “Summingbird is a library that lets you write MapReduce programs that look like native Scala or Java collection transformations and execute them on a number of well-known distributed MapReduce platforms, including Storm and Scalding.” Summingbird is open-source and

is a domain-specific Scala implemented language [101]. It combines online and batch MapReduce computations into one framework [101]. It utilizes the platforms Hadoop for batch and Storm for online process execution [101]. The open-source Hadoop implementation of MapReduce is a tool which those responsible for data management use to handle problems related to big data [101]. Summingbird uses an algebraic structure called a commutative semigroup to perform aggregations of both batch and online processes [101]. A commutative semigroup is a particular type of semigroup “where the associated binary operation is also commutative” [101]. The types of data that Summingbird takes as inputs are streams and snapshots [101]. The types of data Summingbird jobs generate are called stores and sinks [101]. Stores are “an abstract model of a key-value store” while sinks are unaggregated tuples from a producer [101]. Summingbird aims to simplify the process of both batch and online analytics by exploiting “the formal properties of algebraic structures” to integrate the various modes of distributed processing [101].

39.4.23 Lumberyard

It is powerful and full-featured enough to develop triple-A, current-gen console games and is deeply integrated with AWS and Twitch(an online steaming service) [432]. Lumberyard’s core engine technology is based on Crytek’s CryEngine [216]. The goal is ”creating experiences that embrace the notion of a player, broadcaster, and viewer all joining together”[432]. Monetization for Lumberyard will come strictly through the use of Amazon Web Services’ cloud computing. If you use the engine for your game, you’re permitted to roll your own server tech, but if you’re using a third-party provider, it has to be Amazon [639].

39.5 Streams

39.5.1 Storm

Apache Storm is an open source distributed computing framework for analyzing big data in real time [314]. refers storm as the Hadoop of real time data. Storm operates by reading real time input data from one end and passes it through a sequence of processing units delivering output at the other end. The basic element of Storm is called topology. A topology consists of many other elements interconnected in a sequential fashion. Storm allows us to define and submit topologies written in any programming language.

Once under execution, a storm topology runs indefinitely unless killed explicitly. The key elements in a topology are the spout and the bolt. A spout is a source of input which can read data from various datasources and passes it to a bolt. A bolt is the actual processing unit that processes data and produces a new output stream. An output stream from a bolt can be given as an input to another bolt [576].

39.5.2 S4

S4 [48] is a distributed, scalable, fault-tolerant, pluggable platform that allows programmers to easily develop applications for processing continuous unbounded streams of data. It is built on similar concept of key-value pairs like the MapReduce. The core platform is written in Java [523]. S4 provides a runtime distributed platform that handles communication, scheduling and distribution across containers. The containers are called S4 nodes. The data is executed and processed on these S4 nodes. These S4 nodes are then deployed on S4 clusters. The user develops applications and deploys them on S4 clusters for its processing. The applications are built as a graph of Processing Elements (PEs) and Stream that interconnects the PEs. All PEs communicate asynchronously by

sending events on streams. Events are dispatched to nodes according to their key in the program [48]. All nodes are symmetric with no centralized service and no single point of failure. Additionally there is no limit on the number of nodes that can be supported. [389]. In S4, both the platform and the applications are built by dependency injection, and configured through independent modules.

39.5.3 Samza

Apache Samza is an open-source near-realtime, asynchronous computational framework for stream processing developed by the Apache Software Foundation in Scala and Java [49]. Apache Samza is a distributed stream processing framework. It uses Apache Kafka for messaging, and Apache Hadoop YARN to provide fault tolerance, processor isolation, security, and resource management. Samza processes streams. A stream is composed of immutable messages of a similar type or category. Messages can be appended to a stream or read from a stream. Samza supports pluggable systems that implement the stream abstraction: in Kafka a stream is a topic, in a database we might read a stream by consuming updates from a table, in Hadoop we might tail a directory of files in HDFS. Samza is a stream processing framework. Samza provides a very simple callback-based *process message* API comparable to MapReduce. Samza manages snapshotting and restoration of a stream processor's state. Samza is built to handle large amounts of state (many gigabytes per partition) [526]. Whenever a machine in the cluster fails, Samza works with YARN to transparently migrate your tasks to another machine. Samza uses Kafka to guarantee that messages are processed in the order they were written to a partition, and that no messages are ever lost. Samza is partitioned and distributed at every level. Kafka provides ordered, partitioned, replayable, fault-tolerant streams. YARN provides a distributed environment for Samza containers to run in. Samza works with Apache YARN, which supports Hadoop's security model, and resource isolation through Linux CGroups [50] [49].

39.5.4 Granules

Granules are used for execution or processing of data streams in distributed environment. When applications are running concurrently on multiple computational resources, granules manage their parallel execution. The MapReduce implementation in Granules is responsible for providing better performance. It has the capability of expressing computations like graphs. Computations can be scheduled based on periodicity or other activity. Computations can be developed in C, C++, Java, Python, C#, R. It also provides support for extending basic Map reduce framework. Its application domains include hand writing recognition, bio informatics and computer brain interface [544].

39.5.5 Neptune

39.5.6 Google MillWheel

MillWheel is a framework for building low-latency data-processing applications. Users specify a directed computation graph and application code for individual nodes, and the system manages persistent state and the continuous flow of records, all within the envelope of the framework's fault-tolerance guarantees. Other streaming systems do not provide this combination of fault tolerance, versatility, and scalability. MillWheel allows for complex streaming systems to be created without distributed systems expertise. MillWheel's programming model provides a notion of logical time, making it simple to write time-based aggregations. MillWheel was designed from the outset with fault tolerance and scalability in mind. In practice, we find that MillWheel's unique combination of scalability, fault tolerance, and a versatile programming model [17].

39.5.7 Amazon Kinesis

Kinesis is Amazon's [354] real time data processing engine. It is designed to provide scalable, durable and reliable data processing platform with low latency. The data to Kinesis can be ingested from multiple sources in different format. This data is further made available by Kinesis to multiple applications or consumers interested in the data. Kinesis provides robust and fault tolerant system to handle this high volume of data. Data sharding mechanism in Kinesis makes it horizontally scalable. Each of these shards in Kinesis process a group of records which are partitioned by the shard key. Each record processed by Kinesis is identified by sequence number, partition key and data blob. Sequence number to records is assigned by the stream. Partition keys are used by partitioner(a hash function) to map the records to the shards i.e. which records should go to which shard. Producers like web servers, client applications, logs push the data to Kinesis whereas Kinesis applications act as consumers of the data from Kinesis engine. It also provides data retention for certain time for example 24 hours default. This data retention window is a sliding window. Kinesis collects lot of metrics which can be used to understand the amount of data being processed by Kinesis. User can use this metrics to do some analytics and visualize the metrics data. Kinesis is one of the tools part of AWS infrastructure and provides its users a complete software-as-a-service. Kinesis [542] in the area of real-time processing provides following key benefits: ease of use, parallel processing, scalable, cost effective, fault tolerant and highly available.

39.5.8 LinkedIn

LinkedIn is a social networking website for Business and employment [703]. LinkedIn has more than 400 million user profiles (as per 10 March 2016 news), and increasing at a rate of 2 new member every second [157]. LinkedIn provides different products like:

- People You May Know - Skill Endorsements - Jobs You May Be Interested In - News Feed Updates

Such products are based on big data. To achieve such big data tasks, LinkedIn has its ecosystem consist of Oracle, Hadoop, Pig, Hive, Azkaban (Workflow), Avro Data, Zookeeper, Aster Data, Data In- Apache Kafka, Data Out- Apache Kafka and Voldemort [157]. LinkedIn uses Hadoop and Aster Data as an analytics layer [498]. LinkedIn partitioned the user's data into separate DB's stored in XML format. Voldemort is a key lookup system used to store the analytically-derived data for the products like "People You May Know". Voldemort stores the data in key-value form [498]. LinkedIn has exposed REST API to get the user data [375].

39.5.9 Twitter Heron

Heron is a real-time analytics platform that was developed at Twitter for distributed streaming processing. Heron was introduced at SIGMOD 2015 to overcome the shortcomings of Twitter Storm as the scale and diversity of Twitter data increased. As mentioned in [629] The primary advantages of Heron were: API compatible with Storm: Back compatibility with Twitter Storm reduced migration time. Task-Isolation: Every task runs in process-level isolation, making it easy to debug/ profile. Use of main stream languages: C++, Java, Python for efficiency, maintainability, and easier community adoption. Support for backpressure: dynamically adjusts the rate of data flow in a topology during run-time, to ensure data accuracy. Batching of tuples: Amortizing the cost of transferring tuples. Efficiency: Reduce resource consumption by 2-5x and Heron latency is 5-15x lower than Storm's latency. The architecture of Heron (as shown in [628]) uses the Storm API to submit topologies to a scheduler. The scheduler runs each topology as a job consisting of several containers. The containers run the topology master, stream manager, metrics manager and Heron

instances. These containers are managed by the scheduler depending on resource availability.

39.5.10 Databus

39.5.11 Facebook Puma/Ptail/Scribe/ODS

The real time data Processing at Facebook is carried out using the technologies like Scribe, Ptail, Puma, and ODS. While designing the system, facebook primarily focused on the five key decisions that the system should incorporate which were Ease of Use, Performance, Fault-tolerance, Scalability, and Correctness. “The real time data analytics ecosystem at facebook is designed to handle hundreds of Gigabytes of data per second via hundreds of data pipelines and this system handles over 200,000 events per second with a maximum latency of 30 seconds” [186]. Facebook focused on the Seconds of latency while designing the system and not milliseconds as seconds are fast enough to for all the use case that needs to be supported, and it allowed facebook to use persistent message bus for data transport and this also made the system more fault tolerant and scalable [186]. The large infrastructure of facebook comprises of hundreds of systems distributed across multiple data centers that needs a continuous monitoring to track their health and performance which is done by Operational Data Store(ODS) [116]. ODS comprises of a time series database (TSDB), which is a query service, and a detection and alerting system. ODS’s TSDB is built atop the HBase storage system. Time series data from services running on Facebook hosts is collected by the ODS write service and written to HBase.

When the data is generated by the user from their devices, an AJAX request is fired to facebook, and these requests are then written to a log file using Scribe (distributed data transport system), this messaging system collects, aggregates, and delivers high volume of log data with few seconds of latency and high throughput. Scribe stores the data in the HDFS (Hadoop Distributed File System) in a tailing fashion, where the new events are stored in log files and the files are tailed below the current events. The events are then written into the storage HBase on distributed machines. This makes the data available for both batch and real-time processing. Ptail is an internal tool built to aggregate data from multiple Scribe stores. It then tails the log files and pulls data out for processing. Puma is a stream processing system which is the real-time aggregation/storage of data. Puma provides filtering and processing of Scribe streams (with a few seconds delay), usually Puma batches the storage per 1.5 seconds on average and when the last flush completes, then only a new batch starts to avoid the contention issues, which makes it fairly real time.

39.5.12 Azure Stream Analytics

Azure Stream Analytics is a platform that manages data streaming from devices, web sites, infrastructure systems, social media, internet of things analytics, and other sources using real-time event processing engine [82]. Jobs are authored by “specifying the input source of the streaming data, the output sink for the results of your job, and a data transformation expressed in a SQL-like language.” Some key capabilities and benefits include ease of use, scalability, reliability, repeatability, quick recovery, low cost, reference data use, user defined functions capability, and connectivity. [161] Available documentation to get started with Azure Stream Analytics. [80] Azure Stream Analytics has a development project available on github.

39.5.13 Floe**39.5.14 Spark Streaming (567)**

Spark Streaming is a library built on top of Spark Core which enables Spark to process real-time streaming data. The streaming jobs can be written similar to batch jobs in Spark, using either Java, Scala or Python. The input to Spark Streaming applications can be fed from multiple data sources such HDFS, Kafka, Flume, Twitter, ZeroMQ, or custom-defined sources. It also provides a basic abstraction called Discretized Streams or DStreams to represent the continuous data streams. Spark's API for manipulating these data streams is very similar to the Spark Core's Resilient Distributed Dataset(RDD) API [563] which makes it easier for users to move between projects with stored and real-time data as the learning curve is short. Spark Streaming is designed to provide fault-tolerance, throughput, and scalability. Examples of streaming data are messages being published to a queue for real-time flight status update or the log files for a production server.

39.5.15 Flink Streaming**39.5.16 DataTurbine**

Data Turbine [711] is open source engine that allows to stream data from various sources, process it and sink it to different destinations. The streaming sources can be labs, web cams and Java enabled cell phones. The sinks can be visualizations, interfaces and databases. Data Turbine can be used to stream data formats like numbers, text, sound and video.

[207] explains that the Data Turbine middleware provides the cyber-infrastructure that integrates disparate elements of complex distributed real time application. Data Turbine acts as a middleware black box using which applications and devices can send and receive data. Data Turbine manages the management operations like memory and file management as well as book-keeping and reconnection logic. Data Turbine also provides Android based controller which allows algorithms to run close to sensors.

39.6 Basic Programming Model and Runtime, SPMD, MapReduce**39.6.1 Hadoop**

Apache Hadoop is an open source framework written in Java that utilizes distributed storage and the MapReduce programming model for processing of big data. Hadoop utilizes commodity hardware to build fault tolerant clusters. Hadoop was developed based on papers published by Google on the Google File System (2003) and MapReduce (2004) [682].

Hadoop consists of several modules: the Cluster, Storage, Hadoop Distributed File System (HDFS) Federation, Yarn Infrastructure, MapReduce Framework, and the Hadoop Common Package. The Cluster is comprised of multiple machines, otherwise referred to as nodes. Storage is typically in the HDFS. HDFS federation is the framework responsible for this storage layer. YARN Infrastructure provides computational resources such as CPU and memory. The MapReduce layer is responsible for implementing MapReduce [136]. The Hadoop Common Package which includes operating and file system abstractions and JAR files needed to start Hadoop [682].

39.6.2 Spark (543)

Apache Spark which is an open source cluster computing framework has emerged as the next generation big data processing engine surpassing Hadoop MapReduce. “Spark engine is developed for in-memory processing as well a disk based processing. This system also provides large number of impressive high level tools such as machine learning tool M Lib, structured data processing, Spark SQL, graph processing took Graph X, stream processing engine called Spark Streaming, and Shark for fast interactive question device.” The ability of spark to join datasets across various heterogeneous data sources is one of its prized attributes. Apache Spark is not the most suitable data analysis engine when it comes to processing (1) data streams where latency is the most crucial aspect and (2) when the available memory for processing is restricted. “When available memory is very limited, Apache Hadoop Map Reduce may help better, considering huge performance gap.” In cases where latency is the most crucial aspect we can get better results using Apache Storm.

39.6.3 Twister

Twister is a new software tool released by Indiana University, which is an extension to MapReduce architectures currently used in the academia and industry [320]. It supports faster execution of many data mining applications implemented as MapReduce programs. Applications that currently use Twister include: K-means clustering, Google’s page rank, Breadth first graph search , Matrix multiplication, and Multidimensional scaling. Twister also builds on the SALSA team’s work related to commercial MapReduce runtimes, including Microsoft Dryad software and open source Hadoop software. SALSA project work is funded in part by an award from Microsoft, Inc. The architecture is based on pub/sub messaging that enables it to perform faster data transfers, minimizing the overhead of the runtime. Also, the support for long running processes improves the efficiency of the runtime for many iterative MapReduce computations. [627] [319] [733].

39.6.4 MR-MPI

[409] MR-MPI stands for Map Reduce-Message Passing Interface is open source library build on top of standard MPI. It basically implements mapReduce operation providing a interface for user to simplify writing mapReduce program. It is written in C++ and needs to be linked to MPI library in order to make the basic map reduce functionality to be executed in parallel on distributed memory architecture. It provides interface for c, c++ and python. Using C interface the library can also be called from Fortrain.

39.6.5 Stratosphere (Apache Flink)

Apache Flink is an open-source stream processing framework for distributed, high-performing, always-available, and accurate data streaming applications. Apache Flink is used in big data application primarily involving analysis of data stored in Hadoop clusters. It also supports a combination of in-memory and disk-based processing as well as handles both batch and stream processing jobs, with data streaming the default implementation and batch jobs running as special-case versions of streaming application [681].

39.6.6 Reef

REEF (Retainable Evaluator Execution Framework) [107] is a scale-out computing fabric that eases the development of Big Data applications on top of resource managers such as Apache YARN and Mesos. It is a Big Data system that makes it easy to implement scalable, fault-tolerant runtime

environments for a range of data processing models on top of resource managers. REEF provides capabilities to run multiple heterogeneous frameworks and workflows of those efficiently. REEF contains two libraries, Wake and Tang where Wake is an event-based-programming framework inspired by Rx and SEDA and Tang is a dependency injection framework inspired by Google Guice, but designed specifically for configuring distributed systems.

39.6.7 Disco

- a. Disco from discoproject.org represents an implementation of mapreduce for distributed computing that benefits end users by relieving them of the need to handle “difficult technicalities related to distribution such as communication protocols, load balancing, locking, job scheduling, and fault tolerance.” [599] Its designers wrote the software in Erlang, an inherently fault tolerant language. In addition, Disco’s creators chose Erlang because they believe it best meets the software’s need to handle “tens of thousands of tasks in parallel.” [600] Python was used for Disco’s libraries. Finally, Disco supports pipelines, “a linear sequence of stages, where the outputs of each stage are grouped into the input of the subsequent stage.” [121] Its designers implemented Disco’s libraries in Python. Disco originated within Nokia Corp. to handle large data sets. Since then it has proven itself reliable in production environments outside of Nokia. [428]
- b. DISCO from the research group Service Engineering (SE), [[www-discoabout-discoabstractionlayer](#)] serves as “an abstraction layer for OpenStack’s orchestration component [Heat]” SE based DISCO on its prior orchestration framework, Hurtle. The software sets up a computer cluster and deploys the user’s choice of distributed computing architecture onto the cluster based on setup inputs provided by the user. DISCO offers a command line interface via HTTP to directly access OpenStack. [[www-discodescribed-discoabstractionlayer](#)]

39.6.8 Hama

Apache Hama is a framework for Big Data analytics which uses the Bulk Synchronous Parallel (BSP) computing model, which was established in 2012 as a Top-Level Project of The Apache Software Foundation. It provides not only pure BSP programming model but also vertex and neuron centric programming models, inspired by Google’s Pregel and DistBelief [35]. It avoids the processing overhead of MapReduce approach such as sorting, shuffling, reducing the vertices etc. Hama provides a message passing interface and each superstep in BSP is faster than a full job execution in MApReduce framework, such as Hadoop [463].

39.6.9 Giraph

Apache Giraph is an iterative graph processing system built for big data [42]. It utilizes Hadoop Mapreduce technology for processing graphs [43] Giraph was initially developed by Yahoo based on the paper published by Google on Pregel. [185] Facebook with some improvements on Giraph could analyze real world graphs up to a scale of a trillion. Giraph can directly interface with HDFS and Hive (As it’s developed in Java). [530]

39.6.10 Pregel

39.6.11 Pegasus

See Section 39.1.4.

39.6.12 Ligra

Ligra is a Light Weight Graph Processing Framework for the graph manipulation and analysis in shared memory system. It is particularly suited for implementing on parallel graph traversal algorithms where only a subset of the vertices are processed in an iteration. The interface is lightweight as it supplies only a few functions. The Ligra framework has two very simple routines, one for mapping over edges and one for mapping over vertices.

The implementations of several graph algorithms like BFS, breadth-first search, betweenness centrality, graph radii estimation, graph-connectivity, PageRank and Bellman-Ford single-source shortest paths efficient and scalable, and often achieve better running times than ones reported by other graph libraries/systems [545]. Although the shared memory machines cannot be scaled to the same size as distributed memory clusters, but the current commodity single unit servers can easily fit graphs with well over a hundred billion edges in the shared memory systems that are large enough for any of the graphs reported in the paper [339].

39.6.13 GraphChi

GraphChi is a disk-based system for computing efficiently on graphs with large number of edges. It uses a well-known method to break large graphs into small parts, and executes data mining, graph mining, machine learning algorithms. GraphChi can process over one hundred thousand graph updates per second, while simultaneously performing computation [365]. GraphChi is a spin-off of the GraphLab. GraphChi brings web-scale graph computation, such as analysis of social networks, available to anyone with a modern laptop

39.6.14 Galois

Galois system was built by intelligent software systems team at University of Texas, Austin. As explained in [634], “Galois is a system that automatically executes ‘Galoized’ serial C++ or Java code in parallel on shared-memory machines. It works by exploiting amorphous data-parallelism, which is present even in irregular codes that are organized around pointer-based data structures such as graphs and trees”. By using Galois provided data structures programmers can write serial programs that gives the performance of parallel execution. Galois employs annotations at loop levels to understand correct context during concurrent execution and executes the code that could be run in parallel. The key idea behind Galois is Tao-analysis, in which parallelism is exploited at compile time rather than at run time by creating operators equivalent of the code by employing data driven local computation algorithm [471]. Galois currently supports C++ and Java.

39.6.15 Medusa-GPU

Graphs are commonly used data structures . However, developers may find it challenging to write correct and efficient programs. Furthermore, graph processing is further complicated by irregularities of graph structures. Medusa enables the developers to write sequential C/C++ code. According to [329] it provides a set of APIs which embraces a runtime system to automatically execute those APIs in parallel. A number of optimization techniques are implemented to improvise the efficiency of graph processing. The experimental results provided in the paper [329] demonstrate that (1) Medusa greatly simplifies implementation of GPGPU programs for graph processing, with many fewer lines of source code written by developers; (2) The optimization techniques significantly improve the performance of the runtime system, making its performance comparable with or better than manually tuned GPU graph operations. [108] Medusa has proved to be a powerful framework

for networked digital audio and video framework. [108] By exploiting the APIs it takes a modular approach to construct complex graph systems.

39.6.16 MapGraph

39.6.17 Totem

Totem is a project to overcome the current challenges in graph algorithms. The project is research the Networked Systems Laboratory (NetSysLab) The issue resides in the scale of real world graphs and the inability to process them on platforms other than a supercomputer. Totem is based on a bulk synchronous parallel(BSP) model that can enable hybrid CPU/GPU systems to process graph based applications in a cost effective manner. [423]

39.7 Inter process communication Collectives

39.7.1 point-to-point

39.7.2 (a) publish-subscribe: MPI

see

- <http://www.slideshare.net/Foxsden/high-performance-processing-of-streaming-data>

39.7.3 (b) publish-subscribe: Big Data

Publish/Subscribe (Pub/Sub) [538] is a communication paradigm in which subscribers register their interest as a pattern of events or topics and then asynchronously receive events matching their interest. On the other hand, publishers generate events that are delivered to subscribers with matching interests. In Pub/sub systems, publishers and subscribers need not know each other. Pub/sub technology is widely used for a loosely coupled interaction between disparate publishing data-sources and numerous subscribing data-sinks. The two most widely used pub/sub schemes are - Topic-Based Publish/Subscribe (TBPS) and Content-Based Publish/Subscribe (CBPS) [182].

Big Data analytics architecture are being built on top of a publish/subscribe service stratum, serving as the communication facility used to exchange data among the involved components [181]. Such a publish/subscribe service stratum brilliantly solves several interoperability issues due to the heterogeneity of the data to be handled in typical Big Data scenarios.

Pub/Sub systems are being widely deployed in Centralized datacenters, P2P environments, RSS feed notifications, financial data dissemination, business process management, Social interaction message notifications- Facebook, Twitter, Spotify, etc.

39.7.4 HPX-5

Based on [713], High Performance ParallelX (HPX-5) is an open source, distributed model that provides opportunity for operations to run unmodified on one-to-many nodes. The dynamic nature of the model accommodates effective “computing resource management and task scheduling”. It is portable and performance-oriented. HPX-5 was developed by IU Center for Research in Extreme Scale Technologies (CREST). Concurrency is provided by lightweight control object (LCO) synchronization and asynchronous remote procedure calls. ParallelX component allows for termination detection and supplies per-process collectives. It “addresses the challenges of starvation, latency, overhead, waiting, energy and reliability”. Finally, it supports OpenCL to use

distributed GPU and coprocessors. HPX-5 could be compiled on various OS platforms , however it was only tested on several Linux and Darwin (10.11) platforms. Required configurations and environments could be accessed via [714].

39.7.5 Argo BEAST HPX-5 BEAST PULSAR

Search on the internet was not successsful.

39.7.6 Harp

Harp [271] is a simple, easy to maintain, low risk and easy to scale static web server that also serves Jade, Markdown, EJS, Less, Stylus, Sass, and CoffeeScript as HTML, CSS, and JavaScript without any configuration and requires low cognitive overhead. It supports the beloved layout/partial paradigm and it has flexible metadata and global objects for traversing the file system and injecting custom data into templates. It acts like a lightweight web server that was powerful enough for me to abandon web frameworks for dead simple front-end publishing. Harp can also compile your project down to static assets for hosting behind any valid HTTP server.

39.7.7 Netty

Netty [424] “is an asynchronous event-driven network application framework for rapid development of maintainable high performance protocol servers & clients”. Netty [390] “is more than a collection of interfaces and classes; it also defines an architectural model and a rich set of design patterns”. It is protocol agnostic, supports both connection oriented protocols using TCP and connection less protocols built using UDP. Netty offers performance superior to standard Java NIO API thanks to optimized resource management, pooling and reuse and low memory copying.

39.7.8 ZeroMQ

In [141], ZeroMQ is introduced as a software product that can “connect your code in any language, on any platform” by leveraging “smart patterns like pub-sub, push-pull, and router-dealer” to carry “messages across inproc, IPC, TCP, TIPC, [and] multicast.” In [140], it is explained that ZeroMQ’s “asynchronous I/O model“ causes this “tiny library” to be ”fast enough to be the fabric for clustered products.“ In [141], it is made clear that ZeroMQ is “backed by a large and open source community” with ”full commercial support.” In contrast to Message Passing Interface (i.e. MPI), which is popular among parallel scientific applications, ZeroMQ is designed as a fault tolerant method to communicate across highly distributed systems.

39.7.9 ActiveMQ

Apache ActiveMQ is a powerful open source messaging and Integration Patterns server [11]. It is a message oriented middleware(MOM) for the Apache Software Foundation that provides high availability, reliability, performance, scalability and security for enterprise messaging [579]. The goal of ActiveMQ is to provide standard-based, message-oriented application integration across as many languages and platforms as possible. ActiveMQ implements the JMS spec and offers dozens of additional features and value on top of this specifications. ActiveMQ is used in many scenarios such as heterogeneous application integration, as a replacement for RPC and to loosen the coupling between applications.

39.7.10 RabbitMQ

RabbitMQ is a message broker [500] which allows services to exchange messages in a fault tolerant manner. It provides variety of features which “enables software applications to connect and scale”. Features are: reliability, flexible routing, clustering, federation, highly available queues, multi-protocol, many clients, management UI, tracing, plugin system, commercial support, large community and user base. RabbitMQ can work in multiple scenarios:

1. Simple messaging: producers write messages to the queue and consumers read messages from the queue. This is synonymous to a simple message queue.
2. Producer-consumer: Producers produce messages and consumers receive messages from the queue. The messages are delivered to multiple consumers in round robin manner.
3. Publish-subscribe: Producers publish messages to exchanges and consumers subscribe to these exchanges. Consumers receive those messages when the messages are available in those exchanges.
4. Routing: In this mode consumers can subscribe to a subset of messages instead of receiving all messages from the queue.
5. Topics: Producers can produce messages to a topic multiple consumers registered to receive messages from those topics get those messages. These topics can be handled by a single exchange or multiple exchanges.
6. RPC: In this mode the client sends messages as well as registers a callback message queue. The consumers consume the message and post the response message to the callback queue.

RabbitMQ is based on AMPQ [436] (Advanced Message Queuing Protocol) messaging model. AMPQ is described as follows “messages are published to exchanges, which are often compared to post offices or mailboxes. Exchanges then distribute message copies to queues using rules called bindings. Then AMQP brokers either deliver messages to consumers subscribed to queues, or consumers fetch/pull messages from queues on demand”

39.7.11 NaradaBrokering

NaradaBrokering [133], is a content distribution infrastructure for voluminous data streams. The substrate places no limits on the size, rate and scope of the information encapsulated within these streams or on the number of entities within the system. The smallest unit of this substrate called as broker, intelligently process and route messages, while working with multiple underlying communication protocols. The major capabilities of NaradaBrokering consists of providing a message oriented middleware (MoM) which facilitates communications between entities (which includes clients, resources, services and proxies thereto) through the exchange of messages and providing a notification framework by efficiently routing messages from the originators to only the registered consumers of the message in question [208]. Also, it provides salient stream oriented features such as their Secure end-to-end delivery, Robust disseminations, jitter reductions.

NaradaBrokering incorporates support for several communication protocol such as TCP, UDP, Multicast, HTTP, SSL, IPSec and Parallel TCP as well as supports enterprise messaging standards such as the Java Message Service, and a slew of Web Service specifications such as SOAP, WS-Eventing, WS-Reliable Messaging and WS-Reliability [132].

39.7.12 QPid

39.7.13 Kafka

Apache Kafka is a streaming platform, which works based on publish-subscribe messaging system and supports distributed environment.

Kafka lets you publish and subscribe to the messages. Kafka maintains message feeds based on ‘topic’. A topic is a category or feed name to which records are published. Kafka’s Connector APIs are used to publish the messages to one or more topics, whereas, Consumer APIs are used to subscribe to the topics.

Kafka lets you process the stream of data at real time. Kafka’s stream processor takes continual stream of data from input topics, processes the data in real time and produces streams of data to output topics. Kafka’s Streams API are used for data transformation.

Kafka lets you store the stream of data in distributed clusters. Kafka acts as a storage system for incoming data stream. As Kafka is a distributed system, data streams are partitioned and replicated across nodes.

Thus, a combination of messaging, storage and processing data stream makes Kafka a ‘streaming platform’. It can be used for building data pipelines where data is transferred between systems or applications. Kafka can also be used by applications that transform real time incoming data. [201]

39.7.14 Kestrel

Kestrel is a distributed message queue, with added features and bulletproofing, as well as the scalability offered by actors and the Java virtual machine. It supports multiple protocols: memcache: the memcache protocol; thrift: Apache Thrift-based RPC; text: a simple text-based protocol. Each queue is strictly ordered following the FIFO (first in, first out) principle. To keep up with performance items are cached in system memory. Kestrel is more durable as queues are stored in memory for speed, but logged into a journal on disk so that servers can be shutdown or moved without losing any data. When kestrel starts up, it scans the journal folder and creates queues based on any journal files it finds there, to restore state to the way it was when it last shutdown (or was killed or died).

Kestrel uses a pull-based data aggregator system that convey data without prior definition on its destination. So the destination can be defined later on either storage system, like HDFS or NoSQL, or processing system, like storm and spark streaming. Each server handles a set of reliable, ordered message queues. When you put a cluster of these servers together, with no cross communication, and pick a server at random whenever you do a set or get, you end up with a reliable, loosely ordered message queue [349].

39.7.15 JMS

JMS (Java Messaging Service) is a java oriented messaging standard that defines a set of interfaces and semantics which allows applications to send, receive, create, and read messages. It allows the communication between different components of a distributed application to be loosely coupled, reliable, and asynchronous [667]. JMS overcomes the drawbacks of RMI (Remote Method Invocation) where the sender needs to know the method signature of the remote object to invoke it and RPC(Remote Procedure Call), which is tightly coupled i.e it cannot function unless the sender has important information about the receiver.

JMS establishes a standard that provides loosely coupled communication i.e the sender and receiver need not be present at the same time or know anything about each other before initiating the communication. JMS provides two communication domains. A point-to-point messaging domain where there is one producer and one consumer. On generating message, a producer simply pushes the message to a message queue which is known to the consumer. The other communication domain is publish/subscribe model, where one message can have multiple receivers [333].

39.7.16 AMQP

[32] AMQP stands for Advanced Message Queueing Protocol. AMQP is open internet protocol that allows secure and reliable communication between applications in different organization and different applications which are on different platforms. AMQP allows businesses to implement middleware applications interoperability by allowing secure message transfer between the applications on timely manner. AMQP is mainly used by financial and banking business. Other sectors that also use AMQP are Defence, Telecommunication, cloud Computing and so on. Apache Qpid, StormMQ, RabbitMQ, MQlight, Microsoft's Windows Azure Service Bus, IIT Software's SwiftMQ and Joram are some of the products that implement AMQP protocol.

39.7.17 Stomp

39.7.18 MQTT

According to [407], Message Queueing Telemetry Transport (MQTT) protocol is an Interprocess communication protocol that could serve as better alternative to HTTP in certain cases. It is based on a publish-subscribe messaging pattern. Any sensor or remote machine can publish its data and any registered client can subscribe the data. A broker takes care of the message being published by the remote machine and updates the subscriber in case of new message from the remote machine. The data is sent in binary format which makes it use less bandwidth. It is designed mainly to cater to the needs of devices that have access to minimal network bandwidth and device resources without affecting reliability and quality assurance of delivery. MQTT protocol has been in use since 1999. One of the notable work is project Floodnet [408], which monitors river and floodplains through a set of sensors.

39.7.19 Marionette Collective

It is basically a framework for management of a system where the systems undergo an organized coordination resulting in an automated deployment of systems which creates an orderly workflow or a parallel wise job execution. It doesn't rely on central inventories such as SSH and uses tools such as Middleware :cite: 'www-marionette-webpage'. This gives an advantage of delivering a very scalable and quick execution environment. Mcollective gives us a huge advantage of working with a large number of servers , it uses publish/subscribe middleware for communicating with many hosts at once in a parallel manner. Mcollective allows us to interact with a cluster of servers at the same time, it allows us to use a simple command line to call remote agents and there isn't a centralized inventory. Mcollective uses a broadcast paradigm to distribute the requests , where all the servers receives the request at the same time which are also attached with a filter. The servers which match the filter will act on these requests.

39.7.20 Public Cloud: Amazon SNS

Amazon SNS is an Inter process communication service which gives the user simple, end-to-end push messaging service allowing them to send messages, alerts, or notifications. According to [558], it can be used to send a directed message intended for an entity or to broadcast messages to list of selected entities. It is an easy to use and cost effective mechanism to send push messages. Amazon SNS is compatible to send push notifications to iOS, Windows, Fire OS and Android OS devices.

According to [559] SNS system architecture consists of four elements: (1) Topics, (2) Owners, (3) Publishers, and (4) Subscribers. Topics are events or access points that identifies the subject of the event and can be accessed by an unique identifier(URI). Owners create topics and control all access to the topic and define the corresponding permission for each topic. Subscribers are clients (applications, end-users, servers, or other devices) that want to receive messages or notifications on specific topics of interest to them. Publishers send messages to topics. SNS matches the topic with the list of subscribers interested in the topic, and delivers the message to them.

According to [560], Amazon SNS follows pay as per usage. In general it is \$0.50 per 1 million Amazon SNS Requests. Amazon SNS supports notifications over multiple transport protocols such as HTTP/HTTPS, Email/Email-JSON, SQS(Message queue) and SMS. Amazon SNS can be used with other AWS services such as Amazon SQS, Amazon EC2 and Amazon S3.

39.7.21 Lambda

AWS Lambda is a product from amazon which facilitates serverless computing [77]. AWS Lambda allows for running the code without the need for provisioning or managing servers, all server management is taken care by AWS. The code to be run on AWS Lambda is called a server function which can be written in Node.js, Python, Java, C#. Each Lambda function is to be stateless and any persistent data needs are to be handled through storage devices. AWS Lambda function can be setup using the AWS Lambda console where one can setup the function code and specify the event that triggers the functional call. AWS Lamda service supports multiple event sources as identified in [78]. AWS Lambda is designed to use replication and redundancy to provide for high availability both for the service itself and the function it runs. AWS Lambda automatically scales your application by running the code in response to each trigger. The code runs in parallel and processes each trigger individually, scaling precisely with the size of the workload. Billing for AWS Lambda is based on the number of times the code executes and in 100 ms increments of the duration of the processing.

39.7.22 Google Pub Sub

Google Pub/Sub provides an asynchronous messaging facility which assists the communication between independent applications [253]. It works in real time and helps keep the two interacting systems independent. It is the same technology used by many of the Google apps like GMail, Ads, etc. and so integration with them becomes very easy. Some of the typical features it provides are: (1) Push and Pull - Google Pub/Sub integrates quickly and easily with the systems hosted on the Google Cloud Platform thereby supporting one-to-many, one-to-one and many-to-many communication, using the push and pull requests. (2) Scalability - It provides high scalability and availability even under heavy load without any degradation of latency. This is done by using a global and highly scalable design. (3) Encryption - It provides security by encryption of the stored data as well as that in transit. Other than these important features, it provides some others as well, like the usage of RESTful APIs, end-to-end acknowledgement, replicated storage, etc [252].

39.7.23 Azure Queues

Azure Queues storage is a Microsoft Azure service, providing inter -process communication by message passing [548]. A sender sends the message and a client receives and processes them. The messages are stored in a queue which can contain millions of messages, up to the total capacity limit of a storage account [79]. Each message can be up to 64 KB in size. These messages can then be accessed from anywhere in the world via authenticated calls using HTTP or HTTPS. Similar to the other message queue services, Azure Queues enables decoupling of the components [625]. It runs in an asynchronous environment where messages can be sent among the different components of an application. Thus, it provides an efficient solution for managing workflows and tasks. The messages can remain in the queue up to 7 days, and afterwards, they will be deleted automatically.

39.7.24 Event Hubs

Azure Event Hubs is a hyper-scale telemetry ingestion service. It collects, transforms, and stores millions of events. As a distributed streaming platform, it offers low latency and configurable time retention enabling one to ingest massive amounts of telemetry into the cloud and read the data from multiple applications using publish-subscribe semantics. [396] It is a highly scalable data streaming platform. Data sent to an Event Hub can be transformed and stored using any real-time analytics provider or batching/storage adapters. With the ability to provide publish-subscribe capabilities , Event Hubs serves as the “on ramp” for Big Data.

39.8 In-memory databases/caches

39.8.1 Gora (general object from NoSQL)

Gora is a in-memory data model [254] which also provides persistence to the big data. Gora provides persistence to different types of data stores. Primary goals of Gora are:

1. data persistence
2. indexing
3. data access
4. analysis
5. map reduce support

Unlike ORM models which mostly work with relational databases for example hibernate gora works for most type of data stores like documents, columnar, key value as well as relational. Gora uses beans to maintain the data in-memory and persist it on disk. Beans are defined using apache avro schema. Gora provides modules for each type of data store it supports. The mapping between bean definition and datastore is done in a mapping file which is specific to a data store. Type Gora workflow will be:

1. define the bean used as model for persistence
2. use gora compiler to compile the bean
3. create a mapping file to map bean definition to datastore
4. update gora.properties to specify the datastore to use
5. get an instance of corresponding data store using datastore factory.

Gora has a query interface to query the underlying data store. Its configuration is stored in gora.properties which should be present in classpath. In the file you can specify default data store used by Gora engine. Gora also has a CI/CD library call GoraCI which is used to write integration tests.

39.8.2 Memcached

Memcached is a free and open-source, high performance, distributed memory object caching system. [163] Although, generic in nature,it is intended for se in speeding up dynamic web applications by reducing the database load.

It can be thought of as a short term memory for your applications. Memcached is an in-memory key-value store for small chunks of arbitrary data from the results of database calls, API calls and page rendering. Its API is available in most of the popular languages. In simple terms, it allows you to take memory from parts of your system where you have more memory than you need and allocate it to parts of your system where you have less memory than you need.

39.8.3 Redis

Redis (Remote Dictionary Server) is an open source ,in-memory, key-value database which is commonly referred as a data structure server. “It is called a data structure server and not simply a key-value store because Redis implements data structure which allows keys to contain binary safe strings, hashes, sets, and sortedsets as well as lists” [383]. Redis’s better performance, easy to use and implement, and atomic manipulation of data structures lends itself to solving problems that are difficult to solve or perform poorly when implemented with traditional relational databases. “Salivator Sanfilippo (Creator of open-source database Redis) makes a strong case that Redis does not need to replace the existing database but is an excellent addition to an enterprise for new functionalities or to solve sometimes intractable problems.” [418]

A widely used use pattern for Redis is an in-memory cache for web-applications and the other being the use of pattern for REDIS for metric storage of such quantitative data such as the web page usage and user behavior on gamer leaderboards where using a bit operations on strings, Redis very efficiently stores binary information on a particular characteristics [418].The other popular Redis use pattern is a communication layer between different systems through a publish/subscribe (pub/sub for short), where one can post the message to one or more channels that can be acted upon by other systems that are subscribed to or listening to that channel for incoming messages. The Companies using REDIS includes Twitter to store the timelines of all the user , Pinterest stores the user follower graph, Github, popular web frameworks like Node.js , Django, Ruby-on-Rails etc.

39.8.4 LMDB (key value)

LMDB (Lighting memory-mapped Database) is a high performance embedded transactional database in form of a key-value store [675]. LMDB is designed around virtual memory facilities found in modern operating systems, multi-version concurrency control (MVCC) and single-level store (SLS) concepts. LMDB stores arbitrary key/data pairs as byte arrays, provides a range-based search capability, supports multiple data items for a single key and has a special mode for appending records at the end of the database (MDB_APPEND) which significantly increases its write performance compared to other similar databases.

LMDB is not a relational database [699] and strictly uses key-value store. Key-value databases allows one write at a time, the difference that LMDB highlights is that write transactions do not block readers nor do readers block writes. Also, it does allow multiple applications on the same system to open and use the store simultaneously which helps in scaling up performance [270].

39.8.5 Hazelcast

Hazelcast is a java based, in memory data grid [693]. It is open source software, released under the Apache 2.0 License [273]. Hazelcast enables predictable scaling for applications by providing in memory access to data. Hazelcast uses a grid to distribute data evenly across a cluster. Clusters allow processing and storage to scale horizontally. Hazelcast can run locally, in the cloud, in virtual machines, or in Docker containers. Hazelcast can be utilized for a wide variety of applications. It

has APIs for many programming languages including Python, Java, Scala, C++, .NET and Node.js and supports any binary languages through an Open Binary Client Protocol [693].

39.8.6 Ehcache

EHCACHE is an open-source Java-based cache. It supports distributed caching and could scale to hundred of caches. It comes with REST APIs and could be integrated with popular frameworks like Hibernate [171]. It offers storage tiers such that less frequently data could be moved to slower tiers [172]. It's XA compliant and supports two-phase commit and recovery for transactions. It's developed and maintained by Terracotta and is available under Apache 2.0 license. It conforms to Java caching standard JSR 107.

39.8.7 Infinispan

Infinispan is a highly available, extremely scalable key/value data store and data grid platform. The design perspective of infinispan is exposing a distributed, highly concurrent data structure to make the most use of modern multi-core as well as multi-processor architectures. It is mostly used as a distributed cache, but also can be used as an object database or NoSQL key/value store [312].

Infinispan is mostly used as a cache store. It is predominantly used for applications that are clustered, and requires a cache coherency for data consistency. Infinispan is written in Java and is open source. It is fully transactional. Infinispan is used to add clusterability as well as high availability to frameworks. Infinispan has many use-cases, they are: 1) it can be used as a distributed cache 2) Storage for temporal data, like web sessions, 3) Cross-JVM communication, 4) Shared storage, 5) In-memory data processing and analytics and 6) MapReduce Implementation in the In-Memory Data Grid. It is also used in research and academia as a framework for distribution execution and storage [307].

39.8.8 VoltDB

VoltDB is an in-memory database. It is an ACID-compliant RDBMS which uses a shared nothing architecture to achieve database parallelism. It includes both enterprise and community editions. VoltDB is a scale-out NewSQL relational database that supports SQL access from within pre-compiled Java stored procedures. VoltDB relies on horizontal partitioning down to the individual hardware thread to scale, k-safety (synchronous replication) to provide high availability, and a combination of continuous snapshots and command logging for durability (crash recovery) [644]. The in-memory, scale-out architecture couples the speed of traditional streaming solutions with the consistency of an operational database. This gives a simplified technology stack that delivers low-latency response times (1ms) and hundreds of thousands of transactions per second. VoltDB allows users to ingest data, analyze data, and act on data in milliseconds, allowing users to create per-person, real-time experiences [644].

39.8.9 H-Store

H-Store is an in memory and parallel database management system for on-line transaction processing (OLTP). Specifically, [287] illustrates that H-Store is a highly distributed, row-store-based relational database that runs on a cluster on shared-nothing, main memory executor nodes. As noted in [285] "the architectural and application shifts have resulted in modern OLTP databases increasingly falling short of optimal performance. In particular, the availability of multiple-cores, the abundance of main memory, the lack of user stalls, and the dominant use of stored procedures

are factors that portend a clean-slate redesign of RDBMSs".The H-store which is a complete redesign has the potential to outperform legacy OLTP databases by a significant factor. As detailed in [288] H-Store is the first implementation of a new class of parallel DBMS, called NewSQL, that provides the high-throughput and high-availability of NoSQL systems, but without giving up the transactional guarantees of a traditional DBMS. The H-Store system is able to scale out horizontally across multiple machines to improve throughput, as opposed to moving to a more powerful , more expensive machine for a single-node system.

39.9 Object-relational mapping

39.9.1 Hibernate

Hibernate is an open source project which provides object relational persistence framework for applications in Java. It is an Object relational mapping library (ORM) which provides the framework for mapping object oriented model to relational database. It provides a query language, a caching layer and Java Management Extensions (JMX) support. Databases supported by Hibernate includes DB2, Oracle, MySQL, PostgreSQL. To provide persistence services, Hibernate uses database and configuration data. For using hibernate, firstly a java class is created which represents table in the database. Then columns in database are mapped to the instance variables of created Java class. Hibernate can perform database operations like select, insert, delete and update records in table by automatically creating query. Connection management and transaction management are provided by hibernate. Hibernate saves development and debugging time in comparison to JDBC. But it is slower at runtime as it generates many SQL statements at runtime. It is database independent. For batch processing it is advisable to use JDBC over Hibernate [83]

39.9.2 OpenJPA

According to [355], Apache OpenJPA is a Java persistence project developed by The Apache Software Foundation that can either be used as Plain old Java Object (POJO) or could be used in any Java EE compliant containers. It provides object relational mapping which effectively simplifies the storing of relational dependencies among objects in databases. [336] mentions that Kodo, an implementation of Java Data Objects acted as a precursor to the development of OpenJPA. In 2006, BEA Systems donated the majority of the source code of Kodo to The Apache Software Foundation under the name OpenJPA. Being a POJO, OPenJPA can be used without needing to extend prespecified classes, implementing predefined interfaces and inclusion of annotations. OOpenJPA can be used in cases where the focus of the project is majorly on business logic and has no dependencies on enterprise frameworks. OOpenJPA can be implemented across multiple operating systems, on account of its function of cross platform support. It is written in Java and a most recent stable release came out in April 20, 2016 under the version 2.4.1 with Apache License 2.0.

39.9.3 EclipseLink

EclipseLink is an open source persistence Services project from Eclipse foundation. It is a framework which provide developers to interact with data services including database and web services, Object XML mapping etc. [672]. This is the project which was developed out of Oracle's Toplink product. The main difference is EclipseLink does not have some key enterprise feature. Eclipselink support a number of persistence standard model like JPA, JAXB, JCA and Service Data Object. Like Toplink, the ORM (Object relational model) is the technique to convert incompatible type system in Object Oriented programming language. It is a framework for storing java object

into relational database.

39.9.4 DataNucleus

DataNucleus (available under Apache 2 open source license) is a data management framework in Java. Formerly known as 'Java Persistent Objects' (JPOX) this was relaunched in 2008 as 'DataNucleus'. According to [666] DataNucleus Access Platform is a fully compliant implementation of the Java Persistent API (JPA) and Java Data Objects (JDO) specifications. It provides persistence and retrieval of data to a number of datastores using a number of APIs, with a number of query languages. In addition to object-relational mapping (ORM) it can also map and manage data from sources other than RDBMS (PostgreSQL, MySQL, Oracle, SQLServer, DB2, H2 etc.) such as Map-based (Cassandra, HBase), Graph-based (Neo4j), Documents (XLS, OOXML, XML, ODF), Web-based (Amazon S3, Google Storage, JSON), Doc-based (MongoDB) and Others (NeoDatis, LDAP). It supports the JPA (Uses JPQL Query language), JDO (Uses JDOQL Query language) and REST APIs [151]. DataNucleus products are built from a sequence of plugins where each of it is an OSGi bundle and can be used in an OSGi environment. Google App Engine uses DataNucleus as the Java persistence layer [337].

39.9.5 ODBC/JDBC

Open Database Connectivity (ODBC) is an open standard application programming interface (API) for accessing database management systems (DBMS) [443]. ODBC was developed by the SQL Access Group and released in September, 1992. Microsoft Windows was the first to provide an ODBC product. Later the versions for UNIX, OS/2, and Macintosh platforms were developed. ODBC is independent of the programming language, database system and platform.

Java Database Connectivity (JDBC) is a API developed specific to the Java programming language. JDBC was released as part of Java Development Kit (JDK) 1.1 on February 19, 1997 by Sun Microsystems [324]. The 'java.sql' and 'javax.sql' packages contain the JDBC classes. JDBC is more suitable for object oriented databases. JDBC can be used for ODBC compliant databases by using a JDBC-to-ODBC bridge.

39.10 Extraction Tools

39.10.1 UIMA

Unstructured Information Management applications (UIMA) provides a framework for content analytics. It searches unstructured data to retrieve specific targets for the user. For example, when a text document is given as input to the system, it identifies targets such as persons, places, objects and even associations. According to , [670] the UIMA architecture can be thought of as four dimensions: 1. Specifies component interfaces in analytics pipeline. 2. Describes a set of Design patterns. 3. Suggests two data representations: an in-memory representation of annotations for high-performance analytics and an XML representation of annotations for integration with remote web services. 4. Suggests development roles allowing tools to be used by users with diverse skills.

UIMA uses different, possibly mixed, approaches which include Natural Language Processing, Machine Learning, IR. UIMA supports multimodal analytics [552] which enables the system to process the resource fro various points of view. UIMA is used in several software projects such as the IBM Research's Watson uses UIMA for analyzing unstructured data and Clinical Text Analysis and Knowledge Extraction System (Apache cTAKES) which is a UIMA-based system for

information extraction from medical records.

39.10.2 Tika

“The Apache Tika toolkit detects and extracts metadata and text from over a thousand different file types (such as PPT, XLS, and PDF). All of these file types can be parsed through a single interface, making Tika useful for search engine indexing, content analysis, translation, and much more [613].”

39.11 SQL and SQL Services

39.11.1 Oracle

Oracle database is an object-relational database management system by Oracle. Following are some of the key features of Oracle [454] 1. ANSI SQL Compliance 2. Multi-version read consistency 3. Procedural extensions: PL/SQL and Java. Apart from above they are performance related features, including but not limited to: indexes, in-memory, partitioning, optimization. As of today the latest release of Oracle is [454] Oracle Database 12c Release 1: 12.1 (Patch set as of June 2013)

39.11.2 DB2

DB2 is a Relational DataBase Management System (RDBMS). Though initially introduced in 1983 by IBM to run exclusively on its MVS (Multiple Virtual Storage) mainframe platform, it was later extended to other operating systems like UNIX, Windows and Linux. It is used to store, analyze and retrieve the data and is extended with the support of Object-Oriented features and non-relational structures with XML [153]. DB2 server editions include: Advanced Enterprise Server Edition and Enterprise Server Edition (AESE / ESE) designed for mid-size to large-size business organizations, Workgroup Server Edition (WSE) designed for Workgroup or mid-size business organizations, Express -C provides the capabilities of DB2 at no charge and can run on any physical or virtual systems, Express Edition designed for entry level and mid-size business organizations, Enterprise Developer Edition offers single application developer useful to design, build and prototype the applications for deployment on the IBM server. DB2 has APIs for REXX, PL/I, COBOL, RPG, FORTRAN, C++, C, Delphi, .NET CLI, Java, Python, Perl, PHP, Ruby, and many other programming languages. DB2 also supports integration into the Eclipse and Visual Studio integrated development environments [152].

39.11.3 SQL Server

SQL Server [571] is a relational database management system from Microsoft. As of Jan 2017, SQL Server is available in below editions

1. Standard - consists of core database engine
2. Web - low cost edition for web hosting
3. Business Intelligence - includes standard edition and business intelligence tools like PowerPivot, PowerBI, Master Data Services
4. Enterprise - consists of core database engine and enterprise services like cluster manager
5. SQL Server Azure - [81] core database engine integrated with Microsoft Azure cloud platform and available in platform-as-a-service mode.

In the book [401], the technical architecture of SQL Server in OLTP(online transaction processing), hybrid cloud and business intelligence modes is explained in detail.

39.11.4 SQLite

SQLite is a severless SQL database engine whose source code resides in the public domain [569]. SQLite databases, including tables, indices, and views, reside on a single file on the disk [569]. It has a compact library, often taking up less than KiB of space, depending on the particular configuration [569]. Performance is the tradeoff with the smaller size, i.e. performance usually runs faster when given more memory [569]. SQLite transactions comply with the ACID (Atomicity, Consistency, Isolation, Durability) [385] properties [569]. SQLite does not require administration or configuration [624]. There are some limitations associated with SQLite, such as the inability to perform Right Outer Joins, read-only views, and access permissions (other than those that are associated with regular file acces permissions) [624] SQLite does not compare directly with client/server databases such as MySQL as they are both trying to solve different problems [570]. While database engines such as MySQL aim to provide a shared database, with different access permissions to different individuals/applications, SQLite has the goal of being a local repository of data for applications [570] While SQLite is not appropriate for every situation, there certainly exists situations where it can prove to be a prudent choice for data management needs [570].

39.11.5 MySQL

MySQL is a relational database management system. [413] SQL is an acronym for Structured Query Language and is a standardized language used to interact with the databases. [413] Databases provide structure to a collection of data while. [413] A database management system allows for the addition, accessing, and processing of the data stored in a database. [413] Relational databases utilize tables that are broken down into columns, representing the various fields of the table, and rows, which correspond to individual entries in the table. [284]

39.11.6 PostgreSQL

PostgreSQL is an open-source relational database management system (DBMS). It runs on all the major operating systems like Linux, Mac OSX, Windows and UNIX. It supports the ACID (Atomicity, Consistency, Isolation and Durability) properties of a conventional DBMS. It supports the standard SQL:2008 data types like INTEGER, NUMERIC, etc. besides providing native interfaces for languages such as C++, C, Java and .Net [605].

With the release of its latest version 9.5, it has included new features like the UPSERT capability, Row Level security and multiple features to support Big Data. These new features rolled out in the latest version make PostgreSQL a very strong contender for modern use. UPSERT feature has predominantly been released for the application developers in order to help them simplify their web application and software development. UPSERT is basically a shorthand of “Insert, on conflict update”. Row Level Security (RLS), as the name suggests, enables the database administrators to control which particular rows could be updated by the users. This helps in ensuring that the users do not inadvertently update rows which they are not meant to. Features such as BRIN indexing, Faster sorts, CUBE, ROLLUP and GROUPING SETS, Foreign Data Wrappers and TABLESAMPLE were added as a part of the new Big Data features. Under BRIN indexing (Block Range Indexing), PostgreSQL supports creating small but powerful indexes for large tables. Using a new algorithm called as “abbreviated keys”, PostgreSQL can sort NUMERIC data very quickly. The CUBE, ROLLUP and GROUPING clauses enable the users to use just a single query to create myriad reports at different levels of summarization. Using the concept of Foreign Data Wrappers (FDWs), PostgreSQL can be used for querying Big Data systems like Cassandra and Hadoop. The TABLESAMPLE clause allows quick statistical sample generation of huge tables without any need

to sort them [606].

39.11.7 CUBRID

CUBRID name is deduced from the combination of word CUBE(security within box) and BRIDGE(data bridge). It is an open source Relational DataBase Management System designed in C programming language with high performance, scalability and availability features. During its development by NCL, korean IT service provider the goal was to optimize database performance for web-applications. [145] Importantly most of the SQL syntax from MYSQL and ORACLE can work on cubrid.CUBRID also provides manager tool for database administration and migration tool for migrating the data from DBMS to CUBRID bridging the dbs. CUBRID enterprise version and all the tools are free and suitable database candidate for web-application development.

39.11.8 Galera Cluster

Galera cluster [215] is a type of database clustering which has all multiple masters and works on synchronous replication. At a deeper level, it was created by extending MySql replication API to provide all support for true multi master synchronous replication. This extended api is called as Write-Set Replication API and is the core of the clustering logic. Each transaction of wsrep API not only contains the record but also other meta-info to requires to commit each node separately or asynchronously. So though it seems synchronous logically but works independently on each node. The approach is also called virtually synchronous replication. This helps in directly read-write on a specific node and can lose a node without handling any complex failover scenarios (zero downtime).

39.11.9 SciDB

SciDB is an open source DBMS based on multi-dimensional array data model and runs on Linux platform. [575] The data store is optimized for mathematical operations such as linear algebra and statistical analysis. The data can be distributed across multiple nodes in a cluster.

The dimensions of the data can be either standard integers or user-defined types. Ragged arrays are also supported. The data is accessed through AQL, a SQL like language designed specifically for array operations. It supports operations such as to filter and join arrays and aggregation over the cell values. It has few similarities to Postgres in terms of user-defined scalar functions and storage manager. Old values of data are updated instead of being deleted to retain different versions of a cell. The arrays are divided into chunks and partitioned across the nodes in the cluster, with provision of caching some of them in the main memory.

39.11.10 Rasdaman

Rasdaman is an specialized database management system which adds capabilities for storage and retrieval of massive multi-dimensional array, such as sensors,image, and statistics data. [676] It is written in C++ language. For example, it can serve 1-D measurement data, 2-D satellite data, 3-D x/y/t image series and x/y/z exploration data, 4-D ocean and climate data, and much more.

[505]: Rasdaman servers provides functionality from geo service up to complex analytics which are related to spatio-temporal raster data. It also integrates smoothly with R, OpenLayers, NASA WorldWind etc. via APIs calls. It is massively used in the domains like earth, space, and social science related fields.

39.11.11 Apache Derby

[595]: Apache Derby is java based relational database system. Apache Derby has JDBC driver which can be used by Java based applications. Apache derby is part of the Apache DB subproject and licensed under Apache version 2.0.

[596]: Derby Embedded Database Engine is the database engine with JDBC and SQL as programming APIs. Client/Server functionality is achieved by Derby network server, it allows connection through TCP/IP using DRDA protocol. ij, database utility makes it possible for SQL scripts to be run on JDBC database. The dblook utility is the schema extraction tool. The sysinfo utility is used for displaying version of Java environment and Derby.

There are two deployment options for Apache Derby , embedded and Derby network server option. In embedded framework, Derby is started and stopped by the single user java application without any administration required. In the case of Derby network server configuration, Derby is started by multi user java application over TCP/IP. Since Apache Derby is written in Java, it runs on any certified JVM(Java Virtual Machine). [594]:

39.11.12 Pivotal Greenplum

Pivotal Greenplum is a commercial fully featured data warehouse. It is powered by Greenplum Database an open source initiative. “It is powered by advanced cost-based query optimizer thereby delivering high analytical query performance on large data volumes”. Pivotal Greenplum is uniquely focused on big data analytics [473].

The system consists of a master node, standy master node and segment nodes. The master node consists of the catalog information whereas the data resides on the segment nodes. The segment nodes runs on one or more segments which are modified PostgreSQL databases and are assigned a content identifier. The data is distributed among these segment nodes. The segment node also supports bult loading and unloading. The master node parses, optimizes an SQL query and dispatch it to all segment nodes. Therefore, it provides powerful and rapid analytics on petabyte scale data volumes [474].

39.11.13 Google Cloud SQL

Google Cloud SQL is a fully managed data base as service developed by Google where google manages the backup,patching and replication of the databases etc [246]. Cloud SQL database aims at developers to focus on app development leaving database admilnstitution to a minimum. This can be understood as 'My SQL on Cloud' as most of the features from MySQL 5.7 are directly supported in Cloud SQL. The service is offered with 'Pay per use' providing the flexibility and 'better performance per dollar'. Cloud SQL is scalable up to 16 processor cores and more than 100GB of RAM. [247]

39.11.14 Azure SQL

39.11.15 Amazon RDS

According to Amazon Web Services, Amazon Relation Database Service (Amazon RDS) is a web service which makes it easy to setup, operate and scale relational databases in the cloud. As mentioned in [510] It allows to create and use MySQL, Oracle, SQL Server, and PostgreSQL databases in the cloud. Thus, codes, applications and tools used with existing databases can be used with Amazon RDS. The basic components of Amazon(As listed in [509]) RDS include: DB

Instances: DB instance is an isolated database environment in the cloud. Regions and availability zones: Region is a data center location which contains Availability Zones. Availability Zone is isolated from failures in other Availability Zones. Security groups: controls access to DB instance by allowing access to IP address ranges or Amazon EC2 instances that is specified. DB parameter groups: manage configuration of DB engine by specifying engine configuration values that are applied to one or more DB instances of the same instance type. DB option groups: Simplifies data management through Oracle Application Express (APEX), SQL Server Transparent Data Encryption, and MySQL memcached support.

39.11.16 Google F1

F1 is a distributed relational database system built at Google to support the AdWords business. It is a hybrid database that combines high availability, the scalability of NoSQL systems like Bigtable, and the consistency and usability of traditional SQL databases. F1 is built on Spanner, which provides synchronous cross-datacenter replication and strong consistency [547].

F1 features include a strictly enforced schema, a powerful parallel SQL query engine, general transactions, change tracking and notification, and indexing, and is built on top of a highly-distributed storage system that scales on standard hardware in Google data centers. The store is dynamically sharded and is able to handle data center outages without data loss [546]. The synchronous cross-datacenter replication and strong consistency results in higher commit latency which can be overcome using hierarchical schema model with structured data types and through smart application design.

39.11.17 IBM dashDB

IBM dashDB is a data warehousing service hosted in cloud , This aims at integrating the data from various sources into a cloud data base. Since the data base is hosted in cloud it would have the benifits of a cloud like scalability and less maintainance. This data base can be configured as 'transaction based' or 'Analytics based' depending on the work load [3] .This is available through ibm blue mix cloud platform.

dash DB has build in analytics based on IBM Netezza Analytics in the PureData System for Analytics. Because of the build in analytics and support of in memory optimization promises better performance efficiency. This can be run alone as a standalone or can be connected to variousBI or analytic tools. [299]

39.11.18 N1QL

39.11.19 BlinkDB

39.11.20 Spark SQL

Spark SQL is Apache Spark's module for working with structured data. Spark SQL is a new module that integrates relational processing with Spark's functional programming API [597]. It is used to seamlessly mix SQL queries with Spark programs. Spark SQL lets you query structured data inside Spark programs, using either SQL or a familiar DataFrame API. it offers much tighter integration between relational and procedural processing, through a declarative DataFrame API that integrates with procedural Spark code. Spark SQL reuses the Hive frontend and metastore, giving you full compatibility with existing Hive data, queries, and UDFs by installing it alongside Hive. Spark SQL includes a cost-based optimizer, columnar storage and code generation to make queries fast

[9]. At the same time, it scales to thousands of nodes and multi hour queries using the Spark engine, which provides full mid-query fault tolerance.

39.12 NoSQL

39.12.1 Lucene

Apache Lucene [46] is a high-performance, full-featured text search engine library. It is originally written in pure Java but also has been ported to few other languages chiefly python. It is suitable for applications that requires full-text search. One of the key implementation of Lucene is Internet search engines and local, single-site searching. Another important implementation usage is its recommendation system. The core idea of Lucene is to extract text from any document that contains text (not image) field, making it format independent.

39.12.2 Solr

39.12.3 Solandra

Solandra is a highly scalable real-time search engine built on Apache Solr and Apache Cassandra. Solandra simplifies maintaining a large scale search engine, something that more and more applications need. At its core, Solandra is a tight integration of Solr and Cassandra, meaning within a single JVM both Solr and Cassandra are running, and documents are stored and distributed using Cassandra's data model. [322]

Solandra supports most out-of-the-box Solr functionality (search, facetting, highlights), multi-master (read/write to any node). It features replication, sharing, caching, and compaction managed by Cassandra. [323]

39.12.4 Voldemort

According to [642], project Voldemort, developed by LinkedIn, is a non-relational database of key-value type that supports eventual consistency. The distributed nature of the system allows pluggable data placement and provides horizontal scalability and high consistency. Replication and partitioning of data is automatic and performed on multiple servers. Independent nodes that comprise the server support transparent handling of server failure and ensure absence of a central point of failure. Essentially, Voldemort is a hashtable. It uses APIs for data replication. In memory caching allows for faster operations. It allows cluster expansion with no data rebalancing. When Voldemort performance was benchmarked with the other key-value databases such as Cassandra, Redis and HBase as well as MySQL relational database [501], the Voldemort's throughput was twice lower than MySQL and Cassandra and six times higher than HBase. Voldemort was slightly underperforming in comparison with Redis. At the same time, it demonstrated consistent linear performance in maximum throughput that supports high scalability. The read latency for Voldemort was fairly consistent and only slightly underperformed Redis. Similar tendency was observed with the read latency that puts Voldemort in the cluster of databases that require good read-write speed for workload operations. However, the same authors noted that Voldemort required creation of the node specific configuration and optimization in order to successfully run a high throughput tests. The default options were not sufficient and were quickly saturated that stall the database.

39.12.5 Riak

Riak is a set of scalable distributed NoSQL databases developed by Basho Technologies. Riak KV is a key-value [85] database with time-to-live feature so that older data is deleted automatically. It can be queried through secondary indexes, search via Apache Solr, and MapReduce. Riak TS is designed for time-series data. It co-locates related data on the same physical cluster for faster access [86]. Riak S2 is designed to store large objects like media files and software binaries [87]. The databases are available in both open source and commercial versions with multicluster replication provided only in later. REST APIs are available for these databases.

39.12.6 ZHT

According to [585], “ZHT is a zero-hop distributed hash table.” Distributed hash tables effectively break a hash table up and assign different nodes responsibility for managing different pieces of the larger hash table. [704] To retrieve a value in a distributed hash table, one needs to find the node that is responsible for the managing the key value pair of interest. [704] In general, every node that is a part of the distributed hash table has a reference to the closest two nodes in the node list. [704] In a ZHT, however, every node contains information concerning the location of every other node. [371] Through this approach, ZHT aims to provide “high availability, good fault tolerance, high throughput, and low latencies, at extreme scales of millions of nodes.” [371] Some of the defining characteristics of ZHT are that it is light-weight, allows nodes to join and leave dynamically, and utilizes replication to obtain fault tolerance among others. [371]

39.12.7 Berkeley DB

Berkeley DB is a family of open source, NoSQL key-value database libraries. [90] It provides a simple function-call API for data access and management over a number of programming languages, including C, C++, Java, Perl, Tcl, Python, and PHP. Berkeley DB is embedded because it links directly into the application and runs in the same address space as the application. [89] As a result, no inter-process communication, either over the network or between processes on the same machine, is required for database operations. It is also extremely portable and scalable, it can manage databases up to 256 terabytes in size.

[456] For data management, Berkeley DB offers advanced services, such as concurrency for many users, ACID transactions, and recovery.

Berkeley DB is used in a wide variety of products and a large number of projects, including gateways from Cisco, Web applications at Amazon.com and open-source projects such as Apache and Linux.

39.12.8 Kyoto/Tokyo Cabinet

Tokyo Cabinet [616] and Kyoto Cabinet [364] are libraries of routines for managing a database. The database normally is a simple data file containing records having a key value pair structure. Every key and value is serial bytes with variable length. Both binary data and character string can be used as a key and a value. There is no concept of data tables nor data types like RDBMS or DBMS. Records are organized in hash table, B+ tree, or fixed-length array. Tokyo and Kyoto cabinets both are developed as a successor of GDBM and QDBM which are library routines for managing database as well. Tokyo Cabinet is written in the C language, and is provided as API of C, Perl, Ruby, Java, and Lua. Tokyo Cabinet is available on platforms which have API conforming to C99 and POSIX. Whereas Kyoto Cabinet is written in the C++ language, and is provided as API

of C++, C, Java, Python, Ruby, Perl, and Lua. Kyoto Cabinet is available on platforms which have API conforming to C++03 with the TR1 library extensions. Both are free software licenced under GNU (General Public Licence). [616] actually mentions that Kyoto Cabinet is more powerful and has convenient library structure than Tokyo and recommends people to use Kyoto. Since they use key-value pair concept, you can store a record with a key and a value, delete a record using the key and even retrieve a record using the key. Both have smaller size of database file, faster processing speed and provide effective backup procedures.

39.12.9 Tycoon

Tycoon/ Kyoto Tycoon [367] is a lightweight database server developed by FLL labs and is a distributed Key-value store [578]. It is very useful in handling cache data persistent data of various applications. Kyoto Tycoon is also a package of network interface to the DBM called Kyoto Cabinet [366] which contains a library of routines for managing a database. Tycoon is composed of a sever process that manger multiple databases. This renders high concurrency enabling it to handle more than 10 thousand connections at the same time.

39.12.10 Tyrant

Tyrant provides network interfaces to the database management system called Tokyo Cabinet. Tyrant is also called as Tokyo Tyrant. Tyrant is implemented in C and it provides APIs for Perl, Ruby and C. Tyrant provides high performance and concurrent access to Tokyo Cabinet. The blog [632] explains the results of performance experiments between Tyrant and Memcached + MySQL.

Tyrant was written and maintained by FAL Labs [631]. However, according to FAL Labs, their latest product [630] Kyoto Tycoon is more powerful and convenient server than Tokyo Tyrant.

39.12.11 MongoDB

MongoDB is a NoSQL database which uses collections and documents to store data as opposed to the relational database where data is stored in tables and rows. In MongoDB a collection is a container for documents, whereas a document contains key-value pairs for storing data. As MongoDB is a NoSQL database, it supports dynamic schema design allowing documents to have different fields. The database uses a document storage and data interchange format called BSON, which provides a binary representation of JSON-like documents.

MongoDB provides high data availability by way of replication and sharding. High cost involved in data replication can be reduced by horizontal data scaling by way of shards where data is scattered across multiple servers. It reduces query cost as the query load is distributed across servers. This means that both read and write performance can be increased by adding more shards to a cluster. Which document resides on which shard is determined by the shard key of each collection.

As far as data backup and restore is concerned the default MongoDB storage engines natively support backup of complete data. For incremental backups one can use MongoRocks that is a third party tool developed by Facebook.

39.12.12 Espresso

Espresso [310] is a document-oriented distributed data serving platform that plays an important role in LinkedIn's central data pipeline. It currently powers approximately 30 LinkedIn applications including Member Profile, InMail, etc and also hosts some of its most important member data.

Espresso provides a hierarchical data model in which the databases and table schema are defined in JSON. Some of the key component of Espresso include : 1) Router: which is a stateless HTTP Proxy and also acts as a entry point for all client requests in Espresso. The Router uses local cached routing table to manage the partition among all the storage nodes within the cluster. 2) Storage Node: are the building blocks of the storage and each one of them hosts a set of partition. 3) Helix: is responsible for cluster management in Espresso. 4) Databus: are responsible for capturing change to transport source transactions in commit order.

All the above mentioned components together enable Espresso to achieve real-time secondary indexing, on-the-fly schema evolution and also a timeline consistent change capture stream.

39.12.13 CouchDB

The Apache Software Foundation makes CouchDB available as an option for those seeking an open-source, NoSQL, document-oriented database. CouchDB, or cluster of unreliable commodity hardware database, [369] stores data as a JSON-formatted document. Documents can consist of a variety of field types, e.g., text, booleans or lists, as well as metadata used by the software. [54] CouchDB does not limit the number of fields per document, and it does not require any two documents to consist of matching or even similar fields. That is, the document has structure, but the structure can vary by document. CouchDB coordinates cluster activities using the master-master mode by default, which means it does not have any one in charge of the cluster. However, a cluster can be set up to write all data to single node, which is then replicated across the cluster. Either way, the system can only offer eventual consistency. [142] CouchDB serves as the basis of Couchbase, Inc's Couchbase Server.

39.12.14 Couchbase Server

Couchbase, Inc. offers Couchbase Server (CBS) to the marketplace as a NoSQL, document-oriented database alternative to traditional relationship- oriented database management systems as well as other NoSQL competitors. The basic storage unit, a *document*, is a “data structure defined as a collection of named fields”. The document utilizes JSON, thereby allowing each document to have its own individual schema. [259]

CBS combines the in-memory capabilities of Membase with CouchDB’s inherent data store reliability and data persistency. Membase functions in RAM only, providing the highest-possible speed capabilities to end users. However, Membase’s in-ram existence limits the amount of data it can use. More importantly, it provides no mechanism for data recovery if the server crashes. Combining Membase with CouchDB provides a persistent data source, mitigating the disadvantages of either product. In addition, CouchDB + membase allows the data size “to grow beyond the size of RAM”. [104]

CBS is written in Erlang/OTP, but generally shortened to just Erlang. In actuality, it is written in “Erlang using components of OTP alongside some C/C++” [180], It runs on an Erlang virtual machine known as BEAM. [690]

Out-of-the-box benefits of Erlang/OTP include dynamic type setting, pattern matching and, most importantly, actor-model concurrency. As a result, Erlang code virtually eliminates the possibility of inadvertent deadlock scenarios. In addition, Erlang/OTP processes are lightweight, spawning new processes does not consume many resources and message passing between processes is fast since they run in the same memory space. Finally, OTP’s process supervision tree makes Erlang/OTP extremely fault-tolerant. Error handling is indistinguishable from a process startup, easing testing

and bug detection. [535]

CouchDB's design adds another layer of reliability to CBS. CouchDB operates in *append-only* mode, so it adds user changes to the tail of database. This setup resists data corruption while taking a snapshot, even if the server continues to run during the procedure. [342]

Finally, CB uses the Apache 2.0 License, one of several open-source license alternatives. [557]

39.12.15 IBM Cloudant

Cloudant is based on both Apache-backed CouchDB project and the open source BigCouch project. IBM Cloudant is an open source non-relational, distributed database service as service (DBaaS) that provides integrated data management, search and analytics engine designed for web applications. Cloudant's distributed service is used the same way as standalone CouchDB, with the added advantage of data being redundantly distributed over multiple machines [697].

39.12.16 Pivotal Gemfire (6)

A real-time, consistent access to data-intensive applications is provided by a open source, data management platform named Pivotal Gemfire. "GemFire pools memory, CPU, network resources, and optionally local disk across multiple processes to manage application objects and behavior". The main features of Gemfire are high scalability, continuous availability, shared nothing disk persistence, heterogeneous data sharing and parallelized application behavior on data stores to name a few. In Gemfire, clients can subscribe to receive notifications to execute their task based on a specific change in data. This is achieved through the continuous querying feature which enables event-driven architecture. The shared nothing architecture of Gemfire suggests that each node is self-sufficient and independent, which means that if the disk or caches in one node fail the remaining nodes remain untouched. Additionally, the support for multi-site configurations enable the user to scale horizontally between different distributed systems spread over a wide geographical network.

39.12.17 HBase

Apache Hbase is a distributed column-oriented database which is built on top of HDFS (Hadoop Distributed File System). According to [61], It is a open source, versioned, distributed, non-relational database modelled after Google's Bigtable. Similar to Bigtable providing harnessing distributed file storage system offered by Google file system, Apache Hbase provides similar capabilities on top of Hadoop and HDFS. Moreover, Hbase supports random, real-time CRUD (Create/Read/Update/Delete) operations.

Hbase is a type of NoSQL database and is classified as a key value store. In HBase, value is identified with a key where both of them are stored as byte arrays. Values are stored in the order of keys. HBase is a database system where the tables have no schema. Some of the companies that use HBase as their core program are Facebook, Twitter, Adobe, Netflix etc.

39.12.18 Google Bigtable

Google Bigtable is a NoSQL database service, built upon several Google technologies, including Google File System, Chubby Lock Service, and SSTable [236]. Designed for Big Data, Bigtable provides high performance and low latency and scales to hundreds of petabytes [236]. Bigtable powers many core Google products, such as Search, Analytics, Maps, Earth, Gmail, and YouTube.

Bigtable also drives Google Cloud Datastore and influenced Spanner, a distributed NewSQL database also developed by Google [677] [684]. Since May 6, 2015, Bigtable has been available to the public as Cloud Bigtable [684].

39.12.19 LevelDB

LevelDB is a light-weight, single-purpose library for persistence with bindings to many platforms. [370] It is a simple open source on-disk key/value data store built by Google, inspired by BigTable and is used in Google Chrome and many other products. It supports arbitrary byte arrays as both keys and values, singular get, put and delete operations, batched put and delete, bi-directional iterators and simple compression using the very fast Snappy algorithm. It is hosted on GitHub under the New BSD License and has been ported to a variety of Unix-based systems, Mac OS X, Windows, and Android. It is not an SQL database and does not support SQL queries. Also, it has no support for indexes. Applications use LevelDB as a library, as it does not provide a server or command-line interface.

39.12.20 Megastore and Spanner

Spanner [137] is Google's distributed database which is used for managing all google services like play, gmail, photos, picasa, app engine etc Spanner is distributed database which spans across multiple clusters, datacenters and geo locations. Spanner is structured in such a way so as to provide non blocking reads, lock free transactions and atomic schema modification. This is unlike other noSql databases which follow the CAP theory i.e. you can choose any two of the three: Consistency, Availability and Partition-tolerance. However, spanner gives an edge by satisfying all three of these. It gives you atomicity and consistency along with availability, partition tolerance and synchronized replication. Megastore bridges the gaps found in google's bigtable. As google realized that it is difficult to use bigtable where the application requires constantly changing schema. Megastore offers a solution in terms of semi-relational data model. Megastore [103] also provides a transactional database which can scale unlike relational data stores and synchronous replication. Replication in megastore is supported using Paxos. Megastore also provides versioning. However, megastore has a poor write performance and lack of a SQL like query language. Spanners basically adds what was missing in Bigtable and megastore. As a global distributed database spanner provides replication and globally consistent reads and writes. Spanner deployment is called universe which is a collections of zones. These zones are managed by singleton universe master and placement driver. Replication in spanner is supported by Paxos state machine. Spanner was put into evaluation in early 2011 as F1 backend(F1 is Google's advertisement system) which was replacement to mysql. Overall spanner fulfills the needs of relational database along with scaling of noSQL database. All these features make google run all their apps seamlessly on spanner infrastructure.

39.12.21 Accumulo

Apache Accumulo, a highly scalable structured store based on Google's BigTable, is a sorted, distributed key/value store that provides robust, scalable data storage and retrieval. Accumulo is written in Java and operates over the Hadoop Distributed File System (HDFS), which is part of the popular Apache Hadoop project. Accumulo supports efficient storage and retrieval of structured data, including queries for ranges, and provides support for using Accumulo tables as input and output for MapReduce jobs. Accumulo features automatic load-balancing and partitioning, data compression and fine-grained security labels. Much of the work Accumulo does involves maintaining certain properties of the data, such as organization, availability, and integrity, across

many commodity-class machines [37].

39.12.22 Cassandra

Apache Cassandra [41] is an open-source distributed database management for handling large volume of data across commodity servers. It works on asynchronous masterless replication technique leading to low latency and high availability. It is a hybrid between a key-value and column oriented database. A table in cassandra can be viewed as a multi dimensional map indexed by a key. It has its own “Cassandra Query language (CQL)” query language for data extraction and mining. One of the demerits of such structure is it does not support joins or subqueries. It is a java based system which can be administered by any JMX compliant tools.

39.12.23 RYA

Rya is a “scalable system for storing and retrieving RDF data in a cluster of nodes.” [492] RDF stands for Resource Description Framework. [492] RDF is a model that facilitates the exchange of data on a network. [262] RDF utilizes a form commonly referred to as a triple, an object that consists of a subject, predicate, and object. [492] These triples are used to describe resources on the Internet. [492] Through new storage and querying techniques, Rya aims to make accessing RDF data fast and easy. [522]

39.12.24 Sqrrl

39.12.25 Neo4J

Neo4J [669] is a popular ACID compliant graph database management system developed by Neo technology. In this database everything is stored as nodes or edges, both of which can be labeled. Labels help in narrowing and simplifying the search process through the database. [506] It is a highly scalable software and can be distributed across multiple machines. The graph query language that accompanies the software has traversal framework which makes it fast and powerful. [420] The Neo4J is often used for clustering. It offers two feature clustering solutions: Causal Clustering and Highly available clustering. [419] Casual clustering focuses on safety, scalability and causal consistency in the graph. [421] The highly available cluster places importance to fault tolerance as each instance in the cluster has full copies of data in their local database.

39.12.26 graphdb

A Graph Database is a database that uses graph structures for semantic queries with nodes, edges and properties to represent and store data. [692] The Graph is a concept which directly relates the data items in the store. The data which is present in the store is linked together directly with the help of relationships. It can be retrieved with a single operation. Graph database allow simple and rapid retrieval of complex hierarchical structures that are difficult to model in relational systems.

There are different underlying storage mechanisms used by graph databases. Some graphdb depend on a relational engine and store the graph data in a table, while others use a key-value store or document-oriented database for storage. Thus, they are inherently caled as NoSQL structures. Data retrieval in a graph database requires a different query language other than SQL. Some of the query languages used to retrieve data from a graph database are Gremlin, SPARQL, and Cypher. Graph databases are based on graph theory. They employ the concepts of nodes, edges and properties.

39.12.27 Yarcdata

Yarcdata is Cray subsidiary providing Analytics products, namely the Urika Agile Analytics Platform and Graph Engine. Cray's Urika (Universal RDF Integration Knowledge Appliance) system [143] is a hardware platform designed specifically to provide high-speed graph-retrieval for relationship analytics. Urika is a massively parallel, multi-threaded, shared-memory computing device designed to store and retrieve massive graph datasets. The system can import and host massive heterogeneous graphs represented in the resource description framework (RDF) format and can retrieve descriptive graph patterns specified in a SPARQL query.

Urika-GD [144] is a big data appliance for graph analytics helps enterprises gain key insights by discovering relationships in big data. Its highly scalable, real-time graph analytics warehouse supports ad hoc queries, pattern-based searches, inferencing and deduction. The Urika-GD appliance complements an existing data warehouse or Hadoop® cluster by offloading graph workloads and interoperating within the existing analytics workflow

Cray Graph Engine [519] is a semantic database using Resource Description Framework (RDF) triples to represent the data, SPARQL as the query language and extensions to support mathematical algorithms.

The paper “Graph mining meets the semantic web” [528] outlines the implementation of graph mining algorithms using SPARQL.

39.12.28 AllegroGraph

“AllegroGraph is a database technology that enables businesses to extract sophisticated decision insights and predictive analytics from their highly complex, distributed data that can't be answered with conventional databases, i.e., it turns complex data into actionable business insights.” [19] It can be viewed as a closed source database that is used for storage and retrieval of data in the form of triples (triple is a data entity composed of subject-predicate-object like “Professor teaches students”). Information in a triple store is retrieved using a query language. Query languages can be classified into database query languages or information retrieval query languages. The difference is that a database query language gives exact answers to exact questions, while an information retrieval query language finds documents containing requested information. Triple format represents information in a machine-readable format. Every part of the triple is individually addressable via unique URLs — for example, the statement “Professor teaches students” might be represented in RDF(Resource Description Framework). Using this representation, semantic data can be queried. [20]

39.12.29 Blazegraph

Blazegraph is a graph database also supporting property graph, capable of clustered deployment. A graph database is a NoSQL database. It is based on a graph theory of nodes and edges where each node represents an element such as user or business and each edge represents relationship between two nodes. It is mainly used for storing and analyzing data where maintaining interconnections is essential. Data pertaining to social media is best example where graph database can be used.

Blazegraph's main focus is large scale complex graph analytics and query. The Blazegraph database runs on graphics processing units (GPU) to speed graph traversals. :cite ‘paper-blzgraph’

Lets now see how Blazegraph handles data. :cite ‘www-blzgraph’ **Blazegraph data can be accessed** using REST APIs.

Blazegraph supports Apache TinkerPop, which is a graph computing framework.

For graph data mining, Blazegraph implements GAS (Gather, Apply, Scatter) model as a service.

39.12.30 Facebook Tao

In the paper published in USENIX annual technical conference, Facebook Inc describes TAO (The Association and Objects) as :cite ‘book-tao’ a geographically distributed data store that provides timely access to the social graph for Facebook’s demanding workload using a fixed set of queries. It is deployed at Facebook for many data types that fit its model. The system runs on thousands of machines, is widely distributed, and provides access to many petabytes of data. TAO represents social data items as Objects (user) and relationship between them as Associations (liked by, friend of). TAO cleanly separates the caching tiers from the persistent data store allowing each of them to be scaled independently. To any user of the system it presents a single unified API that makes the entire system appear like 1 giant graph database. [638].

39.12.31 Titan:db

Titan:db [615] is a distributed graph database that can support of thousands of concurrent users interacting with a single massive graph database that is distributed over the clusters. It is open source with liberal Apache 2 license. Its main components are storage backend, search backend, and TinkerPop graph stack. Titan provides support for various storage backends and also linear scalability for a growing data and user base. It inherits features such as ‘Gremlin’ query language and ‘Rexter’ graph server from TinkerPop [614]. For huge graphs, Titan uses a component called Titan-hadoop which compiles Gremlin queries to Hadoop MapReduce jobs and runs them on the clusters. Titan is basically optimal for smaller graphs.

39.12.32 Jena

Jena is an open source Java Framework provided by Apache for semantic web applications. ([645]) It provides a programmatic environment for RDF, RDFS and OWL, SPARQL, GRDDL, and includes a rule-based inference engine. Semantic web data differs from conventional web applications in that it supports a web of data instead of the classic web of documents format. The presence of a rule based inference engine enable Jena to perform a reasoning based on OWL and RDFS ontologies. [620] ‘The architecture of Jena contains three layers : Graph layer, model layer and Ontology layer. The graph layer forms the base for the architecture. It does not have an extensive RDF implementation and serves more as a Service provider Interface. According to [620] It provides classes/methods that could be further extended. The model layer extends the graph layer and provides objects of type ‘resource’ instead of ‘node’ to work with. The ontology layer enables one to work with triples.

39.12.33 Sesame

Sesame is framework which can be used for the analysis of RDF (Resource Description Framework) data. Resource Description Framework (RDF) [12] is a model that facilitates the interchange of data on the Web. Using RFD enables us to merge data even if the underlying schemas differ. Sesame has now officially been integrated into RDF4J Eclipse project [13]. Sesame takes in the natively written code as the input and then performs a series of transformations, generating kernels for various platforms. In order to achieve this, it makes use of the feature identifier, impact predictor, source-to-source translator and the auto-tuner [328]. The feature identifier is concerned with the

extraction and detection of the architectural features that are important for application performance. The impact predictor determines the performance impact of the core features extracted above. A source-to-source translator transforms the input code into a parametrized one; while the auto-tuner helps find the optimal solution for the processor.

39.12.34 Public Cloud: Azure Table

Microsoft offers its NoSQL Azure Table product to the market as a low-cost, fast and scalable data storage option. [556] Table stores data as collections of key-value combinations, which it terms *properties*. Table refers to a collection of properties as an *entity*. Each entity can contain a mix of properties. The mix of properties can vary between each entity, although each entity may consist of no more than 255 properties. [607]

Although data in Azure Table will be structured via key-value pairs, Table provides just one mechanism for the user to define relationships between entities: the entity's *primary key*. The primary key, which Microsoft sometimes calls a *clustered index*, consists of a PartitionKey and a RowKey. The PartitionKey indicates the group, a.k.a partition, to which the user assigned the entity. The RowKey indicates the entity's relative position in the group. Table sorts in ascending order by the PartitionKey first, then by the RowKey using lexical comparisons. As a result, numeric sorting requires fixed-length, zero-padded strings. For instance, Table sorts *111* before *2*, but will sort *111* after *002*. [399]

Azure Table is considered best-suited for infrequently accessed data storage.

39.12.35 Amazon Dynamo

Amazon explains DynamoDB as [www.dyndb] a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. It is a fully managed cloud database and supports both document and key-value store models. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad tech, IoT, and many other applications. DynamoDB can be easily integrated with big-data processing tools like Hadoop. It can also be integrated with AWS Lambda, an event driven platform, which enables creating applications that can automatically react to data changes. At present there are certain limits to DynamoDB. Amazon has listed all the limits in a web page titled *Limits in DynamoDB* <http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Limits.html>

39.12.36 Google DataStore

Google Cloud Datastore is a NoSQL document database built for automatic scaling, high performance, and ease of application development [232]. Though Cloud Datastore interface has many of the features similar to traditional databases, but as a NoSQL database, it differs from the SQL in the way as it describes relationships between various data objects. It also provides a number of features that relational databases are not optimally suited to provide, including high-performance at a very large scale and high-reliability. The Google Cloud DataStore can have different kinds of properties for the same kind of entities, unlike the Relational Database where they are represented in rows. For example, the difference between entities can have the properties with the same name but having different values. The flexible schema maps naturally to object-oriented and scripting languages.

Non-relational databases have become popular recently, especially for web applications that require high-scalability and performance with high-availability. Non-relational databases such as Cloud DataStore let developers to choose an optimal balance between strong consistency and eventual

consistency for each application. This allows developers to combine the benefits of both the database structures [251]. Datastore is designed to automatically scale to very large data sets, allowing applications to maintain high performance as they receive more traffic. Datastore also provides a number of features that relational databases are not optimally suited to provide, including high-performance at a very large scale and high-reliability [232].

39.13 File management

39.13.1 iRODS

The Integrated Rule-Oriented Data System (iRODS) is open source data management software. iRODS is released as a production-level distribution aimed at deployment in mission critical environments. It virtualizes data storage resources, so users can take control of their data, regardless of where and on what device the data is stored. The development infrastructure supports exhaustive testing on supported platforms. The plugin architecture supports microservices, storage systems, authentication, networking, databases, rule engines, and an extensible API [315]. iRODS implements data virtualization, allowing access to distributed storage assets under a unified namespace, and freeing organizations from getting locked in to single-vendor storage solutions. iRODS enables data discovery using a metadata catalog that describes every file, every directory, and every storage resource in the iRODS Zone. iRODS automates data workflows, with a rule engine that permits any action to be initiated by any trigger on any server or client in the Zone. iRODS enables secure collaboration, so users only need to log in to their home Zone to access data hosted on a remote Zone. [316]

39.13.2 NetCDF

NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array oriented scientific data. NetCDF was developed and is maintained at Unidata , part of the University Corporation for Atmospheric Research (UCAR) Community Programs (UCP). Unidata is funded primarily by the National Science Foundation [170] [422] . The purpose of the Network Common Data Form(netCDF) interface is to support the creation, efficient access, and sharing of data in a form that is self-describing, portable, compact, extendible, and archivable Version 3 of netCDF is widely used in atmospheric and ocean sciences due to its simplicity. NetCDF version 4 has been designed to address limitations of netCDF version 3 while preserving useful forms of compatibility with existing application software and data archives [170]. NetCDF consists of: a) A conceptual data model b) A set of binary data formats c) A set of APIs for C/Fortran/Java

39.13.3 CDF

Common Data Format [416] is a conceptual data abstraction for storing, manipulating, and accessing multidimensional data sets. CDF differs from traditional physical file formats by defining form and function as opposed to a specification of the bits and bytes in an actual physical format.

CDF's integrated dataset is composed by following two categories :(a)Data Objects - scalars, vectors, and n-dimensional arrays.(b)Metadata - set of attributes describing the CDF in global terms or specifically for a single variable [415].

The self-describing property (metadata) allows CDF to be a generic, data-independent format that can store data from a wide variety of disciplines. Hence, the application developer remains insulated

from the actual physical file format for reasons of conceptual simplicity, device independence, and future expandability. CDF data sets are portable on any of the CDF-supported platforms and accessible with CDF applications or layered tools. To ensure the data integrity in a CDF file, checksum method using MD5 algorithm is employed [158].

Compared to HDF format [694], CDF permitted cross-linking data from different instruments and spacecraft in ISTP with one development effort. CDF is widely supported by commercial and open source data analysis/visualization software such as IDL, MATLAB, and IBM's Data Explorer (XP).

39.13.4 HDF

39.13.5 OPeNDAP

39.13.6 FITS

FITS stand for 'Flexible Image Transport System'. It is a standard data format used in astronomy. FITS data format is endorsed by NASA and International Astronomical Union. According to [191], FITS can be used for transport, analysis and archival storage of scientific datasets and support multi-dimensional arrays, tables and headers sections. FITS is actively used and developed - according to [192] newer version of FITS standard document was released in July 2016. FITS can be used for digitization of contents like books and magazines. Vatican Library [189] used FITS for long term preservation of their book, manuscripts and other collection. Matlab, a language used for technical computing supports fits [190]. The 2011 paper [341] explains how to perform processing of astronomical images on Hadoop using FITS.

39.13.7 RCFile

RCFile (Record Columnar File) [508] is a big data placement data structure that supports fast data loading and query processing coupled with efficient storage space utilization and adaptive to dynamic workload environments. It is designed for data warehousing systems that uses map-reduce. The data is stored as a flat file comprising of binary key/value pairs. The rows are partitioned first and then the columns are partitioned in each row and the respective meta-data for each row is stored in the key part for that row and the values comprises of the data part of the row. Storing the data in this format enables RCFile to accomplish fast loading and query processing. A shell utility is available for reading RCFile data and metadata [507]. According to [275], RCFile has been chosen in Facebook data warehouse system as the default option. It has also been adopted by Hive and Pig, the two most widely used data analysis systems developed in Facebook and Yahoo!

39.13.8 ORC

ORC files were created as part of the initiative to massively speed up Apache Hive and improve the storage efficiency of data stored in Apache Hadoop. ORC is a self-describing type-aware columnar file format designed for Hadoop workloads. It is optimized for large streaming reads, but with integrated support for finding required rows quickly. Storing data in a columnar format lets the reader read, decompress, and process only the values that are required for the current query. Because ORC files are type-aware, the writer chooses the most appropriate encoding for the type and builds an internal index as the file is written. ORC files are divided in to stripes that are roughly 64MB by default. The stripes in a file are independent of each other and form the natural unit of distributed work. Within each stripe, the columns are separated from each other so the reader can read just the columns that are required [84].

39.13.9 Parquet

Apache parquet is the column Oriented data store for Apache Hadoop ecosystem and available in any data processing framework, data model or programming language [200]. It stores data such that the values in each column are physically stored in contiguous memory locations. As it has the columnar storage, it provides efficient data compression and encoding schemes which saves storage space as the queries that fetch specific column values need not read the entire row data and thus improving performance. It can be implemented using the Apache Thrift framework which increases its flexibility to work with a number of programming languages like C++, Java, Python, PHP, etc.

39.14 Data Transport

39.14.1 BitTorrent

Bittorrent is P2P communication protocol commonly used for sending and receiving the large digital files like movies and audioclips. In order to upload and download file, user have to download bittorrent client which implement the bittorrent protocol. Bittorrent uses the principle of swarming and tracking. [99] It divides the files in large number of chunk and as soon as file is received it can be served to the other users for downloading. So rather than downloading one entire large file from one source, user can download small chunk from the different sources of linked users in swarm. Bittorrent trackers keeps list of files available for transfer and helps the swarm user find each other.

Using the protocol, machine with less configuration can serve as server for distributing the files. It result in increase in the downloading speed and reduction in origin server configuration.

Few popular bittorrent client in μ Torrent, qBittorrent.

39.14.2 HTTP

39.14.3 FTP

According to [209] FTP is an acronym for File Transfer Protocol. It is network protocol standard used for transferring files between two computer systems or between a client and a server. It is part of the Application layer of the Internet Protocol Suite and works along with HTTP/SSH. It follows a client-server model architecture. Secure systems asks the client to authenticate themselves using a Username and Password registered with the server to access the files via FTP. The specification for FTP was first written by Abhay Bhushan [[www-rfc114](#)] in 1971 and is termed as RFC114. The current specification, RFC959 in use was written in 1985. Several other versions of the specification are available which provides firewall friendly FTP access, additional security extensions, support for IPV6 and passive mode file access respectively. FTP can be used in command line in most of the operating systems to transfer files. There are FTP clients such as WinSCP, FileZilla etc. which provides a graphical user interface to the clients to authenticate themselves (sign on) and access the files from the server.

39.14.4 SSH

SSH is a cryptographic network protocol [572] to provide a secure channel between two clients over an unsecured network. It uses public-key cryptography for authenticating the remote machine and the user. The public-private key pairs could be generated automatically to encrypt the network connection. ssh-keygen utility could be used to generate the keys manually. The public key then could be placed on the all the computers to which the access is required by the owner of the private

key. SSH runs on the client-server model where a server listens for incoming ssh connection requests. It's generally used for remote login and command execution. Its other important uses include tunneling(required in cloud computing) and file transfer(SFTP). OpenSSH is an open source implementation of network utilities based on SSH [573].

39.14.5 Globus Online (GridFTP)

GridFTP is an enhancement on the File Transfer Protocol (FTP) which provides high-performance, secure and reliable data transfer for high-bandwidth wide-area networks. As noted in [260] the most widely used implementation of GridFTP is Globus Online. GridFTP achieves efficient use of bandwidth by using multiple simultaneous TCP streams. Files can be downloaded in pieces simultaneously from multiple sources; or even in separate parallel streams from the same source. GridFTP allows transfers to be restarted automatically and handles network unavailability with a fault tolerant implementation of FTP. The underlying TCP connection in FTP has numerous settings such as window size and buffer size. GridFTP allows automatic (or manual) negotiation of these settings to provide optimal transfer speeds and reliability.

39.14.6 Flume

Flume is distributed, reliable and available service for efficiently collecting, aggregating and moving large amounts of log data [593]. Flume was created to allow you to flow data from a source into your Hadoop® environment. In Flume, the entities you work with are called sources, decorators, and sinks. A source can be any data source, and Flume has many predefined source adapters. A sink is the target of a specific operation. A decorator is an operation on the stream that can transform the stream in some manner, which could be to compress or uncompress data, modify data by adding or removing pieces of information, and more [293].

39.14.7 Sqoop

Apache Sqoop is a tool to transfer large amounts of data between Apache Hadoop and sql databases [591]. The name is a Portmanteau of SQL + Hadoop. It is a command line interface application which supports incremental loads of complete tables, free form (custom) SQL Queries and allows the use of saved and scheduled jobs to import latest updates made since the last import. The imports can also be used to populate tables in Hive or Hbase. Sqoop has the option of export, which allows data to be transferred from Hadoop into a relational database. Sqoop is supported in many different business integration suits like Informatica Big Data Management, Pentaho Data Integration, Microsoft BI Suite and Couchbase [662].

39.14.8 Pivotal GPLOAD/GPFDIST

Greenplum Database [233] is a shared nothing, massively parallel processing solution built to support next generation data warehousing and Big Data analytics processing. In its new distribution under Pivotal, Greenplum Database is called Pivotal(Greenplum) Database.

gpfdist [256] is Greenplum's parallel file distribution program. It is used by readable external tables and gpload to serve external table files to all Greenplum Database segments in parallel. It is used by writable external tables to accept output streams from Greenplum Database segments in parallel and write them out to a file.

gpload [233] is data loading utility is used to load data into Greenplum's external table in parallel.

Google has an invention [131] relating to integrating map-reduce processing techniques into a distributed relational database. An embodiment of the invention is implemented by Greenplum as gpfdist.

39.15 Cluster Resource Management

39.15.1 Mesos

Apache Mesos [395] abstracts CPU, memory, storage, and other compute resources away from machines (physical or virtual), enabling fault-tolerant and elastic distributed systems to easily be built and run effectively. The Mesos kernel runs on every machine and provides applications (e.g., Hadoop, Spark, Kafka, Elasticsearch) with API's for resource management and scheduling across entire datacenter and cloud environments.

The resource scheduler of Mesos supports a generalization of max-min fairness [7], termed Dominant Resource Fairness (DRF) [225] scheduling discipline, which allows to harmonize execution of heterogeneous workloads (in terms of resource demand) by maximizing the share of any resource allocated to a specific framework.

Mesos uses containers for resource isolation between processes. In the context of Mesos, the two most important resource-isolation methods to know about are the control groups (cgroups) built into the Linux kernel, and Docker. The difference between using hyper-V, Docker containers, cgroup is described in detail in the book “Mesos in action” [302]

39.15.2 Yarn

Yarn (Yet Another Resource Negotiator) is Apache Hadoop’s cluster management project [128]. It’s a resource management technology which make a pace between, the way applications use Hadoop system resources & node manager agents. Yarn, “split up the functionalities of resource management and job scheduling/monitoring”. The NodeManager watch the resource (cpu, memory, disk, network) usage the container and report the same to ResourceManager. ResourceManager will take a decision on allocation of resources to the applications. ApplicationMaster is a library specific to application, which requests/negotiate resources from ResourceManager and launch and monitoring the task with NodeManager(s) [574]. ResourceManager have two majors: Scheduler and ApplicationManager. Scheduler have a task to schedule the resources required by the application. ApplicationManger holds the record of application who require resource. It validates (whether to allocate the resource or not) the application’s resource requirement and ensure that no other application already have register for the same resource requirement. Also it keeps the track of release of resource. [266]

39.15.3 Helix

Helix is a data management system getting developed by IBM which helps the users to do explicatory analysis of the data received from various sources following different formats. This system would help organize the data by providing links between data collected across various sources despite of the knowledge of the data sources schemas. It also aims at providing the data really required for the user by extracting the important information from the data. This would plan to target the issue by maintaining the “knowledge base of schemas” and “context-dependent dynamic linkage”, The system can get the schema details either from the knowledge base being maintained or can even get the schema from the data being received. As the number of users for helix increases

the linkages gets stronger and would provide better data quality. [183]

39.15.4 Llama

Llama stands for leveraging learning to automatically manage algorithms. There has been a phenomenal improvement in algorithm portfolio and selection approaches. The main drawback of them is that their implementation is specific to a problem domain and customized which leads to the difficulty of exploring new techniques for certain problem domains. Llama has been developed to provide an extensible toolkit which can initiate exploration of a variety of portfolio techniques over a wide range of problem domains. It is modular and implemented as an R package. It leverages the extensive library of machine learning algorithms and techniques in R [357]. Llama can be regarded as a framework which provides the prerequisites for initiating automatic portfolio selectors. It provides a set of methods for combining several trivial approaches of portfolio selection into sophisticated techniques. The primary reason behind the introduction of Llama was to help the researchers working in algorithm selection, algorithm portfolios, etc. and can be just used as a tool for designing the systems [357].

39.15.5 Google Omega

39.15.6 Facebook Corona

Corona is a new scheduling framework developed by facebook which separates the cluster resource management from job coordination. Facebook, employed the MapReduce implementation from Apache Hadoop since 2011 for job scheduling. The scheduling MapReduce framework has its limitations with the scalability as when the number of jobs at facebook grew in the next few years. Another limitation of Hadoop was it was a pull-based scheduling model as the task tracker have to provide a heartbeat to the job tracker to indicate that it is running which associated with a pre-defined delay, that was problematic for small jobs [184]. Hadoop MapReduce is also constrained by its static slot-based resource management model where a MapReduce cluster is divided into a fixed number of map and reduce slots based on a static configurations so the slots are not utilized completely anytime the cluster workload does not fit the static configuration.

Corona improves over the Hadoop MapReduce by introducing a cluster manager whose only purpose is to track the nodes in the cluster and the amount free resources [184]. A dedicated job tracker is created for each job and can run either in the same process as the client (for small jobs) or as a separate process in the cluster (for large jobs). The other difference is that it uses a push-based scheduling whose implementation does not involve a periodic heartbeat and thus scheduling latency is minimized. The cluster manager also implements a fair-share scheduling as it has access to the full snapshot of the cluster for making the scheduling decisions. Corona is used as an integral part of the Facebook's data infrastructure and is helping power big data analytics for teams across the company.

39.15.7 Celery

“Celery is an asynchronous task queue/job queue based on distributed message passing. The focus of celery is mostly on real-time operation, but it equally scheduling. In celery there are execution units, called tasks, are executed concurrently on a single or more worker servers using multiprocessing, Eventlet,or gevent. Tasks can execute asynchronously (in the background) or synchronously (wait until ready). Celery is easy to integrate with web framework. Celery is written in python whereas the protocol can be implemented in any language” [111]. “Celery is a simple, flexible, and reliable

distributed system to process vast amounts of messages, while providing operations with the tools required to maintain such a system”[\[112\]](#)

39.15.8 HTCondor

HTCondor is a specialized workload management system for compute-intensive jobs. HTCondor provides various features like a) job queuing mechanism, b)scheduling policy, c)resource monitoring, d)priority scheme and e)resource management just as other full-featured batch systems. “Users submit their serial or parallel jobs to HTCondor, HTCondor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion”. HTCondor can be used to manage a cluster of dedicated compute nodes. HTCondor uses unique mechanisms to harness wasted CPU power from idle desktop workstations. “The ClassAd mechanism in HTCondor provides an extremely flexible and expressive framework for matching resource requests (jobs) with resource offers (machines). Jobs can easily state both job requirements and job preferences”. “HTCondor incorporates many of the emerging Grid and Cloud-based computing methodologies and protocols” [\[654\]](#)

39.15.9 SGE

According to [\[453\]](#), Sun Grid Engine (SGE) renamed to Oracle Grid Engine (OGE) is a grid computing cluster software system. Grid Engine is a high performance computing cluster used for managing job queueing in distributed and parallel environment. It can accept, schedule, dispatch and manage the execution of single, parallel user jobs in a remote or distributed manner. It also manages the resource allocation to those jobs. The resources can be anything like processors, storage, RAM and licenses for softwares. The latest stable release of OGE is termed as 6.2u8 which came out in October 1,2012.

OGE supports a vast array of features like: Topology-aware scheduling and thread binding, advanced fault tolerance mechanisms for job scheduling, web interface based status reporting and ability to use different scheduling algorithms,etc. OGE runs on several platforms including AIX, BSD, Linux, Solaris, OS X, Tru64, Windows, etc. It is under deployment phase for IBM’s 64-bit operating system z/OS. Standard Grid cluster comprises of one master host and many execution hosts. There is a option of creating shadow master hosts which would take the master’s place incase of a system crash. Notable deployments of OGE include: TSUBAME supercomputer at the Tokyo Institute of Technology, Ranger at the Texas Advanced Computing Center (TACC) and San Diego Supercomputer Center (SDSC).

39.15.10 OpenPBS

Portable Batch System (or simply PBS) is the name of computer software that performs job scheduling. Its primary task is to allocate computational tasks, i.e., batch jobs, among the available computing resources. It is often used in conjunction with UNIX cluster environments [\[663\]](#). OpenPBS is the original open source version of PBS. There are more commercialized versions of the same software. One of the key feature of OpenPBS is that it supports millions of cores with fast job dispatch and minimal latency. It meets unique site goals and SLAs by balancing job turnaround time and utilization with optimal job placement. OpenPBS also includes automatic fail-over architecture with no single point of failure – jobs are never lost, and jobs continue to run despite failures. It is built upon a Flexible Plugin Framework which simplifies administration with enhanced visibility and extensibility [\[465\]](#).

39.15.11 Moab

Moab HPC Suite is a workload management and resource orchestration platform that automates the scheduling, managing, monitoring, and reporting of HPC workloads on massive scale. It uses multi-dimensional policies and advanced future modeling to optimize workload start and run times on diverse resources. It integrates and accelerates the workloads management across independent clusters by adding grid-optimized job submission. Moab's unique intelligent and predictive capabilities evaluate the impact of future orchestration decisions across diverse workload domains (HPC, HTC, Big Data, and Cloud VMs)[404].

39.15.12 Slurm (553)

Simple Linux Utility for Resource Management (SLURM) workload manager is an open source, scalable cluster resource management tool used for job scheduling in small to large Linux cluster using multi-core architecture. As per, [532] SLURM has three key functions. First, it allocates resources to users for some duration with exclusive and/or non-exclusive access. Second, it enables users to start, execute and monitor jobs on the resources allocated to them. Finally, it intermediates to resolve conflicts on resources for pending work by maintaining them in a queue. The slurm architecture has following components: a centralized manager to monitor resources and work, may have a backup manager, daemon on each server to provide fault-tolerant communications, an optional daemon for clusters with multiple mangers and tools to initiate, terminate and report about jobs in a graphical view with network topology. It also provides around twenty additional plugins that could be used for functionalities like accounting, advanced reservation, gang scheduling, back fill scheduling and multifactor job prioritization. Though originally developed for Linux, SLURM also provides full support on platforms like AIX, FreeBSD, NetBSD and Solaris [533].

39.15.13 Torque

39.15.14 Globus Tools

[562] The Globus Toolkit is an open source toolkit organized as a collection of loosely coupled components. These components consist of services, programming libraries and development tools designed for building Grid-based applications. GT components fall into five broad domain areas: Security, Data Management, Execution Management, Information Services, and Common Runtime. [198] These components enable a broader "Globus ecosystem" of tools and components that build on or interoperate with GT functionality to provide a wide range of useful application-level functions. www-about-globus [231] Since 2000, companies like Fujitsu, IBM, NEC and Oracle have pursued Grid strategies based on the Globus Toolkit.

39.15.15 Pilot Jobs

In pilot job, an application acquires a resource so that it can be delegated some work directly by the application; instead of requiring some job scheduler. The issue of using a job scheduler is that a waiting queue is required. Few examples of Pilot Jobs are the [503] Falkon lightweight framework and [353] HTCaaS. Pilot jobs are typically associated with both Parallel computing as well as Distributed computing. Their main aim is to reduce the dependency on queues and the associated multiple wait times.

Using pilot jobs enables us to have a multilevel technique for the execution of various workloads. This is so because the jobs are typically acquired by a placeholder job and they relayed to the workloads [622].

39.16 File systems

39.16.1 HDFS

Hadoop provides distributed file system framework that uses Map reduce (Distributed computation framework) for transformation and analyses of large dataset. Its main work is to partition the data and other computational tasks to be performed on that data across several clusters. HDFS is the component for distributed file system in Hadoop. An HDFS cluster primarily consists of a Name Node and Data Nodes. Name Node manages the file system metadata such as access permission, modification time, location of data and Data Nodes store the actual data. When user applications or Hadoop frameworks request access to a file in HDFS, Name Node service responds with the Data Node locations for the respective individual data blocks that constitute the whole of the requested file[[274](#)].

39.16.2 Swift

39.16.3 Haystack

Haystack is an open source project working with data from internet of Things, aim to standardise the semantic data model generated from smart devices, homes, factories etc. It include automation, control, energy, HVAC, lighting and other environmental systems. [[272](#)]

Building block of Project haystack is on TagModel tagging of metadata stored in key/value pair applied to entity such id, dis, sites, geoAddr, tz. Structure the primary structure of haystack is based on three entities, Site location of single unit, equip physical or logical piece of equipment within site, point sensor, actuator or setpoint value for equip, it also includes weather outside weather condition. TimeZone time series data is most important factor it is foundation for sensor and operational data. Captured data not always associated with measurable unit, however it provides facility to associate the data points. Commonly Supported units like Misc, Area, Currency, Energy, Power, Temperature, Temperature differential, Time, Volumetric Flow. The data often represented in 2D tabular form for tagged entities. It supports the query language for filtering over the data, data exposed through REST API in JSON format.

39.16.4 f4

As the amount of data Facebook stores continues to increase, the need for quick access and efficient storage of data continues to rise. Facebook stores a class of data in Binary Large OBjects (BLOBs), which can be created once, read many times, never modified, and sometimes deleted. Haystack, Facebook's traditional BLOB storage system is becoming increasingly inefficient. The storage efficiency is measured in the effective-replication-factor of BLOBs.

f4 BLOB storage system provides an effective-replication-factor lower than that of Haystack. f4 is simple, modular, scalable, and fault tolerant. f4 currently stores over 65PBs of logical BLOBs, with a reduced effective-replication-factor from 3.6 to either 2.8 or 2.1 [[410](#)].

39.16.5 Cinder

"Cinder is a block storage service for Openstack" [[119](#)]. Openstack Compute uses ephemeral disks meaning that they exist only for the life of the Openstack instance i.e. when the instance is terminated the disks disappear. Block storage system is a type of persistent storage that can be used to persist data beyond the life of the instance. Cinder provides users with access to persistent block-

level storage devices. It is designed such that users can create block storage devices on demand and attach them to any running instances of OpenStack Compute [502]. This is achieved through the use of either a reference implementation(LVM) or plugin drivers for other storage. Cinder virtualizes the management of block storage devices and provides end users with a self-service API to request and consume those resources without requiring any knowledge of where their storage is actually deployed or on what type of device [119].

39.16.6 Ceph

Ceph is open-source storage platform providing highly scalable object, block as well as file-based storage. Ceph is a unified, distributed storage system designed for excellent performance, reliability and scalability [516]. Ceph Storage clusters are designed to run using an algorithm called CRUSH (Controlled Replication Under Scalable Hashing) which replicates and re-balance data within the cluster dynamically to ensure even data distribution across cluster and quick data retrieval without any centralized bottlenecks.

Ceph's foundation is the Reliable Autonomic Distributed Object Store (RADOS) [518], which provides applications with object, block, and file system storage in a single unified storage cluster—making Ceph flexible, highly reliable and easy to manage. Ceph decouples data and metadata operations by eliminating file allocation tables and replacing them with generating functions which allows RADOS to leverage intelligent OSDs to manage data replication, failure detection and recovery, low-level disk allocation, scheduling, and data migration without encumbering any central server(s) [648].

The Ceph Filesystem [517] is a POSIX-compliant filesystem that uses a Ceph Storage Cluster to store its data. Ceph's dynamic subtree partitioning is a uniquely scalable approach, offering both efficiency and the ability to adapt to varying workloads. Ceph Object Storage supports two compatible interfaces: Amazon S3 and Openstack Swift.

39.16.7 FUSE

FUSE (Filesystem in Userspace) [210] "is an interface for userspace programs to export a filesystem to the Linux kernel". The FUSE project consists of two components: the fuse kernel module and the libfuse userspace library. libfuse provides the reference implementation for communicating with the FUSE kernel module. The code for FUSE itself is in the kernel, but the filesystem is in userspace. As per the 2006 paper [734] on HPTFS which has been built on top of FUSE. It mounts a tape as normal file system based data storage and provides file system interfaces directly to the application. Another implementation of FUSE FS is CloudBB [729]. Unlike conventional filesystems CloudBB creates an on-demand two-level hierarchical storage system and caches popular files to accelerate I/O performance. On evaluating performance of real data-intensive HPC applications in Amazon EC2/S3, results show CloudBB improves performance by up to 28.7 times while reducing cost by up to 94.7% compared to the ones without CloudBB.

Some more implementation examples of FUSE are - mp3fs (A VFS to convert FLAC files to MP3 files instantly), Copy-FUSE(To access cloud storage on Copy.com), mtpfs(To mount MTP devices) etc.

39.16.8 Gluster**39.16.9 Lustre**

The Lustre file system [449] is an open-source, parallel file system that supports many requirements of leadership class HPC simulation environments and Enterprise environments worldwide. Because Lustre file systems have high performance capabilities and open licensing, it is often used in supercomputers. Lustre file systems are scalable and can be part of multiple computer clusters with tens of thousands of client nodes, tens of petabytes of storage on hundreds of servers, and more than a terabyte per second of aggregate I/O throughput. Lustre file systems a popular choice for businesses with large data centers, including those in industries such as meteorology, simulation, oil and gas, life science, rich media, and finance. Lustre provides a POSIX compliant interface and many of the largest and most powerful supercomputers on Earth today are powered by the Lustre file system.

39.16.10 IBM Spectrum Scale, formerly GPFS

General Parallel File System (GPFS) was rebranded as IBM Spectrum Scale on February 17, 2015 [698].

Spectrum Scale is a clustered file system, developed by IBM, designed for high performance. It "provides concurrent high-speed file access to applications executing on multiple nodes of clusters" [698] and can be deployed in either shared-nothing or shared disk modes. Spectrum Scale is available on AIX, Linux, Windows Server, and IBM System Cluster 1350 [698]. Due to its focus on performance and scalability, Spectrum Scale has been utilized in compute clusters, big data and analytics - including support for Hadoop Distributed File System (HDFS), backups and restores, and private clouds [292].

39.16.11 GFFS

The Global Federated File System (GFFS) [479] is a computing technology that allows linking of data from Windows, Mac OS X, Linux, AFS, and Lustre file systems into a global namespace, making them available to multiple systems. It is a federated, secure, standardized, scalable, and transparent mechanism to access and share resources across organizational boundaries. It is useful when, for data resources, boundaries do not require application modification and do not disrupt existing data access patterns. It uses FUSE to handle access control and allows research collaborators on remote systems to access a shared file system. Existing applications can access resources anywhere in the GFFS without modification. It helps in rapid development of code, which can then be exported via GFFS and implemented in-place on a given computational resource or Science Gateway.

39.16.12 Public Cloud: Amazon S3

Amazon Simple Storage Service (Amazon S3) [25] is storage object which provides a simple web service interface to store and retrieve any amount of data from anywhere on the web. With Amazon S3, users can store as much data as they want and can scale it up and down based on the requirements. For developers Amazon S3 provides full REST API's and SDK's which can be integrated with third-party technologies. Amazon S3 is also deeply integrated with other AWS services to make it easier to build solutions that use a range of AWS services which include Amazon CloudFront, Amazon CloudWatch, Amazon Kinesis, Amazon RDS, Amazon Glacier etc. Amazon S3 provides automatic encryption of data once the data is uploaded in the cloud. Amazon S3 uses

the concept of Buckets and Objects for storing data wherein Buckets are used to store objects. Amazon S3 services can be used using the Amazon Console Management. [636] The steps for using the Amazon S3 are as follows: (1) Sign up for Amazon S3 (2) After sign up, create a Bucket in your account, (3) Create and object which might be an file or folder, and (4) Perform operations on the object which is stored in the cloud.

39.16.13 Azure Blob

Azure Blob storage is a service that stores unstructured data in the cloud as objects/blobs. Blob storage can store any type of text or binary data, such as a document, media file, or application installer [33] Blob storage is also referred to as object storage. The word 'Blob' expands to Binary Large OObject. There are three types of blobs in the service offe- red by Windows Azure namely block, append and page blobs. [224] 1. Block blobs are collection of individual blocks with unique block ID. The block blobs allow the users to upload large amount of data. 2. Append blobs are optimized blocks that helps in making the operations efficient. 3. Page blobs are compilation of pages. They allow random read and write operations. While creating a blob, if the type is not specified they are set to block type by default. All the blobs must be inside a container in your storage. Azure Blob storage is a service for storing large amounts of unstructured object data, such as text or binary data, that can be accessed from anywhere in the world via HTTP or HTTPS. You can use Blob storage to expose data publicly to the world, or to store application data privately. Common uses of Blob storage include serving images or documents directly to a browser, storing files for distributed access, streaming video and audio, storing data for backup and restore, disaster recovery, and archiving and storing data for analysis by an on-premises or Azure-hosted service. Azure Storage is massively scalable and elastic with an auto-partitioning system that automatically load-balances your data. Blob storage is a specialized storage account for storing your unstructured data as blobs (objects) in Azure Storage. Blob storage is similar to existing general-purpose storage accounts and shares all the great durability, availability, scalability, and performance features. Blob storage has two types of access tiers that can be specified, hot access tier, which will be accessed more frequently, and a cool access tier, which will be less frequently accessed. There are many reasons why you should consider using BLOB storage. Perhaps you want to share files with clients, or off-load some of the static content from your web servers to reduce the load on them. [33]

39.16.14 Google Cloud Storage

Google Cloud Storage is the cloud enabled storage offered by Google. [248] It is unified object storage. To have high availability and performance among different regions in the geo-redundant storage offering. If you want high availability and redundancy with a single region one can go for "Regional" storage. Nearline and Coldline' are the different archival storage techniques. "Nearline" storage offering is for the archived data which the user access less than once a month . "Coldline" storage is the storage which is used for the data which is touched less than once a year.

All the data in Google Cloud storage belongs inside a project. A project will contains different buckets. Each bucket has different objects. We need to make sure that the name of the bucket is unique across all Google cloud name space . And the name of the objects should unique in a bucket.

39.17 Interoperability

39.17.1 Cloudmesh

Cloudmesh client allows to easily manage virtual machines, containers, HPC tasks, through a convenient client and API. Hence cloudmesh is not only a multi-cloud, but a multi-hpc environment that allows also to use container technologies. Cloudmesh is currently developed as part of classes taught at Indiana University.

39.17.2 Libvirt

Libvirt is an open source API to manage hardware virtualization developed by Red Hat. It is a standard C library but has accessibility from other languages such as Python, Perl, Java and others. [373] Multiple virtual machine monitors(VMM) or hypervisors are supported such as KVM,QEMU, Xen, Virtuozzo, VMWare ESX, LXC, and BHyve. It can be divided into five categories such as hypervisor connection, domain, network, storage volume and pool. [334] It is accessible by many operating systems such as Linux, FreeBSD, Mac OS, and Windows OS.

39.17.3 Libcloud

:cite::‘www-libcloudwiki’ Libcloud is a python library that allows to interact with several popular cloud service providers. It is primarily designed to ease development of software products that work with one or more cloud services supported by Libcloud. It provides a unified API to interact with these different cloud services. Current API includes methods for list, reboot, create, destroy, list images and list sizes. :cite::‘www-libclouddoc’ lists Libcloud key component APIs Compute, Storage, Load Balancers, DNS, Container and Backup. Compute API allows users to manage cloud servers. Storage API allows users to manage cloud object storage and also provides CDN management functionality. Load balancer, DNS and Backup API’s allows users to manage their respective functionalities, as services, and related products of different cloud service providers. Container API allows users to deploy containers on to container virtualization platforms. Libcloud supports Python 2, Python 3 and PyPy.

39.17.4 JClouds

[91] Primary goals of cross-platform cloud APIs is that application built using these APIs can be seamlessly ported to different cloud providers. The APIs also bring interoperability such that cloud platforms can communicate and exchange information using these common or shared interfaces. Jclouds or apache jclouds [325] is a java based library to provide seamless access to cloud platforms. Jclouds library provides interfaces for most of cloud providers like docker, openstack, amazon web services, microsoft azure, google cloud engine etc. It will allow users build applications which can be portable across different cloud environments. Key components of jcloud are:

1. Views: abstracts functionality from a specific vendor and allow user to write more generic code. For example odbc abstracts the underlying relational data source. However, odbc driver converts to native format. In this case user can switch databases without rewriting the application. Jcloud provide following views: blob store, compute service, loadBalancer service
2. API: APIs are requests to execute a particular functionality. Jcloud provide a single set of APIs for all cloud vendors which is also location aware. If a cloud vendor doesn’t support customers from a particular region the API will not work from that region.
3. Provider: a particular cloud vendor is a provider. Jcloud uses provider information to initialize

its context.

4. Context: it can be termed as a handle to a particular provider. Its like a ODBC connection object. Once connection is initialized for a particular database, it can be used to make any api call.

Jclouds provides test library to mock context, APIs etc to different providers so that user can write unit test for his implementation rather than waiting to test with the cloud provider. Jcloud library certifies support after testing the interfaces with live cloud provider. These features make jclouds robust and adoptable, hiding most of the complexity of cloud providers.

39.17.5 TOSCA

39.17.6 OCCI

The Open Cloud Computing Interface (OCCI) is a RESTful Protocol and API that provides specifications and remote management for the development of "interoperable tools" [715]. It supports IaaS, PaaS and SaaS and focuses on integration, portability, interoperability, innovation and extensibility. It provides a set of documents that describe an OCCI Core model, contain best practices of interaction with the model, combined into OCCI Protocols, explain methods of communication between components via HTTP protocol introduced in the OCCI Renderings, and define infrastructure for IaaS presented in the OCCI Extensions.

The current version 1.2 OCCI consists of seven documents that identify required and optional components. Of the Core Model. In particular, the following components are required to implement: a)Core Model, b)HTTP protocol, c)Text rendering and d)JSON rendering. Meanwhile, Infrastructure, Platform and SLA models are optional. The OCCI Core model defines instance types and

provides a layer of abstraction that allows the OCCI client to interact with the model without knowing of its potential structural changes. The model supports extensibility via inheritance and using mixin types that represent ability to add new components and capabilities at run-time. [716]

The OCCI Protocol defines the common set of names provided for the IaaS cloud services user that specify requested system requirements. It is often denoted as "resource templates" or "flavours" [719].

OCCI RESTful HTTP Protocol describes communications between server and client on OCCI platform via HTTP protocol [717]. It defines a minimum set of HTTP headers and status codes to ensure compliance with the OCCI Protocol. Separate requirements for Server and Client for versioning need to be implemented using HTTP 'Server' header and 'User-Agent' header respectively.

JSON rendering [718] protocol provides JSON specifications to allow "render OCCI instances independently of the protocol being used." In addition, it provides details of the JSON object declaration, OCCI Action Invocation, object members required for OCCI Link Instance Rendering, "location maps to OCCI Core's source and target model attributes and kind maps to OCCI Core's target" to satisfy OCCI Link Instance Source/Target Rendering requirements. Finally, it specifies various attributes and collection rendering requirements. The text rendering process is deprecated and will be removed from the next major version [720].

39.17.7 CDMI

The Storage Networking Industry Association (SNIA) [71] is a non-profit organization formed by various companies, suppliers and consumers of data storage and network products. SNIA defines

various standards to ensure the quality and interoperability of various storage systems. One of the standards defined by SNIA to for providers and users of cloud is Cloud Data Management Interface (CDMI). According latest issue of CDMI [72], "CDMI International Standard is intended for application developers who are implementing or using cloud storage. It documents how to access cloud storage and to manage the data stored there." It defines functional interface for applications that will use cloud for various functionalities like create, retrieve, update and delete data elements from the cloud. These interface could be used to manage containers along with the data. The interface could be used by administrative and management applications as well. Also, the CDMI specification uses RESTful principles in the interface design. All the standards issued on CDMI can be found on SNIA web page [73].

39.17.8 Whirr

Apache Whirr is a set of libraries for running cloud services, which provides a cloud-neutral way to run services [657]. This is achieved by using cloud-neutral provisioning and storage libraries such as jclouds and libcloud. Whirr's API should be built on top these libraries and is not exposed to the users. It is also a common service API, in which the details of its working are, particular to the service. Whirr provides smart defaults for services by which any properly configured system can run quickly, while still being able to override settings as needed. Whirr can also be used as a command line tool for deploying clusters. It uses low level API libraries to work with providers which was mentioned in the [658].

39.17.9 Saga

SAGA(Simple API for Grid Applications) provides an abstraction layer to make it easier for applications to utilize and exploit infra effectively. With infrastructure being changed continuously its becoming difficult for most applications to utilize the advances in hardware. SAGA API provides a high level abstraction of the most common Grid functions so as to be independent of the diverse and dynamic Grid environments [327]. This shall address the problem of applications developers developing an application tailored to a specific set of infrastructure. SAGA allows computer scientists to write their applications at high level just once and not to worry about low level hardware changes. SAGA provides this high level interface which has the underlying mechanisms and adapters to make the appropriate calls in an intelligent fashion so that it can work on any underlying grid system. "SAGA was built to provide a standardized, common interface across various grid middleware systems and their versions" [235].

As SAGA is to be implemented on different types of middleware it does not specify a single security model but provides hooks to interfaces of various security models. The SAGA API provides a set of packages to implement its objectivity : SAGA supports data management, resource discovery, asynchronous notification, event generation, event delivery etc. It does so by providing set of functional packages namely SAGA file package, replica package, stream package, RPC package, etc. SAGA provides interoperability by allowing the same application code to run on multiple grids and also communicate with applications running on others [327].

39.17.10 Genesis

39.18 DevOps

39.18.1 Docker (Machine, Swarm)

Docker is an open-source container-based technology. A container allows a developer to package up an application and all its parts including the stack it runs on, dependencies it is associated with and everything the application requires to run within an isolated environment. Docker separates Application from the underlying Operating System in a similar way as Virtual Machines separates the Operating System from the underlying hardware. Dockerizing an application is lightweight in comparison with running the application on the Virtual Machine as all the containers share the same underlying kernel, the Host OS should be same as the container OS (eliminating guest OS) and an average machine cannot have more than few VMs running on them.

Docker Machine is a tool that lets you install Docker Engine on virtual hosts, and manage the hosts with docker-machine commands [388]. You can use Machine to create Docker hosts on your local Mac or Windows machine, on your company network, in your data center, or on cloud providers like AWS or Digital Ocean. For Docker 1.12 or higher swarm mode is integrated with the Docker Engine, but on the older versions with Machine's swarm option, user can configure a swarm cluster. Docker Swarm provides native clustering capabilities to turn a group of Docker engines into a single, virtual Docker Engine. "With these pooled resources user can scale out your application as if it were running on a single, huge computer" [160]. Docker Swarm can be scaled up to 1000 Nodes or up to 50,000 containers

39.18.2 Puppet

Puppet is an open source software configuration management tool [494]. This aims at automatic configuration of the software applications and infrastructure. This configuration is done using the easy to use language. Puppet works on major linux distributions and also on Microsoft Windows, it is also cross-platform application making it easy to manage and portable. [493]

Puppet works with a client server model. All the clients (nodes) which need to be managed will have 'Puppet Agent' installed and 'Puppet Master' contains the configuration for different hosts. This daemon process runs on master server. The connection between 'Puppet Master' and 'Puppet agent' will be established using the secured SSL connection. The configuration at client will be validated as per the setup in Puppet master at a predefined interval. If configuration at client is not matching with the master puppet agent fetches the changes from master. [281]

Puppet is developed by Puppet Labs using Ruby language and released as GNU General Public License (GPL) until version 2.7.0 and the Apache License 2.0 after that. [494]

39.18.3 Chef

Chef is a configuration management tool. It is implemented in Ruby and Erlang. Chef can be used to configure and maintain servers on-premise as well as cloud platforms like Amazon EC2, Google Cloud Platform and Open Stack. The book [386] explains the use of concept called 'recipes' in Chef to manage server applications and utilities such as database servers like MySQL, or HTTP servers like Apache Hadoop and systems like Apache Hadoop.

Chef is available in open source version and it also has commercial products for the companies which need it [115]

39.18.4 Ansible

Ansible is an IT automation tool that automates cloud provisioning, configuration management, and application deployment. [515] Once Ansible gets installed on a control node, which is an agentless architecture, it connects to a managed node through the default OpenSSH connection type. [679]

As with most configuration management softwares, Ansible distinguishes two types of servers: controlling machines and nodes. First, there is a single controlling machine which is where orchestration begins. Nodes are managed by a controlling machine over SSH. The controlling machine describes the location of nodes through its inventory.

Ansible manages machines in an agent-less manner. Ansible is decentralized, if needed, Ansible can easily connect with Kerberos, LDAP, and other centralized authentication management systems.

39.18.5 SaltStack

SaltStack (also Salt) platform is a Python-based open-source configuration management software and remote execution engine, which makes systems and configuration management software for the orchestration and automation of CloudOps, ITOps and DevOps at scale [525]. SaltStack is used to manage all the data center things including any cloud, infrastructure, virtualization, application stack, software or code. Salt is built on two major concepts, which are clearly mentioned in [412] as remote execution and configuration management. In the remote execution system, Salt leverages Python to accomplish complex tasks with single-function calls. The configuration management system in Salt, called States, builds upon the remote execution foundation to create repeatable, enforceable configuration for the minions (connects to the master and treats the master as the source)

39.18.6 Boto

The latest version of Boto is Boto3 [219]. Boto3 is the Amazon Web Services (AWS) Development Kit (SDK) for Python [222]. It enables the Python developers to make use of services like Amazon S3 and Amazon EC2 [221]. It provides object oriented APIs along with low-level direct service [220]. It provides simple in-built functions and interfaces to work with Amazon S3 and EC2.

Boto3 has two distinct levels of APIs - client and resource [221]. One-to-one mappings to underlying HTTP API is provided by the client APIs. Resource APIs provide resource objects and collections to perform various actions by accessing the attributes. Boto3 also comes with 'waiters'. Waiters are used for polling status changes in AWS, automatically. Boto3 has these waiters for both the APIs - client as well as resource.

39.18.7 Cobbler

Cobbler is a Linux provisioning system that facilitates and automates the network based system installation of multiple computer operating systems from a central point using services such as DHCP, TFTP and DNS [405]. It is a nifty piece of code that assembles all the usual setup bits required for a large network installation like TFTP, DNS, PXE installation trees and automates the process. It can be configured for PXE, reinstallations and virtualized guests using Xen, KVM or VMware. Cobbler interacts with the koan program for re-installation and virtualization support. Cobbler builds the Kickstart mechanism and offers installation profiles that can be applied to one or many machines. Cobbler has features to dynamically change the information contained in a kickstart template (definition), either by passing variables called ksmeta or by using so-called

snippets.

39.18.8 Xcat

xCAT is defined as extreme cloud/cluster administration toolkit. Tnd his open source software was developed by IBM and utilized on clusters based on either linux or a version of UNIX called AIX. With this service administrator is enabled with a number of capabilities including parallel system management, provision OS usage on virtual machines, and manage all systems remotely. [724] xCAT works with various cluster types such as high performance computing, horizontal scaling web farms, administrative, and operating systems. [296]

39.18.9 Razor

Razor is a hardware provisioning application, developed by Puppet Labs and EMC. Razor was introduced as open, pluggable, and programmable since most of the provisioning tools that existed were vendor-specific, monolithic, and closed. According to [495] it can deploy both bare-metal and virtual systems. During boot the Razor client automatically discovers the inventory of the server hardware – CPUs, disk, memory, etc., feeds this to the Razor server in real-time and the latest state of every server is updated. It maintains a set of rules to dynamically match the appropriate operating system images with server capabilities as expressed in metadata. User-created policy rules are referred to choose the preconfigured model to be applied to a new node. The node follows the model's directions, giving feedback to Razor as it completes various steps as specified in [496]. Models can include steps for handoff to a DevOps system or to any other system capable of controlling the node.

39.18.10 Juju

Juju (formerly Ensemble) [88] is software from Canonical that provides open source service orchestration. It is used to easily and quickly deploy and manage services on cloud and physical servers. Juju charms can be deployed on cloud services such as Amazon Web Services (AWS), Microsoft Azure and OpenStack. It can also be used on bare metal using MAAS. Specifically [338] lists around 300 charms available for services available in the Juju store. Charms can be written in any language. It also supports Bundles which are pre-configured collection of Charms that helps in quick deployment of whole infrastructure.

39.18.11 Foreman

39.18.12 OpenStack Heat

Openstack Heat, a template deployment service was the project launched by Openstack, a cloud operating system similar to AWS Cloud Formation. [380] states - Heat is an orchestration service which allows us to define resources over the cloud and connections amongst them using a simple text file called referred as a ‘template’ . "A Heat template describes the infrastructure for a cloud application in a text file that is readable and writable by humans, and can be checked into version control" [277].

Once the execution environment has been setup and a user wants to modify the architecture of resources in the future, a user needs to simply change the template and check it in. Heat shall make the necessary changes. Heat provides 2 types of template - HOT(Heat Orchestration Template) and CFN (AWS Cloud Formation Template). The HOT can be defined as YAML and is not compatible

with AWS. The CFN is expressed as JSON and follows the syntax of AWS Cloud Formation and thus is AWS compatible. Further, heat provides an additional @parameters section in its template which can be used to parameterize resources to make the template generic.

39.18.13 Sahara

The Sahara product provides users with the capability to provision data processing frameworks (such as Hadoop, Spark and Storm) on OpenStack [450] by specifying several parameters such as the version, cluster topology and hardware node details. As specified in [524] the solution allows for fast provisioning of data processing clusters on OpenStack for development and quality assurance and utilisation of unused computer power from a general purpose OpenStack IaaS Cloud. Sahara is managed via a REST API with a User Interface available as part of OpenStack Dashboard.

39.18.14 Rocks

[520] Rocks provides open cluster distribution solution is build targeting the scientist with less cluster experience to ease the process of deployment, managing, upgrading and scaling high performance parallel computing cluster. It was initially build on linux however the latest version Rocks 6.2 Sidewinder is also available on CentOS. Rocks can help create a cluster in few days with default configuration and software packages. Rocks distribution package comes with high-performance distributed and parallel computing tools. It is used by NASA, the NSA, IBM Austin Research LAB, US Navy and many other institution for their projects.

39.18.15 Cisco Intelligent Automation for Cloud

Cisco Intelligent automation for cloud desires to help different service providers and software professionals in delivering highly secure infrastructure as a service on demand. It provides a foundation for organizational transformation by expanding the uses of cloud technology beyond its infrastructure [122]. From a single self-service portal, it automates standard business processes and sophisticated data center which is beyond the provision of virtual machines. Cisco Intelligent automation for cloud is a unified cloud platform that can deliver any type of service across mixed environments [400]. This leads to an increase in cloud penetration across different business and IT holdings. Its services range from underlying infrastructure to anything-as-a-service by allowing its users to evaluate, transform and deploy the IT and business services in a way they desire.

39.18.16 Ubuntu MaaS

39.18.17 Facebook Tupperware

Facebook Tupperware is a system which provisions services by taking requirements from engineers and mapping them to actual hardware allocations using containers [487]. Facebook Tupperware simplifies the task of configuring and running services in production and allows engineers to focus on actual application logic. The tupperware system consists of a Scheduler, Agent process and a Server Database. The Scheduler consists of set of machines with one of them as master and the others in standby. The machines share state among them. The Agent process runs on each and every machine and manages all the tasks and co-ordinates with the Scheduler. The Server database stores the details of resources available across machines which is used by the scheduler for scheduling jobs and tasks. Tupperware allows for sandboxing of the tasks which allows for isolation of the tasks. Initially isolation was implemented using chroots but now it is switched to

Linux Containers(LXC) .The configuration for the container is done by a specific config file written in a dialect of python by the owner of the process.

39.18.18 AWS OpsWorks

AWS Opsworks is a configuration service provided by Amazon Web Services that uses Chef, a Ruby and Erlang based configuration management tool [685], to automate the configuration, deployment, and management of servers and applications. There are two versions of AWS Opsworks. The first, a fee based offering called AWS OpsWorks for Chef Automate, provides a Chef Server and suite of tools to enable full stack automation. The second, AWS OpsWorks Stacks, is a free offering in which applications are modeled as stacks containing various layers. Amazon Elastic Cloud Compute (EC2) instances or other resources can be deployed and configured in each layer of AWS OpsWorks Stacks [23].

39.18.19 OpenStack Ironic

Ironic [637] project is developed and supported by OpenStack. Ironic provisions bare metal machines instead of virtual machines and functions as hypervisor API that is developed using open source technologies like Preboot Execution Environment (PXE), Dynamic Host Configuration Protocol (DHCP), Network Bootstrap Program (NBP), Trivial File Transfer Protocol (TFTP) and Intelligent Platform Management Interface (IPMI). A properly configured Bare Metal service with the Compute and Network services, could provision both virtual and physical machines through the Compute service's API. But, the number of instance actions are limited, due to physical servers and switch hardware. For example, live migration is not possible on a bare metal instance. The Ironic service has five key components. A RESTful API service, through which other components would interact with the bare metal servers, a Conductor service, various drivers, messaging queue and a database. Ironic could be integrated with other OpenStack projects like Identity (keystone), Compute (nova), Network (neutron), Image (glance) and Object (swift) services.

39.18.20 Google Kubernetes

Google Kubernetes is a cluster management platform developed by Google. According to [359] is an open source system for "automating deployment, scaling and management of containerized applications". It primarily manages clusters through containers as they decouple applications from the host operating system dependencies and allowing their quick and seamless deployment, maintenance and scaling.

Kubernetes components are designed to extensible primarily through Kubernetes API. Kubernetes follows a master-slave architecture, according to [668] Kubernetes Master controls and manages the clusters workload and communications of the system. Its main components are etcd, API server, scheduler and controller manager. The individual Kubernetes nodes are the workers where containers are deployed. The components of a node are Kubelet, Kube-proxy and cAdvisor. Kunernetes makes it easier to run application on public and private clouds. It is also said to be self-healing due to features like auto-restart and auto-scaling.

39.18.21 Buildstep

Buildsteps is an open software developed under MIT license. It is a base for Dockerfile and it activates Heroku-style application. Heroku is a platform-as-service (PaaS) that automates deployment of applications on the cloud. The program is pushed to the PaaS using git push, and

then PaaS detects the programming language, builds, and runs application on a cloud platform [478]. Buildstep takes two parameters: a tar file that contains the application and a new application container name to create a new container for this application. Build script is dependent on buildpacks that are pre-requisites for buildstep to run. The builder script runs inside the new container. The resulting build app can be run with Docker using docker build -t your_app_name command. [234].

39.18.22 Gitreceive

Gitreceive is used to create an ssh+git user which can accept repository pushes right away and also triggers a hook script. Gitreceive is used to push code anywhere as well as extend your Git workflow. "Gitreceive dynamically creates bare repositories with a special pre-receive hook that triggers your own general gitreceive hook giving you easy access to the code that was pushed while still being able to send output back to the git user" Gitreceive can also be used to provide feedback to the user not only just to trigger code on git push. Gitreceive can be used for the following: "a)for putting a git push deploy interface in front of App Engine b)Run your company build/test system as a separate remote c)Integrate custom systems into your workflow d)Build your own Heroku e)Push code anywhere" [712].

39.18.23 OpenTOSCA

The Topology and Orchestration Specification for Cloud Applications, TOSCA is a new standard facilitating platform independent description of Cloud applications. OpenTOSCA is a runtime for TOSCA-based Cloud applications. The runtime enables fully automated plan-based deployment and management of applications defined in the OASIS TOSCA packaging format CSAR, Cloud Service ARchive. The key tasks of OpenTOSCA, are to operate management operations, run plans, and manage state of the TOSCA [94].

39.18.24 Winery

Eclipse Winery [206] is a "web-based environment to graphically model [Topology and Orchestration Specification for Cloud Applications] TOSCA topologies and plans managing these topologies." Winery [356] is a "tool offering an HTML5-based environment for graph-based modeling of application topologies and defining reusable component and relationship types." This web-based [356] interface enables users to drag and drop icons to create automated "provisioning, management, and termination of applications in a portable and interoperable way." Essentially, this web-based interface [356] allows users to create an application topology, which "describes software and hardware components involved and relationships between them" as well a management plan, which "captures knowledge [regarding how] to deploy and manage an application."

39.18.25 CloudML

CloudML a research project initiated by SINTEF in 2011 [549]. Cloud computing facilitates to shared and virtualized computer capabilities like storage, memory, CPU, GPU and networks, to user. There is multiple cloud provider, also the IaaS(Infrastructure-as-a-service) and PaaS(Platform-as-a-service). To operate multiple cloud for applications, which requires multiple private, public, or hybrid clouds, limit the capability of each cloud solution. Solution provided by such cloud will get incompatible with others. So, to providing the solution which can compatible with multi-cloud platform is a tedious job. To achieve this CloudML provides a "domain-specific modelling language along with run time environment" [549]. It provides the interoperability and provide vendor lock-in,

also it provides the solution on specification of provisioning, deployment, and adaptation concerns of multi-cloud systems. At design time as well as runtime [549]. CloudML provides two level of abstraction while developing model for multi-cloud application:

- Cloud Provider-Independent Model (CPIM), this specifies the provisioning and deployment.
- Cloud Provider-Specific Model (CPSM), which filters the provisioning and deployment of multiple cloud application, according to its cloud.

This two abstract approach help CloudML to achieve the multi-cloud application support [550].

39.18.26 Blueprints

In [290], it is explained that "IBM Blueprint has been replaced by IBM Blueworks Live." In [289], IBM Blueworks Live is described "as a cloud-based business process modeller, belonging under the set of IBM SmartCloud applications" that as [291] states "drive[s] out inefficiencies and improve[s] business operations." Similarly to Google Docs, IBM Blueworks Live is "designed to help organizations discover and document their business processes, business decisions and policies in a collaborative manner." While Google Docs and IBM Blueworks Live are both simple to use in a collaborative manner, [289] explains that IBM Blueworks Live has the "capabilities to implement more complex models."

39.18.27 Terraform

Terraform, developed by HashiCorp, is an infrastructure management tool, it has an open source platform as well as an enterprise version and uses infrastructure as a code to increase operator productivity. It's latest release is Terraform 0.8 According to the website [589] it enables users to safely and predictably create, change and improve the production infrastructure and codifies APIs into declarative configuration files that can be shared amongst other users and can be treated as a code, edited, reviewed and versioned at the same time. The book [623] explains that it can manage the existing and popular service it provides as well as create customized in-house solutions. It builds an execution plan that describes what it can do next after it reaches a desired state to accomplish the goal state. It provides a declarative executive plan which is used for creating applications and implementing the infrastructures. Terraform is mainly used to manage cloud based and SaaS infrastructure, it also supports Docker and VMWare vSphere.

39.18.28 DevOpSlang

DevOpSlang serves as means of collaboration and provides the foundation to automate deployment and operations of an application. Technically, it is a domain specific language based on JavaScript Object Notation (JSON). JSON Schema is used to define a formal schema for DevOpSlang and complete JSON Schema definition of DevOpSlang is publicly available on GitHub project DevOpSlang: <http://github.com/jojow/devopslang> Devopsfiles are the technical artifacts (Unix shell commands, Chef Scripts, etc.) rendered using DevOpSlang to implement operations. Beside some meta data such as 'version' and 'author' Devopsfile defines operations like 'start' consisting of a single or multiple actions which specifies the command to run the application. Similarly, a 'build' operation can be defined to install the dependencies required to run the application. Different abstraction levels may be combined consistently such as a 'deploy' operation consisting of actions on the level of Unix shell commands and actions using portable Chef cookbooks [650].

39.18.29 Any2Api

This framework [651] allows user to wrap an executable program or scripts, for example scripts, chef cookbooks, ansible playbooks, juju charms, other compiled programs etc. to generate APIs from your existing code. These APIs are also containerized so that they can be hosted on a docker container, vagrant box etc Any2Api helps to deal with problems like scale of application, technical expertise, large codebase and different API formats. The generated API hide the tool specific details simplifying the integration and orchestration different kinds of artifacts. The APIfication framework contains various modules:

1. Invokers, which are capable of running a given type of executable for example cookbook invoker can be used to run Chef cookbooks
2. Scanners, which are capable of scanning modules of certain type for example cookbook scanner scans Chef cookbooks.
3. API impl generators, which are doing the actual work to generate the API implementation.

The final API implementation [649] is packaged with executable in container. The module is packaged as npm module. Currently any2api-cli provides a command line interface and web based interface is planned for future development. Any2Api is very useful for by devops to orchestrate open source ecosystem without dealing with low level details of chef cookbook or ansible playbook or puppet. It can also be very useful in writing microservices where services talk to each other using well defined APIs.

39.19 IaaS Management from HPC to hypervisors

39.19.1 Xen

Xen is the only open-source bare-metal hypervisor based on microkernel design [725]. The hypervisor runs at the highest privilege among all the processes on the host. Its responsibility is to manage CPU and memory and handle interrupts [726]. Virtual machines are deployed in the guest domain called DomU which has no access privilege to hardware. A special virtual machine is deployed in the control domain called Domain 0. It contains hardware drivers and the toolstack to control the VMs and is the first VM to be deployed. Xen supports both Paravirtualization and hardware assisted virtualization. The hypervisor itself has a very small footprint. It's being actively maintained by Linux Foundation under the trademark *XEN Project*. Some of the features included in the latest releases include em Reboot-free Live Patching (to enable application of security patches without rebooting the system) and KCONFIG support (compilation support to create a lighter version for requirements such as embedded systems) [727].

39.19.2 KVM

It is an acronym for Kernel-based Virtual Machine for the Linux Kernel that turns it into a hypervisor upon installation. It was originally developed by Qumranet in 2007 [361]. It has a kernel model and uses kernel as VMM. It only supports fully virtualized VMs. It is very active for Linux users due to its ease of use, it can be completely controlled by ourselves and there is an ease for migration from or to other platforms. It is built to run on a x86 machine on an Intel processor with virtualization technology extensions (VT-x) or an AMD-V. It supports 32 and 64 bit guests on a 64 bit host and hardware visualization features. The supported guest systems are Solaris, Linux, Windows and BSD Unix [362].

39.19.3 QEMU

QEMU (Quick Emulator) is a generic open source hosted hypervisor [696] that performs hardware virtualization (virtualization of computers as complete hardware platform, certain logical abstraction of their componentry or only the certain functionality required to run various operating systems) :cite-‘www-qemu’ and also emulates CPUs through dynamic binary translations and provides a set of device models, enabling it to run a variety of unmodified guest operating systems.

When used as an emulator, QEMU can run Operating Systems and programs made for one machine (ARM board) on a different machine (e.g. a personal computer) and achieve good performance by using dynamic translations. When used as a virtualizer, QEMU achieves near native performance by executing the guest code directly on the host CPU. QEMU supports virtualization when executing under the Xen hypervisor or using KVM kernel module in Linux [497].

Compared to other virtualization programs like VMWare and VirtualBox, QEMU does not provide a GUI interface to manage virtual machines nor does it provide a way to create persistent virtual machine with saved settings. All parameters to run virtual machine have to be specified on a command line at every launch. It’s worth noting that there are several GUI front-ends for QEMU like virt-manager and gnome-box.

39.19.4 Hyper-V

Hyper-V is a native hypervisor which was first released alongside Windows Server 2008. It is available free of charge for all the Windows Server and some client operating systems since the release. Microsoft Hyper-V, is also codenamed as Viridian and formerly known as Windows Server Virtualization, is a native hypervisor. Xbox One also include Hyper-V, in which it would launch both Xbox OS and Windows 10. [695]

Hyper-V is used to create virtual machines on x86-64 systems which are running Windows. Windows 8 onwards, Hyper-V supersedes Windows Virtual PC as the hardware virtualization component of the client editions of Windows NT. A server computer running Hyper-V can be configured to expose individual virtual machines to one or more networks.

39.19.5 VirtualBox

39.19.6 OpenVZ

OpenVZ (Open Virtuozzo) is an operating system-level virtualization technology for Linux. It allows a physical server to run multiple isolated operating system instances, called containers, virtual private servers, or virtual environments (VEs). OpenVZ is similar to Solaris Containers and LXC. [452] While virtualization technologies like VMware and Xen provide full virtualization and can run multiple operating systems and different kernel versions, OpenVZ uses a single patched Linux kernel and therefore can run only Linux. All OpenVZ containers share the same architecture and kernel version. This can be a disadvantage in situations where guests require different kernel versions than that of the host. However, as it does not have the overhead of a true hypervisor, it is very fast and efficient. Memory allocation with OpenVZ is soft in that memory not used in one virtual environment can be used by others or for disk caching. [282] While old versions of OpenVZ used a common file system (where each virtual environment is just a directory of files that is isolated using chroot), current versions of OpenVZ allow each container to have its own file system. OpenVZ has four main features, [187] 1. OS virtualization: A container (CT) looks and behaves like a regular Linux system. It has standard startup scripts; software from vendors can run inside a container without OpenVZ-specific modifications or adjustment; A user can change any

configuration file and install additional software; Containers are completely isolated from each other and are not bound to only one CPU and can use all available CPU power. 2. Network virtualization: Each CT has its own IP address and CTs are isolated from the other CTs meaning containers are protected from each other in the way that makes traffic snooping impossible; Firewalling may be used inside a CT 3. Resource management: All the CTs are use the same kernel. OpenVZ resource management consists of four main components: two-level disk quota, fair CPU scheduler, disk I/O scheduler, and user bean counters. 4. Checkpointing and live migration: Checkpointing allows to migrate a container from one physical server to another without a need to shutdown/restart a container. This feature makes possible scenarios such as upgrading your server without any need to reboot it: if your database needs more memory or CPU resources, you just buy a newer better server and live migrate your container to it, then increase its limits.

39.19.7 LXC

LXC (Linux Containers) is an operating-system-level virtualization method for running multiple isolated Linux systems (containers) on a control host using a single Linux kernel [376]. LXC are similar to the traditional virtual machines but instead of having separate kernel process for the guest operating system being run, containers would share the kernel process with the host operating system. This is made possible with the implementation of namespaces and cgroups. [660]

Containers are light weighed (As guest operating system loading and booting is eliminated) and more customizable compared to VM technologies. The basis for docker development is also LXC. [377]. Linux containers would work on the major distributions of linux this would not work on Microsoft Windows.

39.19.8 Linux-Vserver

Linux-VServers are used on web hosting services, pooling resources and containing any security breach. [485] “Linux servers consist of three building blocks Hardware, Kernel and Applications” the purpose of kernel is to provide abstraction layer between hardware and application. Linux-Vserver provides VPS securely partitioning the resources on computer system in such a way that process cannot mount denial of service out of the partition.

It utilises the power of Linux kernel and top of it with additional modification provides secure layer to each process (VPS) feel like it is running separate system. By providing context separation, context capabilities, each partition called as security context, chroot barrier created on private directory of each VPS to prevent unauthorized modification. Booting VPS in new secure context is just matter of booting server, context is so robust to boot many server simultaneously.

The virtual servers shares same system calls, shares common file system, process within VS are queued to same scheduler that of host allowing guest process to run concurrently on SMP systems. No additional overhead of network virtualization. These few advantages of Linux-VServer.

39.19.9 OpenStack

OpenStack [561] is a free and open source cloud operating system mostly deployed as infrastructure as a service(IaaS) that allows us to control large pool of computers, storage, and networking resources. OpenStack is managed by OpenStack Foundation [199].

Just like cloud, OpenStack provides infrastructure which runs as platform upon which end users can create applications. Key components of OpenStack include: Nova: which is the primary computing

engine, Swift: which is a storage system for object and files, Neutron: which ensures effective communication between each of the components of the OpenStack. Other components include: Cinder, Horizon, Keystone, Glance, Ceilometer and Heat. The main goal of Openstack is to allow business to build Amazon-like cloud services in their own data centers. OpenStack is licensed under the Apache 2.0 license [45]

39.19.10 OpenNebula

According to OpenNebula webpage [5] it provides simple but feature-rich and flexible solutions for the comprehensive management of virtualized data centers to enable private, public and hybrid IaaS clouds. It is a cloud computing platform for managing heterogeneous distributed data centers infrastructures. The OpenNebula toolkit includes features for management, scalability, security and accounting. It is used in various sectors like hosting providers, telecom providers, telecom operators, IT service providers, supercomputing centers, research labs, and international research projects [447]. More about OpenNebula can be found in the following paper that is published at IEEE computer society [406]

39.19.11 Eucalyptus

Eucalyptus is a Linux-based open source software framework for cloud computing that implements Infrastructure as a Service (IaaS). IaaS are systems that give users the ability to run and control entire virtual machine instances deployed across a variety of physical resources [431]. Eucalyptus is an acronym for “Elastic Utility Computing Architecture for Linking Your Programs to Useful Systems.”

A Eucalyptus private cloud is deployed on an enterprise’s data center infrastructure and is accessed by users over the enterprise’s intranet. Sensitive data remains entirely secure from external interference behind the enterprise firewall [601].

39.19.12 Nimbus

Nimbus Infrastructure [426] is an open source IaaS implementation. It allows deployment of self-configured virtual clusters and it supports configuration of scheduling, networking leases, and usage metering.

Nimbus Platform [425] provides an integrated set of tools which enable users to launch large virtual clusters as well as launch and monitor the cloud apps. It also includes service that provides auto-scaling and high availability of resources deployed over multiple IaaS clouds. The Nimbus Platform tools are cloudinit.d, Phantom and Context Broker. In this paper [167], the use of Nimbus Phantom to deploy auto-scaling solution across multiple NSF FutureGrid clouds is explained. In this implementation Phantom was responsible for deploying instances across multiple clouds and monitoring those instances. Nimbus platform supports Nimbus, Open Stack, Amazon and several other clouds.

39.19.13 CloudStack

Apache CloudStack is open source software designed to deploy and manage large networks of virtual machines, as a highly available, highly scalable Infrastructure as a Service (IaaS) cloud computing platform. It uses existing hypervisors such as KVM, VMware vSphere, and XenServer/XCP for virtualization. In addition to its own API, CloudStack also supports the Amazon Web Services

(AWS) API and the Open Cloud Computing Interface from the Open Grid Forum. [598]

CloudStack features like built-in high-availability for hosts and VMs, AJAX web GUI for management, AWS API compatibility, Hypervisor agnostic, snapshot management, usage metering, network management (VLAN's, security groups), virtual routers, firewalls, load balancers and multi-role support. [680]

39.19.14 CoreOS

[139] states that “CoreOS is a linux operating system used for clustered deployments.” CoreOS allows applications to run on containers. CoreOS can be run on clouds, virtual or physical servers. CoreOS allows the ability for automatic software updates in order to make sure containers in cluster are secure and reliable. It also makes managing large cluster environments easier. CoreOS provides open source tools like CoreOS Linux, etcd,rkt and flannel. CoreOS also has commercial products Kubernetes and CoreOS stack. In CoreOS linux service discovery is achieved by etcd, applications are run on Docker and process management is achieved by fleet.

39.19.15 rkt

rkt is an container manager developed by CoreOS [138] designed for Linux clusters. It is an alternative for Docker runtime and is designed for server environments with high security and compositiby requirement. It is the first implementation of the open container standard called *App Container* or *appc* specification but not the only one. It is a standalone tool that lives outside of the core operating system and can be used on variety of platforms such as Ubuntu, RHEL, CentOS, etc. rkt implements the facilities specified by the App Container as a command line tool. It allows execution of App Containers with pluggable isolation and also varying degrees of protection. Unlike Docker, rkt runs containers as un-privileged users making it impossible for attackers to break out of the containers and take control of the entire physical server. rkt’s primary interface comprises a single executable allowing it easily integrate with existing init systems and also advanced cluster environments. rkt is open source and is written in the Go programming language [230].

39.19.16 VMware ESXi

VMware ESXi (formerly ESX) is an enterprise-class, type-1 hypervisor developed by VMware for deploying and serving virtual computers [664]. The name ESX originated as an abbreviation of Elastic Sky X. ESXi installs directly onto your physical server enabling it to be partitioned into multiple logical servers referred to as virtual machines. Management of VMware ESXi is done via APIs. This allows for an *agent-less* approach to hardware monitoring and system management. VMware also provides remote command lines, such as the vSphere Command Line Interface (vCLI) and PowerCLI, to provide command and scripting capabilities in a more controlled manner. These remote command line sets include a variety of commands for configuration, diagnostics and troubleshooting. For low-level diagnostics and the initial configuration, menu-driven and command line interfaces are available on the local console of the server [640].

39.19.17 vSphere and vCloud

vSphere was developed by VMware and is a cloud computing virtualization platform. [641] vSphere is not one piece of software but a suite of tools that contains software such as vCenter, ESXi, vSphere client and a number of other technologies. ESXi server is a type 1 hypervisor on a physical machine of which all virtual machines are installed. The vSphere client then allows

administrators to connect to the ESXi and manage the virtual machines. The vCenter server is a virtual machine that is also installed on the ESXi server which is used in environments when multiple ESXi servers exist. Similarly, vCloud is also a suite of applications but for establishing an infrastructure for a private cloud. [98] The suite includes the vsphere suite, but also contains site recovery management for disaster recovery, site networking and security. Additionally, a management suite that can give a visual of the infrastructure to determine where potential issues might arise.

39.19.18 Amazon

Amazon's AWS (Amazon Web Services) is a provider of Infrastructure as a Service (IaaS) on cloud. It provides a broad set of infrastructure services, such as computing, data storage, networking and databases. One can leverage AWS services by creating an account with AWS and then creating a virtual server, called as an instance, on the AWS cloud. In this instance you can select the hard disk volume, number of CPUs and other hardware configuration based on your application needs. You can also select operating system and other software required to run your application. AWS lets you select from the countless services. Some of them are mentioned below:

- Amazon Elastic Computer Cloud (EC2) - Amazon Simple Storage Service (Amazon S3) - Amazon CloudFront - Amazon Relational Database Service (Amazon RDS) - Amazon SimpleDB - Amazon Simple Notification Service (Amazon SNS) - Amazon Simple Queue Service (Amazon SQS) - Amazon Virtual Private Cloud (Amazon VPC)

Amazon EC2 and Amazon S3 are the two core IaaS services, which are used by cloud application solution developers worldwide. [586]

Improve: all of them need bibentries

39.19.19 Azure

39.19.20 Google and other public Clouds

A public cloud is a scenario where a provider provides services such as infrastructure or applications to the public over the internet. Google cloud generally refers to services such as cloud print, connect, messaging, storage and platform [244]. Google cloud print allows a print-aware application on a device, installed on a network, to provide prints to any printer on that network. Cloud connect allows an automatic storage and synchronization of Microsoft word documents, power-points and excel sheets to Google docs while preserving the Microsoft office formats. In certain cases, developers require important notifications to be sent to applications targeting android operating system. Google cloud messaging provides such services. Google cloud platform allows the developers to deploy their mobile, web and backend solutions on a highly scalable and reliable infrastructure [245]. It gives developers a privilege of using any programming language. Google cloud platform provides a wide range of products and services including networking, storage, machine learning, big data, authentication and security, resource management, etc. In general, public clouds provide services to different end users with the usage of the same shared infrastructure [652]. Windows Azure services platform, Amazon elastic compute cloud and Sun cloud are few examples of public clouds.

39.19.21 Networking: Google Cloud DNS

Under the umbrella of google cloud platform, helps user to publish their domain using Google's infrastructure. It is highly scalable, low latency, high availability DNS service residing on infrastr-

ture same as google.

It is build around projects a resource container, domain for access control, and billing configuration. Managed zones holds records for same DNS name. The resource record sets collection holds current state of the DNS that make up managed zones it is unmodifiable or cannot be modified easily and changes to record sets. It supports A address records, AAAA IPv6, CAA Certificate authority, CNAME canonical name, MX mail exchange, NAPTR naming authority pointer, NS Name server record, SOA start of authority, SPF Sender policy framework, "SRV" service locator, TXT text record.

39.19.22 Amazon Route 53

Amazon Route 53 is a DNS (Domain Name System) service that gives developers and businesses a reliable way to route end users to Internet applications. The number 53 refers to TCP or UDP port 53, where DNS server requests are addressed [24].

When using Route 53 as your DNS provider, in case of a recursion, the query of fetching an IP address (of a website or application) always goes to the closest server location to reduce query latency. The Route 53 server returns the IP address enabling the browser to load the website or application. Route 53 can also be used for registering domain names and arranging DNS health checks to monitor the server [653].

39.20 Cross-Cutting Functions

39.20.1 Monitoring

Ambari

Apache Ambari is an open source platform that enables easy management and maintenance of Hadoop clusters, regardless of cluster size. Ambari has a simplified Web UI and robust REST API for automating and controlling cluster operations. [31] illustrates Ambari to provide key benefits including easy installation, configuration, and management with features such as Smart Configs and cluster recommendations and Ambari Blueprints, to provide repeatable and automated cluster creation. Ambari provides a centralized security setup that automates security capabilities of clusters. Ambari provides a holistic view for cluster monitoring and provides visualizations for operation metrics. [29] provides documentation about Ambari, including a quick start guide for installing a cluster with Ambari. [30] provides the project documents for ambari on github.

Ganglia

Ganglia is a scalable distributed monitoring system for high-performance computing systems (clusters and grids). It is a BSD-licensed open-source project that grew out of the University of California, Berkeley Millennium Project which was initially funded in large part by the National Partnership for Advanced Computational Infrastructure (NPACI) and National Science Foundation RI Award EIA-9802069 [217].

It relies on a multicast-based listen/announce protocol to monitor state within clusters. It uses a tree of point-to-point connections amongst representative cluster nodes to unite clusters and aggregate their state [218]. It leverages technologies such as XML for data representation, XDR for compact, portable data transport, and RRDtool for data storage and visualization. The implementation is robust, has been ported to an extensive set of operating systems and processor architectures, and is currently in use on thousands of clusters around the world, handling clusters with 2000 nodes.

Nagios (414)

Nagios is a platform, which provides a set of software for network infrastructure monitoring. It also offers administrative tools to diagnose when failure events happen, and to notify operators when hardware issues are detected. Specifically, illustrates that Nagios is consist of modules including [335]: a core and its dedicated tool for core configuration, extensible plugins and its frontend. Nagios core is designed with scalability in mind. Nagios contains a specification language allowing for building an extensible monitoring systems. Through the Nagios API components can integrate with the Nagios core services. Plugins can be developed via static languages like C or script languages. This mechanism empowers Nagios to monitor a large set of various scenarios yet being very flexible. [318] Besides its open source components, Nagios also has commercial products to serve needing clients.

Inca

Inca is a grid monitoring [379] software suite. It provides grid monitoring features. These monitoring features provide operators failure trends, debugging support, email notifications, environmental issues etc. [304]. It enables users to automate the tests which can be executed on a periodic basis. Tests can be added and configured as and when needed. It helps users with different portfolios like system administrators, grid operators, end users etc Inca provides user-level grid monitoring. For each user it stores results as well as allows users to deploy new tests as well as share the results with other users. The incat web ui allows users to view the status of test, manage test and results. The architectural blocks of inca include report repository, agent, data consumers and depot. Reporter is an executable program which is used to collect the data from grid source. Reporters can be written in perl and python. Inca repository is a collection of pre build reporters. These can be accessed using a web url. Inca repository has 150+ reporters available. Reporters are versioned and allow automatic updates. Inca agent does the configuration management. Agent can be managed using the incat web ui. Inca depot provides storage and archival of reports. Depot uses relational database for this purpose. The database is accessed using hibernate backend. Inca web UI or incat provides real time as well as historical view of inca data. All communication between inca components is secured using SSL certificates. It requires user credentials for any access to the system. Credentials are created at the time of the setup and installation. Inca's performance has been phenomenal in production deployments. Some of the deployments are running for more than a decade and has been very stable. Overall Inca provides a solid monitoring system which not only monitors but also detects problems very early on.

39.20.2 Security and Privacy

InCommon

The mission of InCommon is to “create and support a common trust framework for U.S. education and research. This includes trustworthy shared management of access to on-line resources in support of education and research in the United States.” [305] This mission ultimately is a simplification and an elimination of the need for multiple accounts across various websites that are at risk of data spills or misuse. In the academic setting, this helps assist researchers to focus on their area of study, and enabling the cross collaboration which is happening on a global scale. Currently any two and four year higher education institution that is accredited is eligible for joining InCommon.

Eduroam (169)

Eduroam is an initiative started in the year 2003 when the number of personal computers with in the academia are growing rapidly. The goal is to solve the problem of secure access to WI-FI due to increasing number of students and research teams becoming mobile which was increasing the administrative problems for provide access to WI-FI. Eduroam provides any user from an eduroam participating site to get network access at any institution connected through eduroam. According to the orgnizatio in it uses a combination of radius-based infrastructure with 802.1X standard technology to provide roaming access across research and educational networks. The role of the RADIUS hierarchy is to forward user credentials to the users home institution where they can be verified. This proved to be a successful solution when compared to other traditional ways like using MAC-adress, SSID, WEP, 802.1x(EAP-TLS, EAP-TTLS), VPN Clients, Mobile-IP etc which have their own short comings when used for this purpose [195]. Today by enabling eduroam users get access to internet across 70 countries and tens of thousands of access points worldwide.

OpenStack Keystone

[350] Keystone is the identity service used by OpenStack for authentication (authN) and high-level authorization (authZ). There are two authentication mechanisms in Keystone, UUID token, and PKI. Universally unique identifier (UUID) is a 128-bit number used to identify information (user). Each application after each request of the client checks token validity online. PKI was introduced later and improved the security of Keystone [146]. In PKI, each token has its own digital signature that can be checked by any service and OpenStack application with no necessity to ask for Keystone database [127].

Thus, Keystone enables ensuring user's identity with no need to transmit its password to applications. It has recently been rearchitected to allow for expansion to support proxying external services and AuthN/AuthZ mechanisms such as oAuth, SAML and openID in future versions [451].

LDAP

LDAP stands for Lightweight Directory Access Protocol. It is a software protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network, whether on the Internet or on corporate internet. [587]

LDAP is a lightweight (smaller amount of code) version of Directory Access Protocol (DAP), which is part of X.500, a standard for directory services in a network. In a network, a directory tells you where in the network something is located. On TCP/IP networks (including the Internet), the domain name system (DNS) is the directory system used to relate the domain name to a specific network address (a unique location on the network). However, you may not know the domain name. LDAP allows you to search for an individual without knowing where they're located (although additional information will help with the search).An LDAP directory can be distributed among many servers. Each server can have a replicated version of the total directory that is synchronized periodically. An LDAP server is called a Directory System Agent (DSA). An LDAP server that receives a request from a user takes responsibility for the request, passing it to other DSAs as necessary, but ensuring a single coordinated response for the user.

Sentry

“Apache Sentry [63] is a granular, role-based authorization module for Hadoop. Sentry provides the ability to control and enforce precise levels of privileges on data for authenticated users and applications on a Hadoop cluster. Sentry currently works out of the box with Apache Hive, Hive

Metastore/HCatalog, Apache Solr, Impala and HDFS (limited to Hive table data). Sentry is designed to be a pluggable authorization engine for Hadoop components. It allows the client to define authorization rules to validate a user or application's access requests for Hadoop resources. Sentry is highly modular and can support authorization for a wide variety of data models in Hadoop.”

Sqrrl

OpenID

OpenID is an authentication protocol that allows users to log in to different websites, which are not related, using the same login credentials for each, i.e. without having to create separate id and password for all the websites. The login credentials used are of the existing account. The password is known only to the identity provider and nobody else which relieves the users' concern about identity being known to an insecure website. [445] It provides a mechanism that makes the users control the information that can be shared among multiple websites. OpenID is being adopted all over the web. Most of the leading organizations including Microsoft, Facebook, Google, etc. are accepting the OpenIDs [446]. It is an open source and not owned by anyone. Anyone can use OpenID or be an OpenID provider and there is no need for an individual to be approved.

SAML OAuth

As explained in [527], Security Assertion Markup Language (SAML) is a secured XML based communication mechanism for communicating identities between organizations. The primary use case of SAML is Internet SSO. It eliminates the need to maintain multiple authentication credentials in multiple locations. This enhances security by elimination opportunities for identity theft/Phishing. It increases application access by eliminating barriers to usage. It reduces administration time and cost by excluding the effort to maintain duplicate credentials and helpdesk calls to reset forgotten passwords. Three entities of SAML are the users, Identity Provider (IdP-Organization that maintains a directory of users and an authentication mechanism) and Service Provider(SP-Hosts the application /service). User tries to access the application by clicking on a link or through an URL on the internet. The Federated identity software running in the IdP validates the user's identity and the user is then authenticated. A specifically formatted message is then communicated to the federated identity software running at SP. SP creates a session for the user in the target application and allows the user to get direct access once it receives the authorization message from a known identity provider.

39.20.3 Distributed Coordination

Google Chubby

Chubby Distributed lock service [243] is intended for use within a loosely-coupled distributed system consisting of moderately large numbers of small machines connected by a high-speed network. Asynchronous consensus is solved by the Paxos protocol. The implementation in Chubby is based on coarse grained lock server and a library that the client applications link against. As per the 2016 paper [16], an open-source implementation of the Google Chubby lock service was provided by the Apache ZooKeeper project. ZooKeeper used a Paxos-variant protocol Zab for solving the distributed consensus problem. Google stack and Facebook stack both use versions of zookeeper.

Zookeeper

Zookeeper provides coordination services to distributed applications. It includes synchronization, configuration management and naming services among others. The interfaces are available in Java and C [735]. The services themselves can be distributed across multiple Zookeeper servers to avoid single point of failure. If the leader fails to answer, the clients can fall-back to other nodes. The state of the cluster is maintained in an in-memory image along with a persistent storage file called znode by each server. The cluster namespace is maintained in a hierarchical order. The changes to the data are totally ordered [736] by stamping each update with a number. Clients can also set a watch on a znode to be notified of any change [737]. The performance of the ZooKeeper is optimum for read-dominant workloads. It's maintained by Apache and is open-source.

Giraffe

Giraffe is a scalable distributed coordination service. Distributed coordination is a media access technique used in distributed systems to perform functions like providing group membership, gaining lock over resources, publishing, subscribing, granting ownership and synchronization together among multiple servers without issues. Giraffe was proposed as alternative to coordinating services like Zookeeper and Chubby which were efficient only in read-intensive scenario and small ensembles. To overcome this three important aspects were included in the design of Giraffe [541]. First feature is Giraffe uses interior-node joint trees to organize coordination servers for better scalability. Second, Giraffe uses Paxos protocol for better consistency and to provide more fault-tolerance. Finally, Giraffe also facilitates hierarchical data organization and in-memory storage for high throughput and low latency.

JGroups

39.21 Message and Data Protocols

39.21.1 MQTT

39.21.2 Avro

Apache Avro is a data serialization system, which provides rich data structures, remote procedure call(RPC), a container file to store persistent data and simple integration with dynamic languages [40]. Avro depends on schemas, which are defined with JSON. This facilitates implementation in other languages that have the JSON libraries. The key advantages of Avro are schema evolution - Avro will handle the missing/extra/modified fields, dynamic typing - serialization and deserialization without code generation, untagged data - data encoding and faster data processing by allowing data to be written without overhead.

39.21.3 Thrift

The Apache Thrift software framework, for scalable cross-language services development, combines a software stack with a code generation engine to build services that work efficiently and seamlessly between C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Node.js, Smalltalk, OCaml and Delphi and other languages. [551] It includes a complete stack for creating clients and servers. It includes a server infrastructure to tie the protocols and transports together. There are blocking, non-blocking, single and multithreaded servers available. Thrift was originally developed at Facebook, it was open sourced in April 2007 and entered the Apache Incubator in May, 2008. It became an Apache TLP in October, 2010. [612]

39.21.4 Protobuf

Protocol Buffer [491] is a way to serialize structured data into binary form (stream of bytes) in order to transfer it over wires or for storage. It is used for inter application communication or for remote procedure call (RPC). It involves a interface description that describes the structure of some data and a program that can generate source code or parse it back to the binary form. It emphasizes on simplicity and performance over xml. Though xml is more readable but requires more resources in parsing and storing. This is developed by Google and available under open source licensing. The parser program is available in many languages including java and python.

39.22 Technologies To Be Integrated

39.22.1 Snort

[700] Snort is a Network Intrusion Prevention System (NIPS) and Network Intrusion Detection System (NIDS). Snort's open source network-based intrusion detection system (NIDS) has the ability to perform real-time traffic analysis and packet logging on Internet Protocol (IP) networks. Snort performs protocol analysis, content searching and matching. These basic services have many purposes including application-aware triggered quality of service, to de-prioritize bulk traffic when latency-sensitive applications are in use. The program can also be used to detect probes or attacks, including, but not limited to, operating system fingerprinting attempts, common gateway interface, buffer overflows, server message block probes, and stealth port scans. Snort can be configured in three main modes: sniffer, packet logger, and network intrusion detection. In sniffer mode, the program will read network packets and display them on the console. In packet logger mode, the program will log packets to the disk. In intrusion detection mode, the program will monitor network traffic and analyze it against a rule set defined by the user. The program will then perform a specific action based on what has been identified.

39.22.2 Fiddler

Fiddler is an HTTP debugging proxy server application. Fiddler captures HTTP and HTTPS traffic and logs it for the user to review by implementing man-in-the-middle interception using self-signed certificates. Fiddler can also be used to modify (fiddle with) HTTP traffic for troubleshooting purposes as it is being sent or received.[5] By default, traffic from Microsoft's WinINET HTTP(S) stack is automatically directed to the proxy at runtime, but any browser or Web application (and most mobile devices) can be configured to route its traffic through Fiddler [691].

39.22.3 Zeppelin

Apache Zeppelin [732] provides an interactive environment for big data data analytics on applications using distributed data processing systems like Hadoop and Spark. It supports various tasks like data ingestion, data discovery, data visualization, data analytics and collaboration. Apache Zeppelin provides built-in Apache Spark integration and is compatible with many languages/data-processing backends like Python, R, SQL, Cassandra and JDBC. It also supports adding new language backend. Zeppelin also lets users to collaborate by sharing their Notebooks, Paragraph and has option to broadcast any changes in realtime.

39.22.4 Open MPI

The Open MPI Project [604] is an open source Message Passing Interface implementation that is developed and maintained by a consortium of academic, research, and industry partners. Open MPI is therefore able to combine the expertise, technologies, and resources from all across the High Performance Computing community in order to build the best MPI library available. Open MPI offers advantages for system and software vendors, application developers and computer science researchers. Open MPI [211] provides functionality that has not previously been available in any single, production-quality MPI implementation, including support for all of MPI-2, multiple concurrent user threads, and multiple options for handling process and network failures.

39.22.5 Apache Tomcat

Apache tomcat is an open source java servlet container. [457] It is used in IT industry as a HTTP web server which listens to the requests made by web client and send responses. The main components of tomcat are cataline, coyote and jasper. The most stable version of Apache Tomcat server is version 8.5.11. Apache tomcat is released under Apache License version 2. [671] As it is cross platform, it can run in any platform or OS like Windows, UNIX, AIX or SOLARIS etc. It is basically an integral part of many java based web application.

39.22.6 Apache Beam

Apache Beam attempts to abstract away the need to write code for multiple data-oriented workflows, e.g., batch, interactive and streaming, as well as multiple big data tools, e.g., Storm, Spark and Flink. Instead, Beam attempts to automatically map a dataflow process written in Java or Python to the target runtime environment via *runners*. As a result, switching a data processing routine from Spark to Flink only requires changing the target runtime environment as opposed to re-writing the entire process [480] (perhaps in a completely different language). Google contributed its Dataflow SDK, the Dataflow model and three runners [707] to the Apache Software Foundation in the first half of 2016. The ASF elevated Beam to a Top-Level project in January 2017. Jean-Baptiste Onofre of French tech company Talend, and a frequent Apache project contributor, champions the project. [438] It should be grouped with the technologies in the *Interoperability* section.

39.22.7 Cloudability

Cloudability is a financial management tool for analyzing and monitoring all cloud expenses across an organization. It can be used for cost monitoring, usage rightsizing, reserved instance planning, cost allocation, role-based visibility. It aggregates expenditures into reports, helps identify opportunities for reducing costs, offers budget alerts and recommendations via SMS and email, and provides APIs for connecting cloud billing and usage data to any business or financial system. [124]

39.22.8 CUDA

It is a parallel computing platform and application programming interface(API) model created by Nvidia. It allows software developers to use a CUDA-enabled graphics processing unit for general purpose processing. The CUDA platform is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements, for the execution of compute kernels. CUDA platform has advantages such as scattered reads i.e the code can read from arbitrary addresses in memory, unified virtual memory, unified memory, faster downloads and readbacks to

and from the GPU and full support for integer and bitwise operations. [687]. CUDA is used for accelerated rendering of 3D graphics, accelerated interconversion of video file formats, encryption, decryption and compression of files. It is also used for distributed calculations, face recognition and distributed computing. [687]

39.22.9 Blaze

Blaze library translates NumPy/Pandas-like syntax to data computing systems (e.g. database , in-memory, distributed-computing). This provides Python users with a familiar interface to query data in a variety of other data storage systems. One Blaze query can work across data ranging from a CSV file to a distributed database.

Blaze presents a pleasant and familiar interface regardless of what computational solution or database we use (e.g. Spark, Impala, SQL databases, No-SQL data-stores, raw-files). It mediates the users interaction with files, data structures, and databases, optimizing and translating the query as appropriate to provide a smooth and interactive session. It allows the data scientists and analyst to write their queries in a unified way that does not have to change because the data is stored in another format or a different data-store. [134]

39.22.10 CDAP

CDAP [709] stands for Cask Data Application Platform. CDAP is an application development platform using which developers can build, deploy and monitor applications on Apache Hadoop. In a typical CDAP application, a developer can ingest data, store and manage datasets on Hadoop, perform batch mode data analysis, and develop web services to expose the data. They can also schedule and monitor the execution of the application. This way, CDAP enables the developers to use single platform to develop the end to end application on Apache Hadoop.

CDAP documentation [710] explains the important CDAP concepts of CDAP Dataset, CDAP Application and CDAP Services. CDAP Datasets provide logical abstraction over the data stored in Hadoop. CDAP Applications provide containers to implement application business logic in open source processing frameworks like map reduce, Spark and real time flow. CDAP applications also provide standardize way to deploy and manage the apps. CDAP Services provide services for application management, metadata management, and streams management. CDAP can be deployed on various Hadoop Platforms such as Apache Hadoop, Cloudera Hadoop, Hortonworks Hadoop and Amazon EMR. CDAP sample apps [229] provide explain how to implement apps on CDAP platform.

39.22.11 Apache Arrow

Apache arrow allows execution engines to utilize what is known as Single Input multiple data (SIMD). [36] This SIMD is an operation that allows modern processors to take advantage of this engine. Performance is enhanced by grouping relevant data as close as possible in a column format. Many programming languages are supported such a Java, C, C++, Python and it is anticipated that languages will be added as it grows. It is still in early development but has released a 0.1.0 build.

39.22.12 OpenRefine

OpenRefine (formerly GoogleRefine) is an open source tool that is dedicated to cleaning messy data. With the help of this user-friendly tool you can explore huge data sets easily and quickly even

if the data is a little unstructured. It allows you to load data, understand it, clean it up, reconcile it, and augment it with data coming from the web [448]. It operates on rows of data which have cells under columns, which is very similar to relational database tables. One OpenRefine project is one table. The user can filter the rows to display using facets that define filtering criteria. most operations in OpenRefine are done on all visible rows: transformation of all cells in all rows under one column, creation of a new column based on existing column data, etc. All actions that were done on a dataset are stored in a project and can be replayed on another dataset. It has a huge community with lots of contributors meaning that the software is constantly getting better and better.

39.22.13 Apache OODT

Apache Object Oriented Data Technology (OODT) [439] is a distributed data management technology that helps to integrate and archive your processes, your data, and its metadata. OODT allows to generate, process, manage and analyze distributed and heterogeneous data enabling integration of different, distributed software systems. Apache OODT uses structured XML-based capturing of the processing pipeline which is used to create, edit, manage and provision workflow and task execution. OODT is written in Java programming language and provides its own set of APIs for storing and processing data. [440] It provides three core services. A File Manager is responsible for tracking file locations, their metadata, and for transferring files from a staging area to controlled access storage. A Workflow Manager captures control flow and data flow for complex processes, and allows for reproducibility and the construction of scientific pipelines. A Resource Manager handles allocation of workflow tasks and other jobs to underlying resources, e.g., Python jobs go to nodes with Python installed on them similarly jobs that require a large disk or CPU are properly sent to those nodes that fulfill those requirements. OODT is now supported with Apache Mesos and Grid Computing which can allow for creating of highly distributed, scalable data platforms that can process large amounts of data. OODT technology is used in NASA's Jet Propulsion Labatory.

39.22.14 Omid

Omid is a “flexible, reliable, high performant and scalable ACID transactional framework” [53] for NoSQL databases, developed by Yahoo for HBase and contributed to the Apache community. Most NoSQL databases, do not natively support ACID transactions. Omid employs a lock free approach from concurrency and can scale beyond 100,000 transactions per second. At Yahoo, millions of transactions per day are processed by Omid. [468].

Omid is currently in the Apache Incubator. All projects accepted by the Apache Software Foundation (ASF) undergo an incubation period until a review indicates that the project meets the standards of other ASF projects [51]

39.22.15 Apache Ant

Apache Ant is a Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other. The main known usage of Ant is the build of Java applications. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications. Ant can also be used effectively to build non Java applications, for instance C or C++ applications. More generally, Ant can be used to pilot any type of process which can be described in terms of targets and tasks. Ant is written in Java. Users of Ant can develop their own “antlibs” containing Ant tasks and types, and are offered a large number of ready-made commercial or open-source “antlibs”. Ant is extremely flexible and does

not impose coding conventions or directory layouts to the Java projects which adopt it as a build tool. Software development projects looking for a solution combining build tool and dependency management can use Ant in combination with Apache Ivy. The Apache Ant project is part of the Apache Software Foundation [592].

39.22.16 LXD

LXD is a demon processes established to manage the containers. It can be understood as hypervisor for linux containers. It is implemented by exporting RESTful API for libxl to the remote network or local unix socket. [633]. It implements the under privilized containers by default adding more security. It works with Image based work flow supports online snapshopping and live container migration. [382].It was build with aim of providing VM like virtualization with container like performance. [603]

39.22.17 Wink

Apache wink [60] provides a framework to develop and use RESTful web services. It implements using JAX-RS v1.1 specification. The project provides server module which integrates with all popular web servers and a client module which can used to write RESTful web services. This project will be integrated with Geronimo and other opensource REST projects to build a vendor neutral community. Currently IBM and HP have taken lead. IBM is writing a full JAX-RS implementation while HP is working on RESTful SDK for client and server components. Portion of initial project was also taken from Apache CXF which uses other Apache components like commons-codec, commons-logging, Apache-Abdera. Apache wink will simply web services development using one single standard.

39.22.18 Apache Apex

Apache Apex is “a YARN(Hadoop 2.0)-native platform that unifies cloud and batch processing” [665].This project was developed under Apache License 2.0 and was driven by Data Torrent. It can be used for processing both streams of data and static files making it more relevant in the context of present day internet and social media. It is aimed at leveraging the present Hadoop platform and reducing the learning curve for development of applications over it. It is aimed at It can used through a simple API. It enables reuse of code by not having to make drastic changes to the applications by providing interoperability with existing technology stack. It leverages the existing Hadoop platform investments.

Apart from the Apex core component, it also has Apex Malhar which provides a library of connectors and logic functions. It provides connectors to existing file systems, message systems and relational, NoSQL and Hadoop databases, social media. It also provides a library of compute operators like Machine Learning, Stats and Math, Pattern Matching, Query and Scripting, Stream manipulators, Parsers and UI & Charting operators [345].

39.22.19 Apache Knox

According to [44], “the Apache Knox Gateway is a REST API Gateway for interacting with Apache Hadoop clusters.” REST stands for Representational State Transfer and is web architectural style designed for distributed hypermedia systems and defines a set of constraints. [188] API Gateways manage concerns related to “Authentication, Transport Security, Load-balancing, Request Dispatching (including fault tolerance and service discovery), Dependency Resolution, Transport

Transformations.” [469] Although every Apache Hadoop cluster has its own set of REST APIs, Knox will represent all of them as “a single cluster specific application context path.” [44] Knox protects Apache Hadoop clusters, by way of its gateway function, by aiding “the control, integration, monitoring and automation of critical administrative and analytical needs.” [44] Some Apache Hadoop Services that integrate with Knox are, “Ambari, WebHDFS (HDFS), Templeton (Hcatalog), Stargate (Hbase), Oozie, Hive/JDBC, Yarn RM, [and] Storm.” [44] Apache Knox has a configuration driven method to aid in the addition of new routing services. [44] This allows support for new and custom Apache Hadoop REST APIs to be added to the Knox gateway quickly and easily. [44] This technology would be best placed under the interoperability category.

39.22.20 Apache Apex

The Apex platform is designed to process real-time events with streaming data natively in Hadoop. The platform handles application execution, dynamic scaling, state checkpointing and recovery, etc. This allows the users to focus on writing their application logic without mixing operational and functional concerns [38]. In the platform, building a streaming application is easy and intuitive.

An application may consist of one or more operators each of which define some logical operation to be done on the tuples arriving at the operator. These operators are connected together to form streams. A streaming application is represented by a DAG that consists of operators and streams [39]. The Apex platform comes with support for web services and metrics. This enables ease of use and easy integration with current data pipeline components. DevOps teams can monitor data in action using existing systems and dashboards with minimal changes, thereby easily integrating with the current setup. With different connectors and the ease of adding more connectors, Apex easily integrates with an existing dataflow [114].

39.22.21 Robot Operating System (ROS)

The aptly-named *Robot Operating System*, or ROS, provides a framework for writing operating systems for robots. ROS offers “a collection of tools, libraries, and conventions [meant to] simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms” [444]. ROS’ designers, the Open Source Robotics Foundation, hereinafter OSRF or the Foundation, attempt to meet the aforementioned objective by implementing ROS as a modular system. That is, ROS offers a core set of features, such as inter-process communication, that work with or without pre-existing, self-contained components for other tasks.

The OSRF designed ROS as a distributed, modular system. The OSRF maintains a subset of essential features for ROS, i.e., *ROS core*, to provide an extensible platform for other roboticists. The Foundation also coordinates the maintenance and distribution of a vast array of ROS add-ons, referred to as modules. ROS’ core consists of the following components: a) communications infrastructure; b) robot-specific features; and, c) tools. The modules, analogous to packages in Linux repositories or libraries in other software packages such as *R*, provide solutions for numerous robot-related problems. General categories include a) drivers, such as sensor and actuator interfaces; b) platforms, for steering and image processing, etc.; c) algorithms, for task planning and obstacle avoidance; and, d) user interfaces, such as tele-operation and sensor data display [417].

39.22.22 Apache Flex

Apache Flex [193] is an open source application framework for building and maintaining mobile and web applications that deploy consistently on multiple browsers, desktops and mobile devices.

It was initially developed by Macromedia and then acquired by Adobe Systems. It was later donated to the Apache Software Foundation in 2011 [194]. It can pull data from multiple back-end sources such as Java, Spring, PHP, Ruby, .NET, Adobe ColdFusion, and SAP and display it visually allowing users to drill down into the data for deeper insight and even change the data and have it automatically updated on the back end [661].

39.22.23 Apache Ranger

Apache Ranger [708] is open source software project designed to provide centralized security services to various components of Apache Hadoop. Apache Hadoop provides various mechanism to store, process and access the data. Each Apache tool has its own security mechanism. This increases administrative overhead and is also error prone. Apache Ranger fills this gap to provide a central security and auditing mechanism for various Hadoop components [721]. Using Ranger, Hadoop administrators can perform security administration tasks using a central UI or Restful web services. He can define policies which enable users/user-groups to perform specific action using Hadoop components and tools. Ranger provides role based access control for datasets on Hadoop at column and row level. The blog article [722] explains that the row level filtering and dynamic data masking are most important features of Apache Ranger. Ranger also provides centralized auditing of user acces and security related administrative actions.

39.22.24 Google Cloud Machine Learning

Google Could Machi<ne Leaning is a Googles cloud based managed system for building machine learning model, capable to work on any type and volume of data. User can create their own machine learning model using GoogleTensorFlow framework, which helps to use the range of Google products from Google Photos to Google Cloud Speech. We can build our machine learning model regardless the size, google will managed it infrastructure according to requirement. User can immediately host the created model and start predicting on new data [249].Cloud Machine Learning provides two important things:

- Help user to train the machine learning model at large scale with the help of TensorFlow training application.
- User can host the trained model on cloud, this will help to use the large and new data available on cloud, which help in creating good model.

Google CloudML will help user to focus on model instead of hardware configuration and resource management [250].

39.22.25 Karajan

Karajan is used to allow users to describe various workflows using XML [483]. It also uses a custom yet user friendly language called K. The advantages of using XML and K is that we can use Directed Acyclic Graphs (DAGs) to describe hierarchical workflows. Besides, it is also very easy to handle concurrency using trivial programming constructs like if/while orders. It can also use tools such as Globus GRAM for parallel or distributed execution of various workflows. From an architectural perspective, Karajan mainly consists of three components: Workflow engine, that monitors the execution and is responsible for the higher level interaction with higher level components like the Graphical User Interface Module (GUI) for the description of various workflows; Workflow service, that is used to allow the execution of various workflows using specific functionalities that can be accessed by the workflow engine using specific libraries; and the Checkpointing subsystem that

monitors and checks the current state of the workflow. Karajan is typically used as a scientific workflow scheduling technique for various Big Data platforms.

The Karajan code, that can be obtained from Java CoG Kit CVS archive has two interfaces: the command line interface (CLI) and the GUI. The CLI can be accessed via bin/karajan and provides a minimalist interface that is non-interactive and doesn't provide much feedback on the execution status. As against this, the GUI can be accessed via bin/karajan-gui and provides an enriched interface that provides visual features to determine the execution status besides being interactive in real time [368].

39.23 Exercise

TechList.1: In class you will be given an HID and you will be assigned a number of technologies that you need to research and create a summary as well as one or more relevant references to be added to the Web page. All technologies for TechList.1 are marked with a (1) behind the technology. An example text is given for Nagios in this page. Please create a pull request with your responses. You are responsible for making sure the request shows up and each commit is using gitchangelog in the commit message::

new:usr: added paragraph about <PUTTECHHERE>

You can create one or more pull requests for the technology and the references. We have created in the referens file a placeholder using your HID to simplify the management of the references while avoiding conflicts. For the technologies you are responsible to investigate them and write an academic summary of the technology. Make sure to add your reference to refs.bib. Many technologies may have additional references than the Web page. Please add the most important once while limiting it to three if you can. Avoid plagiarism and use proper quotations or better rewrite the text.

You must look at :doc:‘technologies-hw’ to successfully complete the homework

A video about this homework is posted at <https://www.youtube.com/watch?v=roi7vezNmfo> showing how to do references in emacs and jabref, it shows you how to configure git, it shows you how to do the fork request while asking you to add “new:usr” to the commit messages). As this is a homework related video we put a lot of information in it that is not only useful for beginners. We recommend you watch it.

This homework can be done in steps. First you can collect all the content in an editor. Second you can create a fork. Third you can add the new content to the fork. Fourth you can commit. Fifth you can push. Sixth if the TAs have command improve. The commit message must have new:usr: at the beginning.

While the Nagios entry is a good example (make sure grammar is ok the Google app engine is an example for a bad entry).

Do Techlist 1.a 1.b 1.c first. We will assign Techlist 1.d and TechList 2 in February.

TechList.1.a: Complete the pull request with the technologies assigned to you. Details for the assignment are posted in Piazza. Search for TechList.

TechList.1.b: Identify how to cite. We are using *scientific* citation formats such as IEEEtran, and ACM. We are **not** using citation formats such as Chicago, MLA, or ALP. The later are all for non scientific publications and thus of no use to us. Also when writing about a technology do not use the names of the person, simply say something like. In [1] the definition of a turing machine is given as follows, ... and do not use elaborate sentences such as: In his groundbreaking work conducted in England, Allan Turing, introduced the turing machine in the years 1936-37 [2]. Its definition is based on ... The difference is clear, while the first focusses on results and technological concepts, the second introduces a colorful description that is more suitable for a magazine or a computer history paper.

TechList 1.c: Learn about plagiarism and how to avoid it. Many Web pages will conduct self advertisement while adding suspicious and subjective adjectives or phrases such as cheaper, superior, best, most important, with no equal, and others that you may not want to copy into your descriptions. Please focus on facts, not on what the author of the Web page claims.

TechList 1.d: Identify technologies from the Apache Project <https://projects.apache.org/> or other Big Data related Web pages and projects that are not yet listed here. Add them at the end of the Technologies page under the :ref:`New Technologies <new-techs>` section, together with a description and appropriate references just like you did for your list of technologies in TechList 1a-1c. As part of your paragraph, please suggest a section where you think is best to add the technologies. Once the new technologies have been submitted, the AIs will integrate them in the appropriate sections. Please, only add new techs to the last section, otherwise it will be easy to introduce conflicts in the file.

TechList.2: In this hopweork we provide you with additional technologies that you need to complete. They are marked with (2) in the :doc:`HID Assignment page <hids-techs>`.

TechList.3: Identify technologies that are not listed here and add them. Provide a description and a reference just as you did before. Before you add a technology, verify that it is not on the **new technologies** list already. Duplicated entries will be merged.

TechList.4: For useful information on how to correctly create BibTeX entries, see and contribute to :ref:`these open discussion threads Piazza <bibtex-discussions>`.

Container

40	REST	527
40.1	Swagger	
40.2	FlaskRESTful	
40.3	Django REST Framework	
40.4	Eve	
41	Container	533
41.1	Kubernetes	
42	Run Docker Locally on your Machine	535
42.1	Installing Docker Community Edition	
42.2	Testing if the install works	
43	Running Docker on FutureSystems ..	539
43.1	Overview	
43.2	Getting Access	
43.3	Creating a service and deploy to the swarm cluster	
	Bibliography	543
	Index	583



40. REST

F section/container/rest.tex

REST stands for REpresentational State Transfer. REST is an architecture style for designing networked applications. It is based on stateless, client-server, cacheable communications protocol. Although not based on http, in most cases, the HTTP protocol is used. In contrast to what some others write or say, REST is not a *standard*. RESTful applications use HTTP requests to (a) post data while creating and/or updating it, (b) read data while making queries, and (c) delete data.

<https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>

Hence REST uses HTTP for the four CRUD operations:

- Create
- Read
- Update
- Delete

As part of the HTTP protocol we have methods such as GET, PUT, POST, and DELETE. These methods can than be used to implement a REST service. As REST introduces collections and items we need to implement the CRUD functions for them. The semantics is explained in the Table illustrating how to implement them with HTTP methods.

http://.../resources/

GET List the URIs and perhaps other details of the collection's members

PUT Replace the entire collection with another collection.

POST Create a new entry in the collection. The new entry's URI is assigned automatically and is usually returned by the operation.

DELETE Delete the entire collection.

http://.../resources/item17

- GET** Retrieve a representation of the addressed member of the collection, expressed in an appropriate Internet media type.
- PUT** Replace the addressed member of the collection, or if it does not exist, create it.
- POST** Not generally used. Treat the addressed member as a collection in its own right and create a new entry within it
- DELETE** Delete the addressed member of the collection.

Source: https://en.wikipedia.org/wiki/Representational_state_transfer

Due to the structure that REST provides a number of tools have been created that manage the creation of the specification for rest services and their programming. We distinguish several different categories:

- (1) REST programming language support: the tools and services are targeting a particular programming language.
- (3) REST documentation based tools that are primarily oriented towards documenting REST specifications.
- (2) REST design support: When abstracting the programming language out define reusable specifications that can be used to create clients and servers for particular technology targets.

40.1 Swagger

Swagger <https://swagger.io/> is a tool for developing API specifications based on the OpenAPI Specification (OAS). It allows not only the specification, but the generation of code based on the specification in a variety of languages.

Swagger itself has a number of tools which together

40.1.1 Tools

The major Swagger tools of interest are:

Swagger Core includes Java libraries for working with Swagger specifications <https://github.com/swagger-api/swagger-core>.

Swagger Codegen allows to generate code from the specifications to develop Client SDKs, servers, and documentation. <https://github.com/swagger-api/swagger-codegen>

Swagger UI is an HTML5 based UI for exploring and interacting with the specified APIs <https://github.com/swagger-api/swagger-ui>

Swagger Editor is a Web-browser based editor for composing specifications using YAML <https://github.com/swagger-api/swagger-editor>

The developed APIs can be hosted and further developed on an online repository named Swagger-Hub <https://app.swaggerhub.com/home>. The convenient online editor is available which also can be installed locally on a variety of operating systems including OSX, Linux, and Windows.

40.2 FlaskRESTful

“Flask-RESTful integrates a framework for creating REST APIs within Flask. The abstraction provided by it works with your existing ORM/libraries. Through its minimalistic approach It

encourages best practices with minimal setup. If you are familiar with Flask, Flask-RESTful should be easy to pick up.” <https://flask-restful.readthedocs.io/en/latest/>

40.3 Django REST Framework

<http://www.django-rest-framework.org/>

40.4 Eve

<http://python-eve.org/>

Eve makes the creation of a REST implementation in python easy. We will provide you with an implementation example that showcases that we can create REST services without writing a single line of code. The code for this is located at <https://github.com/cloudmesh/rest>

This code will have a master branch but will also have a dev branch in which we will add gradually more objects. Objects in the dev branch will include:

- virtual directories
- virtual clusters
- job sequences
- inventories

; You may want to check our active development work in the dev branch. However for the purpose of this class the master branch will be sufficient.

Installation

First we have to install mongodb. The installation will depend on your operating system. For the use of the rest service it is not important to integrate mongodb into the system upon reboot, which is focus of many online documents. However, for us it is better if we can start and stop the services explicitly for now.

On ubuntu, you need to do the following steps:

TO BE CONTRIBUTED BY THE STUDENTS OF THE CLASS as homework

On windows 10, you need to do the following steps:

TO BE CONTRIBUTED BY THE STUDENTS OF THE CLASS as homework, if you
elect Windows 10. You could be using the online documentation
provided by starting it on Windows, or running it in a docker container.

On OSX you can use homebrew and install it with:

```
brew update  
brew install mongodb
```

In future we may want to add ssl authentication in which case you may need to install it as follows:

```
brew install mongodb --with-openssl
```

Starting the service

We have provided a convenient Makefile that currently only works for OSX. It will be easy for you to adapt it to Linux. Certainly you can look at the targets in the makefile and replicate them one by

one. Improtah targets are deploy and test.

When using the makefile you can start the services with:

```
make deploy
```

IT will start two terminals. IN one you will see the mongo service, in the other you will see the eve service. The eve service will take a file called sample.settings.py that is base on sample.json for the start of the eve service. The mongo servide is configured in suc a wavy that it only accepts incimming connections from the local host which will be sufficient fpr our case. The mongo data is written into the \$USER/.cloudmesh directory, so make sure it exists.

To test the services you can say:

```
make test
```

YOu will se a number of json text been written to the screen.

40.4.1 Creating your own objects

The example demonstrated how easy it is to create a mongodb and an eve rest service. Now lets use this example to creat your own. FOr this we have modified a tool called evegenie to install it onto your system.

The original documentation for evegenie is located at:

- <http://evegenie.readthedocs.io/en/latest/>

However, we have improved evegenie while providing a commandline tool based on it. The improved code is located at:

- <https://github.com/cloudmesh/evegenie>

You clone it and install on your system as follows:

```
cd ~/github
git clone https://github.com/cloudmesh/evegenie
cd evegenie
python setup.py install
pip install .
```

This shoudl install in your system evegenie. YOu can verify this by typing:

```
which evegenie
```

If you see the path evegenie is installed. With evegenie installed its usaage is simple:

```
$ evegenie
```

```
Usage:
evegenie --help
evegenie FILENAME
```

It takes a json file as input and writes out a settings file for the use in eve. Lets assume the file is called sample.json, than the settings file will be called sample.settings.py. Having the evegenie programm will allow us to generate the settings files easily. You can include them into your project and leverage the Makefile targets to start the services in your project. In case you generate new objects, make sure you rerun evegenie, kill all previous windows in whcih you run eve and mongo and restart. In case of changes to objects that you have designed and run previously, you need to also delete the mongod database.

40.4.2 Towards cmd5 extensions to manage eve and mongo

Naturally it is of advantage to have in cms administration commands to manage mongo and eve from cmd instead of targets in the Makefile. Hence, we **propose** that the class develops such an extension. We will create in the repository the extension called admin and hope that students through collaborative work and pull requests complete such an admin command.

The proposed command is located at:

- <https://github.com/cloudmesh/rest/blob/master/cloudmesh/ext/command/admin.py>

It will be up to the class to implement such a command. Please coordinate with each other.

The implementation based on what we provided in the Make file seems straight forward. A great extensinon is to load the objects definitions or eve e.g. settings.py not from the class, but forma place in .cloudmesh. I propose to place the file at:

.cloudmesh/db/settings.py

the location of this file is used whne the Service class is initialized with None. Prior to starting the service the file needs to be copied there. This could be achived with a set commad.



41. Container



section/container/introduction.tex

41.1 Kubernetes

section/container/docker.tex

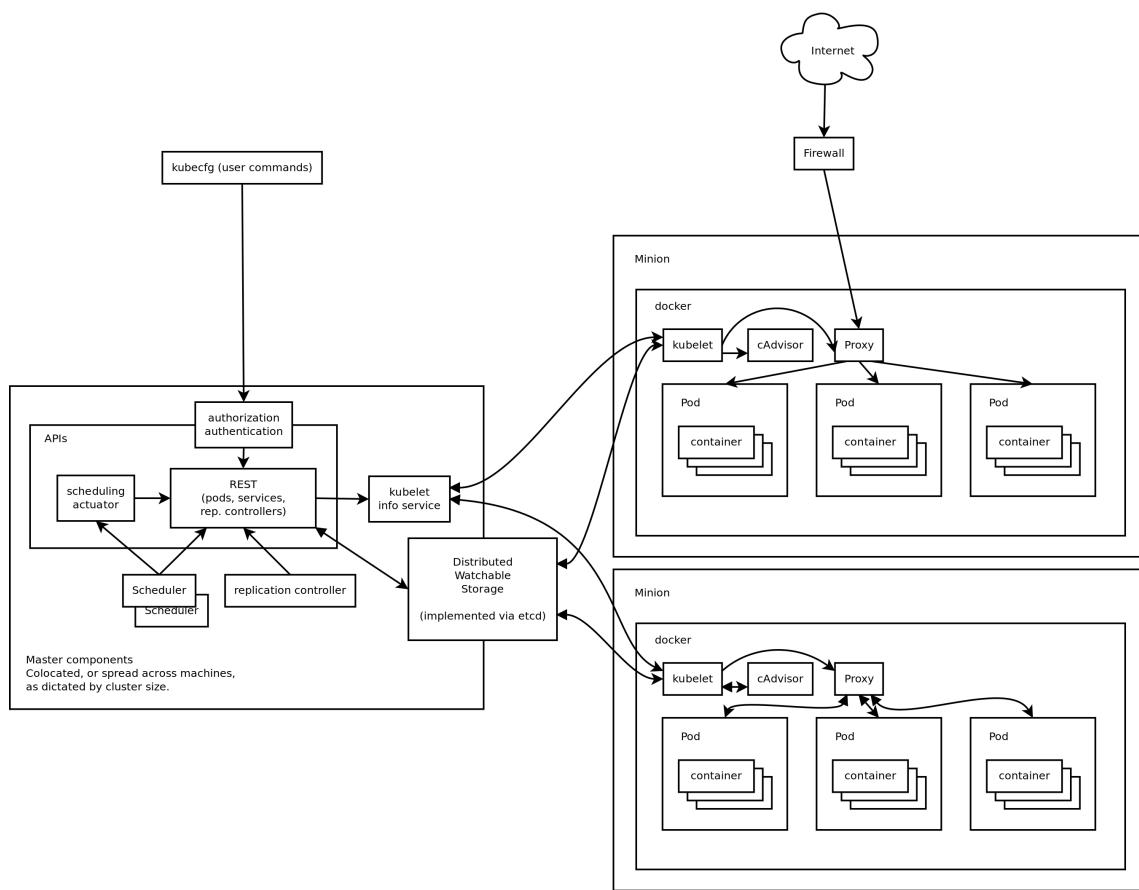


Figure 41.1: Kubernetes (Source: Google)



42. Run Docker Locally on your Machine

F section/container/docker.tex

42.1 Installing Docker Community Edition

To install docker on your computer, please visit the page:

- <https://www.docker.com/community-edition>

Docker Community Edition

Here you will find a variety of packages, one of which will hopefully suitable for your OS. The supported operating systems currently include:

- OSX, Windows, Centos, Debian, Fedora, Ubuntu, AWS, Azure

Please chose the one most suitable for you.

42.1.1 Instalation for OSX

The docker community edition for OSX can be found at the following link

[Information for OSX](#)

We recommend that at this time you get the version *Docker CE for MAC (stable)*

- <https://download.docker.com/mac/stable/Docker.dmg>

CLicking on the link will download a dmg file to your machine, that you than will need to install by double clicking and allowing access to the dmg file. Upon instalation a whale in the top status bar shows that Docker is running, and you can acess it via a terminal.



Figure 42.1: Docker integrated in the menu bar on OSX

42.2 Testing if the install works

To test if it works execute the following commands in a terminal:

```
docker version
```

You should see an output similar to

```
docker version

Client:
  Version:      17.03.1-ce
  API version:  1.27
  Go version:   go1.7.5
  Git commit:   c6d412e
  Built:        Tue Mar 28 00:40:02 2017
  OS/Arch:      darwin/amd64

Server:
  Version:      17.03.1-ce
  API version:  1.27 (minimum version 1.12)
  Go version:   go1.7.5
  Git commit:   c6d412e
  Built:        Fri Mar 24 00:00:50 2017
  OS/Arch:      linux/amd64
  Experimental: true
```

To see if you can run a container use

```
docker run hello-world
```

Once executed you should see an output similar to

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
78445dd45222: Pull complete
Digest: sha256:c5515758d4c5e1e838e9cd307f6c6a .....
Status: Downloaded newer image for hello-world:latest
```

```
Hello from Docker!
This message shows that your installation appears to
be working correctly.
```

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

```
$ docker run -it ubuntu bash
```

Share images, automate workflows, and more with a free Docker ID:
<https://cloud.docker.com/>

For more examples and ideas, visit:
<https://docs.docker.com/engine/userguide/>



43. Running Docker on FutureSystems



section/container/docker-fs.tex

43.1 Overview

This documentation introduces how to run Docker container on FutureSystems. Currently we have deployed Docker swarm on Echo.

43.2 Getting Access

You will need an account on FutureSystems. To verify, try to see if you can log into india.futuresystems.org. You need to be a member of a valid FutureSystems project, and had submitted an ssh public key via the FutureSystems portal.

If your access to the india host has been verified, try to login to the docker swarm head node with the same username and key:

NOTE: If you have access to india but not the docker swarm system, your project may not have been authorized to access the docker swarm cluster. Send a ticket to FutureSystems ticket system to request this.

Once logged in to the docker swarm head node, try to run:
to verify ‘docker run’ works.

43.3 Creating a service and deploy to the swarm cluster

While ‘docker run’ can start a container and you may even attach to its console, the recommended way to use a docker swarm cluster is to create a service and have it run on the swarm cluster. The

service will be scheduled to one or many number of the nodes of the swarm cluster, based on the configuration. It's also easy to scale up the service when more swarm nodes are available. Docker swarm really makes it easier for service/application developers to focus on the functionality development but not worrying about how and where to bind the service to some resources/server. The deployment, access, and scaling up/down when necessary, are all managed transparently. Thus achieving the new paradigm of ‘serverless computing’.

As an example, the following command creates a service and deploy it to the swarm cluster:

```
docker service create --name notebook_test -p 9001:8888 jupyter/datascience-notebook
start-notebook.sh --NotebookApp.password=NOTEBOOK_PASS
```

It pulls a published image from docker cloud, starts a container and runs a script to start the service inside the container with necessary parameters. The option “-p 9001:8888” maps the service port inside the container (8888) to an external port of the cluster node (9001) so the service could be accessed from the Internet. In this example, you can then visit the URL:

<http://149.165.150.76:9001>

to access the Jupyter notebook. Using the specified password when you create the service to login.

Please note the service will be dynamically deployed to a container instance, which would be allocated to a swarm node based on the allocation policy. Docker makes this process transparent to the user and even created mesh routing so you can access the service using the IP address of the management head node of the swarm cluster, no matter which actual physical node the service was deployed to.

This also implies that the external port number used has to be free at the time when the service was created.

Some useful related commands:

```
docker service ls
```

lists the currently running services.

```
docker service ps notebook_test
```

lists the detailed info of the container where the service is running.

```
docker node ps NODE
```

lists all the running containers of a node.

```
docker node ls
```

lists all the nodes in the swarm cluster.

To stop the service and the container:

```
docker service rm notebook_test
```

43.3.1 Create your own service

You can create your own service and run it. To do so, start from a base image, e.g., a ubuntu image from the docker cloud. Then you could:

- Run a container from the image and attach to its console to develop the service, and create a new image from the changed instance using command ‘docker commit’.
- Create a dockerfile, which has the step by step building process of the service, and then build an image from it.

In reality, the first approach is probably useful when you are in the phase of develop and debug your application/service. Once you have the step by step instructions developed the latter approach is the recommended way.

Publish the image to the docker cloud by following this documentation:

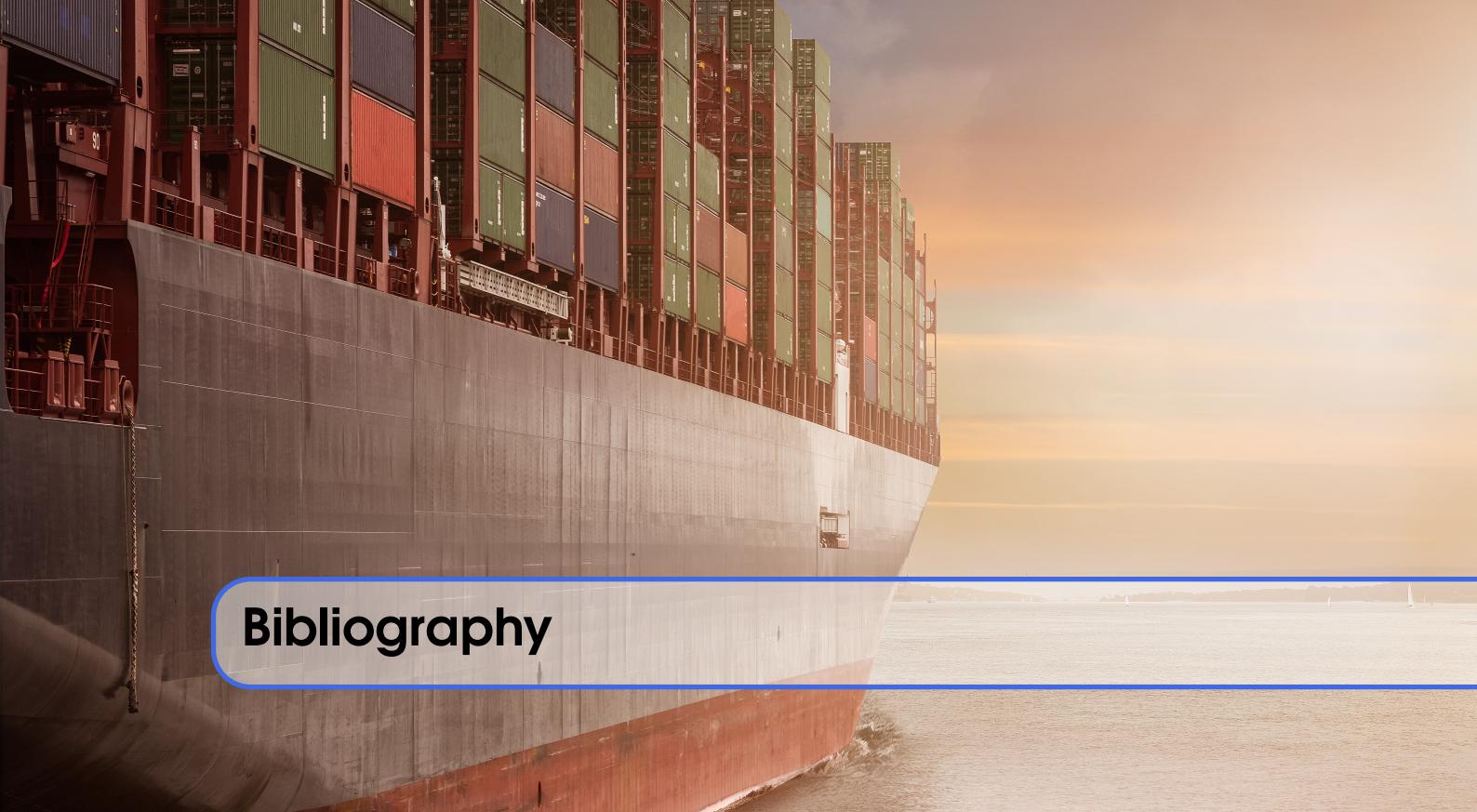
<https://docs.docker.com/docker-cloud/builds/push-images/>

Please make sure no sensitive information is included in the image to be published. Alternatively you could publish the image internally to the swarm cluster.

Publish an image privately within the swarm cluster

TODO: Fugang: create image for distribution

Once the image is published and available to the swarm cluster, you could start a new service from the image similar to the Jupyter Notebook example.



Bibliography

References

- [1] web page. accessed 2017-02-13. URL: <http://opencv.org/> (cited on page 424).
- [2] web page. accessed 2017-02-13. URL: <http://opencv.org/opencv-3-2.html> (cited on page 424).
- [3] *5 Things to Know about dashDB*. web page. URL: https://www.ibm.com/developerworks/community/blogs/5things/entry/5_things_to_know_about_dashdb_placeholder?lang=en (cited on page 470).
- [4] M. Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. Technical report. Accessed: 2017-1-24. Cornell University, Mar. 2016 (cited on page 434).
- [5] *About OpenNebula*. URL: <https://opennebula.org/about/technology/> (cited on page 506).
- [6] *About Pivotal Gemfire*. Web Page. Version 9.0.1. Accessed: 2017-01-28. URL: http://gemfire.docs.pivotal.io/gemfire/getting_started/gemfire_overview.html (cited on page 475).
- [7] Abed Abu-Dbai et al. “Enterprise Resource Management in Mesos Clusters”. In: *Proceedings of the 9th ACM International on Systems and Storage Conference*. SYSTOR ’16. Haifa, Israel: ACM, 2016, 17:1–17:1. ISBN: 978-1-4503-4381-7. DOI: [10.1145/2928275.2933272](https://doi.acm.org/10.1145/2928275.2933272). URL: <http://doi.acm.org/10.1145/2928275.2933272> (cited on page 485).
- [8] ACADGILD. *Beginner’s Guide for Impala*. Web page. Online; accessed 7-Apr-2017. Mar. 2016. URL: <https://acadgild.com/blog/beginners-guide-impala/> (cited on page 442).
- [9] ACM, Inc. *Spark SQL: Relational Data Processing in Spark*. Web Page. accessed 2017-02-19. Feb. 2016. URL: <http://dl.acm.org/citation.cfm?id=2742797> (cited on page 471).
- [10] ActiveBPEL. *Communicating with the ActiveBPEL Server Administration Interface via Web Services*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: <http://www.activevos.com>

- com/content/developers/education/sample_active_bpel_admin_api/doc/index.html (cited on page 414).
- [11] *ActiveMQ technology*. webpage. accessed: 2017-2-4. URL: <http://activemq.apache.org/> (cited on page 456).
- [12] Aduna. *RDF components*. Web Page. Accessed: 2017-1-25. URL: <https://www.w3.org/RDF/> (cited on page 479).
- [13] Aduna. *sesame components*. Web Page. Accessed: 2017-1-26. URL: <https://projects.eclipse.org/projects/technology.rdf4j> (cited on page 479).
- [14] Aerobatic. *Aerobatic - Overview*. Web Page. accessed: 2017-01-25. Jan. 2017. URL: <https://www.aerobatic.com/docs/overview/> (cited on page 436).
- [15] *Agave API Home – Features Tab*. webpage. Accessed : 02-04-2017. URL: <https://agaveapi.co/platform/features/> (cited on page 440).
- [16] A. Ailijiang, A. Charapko, and M. Demirbas. “Consensus in the Cloud: Paxos Systems Demystified”. In: *2016 25th International Conference on Computer Communication and Networks (ICCCN)*. Aug. 2016, pages 1–10. DOI: [10.1109/ICCCN.2016.7568499](https://doi.org/10.1109/ICCCN.2016.7568499). URL: <http://ieeexplore.ieee.org/abstract/document/7568499/> (cited on page 512).
- [17] Tyler Akidau et al. “MillWheel: Fault-Tolerant Stream Processing at Internet Scale”. In: *Very Large Data Bases*. 2013, pages 734–746 (cited on page 448).
- [18] Davide Albanese et al. “mlpy: Machine Learning Python”. In: *CoRR* abs/1202.6548 (2012). URL: <http://arxiv.org/abs/1202.6548> (cited on page 425).
- [19] *Allegro*. Web Page. Accessed: 2017-1-25. URL: <http://allegrograph.com/> (cited on page 478).
- [20] *Allegrow*. Web Page. Accessed: 2017-1-20. URL: <https://en.wikipedia.org/wiki/AllegroGraph> (cited on page 478).
- [21] Alphabet, Inc. *szl - Overview.wiki*. Code Repository. Online; accessed 30-jan-2017. URL: <https://code.google.com/archive/p/szl/wikis/Overview.wiki> (cited on page 446).
- [22] Alphabet, Inc. Web Page. Online; accessed 25-jan-2017. Jan. 2017. URL: <https://cloud.google.com/> (cited on page 437).
- [23] Amazon. *AWS OpsWorks*. Web Page. accessed 2017-01-25. URL: <https://aws.amazon.com/opsworks/> (cited on page 500).
- [24] *Amazon Route 53*. Web Page. Accessed: 2017-02-24. URL: https://en.wikipedia.org/wiki/Amazon_Route_53 (cited on page 509).
- [25] *Amazon S3*. Web Page. Accessed: 2017-1-27. URL: <https://aws.amazon.com/s3/> (cited on page 491).
- [26] Amazon Web services. *Amazon Redshift Management Services*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: <http://docs.aws.amazon.com/redshift/latest/mgmt/overview.html> (cited on page 445).
- [27] Amazon Web Services,Inc. *AWS Elastic Beanstalk*. Web Page. accessed 2017-2-14. URL: <https://aws.amazon.com/elasticbeanstalk/> (cited on page 436).
- [28] Amazon.com, Inc. Web Page. Online; accessed 25-jan-2017. Jan. 2017. URL: <https://aws.amazon.com/> (cited on page 437).
- [29] *Ambari*. Web Page. Accessed: 2017-2-04. URL: <https://ambari.apache.org/> (cited on page 509).
- [30] *Github apache/ambari*. Web Page. Accessed: 2017-2-04. URL: <https://github.com/apache/ambari/> (cited on page 509).
- [31] *Hortonworks Apache Ambari*. Web Page. Accessed: 2017-2-04. URL: <http://hortonworks.com/apache/ambari/> (cited on page 509).

- [32] AMQP. *AMQP is the Internet Protocol for Business Messaging*. Web Page. accessed: 2017-01-31. Jan. 2017. URL: <http://www.amqp.org/about/what> (cited on page 459).
- [33] *An Introduction to Windows Azure BLOB Storage*. Web Page. July 2013. URL: <https://www.simple-talk.com/cloud/cloud-data/an-introduction-to-windows-azure-blob-storage/> (visited on 02/13/2017) (cited on page 492).
- [34] *Any Analytics, Any Data, Simplified*. webpage. URL: <http://www.pentaho.com/product/product-overview> (cited on page 421).
- [35] Apache. web-page. accessed 2017-02-13. URL: <https://hama.apache.org/> (cited on page 453).
- [36] Apache. *Apache arrow*. Webpage. URL: <http://arrow.apache.org/> (cited on page 516).
- [37] *Apache Accumulo User Manual Version 1.8*. accessed 2017-02-26. URL: https://accumulo.apache.org/1.8/accumulo_user_manual.html (cited on page 477).
- [38] Apache Apex. *Application Developer Guide*. Web-page. URL: https://apex.apache.org/docs/apex/application_development/#application-developer-guide (cited on page 519).
- [39] Apache Apex. *Operator Development Guide*. Web-page. accessed 2017-02-27. URL: https://apex.apache.org/docs/apex/operator_development/ (cited on page 519).
- [40] Apache Avro. webpage. accessed: 2017-2-6. URL: <https://avro.apache.org/docs/current/> (cited on page 513).
- [41] Apache Cassandra. Web Page. 2016. URL: <http://cassandra.apache.org/> (cited on page 477).
- [42] Apache Giraph. web page. URL: <https://giraph.apache.org> (cited on page 453).
- [43] Apache Giraph Wiki. web page. URL: https://en.wikipedia.org/wiki/Apache_Giraph (cited on page 453).
- [44] Apache Knox. *Apache Knox Home*. Website. Feb. 2017. URL: <https://knox.apache.org/> (cited on pages 518, 519).
- [45] *Apache License, Version 2.0*. Web Page. Online; accessed 23-Feb-2017. URL: <https://www.apache.org/licenses/LICENSE-2.0> (cited on page 506).
- [46] Apache Lucene. Web Page. Jan. 2017. URL: <http://lucene.apache.org/> (cited on page 471).
- [47] *Apache Nifi (aka HDF) data flow across data center*. Web Page. URL: <https://hortonworks.com/articles/9933/apache-nifi-aka-hdf-data-flow-across-data-center.html> (cited on page 420).
- [48] *S4: Distributed Stream Computing Platform*. Web Page. Accessed: 2017-02-24. URL: <http://incubator.apache.org/s4/> (cited on pages 447, 448).
- [49] Apache Samza. en. Web Page. Page Version ID: 764035647. Feb. 2017. URL: https://en.wikipedia.org/w/index.php?title=Apache_Samza&oldid=764035647 (visited on 02/13/2017) (cited on page 448).
- [50] Apache Samza, LinkedIn's Framework for Stream Processing. Web Page. Jan. 2015. URL: <https://thenewstack.io/apache-samza-linkedin-s-framework-for-stream-processing/> (visited on 02/13/2017) (cited on page 448).
- [51] Apache Software Foundation. *Apache Incubator*. Web Page. accessed 2017-01-29. URL: <http://incubator.apache.org/> (cited on pages 443, 517).
- [52] Apache Software Foundation. *Apache MRQL*. Web Page. accessed 2017-01-29. Apr. 2016. URL: <https://mrql.incubator.apache.org/> (cited on page 443).
- [53] Apache Software Foundation. *Omid Project Incubation Status*. Web Page. accessed 2017-02-25. Apr. 2016. URL: <https://mrql.incubator.apache.org/> (cited on page 517).

- [54] Apache Software Foundation. *1.1. Document Storage*. Web page. accessed 26-feb-2017. Feb. 2017. URL: <http://docs.couchdb.org/en/stable/intro/overview.html> (cited on page 474).
- [55] Apache Software Foundation. *Apache Phoenix: OLTP and operational analytics for Apache Hadoop*. Web page. Online; accessed 25-jan-2017. Jan. 2017. URL: <http://phoenix.apache.org/> (cited on page 442).
- [56] Apache Software Foundation. *Apache Spark GraphX*. Web Page. accessed 2017-01-30. Feb. 2017. URL: <http://spark.apache.org/graphx/> (cited on page 429).
- [57] *Apache Tajo*. Web Page. Accessed: 2017-1-27. URL: <https://tajo.apache.org> (cited on page 441).
- [58] *Apache Tajo Quick Guide*. Web Page. Accessed: 2017-1-25. URL: https://www.tutorialspoint.com/apache_tajo/apache_tajo_quick_guide.htm (cited on page 441).
- [59] Apache Tinker Pop Home. *Apache TinkerPop Home*. Web Page. 2016. URL: <https://tinkerpop.apache.org/> (cited on page 430).
- [60] *Apache Wink*. Web Page. Accessed: 2017-02-24. URL: <http://wink.apache.org/index.html> (cited on page 518).
- [61] *Apache Hbase*. Web Page. Accessed: 2017-1-26. URL: <https://hbase.apache.org/> (cited on page 475).
- [62] ApacheTinkerPopDoc. *Apache TinkerPop*. Web Page. 2016. URL: <http://tinkerpop.apache.org/docs/3.1.1-incubating/tutorials/getting-started/> (cited on page 430).
- [63] *Apache Sentry Tutorial*. Web Page. Accessed: 2017-2-11. URL: <https://cwiki.apache.org/confluence/display/SENTRY/Sentry+Tutorial> (cited on page 511).
- [64] *Apache NiFi*. Web Page. URL: <https://nifi.apache.org> (cited on page 420).
- [65] *AppEngine - platform as a service*. webpage. Accessed : 02-22-2017. URL: <https://cloud.google.com/appengine> (cited on page 435).
- [66] Wikipedia. *Google App Engine*. webpage. Accessed : 02-22-2017. Jan. 2017. URL: https://en.wikipedia.org/wiki/Google_App_Engine (cited on page 435).
- [67] appfog. *Overview*. Online. URL: <https://www.ctl.io/appfog/#Overview> (cited on page 438).
- [68] AppScale. *what-is-appscale*. Web Page. 2016. URL: <https://www.appspot.com/community/what-is-appscale/> (cited on page 435).
- [69] Appscale. *why-use-appscale*. Web Page. 2016. URL: <https://www.appspot.com/get-started/deployment-types/> (cited on page 435).
- [70] Thusoo Ashish et al. *Hive – A Petabyte Scale Data Warehouse Using Hadoop*. Paper. Apr. 2010. URL: [http://datamining.uos.ac.kr/wp-content/uploads/2015/07/Hive-A-Petabyte-Scale-Data-Warehouse-Using-Hadoop.pdf/](http://datamining.uos.ac.kr/wp-content/uploads/2015/07/Hive-A-Petabyte-Scale-Data-Warehouse-Using-Hadoop.pdf) (cited on page 441).
- [71] Storage Networking Industry Association. *About the SNIA*. Web Page. Accessed: 2017-2-27. URL: <https://www.snia.org/about> (cited on page 494).
- [72] Storage Networking Industry Association. *Cloud Data Management Interface*. 1.1.1. Accessed: 2017-2-27. Storage Networking Industry Association. Colorado Springs, CO, Mar. 2015. URL: https://www.snia.org/sites/default/files/CDMI_Spec_v1.1.1.pdf (cited on page 495).
- [73] Storage Networking Industry Association. *Cloud Data Management Interface*. Web Page. Accessed: 2017-2-27. Mar. 2015. URL: <https://www.snia.org/cdmi> (cited on page 495).
- [74] *at1*. Web Page. Accessed: 2017-1-22. URL: <http://www.cyverse.org/atmosphere> (cited on page 440).

- [75] Atlassian. *eScience Central Overview*. Web Page. Accessed: 2017-1-24. URL: <https://bitbucket.org/digitalinstitute/esciencecentral/> (cited on page 419).
- [76] Abel Avram. *Phoenix: Running SQL Queries on Apache HBase [Updated]*. Web page. Online; accessed 25-jan-2017. Jan. 2013. URL: <https://www.infoq.com/news/2013/01/Phoenix-HBase-SQL> (cited on page 442).
- [77] Amazon. *AWSLambda*. Web Page. Accessed: 2/18/2017. URL: <https://aws.amazon.com/lambda/faqs/> (cited on page 460).
- [78] Amazon. *AWSLambdaEvent*. Web Page. Accessed: 2/18/2017. URL: <http://docs.aws.amazon.com/lambda/latest/dg/invoking-lambda-function.html#intro-core-components-event-sources> (cited on page 460).
- [79] *Get started with Azure Queue storage using .NET*. Web Page. Accessed: 2017-2-10. URL: <https://docs.microsoft.com/en-us/azure/storage/storage-dotnet-how-to-use-queues> (cited on page 461).
- [80] *Github Azure/ azure-stream-analytics*. Web Page. Accessed: 2017-2-03. URL: <https://github.com/Azure/azure-stream-analytics/> (cited on page 450).
- [81] *SQL Server Azure*. Web Page. URL: <https://azure.microsoft.com/en-us/services/sql-database/?b=16.50> (cited on page 466).
- [82] *Microsoft Azure Real-time data analytics*. Web Page. Accessed: 2017-2-03. URL: <https://azure.microsoft.com/en-us/services/stream-analytics/> (cited on page 450).
- [83] Vasavi*1 B et al. *HIBERNATE TECHNOLOGY FOR AN EFFICIENT BUSINESS APPLICATION EXTENSION*: Paper. June 2011. URL: <https://www.rroij.com/open-access/hibernate-technology-for-an-efficient-business-application-extension-118-125.pdf> (cited on page 464).
- [84] *Background*. web-page. accessed 2017-02-13. URL: <https://orc.apache.org/docs/> (cited on page 482).
- [85] *Riak-KV - NOSQL Key Value Database*. Web Page. Accessed: 2017-01-20. URL: <http://basho.com/products/riak-kv/> (cited on page 472).
- [86] *Riak-TS - NOSQL Time Series Database*. Web Page. Accessed: 2017-01-20. URL: <http://basho.com/products/riak-ts/> (cited on page 472).
- [87] *Riak-S2 - Cloud Object Storage Software*. Web Page. Accessed: 2017-01-20. URL: <http://basho.com/products/riak-s2/> (cited on page 472).
- [88] Kent Baxley, JD la Rosa, and Mark Wenning. “Deploying workloads with Juju and MAAS in Ubuntu 14.04 LTS”. In: *Deploying workloads with Juju and MAAS in Ubuntu 14.04 LTS*. Dell Inc, Technical White Paper, May 2014 (cited on page 498).
- [89] *Berkeley DB Tutorial and Reference Guide*. Web Page. Accessed: 2017-2-11. URL: <https://web.stanford.edu/class/cs276a/projects/docs/berkeleydb/reftoc.html> (cited on page 472).
- [90] *Berkeley DB Wiki*. Web Page. Accessed: 2017-2-11. URL: https://en.wikipedia.org/wiki/Berkeley_DB (cited on page 472).
- [91] Antonio Esposito Beniamino Di Martino Giuseppina Cretella. *Cloud Portability and Interoperability*. illustrated. New York City, USA: Springer International Publishing, 2015. ISBN: 331913700X, 9783319137001 (cited on page 493).
- [92] *What is BigQuery? Google Cloud Platform*. Web Page. Accessed: 2017-2-23. URL: <https://cloud.google.com/bigquery> (cited on page 444).
- [93] *What is BigQuery? BigQuery Documentation Google Cloud Platform*. Web Page. Accessed: 2017-2-23. URL: <https://cloud.google.com/bigquery/what-is-bigquery> (cited on pages 444, 445).
- [94] Tobias Binz et al. *OpenTOSCA – A Runtime for TOSCA-Based Cloud Applications*. Edited by Samik Basu et al. Springer Berlin Heidelberg, 2013, pages 692–695. ISBN: 978-3-642-

- 45005-1. URL: http://dx.doi.org/10.1007/978-3-642-45005-1_62 (cited on page 501).
- [95] *About Bioconductor*. Web Page. Accessed: 2017-02-10. URL: <https://www.bioconductor.org/about/> (cited on page 423).
- [96] bioKepler. *Demo Workflow*. WebPage. URL: <http://www.biokepler.org/userguide#demos> (cited on page 416).
- [97] bioKepler. *What is bioKepler*. WebPage. URL: <http://www.biokepler.org/faq#what-is-biokepler> (cited on page 415).
- [98] Bipin. *Difference between vSphere, ESXi and vCenter*. Webpage. Aug. 2012. URL: <http://www.mustbegeek.com/difference-between-vsphere-esxi-and-vcenter/> (cited on page 508).
- [99] BITTORRENT. Web Page. 2017. URL: <https://www.lifewire.com/how-torrent-downloading-works-2483513> (cited on page 483).
- [100] Oscar Boykin et al. *Summingbird*. Website. README.md, github. URL: <https://github.com/twitter/summingbird> (cited on page 446).
- [101] Oscar Boykin et al. “Summingbird: A framework for integrating batch and online mapreduce computations”. In: *Proceedings of the VLDB Endowment* 7.13 (2014), pages 1441–1451 (cited on page 447).
- [102] *Business Process Execution Language*. Web Page. Accessed: 2017-2-11. URL: https://en.wikipedia.org/wiki/Business_Process_Execution_Language (cited on page 413).
- [103] Mikio L. Braun. *Magastore, Spanner - distributed databases*. Web Page. Accessed: 2017-01-28. Nov. 2013. URL: <http://blog.mikiobraun.de/2013/03/more-google-papers-megastore-spanner-voted-commits.html> (cited on page 476).
- [104] Martin Brown. *The Technology Behind Couchbase*. Web page. Online; accessed 29-jan-2017. Mar. 2012. URL: <https://www.safaribooksonline.com/blog/2012/03/01/the-technology-behind-couchbase/> (cited on page 474).
- [105] Jason Brownlee. *A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library*. webpage. Apr. 16, 2014. URL: <http://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/> (cited on page 426).
- [106] Alfredo Buttari et al. “A class of parallel tiled linear algebra algorithms for multicore architectures”. In: *Parallel Computing* 35.1 (2009), pages 38–53. URL: <http://www.sciencedirect.com/science/article/pii/S0167819108001117> (visited on 02/13/2017) (cited on page 424).
- [107] Byung-Gon Chun. *REEFProposal - Incubator*. Web Page. accessed 2017-01-28. Aug. 2014. URL: <https://wiki.apache.org/incubator/ReefProposal> (cited on page 452).
- [108] AT&T Laboratories Cambridge. *The Medusa Applications Environment*. Web page. Last accessed: 2017.02.25. URL: <http://www.cl.cam.ac.uk/research/dtg/attarchive/medusa.html> (cited on pages 454, 455).
- [109] CASCADING. Web Page. 2017. URL: <http://www.cascading.org/projects/cascading/> (cited on page 418).
- [110] *Cedar stack*. Web Page. Accessed: 1/27/2017. URL: <https://devcenter.heroku.com/articles/stack#cedar> (cited on page 436).
- [111] Celery. Webpage. URL: <http://www.celeryproject.org/> (cited on page 486).
- [112] Celery - Distributed Task Queue. Web Page. URL: <http://docs.celeryproject.org/en/latest/index.html> (cited on page 487).
- [113] Craig Chambers et al. “FlumeJava: Easy, Efficient Data-Parallel Pipelines”. In: *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. Ac-

- cessed : 04-09-2017. 2 Penn Plaza, Suite 701 New York, NY 10121-0701, 2010, pages 363–375. URL: <http://dl.acm.org/citation.cfm?id=1806638> (cited on page 418).
- [114] Desmond Chan. *Big Data with Apache Apex*. Web-page. accessed 2017-02-27. Jan. 2016. URL: <https://jaxenter.com/big-data-apache-apex-122839.html> (cited on page 519).
- [115] *Chef Commercial Support*. Web Page. URL: <https://www.chef.io/support/> (cited on page 496).
- [116] Guoqiang Jerry Chen et al. “Realtime Data Processing at Facebook”. In: *Proceedings of the 2016 International Conference on Management of Data*. SIGMOD ’16. San Francisco, California, USA: ACM, 2016, pages 1087–1098. ISBN: 978-1-4503-3531-7. DOI: [10.1145/2882903.2904441](https://doi.acm.org/10.1145/2882903.2904441). URL: <http://doi.acm.org/10.1145/2882903.2904441> (cited on page 450).
- [117] Yueguo Chen et al. “A Study of SQL-on-Hadoop Systems”. In: *Big Data Benchmarks, Performance Optimization, and Emerging Hardware: 4th and 5th Workshops, BPOE 2014, Salt Lake City, USA, March 1, 2014 and Hangzhou, China, September 5, 2014, Revised Selected Papers*. Springer International Publishing, 2014, pages 154–166. ISBN: 978-3-319-13021-7 (cited on page 444).
- [118] Rudi Cilibrasi et al. *What is CompLearn*. webpage. URL: <http://complearn.org/> (cited on page 426).
- [119] *Cinder - Openstack*. Web Page. Accessed: 2017-1-21. URL: <https://wiki.openstack.org/wiki/Cinder> (cited on pages 489, 490).
- [120] *CINET - CyberInfrastructure for Network Science*. Web Page. URL: www.bi.vt.edu (cited on page 431).
- [121] Tait Clarridge. *Disco - A Powerful Erlang and Python Map/Reduce Framework*. Blog. accessed 25-feb-2017. May 2014. URL: <http://www.taitclarridge.com/techlog/2014/05/disco-a-powerful-erlang-and-python-mapreduce-framework.html> (cited on page 453).
- [122] *Cloud and systems management*. webpage. URL: <http://www.cisco.com/c/en/us/products/cloud-systems-management> (cited on page 499).
- [123] Vineet Badola. *Cloud Foundry Blog*. Blog. Sept. 2015. URL: <http://cloudacademy.com/blog/cloud-foundry-benefits/> (cited on page 437).
- [124] Cloudability Inc. *Cloudability Cost Management Tools | Cloudability*. Web Page. Accessed: 2017-02-23. Feb. 2017. URL: <https://www.cloudability.com/product/> (cited on page 515).
- [125] *Cloudbees Wikipedia Documentation*. Web Page. Accessed: 2017-02-13. URL: <https://en.wikipedia.org/wiki/CloudBees> (cited on page 438).
- [126] *Cloudbees Webpage Documentation*. Web Page. Accessed: 2017-02-13. URL: <https://www.cloudbees.com/products> (cited on page 438).
- [127] *OpenStack Keystone Verification*. Web Page. Accessed: 2017-2-12. URL: <https://www.cloudberrylab.com/blog/openstack-keystone-authentication-explained/> (cited on page 511).
- [128] cloudera. *Untangling Apache Hadoop YARN Part 1 Cluster and YARN Basics*. Web Page. 2015. URL: <https://blog.cloudera.com/blog/2015/09/untangling-apache-hadoop-yarn-part-1/> (cited on page 485).
- [129] Cloudera Inc. *Cloudera Impala Overview*. Web page. Online; accessed 7-Apr-2017. URL: https://www.cloudera.com/documentation/enterprise/5-3-x/topics/impala_intro.html (cited on page 442).

- [130] *CNTK/CNTKBook-20160217.pdf at master · Microsoft/CNTK · GitHub*. Web Page. URL: <https://github.com/Microsoft/CNTK/blob/master/Documentation/CNTK-TechReport/lyx/CNTKBook-20160217.pdf> (visited on 02/13/2017) (cited on page 434).
- [131] Jeffrey Ira Cohen, Luke Lonergan, and Caleb E. WeltonCohen. *Integrating map-reduce into a distributed relational database*. grant. Dec. 2016. URL: <https://www.google.com/patents/US9514188> (cited on page 485).
- [132] Community Grids Lab IU. *Some of the salient features in naradabrokering*. Web Page. Accessed: 2017-2-15. 501 N. MORTON ST, SUITE 224 BLOOMINGTON IN 47404: Pervasive Technology Labs at Indiana University, Nov. 2009. URL: <http://www.naradabrokering.org/> (cited on page 457).
- [133] Community Grids Lab IU. *The NaradaBrokering Project @ IU Community Grids Laboratory*. Web Page. Accessed: 2017-2-15. 501 N. MORTON ST, SUITE 224 BLOOMINGTON IN 47404: Pervasive Technology Labs at Indiana University, Nov. 2009. URL: <http://www.naradabrokering.org/> (cited on page 457).
- [134] Continuum Analytics. *Blaze*. Web page. accessed 25-feb-2017. 2015. URL: <http://blaze.readthedocs.io/en/latest/index.html#> (cited on page 516).
- [135] Darren Cook. *Practical Machine Learning with H2O*. O'Reilley Media, Dec. 2017, page 300. ISBN: 978-1-4919-6454-5. URL: <https://books.google.com/books?id=nJWmDQAAQBAJ&pg=PP2&dq=h2o+software> (cited on page 428).
- [136] Emilio Coppa. *Hadoop Architecture Overview*. Code Repository. accessed 2017-03-23. URL: <http://ercoppa.github.io/HadoopInternals/HadoopArchitectureOverview.html> (cited on page 451).
- [137] James C Corbett et al. “Spanner: Google’s globally distributed database”. In: *ACM Transactions on Computer Systems (TOCS)* 31.3 (2013), page 8. URL: http://dl.acm.org/ft_gateway.cfm?id=2491245&type=pdf (cited on page 476).
- [138] CoreOS. Web Page. Accessed: 1/27/2017. URL: <https://www.coreos.com/> (cited on page 507).
- [139] CoreOS. *Why CoreOS*. Web Page. accessed: 2017-01-23. Jan. 2017. URL: <https://coreos.com/why/> (cited on page 507).
- [140] iMatix Corporation. *0MQ - The Guide*. Web Page. Accessed: 2017-1-24. URL: <http://zguide.zeromq.org/page:all> (cited on page 456).
- [141] iMatix Corporation. *Distributed Messaging*. Web Page. Accessed: 2017-1-24. URL: <http://zeromq.org> (cited on page 456).
- [142] Couchbase, Inc. *Couchbase and Apache CouchDB compared*. Web page. accessed 26-feb-2017. Feb. 2017. URL: <https://www.couchbase.com/couchbase-vs-couchdb> (cited on page 474).
- [143] Cray Inc. *Cray Yarcdata Urika appliance*. Web Page. 2017. URL: <http://www.cray.com/products/Urika.aspx> (cited on page 478).
- [144] Cray Inc. *Cray Urika-GD Technical Specification*. Technical report. Cray Inc, 2014. URL: <http://www.cray.com/sites/default/files/resources/Urika-GD-TechSpecs.pdf> (cited on page 478).
- [145] *CUBRID*. Web Page. 2017. URL: <http://www.cubrid.org/> (cited on page 468).
- [146] Baojiang Cui and Tao Xi. “Security analysis of openstack keystone”. In: *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015 9th International Conference on*. IEEE. 2015, pages 283–288 (cited on page 511).
- [147] Curoverse, Inc. *Arvados | Introduction to Crunch*. Web Page. Accessed: 2017-2-22. URL: <http://doc.arvados.org/user/tutorials/intro-crunch.html> (cited on page 418).

- [148] Curoverse, Inc. *Arvados | Open Source Big Data Processing and Bioinformatics*. Web Page. Accessed: 2017-2-22. 2016. URL: <https://arvados.org> (cited on page 418).
- [149] *D3 Data-Driven Documents*. Web Page. Accessed: 2017-02-11. URL: <https://d3js.org/> (cited on page 433).
- [150] *DataFu*. Web Page. Accessed: 1/16/2017. URL: <https://datafu.incubator.apache.org/> (cited on page 423).
- [151] DataNucleus. *DataNucleus Support*. Web Page. Accessed: 2017-02-04. URL: <http://www.datanucleus.com/> (cited on page 465).
- [152] Wikipedia. *IBM DB2-Wikipedia*. Web Page. Online, Accessed: 2017-02-24. Feb. 2017. URL: https://en.wikipedia.org/wiki/IBM_DB2 (cited on page 466).
- [153] *DB2 Introduction*. Web Page. Online, Accessed: 2017-02-24. June 2016. URL: https://www.tutorialspoint.com/db2/db2_introduction.htm (cited on page 466).
- [154] DC.js. *dc.js - Dimensional Charting Javascript Library*. Web Page. accessed: 2017-01-21. Jan. 2017. URL: <https://dc-js.github.io/dc.js/> (cited on page 434).
- [155] John Denero. *CS61A: Online Textbook*. This book is derived from the classic textbook Structure and Interpretation of Computer Programs by Abelson, Sussman, and Sussman. John Denero originally modified it for Python for the Fall 2011 semester. <http://www-inst.eecs.berkeley.edu/cs61a/sp12/book/>. Berkeley, 2011. URL: <http://www-inst.eecs.berkeley.edu/~cs61a/sp12/book/communication.html> (cited on page 430).
- [156] DevTopics. Web Page. URL: <http://www.devtopics.com/kite-obscur-programming-language-of-the-month/> (cited on page 440).
- [157] DeZyre. *How LinkedIn Uses Hadoop To Leverage Big Data Analytics*. Web page. accessed 10-March-2016. This page was last modified on 18 March 2017, at 22:27. URL: <https://www.dezyre.com/article/how-linkedin-uses-hadoop-to-leverage-big-data-analytics/229> (cited on page 449).
- [158] *CDF, Common Data Format (multidimensional datasets)*. Web page. Accessed: 2017-1-28. Mar. 2014. URL: <http://www.digitalpreservation.gov/formats/fdd/fdd000226.shtml#useful> (cited on page 482).
- [159] *Distributed Machine Learning*. Web page. Accessed 25-feb-2017. Feb. 2017. URL: <http://www.mlbase.org/> (cited on page 422).
- [160] *docker,swarm*. WebPage. Accessed: 2017-8-02. URL: <https://www.docker.com/products/docker-swarm> (cited on pages 421, 496).
- [161] *Microsoft Docs Azure Stream Analytics Documentation*. Web Page. Accessed: 2017-2-03. URL: <https://docs.microsoft.com/en-us/azure/stream-analytics/> (cited on page 450).
- [162] Jack Dongarra et al. “Accelerating numerical dense linear algebra calculations with GPUs”. In: *Numerical Computations with GPUs*. Springer, 2014, pages 3–28. URL: http://link.springer.com/chapter/10.1007/978-3-319-06548-9_1 (visited on 02/13/2017) (cited on page 424).
- [163] Dormando. *Memcached*. Web Page. accessed 2017-01-30. Feb. 2015. URL: <http://www.memcached.org/> (cited on page 461).
- [164] *Apache Drill*. Web Page. Accessed: 2/4/2017. URL: <https://drill.apache.org/> (cited on page 445).
- [165] Wikipedia. *Dryad(Programming)-Wikipedia*. Web Page. Online, Accessed: 2017-02-24. Nov. 2016. URL: [https://en.wikipedia.org/wiki/Dryad_\(programming\)](https://en.wikipedia.org/wiki/Dryad_(programming)) (cited on page 416).
- [166] Michael Isard et al. “Dryad: distributed data-parallel programs from sequential building blocks”. In: *ACM SIGOPS operating systems review*. Volume 41. 3. Online, Accessed:

- 2017-02-24. ACM. 2007, pages 59–72. URL: <https://www.microsoft.com/en-us/research/wp-content/uploads/2007/03/eurosyst07.pdf> (cited on page 417).
- [167] *Rebalancing in a multi-cloud environment in Science Cloud '13*. ACM, 2013, pages 21–28. ISBN: 978-1-4503-1979-9. DOI: [10.1145/2465848.2465854](https://doi.org/10.1145/2465848.2465854). URL: <http://dl.acm.org/citation.cfm?id=2465854> (cited on page 506).
- [168] Dylan Raithel. *Apache TinkerPop Graduates to Top-Level Project*. Web Page. 2016. URL: <https://www.infoq.com/news/2016/06/tinkerpop-top-level-apache/> (cited on page 430).
- [169] *EDUROAM*. Web Page. URL: <https://www.eduroam.org/about/> (cited on page 511).
- [170] UCAR Edward Hartnett and Rew RK. “Experience with an enhanced netCDF data model and interface for scientific data access”. In: *24th Conference on IIPS*. 2008 (cited on page 481).
- [171] *Ehcache - Features*. Web Page. Accessed: 2017-01-21. URL: <http://www.ehcache.org/about/features.html> (cited on page 463).
- [172] *Ehcache - Documentation*. Web Page. Accessed: 2017-01-21. URL: <http://www.ehcache.org/documentation/3.2/getting-started.html> (cited on page 463).
- [173] *Elastic Search*. Web Page. Accessed: 2017-1-25. URL: <https://www.elastic.co/products/elasticsearch> (cited on page 431).
- [174] *Elastic Search*. Web Page. Accessed: 2017-1-25. URL: <https://en.wikipedia.org/wiki/Elasticsearch> (cited on page 432).
- [175] *Elastic Search Getting Started*. Web Page. Accessed: 2017-1-25. URL: <https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html> (cited on page 431).
- [176] *Elastic Search on Hadoop*. Web Page. Accessed: 2017-1-25. URL: <https://www.elastic.co/products/hadoop> (cited on page 432).
- [177] *Elasticsearch*. en. Web Page. Page Version ID: 767434249. Feb. 2017. URL: <https://en.wikipedia.org/w/index.php?title=Elasticsearch&oldid=767434249> (visited on 02/27/2017) (cited on page 432).
- [178] Elasticsearch. *Logstash Introduction*. Web Page. Accessed: 2017-02-11. Feb. 2017. URL: <https://www.elastic.co/guide/en/logstash/current/introduction.html> (cited on page 432).
- [179] Engle et al. “Shark: Fast Data Analysis Using Coarse-grained Distributed Memory”. In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. SIGMOD '12. Scottsdale, Arizona, USA: ACM, 2012, pages 689–692. ISBN: 978-1-4503-1247-9. DOI: [10.1145/2213836.2213934](https://doi.org/10.1145/2213836.2213934). URL: <http://doi.acm.org/10.1145/2213836.2213934> (cited on page 441).
- [180] Erlang Central. *Couchbase Performance and Scalability: Iterating with DTrace Observability*. Web page. Online; accessed 29-jan-2017. Mar. 2012. URL: <http://erlangcentral.org/videos/couchbase-performance-and-scalability-iterating-with-dtrace-observability/#.WI5uYephnRY> (cited on page 474).
- [181] Christian Esposito et al. “A Knowledge-based Platform for Big Data Analytics Based on Publish/Subscribe Services and Stream Processing”. In: *Know.-Based Syst.* 79.C (May 2015). Accessed: 2017-1-27, pages 3–17. ISSN: 0950-7051. DOI: [10.1016/j.knosys.2014.05.003](https://doi.org/10.1016/j.knosys.2014.05.003). URL: <http://dx.doi.org/10.1016/j.knosys.2014.05.003> (cited on page 455).
- [182] Patrick Th. Eugster et al. “The Many Faces of Publish/Subscribe”. In: *ACM Comput. Surv.* 35.2 (June 2003). Accessed: 2017-1-27, pages 114–131. ISSN: 0360-0300. DOI: [10.1145/857076.857078](https://doi.org/10.1145/857076.857078). URL: <http://doi.acm.org/10.1145/857076.857078> (cited on page 455).

- [183] *Exploring Big Data with Helix: Finding Needles in a Big Haystack*. Web Page. URL: https://sigmodrecord.org/publications/sigmodRecord/1412/pdfs/09_industry_Ellis.pdf (cited on page 486).
- [184] Facebook Inc. *Under the Hood: Scheduling MapReduce jobs more efficiently with Corona*. Web Page. accessed 2017-2-13. Nov. 2012. URL: <https://www.facebook.com/notes/facebook-engineering/under-the-hood-scheduling-mapreduce-jobs-more-efficiently-with-corona/10151142560538920/> (cited on page 486).
- [185] *Facebook's Graph Search puts Apache Giraph on the map*. web page. URL: <https://www.pcworld.com/article/2046680/facebook-graph-search-puts-apache-giraph-on-the-map.html> (cited on page 453).
- [186] *Facebook's New Realtime Analytics System: HBase To Process 20 Billion Events Per Day*. Web Page. Accessed:2017-2-8. Mar. 2011. URL: <http://highscalability.com/blog/2011/3/22/facebook-new-realtime-analytics-system-hbase-to-process-20.html> (cited on page 450).
- [187] *Features*. Web Page. URL: <https://openvz.org/Features> (visited on 02/13/2017) (cited on page 504).
- [188] Roy Thomas Fielding. "Architectural Styles and the Design of Network-based Software Architectures". Doctoral Dissertation. University of California, Irvine, 2000. URL: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (cited on page 518).
- [189] *FITS Vatican Library*. Web Page. URL: <https://www.vatlib.it/home.php?pag=digitalizzazione&ling=eng> (cited on page 482).
- [190] *Fits Matlab*. Web Page. URL: <https://www.mathworks.com/help/matlab/import-export/importing-flexible-image-transport-system-fits-files.html?requestedDomain=www.mathworks.com> (cited on page 482).
- [191] *FITS Nasa*. web. URL: <https://fits.gsfc.nasa.gov/> (cited on page 482).
- [192] *FITS News*. Web Page. URL: https://fits.gsfc.nasa.gov/fits_standard.html (cited on page 482).
- [193] Apache Software Foundation. *About Apache Flex*. Web Page. Accessed: 2017-03-01. Feb. 2017. URL: <http://flex.apache.org/about-whatis.html> (cited on page 519).
- [194] Joab Jackson. "Adobe donates Flex to Apache". In: *The IDG News Service* (Nov. 2011). Accessed: 2011-11-17. URL: https://www.techworld.com.au/article/407714/adobe_donates_flex_apache (cited on page 520).
- [195] Licia Florio and Klaas Wierenga. "Eduroam, providing mobility for roaming users". In: TERENA (2005). URL: <https://www.terena.org/activities/tf-mobility/docs/ppt/eunis-eduroamfinal-LF.pdf> (cited on page 511).
- [196] Google. *Google App Engine*. webpage. Accessed : 04-09-2017. URL: <https://research.google.com/pubs/pub35650.html> (cited on page 418).
- [197] For Dummies, Inc. *Cloudera Impala and Hadoop*. Web page. Online; accessed 7-Apr-2017. URL: <http://www.dummies.com/programming/big-data/hadoop/cloudera-impala-and-hadoop/> (cited on page 442).
- [198] Ian Foster. "Globus toolkit version 4: Software for service-oriented systems". In: *Journal of computer science and technology* 21.4 (2006), page 513 (cited on page 488).
- [199] *Foundation » OpenStack Open Source Cloud Computing Software*. Web Page. Online; accessed 23-Feb-2017. URL: <https://www.openstack.org/foundation/> (cited on page 505).
- [200] Apache Software Foundation. *Apache Parquet*. Web Page. Accessed: 02-06-2017. URL: <https://parquet.apache.org/documentation/latest> (cited on page 483).

- [201] Apache Software Foundation. *Kafka-A distributed streaming platform*. Web Page. URL: <https://kafka.apache.org/> (cited on page 458).
- [202] Apache Software Foundation. *GenApp - Apache Airavata - Apache Software Foundation*. Web page. Accessed: 2017-2-22. Aug. 2014. URL: <https://cwiki.apache.org/confluence/display/AIRAVATA/GenApp> (cited on page 414).
- [203] Apache Software Foundation. *Apache Airavata*. Web page. Accessed: 2017-2-22. 2016. URL: <http://airavata.apache.org> (cited on page 414).
- [204] The Apache Software Foundation. *Trident API Overview*. Web Page. Accessed: 2017-1-24. URL: <http://storm.apache.org/releases/1.0.0/Trident-API-Overview.html> (cited on page 415).
- [205] The Apache Software Foundation. *Trident Tutorial*. Web Page. Accessed: 2017-1-24. URL: <http://storm.apache.org/releases/0.10.1/Trident-tutorial.html> (cited on page 415).
- [206] The Eclipse Foundation. *Eclipse Winery*. Web Page. Accessed: 2017-1-24. URL: <https://projects.eclipse.org/projects/soa.winery> (cited on page 501).
- [207] Tony Fountain et al. “The Open Source DataTurbine Initiative: Empowering the Scientific Community with Streaming Data Middleware”. In: *Bulletin - Ecological Society of America* 93.3 (July 2012), pages 242–252. URL: <http://onlinelibrary.wiley.com/doi/10.1890/0012-9623-93.3.242/abstract> (cited on page 451).
- [208] G. Fox and S. Pallickara. “Deploying the NaradaBroker Substrate in Aiding Efficient Web and Grid Service Interactions”. In: *Proceedings of the IEEE* 93.3 (Mar. 2005). Accessed : 2017-02-15, pages 564–577. ISSN: 0018-9219. DOI: [10.1109/JPROC.2004.842759](https://doi.org/10.1109/JPROC.2004.842759) (cited on page 457).
- [209] *FTP Wikipedia*. Web Page. Accessed: 2017-02-02. URL: <https://cwiki.apache.org/confluence/display/Hive/RCFileCat> (cited on page 483).
- [210] *FUSE Site*. Web Page. URL: <http://fuse.sourceforge.net> (cited on page 490).
- [211] Edgar Gabriel et al. “Open MPI: Goals, Concept, and Design of a Next Generation MPI Implementation”. In: *Proceedings, 11th European PVM/MPI Users’ Group Meeting*. Budapest, Hungary, Sept. 2004, pages 97–104. URL: <https://www.open-mpi.org/papers/euro-pvmm MPI-2004-overview/euro-pvmm MPI-2004-overview.pdf> (cited on page 515).
- [212] *About Ansible Galaxy*. Web Page. Accessed: 2017-2-06. URL: <https://galaxy.ansible.com/intro/> (cited on page 416).
- [213] *GitHub ansible/galaxy*. Web Page. Accessed: 2017-2-06. URL: <https://github.com/ansible/galaxy/> (cited on page 416).
- [214] Luke Galea. *Graylog2 optimization for High-log Environments*. webpage. Accessed : 02-21-2017. July 2012. URL: <https://dzone.com/articles/graylog2-optimization-high-log> (cited on page 432).
- [215] *Galera Cluster*. Web Page. 2017. URL: <http://galeracluster.com/> (cited on page 468).
- [216] Gamefromscratch. *Hands On With Amazon’s Lumberyard Game Engine - YouTube*. URL: <https://www.youtube.com/watch?v=FUD1TTbt4qE> (visited on 02/27/2017) (cited on page 447).
- [217] *Ganglia Monitoring System*. Web Page. Accessed: 2017-02-24. URL: <http://ganglia.info/> (cited on page 509).
- [218] *Ganglia (software)*. Web Page. Accessed: 2017-02-24. URL: [https://en.wikipedia.org/wiki/Ganglia_\(software\)](https://en.wikipedia.org/wiki/Ganglia_(software)) (cited on page 509).
- [219] Mitch Garnaat. *boto components*. Web Page. Accessed: 2017-1-25. URL: <http://boto.cloudhackers.com/en/latest/> (cited on page 497).
- [220] Mitch Garnaat. *boto3-documentation components*. Web Page. Accessed: 2017-1-26. URL: <https://boto3.readthedocs.io/en/latest/> (cited on page 497).

- [221] Mitch Garnaat. *boto-amazon-python-sdk components*. Web Page. Accessed: 2017-1-26. URL: <https://aws.amazon.com/sdk-for-python/> (cited on page 497).
- [222] Mitch Garnaat. *boto-github components*. Web Page. Accessed: 2017-1-25. URL: <https://github.com/boto/boto3> (cited on page 497).
- [223] Robert C. Gentleman et al. “Bioconductor: open software development for computational biology and bioinformatics”. In: *Genome Biology* 5.10 (2004), R80. ISSN: 1474-760X. DOI: [10.1186/gb-2004-5-10-r80](https://doi.org/10.1186/gb-2004-5-10-r80). URL: <http://dx.doi.org/10.1186/gb-2004-5-10-r80> (cited on page 423).
- [224] *Get started with Azure Blob storage (object storage) using .NET | Microsoft Docs*. Web Page. URL: <https://docs.microsoft.com/en-us/azure/storage/storage-dotnet-how-to-use-blobs> (visited on 02/13/2017) (cited on page 492).
- [225] Ali Ghodsi et al. “Dominant Resource Fairness: Fair Allocation of Multiple Resource Types.” In: *NSDI*. Volume 11. 11. 2011, pages 24–24 (cited on page 485).
- [226] *GitHub - elastic/kibana: Kibana analytics and search dashboard for Elasticsearch*. Web Page. URL: <https://github.com/elastic/kibana> (visited on 02/27/2017) (cited on page 432).
- [227] *GitHub - jupyter/jupyter: Jupyter metapackage for installation, docs and chat*. Web page. URL: <https://github.com/jupyter/jupyter> (visited on 02/27/2017) (cited on page 416).
- [228] *GitHub - jupyter/notebook: Jupyter Interactive Notebook*. Web page. URL: <https://github.com/jupyter/notebook> (visited on 02/27/2017) (cited on page 416).
- [229] *CDAP Applications*. Code Repository. Accessed: 2017-2-18. May 2015. URL: <https://github.com/caskdata/cdap-apps> (cited on page 516).
- [230] *coreos/rkt*. Web Page. Accessed: 1/27/2017. URL: <https://github.com/coreos/rkt/> (cited on page 507).
- [231] *About the Globus Toolkit*. Web Page. Accessed: 2017-2-26. URL: <http://toolkit.globus.org/toolkit/about.html> (cited on page 488).
- [232] Google Inc. *Google Cloud Datastore Overview*. Web page. Online; accessed 9-Apr-2017. URL: <https://cloud.google.com/datastore/docs/concepts/overview> (cited on pages 480, 481).
- [233] Sunila Gollapudi. *Getting Started with Greenplum for Big Data Analytics*. Packt Publishing, Oct. 25, 2013. 172 pages. ISBN: 1782177043. URL: http://www.ebook.de/de/product/21653990/sunila_gollapudi_getting_started_with_greenplum_for_big_data_analytics.html (cited on page 484).
- [234] Jose Gonzalez and Jeff Lindsay. *Buildstep*. Code repository. Accessed: 2017-1-24. July 2015. URL: <https://github.com/program/buildstep> (cited on page 501).
- [235] Tom Goodale et al. *A Simple API for Grid Applications*. webpage. Accessed : 02-01-2017. Sept. 2013. URL: <https://www.ogf.org/documents/GFD.90.pdf> (cited on page 495).
- [236] Google. *Cloud Bigtable*. Web Page. accessed 2017-01-29. URL: <https://cloud.google.com/bigtable/> (cited on page 475).
- [237] Google. *CLOUD DATAFLOW*. WebPage. URL: <https://cloud.google.com/dataflow/> (cited on page 420).
- [238] Google. *Cloud Dataflow*. Web Page. Accessed: 02/03/2016. URL: <https://cloud.google.com/dataflow/> (cited on page 446).
- [239] Google. *Google Cloud Prediction API Documentation*. Web Page. Accessed 2017-1-26. URL: <https://cloud.google.com/prediction/docs/> (cited on page 425).
- [240] Google. *Google Cloud Translation API Documentation*. Web Page. Accessed 2017-2-20. URL: <https://cloud.google.com/translate/docs/> (cited on page 425).

- [241] Google. *Google Fusion Tables: Data Management, Integration and Collaboration in the Cloud*. 2012. URL: <http://homes.cs.washington.edu/~alon/files/socc10.pdf> (cited on page 431).
- [242] Google. *FusionTableSupportHelp*. Web Page. 2017. URL: <https://support.google.com/fusiontables/answer/171181?hl=en> (cited on page 431).
- [243] Mike Burrows. *Chubby Site*. Web Page. URL: <https://research.google.com/archive/chubby.html> (cited on page 512).
- [244] *Google Cloud*. Webpage. URL: https://en.wikipedia.org/wiki/Google_Cloud (cited on page 508).
- [245] *Google cloud platform*. Webpage. URL: <https://cloud.google.com> (cited on page 508).
- [246] *Google Cloud SQL*. web page. URL: <https://cloud.google.com/sql/> (cited on page 469).
- [247] *Google Cloud SQL FAQ*. Webpage. URL: <https://cloud.google.com/sql/faq#version> (cited on page 469).
- [248] *Google Cloud Storage Documentation*. Web Page. Accessed: 02-06-2017. URL: <https://cloud.google.com/storage/docs> (cited on page 492).
- [249] Google Developers. *Cloud Machine Learning*. Web page. accessed 16-March-2017. URL: <https://cloud.google.com/ml-engine/> (cited on page 520).
- [250] Google Developers. *Cloud ML Engine Overview*. Web page. accessed 16-March-2017. URL: <https://cloud.google.com/ml-engine/docs/concepts/technical-overview> (cited on page 520).
- [251] Google Inc. *Balancing Strong and Eventual Consistency with Google Cloud Datastore*. Web page. Online; accessed 9-Apr-2017. URL: <https://cloud.google.com/datastore/docs/articles/balancing-strong-and-eventual-consistency-with-google-cloud-datastore/> (cited on page 481).
- [252] Google Inc. *Google-pub-sub-scalable-messaging-middleware*. Web Page. Accessed: 2017-2-10. URL: <https://cloud.google.com/pubsub/> (cited on page 460).
- [253] Google Inc. *What-is-Google-Pub-Sub*. Web Page. Accessed: 2017-2-09. URL: <https://cloud.google.com/pubsub/docs/overview> (cited on page 460).
- [254] Gora - *in-memory data model and persistence for big data*. Web Page. Accessed: 2017-01-18. URL: <http://gora.apache.org/> (cited on page 461).
- [255] Clinton Gormley and Zachary Tong. *Elasticsearch - The Definitive Guide : A Distributed REAL-TIME SEARCH AND ANALYTICS ENGINE*. 1st. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc, 2015. ISBN: 9781449358549 (cited on page 432).
- [256] Pivotal Software Inc. *gpfdist*. Web Page. 2017. URL: http://gpdb.docs.pivotal.io/4330/utility_guide/admin_utilities/gpfdist.html (cited on page 484).
- [257] GraphLab. *GraphLab*. Web Page. accessed 2017-01-28. Dec. 2012. URL: <http://www.select.cs.cmu.edu/code/graphlab/> (cited on page 429).
- [258] *Dashboards - 2.2.1 documentation*. webpage. Accessed : 02-21-2017. 2012. URL: <http://docs.graylog.org/en/2.2/pages/dashboards.html> (cited on page 433).
- [259] Rick Grehan. *NoSQL Showdown: MongoDB vs. Couchbase*. Web page. Online; accessed 29-jan-2017. Mar. 2013. URL: <http://www.infoworld.com/article/2613970/nosql/nosql-showdown--mongodb-vs--couchbase.html> (cited on page 474).
- [260] *Globus Online (GridFTP)*. Web Page. Accessed:1/16/2017. URL: <https://en.wikipedia.org/wiki/GridFTP> (cited on page 484).
- [261] C.J. Gronlund et al. *Azure Machine Learning*. Web Page. Accessed: 2017-2-27. Jan. 2017. URL: <https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-what-is-machine-learning#what-is-machine-learning-in-the-microsoft-azure-cloud> (cited on page 425).

- [262] RDF Working Group. *Resource Description Framework (RDF)*. Online. Feb. 2014. URL: <https://www.w3.org/RDF/> (cited on page 477).
- [263] *Grow Hosting Business with Software Platform*. Web Page. URL: <https://jelastic.com/cloud-business-for-hosting-providers/> (visited on 02/13/2017) (cited on page 438).
- [264] *H2O Website Documentation components*. Web Page. URL: <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/welcome.html> (cited on page 428).
- [265] *H2O Wikipedia Documentation components*. Web Page. Accessed : 2017-02-12. URL: [https://en.wikipedia.org/wiki/H2O_\(software\)](https://en.wikipedia.org/wiki/H2O_(software)) (cited on page 428).
- [266] Hadoop Apache. *Apache Software Foundation*. Web Page. 2016. URL: <https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html> (cited on page 485).
- [267] Bajda Pawlikowski et al. *An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads*. Web Page. Accessed: 2017-02-12. URL: <db.cs.yale.edu/hadoopdb/hadoopdb.html> (cited on page 443).
- [268] Azzam Haidar, Jakub Kurzak, and Piotr Luszczek. “An improved parallel singular value algorithm and its implementation for multicore hardware”. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. ACM, 2013, page 90. URL: <http://dl.acm.org/citation.cfm?id=2503292> (visited on 02/13/2017) (cited on page 424).
- [269] H. F. Halaoui. “A spatio temporal indexing structure for efficient retrieval and manipulation of discretely changing spatial data”. In: *Journal of Spatial Science* 53.2 (2008), pages 1–12. DOI: <10.1080/14498596.2008.9635146>. URL: <http://dx.doi.org/10.1080/14498596.2008.9635146> (cited on page 434).
- [270] Matthew Hardin. *Understanding LMDB Database File Sizes and Memory Utilization*. WebPage. May 2016. URL: <https://symas.com/understanding-lmdb-database-file-sizes-and-memory-utilization/> (cited on page 462).
- [271] Harp. *Harp*. Web Page. accessed 2017-01-28. Jan. 2012. URL: <http://harpjs.com> (cited on page 456).
- [272] *Haystack*. Web Page. URL: www.project-haystack.org (cited on page 489).
- [273] Hazelcast. *Open Source In-Memory Data Grid*. Code Repository. accessed 2017-01-29. Jan. 2017. URL: <https://github.com/hazelcast/hazelcast> (cited on page 462).
- [274] *hdfs*. Web Page. Accessed: 2017-1-21. URL: <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html> (cited on page 489).
- [275] Yongqiang He et al. “RCFile: A fast and space-efficient data placement structure in MapReduce-based warehouse systems”. In: *Data Engineering (ICDE), 2011 IEEE 27th International Conference on Engineering*. IEEE. 2011, pages 1199–1208 (cited on page 482).
- [276] Michael Heap. *Ansible From Beginner to Pro*. Edited by L. Corrigan et al. apress, 2016 (cited on page 416).
- [277] *Heat - Openstack*. webpage. Accessed : 01-15-2017. URL: <https://wiki.openstack.org/wiki/Heat> (cited on page 498).
- [278] *Heroku*. Web Page. Accessed: 1/27/2017. URL: <https://devcenter.heroku.com/> (cited on page 436).
- [279] Hugo Hiden et al. *e-Science Central: Cloud-based e-Science and its application to chemical property modelling*. Technical report. Newcastle, England: University of Newcastle upon Tyne, Nov. 2010. URL: http://eprint.ncl.ac.uk/file_store/production/179301/8EA32A5C-DE97-44A8-869B-81731044F2A8.pdf (cited on page 419).

- [280] *Home · Microsoft/CNTK Wiki · GitHub*. Web Page. URL: <https://github.com/Microsoft/CNTK/wiki> (visited on 02/13/2017) (cited on page 435).
- [281] *How Puppet Works*. Web Page. URL: <http://www.slashroot.in/puppet-tutorial-how-does-puppet-work> (cited on page 496).
- [282] *How To Create OpenVZ Container In OpenVZ | Unixmen*. Web Page. URL: <https://www.unixmen.com/how-to-create-openvz-container-in-openvz/> (visited on 02/13/2017) (cited on page 504).
- [283] *How To Use Logstash and Kibana To Centralize Logs On Ubuntu 14.04*. Web Page. URL: <https://www.digitalocean.com/community/tutorials/how-to-use-logstash-and-kibana-to-centralize-and-visualize-logs-on-ubuntu-14-04> (visited on 02/27/2017) (cited on page 432).
- [284] HowStuffWorks.com. *What are relational databases?* Online. Mar. 2001. URL: <http://computer.howstuffworks.com/question599.htm> (cited on page 467).
- [285] Robert Kallman et al. “H-Store: a High-Performance, Distributed Main Memory Transaction Processing System”. In: *Proc. VLDB Endow.* 1.2 (2008), pages 1496–1499. ISSN: 2150-8097. DOI: <http://doi.acm.org/10.1145/1454159.1454211>. URL: <http://hstore.cs.brown.edu/papers/hstore-demo.pdf> (cited on page 463).
- [286] HPE. *HPE Helion Stackato*. Webpage. URL: <https://www.hpe.com/us/en/software/multi-cloud-platform.html> (cited on page 438).
- [287] *H-Store*. Web Page. Accessed: 1/16/2017. URL: <http://hstore.cs.brown.edu/> (cited on page 463).
- [288] *H-StoreWiki*. Web Page. Accessed: 1/16/2017. URL: <https://en.wikipedia.org/wiki/H-Store> (cited on page 464).
- [289] IBM. *BlueworksLive*. Web Page. Accessed: 2017-1-24. URL: <https://www.blueworkslive.com/home> (cited on page 502).
- [290] IBM. *Exercise: Analyze business processes with IBM BPM Blueprint*. Web Page. Accessed: 2017-1-24. URL: <http://www.ibm.com/developerworks/downloads/soasandbox/blueprint.html> (cited on page 502).
- [291] IBM. *IBM Blueworks Live*. Web Page. Accessed: 2017-1-24. URL: https://en.wikipedia.org/wiki/IBM_Blueworks_Live (cited on page 502).
- [292] IBM. *IBM Spectrum Scale*. Web Page. accessed 2017-01-29. URL: <http://www-03.ibm.com/systems/storage/spectrum/scale/> (cited on page 491).
- [293] IBM. *What is Flume?* web page. accessed 201-02-06. URL: <https://www-01.ibm.com/software/data/infosphere/hadoop/flume/> (cited on page 484).
- [294] IBM Corporation. *IBM System G documentation*. Web Page. Accessed: 2017-2-19. IBM, 2014. URL: <http://systemg.research.ibm.com/> (cited on page 429).
- [295] IBM Corporation. *IBM System G documentation-2*. Web Page. Accessed: 2017-2-17. Predictive Analytics Today, 2017. URL: <http://www.predictiveanalyticstoday.com/ibm-system-g-native-store/> (cited on page 429).
- [296] ibm. *Extreme cloud administration toolkit*. Webpage. URL: <http://www-03.ibm.com/systems/technicalcomputing/xcat/> (cited on page 498).
- [297] IBM Corp. Web Page. Online; accessed 25-jan-2017. Jan. 2017. URL: www.softlayer.com (cited on page 437).
- [298] IBM Corp. Web Page. Online; accessed 25-jan-2017. Jan. 2017. URL: [%7Bhttps://www.ibm.com/cloud-computing/bluemix/%7D](https://www.ibm.com/cloud-computing/bluemix/%7D) (cited on page 437).
- [299] *IBM dashDB*. Web Page. URL: <https://www.ibm.com/analytics/us/en/technology/cloud-data-services/dashdb/> (cited on page 470).
- [300] *IBM Watson*. Web Page. Accessed: 2017-1-25. URL: [https://en.wikipedia.org/wiki/Watson_\(computer\)](https://en.wikipedia.org/wiki/Watson_(computer)) (cited on page 428).

- [301] *IBM Watson Product Page*. Web Page. Accessed: 2017-1-25. URL: <https://www.ibm.com/watson> (cited on page 428).
- [302] Roger Ignazio. *Mesos in Action*. 1st. Greenwich, CT, USA: Manning Publications Co., May 17, 2016. 272 pages. ISBN: 1617292923. URL: <http://dl.acm.org/citation.cfm?id=3006364> (cited on page 485).
- [303] ImageJ. *ImageJ Introduction*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: <https://imagej.nih.gov/ij/docs/intro.html> (cited on page 424).
- [304] Inca - Periodic, automated, user-level cyberinfrastructure testing. Web Page. Accessed: 2017-01-16. URL: <http://inca.sdsc.edu/> (cited on page 510).
- [305] InCommon. *What is the InCommon Federation?* Webpage. URL: https://spaces.internet2.edu/download/attachments/2764/final_InCommon.pdf (cited on page 510).
- [306] Indiana University et al. *SEAGrid Portal*. Web page. Accessed: 2017-2-22. July 2016. URL: <https://seagrid.org> (cited on page 414).
- [307] Infinispan. Webpage. URL: <https://en.wikipedia.org/wiki/Infinispan> (cited on page 463).
- [308] Intel® Data Analytics Acceleration Library (Intel® DAAL). Web Page. Accessed: 02-23-2017. URL: <https://software.intel.com/en-us/intel-daal> (cited on page 427).
- [309] Intel Graph Analytics Solutions. Intel® Graph Builder for Apache Hadoop® Software v2. Web Page. URL: <https://www-ssl.intel.com/content/www/us/en/software/intel-graph-builder-for-apache-hadoop-software-v2-product-detail.html> (cited on page 430).
- [310] Introducing Espresso - LinkedIn's hot new distributed document store | LinkedIn Engineering. Web Page. Online; accessed 23-Feb-2017. URL: <https://engineering.linkedin.com/espresso/introducing-espresso-linkedin-s-hot-new-distributed-document-store> (cited on page 473).
- [311] Introduction | Kibana User Guide [5.2] | Elastic. Web Page. Docs/Kibana/Reference/5.2. URL: <https://www.elastic.co/guide/en/kibana/current/introduction.html> (visited on 02/27/2017) (cited on page 432).
- [312] Introduction to Infinispan. Distributed in-memory key/value data grid and cache. Webpage. URL: <http://infinispan.org/about/> (cited on page 463).
- [313] IPython. en. Web page. Page Version ID: 767266837. Feb. 2017. URL: <https://en.wikipedia.org/w/index.php?title=IPython&oldid=767266837> (visited on 02/27/2017) (cited on page 416).
- [314] Muhammad Hussain Iqbal and Tariq Rahim Soomro. “Big Data Analysis: Apache Storm Perspective”. In: *International Journal of Computer Trends and Technology (IJCTT)-2015*. Jan. 2015 (cited on page 447).
- [315] iRod. Web Page. URL: <https://irods.org/> (cited on page 481).
- [316] iRod. Web Page. URL: <https://github.com/irods/irods> (cited on page 481).
- [317] Mohammad Islam et al. “Oozie: towards a scalable workflow management system for hadoop”. In: *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*. ACM. 2012, page 4 (cited on page 417).
- [318] C. Issariyapat et al. “Using Nagios as a groundwork for developing a better network monitoring system”. In: *2012 Proceedings of PICMET '12: Technology Management for Emerging Technologies*. July 2012, pages 2771–2777 (cited on page 510).
- [319] IU 'Twister' software improves Google's MapReduce for large-scale scientific data analysis. Web Page. URL: <https://phys.org/news/2010-03-iu-twister-software-google-mapreduce.html> (visited on 02/13/2017) (cited on page 452).

- [320] *IU Twister software improves Google's MapReduce for large-scale scientific data analysis: IU News Room: Indiana University.* Web Page. URL: <http://newsinfo.iu.edu/news-archive/13726.html> (visited on 02/13/2017) (cited on page 452).
- [321] Jacob Jackson. *Google service analyzes live streaming data.* WebPage. June 2014. URL: <http://www.infoworld.com/article/2607938/data-mining/google-service-analyzes-live-streaming-data.html> (cited on page 420).
- [322] Jake Luciani. *Solandra Wiki.* Code Repository. Accessed: 2017-02-12. Feb. 2017. URL: <https://github.com/tjake/Solandra/wiki/Solandra-Wiki> (cited on page 471).
- [323] Jake Luciani. *Solandra Wiki.* Code Repository. Accessed: 2017-02-12. Feb. 2017. URL: <https://github.com/tjake/Solandra> (cited on page 471).
- [324] *Java Database Connectivity.* Web Page. Accessed: 2017-1-30. URL: https://en.wikipedia.org/wiki/Java_Database_Connectivity (cited on page 465).
- [325] *JClouds - The Java Multi-Cloud Toolkit.* Web Page. Accessed: 2017-01-20. URL: <https://jclouds.apache.org/> (cited on page 493).
- [326] *Jelastic.* en. Web Page. Page Version ID: 754931676. Dec. 2016. URL: <https://en.wikipedia.org/w/index.php?title=Jelastic&oldid=754931676> (visited on 02/13/2017) (cited on page 437).
- [327] Shantenu Jha et al. "Grid Interoperability at the Application Level Using SAGA". In: *07 Proceedings of the Third IEEE International Conference on e-Science and Grid Computing.* Dec. 2007. ISBN: 0769530648 (cited on page 495).
- [328] Ana Lucia Varbanescu Jianbin Fang and Henk Sips. "Sesame: A User-Transparent Optimizing Framework for Many-Core Processors". In: *IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing.* 2013, pages 70–73 (cited on page 479).
- [329] Bingsheng He Jianlong Zhong. "Medusa: Simplified Graph Processing on GPUs". In: *IEEE Transactions on Parallel and Distributed Systems* 25.6 (Apr. 2013), pages 1–11. URL: http://pdcc.ntu.edu.sg/xtra/paper/2013ago/Medusa_TPDS13.pdf (cited on page 454).
- [330] Jitterbit. *Data Integration and ETL | Jitterbit | Integrate data from any source.* Web Page. accessed: 2017-1-26. Jitterbit. URL: <https://www.jitterbit.com/etl-data-integration/> (cited on page 420).
- [331] *Integration Made Easy - Jitterbit.* Accessed: 2017-1-26. Jitterbit, Inc. 1 Kaiser Plaza Suite 701 Oakland, CA 94612. URL: <http://www.jitterbit.com/Files/Product/Jitterbit-General-Datasheet.pdf> (cited on page 420).
- [332] *Technical Overview - Jitterbit.* Accessed: 2017-1-26. Jitterbit, Inc. 2011. URL: <http://www.jitterbit.com/Files/Product/JitterbitTechnicalOverview.pdf> (cited on page 420).
- [333] *The Java EE 6 Tutorial.* webpage. Accessed : 01-18-2017. URL: <http://docs.oracle.com/javaee/6/tutorial/doc/bncest.html> (cited on page 459).
- [334] Tim Jones. *Anatomy of the libvirt virtualization.* Webpage. Jan. 2010. URL: <https://www.ibm.com/developerworks/library/l-libvirt/> (cited on page 493).
- [335] David Josephsen. *Nagios: Building Enterprise-Grade Monitoring Infrastructures for Systems and Networks.* 2nd. Upper Saddle River, NJ, USA: Prentice Hall Press, 2013. ISBN: 013313573X, 9780133135732 (cited on page 510).
- [336] The Apache Software Foundation. *Apache OpenJPA - Wikipedia.* webpage. Accessed : 02-22-2017. Jan. 2017. URL: https://en.wikipedia.org/wiki/Apache_OpenJPA (cited on page 464).
- [337] JPAB. *JPA Performance Benchmark.* Web Page. Accessed: 2017-02-04. URL: <http://www.jpab.org/DataNucleus.html> (cited on page 465).

- [338] Canonical Ltd. *Juju*. Web Page. URL: <https://www.ubuntu.com/cloud/juju> (cited on page 498).
- [339] Guy E. Blelloch Julian Shun and Laxman Dhulipala. “Smaller and Faster: Parallel Processing of Compressed Graphs with Ligra+”. In: *ACM SIGPLAN Notices - PPoPP '13DCC '15 Proceedings of the 2015 Data Compression Conference*. Pittsburgh, Pennsylvania USA: IEEE Computer Society Washington, DC, USA, 2015, pages 403–412. ISBN: 978-1-4799-8430-55. DOI: [10.1109/DCC.2015.8](https://doi.org/10.1109/DCC.2015.8). URL: <http://dl.acm.org/citation.cfm?id=2860198> (cited on page 454).
- [340] Steven van Wel Jurg Vliet Flavia Paganelli and Dara Dowd. *Elastic Beanstalk*. 1st edition. O’Reilly Media, Inc., 2011. ISBN: 1449306640 (cited on page 436).
- [341] *Astronomical Image Processing with Hadoop*. Volume 442. Astronomical Data Analysis Software and Systems XX. ASP Conference Proceedings, July 2011. URL: <http://adsabs.harvard.edu/abs/2011ASPC..442...93W> (cited on page 482).
- [342] Riyad Kalla. *Well put! When should you use MongoDB vs Couchbase versus Redis...* Web page. Online; accessed 29-jan-2017. Oct. 2011. URL: <http://rick-hightower.blogspot.com/2014/04/well-put-when-should-you-use-mongodb-vs.html> (cited on page 475).
- [343] Mike Kavis. *THE PROS AND CONS OF PRIVATE AND PUBLIC PAAS*. Webpage. June 2013. URL: <https://www.virtualizationpractice.com/the-pros-and-cons-of-private-and-public-paas-21961/> (cited on page 438).
- [344] Keith-Bisset. *CINET - A Cyber-Infrastructure for Network Science*. Web Page. URL: www.portal.futuresystems.org/project/233 (cited on page 431).
- [345] Amol Kekre. *Apache Apex blog*. Web Page. Accessed: 2017-02-26. Sept. 2015. URL: <https://www.datatorrent.com/blog/introducing-apache-apex-incubating/> (cited on page 518).
- [346] Ben Kepes. *Understanding the Cloud Computing Stack: SaaS, PaaS, IaaS*. white paper. Rackspace, 2015. URL: <https://support.rackspace.com/white-paper/understanding-the-cloud-computing-stack-saas-paas-iaas/> (cited on page 438).
- [347] Kepler Project. *The Kepler Project*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: <https://kepler-project.org> (cited on page 414).
- [348] Justin Kestelyn. *Phoenix in 15 Minutes or Less*. Web page. Online; accessed 25-jan-2017. Mar. 2013. URL: <http://blog.cloudera.com/blog/2013/03/phoenix-in-15-minutes-or-less/> (cited on page 442).
- [349] Agargent. *About kestrel*. Web Page. Accessed: 2017-02-12. URL: <https://github.com/twitter-archive/kestrel> (cited on page 458).
- [350] *Keystone Wiki*. Web Page. Accessed: 2017-2-12. URL: <https://wiki.openstack.org/wiki/Keystone> (cited on page 511).
- [351] *Kibana*. en. Web Page. Page Version ID: 767434139. Feb. 2017. URL: <https://en.wikipedia.org/w/index.php?title=Kibana&oldid=767434139> (visited on 02/27/2017) (cited on page 432).
- [352] *Kibana: Explore, Visualize, Discover Data |Elastic*. Web Page. URL: <https://www.elastic.co/products/kibana> (visited on 02/27/2017) (cited on page 432).
- [353] Jik-Soo Kim et al. “HTCaaS: Leveraging Distributed Supercomputing Infrastructures for Large-Scale Scientific Computing”. In: *ACM MTAGS 13*. 2013 (cited on page 488).
- [354] *Kinesis - real-time streaming data in the AWS cloud*. Web Page. Accessed: 2017-01-17. URL: <https://aws.amazon.com/kinesis/> (cited on page 449).
- [355] Wikipedia. *Apache OpenJPA*. webpage. Accessed : 02-22-2017. Dec. 2016. URL: <http://openjpa.apache.org/> (cited on page 464).

- [356] Oliver Kopp et al. “Winery – A Modeling Tool for TOSCA-based Cloud Applications”. In: *11th International Conference on Service-Oriented Computing*. Springer, Dec. 2013, pages 700–704. URL: http://www.iaas.uni-stuttgart.de/RUS-data/INPROC-2013-46%20-%20Winery_A_Modeling_Tool_for_TOSCA-based_Cloud_Applications.pdf (cited on page 501).
- [357] Lars Kotthoff. *LLAMA: Leveraging Learning to Automatically Manage Algorithms*. Research paper. Apr. 30, 2014. URL: <https://arxiv.org/abs/1306.1031> (cited on page 486).
- [358] T. Kraska et al. “MLbase: A Distributed Machine Learning System”. In: *MLbase: A Distributed Machine Learning System*. Conference on Innovative Data Systems Research. 2013 (cited on page 422).
- [359] Kubernetes. *Kubernetes Site*. Web Page. URL: <https://kubernetes.io/docs/whatisk8s/> (cited on page 500).
- [360] Jakub Kurzak et al. “Scheduling dense linear algebra operations on multicore processors”. In: *Concurrency and Computation: Practice and Experience* 22.1 (2010), pages 15–44. URL: <http://onlinelibrary.wiley.com/doi/10.1002/cpe.1467/full> (visited on 02/13/2017) (cited on page 425).
- [361] *KVM Wikipedia Documentation*. Web Page. Accessed : 2017-02-12. URL: https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine (cited on page 503).
- [362] *KVM webpage documentation*. Web Page. Accessed : 2017-02-13. URL: http://www.linux-kvm.org/page/Main_Page (cited on page 503).
- [363] *Kyoto Cabinet*. Web Page. Accessed:1/16/2017. URL: <http://fallabs.com/kyotocabinet/> (cited on page 445).
- [364] *Kyoto Cabinet*. Web Page. Accessed: 2017-1-27. URL: <http://fallabs.com/kyotocabinet/> (cited on page 472).
- [365] Aapo Kyrola, Guy Blelloch, and Carlos Guestrin. “GraphChi: Large-scale Graph Computation on Just a PC”. In: *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation*. USENIX Association, 2012, pages 31–46. ISBN: 978-1-931971-96-6. URL: <http://dl.acm.org/citation.cfm?id=2387880.2387884> (cited on page 454).
- [366] Fal labs. *Nijo Cabinet: a straightforward implementation of DBM*. Web Page. Accessed: 02/03/2016. URL: <http://fallabs.com/kyotocabinet/> (cited on page 473).
- [367] Fal labs. *Shijo Cabinet: a handy cache/storage server*. Web Page. Accessed: 02/03/2016. URL: <http://fallabs.com/kyototycoon/> (cited on page 473).
- [368] Gregor von Laszewski and Mike Hategan. “Using Karajan”. In: *Java CoG Kit Karajan-Gridant Workflow Guide*. 2005. URL: <https://pdfs.semanticscholar.org/48e1/31ca4e7a76ca98c43d46975cd79c3dbfd4cb.pdf> (cited on page 521).
- [369] Joe Lennon. *Exploring CouchDB: A document-oriented database for Web applications*. Web page. accessed feb-26-2017. Mar. 2009. URL: <http://www.ibm.com/developerworks.opensource/library/os-couchdb/index.html> (cited on page 474).
- [370] LevelDB. *LevelDB*. Web Page. accessed 2017-02-25. Feb. 2017. URL: leveldb.org (cited on page 476).
- [371] T. Li et al. “ZHT: A Light-Weight Reliable Persistent Dynamic Scalable Zero-Hop Distributed Hash Table”. In: *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*. May 2013, pages 775–787. DOI: [10.1109/IPDPS.2013.110](https://doi.org/10.1109/IPDPS.2013.110). URL: http://datasys.cs.iit.edu/publications/2013_IPDPS13_ZHT.pdf (cited on page 472).
- [372] Weizhong Li. “Introduction to bioActors”. In: *bioKepler Tools and Its Applications*. Edited by bioKepler. 1st Workshop on bioKepler Tools and Its Applications 1. UCSD;SDSC.

- San Diego Supercomputer Center 10100 Hopkins Dr, La Jolla, CA 92093: bioKepler, Sept. 2012, page 31. URL: <http://www.biokepler.org/sites/swat.sdsc.edu.biokepler/files/workshops/2012-09-05/slides/2012-09-05-02-Li.pdf> (cited on page 416).
- [373] *Libvirt virtualization API*. Web Page. URL: <https://libvirt.org/> (cited on page 493).
- [374] Ching Yung Lin. “Graph Computing and Linked Big Data”. In: *8th IEEE/ICSC International Conference on Semantic Computing*. IBM Corporation, 2014, pages 1–63. URL: http://ieee-icsc.org/icsc2014/GraphComputing_IBM_Lin.pdf (cited on page 429).
- [375] LinkedIn Developers. *Getting started with the REST API*. Web page. accessed 17-March-2017. URL: <https://developer.linkedin.com/docs/rest-api> (cited on page 449).
- [376] *Linux containers*. web page. URL: <https://en.wikipedia.org/wiki/LXC> (cited on page 505).
- [377] *Linux containers in use*. web page. URL: <http://www.infoworld.com/article/3072929/linux/containers-101-linux-containers-and-docker-explained.html> (cited on page 505).
- [378] Martin Lisa et al. *Data mining with iPlant*: Paper. Oct. 2014. URL: <https://academic.oup.com/jxb/article/66/1/1/2893405/Data-mining-with-iPlantA-meeting-report-from-the> (cited on page 440).
- [379] Jinjun Chen Lizhe Wang Wei Jie. *Grid Computing: Infrastructure, Service, and Applications*. 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL 33487: Taylor & Francis, 2009. ISBN: 1420067664 (cited on page 510).
- [380] Scott Lowe. *An introduction to Openstack Heat*. webpage. Accessed : 01-15-2017. May 2014. URL: <http://blog.scottlowe.org/2014/05/01/an-introduction-to-openstack-heat/> (cited on page 498).
- [381] Andy Lurie. *Top 47 Log Management Tools*. webpage. Accessed : 02-21-2017. May 2014. URL: <http://blog.profitbricks.com/top-47-log-management-tools> (cited on page 432).
- [382] *LXD an hypervisor for containers*. web page. URL: <https://lists.linuxcontainers.org/pipermail/lxc-devel/2014-November/010817.html> (cited on page 518).
- [383] Tiago Macedo and Fred Oliveira. *Redis Cookbook: Practical Technique for fast Data Manipulation*. Sonoma County, California, United States: O'Reilly Media, Aug. 2011. ISBN: 978-1-4493-0503-1 (cited on page 462).
- [384] *Apache Mahout*. Web Page. URL: <http://mahout.apache.org/> (cited on page 422).
- [385] Jose C Elias Margaret Rouse. *ACID (atomicity, consistency, isolation, and durability)*. Website. July 2006. URL: <http://searchsqlserver.techtarget.com/definition/ACID> (cited on page 467).
- [386] Matthias Marschall. *Chef Infrastructure Automation Cookbook*. Packt Publishing, 2013. ISBN: 9351105164 and 9789351105169 (cited on page 496).
- [387] Martinez-Rubi.O et al. *Taming the beast: Free and open-source massive point cloud web visualization*. 2015. URL: <http://repository.tudelft.nl/islandora/object/uuid:0472e0d1-ec75-465a-840e-fd53d427c177?collection=research> (cited on page 434).
- [388] Karl Matthias and Sean P. Kane. *Docker Up and Running, Shipping Reliable Containers In Production*. 1005 Gravenstein Highway North, Sebastopol, CA 95472.: O'REILLY, 2015. ISBN: 978-1-491-91757-2 (cited on pages 421, 496).
- [389] Morel Matthieu. *S4 0.5.0 Overview*. Web Page. Accessed: 2017-02-24. Feb. 2013. URL: <https://cwiki.apache.org/confluence/display/S4/S4+0.5.0+Overview> (cited on page 448).

- [390] Norman Maurer and Marvin Wolfthal. *Netty in Action*. 1st. Greenwich, CT, USA: Manning Publications, Dec. 23, 2015. 296 pages. ISBN: 1617291471. URL: http://www.ebook.de/de/product/21687528/norman_maurer.netty_in_action.html (cited on page 456).
- [391] M. McLennan and R. Kennell. “HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering”. In: *Computing in Science Engineering* 12.2 (Mar. 2010), pages 48–53. ISSN: 1521-9615. DOI: [10.1109/mcse.2010.41](https://doi.org/10.1109/mcse.2010.41) (cited on page 439).
- [392] Eileen McNulty. *What Is Google Cloud Dataflow?* Web Page. Accessed: 02/03/2016. URL: <http://dataconomy.com/2014/08/google-cloud-dataflow/> (cited on page 446).
- [393] Sergey Melnik et al. “Dremel: Interactive Analysis of Web-Scale Datasets”. In: *Communications of the ACM* 54 (June 2011), pages 114–123. ISSN: 0001-0782. DOI: [10.1145/1953122.1953148](https://doi.org/10.1145/1953122.1953148). URL: <http://cacm.acm.org/magazines/2011/6/108648-dremel-interactive-analysis-of-web-scale-datasets/fulltext> (cited on page 444).
- [394] Xiangrui Meng et al. “MLlib: Machine Learning in Apache Spark”. In: *CoRR* abs/1505.06807 (2015). URL: <http://arxiv.org/abs/1505.06807> (cited on page 422).
- [395] *Mesos Site*. Web Page. URL: <http://mesos.apache.org/> (cited on page 485).
- [396] Microsoft. *Event Hubs*. Web Page. accessed 2017-02-25. Dec. 2016. URL: <https://azure.microsoft.com/en-us/services/event-hubs/> (cited on page 461).
- [397] Microsoft Corp. *Form 10-K*. Web page. Online; accessed 21-jan-2017. July 2016. URL: <https://www.sec.gov/Archives/edgar/data/789019/000119312516662209/d187868d10k.htm> (cited on page 437).
- [398] Microsoft Corp. Web Page. Online; accessed 21-jan-2017. Jan. 2017. URL: <https://azure.microsoft.com/en-us/> (cited on page 437).
- [399] Microsoft Corp. *Designing a Scalable Partitioning Strategy for Azure Table Storage*. Web page. Online; accessed 28-jan-2017. Jan. 2017. URL: <https://docs.microsoft.com/en-us/rest/api/storageservices/fileservices/designing-a-scalable-partitioning-strategy-for-azure-table-storage> (cited on page 480).
- [400] Stuart Miniman. *Cisco Moves Up the Cloud Stack with Intelligent Automation*. webpage. 2011. URL: http://wikibon.org/wiki/v/Cisco_Moves_Up_the_Cloud_Stack_with_Intelligent_Automation (cited on page 499).
- [401] Ross Mistry and Stacia Misner. *Introducing Microsoft SQL Server 2014 Technical Overview*. Microsoft Press, 2014. ISBN: 978-0-7356-8475-1 (cited on page 466).
- [402] *Apache Spark’s MLlib*. Web Page. URL: <http://spark.apache.org/mllib/> (cited on page 422).
- [403] *MLPY documentation*. Web Page. 2012. URL: <http://mlpy.sourceforge.net/docs/3.5/> (cited on page 426).
- [404] AdaptiveComputing. *Moab Cluster Suite*. Web Page. Accessed: 2017-02-24. URL: <https://www.adaptivecomputing.com/products/> (cited on page 488).
- [405] Modine Austin. *Cobbler*. Web Page. accessed 2017-01-30. Feb. 2008. URL: http://www.theregister.co.uk/2008/06/19/red_hat_summit_2008_cobbler/ (cited on page 497).
- [406] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente. “IaaS Cloud Architecture From Virtualized Datacenters to Federated Cloud Infrastructures”. In: *Computer*. Edited by IEEE. Volume 45. 12. IEEE Computer Society. IEEE, 2012, pages 65–72. DOI: [10.1109/MC.2012.76](https://doi.org/10.1109/MC.2012.76). URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6165242&isnumber=6383143> (cited on page 506).
- [407] *MQTT*. Web Page. Accessed: 2017-1-27. URL: <http://mqtt.org/> (cited on page 459).

- [408] *MQTT Floodnet*. Web Page. Accessed: 2017-1-27. URL: <http://mqtt.org/projects/floodnet> (cited on page 459).
- [409] *MR-MPI*. Web Page. 2017. URL: <http://mapreduce.sandia.gov/doc> (cited on page 452).
- [410] Subramanian Muralidhar et al. “f4: Facebook’s warm blob storage system”. In: *Proceedings of the 11th USENIX conference on Operating Systems Design and Implementation*. USENIX Association. 2014, pages 383–398 (cited on page 489).
- [411] Derek G. Murray et al. “Naiad: A Timely Dataflow System”. In: *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*. SOSP ’13. Accessed: 2017-1-27. Farmington, Pennsylvania: ACM, 2013, pages 439–455. ISBN: 978-1-4503-2388-8. DOI: [10.1145/2517349.2522738](https://doi.acm.org/10.1145/2517349.2522738). URL: <http://doi.acm.org/10.1145/2517349.2522738> (cited on page 417).
- [412] Colton Myers. *Learning SaltStack*. 2nd. Packt Publishing, 2015. ISBN: 978-1-78439-460-8, 1784394602. URL: <https://books.google.com/books?id=pVnsrQEACAAJ> (cited on page 497).
- [413] mysql. *MySQL 5.7 Reference Manual, What is MySQL*. Online. URL: <https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html> (cited on page 467).
- [414] *Nagios components*. Web Page. Accessed: 2017-1-11. URL: <https://www.nagios.org/projects/> (cited on page 510).
- [415] NASA/GSFC. *CDF User’s Guide (V3.3.6)*. Version 3.3.6. Accessed: 2017-1-28. NASA/GSFC Space Physics Data Facility. NASA / Goddard Space Flight Center, Greenbelt, Maryland 20771 (U.S.A.), Oct. 2016. URL: <http://spdf.gsfc.nasa.gov/pub/software/cdf/doc/cdf363/cdf363ug.pdf> (cited on page 481).
- [416] NASA/GSFC. *CDF Home Page*. webpage. accessed: 2017-1-28. NASA, Goddard Space Flight Center, 2017. URL: <http://cdf.gsfc.nasa.gov/> (cited on page 481).
- [417] National Instruments. *A Layered Approach to Designing Robot Software*. Web page. accessed 18-mar-2017. Mar. 2017. URL: <http://www.ni.com/white-paper/13929/en/> (cited on page 519).
- [418] Jeremy Nelson. *Mastering Redis*. Packt Publishing, May 2016. ISBN: 978-1783988181 (cited on page 462).
- [419] Neo4j. *Causal Cluster*. Web page. Last Accessed: 2017.02.25. URL: <https://neo4j.com/docs/operations-manual/current/clustering/> (cited on page 477).
- [420] Neo4j. *Chapter 4. Clustering*. Web page. Last Accessed: 2017.02.25. URL: <https://neo4j.com/docs/operations-manual/current/clustering/> (cited on page 477).
- [421] Neo4j. *Highly Available cluster*. Web page. Last Accessed: 2017.02.25. URL: <https://neo4j.com/docs/operations-manual/current/clustering/high-availability/architecture/> (cited on page 477).
- [422] *NetCDF: Introduction and Overview*. Web Page. URL: https://www.unidata.ucar.edu/software/netcdf/docs_rc/ (visited on 02/13/2017) (cited on page 481).
- [423] Netsyslab. *TOTEM*. Webpage. URL: <http://netsyslab.ece.ubc.ca/wiki/index.php/Totem> (cited on page 455).
- [424] *Netty Site*. Web Page. URL: <http://netty.io/> (cited on page 456).
- [425] *Nimbus*. Web Page. URL: <http://www.nimbusproject.org/doc/nimbus/platform/> (cited on page 506).
- [426] *Nimbus Wiki*. Web Page. URL: [https://en.wikipedia.org/wiki/Nimbus_\(cloud_computing\)](https://en.wikipedia.org/wiki/Nimbus_(cloud_computing)) (cited on page 506).
- [427] Ninefold. *Ninefold news*. Web Page. Accessed: 2017-2-27. Nov. 2015. URL: <http://ninefold.com/news/> (cited on page 437).

- [428] Nokia Corp. Web page. accessed 25-feb-2017. Feb. 2017. URL: http://www.nokia.com/en_int (cited on page 453).
- [429] Jordan Novet. *dotCloud*. Web Page. Accessed: 2017-1-31. Jan. 2016. URL: <http://venturebeat.com/2016/01/22/dotcloud-the-cloud-service-that-gave-birth-to-docker-is-shutting-down-on-february-29/> (cited on page 439).
- [430] NSA 'NiFi' Big Data Automation Project Out In The Open. Web Page. URL: <http://www.forbes.com/sites/adrianbridgwater/2015/07/21/nsa-nifi-big-data-automation-project-out-in-the-open/#6b37cac55d9a> (cited on page 420).
- [431] Daniel Nurmi et al. "The eucalyptus open-source cloud-computing system". In: *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*. IEEE Computer Society. 2009, pages 124–131 (cited on page 506).
- [432] Christian Nutt. *Gamasutra - Amazon launches new, free, high-quality game engine: Lumberyard*. URL: http://www.gamasutra.com/view/news/265425/Amazon_launches_new_free_highquality_game_engine_Lumberyard.php (visited on 02/27/2017) (cited on page 447).
- [433] NWB. Web Page. 2017. URL: <http://nwb.cns.iu.edu/about.html> (cited on page 431).
- [434] *ODE Website*. Web Page. Accessed: 2017-2-11. URL: <http://ode.apache.org/> (cited on page 413).
- [435] *Apache ODE*. Web Page. Accessed: 2017-2-11. URL: https://en.wikipedia.org/wiki/Apache_ODE (cited on page 413).
- [436] John O'Hara. "Toward a commodity enterprise middleware". In: *Queue* 5.4 (2007), pages 48–55 (cited on page 457).
- [437] Carl W Olofson. *The Analytic-Transactional Data Platform: Enabling the Real-Time Enterprise (IDC)*. Technical report. Accessed: 2017-1-17. International Data Corporation (IDC), Dec. 2014. URL: <http://www.sap.com/documents/2016/08/3c4e546e-817c-0010-82c7-eda71af511fa.html> (cited on page 443).
- [438] Jean-Baptiste Onofre. *The Future of Apache Beam, Now a Top-Level Apache Software Foundation Project*. Blog. accessed 25-feb-2017. Jan. 2017. URL: <https://www.talend.com/blog/2017/01/13/future-apache-beam-now-top-level-apache-software-foundation-project/> (cited on page 515).
- [439] *Apache OODT*. Web Page. Accessed: 2017-2-25. URL: <http://oodt.apache.org/> (cited on page 517).
- [440] *Getting Started*. Web Page. Accessed: 2017-2-25. URL: <http://oodt.apache.org/documentation.htm> (cited on page 517).
- [441] Apache. *About OODT*. Web Page. Accessed: 2017-02-12. URL: <http://oodt.apache.org/> (cited on page 439).
- [442] *Oozie - Apache Oozie Workflow Scheduler for Hadoop*. Web Page. URL: <http://oozie.apache.org/> (visited on 02/13/2017) (cited on page 417).
- [443] *Open Database Connectivity*. Web Page. Accessed: 2017-1-30. URL: https://en.wikipedia.org/wiki/Open_Database_Connectivity (cited on page 465).
- [444] Open Source Robotics Foundation. *About ROS*. Web page. accessed 16-mar-2017. Mar. 2017. URL: <http://www.ros.org/about-ros/> (cited on page 519).
- [445] *OpenID*. website. URL: <http://openid.net/> (cited on page 512).
- [446] *OpenID*. webpage. URL: <https://en.wikipedia.org/wiki/OpenID> (cited on page 512).
- [447] *OpenNebula Wiki*. URL: <https://opennebula.org/about/technology/> (cited on page 506).
- [448] OpenRefine. *OpenRefine*. Web Page. accessed 2017-01-13. Feb. 2016. URL: <http://openrefine.org/> (cited on page 517).

- [449] OpenSFS, EOFS. *Welcome to the official home of the Lustre filesystem*. Web Page. accessed 2017-01-28. Dec. 2016. URL: <http://lustre.org/> (cited on page 491).
- [450] OpenStack. Web Page. Accessed:1/16/2017. URL: <https://www.openstack.org/> (cited on page 499).
- [451] Keystone Architecture. Web Page. Accessed: 2017-2-12. URL: <http://docs.openstack.org/developer/keystone/architecture.html> (cited on page 511).
- [452] OpenVZ. en. Web Page. Page Version ID: 764498783. Feb. 2017. URL: <https://en.wikipedia.org/w/index.php?title=OpenVZ&oldid=764498783> (visited on 02/13/2017) (cited on page 504).
- [453] Wikipedia. *Oracle Grid Engine - Wikipedia*. webpage. Accessed : 02-22-2017. Feb. 2017. URL: https://en.wikipedia.org/wiki/Oracle_Grid_Engine (cited on page 487).
- [454] Oracle Database - Wikipedia. Web Page. URL: https://en.wikipedia.org/wiki/Oracle_Database (visited on 02/13/2017) (cited on page 466).
- [455] Oracle Labs PGX. Web Page. Accessed: 2017-02-07. URL: https://docs.oracle.com/cd/E56133_01/2.3.1/index.html (cited on page 428).
- [456] Oracle Website. Web Page. Accessed: 2017-2-11. URL: <http://www.oracle.com/technetwork/database/database-technologies/berkeleydb> (cited on page 472).
- [457] Apache org. Apache Tomcat. Web Page. Accessed: 02-24-2017. URL: <http://tomcat.apache.org/> (cited on page 515).
- [458] G. Ostrouchov et al. Web Page. 2012. URL: <http://r-pbd.org/> (cited on page 423).
- [459] Overleaf. URL: <https://www.overleaf.com> (cited on page 85).
- [460] Overview. Online. The contact information listed on the webpage was for Yogesh Simmhan. URL: <http://dream-lab.cds.iisc.ac.in/about/overview/> (cited on page 430).
- [461] Parallel Graph Analytics (PGX). Web Page. Accessed: 2017-02-11. URL: <http://www.oracle.com/technetwork/oracle-labs/parallel-graph-analytics/overview/index.html> (cited on page 429).
- [462] Parasol. Webpage. URL: <https://parasol.tamu.edu/research.php> (cited on page 430).
- [463] James J. (Jong Hyuk) Park et al. *Advanced Multimedia and Ubiquitous Engineering*. Edited by James J. (Jong Hyuk) Park et al. accessed 2017-02-13. Springer, 2016 (cited on page 453).
- [464] Moshar Pasumansky. *Inside Capacitor, BigQuery's next-generation columnar storage format*. Blog. Accessed: 2017-2-23. Apr. 2016. URL: <https://cloud.google.com/blog/big-data/2016/04/inside-capacitor-bigquerys-next-generation-columnar-storage-format> (cited on page 445).
- [465] PBS. Web Page. URL: <http://www.pbspro.org/> (cited on page 487).
- [466] Pegasus. Web Page. Accessed:2/3/2017. URL: <https://pegasus.isi.edu/> (cited on page 414).
- [467] Pentaho. webpage. URL: <https://en.wikipedia.org/wiki/Pentaho> (cited on page 421).
- [468] Francisco Perez-Sorrosal et al. *Omid's First Step in the Apache Community*. Web Page. accessed 2017-02-25. Sept. 2016. URL: <https://yahoeng.tumblr.com/post/151015726181/omids-first-step-in-the-apache-community> (cited on page 517).
- [469] Sebastian Peyrott. *API Gateway. An Introduction to Microservices, Part 2*. Website. Blog Post. Sept. 2015. URL: <https://auth0.com/blog/an-introduction-to-microservices-part-2-API-gateway/> (cited on page 519).
- [470] Rob Pike et al. "Interpreting the Data: Parallel Analysis with Sawzall". In: *Scientific Programming Journal* 13.4 (Oct. 2005), pages 277–298. ISSN: 1058-9244. DOI: [10.1155/1058-9244-13-4](https://doi.org/10.1155/1058-9244-13-4)

- 2005/962135. URL: <https://research.google.com/archive/sawzall-sciprog.pdf> (cited on page 446).
- [471] Keshav Pingali et al. “The Tao of Parallelism in Algorithms”. In: *The Tao of Parallelism in Algorithms*. Accessed: 2017-2-27. June 2011, pages 1–14. URL: <http://iss.ices.utexas.edu/Publications/Papers/pingali11.pdf> (cited on page 454).
- [472] Pivotal. Web Page. URL: <https://run.pivotal.io/features/> (cited on page 437).
- [473] *Pivotal Greenplum. The Open Source Massively Parallel Data Warehouse.* Webpage. URL: <https://pivotal.io/pivotal-greenplum> (cited on page 469).
- [474] *Pivotal Greenplum Database.* Webpage. URL: https://en.wikipedia.org/wiki/Pivotal_Greenplum_Database (cited on page 469).
- [475] *Pivotal HAWQ.* URL: <http://hawq.incubator.apache.org/> (cited on page 444).
- [476] *Pivotal HDB.* URL: <https://pivotal.io/pivotal-hdb> (cited on page 444).
- [477] *PLASMA README.* Web Page. URL: <http://icl.cs.utk.edu/projectsfiles/plasma/html/README.html> (visited on 02/13/2017) (cited on page 424).
- [478] Moritz Plassnig. *Heroku-style Application Deployments with Docker - DZone Cloud.* Web Page. Accessed: 2017-1-17. Nov. 2015. URL: <https://dzone.com/articles/heroku-style-application-deployments-with-docker> (cited on page 501).
- [479] PmWiki. *GFFS – Global Federated File System.* Web Page. accessed 2017-01-28. Jan. 2012. URL: <http://genesis2.virginia.edu/wiki/Main/GFFS> (cited on page 491).
- [480] Ian Pointer. *Apache Beam want to be uber-API for big data.* Web page. accessed 25-feb-2017. Apr. 2016. URL: <http://www.infoworld.com/article/3056172/application-development/apache-beam-wants-to-be-uber-api-for-big-data.html> (cited on page 515).
- [481] *PolyBase.* Web Page. URL: www.microsoft.com/en-us/library/mt143171.aspx (cited on pages 443, 444).
- [482] Florin Pop, Joanna Kolodziej, and Beniamino DiMartino. “Resource Management for Big Data Platforms”. In: Springer Nature, 2016, pages 49–50 (cited on page 419).
- [483] Florin Pop, Joanna Kolodziej, and Beniamino DiMartino. “Resource Management for Big Data Platforms”. In: Springer Nature, 2016, pages 50–51 (cited on page 520).
- [484] *Potree.* URL: <http://potree.org/wp/about/> (cited on page 434).
- [485] Herbert Pötzl. *Linux-Vserver.* Web Page. URL: www.linux-vserver.org (cited on page 505).
- [486] Vignesh Prajapati. *Big data analytics with R and Hadoop.* Packt Publishing, 2013. ISBN: 9781782163282 (cited on page 423).
- [487] Prashanth. *Facebook Tupperware.* Blog. Accessed: 2/18/2017. Dec. 2015. URL: <http://blog.cspp.in/index.php/2015/12/17/facebook-tupperware/> (cited on page 499).
- [488] *Presto.* Web Page. Accessed: 2017-1-24. URL: <https://prestodb.io/> (cited on page 444).
- [489] *Project Jupyter.* Web page. URL: <http://www.jupyter.org> (visited on 02/27/2017) (cited on page 416).
- [490] UltraScan Project. *UltraScan Analysis Software.* Web page. Accessed: 2017-2-22. Apr. 2015. URL: <http://ultrascan.uthscsa.edu> (cited on page 414).
- [491] *Protocol Buffer.* Web Page. Sept. 2016. URL: <https://developers.google.com/protocol-buffers/> (cited on page 514).
- [492] Roshan Punnoose, Adina Crainiceanu, and David Rapp. “Rya: A Scalable RDF Triple Store for the Clouds”. In: *Proceedings of the 1st International Workshop on Cloud Intelligence. Cloud-I ’12.* Istanbul, Turkey: ACM, 2012, 4:1–4:8. ISBN: 978-1-4503-1596-8. DOI: [10](https://doi.org/10.1145/2387000.2387004).

- 1145/2347673.2347677. URL: <http://doi.acm.org/10.1145/2347673.2347677> (cited on page 477).
- [493] *Puppet FAQ*. Web Page. URL: <https://puppet.com/product/faq> (cited on page 496).
- [494] *Puppet software*. Web Page. URL: [https://en.wikipedia.org/wiki/Puppet_\(software\)](https://en.wikipedia.org/wiki/Puppet_(software)) (cited on page 496).
- [495] PuppetLabsRazor. *Home.PuppetLabs/Razor Wiki*. Web Page. Accessed: 2017-02-04. URL: <https://github.com/puppetlabs/Razor/wiki> (cited on page 498).
- [496] PuppetLabsRazor. *Introducing Razor; a next Generation provisioning Solution-Puppet*. Web Page. Accessed: 2017-02-04. URL: <https://puppet.com/blog/introducing-razor-a-next-generation-provisioning-solution> (cited on page 498).
- [497] Qemu. *QEMU*. WebPage. Feb. 2017. URL: http://wiki.qemu-project.org/index.php/Main_Page (cited on page 504).
- [498] Quora. *LinkedIn*. Web page. accessed 18-March-2017. URL: <https://www.quora.com/What-is-LinkedIn-s-database-architecture-like> (cited on page 449).
- [499] R: What is R? Web Page. Accessed: 2017-1-26. URL: <https://www.r-project.org/about.html> (cited on page 423).
- [500] RabbitMQ, components. Web Page. Accessed: 2017-01-19. URL: <https://www.rabbitmq.com/> (cited on page 457).
- [501] Tilmann Rabl et al. “Solving Big Data Challenges for Enterprise Application Performance Management”. In: *Proc. VLDB Endow.* 5.12 (Aug. 2012), pages 1724–1735. ISSN: 2150-8097. DOI: [10.14778/2367502.2367512](https://doi.org/10.14778/2367502.2367512). URL: <https://arxiv.org/pdf/1208.4167.pdf> (cited on page 471).
- [502] Dan Radez. *OpenStack Essentials*. Packt Publishing Ltd., May 26, 2015. 182 pages. ISBN: 978-1-78398-708-5. URL: <http://ebook.konfigurasi.net/Openstack/OpenStack%5C%20Essentials.pdf> (cited on page 490).
- [503] Ioan Raicu et al. “Falkon: a Fast and Light-weight tasK executiON framework”. In: *ACM SC07*. 2007 (cited on page 488).
- [504] Siddharth Rao. *DREAM:Lab – Democratising Computing through the Cloud*. Online. Mar. 2016. URL: <http://iisc.researchmedia.center/article/dreamlab-%E2%80%93-democratising-computing-through-cloud> (cited on page 430).
- [505] *Rasdaman raster array database*. Web Page. Accessed: 02-23-2017. URL: <http://www.rasdaman.org/> (cited on page 468).
- [506] Syed Ali Raza. *Neo4j*. Presentation/ Slides. Last Accessed: 2017.02.25. URL: <https://www.slideshare.net/aliraza995/neo4j-graph-storage-27104408> (cited on page 477).
- [507] *RCFile Cat*. Web Page. Accessed: 2017-02-02. URL: <https://cwiki.apache.org/confluence/display/Hive/RCFileCat> (cited on page 482).
- [508] *RCFileCat - Apache Hive*. Web Page. Accessed: 2017-1-27. URL: <https://cwiki.apache.org/confluence/display/Hive/RCFileCat> (cited on page 482).
- [509] Amazon RDS. *What is Amazon RDS*. Web Page. Accessed: 2017-02-04. URL: <http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html#Welcome.Concepts> (cited on page 469).
- [510] Amazon RDS. *What is Amazon RDS?Definition*. Web Page. Accessed: 2017-02-04. URL: <http://searchaws.techtarget.com/definition/Amazon-Relational-Database-Service-RDS> (cited on page 469).
- [511] Red Hat, Inc. *developers-openshift components*. Web Page. Accessed: 2017-1-26. URL: <https://developers.openshift.com/> (cited on page 435).
- [512] Red Hat, Inc. *openshift components*. Web Page. Accessed: 2017-1-26. URL: <https://www.openshift.org/> (cited on page 435).

- [513] Red Hat, Inc. *openshift-blog components*. Web Page. Accessed: 2017-1-27. URL: <https://blog.openshift.com/getting-started-with-openshift-origin-the-open-source-platform-as-a-service-paas/> (cited on page 435).
- [514] Red Hat, Inc. *paas-openshift components*. Web Page. Accessed: 2017-1-25. URL: <https://blog.openshift.com/announcing-openshift-origin-the-open-source-platform-as-a-service-paas/> (cited on page 435).
- [515] Red Hat, Inc. *Ansible Documentation*. Web Page. Accessed: 2017-02-12. Feb. 2015. URL: <https://docs.ansible.com/ansible/index.html> (cited on page 497).
- [516] Red Hat, Inc. *Ceph Homepage-Ceph*. Web Page. Accessed: 2017-1-26. Red Hat, Inc., 2017. URL: <https://ceph.com/> (cited on page 490).
- [517] Red Hat, Inc. *Ceph Filesystem - Ceph Documentation*. Web Page. Accessed: 2017-1-24. Red Hat, Inc. URL: <http://docs.ceph.com/docs/master/cephfs/> (cited on page 490).
- [518] Red Hat, Inc. *Ceph Architecture-Ceph Documentation*. Web Page. Accessed: 2017-1-24. 2017. URL: <http://docs.ceph.com/docs/master/architecture/> (cited on page 490).
- [519] Bloor Robin and Jozwiak Rebecca. *THE NATURE OF GRAPH DATA*. Technical report. Austin, TX 78720|512-524-3689: The Bloor Group, 2017. URL: <http://web.cray.com/bloor-graph-data> (cited on page 478).
- [520] *Rocks*. Web Page. 2017. URL: <https://www.rockscluster.org> (cited on page 499).
- [521] Ryan Rosario. *Exciting Tools for Big Data: S4, Sawzall and mrjob!* Web page. Online; accessed 30-jan-2017. Nov. 2010. URL: <http://www.bytemining.com/2010/11/exciting-tools-for-big-data-s4-sawzall-and-mrjob/> (cited on page 446).
- [522] Apache Rya. *Apache Rya*. Online. URL: <https://rya.apache.org/> (cited on page 477).
- [523] *S4: S4 0.6.0 overview*. Web Page. Accessed: 2017-02-24. URL: <http://incubator.apache.org/s4/doc/0.6.0/overview> (cited on page 447).
- [524] *Sahara*. Web Page. Accessed: 1/16/2017. URL: <http://docs.openstack.org/developer/sahara/> (cited on page 499).
- [525] *saltstack components*. webpage. accessed: 2017-2-4. URL: <https://saltstack.com/> (cited on page 497).
- [526] *Samza*. Web Page. URL: <http://samza.apache.org/> (visited on 02/13/2017) (cited on page 448).
- [527] Abhijeet Sandil. *SSO Strategy: Authentication (SAML) –vs– Authorization (OAuth)*. Web Page. Accessed: 2017-02-04. URL: <https://www.linkedin.com/pulse/sso-strategy-authentication-vs-authorization-saml-oauth-sandil> (cited on page 512).
- [528] Lee Sangkeun, Rangan Sukumar Sreenivas, and Lim Seung-Hwan. “Graph mining meets the semantic web.” In: *2015 31st IEEE International Conference on Data Engineering Workshops*. (Seoul, South Korea). Edited by Johannes Gehrke, Wolfgang Lehner, and Kyuseok Shim. IEEE. Apr. 2015, pages 53–58. DOI: [10.1109/ICDEW.2015.7129544](https://doi.org/10.1109/ICDEW.2015.7129544). URL: https://www.researchgate.net/profile/Sreenivas_Rangan_Sukumar/publication/273755615_Graph_Mining_Meets_the_Semantic_Web/links/550a27cb0cf20ed529e26260.pdf (cited on page 478).
- [529] netlib. *ScalAPACK Users Guide*. Web Page. Accessed: 2017-02-12. URL: <http://www.netlib.org/scalapack/slug/> (cited on page 424).
- [530] *Scaling Apache Giraph to a trillion edges*. web page. URL: <https://www.facebook.com/notes/facebook-engineering/scaling-apache-giraph-to-a-trillion-edges/10151617006153920> (cited on page 453).
- [531] Tom Schaul et al. “PyBrain”. In: *Journal of Machine Learning Research* 11 (Mar. 2010). Edited by Soeren Sonnenburg, pages 743–746. ISSN: 1532-4435. URL: <http://dl.acm.org/citation.cfm?id=1756006.1756030> (cited on page 426).

- [532] SchedMd. *Slurm website*. Web Page. Accessed: 2017-2-27. Mar. 2013. URL: <https://slurm.schedmd.com/overview.html> (cited on page 488).
- [533] SchedMd. *Slurm Supported Platforms*. Web Page. Accessed: 2017-2-27. Apr. 2015. URL: <https://slurm.schedmd.com/platforms.html> (cited on page 488).
- [534] scikit-Learn. webpage. URL: <https://en.wikipedia.org/wiki/Scikit-learn> (cited on page 426).
- [535] Sean Lynch. *Why Membase Uses Erlang*. web page. accessed 2017-01-29. Oct. 2010. URL: <https://blog.couchbase.com/why-membase-uses-erlang> (cited on page 475).
- [536] Adam Seligman. *Apache Phoenix - A Small Step for Big Data*. Web page. Online; accessed 25-jan-2017. May 2014. URL: <https://developer.salesforce.com/blogs/developer-relations/2014/05/apache-phoenix-small-step-big-data.html> (cited on page 442).
- [537] James Serra. *What is Azure Data Factory?* Web Page. Accessed: 02/03/2016. URL: <http://www.jamesserra.com/archive/2014/11/what-is-azure-data-factory/> (cited on page 419).
- [538] Vinay Jayarama Setty. “Publish/subscribe for large-scale social interaction: Design, analysis and resource provisioning”. Accessed:2017-1-29. PhD thesis. Faculty of Mathematics and Natural Sciences, University of Oslo: Department of Informatics,UNIVERSITY OF OSLO, Jan. 2015. URL: <https://www.duo.uio.no/bitstream/handle/10852/43117/1595-Setty-DUO-Thesis.pdf?sequence=1&isAllowed=y> (cited on page 455).
- [539] ShareLaTex. URL: <https://www.sharelatex.com> (cited on page 85).
- [540] Caro Caserio Sharon Lo Sreedhar Pelluru. *Introduction to Azure Data Factory Service, a data integration service in the cloud*. Web Page. Accessed: 02/03/2016. URL: <https://docs.microsoft.com/en-us/azure/data-factory/data-factory-introduction> (cited on page 420).
- [541] Xuanhua Shi et al. “GIRAFFE: A Scalable Distributed Coordination Service for Large-scale Systems”. In: *GIRAFFE: A Scalable Distributed Coordination Service for Large-scale Systems*. Accessed: 2017-2-27. Sept. 2014, pages 1–10. URL: <http://www.mcs.anl.gov/papers/P5157-0714.pdf> (cited on page 513).
- [542] Sumit Gupta Shilpi Saxena. *Real-Time Big Data Analytics*. 1st. 35 Livery Street, Birmingham B3 2PB, UK: Packt Publishing, 2016. ISBN: 9781784391409 (cited on page 449).
- [543] Abdul Ghaffar Shoro and Tariq Rahim Soomro. “Big data analysis: Apache spark perspective”. In: *Global Journal of Computer Science and Technology* 15.1 (2015). Accessed: 2017-1-21. ISSN: 0975-4172. URL: <http://www.computerresearch.org/index.php/computer/article/view/1137/1124> (cited on pages 422, 452).
- [544] Pallickara Shrideep et al. *granules*. Project. July 2016. URL: <http://granules.cs.colostate.edu/> (cited on page 448).
- [545] Julian Shun and Guy E. Blelloch. “Ligra: A Lightweight Graph Processing Framework for Shared Memory”. In: *ACM SIGPLAN Notices - PPoPP '13*. Pittsburgh, Pennsylvania, USA: ACM New York, NY, USA, 2013. ISBN: 978-1-4503-1922-5. DOI: [10.1145/2517327.2442530](https://doi.org/10.1145/2517327.2442530). URL: <http://dl.acm.org/citation.cfm?id=2442530> (cited on page 454).
- [546] Jeff Shute et al. “F1: The Fault-tolerant Distributed RDBMS Supporting Google’s Ad Business”. In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. SIGMOD ’12. Accessed : 2017-02-15. Scottsdale, Arizona, USA: ACM, 2012, pages 777–778. ISBN: 978-1-4503-1247-9. DOI: [10.1145/2213836.2213954](https://doi.org/10.1145/2213836.2213954). URL: <http://doi.acm.org/10.1145/2213836.2213954> (cited on page 470).
- [547] Jeff Shute et al. “F1: A Distributed SQL Database That Scales”. In: *Proc. VLDB Endow.* 6.11 (Aug. 2013). Accessed:2017-02-15, pages 1068–1079. ISSN: 2150-8097. DOI: [10.1145/2213836.2213954](https://doi.org/10.1145/2213836.2213954).

- 14778/2536222.2536232. URL: <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/41344.pdf> (cited on page 470).
- [548] Abraham Silberschatz et al. *Operating system concepts*. Volume 4. Addison-wesley Reading, 1998 (cited on page 461).
- [549] SINTEF. *CloudML*. Web page. accessed 15-March-2017. 2013. URL: <http://cloudml.org/> (cited on pages 501, 502).
- [550] SINTEF Git. *CloudML Wiki*. Web page. accessed 16-March-2017. Last updated in 2 Oct 2015. URL: <https://github.com/SINTEF-9012/cloudml/wiki> (cited on page 502).
- [551] Mark Slee, Aditya Agarwal, and Marc Kwiatkowski. “Thrift: Scalable Cross-Language Services Implementation”. In: *Thrift* (Apr. 1, 2007). URL: <https://thrift.apache.org/static/files/thrift-20070401.pdf> (cited on page 513).
- [552] Slideshare. *Apache UIMA Introduction*. Web Page. Accessed: 02/03/2016. URL: <http://www.slideshare.net/teofili/apache-uima-introduction> (cited on page 465).
- [553] SLURM. Web Page. URL: <https://slurm.schedmd.com/> (cited on page 488).
- [554] James Smith. “Article title”. In: 14.6 (Mar. 2013), pages 1–8 (cited on page 140).
- [555] John Smith. *Book title*. 1st edition. Volume 3. 2. City: Publisher, Jan. 2012, pages 123–200 (cited on page 140).
- [556] Ken Smith. *Azure: What to Use, What to Avoid*. Web page. Online; accessed 28-jan-2017. Oct. 2014. URL: <http://blog.wouldbethelogian.com/2014/10/azure-what-to-use-what-to-avoid.html> (cited on page 480).
- [557] Russell Smith. *What are the advantages and disadvantages of using MongoDB vs CouchDB vs Cassandra vs Redis?* Web page. Online; accessed 29-jan-2017. Nov. 2015. URL: <https://www.quora.com/What-are-the-advantages-and-disadvantages-of-using-MongoDB-vs-CouchDB-vs-Cassandra-vs-Redis> (cited on page 475).
- [558] Amazon SNS. Web Page. Accessed: 2017-02-02. URL: <https://aws.amazon.com/sns/> (cited on page 460).
- [559] Amazon SNS Blog. Web Page. Accessed: 2017-02-02. URL: <https://aws.amazon.com/blogs/aws/introducing-the-amazon-simple-notification-service/> (cited on page 460).
- [560] Amazon SNS FAQs. Web Page. Accessed: 2017-02-02. URL: <https://aws.amazon.com/sns/faqs/> (cited on page 460).
- [561] Software » OpenStack Open Source Cloud Computing Software. Web Page. Online; accessed 23-Feb-2017. URL: <https://www.openstack.org/software/> (cited on page 505).
- [562] Borja Sotomayor and Lisa Childers. *Globus® Toolkit 4: Programming Java Services*. Morgan Kaufmann, 2006 (cited on page 488).
- [563] Apache Software Foundation. *Spark Programming Guide - Spark 2.1.0 Documentation*. Web Page. Accessed: 04-09-2017. URL: <http://spark.apache.org/docs/latest/programming-guide.html#resilient-distributed-datasets-rdds> (cited on page 451).
- [564] Evan R Sparks et al. “KeystoneML: Optimizing Pipelines for Large-Scale Advanced Analytics”. In: *arXiv preprint arXiv:1610.09451* (2016) (cited on page 422).
- [565] Evan Sparks and Shivaram Venkataraman. *Building Large Scale Machine Learning Applications with Pipelines-(Ev...)* URL: <https://www.slideshare.net/SparkSummit/evan-sparks> (visited on 02/27/2017) (cited on page 422).
- [566] E. Sparks et al. “MLI: An API for Distributed Machine Learning”. In: *MLI: An API for Distributed Machine Learning*. IEEE 13th International Conference on Data Mining. 2013 (cited on page 422).

- [567] Apache Software Foundation. *Spark Streaming - Spark 2.1.0 Documentation*. Web Page. Accessed: 04-09-2017. URL: <http://spark.apache.org/docs/latest/streaming-programming-guide.html> (cited on page 451).
- [568] Splunk, Inc. *Splunk Enterprise Overview*. Web Page. accessed: 2017-02-18. Feb. 2015. URL: <http://docs.splunk.com/Documentation/Splunk/6.5.2/Overview/AboutSplunkEnterprise> (cited on page 433).
- [569] SQLite. *About SQLite*. Website. URL: <https://www.sqlite.org/about.html> (cited on page 467).
- [570] SQLite. *Appropriate Uses For SQLite*. Website. URL: <https://www.sqlite.org/whentouse.html> (cited on page 467).
- [571] *SQL Server Wiki*. Web Page. URL: https://en.wikipedia.org/wiki/Microsoft_SQL_Server (cited on page 466).
- [572] *SSH - Wikipedia*. Web Page. Accessed: 2017-01-26. URL: https://en.wikipedia.org/wiki/Secure_Shell (cited on page 483).
- [573] *OpenSSH - Wikipedia*. Web Page. Accessed: 2017-01-26. URL: <https://en.wikipedia.org/wiki/OpenSSH> (cited on page 484).
- [574] *Difference between Application Manager and Application Master in YARN?* StackOverflow. 2015. URL: <http://stackoverflow.com/questions/30967247/difference-between-application-manager-and-application-master-in-yarn> (cited on page 485).
- [575] Michael Stonebraker. “SciDB: An Open-Source DBMS for Scientific Data”. In: *ERCIM News* 89 (Apr. 2012), page 56. URL: <https://ercim-news.ercim.eu/en89/special/scidb-an-open-source-dbms-for-scientific-data> (cited on page 468).
- [576] *Apache Storm - Concepts*. webpage. Accessed :01-22-2017. URL: <http://storm.apache.org/releases/1.0.2/Concepts.html> (cited on page 447).
- [577] Dan Sullivan. *Paas Provider Comparison Guide: Engine Yard - Orchestration and Management*. Article. Accessed: 02-26-2017. July 2013. URL: http://www.tomsitpro.com/articles/paas-engine-yard-amazon-elastic-cloud-computing_2-578.html (cited on page 438).
- [578] Nick Sullivan. *Kyoto Tycoon Secure Replication*. Web Page. Accessed: 02/03/2016. URL: <https://blog.cloudflare.com/kyoto-tycoon-secure-replication/> (cited on page 473).
- [579] Bruce Synder and Rob Davies. *ActiveMQ in Action*. 1st. Manning publications, 2013. ISBN: 1933988940, 9781933988948 (cited on page 456).
- [580] Istvan Szegedi. *Apache Phoenix - An SQL Driver for HBase*. Web page. Online; accessed 23-jan-2017. May 2014. URL: <https://bighadoop.wordpress.com/2014/05/17/apache-phoenix-an-sql-driver-for-hbase/> (cited on page 442).
- [581] *Tableau Tutorial*. Web Page. Accessed: 2017-2-10. URL: <https://casci.umd.edu/wp-content/uploads/2013/12/Tableau-Tutorial.pdf> (cited on page 433).
- [582] *Tableau Technology*. Web Page. Accessed: 2017-2-10. URL: <https://www.tableau.com/products/technology> (cited on page 433).
- [583] A. Talwalkar et al. “MLbase: A Distributed Machine Learning Wrapper”. In: *MLbase: A Distributed Machine Learning Wrapper*. Big Learning Workshop at NIPS. 2012 (cited on page 422).
- [584] *Taverna*. Web Page. URL: <https://taverna.incubator.apache.org/introduction/> (cited on page 415).
- [585] Illinois Institute of Technology Department of Computer Science. *ZHT: a zero-hop distributed hashtable*. Online. URL: <http://datasys.cs.iit.edu/projects/ZHT/> (cited on page 472).

- [586] Techopedia. *Amazon Web Services (AWS)*. Web Page. URL: <https://www.techopedia.com/definition/26426/amazon-web-services-aws> (cited on page 508).
- [587] TechTarget. *Lightweight Directory Access Protocol*. Web Page. accessed 2017-01-30. Feb. 2017. URL: <http://searchmobilecomputing.techtarget.com/definition/LDAP> (cited on page 511).
- [588] *TensorFlow*. Web Page. Accessed: 2017-1-24. URL: <https://www.tensorflow.org> (cited on page 434).
- [589] *Terraform components*. Web Page. Accessed: 2017-02-12. URL: <https://www.terraform.io/intro/index.html> (cited on page 502).
- [590] *Tez*. Web Page. 2017. URL: <https://hortonworks.com/apache/tez> (cited on page 417).
- [591] The Apache Software Foundation. Web Page. URL: <http://sqoop.apache.org/> (cited on page 484).
- [592] The Apache Software Foundation. Web Page. URL: <http://ant.apache.org/> (cited on page 518).
- [593] The Apache Software Foundation. *Apache Flume*. web page. accessed 2017-02-06. URL: <https://flume.apache.org/index.html> (cited on page 484).
- [594] The Apache Software Foundation. *Getting Started with Derby*. Web Page. accessed: 2017-02-18. Sept. 2015. URL: <https://db.apache.org/derby/docs/10.12/getstart/index.html> (cited on page 469).
- [595] The Apache Software Foundation. *Apache Derby*. Web Page. accessed: 2017-02-18. Oct. 2016. URL: <https://db.apache.org/derby/> (cited on page 469).
- [596] The Apache Software Foundation. *Apache Derby Project Charter*. Web page. accessed: 2017-02-18. Sept. 2016. URL: https://db.apache.org/derby/derby_charter.html (cited on page 469).
- [597] The Apache Software Foundation. *Spark SQL*. Web Page. accessed 2017-02-19. Feb. 2016. URL: <http://spark.apache.org/sql/> (cited on page 470).
- [598] The Apache Software Foundation. *Apache CloudStack*. Web Page. Accessed: 2017-02-11. Feb. 2017. URL: <https://cloudstack.apache.org/> (cited on page 507).
- [599] The Disco Project. *What is Disco*. Web page. accessed 25-feb-2017. Feb. 2017. URL: <http://disco.readthedocs.io/en/develop/intro.html> (cited on page 453).
- [600] The Disco Project. *Why Not Hadoop?* Web page. accessed 25-feb-2017. Feb. 2017. URL: <http://disco.readthedocs.io/en/develop/faq.html#why-not-hadoop> (cited on page 453).
- [601] *The Eucalyptus Open-Source Private Cloud*. Web Page. Accessed: 2017-02-11. URL: <http://www.cloudbook.net/resources/stories/the-eucalyptus-open-source-private-cloud> (cited on page 506).
- [602] *The Jupyter notebook — Jupyter Notebook 5.0.0.dev documentation*. Web page. URL: <https://jupyter-notebook.readthedocs.io/en/latest/index.html> (visited on 02/27/2017) (cited on page 416).
- [603] *The LXD container hypervisor*. web page. URL: <https://www.ubuntu.com/containers/lxd> (cited on page 518).
- [604] The Open MPI Project. *Open MPI: Open Source High Performance Computing*. Web Page. Accessed: 2017-2-22. Feb. 2017. URL: <https://www.open-mpi.org> (cited on page 515).
- [605] The PostgreSQL Global Development Group. *PostgreSQL 9.5: UPSERT, Row Level Security, and Big Data*. Web Page. Accessed: 2017-4-10. URL: <https://www.postgresql.org/about/> (cited on page 467).

- [606] The PostgreSQL Global Development Group. *PostgreSQL 9.5: UPSERT, Row Level Security, and Big Data*. Web Page. Accessed: 2017-4-10. URL: <https://www.postgresql.org/about/news/1636/> (cited on page 468).
- [607] The Windows Club. *Understanding Blob, Queue and Table Storage for Windows Azure*. Web page. Online; accessed 28-jan-2017. Jan. 2017. URL: <http://www.thewindowsclub.com/understanding-blobqueueutable-storage-windows-azure> (cited on page 480).
- [608] *theano*. Web Page. Accessed: 2017-1-21. URL: <http://deeplearning.net/software/theano/introduction.html> (cited on page 427).
- [609] Chandu Thekkath. *Naiad - Microsoft Research*. webpage. Accessed: 2017-1-27. Microsoft, Oct. 2011. URL: <https://www.microsoft.com/en-us/research/project/naiad/> (cited on page 417).
- [610] *Three.js*. Web Page. Accessed: 2017-02-27. Mar. 2017. URL: <https://en.wikipedia.org/wiki/Three.js> (cited on page 433).
- [611] *Creating a scene*. Web Page. Accessed: 2017-02-27. Dec. 2016. URL: https://threejs.org/docs/index.html#Manual/Getting_Started/Creating_a_scene (cited on page 434).
- [612] Apache. *About Thrift*. Web Page. Accessed: 2017-02-12. URL: <https://thrift.apache.org/> (cited on page 513).
- [613] *Apache Tika*. Web Page. URL: <https://tika.apache.org/> (cited on page 466).
- [614] *TinkerPop*. Web Page. Accessed: 2/6/2017. URL: <http://tinkerpop.apache.org/> (cited on page 479).
- [615] *Titan DB*. Web Page. Accessed: 2/4/2017. URL: <http://titan.thinkaurelius> (cited on page 479).
- [616] *Tokyo Cabinet*. Web Page. Accessed: 2017-1-27. URL: <http://fallabs.com/tokyocabinet/> (cited on pages 472, 473).
- [617] Stanimire Tomov, Jack Dongarra, and Marc Baboulin. “Towards dense linear algebra for hybrid GPU accelerated manycore systems”. In: *Parallel Computing* 36.5 (2010), pages 232–240. URL: <http://www.sciencedirect.com/science/article/pii/S0167819109001276> (visited on 02/13/2017) (cited on page 425).
- [618] Stanimire Tomov et al. “Dense linear algebra solvers for multicore with GPU accelerators”. In: *Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*. IEEE, 2010, pages 1–8. URL: http://link.springer.com/chapter/10.1007/978-3-319-06548-9_1 (visited on 02/13/2017) (cited on page 425).
- [619] Cardiff University. *Triana Documentation*. Web Page. Accessed: 2017-2-20. Cardiff University, 2012. URL: <http://www.trianacode.org/> (cited on page 415).
- [620] Craig Trim. *Jen: Semantic Web Framework*. Web Page. Accessed: 02/03/2016. URL: <http://trimc-nlp.blogspot.com/2013/06/introduction-to-jena.html> (cited on page 479).
- [621] *Taverna Workflows: Syntax and Semantics*. IEEE, Dec. 2007. ISBN: 0-7695-3064-8. DOI: [10.1109/E-SCIENCE.2007.71](https://doi.org/10.1109/E-SCIENCE.2007.71). URL: <http://ieeexplore.ieee.org/document/4426917/> (cited on page 415).
- [622] Matteo Turilli, Mark Santcroos, and Shantenu Jha. “A Comprehensive Perspective on Pilot-Job Systems”. In: *ACM arXiv*. Mar. 2016, pages 1–26 (cited on page 488).
- [623] James Turnbull. *The Terraform Book*. 110th edition. 9780988820258. Turnbull Press, Nov. 2016 (cited on page 502).
- [624] tutorialspoint. *SQLite Overview*. Website. URL: https://www.tutorialspoint.com/sqlite/sqlite_overview.htm (cited on page 467).

- [625] *Microsoft Azure - Queues*. Web Page. Accessed: 2017-2-10. URL: https://www.tutorialspoint.com/microsoft_azure/microsoft_azure_queues.htm (cited on page 461).
- [626] Dylan Tweney. *AppFog gives developers an easier way to deploy cloud apps (interview)*. Online. May 2012. URL: <http://venturebeat.com/2012/05/15/appfog-gives-developers-an-easier-way-to-deploy-cloud-apps-interview/> (cited on page 438).
- [627] *Twister: Iterative MapReduce*. Web Page. URL: <http://www.iterativemapreduce.org/> (visited on 02/13/2017) (cited on page 452).
- [628] Twitter. *Flying faster with Twitter Heron*. Web Page. Accessed: 2017-02-04. URL: <https://blog.twitter.com/2015/flying-faster-with-twitter-heron> (cited on page 449).
- [629] Twitter. *Open Sourcing Twitter Heron*. Web Page. Accessed: 2017-02-04. URL: <https://blog.twitter.com/2016/open-sourcing-twitter-heron> (cited on page 449).
- [630] *Kyoto Tycoon*. Web Page. URL: <http://fallabs.com/kyototycoon/> (cited on page 473).
- [631] *Tyrant FALLabs*. Web Page. URL: <http://fallabs.com/tokyotyrant/> (cited on page 473).
- [632] *Tyrant Blog*. Web Page. URL: https://www.percona.com/blog/2009/10/19/mysql-memcached_tyrant_part3/ (cited on page 473).
- [633] *Understanding LXC and LXD*. web page. URL: <http://thevarguy.com/open-source-application-software-companies/understanding-lxc-and-lxd-canonicals-open-source-containe> (cited on page 518).
- [634] ISS team - University of Texas. *Galois website*. Web Page. Accessed: 2017-2-27. URL: <http://iss.ices.utexas.edu/?p=projects/galois> (cited on page 454).
- [635] *University Policies: Cheating and Plagiarism*, ACA-72. Web Page. updates 1975. 1961. URL: <https://policies.iu.edu/policies/aca-72-cheating-plagiarism/index.html> (cited on page 82).
- [636] *Using Amazon S3*. Web Page. Accessed: 2017-1-27. URL: <http://docs.aws.amazon.com/AmazonS3/latest/gsg/CopyingAnObject.html> (cited on page 492).
- [637] Devananda Van Der Veen and Llama Wrangler. *Introduction to Ironic*. Web Page. Accessed: 2017-2-27. Mar. 2015. URL: <https://docs.openstack.org/developer/ironic/deploy/user-guide.html> (cited on page 500).
- [638] Venkat Venkataramani. *What is the TAO cache used for at Facebook*. Web Page. June 2013. URL: <https://www.quora.com/What-is-the-TAO-cache-used-for-at-Facebook> (cited on page 479).
- [639] Magister Vis. *What Is Lumberyard? (Lumberyard Tutorials Series #1) - YouTube*. URL: <https://www.youtube.com/watch?v=Fxwo3KqSsUI> (visited on 02/27/2017) (cited on page 447).
- [640] VMware. web-page. accessed 2017-02-13. URL: <http://www.vmware.com/products/esxi-and-esx.html> (cited on page 507).
- [641] vmware. *vCloud*. Webpage. URL: <http://www.vmware.com/products/vcloud-suite.html> (cited on page 507).
- [642] LinkedIn. *Project Voldemort*. Web Page. Accessed: 2017-1-17. URL: <http://www.project-voldemort.com/voldemort/> (cited on page 471).
- [643] VoltDB. Web Page. URL: <https://www.wired.com/2016/04/kites-coding-assistant-spots-errors-finds-better-open-source/> (cited on page 440).
- [644] VoltDB. Web Page. URL: <https://www.voltdb.com/> (cited on page 463).
- [645] w3. *Apache Jena*. Web Page. Accessed: 2016.02.03. URL: https://www.w3.org/2001/sw/wiki/Apache_Jena (cited on page 479).

- [646] Liya Wang, Peter Van Buren, and Doreen Ware. “Architecting a distributed bioinformatics platform with iRODS and iPlant Agave API”. In: *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*. Dec. 2015. ISBN: 9781467397957 (cited on pages 439, 440).
- [647] Pau Kiat Wee. “Instant AppFog”. In: Packt Publishing, July 2013. Chapter 1. URL: <https://www.packtpub.com/mapt/book/Web-Development/9781782167624/1/ch01lvl1sec03/So,+what+is+AppFog%3F> (cited on page 438).
- [648] Sage A. Weil et al. “Ceph: A scalable, high-performance distributed file system”. In: *Proceedings of the 7th symposium on Operating systems design and implementation*. OSDI ’06. Accessed: 2017-1-26. Seattle, Washington: USENIX Association, Nov. 2006, pages 307–320. ISBN: 1-931971-47-1. URL: https://www.usenix.org/legacy/event/osdi06/tech/full_papers/weil/weil.pdf (cited on page 490).
- [649] Johannes Wettinger. *any2api - The better way to create awesome APIs*. Web Page. Accessed: 2017-02-02. URL: <http://www.any2api.org/> (cited on page 503).
- [650] Johannes Wettinger, Uwe Breitenbücher, and Frank Leymann. “DevOpSlang – Bridging the Gap Between Development and Operations”. In: *Proceedings of the 3rd European Conference on ServiceOriented and Cloud Computing (ESOCC 2014)*. Lecture Notes in Computer Science (LNCS). accessed 2017-02-26. SpringerVerlag, 2014, pages 108–122 (cited on page 502).
- [651] Johannes Wettinger, Uwe Breitenbücher, and Frank Leymann. “Any2API Automated APIfication”. In: *CLOSER 2015 - Proceedings of the 5th International Conference on Cloud Computing and Services Science*. Edited by Markus Helfert, Donald Ferguson, and Víctor Méndez Muñoz. Lisbon, Portugal: SciTePress, 20-22 May 2015, pages 475–486. DOI: [10.5220/0005472704750486](https://doi.org/10.5220/0005472704750486). URL: <https://pdfs.semanticscholar.org/1cd4/4b87be8cf68ea5c4c642d38678a7b40a86de.pdf> (cited on page 503).
- [652] *What is a public cloud?* Webpage. URL: <http://www.interoute.com/cloud-article/what-public-cloud> (cited on page 508).
- [653] *What Is Amazon Route 53?* Web Page. Accessed: 2017-02-24. URL: <http://docs.aws.amazon.com/Route53/latest/DeveloperGuide>Welcome.html> (cited on page 509).
- [654] *What is HTCondor?* Web Page. URL: <https://research.cs.wisc.edu/htcondor/description.html> (cited on page 487).
- [655] *What is hubzero?* Web Page. URL: <https://hubzero.org/documentation/1.0.0/installation> (cited on page 439).
- [656] *What is the Jupyter Notebook? — Jupyter Notebook 5.0.0.dev documentation*. Web page. (Visited on 02/27/2017) (cited on page 416).
- [657] *Whirr technology*. webpage. accessed: 2017 – 2 – 4. URL: <https://whirr.apache.org/> (cited on page 495).
- [658] *Whirr technology*. webpage. accessed: 2017 – 2 – 4. URL: <http://www.slideshare.net/huguk/apache-whirr> (cited on page 495).
- [659] *Why Oozie? | Just a simple Hadoop DBA*. Web Page. URL: <https://prodlife.wordpress.com/2013/12/09/why-oozie/> (visited on 02/13/2017) (cited on page 417).
- [660] *Why use LXC (LinuX Containers) ?* web page. URL: <http://www.jpablo128.com/why-use-lxc-linux-containers/> (cited on page 505).
- [661] Wikipedia. *Apache Flex*. Web Page. Accessed: 2017-03-06. Mar. 2017. URL: https://en.wikipedia.org/wiki/Apache_Flex (cited on page 520).
- [662] Wikipedia. Web Page. URL: <https://en.wikipedia.org/wiki/Sqoop> (cited on page 484).
- [663] Wikipedia. Web Page. URL: https://en.wikipedia.org/wiki/Portable_Batch_System (cited on page 487).

- [664] Wikipedia. web-page. accessed 2017-02-13. URL: https://en.wikipedia.org/wiki/VMware_ESXi (cited on page 507).
- [665] Wikipedia. *Apache Apex wiki*. Web Page. Accessed: 2017-02-26. URL: https://en.wikipedia.org/wiki/Apache_Apex (cited on page 518).
- [666] Wikipedia. *DataNucleus Wiki*. Web Page. Accessed: 2017-02-04. URL: <https://en.wikipedia.org/wiki/DataNucleus> (cited on page 465).
- [667] Wikipedia. *Java Message Service - Wikipedia*. webpage. Accessed : 01-18-2017. URL: https://en.wikipedia.org/wiki/Java_Message_Service (cited on page 458).
- [668] Wikipedia. *Kubernetes Wiki*. Web Page. URL: <https://en.wikipedia.org/wiki/Kubernetes> (cited on page 500).
- [669] Wikipedia. *Neo4j*. Web page. Last Accessed: 2017.02.25. URL: <https://en.wikipedia.org/wiki/Neo4j> (cited on page 477).
- [670] Wikipedia. *UIMA*. Web Page. Accessed: 2017.02.03. URL: <https://en.wikipedia.org/wiki/UIMA> (cited on page 465).
- [671] Wikipedia. *Apache tomcat — Wikipedia, The Free Encyclopedia*. [Accessed: 02-24-2017]. 2010. URL: https://en.wikipedia.org/wiki/Apache_Tomcat (cited on page 515).
- [672] Wikipedia. *EclipseLink — Wikipedia, The Free Encyclopedia*. [Accessed: 02-06-2017]. 2010. URL: <https://en.wikipedia.org/wiki/EclipseLink> (cited on page 464).
- [673] Wikipedia. *Torch (Machine Learning) — Wikipedia, The Free Encyclopedia*. [Accessed: 02-06-2017]. 2014. URL: [https://en.wikipedia.org/wiki/Torch_\(machine_learning\)](https://en.wikipedia.org/wiki/Torch_(machine_learning)) (cited on page 427).
- [674] Wikipedia. *CloudControl*. Wiki Page. Feb. 2016. URL: <https://en.wikipedia.org/wiki/CloudControl> (cited on page 439).
- [675] Wikipedia. *Key-value database*. WebPage. Nov. 2016. URL: https://en.wikipedia.org/wiki/Key-value_database (cited on page 462).
- [676] Wikipedia. *Rasdaman — Wikipedia, The Free Encyclopedia*. [Accessed: 02-23-2017]. 2016. URL: <https://en.wikipedia.org/wiki/Rasdaman> (cited on page 468).
- [677] Wikipedia. *Spanner (Database)*. Web Page. accessed 2017-01-29. Oct. 2016. URL: [https://en.wikipedia.org/wiki/Spanner_\(database\)](https://en.wikipedia.org/wiki/Spanner_(database)) (cited on page 476).
- [678] Wikipedia. *Amazon Redshift*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: https://en.wikipedia.org/wiki/Amazon_Redshift (cited on page 445).
- [679] Wikipedia. *Ansible (software)*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: [https://en.wikipedia.org/wiki/Ansible_\(software\)](https://en.wikipedia.org/wiki/Ansible_(software)) (cited on page 497).
- [680] Wikipedia. *Apache CloudStack*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: https://en.wikipedia.org/wiki/Apache_CloudStack (cited on page 507).
- [681] Wikipedia. *Apache Flink*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: https://en.wikipedia.org/wiki/Apache_Flink (cited on page 452).
- [682] Wikipedia. *Apache Hadoop*. Web Page. accessed 2017-03-18. Mar. 2017. URL: https://en.wikipedia.org/wiki/Apache%5C_Hadoop (cited on page 451).
- [683] Wikipedia. *Apache Phoenix — Wikipedia, The Free Encyclopedia*. Web page. Online; accessed 25-jan-2017. Jan. 2017. URL: https://en.m.wikipedia.org/wiki/Apache%5C_Phoenix (cited on page 442).
- [684] Wikipedia. *Bigtable*. Web Page. accessed 2017-01-29. Jan. 2017. URL: <https://en.wikipedia.org/wiki/Bigtable> (cited on page 476).
- [685] Wikipedia. *Chef (Software)*. Web Page. accessed 2017-01-26. Jan. 2017. URL: [https://en.wikipedia.org/wiki/Chef_\(software\)](https://en.wikipedia.org/wiki/Chef_(software)) (cited on page 500).
- [686] Wikipedia. *Cloud computing — Wikipedia, The Free Encyclopedia*. web page. Online; accessed 21-jan-2017. Jan. 2017. URL: https://en.wikipedia.org/wiki/Cloud_computing (cited on page 436).

- [687] Wikipedia. *CUDA*. Web Page. Online; accessed 25-Feb-2017. Feb. 2017. URL: <https://en.wikipedia.org/wiki/CUDA> (cited on page 516).
- [688] Wikipedia. *Data Analytics Acceleration Library — Wikipedia, The Free Encyclopedia*. [Accessed: 02-23-2017]. 2017. URL: https://en.wikipedia.org/wiki/Data_Analytics_Acceleration_Library (cited on page 427).
- [689] Wikipedia. *Deeplearning4j*. Web Page. Accessed: 2017-02-11. Feb. 2017. URL: <https://en.wikipedia.org/wiki/Deeplearning4j> (cited on page 428).
- [690] Wikipedia. *Erlang (programming language) — Wikipedia, The Free Encyclopedia*. Web page. Online; accessed: 29-jan-2017. Jan. 2017. URL: [https://en.wikipedia.org/wiki/Erlang_\(programming_language\)](https://en.wikipedia.org/wiki/Erlang_(programming_language)) (cited on page 474).
- [691] Wikipedia. *Fiddler Software*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: [https://en.wikipedia.org/wiki/Fiddler_\(software\)](https://en.wikipedia.org/wiki/Fiddler_(software)) (cited on page 514).
- [692] Wikipedia. *Graph Database*. Web Page. accessed 2017-01-30. Feb. 2017. URL: https://en.wikipedia.org/wiki/Graph_database (cited on page 477).
- [693] Wikipedia. *Hazelcast*. Web Page. accessed 2017-01-29. Jan. 2017. URL: <https://en.wikipedia.org/wiki/Hazelcast> (cited on pages 462, 463).
- [694] Wikipedia. *Hierarchical Data Format — Wikipedia, The Free Encyclopedia*. Web Page. Online,accessed: 2017-1-28. Feb. 2017. URL: https://en.wikipedia.org/wiki/Hierarchical_Data_Format (cited on page 482).
- [695] Wikipedia. *Hyper-V*. Web Page. Accessed: 2017-02-23. Feb. 2017. URL: <https://en.wikipedia.org/wiki/Hyper-V> (cited on page 504).
- [696] Wikipedia. *Hypervisor*. WebPage. Feb. 2017. URL: <https://en.wikipedia.org/wiki/Hypervisor> (cited on page 504).
- [697] Wikipedia. *IBM Cloudant*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: <https://en.wikipedia.org/wiki/Cloudant> (cited on page 475).
- [698] Wikipedia. *IBM General Parallel File System*. Web Page. accessed 2017-01-29. Jan. 2017. URL: https://en.wikipedia.org/wiki/IBM_General_Parallel_File_System (cited on page 491).
- [699] Wikipedia. *Relational database*. WebPage. Jan. 2017. URL: https://en.wikipedia.org/wiki/Relational_database (cited on page 462).
- [700] Wikipedia. *Snort Software*. Web Page. Accessed: 2017-02-12. Feb. 2017. URL: [https://en.wikipedia.org/wiki/Snort_\(software\)](https://en.wikipedia.org/wiki/Snort_(software)) (cited on page 514).
- [701] Wikipedia. *Swift (programming language)*. Web Page. Accessed: 2017-02-23. Feb. 2017. URL: [https://en.wikipedia.org/wiki/Swift_\(programming_language\)](https://en.wikipedia.org/wiki/Swift_(programming_language)) (cited on page 414).
- [702] Wikipedia. *Fusion Table Support*. Web Page. Last updated in 1 November 2016. URL: <https://support.google.com/fusiontables/answer/171181?hl=en> (cited on page 431).
- [703] Wikipedia. *LinkedIn*. Web page. accessed 17-March-2017. This page was last modified on 18 March 2017, at 22:27. URL: <https://cloud.google.com/ml-engine/docs/concepts/technical-overview> (cited on page 449).
- [704] Brandon Wiley. “Distributed Hash TTable, Part 1”. In: *Linux Journal* 114 (Oct. 2003). From issue number 114. URL: <http://www.linuxjournal.com/article/6797?page=0,0> (cited on page 472).
- [705] Duncan C.E Winn. *Cloud Foundry-The Cloud Native Platform*. O'Reiley Media, 2016, page 70. ISBN: 978-4919-6573-3. URL: <https://books.google.com/books?id=XP2wDAAAQBAJ&printsec=frontcover&dq=Cloud+foundry> (cited on page 437).

- [706] Alex Woodie. *Spark Gets New Machine Learning Framework: KeystoneML*. URL: <https://www.datanami.com/2015/05/26/spark-gets-new-machine-learning-framework-keystoneml/> (visited on 02/27/2017) (cited on page 422).
- [707] Alex Woodie. *Apache Beam's Ambitious Goal: Unify Big Data Development*. Web page. accessed 25-feb-2017. Apr. 2016. URL: <https://www.datanami.com/2016/04/22/apache-beam-emerges-ambitious-goal-unify-big-data-development/> (cited on page 515).
- [708] Apache Software Foundation. *Apache Ranger*. Web Page. Online; accessed 9-Mar-2017. URL: <http://ranger.apache.org/> (cited on page 520).
- [709] CASK. *CASK - The First Unified Integration Platform for Big Data*. Web Page. Online; accessed 18-Feb-2017. URL: <http://cask.co/products/cdap/> (cited on page 516).
- [710] CASK. *Getting Started Developing with CDAP*. Web Page. Online; accessed 18-Feb-2017. URL: <http://docs.cask.co/cdap/current/en/developers-manual/getting-started/index.html> (cited on page 516).
- [711] *Open Source Data Turbine Initiative*. Web Page. Accessed: 2017-2-26. URL: <http://dataturbine.org/> (cited on page 451).
- [712] Jeff Lindsay. *Gitreceive*. Web Page. Accessed: 2017-2-13. Feb. 2016. URL: <https://github.com/progrium/gitreceive> (cited on page 501).
- [713] *High Performance ParalleX (HPX-5)*. Web Page. Accessed: 2017-1-17. URL: <https://hpx.crest.iu.edu/> (cited on page 455).
- [714] *HPX-5: user guide*. Accessed: 2017-1-17. HPX-5. URL: https://hpx.crest.iu.edu/users_guide (cited on page 456).
- [715] Open Grid Forum. *Open Cloud Computing Interface*. Web Page. Accessed: 2017-1-17. URL: <http://occi-wg.org/> (cited on page 494).
- [716] Ralf Nyren et al. *Open Cloud Computing Interface Core*. OGF Published Document GWD-R-P.221. Accessed: 2017-1-17. Open Grid Forum, P.O. Box 1738, Muncie IN 47308, USA: Global Grid Forum, Sept. 2016. URL: <https://www.ogf.org/documents/GFD.221.pdf> (cited on page 494).
- [717] Ralf Nyren, Andy Edmonds, and Thijs Metsch atnd Boris Parak. *Open Cloud Computing Interface - HTTP Protocol*. OGF Published Document GWD-R-P.223. Accessed: 2017-1-17. Open Grid Forum, P.O. Box 1738, Muncie IN 47308, USA: Global Grid Forum, Sept. 2016. URL: <https://www.ogf.org/documents/GFD.223.pdf> (cited on page 494).
- [718] Ralf Nyren et al. *Open Cloud Computing Interface -JSON Rendering*. OGF Published Document GWD-R-P.226. Accessed: 2017-1-17. Open Grid Forum, P.O. Box 1738, Muncie IN 47308, USA: Global Grid Forum, Sept. 2016. URL: <https://www.ogf.org/documents/GFD.226.pdf> (cited on page 494).
- [719] Michel Drescher, Boris Parak, and David Wallom. *OCCI Compute Resource Templates Profile*. OGF Published Document GWD-R-P.222. Accessed: 2017-1-17. Open Grid Forum, P.O. Box 1738, Muncie IN 47308, USA: Global Grid Forum, Apr. 2015. URL: <https://www.ogf.org/documents/GFD.222.pdf> (cited on page 494).
- [720] Andy Edmonds and Thijs Metsch. *Open Cloud Computing Interface - Text Rendering*. OGF Published Document GWD-R-P.229. Accessed: 2017-1-17. Open Grid Forum, P.O. Box 1738, Muncie IN 47308, USA: Global Grid Forum, Sept. 2016. URL: <https://www.ogf.org/documents/GFD.229.pdf> (cited on page 494).
- [721] Hortonworks. *Apache Ranger - Overview*. Web Page. Online; accessed 9-Mar-2017. URL: https://hortonworks.com/apache/ranger/#section_2 (cited on page 520).
- [722] Syed Mahmood and Srikanth Venkat. *FOR YOUR EYES ONLY: DYNAMIC COLUMN MASKING & ROW-LEVEL FILTERING IN HDP2.5*. Web Page. Online; accessed 9-Mar-

2017. Sept. 2016. URL: <https://hortonworks.com/blog/eyes-dynamic-column-masking-row-level-filtering-hdp2-5/> (cited on page 520).
- [723] SAP. *What is SAP HANA*. Web Page. Accessed: 2017-1-17. URL: <http://www.sap.com/product/technology-platform/hana.html> (cited on page 443).
- [724] xcat. *Extreme cloud/cluster administration toolkit*. Webpage. 2015. URL: <http://xcat-docs.readthedocs.io/en/stable/> (cited on page 498).
- [725] Xen - Wikipedia. Web Page. Accessed: 2017-02-04. URL: <https://en.wikipedia.org/wiki/Xen> (cited on page 503).
- [726] Xen Project Overview. Web Page. Accessed: 2017-02-04. URL: https://wiki.xenproject.org/wiki/Xen_Project_Software_Overview (cited on page 503).
- [727] Xen Feature List. Web Page. Accessed: 2017-02-04. URL: https://wiki.xenproject.org/wiki/Xen_Project_4.7_Feature_List (cited on page 503).
- [728] Reynold S. Xin et al. “GraphX: A Resilient Distributed Graph System on Spark”. In: *First International Workshop on Graph Data Management Experiences and Systems*. GRADES ’13. New York, New York: ACM, 2013, 2:1–2:6. ISBN: 978-1-4503-2188-4. DOI: [10.1145/2484425.2484427](https://doi.acm.org/10.1145/2484425.2484427). URL: <http://doi.acm.org/10.1145/2484425.2484427> (cited on page 429).
- [729] T. Xu, K. Sato, and S. Matsuoka. “CloudBB: Scalable I/O Accelerator for Shared Cloud Storage”. In: *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*. Institute of Electrical and Electronics Engineers (IEEE), Dec. 2016, pages 509–518. DOI: [10.1109/ICPADS.2016.0074](https://doi.ieeexplore.ieee.org/abstract/document/7807400) (cited on page 490).
- [730] Shelhamer Yangqing Jia Evan. *Caffe | Deep Learning Framework*. Web Page. Accessed: 02-06-2017. URL: <http://caffe.berkeleyvision.org/> (cited on page 427).
- [731] Edward J. Yoon. *MRQL - a SQL on Hadoop Miracle*. Web Page. accessed 2017-01-29. Jan. 2017. URL: <http://www.hadoopsphere.com/2013/04/mrql-sql-on-hadoop-miracle.html> (cited on page 443).
- [732] Apache Zeppelin. *Apache Zeppelin 0.7.0 Documentation*. Web Page. Accessed: 2017-2-27. Feb. 2017. URL: <https://zeppelin.apache.org/docs/0.7.0/> (cited on page 514).
- [733] Bingjing Zhang et al. “Applying twister to scientific applications”. In: *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. IEEE. 2010, pages 25–32 (cited on page 452).
- [734] Xianbo Zhang et al. “Hptfs: A high performance tape file system”. In: *Proceedings of 14th NASA Goddard/23rd IEEE conference on Mass Storage System and Technologies*. Ieee Computer Society (September 1995), 2006. DOI: [10.1.1.184.1133](https://doi.org/10.1.1.184.1133). URL: https://www.dtc.umn.edu/publications/reports/2006_11.pdf (cited on page 490).
- [735] Zookeeper - Overview. Web Page. Accessed: 2017-01-23. URL: <https://zookeeper.apache.org/doc/trunk/zookeeperOver.html> (cited on page 513).
- [736] Zookeeper - Wikipedia. Web Page. Accessed: 2017-01-23. URL: https://en.wikipedia.org/wiki/Apache_ZooKeeper (cited on page 513).
- [737] IBM - What is Zookeeper. Web Page. Accessed: 2017-01-23. URL: <http://www-01.ibm.com/software/data/infosphere/hadoop/zookeeper/> (cited on page 513).



Index

L^AT_EX

- Book
 - Boxes, 145
 - Citation, 140
 - Corollaries, 142
 - Definitions, 141
 - Examples, 143
 - Exercises, 144
 - Figure, 145
 - IU, 145
 - Lists, 140
 - Notations, 142
 - NOTE, 145
 - Paragraphs of Text, 139
 - Problems, 145
 - Propositions, 143
 - Remarks, 142
 - Table, 145
 - Theorems, 140
 - Vocabulary, 145
 - WARNING, 145
- Contributing, 35, 36
- Convention, 37
- Dask, 270
- Fox, Geoffrey C, 34
- Latex
 - #, 96
- \$, 96
- %, 96
- _, 96
- cycle, 107
- Elements, 95
 - description, 97
 - enumerate, 97
 - highlight text, 96
 - images, 97
 - itemiz, 97
 - labels, 99
 - mathematics, 100
 - sections, 96
 - tables, 99
- installation, 93
 - OSX, 94
 - ubuntu, 94
 - Windows, 94
- markdown, 106
- other documentation, 108
- overleaf, 95
- proceedings
 - acm, 101
 - ieee, 101
- Sharelatex, 94
- slides, 102

Python

- 2 and 3, 238
- boolean, 240

- classes, 249
 data types, 240
 date, 242
 dict, 247
 for, 244
 function, 248
 if, 243
 import, 241
 from, 241
 interactive, 237
 keys, 247
 list, 244
 numbers, 240
 REPL, 237
 set, 245
 statements, 239
 strings, 239
 time, 242
 variables, 239
- Slides
 About
 Test slides (10), 37
 Cloud
 Analysis Algorithms (Page 35), 338
 Analysis Algorithms - pptx (Page 35), 338
 Apache Data Analysis OpenStack (Page 1), 345
 Apache Data Analysis OpenStack - pptx (Page 1), 345
 BigTable (Page 28), 336
 BigTable - pptx (Page 28), 336
 BLAST Parallelization (Page 13), 348
 BLAST Parallelization - pptx (Page 13), 348
 Challenges (Page 42), 328
 Checklists and Challenges (Page 11), 331
 Clouds in the Workplace (Page 1), 331
 Clusters and Resource Management (Page 66), 331
 Clusters and Resource Management - pptx (Page 66), 331
 Coding and Iterative Alternatives (Page 43), 354
 Coding and Iterative Alternatives - pptx (Page 43), 354
 CPU, Memory & I/O Devices (Page 58), 330
 CPU, Memory & I/O Devices - pptx (Page 58), 330
 Cultivating Clouds (Page 15), 332
 Cultivating Clouds - Conclusions (Page 1), 332
 Data Center Automation (Page 74), 331
 Data Center Automation - pptx (Page 74), 331
 Data Center Model (Page 9), 328
 Data Center Setup (Page 16), 332
 Data Intensive Sciences (Page 19), 328
 Data Locality (Page 10), 348
 Data Locality - pptx (Page 10), 348
 Discussions and ParallelThinking (Page 10), 352
 Discussions and ParallelThinking - pptx (Page 10), 352
 Everyday Data (Page 4), 357
 Fault Tolerance (Page 36), 346
 Fault Tolerance - pptx (Page 36), 346
 Faults & Frameworks (Page 9), 358
 Google Architecture (Page 6), 341
 Google Architecture - pptx (Page 6), 341
 Google Components (Page 1), 341
 Google Components - pptx (Page 1), 341
 Google History (Page 14), 342
 Google History - pptx (Page 14), 342
 Google Search Engine 1 (Page 15), 351
 Google Search Engine 1 - pptx (Page 15), 351
 Google Search Engine 2 (Page 21), 352
 Google Search Engine 2 - pptx (Page 21), 352
 Growth of Virtual Machines (Page 28), 329
 Growth of Virtual Machines - pptx (Page 28), 329
 Hadoop Extensions (Page 50), 353
 Hadoop Framework (Page 15), 346
 Hadoop Framework - pptx (Page 15), 346
 Hadoop PageRank (Page 1), 352
 Hadoop PageRank - pptx (Page 1), 352
 Hadoop Tasks (Page 24), 346
 Hadoop Tasks - pptx (Page 24), 346
 Hadoop WordCount on VMs (Page 17), 346
 Hadoop WordCount on VMs - pptx (Page

- 17), 346
HBase (Page 44), 336
HBase - pptx (Page 44), 336
HBase Coding (Page 60), 336
HBase Coding - pptx (Page 60), 336
How Hadoop Runs on a MapReduceJob (Page 8), 347
How Hadoop Runs on a MapReduceJob - pptx (Page 8), 347
IaaS, PaaS and SaaS (Page 25), 328
Implementation Levels (Page 41), 330
Implementation Levels - pptx (Page 41), 330
Indexamples (Page 15), 337
Indexamples - pptx (Page 15), 337
Indexing 101 (Page 20), 337
Indexing 101 - pptx (Page 20), 337
Indexing Applications (Page 1), 336
Indexing Applications - pptx (Page 1), 336
Introduction (Page 1), 327
Introduction to BLAST (Page 1), 347
Introduction to BLAST - pptx (Page 1), 347
Iterative MapReduce Models (Page 1), 353
Iterative MapReduce Models - pptx (Page 1), 353
Literature Review (Page 16), 347
Literature Review - pptx (Page 16), 347
MapReduce (Page 6), 345
MapReduce - pptx (Page 6), 346
MapReduce Model Comparison (Page 24), 354
MapReduce Model Comparison - pptx (Page 24), 354
MapReduce Refresher (Page 1), 351
NoSQL Characteristics (Page 11), 335
NoSQL Characteristics - pptx (Page 11), 335
Optimal Data Locality (Page 17), 349
Optimal Data Locality - pptx (Page 17), 349
Parallel Processes (Page 4), 353
Parallel Processes - pptx (Page 4), 353
Programming on a Compute Cluster (Page 1), 347
Programming on a Compute Cluster - pptx (Page 1), 347
RDBMS vs. NoSQL (Page 1), 335
RDBMS vs. NoSQL - pptx (Page 1), 335
Related Work (Page 11), 337
Related Work - pptx (Page 11), 337
Resource Utilization and Speculative Execution (Page 46), 349
Resource Utilization and Speculative Execution - pptx (Page 46), 349
SIMD vs MIMD;SPMD vs MPMD (Page 1), 348
SIMD vs MIMD;SPMD vs MPMD - pptx (Page 1), 348
Social Media Searches (Page 28), 338
Social Media Searches - pptx (Page 28), 338
Spouts to Bolts (Page 15), 358
Static and Variable Data (Page 10), 354
Static and Variable Data - pptx (Page 10), 354
Streaming the Data Ocean (Page 6), 357
Streams of Events (Page 1), 358
Student Work 1 (Page 7), 35
Student Work 1 - pptx (Page 7), 35
Student Work 2 (Page 12), 35
Student Work 2 - pptx (Page 12), 35
Task Granularity (Page 29), 349
Task Granularity - pptx (Page 29), 349
Tools and Mechanisms (Page 47), 330
Tools and Mechanisms - pptx (Page 47), 330
Twister K-means (Page 34), 354
Twister K-means - pptx (Page 34), 354
Health
131 (Health), 372
131 (Proteomics and Information Visualization), 374
i524
Acess Patterns, Data Access Patterns and Introduction to using HPC-ABDS (Pages ???), 409
Course Introduction (Pages 39), 409
Overview (Pages 26), 409
Introduction
Cloud Applications in Research (20), 365
Computing and MapReduce (9), 365
Computing Model I (14), 364
Computing Model II (27), 364

- Conclusions (4), 366
 - Data Deluge (20), 363
 - Data Science Education (19), 365
 - Data Science Process (10), 364
 - Digital Distruption and transformation (28), 363
 - Indusrial Trends II (16), 363
 - Industrial Trends (16), 363
 - Industrial Trends III (21), 363
 - Jobs (8), 363
 - Motivation (30), 362
 - Physics-inforamtics (6), 364
 - Recommender Systems I (9), 365
 - Recommender Systems II (6), 365
 - Research Model (4), 364
 - Web Search and Information Retrieval (6), 365
 - Lifestyle
 - 18 (Filtering), 379
 - 45 (Recommender), 377
 - 49 (Recommender), 378
 - Overview
 - TBD (22), 369
 - TBD (30), 371
 - TBD (35), 370
 - TBD (45), 367
 - Physics
 - 20 (Higgs), 380
 - 29 (Higgs II), 381
 - 39 (Higgs), 382
 - 44 (Higgs III), 383
 - Radar
 - 58 (Radar), 385
 - Sensor
 - 31 (Sensor I), 386
 - 44 (Sensor II), 386
 - Sport
 - 40 (Overview), 388
 - 41 (Sporta II), 389
 - 44 (Sports III), 390
 - Usecases
 - 100 (51), 394
 - 43 (Features), 396
 - 45 (Overview), 391
 - Web
 - 33 (Text Mining), 402
 - 56 (Web Search and Text Mining), 400
 - Video
 - About
- Test Video (25:36), 37
 - Cloud
 - 4:30 (HBase Coding), 336
 - Analysis Algorithms (6:57), 338
 - Apache Data Analysis OpenStack (12:01), 345
 - BigTable (6:55), 336
 - BLAST Parallelization (4:44), 348
 - Challenges (5:27), 328
 - Checklists and Challenges (9:08), 331
 - Clouds in the Workplace (7:13), 331
 - Clusters and Resource Management (5:07), 331
 - Coding and Iterative Alternatives (5:14), 354
 - CPU, Memory & I/O Devices (6:41), 330
 - Cultivating Clouds (5:10), 332
 - Data Center Automation (3:30), 331
 - Data Center Model (8:08), 327
 - Data Center Setup (7:49), 332
 - Data Intensive Sciences (2:44), 328
 - Data Locality (8:36), 348
 - Discussions and ParallelThinking (11:12), 352
 - Emacs org-mode (18:04), 129
 - Everday Data (9:31), 357
 - Fault Tolerance (2:45), 346
 - Faults & Frameworks (7:46), 358
 - Google Architecture (8:40), 341
 - Google Components (7:02), 341
 - Google History (10:36), 341
 - Google Search Engine 1 (8:04), 351
 - Google Search Engine 2 (8:32), 352
 - Growth of Virtual Machines (10:16), 329
 - Hadoop Extensions (5:37), 353
 - Hadoop Framework (8:32), 346
 - Hadoop PageRank (7:58), 352
 - Hadoop Tasks (11:01), 346
 - Hadoop WordCount on VMs (7:30), 346
 - HBase (7:37), 336
 - How Hadoop Runs on a MapReduceJob (9:25), 347
 - IaaS/PaaS/SaaS (10:17), 328
 - Implementation Levels (7:57), 330
 - Indexamples (8:35), 337
 - Indexing 101 (9:53), 337
 - Indexing Applications (9:33), 336
 - Introduction (8:31), 327

- Introduction to BLAST (8:27), 347
Iterative MapReduce Models (6:46), 353
Literature Review (9:43), 347
MapReduce (9:07), 345
MapReduce Model Comparison (6:56), 354
MapReduce Refresher (9:00), 351
NoSQL Characteristics (10:31), 335
Optimal Data Locality (4:17), 349
Parallel Processes (9:44), 353
Programming on a Compute Cluster (6:01), 347
RDBMS vs. NoSQL (9:22), 335
Related Work (5:56), 337
Resource Utilization and Speculative Execution (3:52), 349
SIMD vs MIMD;SPMD vs MPMD (9:42), 348
Social Media Searches (6:19), 338
Spouts to Bolts (8:42), 358
Static and Variable Data (11:01), 354
Streaming the Data Ocean (9:38), 357
Streams of Events (10:44), 357
Student Work 1 (8:48), 35
Student Work 2 (10:03), 35
Task Granularity (9:51), 349
Tools and Mechanisms (7:32), 330
Twister K-means (7:28), 354
- Git
Branch (2:25), 169
Checkout (3:11), 169
Configuration (2:47), 170
Fork (1:41), 168
GUI (3:47), 170
Merge (3:11), 169
Pull Request (4:26), 169
Rebase (4:20), 168
Windows (1:25), 170
- Health
CC) Genomics, Proteomics and Information Visualization (6:56), 374
Big Data and Health (10:02), 372
Big Data and Healthcare Indusry (10:02), 373
Clouds and Health (4:35), 373
EU Report on Redesigning health in Europe for 2020 (5:00), 373
Extrapolating to 2032 (15:13), 374
Genomics, Proteomics and Information Visualization (6:56), 374
Genomics, Proteomics and Information Visualization I (10:33), 374
Genomics, Proteomics and Information Visualization: II (7:41), 374
McKinsey Report (14:53), 373
Medical Big Data in the Clouds (15:02), 373
Medicine and the Internet of Things (8:17), 374
Microsoft Report on Big Data in Health (2:26), 373
Midical Image Big Data (6:33), 373
Status of Healthcare Today (16:09), 372
Telemedicine (8:21), 372
- 1524
A. Introduction to HPC-ABDS Software and Access Patterns (0:27:45), 409
B. Science Examples (Data Access Patterns) (0:18:38), 409
Basic Trends and Jobs (0:10:57), 409
Big Data Patterns - Sources of Parallelism (0:23:51), 410
Big Data Patterns - the Ogres and their Facets I (0:22:44), 410
C. Remaining General Access Patterns (11:26), 410
Clouds vs HPC - Data Intensive vs. Simulation Problems (0:20:26), 410
D. Summary HPC-ABDS Layers 1 - 6 (14:32), 410
E. Summary HPC-ABDS Layers 7 - 13 (30:52), 410
F. Summary HPC-ABDS Layers 14 - 17 (28:02), 410
Facets of the Big Data Ogres II (0:15:09), 410
First and Second Set of Features (0:18:26), 410
G. Summary HPC-ABDS Others (20:20), 410
Image Based Applications II (0:15:23), 410
Internet of Things Based Applications (0:25:25), 410
Introduction (0:13:59), 409
jabref (14:41), 81
Machine Learning Aspect of Second Feature Set and the Third Set (0:18:38),

- 410
 More of Software Stack (0:24:00), 411
 NIST Big Data Sub Groups (0:23:25),
 410
 NIST UseCases and Image Based Applications Examples I (0:25:20), 410
 Other sources of use cases and Classical Databases/SQL Solutions (0:16:50),
 410
 Part 1 (11:29), 409
 Part 2 (04:10), 409
 Part 3 (12:41), 409
 Real World Big Data (0:15:28), 409
 ShareLaTeX (8:49), 81
 SQL Solutions - Machine Learning Example - and MapReduce (0:18:49),
 410
- Introduction**
 Cloud Applications in Research (33:51),
 365
 Computing and MapReduce (14:02), 365
 Computing Model I (24:03), 364
 Computing Model II (28:18), 364
 Conclusions (4:59), 366
 Data Deluge (30:38), 363
 Data Science Education (28:08), 365
 Data Science Process (15:42), 364
 Digital Distruption and transformation (32:54), 363
 Industrial Trends III (30:13), 363
 Industrial Trends (19:25), 363
 Industrial Trends II (16:54), 363
 Jobs (9:39), 363
 Motivation (40:14), 362
 Physics-informatics (13:27), 364
 Recommender Systems I (12:21), 365
 Recommender Systems II (9:44), 365
 Research Model (7:33), 364
 Web Search and Information Retrieval (12:05), 365
- Lifestyle**
 Case Study of Recommender systems (3:21), 379
 Consumer Data Science (13:04), 378
 Examples of Recommender Systems (1:00),
 377
 Examples of Recommender Systems (8:34),
 378
 Item Based Filtering (11:18), 380
- k Nearest Neighbors and High Dimensional Spaces (10:03), 380
 k Nearest Neighbors and High Dimensional Spaces (7:16), 380
 Kaggle Competitions: (3:36), 377
 Netflix on Recommender Systems (14:20),
 377
 Recap and Examples of Recommender Systems (5:48), 378
 Recap of Recommender Systems II (8:46),
 379
 Recap of Recommender Systems III (10:48),
 379
 Recommender Systems I (8:06), 377
 Recommender Systems Introduction (12:56),
 377
 User-based nearest-neighbor collaborative filtering I (7:20), 379
 User-based nearest-neighbor collaborative filtering II (7:29), 379
 Vector Space Formulation of Recommender Systems new (9:06), 379
- Overview**
 TBD (11:27), 369
 TBD (11:28), 371
 TBD (11:49), 369
 TBD (13:04), 368
 TBD (16:04), 370
 TBD (2:10), 369
 TBD (2:58), 368
 TBD (3:42), 368
 TBD (4:27), 368
 TBD (5:07), 369
 TBD (5:45), 369
 TBD (6:00), 368
 TBD (6:24), 371
 TBD (6:51), 371
 TBD (7:28), 371
 TBD (7:34), 368
 TBD (8:02), 370
 TBD (9:37), 368
 TBD (9:49), 367
- Physics**
 Accelerator Picture Gallery of Big Science (11:21), 381
 Accept-Reject (5:54), 384
 Binomial Distribution: (12:38), 384
 Central Limit Theorem (4:47), 384
 Change shape of background & num of

- Higgs Particles (7:01), 382
Discovery of Higgs Particle (13:49), 380
Event Counting (7:02), 382
Gaussian Distributions (9:08), 383
Generators and Seeds II (7:10), 383
Higgs Particle Counting Errors (6:28), 383
Higgs Particle Events and Counting (9:30), 381
Interpretation of Probability (12:39), 384
Looking for Higgs Particle and Counting Introduction II (7:38), 381
Looking for Higgs Particle Experiments (9:29), 381
Monte Carlo Method (2:23), 384
Physics and Random Variables I (8:34), 382
Physics and Random Variables II (5:50), 382
Poisson Distribution (4:37), 384
Random variables and normal distributions (8:19), 382
Statistics of Events with Normal Distributions (11:25), 383
Using Statistics (14:02), 383
With Python examples of Signal plus Background (7:33), 382
- Python
Advanced SSH (2:47), 182
FutureGrid Introduction (11:50), 181
Shell Access via SSH (2:34), 182
ssh-key gen (4:06), 182
Windows users (3:51), 182
- Radar
Global Climate Change (2:51), 385
Ice Sheet Science (1:00), 385
Radar Informatics (3:31), 385
Radio Informatics (3:35), 386
Radio Overview (4:16), 386
Remote Sensing (6:43), 385
- Sensor
Earth/Environment/Polar Science data gathered by Sensors (4:58), 387
Industrial Internet of Things (1:24:02), 387
Internet of Things (12:36), 386
Robotics and IoT Expectations (8:05), 386
Sensor Clouds (4:40), 387
- Smart Grid (6:04), 387
U-Korea (U=Ubiquitous) (2:49), 387
Ubiquitous/Smart Cities (1:44), 387
- Sport
Basic Sabermetrics (26:53), 388
Introduction and Sabermetrics (Baseball Informatics) Lesson (31:4), 388
Other Video Data Gathering in Baseball (18:5), 390
Pitcher Quality (10:02), 390
PITCHf/X (10:39), 390
Pitching Clustering (20:59), 389
Soccer and the Olympics (8:28), 390
Spatial Visualization in NFL and NBA (15:19), 391
Wearables (22:2), 390
Wins Above Replacement (30:43), 389
- Systems
FutureGrid (12:12), 181
- Usecases
Architecture (10:05), 392
Astronomy and Physics Use Cases (17:33), 395
Commercial Use Cases (17:43), 394
Database (SQL) Use Case Classification (11:13), 396
Deep Learning and Social Networks Use Cases (14:19), 395
Defense Use Cases (15:43), 394
Energy Use Case (4:01), 395
Environment, Earth and Polar Science Use Cases (25:29), 395
Government Use Cases (17:43), 394
Healthcare and Life Science Use Cases (30:11), 395
Introduction (13:02), 392
NoSQL Use Case Classification (11:20), 396
Requirements (27:28), 394
Research Ecosystem Use Cases (9:09), 395
Security (9:51), 392
Summary of Use Case Classification (23:39), 396
Taxonomies (7:42), 392
Technology (4:14), 393
Use Case Classifications I (12:42), 396
Use Case Classifications II (20:18), 396
Use Case Classifications III (17:25), 397

- Virtualbox
 - Video (4:46), [176](#)
 - Video (seconds), [176](#)
- Web
 - Boolean and Vector Space Model (6:17),
[401](#)
 - Clustering and Topic Models (6:21), [402](#)
 - Fundametal Principals of Web Search
(5:06), [401](#)
 - Indices (5:44), [401](#)
 - Information Retrieval (6:06), [400](#)
 - Principles (9:30), [400](#)
 - Realated Applications (17:24), [402](#)
 - Search Engines (3:08), [401](#)
 - Text Mining (9:56), [400](#)
 - TF-IDF and Probabilistic Models (3:57),
[401](#)
 - Web Advertising and Search (9:02), [402](#)
 - Web crawling and Document Preparation (4:55), [401](#)
 - Web Search and Text Mining II (6:11),
[402](#)
 - Web Search History (5:48), [400](#)
- Writing
 - How to write a paper by Simon Peyton Jones (57:39), [81](#)
- von Laszewski, Gregor, [34](#)
- Writing, [85](#)
 - Checklist, [87](#)