I524 Lecture Notes

Release Draft

Gregor von Laszewski

Jan 28, 2017

CONTENTS

CHAPTER ONE

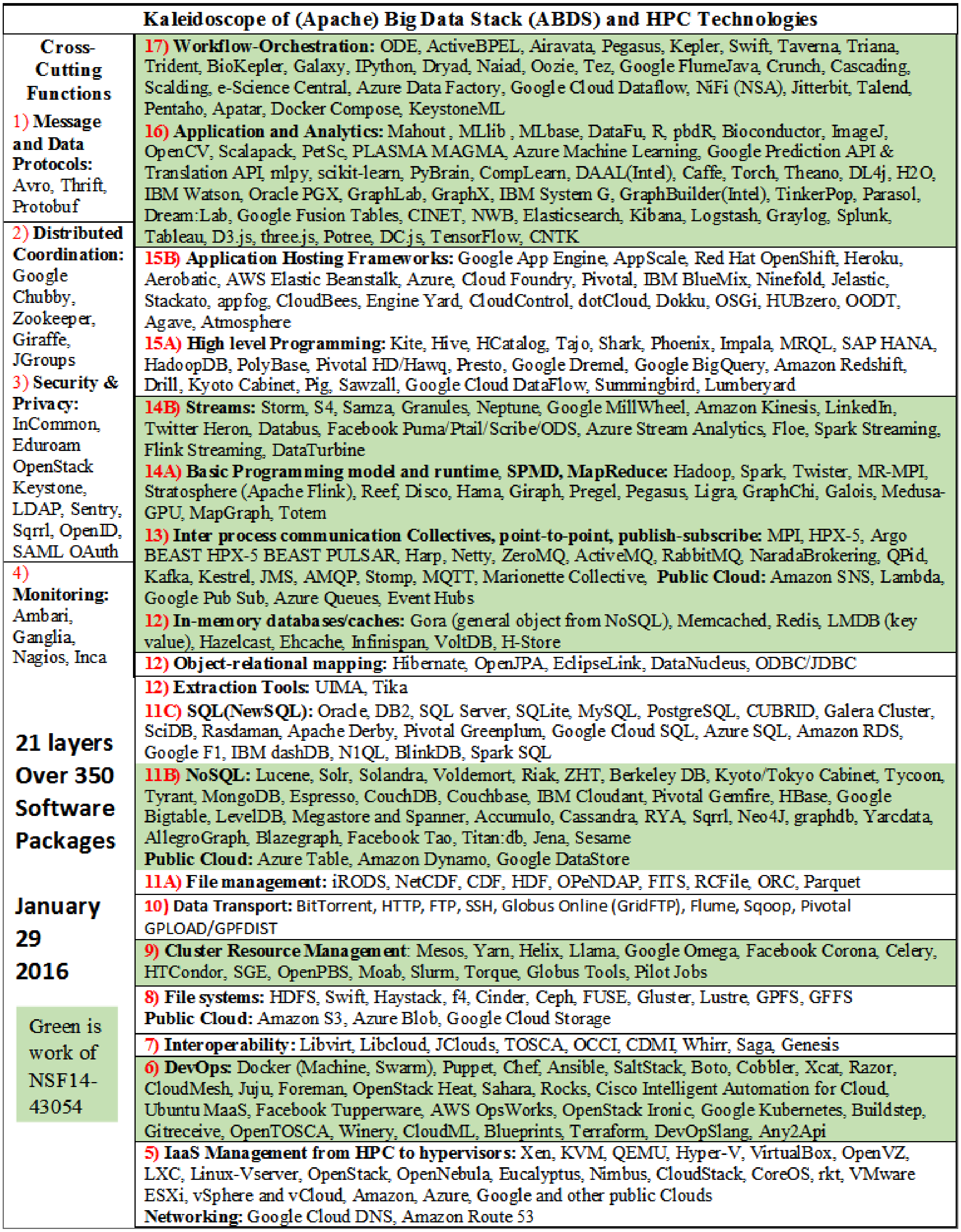
I524 OVERVIEW

This course studies software used in many commercial activities related to Big Data. The backdrop for course contains more than 370 software subsystems illustrated in Figure 1. We will describe the software architecture represented by this collection and work towards identifying best practices to deploy, access and interface with them. Topics of this class will include:

1. The cloud computing architecture underlying open source big data software and frameworks and contrast of them to high performance computing
2. The software architecture with its different layers covering broad functionality and rationale for each layer.
3. We will go through selected big data stack software from the list of more than 370
4. We will be identifying how we can create and replicate software environments based on software deployed and used on clouds while using Containers, OpenStack and Ansible playbooks.
5. Students will chose a number of open source members of the list each and create repeatable deployments as illustrated in class.
6. The main activity of the course will be building a significant project using multiple subsystems combined with user code and data. Projects will be suggested or students can chose their own. A project report will summarize the work conducted.
7. Topics taught in this class will be very relevant for industry as you are not only exposed to big data, but you will also be practically exposed to DevOps and collaborative code development tools as part of your homework and project assignment.

Students of this class will need to conduct their project deployments in python using ansible and enabling a software stack that is useful for a big data analysis. While it is not necessary to know either python or ansible to take the class it is important that you have knowledge of a programming language so you can enhance your knowledge on them throughout the class and succeed. You will be expected to have a computer on which you have python 2.7.x installed. You will be using chameleon and possibly our local cloud. Optionally some projects may use docker.

Figure 2 illustrates that you can follow the components of the class in a variety of ways and in parallel. For example, you do not have to wait to start the project or to find out more about any of the subsystems.



Kaleidoscope of (Apache) Big Data Stack (ABDS) and HPC Technologies	
Cross-Cutting Functions 1) **Message and Data Protocols:** Avro, Thrift, Protobuf	17) **Workflow-Orchestration:** ODE, ActiveBPEL, Airavata, Pegasus, Kepler, Swift, Taverna, Triana, Trident, BioKepler, Galaxy, IPython, Dryad, Naiad, Oozie, Tez, Google FlumeJava, Crunch, Cascading, Scalding, e-Science Central, Azure Data Factory, Google Cloud Dataflow, NiFi (NSA), Jitterbit, Talend, Pentaho, Apatar, Docker Compose, KeystoneML
	16) **Application and Analytics:** Mahout , MLlib , MLbase, DataFu, R, pbdR, Bioconductor, ImageJ, OpenCV, Scalapack, PetSc, PLASMA MAGMA, Azure Machine Learning, Google Prediction API & Translation API, mlpy, scikit-learn, PyBrain, CompLearn, DAAL(Intel), Caffe, Torch, Theano, DL4j, H2O, IBM Watson, Oracle PGX, GraphLab, GraphX, IBM System G, GraphBuilder(Intel), TinkerPop, Parasol, Dream:Lab, Google Fusion Tables, CINET, NWB, Elasticsearch, Kibana, Logstash, Graylog, Splunk, Tableau, D3.js, three.js, Potree, DC.js, TensorFlow, CNTK
2) **Distributed Coordination:** Google Chubby, Zookeeper, Giraffe, JGroups	15B) **Application Hosting Frameworks:** Google App Engine, AppScale, Red Hat OpenShift, Heroku, Aerobatic, AWS Elastic Beanstalk, Azure, Cloud Foundry, Pivotal, IBM BlueMix, Ninefold, Jelastic, Stackato, appfog, CloudBees, Engine Yard, CloudControl, dotCloud, Dokku, OSGi, HUBzero, OODT, Agave, Atmosphere
3) **Security & Privacy:** InCommon, Eduroam OpenStack Keystone, LDAP, Sentry, Sqrrl, OpenID, SAML OAuth	15A) **High level Programming:** Kite, Hive, HCatalog, Tajo, Shark, Phoenix, Impala, MRQL, SAP HANA, HadoopDB, PolyBase, Pivotal HD/Hawq, Presto, Google Dremel, Google BigQuery, Amazon Redshift, Drill, Kyoto Cabinet, Pig, Sawzall, Google Cloud DataFlow, Summingbird, Lumberyard
	14B) **Streams:** Storm, S4, Samza, Granules, Neptune, Google MillWheel, Amazon Kinesis, LinkedIn, Twitter Heron, Databus, Facebook Puma/Ptail/Scribe/ODS, Azure Stream Analytics, Floe, Spark Streaming, Flink Streaming, DataTurbine
	14A) **Basic Programming model and runtime, SPMD, MapReduce:** Hadoop, Spark, Twister, MR-MPI, Stratosphere (Apache Flink), Reef, Disco, Hama, Giraph, Pregel, Pegasus, Ligra, GraphChi, Galois, Medusa-GPU, MapGraph, Totem
4) **Monitoring:** Ambari, Ganglia, Nagios, Inca	13) **Inter process communication Collectives, point-to-point, publish-subscribe:** MPI, HPX-5, Argo BEAST HPX-5 BEAST PULSAR, Harp, Netty, ZeroMQ, ActiveMQ, RabbitMQ, NaradaBrokering, QPid, Kafka, Kestrel, JMS, AMQP, Stomp, MQTT, Marionette Collective, **Public Cloud:** Amazon SNS, Lambda, Google Pub Sub, Azure Queues, Event Hubs
	12) **In-memory databases/caches:** Gora (general object from NoSQL), Memcached, Redis, LMDB (key value), Hazelcast, Ehcache, Infinispan, VoltDB, H-Store
	12) **Object-relational mapping:** Hibernate, OpenJPA, EclipseLink, DataNucleus, ODBC/JDBC
	12) **Extraction Tools:** UIMA, Tika
21 layers Over 350 Software Packages	11C) **SQL(NewSQL):** Oracle, DB2, SQL Server, SQLite, MySQL, PostgreSQL, CUBRID, Galera Cluster, SciDB, Rasdaman, Apache Derby, Pivotal Greenplum, Google Cloud SQL, Azure SQL, Amazon RDS, Google F1, IBM dashDB, N1QL, BlinkDB, Spark SQL
	11B) **NoSQL:** Lucene, Solr, Solandra, Voldemort, Riak, ZHT, Berkeley DB, Kyoto/Tokyo Cabinet, Tycoon, Tyrant, MongoDB, Espresso, CouchDB, Couchbase, IBM Cloudant, Pivotal Gemfire, HBase, Google Bigtable, LevelDB, Megastore and Spanner, Accumulo, Cassandra, RYA, Sqrrl, Neo4J, graphdb, Yarcdata, AllegroGraph, Blazegraph, Facebook Tao, Titan:db, Jena, Sesame **Public Cloud:** Azure Table, Amazon Dynamo, Google DataStore
	11A) **File management:** iRODS, NetCDF, CDF, HDF, OPeNDAP, FITS, RCFile, ORC, Parquet
January 29 2016	10) **Data Transport:** BitTorrent, HTTP, FTP, SSH, Globus Online (GridFTP), Flume, Sqoop, Pivotal GPLOAD/GPFDIST
	9) **Cluster Resource Management:** Mesos, Yarn, Helix, Llama, Google Omega, Facebook Corona, Celery, HTCondor, SGE, OpenPBS, Moab, Slurm, Torque, Globus Tools, Pilot Jobs
Green is work of NSF14-43054	8) **File systems:** HDFS, Swift, Haystack, f4, Cinder, Ceph, FUSE, Gluster, Lustre, GPFS, GFFS **Public Cloud:** Amazon S3, Azure Blob, Google Cloud Storage
	7) **Interoperability:** Libvirt, Libcloud, JClouds, TOSCA, OCCI, CDMI, Whirr, Saga, Genesis
	6) **DevOps:** Docker (Machine, Swarm), Puppet, Chef, Ansible, SaltStack, Boto, Cobbler, Xcat, Razor, CloudMesh, Juju, Foreman, OpenStack Heat, Sahara, Rocks, Cisco Intelligent Automation for Cloud, Ubuntu MaaS, Facebook Tupperware, AWS OpsWorks, OpenStack Ironic, Google Kubernetes, Buildstep, Gitreceive, OpenTOSCA, Winery, CloudML, Blueprints, Terraform, DevOpSlang, Any2Api
	5) **IaaS Management from HPC to hypervisors:** Xen, KVM, QEMU, Hyper-V, VirtualBox, OpenVZ, LXC, Linux-Vserver, OpenStack, OpenNebula, Eucalyptus, Nimbus, CloudStack, CoreOS, rkt, VMware ESXi, vSphere and vCloud, Amazon, Azure, Google and other public Clouds **Networking:** Google Cloud DNS, Amazon Route 53

Fig. 1.1: Big data relevant technology layers

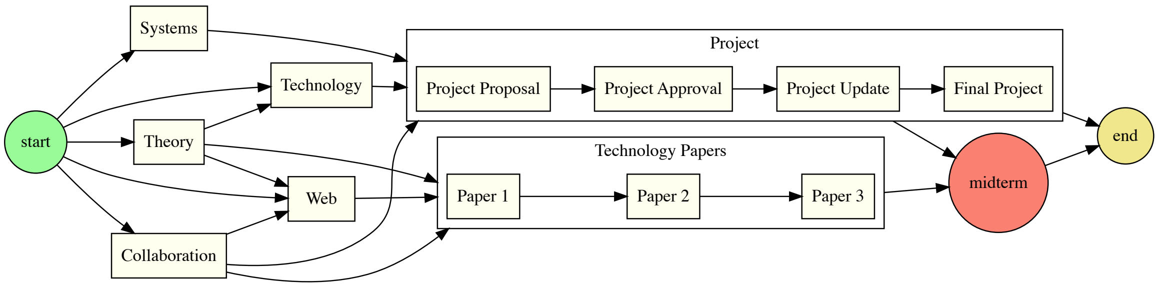


Figure 2: Components of the Class

Note: You do not have to take I523 in order to take I524.

For previous I523 class participants: While I523 is a beginners class I524 is a more advanced class and we expect that you know python which you hopefully have learned as part of I523 while doing a software project. If not, make sure you learn it before you take this class or consider **significant** additional time needed to learn it for the class.

Residential students need to enroll early so we avoid the situation like last year where we had many signing up, but did not even show up to the first lecture. I have asked that students from I523 have preference, but I am not sure if we can enforce this. So enroll ASAP. Those that are on the waiting list are recommended to show up in the first class. It is likely that you can join as others drop.

1.1 Meeting Times

The classes are published online. Residential students at Indiana University will participate in a discussion taking place at the following time:

- Monday 09:30am - 10:45am EST, I2 130

For the 100% online students see the office hours.

1.2 Online Meetings

For the zoom information please go to

https://iu.instructure.com/courses/1603897/assignments/syllabus

A doodle was used and all students that answered the doodle have times that they specified. We covered 100% the time for the students through the following schedule:

All times are in Eastern Standard Time.

Day of Week	Meetings
Monday	8-9am Office Hours 9:30-10:45am Residential Lecture 6-7pm Office Hours
Tuesday	1-2pm Office Hours 4-5pm Office Hours
Wednesday	6-7pm Office Hours
Thursday	6-7pm Office Hours (Gregor)
Friday	4-5pm Office Hours
Saturday	8-9pm Office Hours
Sunday	9-10am Office Hours 8-9pm Office Hours

1.3 Who can take the class?

- Although Undergrads can take this class it will be thaught as graduate class. Make sure you have enough time and fulfill the prerequisites such as knowing a programming language well. You need to have enough time to learn python if you do not know it.
- You can take I524 without taking I523, but you must be proficient in python. Overlap between I523 and I524 only relates to some introduction lectures and naturally lectures from the systems track such as github, report writing, introduction to python.
- Online students
- Residential students

1.4 Homework

Grading policies are listed in Table 1.

Table 1.1: Table 1: Grading

Percent	Description
10%	Class participation and contribution to Web pages.
30%	Three unique technology papers per student of the 370 systems. Each paper as at least 2 pages per technology without references.
60%	Project code and report with at least 6 pages without references. Much shorter reports will be returned without review. Do not artificially inflate contents.

- **Technology papers:** Technology papers must be non-overlapping in the entire class. As we have over 370 such technologies we should have enough for the entire class. If you see technologies missing, let us know and we see how to add them. Technology papers could be a survey of multiple technologies or an indepth analysis of a particular technology.

- **Technology paper groups:** Groups of up to three students can work also on the technology papers. However the workload is not reduced, you will produce 3 times the number of group members technology papers of unique technologies. However, you can have multiple coauthors for each paper (up to thre) that are part of your group. Please do not ask us how many technology papers you need to write if you are in a group. The rule is clearly specified. Example: Your group has 3 members, each of them has to procude 3 unique papers, thus you have to produce 9 unique technology papers for this group. If you have 2 members you have to produce 6, if you work alone you have to produce 3.
- **Technology deployment Homework:** Each student will develop as a preparation for the project a deployment of a technology. Points may depend on completeness, effort of the deployment. Technology deployments should as much as possible be non overlapping. In many cases you chose wisely such deployments may line up with your technology papers as you can add a section reporting on your achievement and experience with such deployments.
- **Project groups:** Groups of up to three students can work on a project but workload increases with each student and a work break down must be provided. More than three students are not allowed. If you work in a group you will be asked to deploy a larger system or demonstrate deployability on multiple clouds or container frameworks while benchmarking and comparing them. A group project containing 2 or 3 team members shoudl not look like a project done by an individual. Please plan careful and make sure all team members contribute.
- **Frequent checkins**: It is **important** to make frequent and often commits to the github repository as the activities will be monitored and will be integrated into the project grade. Note that paper and project 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.
- **No bonus projects:** This class will not have any bonus projects or regrading requests. Instead you need to focus your time on the papers and the project assignments and homework.
- **Voluntary work:** You are welcome to conduct assignments and excerises you find on the class Web page on your own. However they are not graded or considered for extra credit.
- **Late homework**: Any late homework will be receiving a 10% grade reduction. As this is a large class and the assignments are not standard multiple choice questions, grading will take a considerable time. Some homework can not be delivered late as they are related to establish communication with you. Such **deadline specific** homework will receive 0 points in case they are late. See course calendar. 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.
- **Chance for publishing a paper:** If however you find that the work you do could lead to a publishable paper, you could work together with the course instructor as coauthors to conduct such an activity. However, this is going to be a significant effort and you need to decide if you like to conduct this. In such cases if the work is sufficient for publication submission, an A+ for the class could be considered. It will be a lot of work. The length of such a paper is typically 10-12 high quality pages including figures and references. We may elect for the final submission to use a different LaTeX style

1.5 Prerequisites

We expect you are familiar with:

- Linux and the Operating system on which you will focus your deployment.
- Note that Windows as OS will not be sufficient as Ansible is not supported on it. However you can use virtualbox or log onto one of the clouds to get access to an OS that supports ansible. So you can use your Windows computer if it is powerful enough.

- Python 2.7.x (we will not use python 3 for this class as it is not yet portable with all systems) Although python is considered to be a straight forward language to learn, students that have not done any programming my find it challanging.
- Familaiarity with th Python eco system. The best way to install python on a computer is to use virtualenv, and pip (which we will teach you as part of the class).
- Familiarity with an editor such as emacs, vi, jedit, pyCharm, eclipse, or other that you can use to program in and write your reports.

If you are not familiar with these technologies, we expect that you get to know them before or during class. This may pose additional time commitment.

1.6 Open Source Publication of Homework

As this class is about open source technologies, we will be using such technologies to gather the homework submissions. We will not be using CANVAS so we teach you these technologies that are often mandated in industry. CANVAS is not.

As a consequence all technology papers from all students will be available as a single big technical report. To achieve this all reports must be written in the same format. This wil be LaTeX and all refernces have to be provided a bibtex file while you use jabref. Alternatively lyx.org can be used, if you prefer to edit latex in *what you see is almost what you get* format. The use of sharelatex or overleave or lyx.org is allowed.

1.7 Piazza

All communication will be done via Piazza. We will not read e-mail send to our university or private e-mails. All instructors are following this rule. Any mail that is not send via Piazza will be **not read** and **deleted**. This is also true for any mail send to the inbox system in CANVAS. We found CANVAS a not scalable solution for our class and will not use CANVAS for reaching out to you. If you need a different mechanism to communicate with us, please ask on Piazza how to do that. Please note that private posts in piazza are shared among all instructors and TAs.

To sign up in piazza please follow this link:

- https://piazza.com/iu/spring2017/i524

We have created a number of piazza folder to organize the posts int topics. Thes folders are:

help: Our help folder is just like a ticket system that is monitored by the TA's. Once you file a question here, a TA will attend to it, and work with you. Once the issue you had is resolved, the TA is marking it as resolved. If you need to reopen a help message, please mark it again as unresolved or post a follow up.

project: Questions related to projects are posted here.

logistics: Question regarding the logistics of the class are posted here. This includes questions about communication, meeting times, and other administrative activities.

papers: Questions regarding the paper are posted here.

grading: Questions regarding grades are posted here.

clouds: Questions regarding cloud resources are posted here.

faq: Some questions will be transformed to FAQ's that we post here. Note also that we have an FAQ on the class Web page that you may want to visit. We try to move important FAQ's from Piazza into the Web page, so it is important that you check both.

meetings: Here we will post times for meetings with TA's and instructors that are not yet posted on the Web page as part of the regular meeting times. Class participants are allowed to attend any Zoom meeting that we annonce in this folder. For online students we will also determine a time for regulare meetings. The TAs are required to hold 10 hours of meeting times upon request with you. Please make use of this.

other: In case no folder matches for your question use other.

1.8 Tips on how to achieve your best

While teaching our classes we noticed the following tips to achieve your best:

- Listening to the lectures
- **Set aside enough undisturbed time for the class**. Switch off facebook, twitter, or other distracting social media systems when focussing on the class.
- **Ask for help**. The TAs can schedule custom help office hours on appointment during reasonable times.
- Do not **Procrastinate**.
- Do not **take your other classes more serious**.
- **Start the project in the first 4 weeks of the class**
- Be aware that this class is not based on a text book and what this implies.
- Do not overestimating the technical abilities.
- Do not underestimating the time it takes to do the project.
- Do not forget to include benchmarks in your project.
- Unnecessarily struggling with LaTeX as you do not use an example we provide.
- Trying to do things just on Windows which is typically more difficult than using Linux.
- Not having a computer that is up to date. Update your memory and have a SSD
- Ignoring obvious security rules and not integrating ssh form the start into your projects.
- Not posting passwords into git. For example git does **not** allow to **easily** completely delete files that contain secret information such as passwords. It takes significant effort to do that. Make sure you do add in git on individual files and never just a bulk add.
- Having your coleagues do the work for you
- Underestimating the **time** it takes to do deployments
- Not reading our piazza posts and repeating the same question over and over
- Use Piazza to communicate and not CANVAS or e-mail.
- When you receive an e-mail from piazza, reply to it while clicking on the link instead of replying via e-mail directly. This is more reliable.

1.9 Submissions

Your papers and projects will be developed on GitHub and submitted using Pull Requests. The process is as follows:

1. fork the sp17-i524 repository.
2. clone your fork and commit and push your changes.

3. submit a pull request to the master branch of the origin repository.

See the repository for details on the individual assignments. As it will periodically be updated, make sure you are familiar with the process of Syncing a fork.

Some things to keep in mind:

- space on github is limited, so do not add datasets to the repository. Any datasets you use should be publicly hosted and deployed as part of your project ansible deployment scripts.
- never add ssh private keys to the repository. This results in a security risk, possible point deductions, and lots of time and effort to fix.
- all work will be licensed under the Apache 2 open source license.
- all submissions and discussion will be visible to the world.

1.10 Selected Project Ideas

Students can select from a number of project ideas. We will improve and add additional ideas throughout the semester. Students can also conduct their own projects. We recommend that you identify a project idea by the end of the first month of the class. Example project descriptions that you may want to take a look at include:

- robotswarm
- dockerswarm
- kubernetes
- slurmcluster
- authordisambiguity_b
- NIST Big Data Working group examples: Selected and approved use case from http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf
- Selected examples from Fall I523: Some students may have created an example as part of I523. Not all examples created as part of this class qualify for a I524 project. Please contact Gregor von Laszewski via Piazza to discuss suitability of your previous I523 project. If such a project is selected, approved and used it is expected it is significantly enhanced.
- Cloudmesh Enhancements: A number of projects could center around the enhancements of cloudmesh for the improvement of big data projects using virtual machines and containers. This includes:
  - Development of REST services for cloudmesh while using cloudmesh client
  - Development of benchmarking examples while using cloudmesh client
  - Development of a better Azure interface to additinal services
  - Development of a better AWS interfac to additinal services
  - Development of a Web interface while using django
  - SLURM integration to create virtual clusters on comet
  - Port cloudmesh client to Windows 10
  - Integrate docker into cloudmesh and demonstrate its use
  - Integrate kubernetes into cloudmesh and demonstrate its use
  - Expand the HPC capabilities of cloudmesh

1.11 Software Project

For a software project you have the choice of working indifidualy or working in a team of up to three students. You can use the **search teammate** folder to find and form groups:

- https://piazza.com/class/ix39m27czn5uw?cid=5

The following artifacts are part of the deliverables for a project

Code: You must deliver the 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 part of the project.

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

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

Work Breakdown: The report contains in an appendix a section that is only needed for team projects. Include in the section a short but sufficiently detailed work breakdown documenting what the team has done. Back it up with commit information from github. Such as how many commits and lines of code a team member has contributed. The section does not count towards the overall length of the paper.

In addition the graders will check the history of checkins to verify each team member has used github to checkin their contributions frequently. E.g. if we find that one of the students has not checked in code or documentation in the same way at other teammates, it will be questioned. An oral exam may be scheduled to verify that the student has contributed to the project. In an oral exam the student must be familiar with **all** aspects of the project not just the part you contributed.

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.md file which packages are used and which license these packages have while adding a licenses file.

Additional links:

- projects

Reproducability: The reproducability of your code will be tested twice. It is tetes by another student or team, it is also tested by a TA. A report of the testing team is provided. Your team will also be responsible for executing as many tests as you have team members on other projects. A reproducability statement should be written with details about functionality, readbility, and report quality. This statement does not have to be written in latex but uses RST.

1.12 Report Format

All reports will be using the format specified in Section *Report Format*.

There will be **NO EXCEPTION** to this format. Documents not following this format and are not professionally looking, will be returned without review.

1.13 Github repositories

Class content repository: https://github.com/cloudmesh/classes

Class homework repository: https://github.com/cloudmesh/sp17-i524

1.14 Code Repositories Deliverables

Code repositories are for code, if you have additional libraries or data that are needed you need to develop a script or use a DevOps framework to install such software. They **must** not be checked into github. Thus zip files and .class, .o, precompiled python, .exe, core dumps, and other such files files are not permissible in the project. If we find such files you will get a 20% deduction in your grade. 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. Naturally you can use ansible or shell scripts. It is not permissible to use GUI based DevOps preinstalled frameworks (such as the one you may have installed in your company or as part of another project). Everything must be installable form the command line.

1.15 Learning Outcomes

Students will

1. gain broad understanding of Big Data applications and open source technologies supporting them.
2. have intense programming experience in Python and ansible and DevOps.
3. use open source technologies to manage code in large groups of individuals.
4. be able to communicate reserach in professional scientific reports.

Outcome 1 is supported by a series of lectures around open source technologies for big data.

Outcome 2 is supported by a significant software project that will take up a considerable amount of time to plan and execute.

Outcome 1 and 4 is supported by writing 3 technology papers and a project report that is shared with all students. Students can gain additional insight from reading and reviewing other students contributions.

Outcome 3 is supported by using piazza and github as well as contributiong to the class Web page with git pull requests.

1.16 Academic Integrity Policy

We take academic integrity very seriously. You are required to abide by the Indiana University policy on academic integrity, as described in the Code of Student Rights, Responsibilities, and Conduct, as well as the Computer Science Statement on Academic Integrity (http://www.soic.indiana.edu/doc/graduate/graduate-forms/Academic-Integrity-Guideline-FINAL-2015.pdf). It is your responsibility to understand these policies. Briefly summarized, the work you submit for course assignments, projects, quizzes, and exams must be your own or that of your group, if group work is permitted. You may use the ideas of others but you must give proper credit. You may discuss assignments with other students but you must acknowledge them in the reference section according to scholarly citation rules. Please also make sure that you know how to not plagiarize text from other sources while reviewing citation rules.

We will respond to acts of plagiarism and academic misconduct according to university policy. Sanctions typically involve a grade of 0 for the assignment in question and/or a grade of F in the course. In addition, University policy requires us to report the incident to the Dean of Students, who may apply additional sanctions, including expulsion from the university.

Students agree that by taking this course, papers and source code submitted to us may be subject to textual similarity review, for example by Turnitin.com. These submissions may be included as source documents in reference databases for the purpose of detecting plagiarism of such papers or codes.

It is not acceptable to use for pay services to conduct your project. Please be aware that we monitor such services and have TAs speaking various languages and know about these services even in different countries. Also do not just translate a report written by someone in a different language and claim it to be your project.

1.17 Links

This page is conveniently managed with git. The location for the changes can be found at

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

The repository is at

- https://github.com/cloudmesh/classes

Issues can be submitted at

- https://github.com/cloudmesh/classes/issues

Or better use piazza so you notify us in our discussion lists.

- https://piazza.com/iu/i524

If you detect errors, you could also create a merge request at

- https://github.com/cloudmesh/classes

1.18 Course Numbers

This course is offered for Graduate (and Undergraduate students with permission) at Indiana University and as an online course. To Register, for University credit please go to:

- http://registrar.indiana.edu/browser/soc4172/INFO/INFO-I524.shtml
- http://registrar.indiana.edu/browser/soc4172/ENGR/ENGR-E599.shtml

Please, select the course that is most suitable for your program:

```
INFO-I 524  BIG DATA SOFTWARE AND PROJECTS (3 CR)    Von Laszewski G
         Above class open to graduates only
         Above class taught online
         Discussion (DIS)
      30672 RSTR     09:30A-10:45A   M      I2 130    Von Laszewski G
         Above class meets with ENGR-E 599
INFO-I 524  BIG DATA SOFTWARE AND PROJECTS (3 CR)
      30673 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

ENGR-E 599  TOPICS IN INTELL SYS ENGINEER (3 CR)
      VT: BIG DATA SOFTWARE AND PROJECTS
         ***** RSTR     ARR             ARR    ARR       Von Laszewski G
            Discussion (DIS)
      VT: BIG DATA SOFTWARE AND PROJECTS
         33924 RSTR     09:30A-10:45A   M      I2 130    Von Laszewski G
            Above class meets with INFO-I 524
```

1.19 Refernces

http://hpc-abds.org/kaleidoscope/

CHAPTER

TWO

I524 CALENDAR

Warning: This calendar may be updated based on experience from the class. Please check back here.

Residential classes meet Mondays 09:30A-10:45A, I2 130

Description	**Date/Due Dates**	No
Class Begins	Mon, Jan 9	
Assignment of HID	Fri, Jan 13	
Piazza access verified (HD)	Mon, Jan 16, 9am	C1
Surveys (HD)	Mon, Jan 16, 9am	C2
MLK Jr. Day. Good time for additional class work	Mon, Jan 16	
Github access (needed for TechList)	Mon, Jan 30, 9am	C3
Acces and use of cloud verified (HD)	Mon, Jan 30, 9am	C4
Acces to python 2.7.x verified (HD)	Mon, Jan 30, 9am	C5
TechList.1a - 1.c	Mon, Feb 15, 9am	T0a
Technology Paper 1	Mon, Feb 27, 9am	T1
TechList.1d and Techlist 2	Mon, Feb 27, 9am	T0b
Technology Paper 2 due	Mon, Feb 27, 9am	T2
Technology Paper 3 due	Mon, Mar 27, 9am	T3
Auto Witdraw	Sun, Mar 12	
Project Execution plan and draft due (HD)	Mon, Mar 13, 9am	P1
Spring Break. Good time for additional class work	Mar 12 - Mar 19	
Project Updates due (HD)	Mon, Mar 27, 9am	P2
Project due	Mon, Apr 24, 9am	P3
Last day to submit late Homework	May 1	
Ends	Fri, May 5	

- (HD) hard deadlines must be done in order to obtain full points. These deadlines are important to assure you have access to the resources for the class.

2.1 Comments

- Any late homework will have an automatic 10% grade deduction.
- Any late homework may result in substential delay in grading (one month or more).
- Hard deadlines can not receive any points for late submissions as they are essential to the communication and operation of the class. If you can naturally not communicate with we can not review your work or you can not even execute your work.

- Experience shows that those using additional time during the spring break do typically better. We recommend that you use this time wisely.
- You can start earlier if you like to prepare for this class, to for example learn Python and ansible. However, lectures may change.
- It would be a mistke not to start working on your project by February 1st. You will run out of time. In order to accomodate for this we have segnificantly reduced other homework requirements in contrast to previous classes we taught.

2.2 Official University calendar

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

CHAPTER

THREE

I524 LECTURES

Warning: This page is under construction, but most lectures are already available. All tracks will change considerably. If you want to work ahead, start with the theory track.

Warning: Lectures listed on this page will be ready for review when they are marked as released. If they are not released they will be updated as part of the class. However instead of waiting for the release, we have opted to show you our current draft lectures.

Based on our experience with residential and online classes we will for the first time not require that you have to do the class videos at a prticular time once they are released. This however has the danger that you are not watching them at all and you cheat yourself as you do not allow yourself the educational lessons that this class offers to you. It also requires you to assemble your own schedule for watching the videos that will have to be managed through github as part of a README.md file in your git repository. You will need to do the technology track, the communications track, as well as the theory track.

Theory Track: Some lectures have been designed to introduce you to a number of technologies. These lectures are of more theoretical nature and do not require much hands on activities. THus you can start them any time.

Collaboration Track: These lectures provide the tools for you to collaborate with your peers and with instructors.

Systems Track: These lectures cover topics that are fundamental to executing your project.

Technology Track: These are lectures with strong technology content and introduce you to using a selected number of technologies as part of the class. It is expected that you will use them as part of the project. Instead of slowing you down with graded homework we expect that you learn these technologies and reuse them as part of the project. It would be a big mistake to start the project 2 weeks befor the semester ends, you will not succeed. You must start your project in the first month of the course. Progress is reported on monthly basis while the report is updated and snapshot every month. We will monitor your progress and include them into the discussion grade. For residential students ther should be no reason why you can not provide a monthly update. For online students a valid update would be: “I changed my company and could not work on the project due to moving”. This will give you some points if submitted in time. However, if you submit nothing, we will not issue any points.

3.1 Lectures

3.2 Lectures - Theory Track

Table 3.1: Theory Track

Topic	Description	Resources	Length
Overview	Course Overview	Slides	
	Class Overview - Part 1	Video	11:29
	Class Overview - Part 2	Video	04:10
	Class Overview - Part 3	Video	12:41
Web Page	Course Web Page	**Videos_**	
	Class Web Page - Part 1	Video	11:25
	Class Web Page - Part 2	Video	17:31
Techlist.1	TechList.1 Web Page	Web Page	
	TechList.1 Homework	Video	40:08
Introduction	Course Introduction	Slides	
	Introduction	Video	0:13:59
	Introduction - Real World Big Data	Video	0:15:28
	Introduction - Basic Trends and Jobs	Video	0:10:57
Acess Patterns	Data Access Patterns and Introduction to using HPC-ABDS	Slides	
	1. Introduction to HPC-ABDS Software and Access Patterns	Video - Resource 1	0:27:45
	2. Science Examples (Data Access Patterns)	Video - Resource 2	0:18:38
	3. Remaining General Access Patterns	Video	0:11:26
	4. Summary of HPC-ABDS Layers 1 - 6	Video	0:14:32
	5. Summary of HPC-ABDS Layers 7 - 13	Video	0:30:52
	6. Summary of HPC-ABDS Layers 14 - 17	Video	0:28:02
	Final Part Summary of Stack	Video	0:20:20

3.3 Lectures - Collaboration Track

Table 3.2: Collaboration Track

Topic	Description	Resources	Length
Organization	Lessons vs Lectures	Web Page	
Web Page	Contributiing to the Web Page	Web Page	
Github	Overview and Introduction	Web page	
	Install Instructions	Web page	
	config	Video	2:47
	fork	Video	1:41
	checkout	Video	3:11
	pull	Video	4:26
	branch	Video	2:25
	merge	Video	4:50
	rebase	Video	4:20
	GUI	Video	3:47
	Windows - unsupported	Video	1:25
Paper	How to write a paper by Simon Peyton Jones	Video	34:24
	LaTeX - Overview of LaTeX Resources	Web Page	
	(optional) ShareLaTeX	Video	8:49
	jabref	Video	14:41
	Report Format	Web Page - Git - PDF	

3.4 Lectures - Systems

Table 3.3: Systems Track

Treat	Quantity	Description	Length
Ubuntu	Development OS for the class	Web page	
Virtualbox	Virtualbox for class	Web page	
	Instalation of ubuntu in virtualbox	Video	
	Instalation of guest additions in virtualbox	Video	
Shell	Linux Shell	Web Page	
Python	(Draft) Introduction to Python	Web page	
	(Draft) Python for Big Data	Web Page	
	(Draft - Advanced) Python Fingerprint example	Web page	
	PyCharm	Video	

3.5 Unreleased Lectures

A list of unrelease lectures that we are currently working on is available here: ref-unreleased

CHAPTER

FOUR

HID ASSIGNMENT

As part of the class you will be assigne a Homework ID (HID). Some assignments in the class will use this HID to identify which homework you will be doing. Technologies listed with (1) behind it are for the homework TechList.1 and Technologies with a (2) are for TechList.2

Note: The folloing list is the original assignment of the technologies to HIDs. The mapping from the HID to names is stored at this time in Piazza at https://piazza.com/class/ix39m27czn5uw?cid=33 please make sure we did not make a mistake and if so, please notify us.

Table 4.1: Mappings of HIDs to Techs

Name	HID	Technologies
Sheybani Moghadam Saber	S17-ER-1001	Azure Queues (1) – Sentry (1) – Tableau (1) – Berkeley DB (1) – ODE (1) – OpenStack Keystone (1) – Globus Tools (2)
	•	•
Agasti Avadhoot	S17-IO-3000	SQL Server (1) – Nimbus (1) – Taverna (1) – Chef (1) – Tyrant (1) – FITS (1) – DataTurbine (2)
Bays Christopher	S17-IO-3002	TensorFlow (1) – Azure Stream Analytics (1) – Ambari (1) – Galaxy (1) – Bioconductor (1) – OPeNDAP (2) – BlinkDB (2)
Carmickle Ricky	S17-IO-3003	QPid (1) – Stomp (1) – Apatar (1) – Google FlumeJava (1) – Sqrrl (1) – Scalding (2) – OSGi (2)
Coulter Cory	S17-IO-3004	appfog (1) – Dream:Lab (1) – MySQL (1) – ZHT (1) – RYA (1) – Summingbird (2) – SQLite (2)
Gupta Abhishek	S17-IO-3005	Amazon Kinesis (1) – Inca (1) – Gora (general object from NoSQL) (1) – RabbitMQ (1) – JClouds (1) – Megastore and Spanner (2) – Any2Api (2)

Continued on next page

Table 4.1 – continued from previous page

Name	HID	Technologies
Kodre Vishwanath	S17-IO-3008	CINET (1) – Linux-Vserver (1) – Networking: Google Cloud DNS (1) – Talend (1) – Haystack (1) – PolyBase (2) – Docker (Machine, Swarm) (2)
Kshirsagar Hemant	S17-IO-3009	Flink Streaming (1) – Solr (1) – JGroups (1) – Azure SQL (1) – HDF (1) – Torque (2) – Databus (2)
Lawson Matthew	S17-IO-3010	Azure (1) – Couchbase (1) – Public Cloud: Azure Table (1) – Sawzall (1) – Phoenix (1) – CouchDB (2) – Disco (2)
McClary Scott	S17-IO-3011	ZeroMQ (1) – Blueprints (1) – Trident (1) – e-Science Central (1) – Winery (1) – Crunch (2) – Airavata (2)
McCombe Mark	S17-IO-3012	MRQL (1) – AWS OpsWorks (1) – GPFS (1) – Hazelcast (1) – Google Bigtable (1) – Google Prediction API & Translation API (2)
Mwangi Leonard	S17-IO-3013	Google Prediction API and Translation API (1) – LMDB (key value) (1) – QEMU (1) – BioKepler (1) – Google Cloud Dataflow (1) – Pregel (2)
Rai Piyush	S17-IO-3014	Riak (1) – Ehcache (1) – Xen (1) – Zookeeper (1) – SSH (1) – SciDB (2)
Roy Choudhury Sabyasachi	S17-IO-3015	Lucene (1) – pbdR (1) – Protobuf (1) – Galera Cluster (1) – Cassandra (1) – Mbase (2)
Rufael Ribka	S17-IO-3016	DC.js (1) – Aerobatic (1) – CoreOS (1) – AMQP (1) – Argo BEAST HPX-5 BEAST PULSAR (1) – Apache Derby (2)
Sathe Nandita	S17-IO-3017	Facebook Tao (1) – MongoDB (1) – Amazon (1) – Kafka (1) – Amazon Dynamo (1) – Blazegraph (2)
Shane Kevin	S17-IO-3018	PostgreSQL (1) – Impala (1) – Hadoop (1) – Floe (1) – VirtualBox (1) – Ubuntu MaaS (2) – Pig (2)
Smith Michael	S17-IO-3019	Stackato (1) – Parasol (1) – vSphere and vCloud (1) – Totem (1) – Libvirt (1) – Xcat (2) – InCommon (2)
Suryawanshi Milind	S17-IO-3020	AppScale (1) – CloudControl (1) – Google Fusion Tables (1) – Yarn (1) – TinkerPop (1) – LinkedIn (2) – CloudML (2)

Continued on next page

Table 4.1 – continued from previous page

Name	HID	Technologies
Thakre Abhijit	S17-IO-3021	CUBRID (1) – MR-MPI (1) – NWB (1) – Cascading (1) – BitTorrent (1) – Tez (2) – Rocks (2)
Unni Sunanda	S17-IO-3022	Juju (1) – Netty (1) – FUSE (1) – Google Chubby (1) – Mesos (1) – Pivotal GPLOAD/GPFDIST (2) – Yarcdata (2)
Venkatesan Karthick	S17-IO-3023	H-Store (1) – Kyoto Cabinet (1) – Globus Online (GridFTP) (1) – Sahara (1) – DataFu (1) – Facebook Tupperware (2) – Lambda (2)
Vuppada Ashok	S17-IO-3024	NiFi (NSA) (1) – LXC (1) – Helix (1) – IBM dashDB (1) – Puppet (1) – Google Cloud SQL (2) – Giraph (2)
Yezerets Helen	S17-IO-3025	Voldemort (1) – Buildstep (1) – OCCI (1) – SAP HANA (1) – HPX-5 (1) – IPython (2) – CloudMesh (2)
	•	•
Akurati Niteesh Kumar	S17-IR-2001	Celery (1) – GraphBuilder(Intel) (1) – HTCondor (1) – HUBzero (1) – Gitreceive (1) – Pivotal Greenplum (2) – Infinispan (2)
ARDIANSYAH JIMMY	S17-IR-2002	Stratosphere (Apache Flink) (1) – ActiveBPEL (1) – Google Dremel (1) – ImageJ (1) – IBM Cloudant (1) – Kepler (2) – Amazon Redshift (2)
Balaga Ajit	S17-IR-2004	PLASMA MAGMA (1) – Samza (1) – Azure Blob (1) – OpenVZ (1) – Jelastic (1) – Jupyter (2) – Kibana (2)
Chemburkar Snehal Shrish	S17-IR-2006	Cinder (1) – Spark (1) – R (1) – dotCloud (1) – Pivotal Gemfire (1) – PyBrain (2) – Engine Yard (2)
Anbazhagan Karthik	S17-IR-2008	Kestrel (1) – Scalapack (1) – HadoopDB (1) – OODT (1) – Thrift (1) – Mahout (2) – Moab (2)
Jain Anurag Kumar	S17-IR-2011	DL4j (1) – Solandra (1) – CloudStack (1) – Logstash (1) – Ansible (1) – Hyper-V (2) – Swift (2)
Jain Pratik	S17-IR-2012	GraphLab (1) – GFFS (1) – Lustre (1) – Reef (1) – Harp (1) – LevelDB (2) – Event Hubs (2)
Korrapati Sahiti	S17-IR-2013	Flume (1) – OpenCV (1) – ORC (1) – VMware ESXi (1) – Hama (1) – DevOpSlang (2) – Accumulo (2)

Continued on next page

Table 4.1 – continued from previous page

Name	HID	Technologies
Krishnakumar Harshit	S17-IR-2014	Sqoop (1) – Pivotal (1) – Google MillWheel (1) – iRODS (1) – VoltDB (1) – OpenPBS (2) – Kite (2)
Lingampalli Anvesh Nayan	S17-IR-2016	ActiveMQ (1) – OpenTOSCA (1) – Avro (1) – SaltStack (1) – Whirr (1) – MLlib (2) – GraphChi (2)
Marni Veera	S17-IR-2017	Eduroam (1) – Potree (1) – Pivotal HD/Hawq (1) – Docker Compose (1) – OpenNebula (1) – point-to-point (2) – Neptune (2)
Merugureddy Bhavesh Reddy	S17-IR-2018	CompLearn (1) – OpenID (1) – Cisco Intelligent Automation for Cloud (1) – Pentaho (1) – scikit-learn (1) – Google and other public Clouds (2) – Llama (2)
Methkupalli Vasanth	S17-IR-2019	Oracle (1) – CNTK (1) – Twister (1) – NetCDF (1) – Oozie (1) – KeystoneML (2) – Lumberyard (2)
Mishra Govind	S17-IR-2021	Docker Machine and Swarm (1) – Shark (1) – Ligra (1) – Redis (1) – Facebook Puma/Ptail/Scribe/ODS (1) – AWS Elastic Beanstalk (2) – Facebook Corona (2)
Naik Abhishek	S17-IR-2022	Sesame (1) – Pilot Jobs (1) – Red Hat OpenShift (1) – Google Pub Sub (1) – Boto (1) – Triana (2) – IBM System G (2)
Parekh Ronak	S17-IR-2024	Cobbler (1) – GraphX (1) – Memcached (1) – graphdb (1) – LDAP (1) – Spark SQL (2) – Splunk (2)
Raghatate Rahul	S17-IR-2026	Ceph (1) – CDF (1) – Jitterbit (1) – Naiad (1) – publish-subscribe: MPI (1) – Google F1 (2) – NaradaBrokering (2)
Ramachandran Shahidhya	S17-IR-2027	DataNucleus (1) – Razor (1) – Twitter Heron (1) – Amazon RDS (1) – SAML OAuth (1) – (Dryad) (2) – DB2 (2)
Ramanam Srikanth	S17-IR-2028	Spark Streaming (1) – Libcloud (1) – Google Kubernetes (1) – mlpy (1) – Dokku (1) – N1QL (2) – PetSc (2)
Ramaraju Naveenkumar	S17-IR-2029	Galois (1) – Slurm [Slu] (1) – Giraffe (1) – Azure Machine Learning (1) – Ninefold (1) – CDMI (2) – OpenStack Ironic (2)
Ravi Sowmya	S17-IR-2030	UIMA (1) – Jena (1) – Tycoon (1) – Azure Data Factory (1) – Google Cloud DataFlow (1) – Medusa-GPU (2) – Neo4J (2)

Continued on next page

Table 4.1 – continued from previous page

Name	HID	Technologies
Satyam Kumar	S17-IR-2031	Google Cloud Storage (1) – EclipseLink (1) – Torch (1) – Caffe (1) – Parquet (1) – Rasdaman (2) – DAAL(Intel) (2)
Sharma Yatin	S17-IR-2034	rkt (1) – Heroku (1) – Pegasus (1) – Drill (1) – Titan:db (1) – OpenStack (2) – Espresso (2)
Shinde Piyush	S17-IR-2035	ODBC/JDBC (1) – f4 (1) – Oracle PGX (1) – Eucalyptus (1) – D3.js (1) – Ganglia (2) – Amazon Route 53 (2)
Singh Rahul	S17-IR-2036	OpenStack Heat (1) – Saga (1) – Agave (1) – Storm (1) – JMS (1) – Graylog (2) – Google App Engine (2)
Sitharaman Sriram	S17-IR-2037	Public Cloud: Amazon SNS (1) – FTP (1) – HBase (1) – MQTT (1) – RCFile (1) – OpenJPA (2) – SGE (2)
Sivaprasad Sushmita	S17-IR-2038	Terraform (1) – H2O (1) – KVM (1) – Cloud Foundry (1) – CloudBees (1) – Marionette Collective (2) – three.js (2)
Suri Naren	S17-IR-2039	TOSCA (1) – HTTP (1) – IBM BlueMix (1) – Google Omega (1) – Gluster (1) – Google DataStore (2) – MapGraph (2)
Vora Sagar	S17-IR-2041	IBM Watson (1) – Public Cloud: Amazon S3 (1) – Kyoto/Tokyo Cabinet (1) – Elasticsearch (1) – Tajo (1) – Google BigQuery (2) – S4 (2)
Yadav Diksha	S17-IR-2044	AllegroGraph (1) – Theano (1) – Atmosphere (1) – Granules (1) – HDFS (1) – Hibernate (2) – Hive (2)
	•	•
TA	S17-TS-0001	Mahout (1)
TA	S17-TS-0001	Tika (1)
TA	S17-TS-0003	HCatalog (1)
TA	S17-TS-0004	Foreman (1)
TA	S17-TS-0005	Genesis (1)
TA	S17-TS-0006	Presto (1)
TA	S17-TS-0007	Nagios (1)

CHAPTER

FIVE

TECHNOLOGIES

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

5.1 Workflow-Orchestration

1. ODE
2. ActiveBPEL S17-IR-2002
3. Airavata
4. Pegasus
5. Kepler
6. Swift
7. Taverna

   Taverna is workflow management system. According to *[Tav]*, 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 his *[tav07]* paper, Daniele Turi presented the formal syntax and operational semantics of Taverna.
8. Triana
9. Trident
10. BioKepler
11. Galaxy
12. IPython
13. Jupyter
14. (Dryad)
15. Naiad
16. Oozie

17. Tez
18. Google FlumeJava
19. Crunch
20. Cascading
21. Scalding
22. e-Science Central
23. Azure Data Factory
24. Google Cloud Dataflow
25. NiFi (NSA)
26. Jitterbit
27. Talend
28. Pentaho
29. Apatar
30. Docker Compose
31. KeystoneML

5.2 Application and Analytics

32. Mahout *[Mah]*

    "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"

33. MLlib
34. Mbase
35. 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 *[Dat]* 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.

36. R
37. pbdR

    Programming with Big Data in R (pbdR) *[OCSP12]* 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 accross 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. Currenly there are two implementation of pbdR, one Rmpi and another being pdbMpi. Rmpi uses SPMD parallelism while pbdRMpi uses manager/worker parallelism.

38. Bioconductor
39. ImageJ
40. OpenCV
41. Scalapack
42. PetSc
43. PLASMA MAGMA
44. Azure Machine Learning
45. Google Prediction API & Translation API
46. mlpy
47. scikit-learn
48. PyBrain
49. CompLearn
50. DAAL(Intel)
51. Caffe
52. Torch
53. Theano
54. DL4j
55. H2O
56. IBM Watson
57. Oracle PGX
58. GraphLab
59. GraphX
60. IBM System G
61. GraphBuilder(Intel)
62. TinkerPop
63. Parasol
64. Dream:Lab
65. Google Fusion Tables
66. CINET
67. NWB
68. Elasticsearch
69. Kibana
70. Logstash
71. Graylog

72. Splunk
73. Tableau
74. D3.js
75. three.js
76. Potree
77. DC.js
78. TensorFlow
79. CNTK

5.3 Application Hosting Frameworks

80. Google App Engine *[App]*

    On purpose we put in here a "good" example of a bad entry that woudl receive 10 out of 100 points, e.g. an F:

    "Google App Engine" provides platform as a service. There are major advantages from this framework:

    (a) Scalable Applications

    (b) Easier to maintain

    (c) Publishing services easily

    Reasons: (a) "major advantages is advertisement" if you add word major (b) grammar needs to be improved (c) the three points do not realy say anything about Google App Engine (d) the reader will after reading this have not much information about what it is (e) a refernce is not included. (f) enumeration should be in this page avoided. We like to see a number of paragraphs with text.

    Note: This is an example for a bad entry

81. AppScale
82. Red Hat OpenShift
83. Heroku
84. Aerobatic
85. AWS Elastic Beanstalk
86. Azure

    Microsoft Corporation 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. The Wikipedia entry on IaaS notes that IaaS "abstract[s] the user from the details of infrasctructure like physical computing resources, location, data partitioning, scaling, security, backup, etc." :cite:www-wikipedia-cloud

    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 *[MicrosoftCorp]*. For instance, Azure users can access predictive analytics, Bots and Blockchain-as-a-Service :cite:www-azure-msft as well as more-basic computing, networking, storage, database and management components *[MicrosoftCorp16]*. 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*, :cite:www-aws-amzn even though IBM (*SoftLayer* :cite:www-softlayer-ibm and *Bluemix* :cite :*www-bluemix-ibm*) and Google (*Google Cloud Platform*) [www-cloud- google] offer IaaS to the market. As of January 2017, Azure's datacenters span 32 Microsoft-defined *regions*, or 38 *declared regions*, throughout the world. *[MicrosoftCorp]*

87. Cloud Foundry
88. Pivotal
89. IBM BlueMix
90. (Ninefold)

    no longer active

91. Jelastic
92. Stackato
93. appfog
94. CloudBees
95. Engine Yard
96. (CloudControl)

    No Longer active as of Feb. 2016

97. dotCloud
98. Dokku
99. OSGi
100. HUBzero
101. OODT
102. Agave
103. Atmosphere

5.4 High level Programming

104. Kite
105. Hive
106. HCatalog
107. Tajo
108. Shark
109. 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 accessiblity 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. *[Kes13]* Although Salesforce.com initially *open-sourced* it via Github, by May of 2014 it had become a top-level Apache project. *[Anonb]*

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." *[Anona]* 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. [www-phoenix- salesforcedev]

Despite adding an abstraction layer, Phoenix can actually speed up queries because it optimizes the query during the translation process. [www- phoenix-cloudera] For example, "Phoenix beats Hive for a simple query spanning 10M-100M rows." *[Avr13]*

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. [www-phoenix- bighadoop]

110. Impala

111. MRQL

112. SAP HANA

   As noted in [www-sap-hana], 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 *[Olo14]*. It provides extensive administrative, security features and data access that ensures high data availability, data protection and data quality.

113. HadoopDB

114. PolyBase

115. Pivotal HD/Hawq

116. Presto

   Presto *[wwwa]* 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 *[CQB+14]*.

117. Google Dremel

118. Google BigQuery

119. Amazon Redshift

120. Drill

121. Kyoto Cabinet

   Kyoto Cabinet as specified in *[Kyo]* 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).

122. Pig

123. Sawzall
124. Google Cloud DataFlow
125. Summingbird
126. Lumberyard

5.5 Streams

127. Storm
128. S4
129. Samza
130. Granules
131. Neptune
132. Google MillWheel
133. Amazon Kinesis

     Kinesis is Amazon's *[wwwe]* 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 is 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 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 *[SS16]* in the area of real-time processing provides following key benefits: ease of use, parellel processing, scalable, cost effective, fault tolerant and highly available.

134. LinkedIn
135. Twitter Heron
136. Databus
137. Facebook Puma/Ptail/Scribe/ODS
138. Azure Stream Analytics
139. Floe
140. Spark Streaming
141. Flink Streaming
142. DataTurbine

5.6 Basic Programming model and runtime, SPMD, MapReduce

143. Hadoop
144. Spark
145. Twister
146. MR-MPI
147. Stratosphere (Apache Flink)
148. Reef
149. Disco
150. Hama
151. Giraph
152. Pregel
153. Pegasus
154. Ligra
155. GraphChi
156. Galois
157. Medusa-GPU
158. MapGraph
159. Totem

5.7 Inter process communication Collectives

160. point-to-point
161. publish-subscribe: MPI
162. HPX-5

    Based on :cite:‘ www-hpx-5 , *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 :cite:‘www-hpx-5-user-guide.*

163. Argo BEAST HPX-5 BEAST PULSAR
164. Harp
165. Netty
166. ZeroMQ
167. ActiveMQ

168. RabbitMQ

     RabbitMQ is a message broker *[wwwg]* 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:

     (a) Simple messaging: producers write messages to the queue and consumers read messages from the the queue. This is synonymous to a simple message queue.

     (b) Producer-consumer: Producers produce messages and consumers receive messages from the queue. The messages are delivered to multiple consumers in round robin manner.

     (c) 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.

     (d) Routing: In this mode consumers can subscribe to a subset of messages instead of receiving all messages from the queue.

     (e) 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.

     (f) 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 *[OHara07]* (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"

169. NaradaBrokering

170. QPid

171. Kafka

     Apache Kafka is a streaming platform, which works based on publish-subscribe messaging system and supports distributed environment. Kafka lets publish and subscribe to the messages.

     In a publish-subscribe messaging system, publishers are sender of messages. They publish the messages without the knowledge of who is going to 'subscribe' to them for processing. Subscribers are users of these messages. They subscribe to only those messages which they are interested in, without knowing who the publishers are. Kafka maintains message feeds based on 'topic'. A topic is a category or feed name to which records are published. Applications can use Kafka's Connector APIs to publish the messages to one or more Kafka topics. Similarly, applications can use Consumer API to subscribe to one or more topics. Kafka has the capability to 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 allows to store the stream of data in distributed clusters.

     Kafka acts as a storage system for incoming data stream. Data is categorised into 'topics'. 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'.

     Kafka is a commonly used for building data pipelines where data is transferred between systems or applications. *[Fou]* Kafka can also be used by applications that transform real time incoming data.

172. Kestrel

173. JMS

174. AMQP
175. Stomp
176. MQTT
177. Marionette Collective
178. Public Cloud: Amazon SNS
179. Lambda
180. Google Pub Sub
181. Azure Queues
182. Event Hubs

5.8 In-memory databases/caches

183. Gora (general object from NoSQL)

    Gora is a in-memory data model *[wwwb]* which also provides persistence to the big data. Gora provides persistence to different types of data stores. Primary goals of Gora are:

    (a) data persistence
    (b) indexing
    (c) data access
    (d) analysis
    (e) 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:

    (a) define the bean used as model for persistence
    (b) use gora compiler to compile the bean
    (c) create a mapping file to map bean definition to datastore
    (d) update gora.properties to specify the datastore to use
    (e) 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.

184. Memcached
185. Redis
186. LMDB (key value)
187. Hazelcast
188. Ehcache
189. Infinispan

190. VoltDB

191. H-Store

     H-Store is an in memory and parallel database management system for on-line transaction processing (OLTP). Specifically , *[HSta]* 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 *[KKN+08]* "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 *[HStb]* 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.

5.9 Object-relational mapping

192. Hibernate

193. OpenJPA

194. EclipseLink

195. DataNucleus

196. ODBC/JDBC

5.10 Extraction Tools

197. UIMA

381. 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. *[Tik]*"

5.11 SQL(NewSQL)

198. Oracle

199. DB2

200. SQL Server

     SQL Server *[sql]* is a relational database management system from Microsoft. As of Jan 2017, SQL Server is available in below editions

     (a) Standard - consists of core database engine

     (b) Web - low cost edition for web hosting

     (c) Business Intelligence - includes standard edition and business intelligence tools like PowerPivot, PowerBI, Master Data Services

(d) Enterprise - consists of core database engine and enterprise services like cluster manager

(e) SQL Server Azure - *[azu]* core database engine integrated with Microsoft Azure cloud platform and available in platform-as-a-service mode.

It is explained that technical architecture of SQL Server in OLTP(online transaction processing), hybrid cloud and business intelligence modes *[RM14]*.

201. SQLite
202. MySQL
203. PostgreSQL
204. 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. *[www17b]* 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.

205. Galera Cluster

Galera cluster *[www17c]* 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).

206. SciDB
207. Rasdaman
208. Apache Derby
209. Pivotal Greenplum
210. Google Cloud SQL
211. Azure SQL
212. Amazon RDS
213. Google F1
214. IBM dashDB
215. N1QL
216. BlinkDB
217. Spark SQL

5.12 NoSQL

218. Lucene

Apache Lucene *[www17a]* 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 recomendation system. The core idea of Lucene is to extract text from any document that contains text (not image) field, making it format idependent.

219. Solr
220. Solandra
221. Voldemort

According to *[Lin]*, 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 ([rabl_sadoghi_jacobsen_2012]), the Voldemart'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 Voldermort 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.

222. Riak
223. ZHT
224. Berkeley DB
225. Kyoto/Tokyo Cabinet
226. Tycoon
227. 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. In his blog *[Tyra]* Matt Yonkovit has explained the results of performance experiments he conducted to compare Tyrant against Memcached and MySQL.

Tyrant was written and maintained by FAL Labs *[Tyrb]*. However, according to FAL Labs, their latest product *[Tyc]* Kyoto Tycoon is more powerful and convenient server than Tokyo Tyrant.

228. MongoDB
229. Espresso
230. CouchDB
231. Couchbase
232. IBM Cloudant
233. Pivotal Gemfire
234. HBase
235. Google Bigtable

236. LevelDB
237. Megastore and Spanner
238. Accumulo
239. Cassandra

    Apache Cassandra *[www16a]* is an open-source distributed database managemment for handling large volume of data accross comodity 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.

240. RYA
241. Sqrrl
242. Neo4J
243. graphdb
244. Yarcdata
245. AllegroGraph
246. Blazegraph
247. Facebook Tao
248. Titan:db
249. Jena
250. Sesame
251. Public Cloud: Azure Table
252. Amazon Dynamo
253. Google DataStore

5.13 File management

254. iRODS
255. NetCDF
256. CDF
257. HDF
258. OPeNDAP
259. FITS

    FITS stand for 'Flexible Image Trasnport System'. It is a standard data format used in astronomy. FITS data format is endorsed by NASA and International Astronomical Union. According to *[FITa]*, 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 *[FITb]* newer version of FITS standard document was released in July 2016. FITS can be used for digitization of contents like books and magzines. *[FITc]* used FITS for long term preservation of their book, manuscripts and other collection. Matlab, a language used for technical computing supports fits *[FITd]*. In his 2011 paper, Keith Wiley *[pap11]* explained how they performed processing of astronomical images on Hadoop. They used FITS format for data storage.

260. RCFile
261. ORC
262. Parquet

5.14 Data Transport

263. BitTorrent
264. HTTP
265. FTP
266. SSH
267. Globus Online (GridFTP)

    GridFTP is a enhancement on the File Tranfer Protocol (FTP) which provides high-performance , secure and reliable data transfer for high-bandwidth wide-area networks. As noted in *[Gri]* 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 .

268. Flume
269. Sqoop
270. Pivotal GPLOAD/GPFDIST

5.15 Cluster Resource Management

271. Mesos
272. Yarn
273. Helix
274. Llama
275. Google Omega
276. Facebook Corona
277. Celery
278. HTCondor
279. SGE
280. OpenPBS
281. Moab
282. Slurm *[Slu]*

283. Torque
284. Globus Tools
285. Pilot Jobs

5.16 File systems

286. HDFS
287. Swift
288. Haystack
289. f4
290. Cinder
291. Ceph
292. FUSE
293. Gluster
294. Lustre
295. GPFS
296. GFFS
297. Public Cloud: Amazon S3
298. Azure Blob
299. Google Cloud Storage

5.17 Interoperability

300. Libvirt
301. Libcloud
302. JClouds

    [BDM15] 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 *[wwwd]* 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:

    (a) 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

    (b) 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.

(c) Provider: a particular cloud vendor is a provider. Jcloud uses provider information to initialize its context.

(d) 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 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.

303. TOSCA

304. 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" *[For]*. 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 require 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. [www-occi-core]

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" [www-occi-temp].

OCCI RESTful HTTP Protocol describes communications between server and client on OCCI platform via HTTP protocol [[www-occi-HTTP]]. 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 [www-occi-json] 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 depricated and will be removed from the next major version [www-occi-text].

305. CDMI

306. Whirr

307. Saga

308. Genesis

5.18 DevOps

309. Docker (Machine, Swarm)

310. Puppet

311. 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. In this book *[Mar13]*, it is mentioned how implementation recipes in Chef to manage server applications and utilities such as database servers like MySQL, or HTTP servers like Apache HTPP and systems like Apache Hadoop.

     Chef is available in open source version and it also has commercial products for the companies which need it *[Che]*

312. Ansible
313. SaltStack
314. Boto
315. Cobbler
316. Xcat
317. Razor
318. CloudMesh
319. Juju

     Juju (formerly Ensemble) *[BlRW14]* 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 *[Canonical]* 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.

320. Foreman
321. OpenStack Heat
322. Sahara

     The Sahara product provides users with the capability to provision data processing frameworks (such as Hadoop, Spark and Storm) on OpenStack *[Ope]* by specifying several parameters such as the version,cluster topology and hardware node details.As specified in *[Sah]* 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.

323. Rocks
324. Cisco Intelligent Automation for Cloud
325. Ubuntu MaaS
326. Facebook Tupperware
327. AWS OpsWorks

     AWS Opsworks is a configuration service provided by Amazon Web Services that uses Chef, a Ruby and Erlang based configuration management tool *[Wik17]*, 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. *[Ama17]*

328. OpenStack Ironic
329. Google Kubernetes
330. Buildstep
331. Gitreceive
332. OpenTOSCA
333. Winery
334. CloudML
335. Blueprints
336. Terraform
337. DevOpSlang
338. Any2Api

5.19 IaaS Management from HPC to hypervisors

339. Xen
340. KVM
341. QEMU
342. Hyper-V
343. VirtualBox
344. OpenVZ
345. LXC
346. Linux-Vserver
347. OpenStack
348. OpenNebula
349. Eucalyptus
350. Nimbus

     Nimbus Infrastructure *[Nimb]* 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 *[Nima]* 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 cloud. The Nimubs Platform tools are cloudinit.d, Phantom and Context Broker. In this paper *[nim13]* it is mentioned how to used Nimbus Phantom to deploy auto-scaling solution across multiple NSF FutureGrid clouds. In this implementation Phantom was responsible for deploying instances across multiple clouds and monitoring those instance. Nimbus platform supports Nimbus, Open Stack, Amazon and several other clouds.

351. CloudStack
352. CoreOS
353. rkt
354. VMware ESXi

355. vSphere and vCloud
356. Amazon
357. Azure
358. Google and other public Clouds
359. Networking: Google Cloud DNS
360. Amazon Route 53

5.20 Cross-Cutting Functions

5.20.1 Monitoring

361. Ambari
362. Ganglia
363. Nagios *[wwwf]*

    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 *[Jos13]*: 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. *[IPMM12]* Besides its open source components, Nagios also has commercial products to serve needing clients.

364. Inca

    Inca is a grid monitoring *[LW09]* software suite. It provides grid monitoring features. These monitoring features provide operators failure trends, debugging support, email notifications, environmental issues etc. *[wwwc]*. 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.

5.20.2 Security & Privacy

365. InCommon

366. Eduroam
367. OpenStack Keystone
368. LDAP
369. Sentry
370. Sqrrl
371. OpenID
372. SAML OAuth

5.20.3 Distributed Coordination

373. Google Chubby
374. Zookeeper
375. Giraffe
376. JGroups

5.20.4 Message and Data Protocols

377. Avro
378. Thrift
379. Protobuf

     Protocol Buffer *[www16b]* 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 apllication 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.

5.21 New Technologies to be integrated

382. TBD

5.22 Excersise

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 refernces 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 refernces. We have created in the referens file a placeholder using your HID to simplify the management of the refernces while avoiding conflicts. For the technologies you are responsible to invesitgate them and write an academic summary of the technology. Make sure to add your refernce to refs.bib. Many technologies may have additional refernces than the Web page. Please add the most important once while limiting it to three if you can. Avoid plagearism and use proper quotations or better rewrite the text.

You must look at technologies-hw to sucessfully complete the homework

A video about this hoemwork 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 realated 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. Fith you can push. Six if the TAs have commend improve. The commit message must have new:usr: at the beginning.

While the Nagios entry is a good example (make sure grammer 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 groundbraking work conducted in England, Allan Turing, introduced the turing machine in the years 1936-37 [2]. Its definition is base 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 Plagearism 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 or other Big Data related Web pages and projects that are not yet listed here and add the name and descriptions as well as references and that you find important.

TechList.2: In this hopweork we provide you with additional technologies that you need to compleate They are marked with (2) in the HID assignment.

TechList.3: Identify technologies that are not listed here and add them. Provide a description and a refrence just as you did before. Make sure duplicated entries will be merged. Before you start doing a technology to avoid adding technologies that have already been done by others.

5.23 Refernces

CHAPTER

SIX

FAQ

6.1 How do I ask a question?

We are using piazza for asking question. Our intend is to first gather the answer on piazza and than move the answer in some form to the web page if appropriate.

However asking a quaestion may require some effort on your part.

We suggest the following

- Before asking the quetion look at the Class Web site and in piazza. Both have search functions that you can use.
- Do not burry your question in an unrelated post. INstead use the **New Post** button
- Provide enough information in the questions. THis may include:
  - yur name
  - your HID
  - the exact URL where the issue aooruce (if related to the Web page or piaza)
  - detailed description of what the error is about
  - Make use of the online sessions so you can demonstrate the issue if it complicated to describe

A student als pointed to the following post:

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

6.2 What are the prerequisites for this class?

We have communicated the prerequisites to the university, but they may have forgotten to add them to the class. The perquisites can be found in the appropriate class overview. Please review them carefully.

See:

- I524 *Prerequisites*

6.3 Why is this class not hosted on EdX?

I523 and I524 were the first classes hosted on EdX. We wanted to continue offering them on EdX. However, the university system administrators mentioned to us that their systems are not ready to support this class in EdX. Unfortunately, this is out of our control and we hope that the university will soon update to an EdX system that can host our classes.

6.4 Why are you not using CANVAS for communicating with students?

We have found that Piazza supports our needs for communication better than Canvas.

6.5 Why are you using github for submitting projects and papers?

This class is about open source technology and we like that you benefit from material others in the class are developing or have developed. All assignments are openly submitted to the class github for everyone to see. As part of the goal of this class is to develop reusable deployments. Such reuse is only possible if the code is publicly available and others can benefit from it.

The technology papers are made accessible so you can read other technology papers and can get an introduction in technologies that you may not yet know about. It also allows others to contribute to your technology papers and improve them.

6.6 I am full time student at IUPUI. Can I take the online version?

Yes you can.

If you are an international student, I suggest you verify this with the office and the registrar. There may be some restrictions for international students. Also some degree programs may have a limit or do not allow to take online classes. It will be up to you to verify the requirements with the appropriate administrators.

6.7 I am a residential student at IU. Can I take the online version only?

We recommend you take the residential class.

If you are an international student or a student of a particular degree program restrictions may be placed in if and how many online courses you can take. It will be up to you to contact the appropriate administrative departments including the international student office to verify what is allowed for you. In general international students have such restrictions. Please find out what they are and which section of the course is appropriate for you.

6.8 The class is full what do I do?

1. Make sure to put yourself on the waiting list.
2. If you are a residential student show up on the first class in the specified lecture room. More likely than not some students will enroll in more classes than they can do and places will free up. We will create a list and discuss with the registrar what to do.

6.9 Do I need to buy a textbook?

No, the resources will be provided for every unit. However, we recommend that you identify useful books for the class that can help you. Examples include

1. "Taming The Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics", Bill Franks Wiley ISBN: 978-1-118-20878-6
2. "Doing Data Science: Straight Talk from the Frontline", Cathy O'Neil, Rachel Schutt, O'Reilly Media, ISBN 978-1449358655

If you find good books, we like to add them here.

6.10 Why is there no textbook?

We cover a wide range of topics and their subject-matter is constantly undergoing changes. A textbook would be out of date by the time of publishing.

6.11 Do I need a computer to participate in this class?

If you are an online student you need access to a computer. If you are a residential student the facilities provided by SOIC will be sufficient. However, as you study involves computers, its probably important to evaluate if a computer will make your work easier.

If it comes to what computer to buy we really do not have a good recommendation as this depends on your budget. A computer running Linux or OSX makes programming probably easier. A windows computer has the advantage of also being able to run Word and ppt (so does OSX). A cheap machine with multiple cores and sufficient memory (16GB+) is a good idea. A SSD will make access to data especially if large data snappy.

For this reason I myself use a Mac, but you probably can get much cheaper machines with similar specs elsewhere.

Other students bought themselves a cheap computer and installed Linux on it so they do not interfere with their work machines or with Windows. Given how inexpensive computers these days are this may be a reasonable idea. However, do not go too cheap have enough memory and use an SSD if you can.

6.12 Representative Bibliography

1. Big data: The next frontier for innovation, competition, and productivity
2. Big Data Spring 2015 Class

6.13 Where is the official IU calendar for the Fall?

Please follow this link

6.14 How to write a research article on computer science?

1. A good lecture about this is presented by Simon Peyton Jones, Microsoft Research https://www.youtube.com/watch?v=g3dkRsTqdDA

Other resources may inspire you also:

1. https://globaljournals.org/guidelines-tips/research-paper-publishing
2. http://www.cs.columbia.edu/~hgs/etc/writing-style.html

3. https://www.quora.com/How-do-I-write-a-research-paper-for-a-computer-science-journal

6.15 Which bibliography manager is required for the class?

We require you use jabref:

1. http://www.jabref.org/

6.16 Can I use endnote or other bibliography managers?

No. Jabref is best for us and we do require that you hand in all bibliographies while cleaning and transferring them to jabref. We will not accept any other bibliography tool such as:

1. http://endnote.com/
2. http://libguides.utoledo.edu/c.php?g=284330&p=1895338
3. https://www.mendeley.com/
4. https://community.mendeley.com/guides/using-citation-editor/05-creating-bibliography
5. https://www.zotero.org

6.17 Plagiarism test and resources related to that

1. https://www.grammarly.com/plagiarism-checker
2. http://turnitin.com/
3. http://www.plagscan.com/plagiarism-check/

6.18 How many hours will this course take to work on every week?

This question can not rely be answered precisely. Typically we have 2-3 hours video per week. However starting from that its difficult to put a real number down as things may also depend on your background.

- The programming load is modest, but requires knowledge in python and linux systems which you may have to learn outside of this class.
- Some students have more experience than others, thus it may be possible to put in 6 hours per week overall, but other may have to put in 12 hours, while yet others may enjoy this class so much that they spend a lot more hours.
- We will certainly not stopping you form spending time in the class. It will be up to you to figure out how much time you will spend.
- Please remember that procrastination will not pay of in this class.
- The project or term paper will take a significant amount of time.

6.19 Is all classes material final?

No. Class material can change. Please remember that in a normal class you will be given several hours of lectures a week. They will be released on a weekly basis. What we do here is to release the material as much as possible upfront and **correct** them when we find it necessary to provide improvements or additions. Additionally, we integrate your feedback into the classes. If you find errors on the class Web page or have additions that you want to add, we would like to hear from you. Pull requests can be issued by you so your contributions get acknowledged and rewarded as part of the grade.

6.20 What are the changes to the web page?

The changes we make are typically fixing errata or clarification of content. We do attempt to indicate when major change is made.

6.21 What lectures should I learn when?

The class is structured in lectures that you can listen to at any time. If you have difficulties with organizing your own calendar, we will develop a sample calendar for you. Please contact us. However we have undergraduates, graduates, residential and online students. We even have students that can only work part of the semester while they use their vacation. Hence, it is impossible for us to provide an exact calendar that satisfies all the different types of students. Hence we appeal to your organizational skills to create a "study" plan for you during the first week of the semester that works for you.

We recommend to do the theory lectures as quickly as possible, but also start learning ansible at the same time as this will be part of your project. You will fail if you assume you can do the project in 2 weeks. You will need to work on it all semester long on weekly basis, starting with learning how to use ansible and cloud resources.

6.22 I524: Why are you doing the papers?

Part of doing research is to communicate your thoughts on topics and to be able to analyze and evaluate technologies that may or may not be useful for you. Our goal within this class is for the first time to gather a significant portion of the technologies that you hear about in class and that you get exposed to as part of the technology list into a "proceedings" developed by all students in class. The papers serve also the dual purpose of you learning how to write a paper and use bibliographies.

6.23 I524: Why are there no homework to test me on skills such as ansible or python?

We used to do smaller homework in previous classes to evaluate you on your skills. However we found that they did not reflect real-world use cases. By focusing on the project instead, you will be forced to develop these skills.

However, we can provide you with additional ungraded homework that you can conduct to test your skills if you like. Please let us know if you like to do that and we can assign such homework to you.

6.24 I524: Why not use chef or another DevOps framework?

We used to use chef and other DevOps frameworks. However we found that for a class grading can not be uniformly conducted while using too many frameworks. We also found that the value of learning on how to collaboratively contribute as part of an opensource class was diminished while a small group were choosing other technologies. These groups complained later on that they had too much work and could not benefit from other students. Hence we make is simple. All DevOps must be provided in ansible. All programming must be provided in python if not an explicit reason exist to use another language or technology such as R or technologies such as neo4j. However all deployment must be done in python and ansible.

6.25 I am lost?

Please contact the instructors for your class.

6.26 I do not like Technology/Topic/Project/etc?

Please contact the instructors for your class.

6.27 I am not able to attend the online hours

Typically we provide many different times for meetings via Zoom. We even schedule within reason special sessions. All of them are however during reasonable hours in United States Easter Standard Time.

6.28 Do I need to attend the online sessions?

No. But you can ask any question you want. We found that in previous classes that some students clearly benefitted from online sessions. If you attend them make sure you have a working and tested microphone if possible.

6.29 What are the leaning outcomes?

If you feel that they are not clearly stated as part of the course please contact us so that we can clarify the material.

6.30 There are so many messages on Piazza I can not keep up.

Residential students typically participate in live lectures in which we discuss with each other important aspects of a topic. As an online class may not have such a lecture, the piazza posts are just a replacement of them. It is required that you read the posts and decide which of them are relevant for you. In a lecture room you will find also that one student asks a question, while the professor answers the question to the entire class.

6.31 I find the hosting Web confusing

Once in a while we find that a student finds the hosting of the class material on the class Web page confusing. This confusion can be overcome by doing the following:

1. You may have to take time to explore the Web page and identify what needs to be done for the class. However each class has a clear overview page.
2. You may have to learn to get used to a class that allows you to work ahead.
3. You may have to learn to appreciate the additional material that assist in learning about python, ansible, LaTex, or the many other topics
4. Please do not blame the instructors for things that are out of their control: You may not be aware that it is not the instructors fault that the university is not able to provide us with an EdX server that works for us. Our choice would be to use EdX.

6.32 I524: I do not know python. What do I do?

This class requires python. Please learn it. We will be using ansible for the project. This you can acquire as part of the class through self study. There is a section under lessosn that has some elementary python included.

CHAPTER

SEVEN

WRITING DOCUMENTS

7.1 LaTeX

7.1.1 Introduction

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

The information provided here is not intended to replace one of the many text books available about LaTeX. For the beginning you might be just fine with the documentation provided here. For serious users I recommend to purchase a book. Examples for books include

- LaTeX Users and Reference Guide, by Leslie Lamport
- LaTeX an Introduction, by Helmut Kopka
- The LaTeX Companion, by Frank Mittelbach

If you do not want to buy a book you can find a lot of useful information in the LaTeX reference manual.

7.1.2 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 incredible stable
- produces high quality output
- is platform independent
- has lots of templates
- has been around for many years so it works well
- removes you form the pain of figure placements
- focusses you on content rather tan 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.

7.1.2.1 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 look. 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.

7.1.2.2 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.

7.1.2.3 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: reduce 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

7.1.3 Editing

7.1.3.1 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 aquaemacs and carbon emacs have build in support for LaTeX. Spell checking is done with flyspell in emacs.

7.1.3.2 Vi/Vim

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

7.1.3.3 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/

7.1.3.3.1 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.

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 ma 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.

7.1.4 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.

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

7.1.5 Installation

7.1.5.1 Local Install

Installing LaTeX is trivial, but requires sufficient space and time as it is a large environment. 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: less fancy GUI to bibtex files

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

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

If these commands are missing, please instal them.

7.1.5.2 Online Services

7.1.5.2.1 Sharelatex

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/PfhSOjuQk8Y

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.

7.1.5.2.2 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, put you are allowed to use the OSA template.

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

7.1.6 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
```

A great example is provided at:

- https://gitlab.com/cloudmesh/project-000/tree/master/report

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
```

Note: for make watch its best to use skim as pdf previewer

7.1.7 Generating 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 rations 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.

7.1.8 Bibliographies

LaTeX integrates very well with bibtex. There are several preformatted styles available. It includes also styles for ACM and IEEE bibliographies. For the ACM style we recommend that you replace abbrv.bst with abbrvurl.bst, add hyperref to your usepackages so you can also display urls in your citations:

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

Than 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*.

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.

7.1.8.1 jabref

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

Video: https://youtu.be/cMtYOHCHZ3k

Video with cc: https://www.youtube.com/watch?v=QVbifcLgMic

7.1.8.2 jabref and MSWord

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

Note: 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 has figures or tables, or you need to add references.

7.1.8.3 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.

7.1.8.3.1 Endnote

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

- http://endnote.com/

7.1.8.3.2 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 Mendeleys integration into word slow.

7.1.8.3.3 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

7.1.9 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.

7.1.10 Links

- The LaTeX Reference Manual provides a good introduction to Latex.

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.

7.1.11 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 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.

7.2 Report Format

Over the years we got tired of student that asked us how many pages a report needs to be and than turn around and play with spacing, fonts and other space manipulations to circumvent these recommendations. Thus we have adopted a much simpler approach. All reports **must** be written in the same format that we define on this page. Thus we require that all the reports and papers be written in LaTeX while using our **trivial** example template(s).

The template for the report is available from:

- https://github.com/cloudmesh/classes/tree/master/docs/source/format/report

An example report in PDF format is available:

- report.pdf

It includes some very simple makefile and allows you to do editing with immediate preview as documented in the LaTeX lesson. Due to LaTeX being a trivial ASCII based format and its superior bibliography management you wil save yourself many hours of work.

In case you are in a team, you can use either github/gitlab while collaboratively developing the LaTeX document, use sharelatex, or overleaf.

Your final submission will include the bibliography file as a separate document. All images must be placed in an images folder and submitted in your repository with the originals. When using sharelatex or overleaf you must replicate the directory layout carefully. YOu must also make sure that all files and directories in sharelatex you use be copied back to github.

Warning: There will be **NO EXCEPTION** to this format. Hence if you do not know latex we recommend you get familiar with it. Documents not written in LaTeX that do not follow the specified format and are not accompanied by references managed with jabref and are not spell checked will be returned without review.

Warning: We found that students using MsWord or Google docs produce generally inferior reports with the danger of having a lower grade. Hence, in order to help you achieve the best grade possible, we no longer accept reports using these tools and require that all documents be written in LaTeX.

7.2.1 Report Checklist

This incomplete list may serve as a way to check if you follow the rules

1. Have you written the report in LaTeX in the specified format?
2. Have you included an Acknowledgement section?
3. Have you included the report in gitlab?
4. Have you specified the HID, names, and e-mails of all team members in your report. E.g. the Real Names that are registered in Canvas?
5. Have you included the project number in the report?
6. Have you included all images in native and PDF format in gitlab in the images folder?
7. Have you added the bibliography file that you managed with jabref
8. Have you added an appendix describing who did what in the project or report?
9. Have you spellchecked the paper?
10. Have you made sure you do not plagiarize?
11. Have you not used phrases such as shown in the Figure below, but instead used as shown in Figure 3 when referring to the 3rd figure?
12. Have you capitalizezed “Figure 3”, “Table 1”, ... ?
13. Any figure that is not referred to explicitly in the text must be removed?
14. Are the figure captions bellow the figures and not on top. (Do not include the titles of the figures but instead use the caption or that information?
15. When using tables put the table caption on top?
16. Make the figures large enough so we can see it, regardless of page restrictions. If needed make the figure over two columns?
17. Do not worry about the figure placement if they are at a different location than you think. Figures are allowed to float. If you want you can place all figures at the end of the report?

18. Do not use the word “I”?
19. Do not artificially inflate your report if you are bellow the page limit and have nothing to say anymore?
20. If your paper limit is 12 pages but you want to hand in 120 pages, please check first with an instructor ;-)
21. Check in your current work of the report on a weekly basis to show consistent progress?
22. Is in your report directory a README.rst file in it as shown in the example project that we introduced you to?

7.2.2 Exercise

Report.1: Install latex and jabref on your system

Report.2: Check out the project-000 example directory. Create a PDF and view it. Modify and recompile.

Report.4: Learn about the different bibliographic entry formats in bibtex

Report.5: What is an article in a magazine? Is it realy an Article or a Misc?

Report.6: What is an InProceedings and how does it differ from Conference?

Report.7: What is a Misc?

Report.8: Why are spaces, underscores in directory names problematic and why should you avoid using them for your projects

Report.9: Write an objective report about the advantages and disadvantages of programs to write reports.

Report.10: Why is it advantageous that directories are lowercas have no underscore or space in the name?

CHAPTER

EIGHT

LINUX

8.1 Linux Shell

There are many good tutorials out there that explain why one needs a linux shell and not just a GUI. Randomly we picked the firts 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 als find in a link to a RefCard.

- http://www.cheat-sheets.org/#Linux

8.1.1 File commands

Find included a number of commands related to file manipulation.

Command	Description
ls	Directory listing
ls -lisa	list details
cd *dirname*	Change directory to *dirname*
mkdir *dirname*	create the directory
pwd	print working directory
rm *file*	remove the file
cp *a b*	copy file *a* to *b*
mv *a b*	move/rename file *a* to *b*
cat *a*	print content of file*a*
less *a*	print paged content of file *a*
head -5 *a*	Display first 5 lines of file *a*
tail -5 *a*	Display last 5 lines of file *a*

8.1.2 Search commands

Find included a number of commands related to seraching.

Command	Description
fgrep	TBD
grep -R “xyz” .	TBD
find . -name “*.py” | TBD |	

8.1.3 Help

Find included a number of commands related to manual pages.

Command	Description
man *command*	manual page for the *command*

8.1.4 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 command you're typing
Ctrl + e	Go to the end of the command you're typing
Ctrl + k	Cut everything after the cursor to a special clipboard
Ctrl + y	Paste from the special clipboard
Ctrl + d	Log out of current session, similar to exit

8.1.5 Exercise

Linux.1: Familiarize yourself with the commands

Linux.2: Find more commands that you find useful and add them to this page.

Linux.3: Use the *sort* command to sort all lines of a file while removing duplicates.

8.2 Refcards

We present you with a list of useful short refrence cards. This cards can be extremly useful to remind yourself about some important commands and features. Having them could simplify your interaction with the systems, We not only collected here some refcards about Linux, but also about other useful tools and services.

If you like to add new topics, let us know via your contribution (see the contribution section).

Emacs	https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf
Vi	http://www.ks.uiuc.edu/Training/Tutorials/Reference/virefcard.pdf
Linux	http://www.cs.jhu.edu/~joanne/unixRC.pdf
Makefile	http://www.tofgarion.net/lectures/IN323/refcards/refcardMakeIN323.pdf
R	https://cran.r-project.org/doc/contrib/Short-refcard.pdf
Python	https://dzone.com/refcardz/core-python
Python Data	https://dzone.com/refcardz/data-mining-discovering-and
SQL	http://www.digilife.be/quickreferences/QRC/MySQL-4.02a.pdf
Vim	http://michaelgoerz.net/refcards/vimqrc.pdf
LaTeX	https://wch.github.io/latexsheet/latexsheet.pdf
Git	https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf
Openstack	http://docs.openstack.org/user-guide/cli_cheat_sheet.html
Openstack	http://cmias.free.fr/IMG/pdf/rc208_010d-openstack_2.pdf
RST	https://github.com/ralsina/rst-cheatsheet/blob/master/rst-cheatsheet.pdf

Others:

- Numpi/Pandas: http://www.cheat-sheets.org/saved-copy/NumPy_SciPy_Pandas_Quandl_Cheat_Sheet.pdf
- Cheat Sheets: http://www.cheat-sheets.org/
- Python Tutorial: http://fivedots.coe.psu.ac.th/Software.coe/learnPython/Cheat%20Sheets/python2.pdf
- Python: http://www.cheat-sheets.org/saved-copy/PQRC-2.4-A4-latest.pdf
- Python: https://www.cheatography.com/davechild/cheat-sheets/python/pdf/
- Python API Index: http://overapi.com/python
- Python 3: https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf

8.3 Using SSH Keys

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 etting started quickly. It can however not replace the more comprehensive documentation.

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.

8.3.1 Using SSH from Windows

Hint: For Linux users, please skip to the section *Generate a SSH key*

Hint: For Mac users, please skip to the section *Using SSH on Mac OS X*

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 completness 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.

The most popular software making ssh clients available to Windows users include

- cygwin
- putty
- or installing a virtualiztion software and running Linux virtual machine on your Windows OS.
- using chocolatey
- using bash ubuntu under WIndows 10 (we need a contribution on this)

We will be discussing here how to use it in Powershell with the help of chopolatey. Other options may be better suited for you and we leave it up to you to make this decission. In general we recommend that you use an ubuntu OS either on bare hardware or a virtual machine. Naturally your computer must support this. It will be up to you to find such a computer.

However if you want a unix like environments with ssh you can use Chocolatey.

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. Before using and installing a package be aware of the consequences when installing software on your computer. Please be aware that there could be malicious code offered in the chocolatey repository although the distributors try to remove them.

The installation is sufficently 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 win32-openssh
```

Warning: If you have a different version such as a 64 bit version please find teh appropriate commands

Other packages of interest include

- LaTeX:: *choco install miktex*
- jabref: *choco install jabref*
- pycharm: *choco install pycharm-community*
- python 2.7.11: *choco install python2*
- pip: *choco install pip*
- virtual box: *choco install virtualbox*
- emacs: *choco install emacs*

- lyx: *choco install lyx*
- vagrant: *choco install vagrant*

Before installing any of them evaluate if you need them.

8.3.2 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.

8.3.3 Generate a SSH key

|info-image| **Hint**

In case you do not want to type in your password everytime, please learn about ssh-agent and ssh-add.

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. 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. Also, please use a strong passphrase to protect it appropriately.

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.

Note: 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** chose 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
The key's random art image is::

+--[ RSA 2048]----+
|.+...Eo= .       |
| ..=.o + o +o    |
|O.  o o +.o      |
| = .   . .       |
+-----------------+
```

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 AAAAB3NzaC1yc2EAAAADAQABAAABAQCXJH2iG2FMHqC6T/U7uB8kt
6KlRh4kUOjgw9sc4Uu+Uwe/EwD0wk6CBQMB+HKb9upvCRW/851UyRUagtlhgy
thkoamyi0VvhTVZhj61pTdhyl1t8hlkoL19JVnVBPP5kIN3wVyNAJjYBrAUNW
4dXKXtmfkXp98T3OW4mxAtTH434MaT+QcPTcxims/hwsUeDAVKZY7UgZhEbiE
xxkejtnRBHTipi0W03W05TOUGRW7EuKf/4ftNVPilCO4DpfY44NFG1xPwHeim
Uk+t9h48pBQj16FrUCp0rS02Pj+4/9dNeS1kmNJu5ZYS8HVRhvuoTXuAY/UVc
ynEPUegkp+qYnR user@myemail.edu
```

8.3.4 Add or Replace Passphrase for an Already Generated Key

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.
```

8.3.5 Upload the key to gitlab

Follow the instructions provided here:

- http://docs.gitlab.com/ce/ssh/README.html

8.3.6 Exercise

SSH.1: create an SSH key pair

SSH.2: upload the key to github and/or gitlab. Create a fork in git and use your ssh key to clone and commit to it

SSH.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.

8.4 Ubuntu Development Configurations

8.4.1 Development Configuration

The documentation on how to configure the virtual machine and install many useful programs is posted at:

- https://github.com/cloudmesh/ansible-cloudmesh-ubuntu-xenial

You simply have to execute the following commands in the terminal of the virtual machine. In order to eliminate confusion with other terminals, we use the prefix *vm> $* to indicate any command that is to be started on the virtual machine. Otherwise it is clear from the context:

```
vm>$ wget https://raw.githubusercontent.com/cloudmesh/ansible-cloudmesh-ubuntu-xenial/
↪master/bootstrap.sh
vm>$ bash bootstrap.sh
```

A video showcasing this install is available:

- Video: https://youtu.be/YqXIj_Wzfsc

A video showcasing the upload to gitlab from within the vm using commandline tools

- Video: https://youtu.be/EnpneUY82I8

8.5 Virtual Box Installation and Instructions

For development purposes we recommend tha you use for this class an ubuntu virtual machine that you set up with the help of virtualbox.

Only after you have successfully used ubuntu in a virtual machine you will be allowed to use virtual machine son clouds.

A "cloud drivers license test" will be conducted to 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 before you have not passed the test. Please be patient. Only students enrolled in the class can get access to the cloud.

8.5.1 Creation

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 the recommended requirements that also apply to a virtual machine:

- 2 GHz dual core processor or better
- 2 GB system memory
- 25 GB of free hard drive space

A video to showcase such an install is available at:

- Video: https://youtu.be/NWibDntN2M4

> **Warning:** If you specify your machien too small you will not be able to install the development environment. Gregor used on his machine 8gb of RAM and 20GB diskspace.
>
> Please let us know the smalest configuration that works for you and share this in Piaza. Only update if yours is smaller and works than a previous post. If not do not post.

8.5.2 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: https://youtu.be/wdCoiNdn2jA

Note: Please reboot the machine after installation and configuration.

On OSX you can once you have enabled bidirectional copying in the Device tab with

OSX -> Vbox: *command c -> shift CONTRL v*

Vbox to OSX: *shift CONTRL v -> shift CONTRL v*

On Windows the key combination is naturally different. Please consult your windows manual.

8.5.3 Exercise

Virtualbox.1: Install ubuntu desktop on your computer with guest additions.

Virtualbox.2: Make sure you know how to paste and copy between your host and guest operating system

Virtualbox.3: Install the programs defined by the development configuration

CHAPTER

NINE

REST WITH EVE

9.1 Overview of REST

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:

- post data: while creating and/or updating it,
- read data: while making queries, and
- delete data.

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 illustrationg how to implement them with HTTP methods.

Table 9.1: IMplementing REST with HTTP methods

URL	GET	PUT	POST	DELETE
http://.../resources/	List the URIs and perhaps other details of the collection's members.	Replace the entire collection with another collection.	Create a new entry in the collection. The new entry's URI is assigned automatically and is usually returned by the operation.	Delete the entire collection.
http://.../resources/item17	Retrieve a representation of the addressed member of the collection, expressed in an appropriate Internet media type.	Replace the addressed member of the collection, or if it does not exist, create it.	Not generally used. Treat the addressed member as a collection in its own right and create a new entry within it.	Delete the addressed member of the collection.

Source: https://en.wikipedia.org/wiki/Representational_state_transfer

9.2 REST and eve

Now that we have outlined the basic functionality that we need, we lke to introduce you to Eve that makes this process rather trivial. IN fact 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/eve

9.2.1 Installation

First we havt to install mongodb. The instalation will depend on your operating system. Note that for this example we do not need to integrate mongodb into the system upon reboot. IN fact for us it is better if we can start and stop the services by hand.

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
```

On OSX you can use homebrew and install it with

```
brew update
brew install mongodb
# brew install mongodb --with-openssl
```

9.2.2 Starting the service

We have provided a convenient Makefile that currently only works for OSX. But you can replicate the steps while looking at the targets we defined in the makefile in a shell program.

```
TODO Provide a shell progra, that runs on all three operating
systems. To be completed by students of the class
```

When using the makefile you can start the services with:

```
make deploy
```

To test the services you can say:

```
make test
```

The program relies on evegenie that we will be added to the repository by executing

```
SOME MAKEFILE TARGET. TO BE COMPLETED BY STUDENT. ITS ALREADY IN MAKEFILE
```

TODO by student add logging

CHAPTER TEN

INTRODUCTION TO PYTHON

10.1 Acknowledgments

Portions of this lesson have been adapted from the official Python Tutorial copyright Python Software Foundation.

10.2 Description

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 is easily 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 The Zen of Python summarizes some of its philosophy including:

- 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.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

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.

10.3 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 2.7.12 but not Python 3.

We assume that you have a computer with python installed. However, we recommend that you use pythons virtualenv to isolate your development python from the system installed python.

Note: If you are not familiar with virtualenv, please read up on it.

10.4 Alternative Installations

The best installation of python is provided by python.og. 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 form 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 then 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 canpoy, 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 instals 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 an 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 anaconde, develop script to generate a conda packge form github
- deploy run and document cloudmesh on canopy, develop script to generate a conda packge form github
- deploy run and document cloudmesh on ironpython
- other documentation that would be useful

10.5 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:

- http://ivory.idyll.org/articles/advanced-swc/
- http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html
- http://www.youtube.com/watch?v=0vJJlVBVTFg
- http://www.korokithakis.net/tutorials/python/
- http://www.afterhoursprogramming.com/tutorial/Python/Introduction/
- http://www.greenteapress.com/thinkpython/thinkCSpy.pdf

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.

10.6 Prerequisite

In order to conduct this lesson you should

- A computer with python 2.7.x
- Familiarity with commandline usage
- A text editor such as PyCharm, emacs, vi or others. You should identity which works best for you and set it up.
- We do not recommend anaconda, or canopy as we ran into issues once you do some more advanced python. Instead we recommend you use pip and virtualenv. If you are unfamiliar with these tools, please consult the manual and tutorials available for it on the internet.

10.7 Dependencies

- Python
- Pip
- Virtualenv
- NumPy
- SciPy
- Matplotlib
- Pandas

10.8 Learning Goals

At the end of this lesson 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 `virtualenv`

10.9 Using Python on FutureSystems

Warning: 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.

Tip: The details of what the `module load` command does are described in the future lesson modules.

10.10 Interactive Python

Python can be used interactively. Start by entering the interactive loop by executing the command:

```
$ python
```

You should see something like the following:

```
Python 2.7 (r27:82500, Aug 10 2010, 11:35:15)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The >>> is the prompt for the interpreter. This is similar to the shell interpreter you have been using.

Tip: Often we 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 prompt 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**: **R**ead-**E**valuate-**P**rint-**L**oop.

10.11 Syntax

10.11.1 Statements and Strings

Let us explore the syntax of Python. Type into the interactive loop and press Enter:

```
print "Hello world from Python!"
```

The output will look like this:

```
>>> print "Hello world from Python!"
Hello world from Python!
```

What happened: the `print` **statement** was given a **string** to process. A **statement** in Python, like `print` tells the interpreter to do some primitive operation. In this case, `print` mean: write the following message to the standard output.

Tip: Standard output is discussed in the /class/lesson/linux/shell lesson.

The “thing” we are print``ing in the case the the **string** ``Hello world from Python!. 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.

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.

10.11.2 Variables

You can store data into a **variable** to access it later. For instance, instead of:

```
>>> print "Hello world from Python!"
```

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:

```
>>> hello = "Hello world from Python!"
>>> print hello
Hello world from Python!
```

10.11.3 Booleans

A **boolean** is a value that indicates the “truthness” 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`

```
>>> print True and True
True
>>> print True and False
False
>>> print False and False
False
>>> print True or True
True
>>> print True or False
True
>>> print False or False
False
```

10.11.4 Numbers and Math

The interactive interpreter can also be used as a calculator. For instance, say we wanted to compute a multiple of 21:

```
>>> print 21 * 2
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
- `**` — exponent

Exponentiation is read as `x**y` is `x` to the `y`th power:

$$x^y$$

You can combine **float**s and **int**s:

```
>>> print 3.14 * 42 / 11 + 4 - 2
13.9890909091
>>> print 2**3
8
```

Note that **operator precedence** is important. Using parenthesis to indicate affect the order of operations gives a difference results, as expected:

```
>>> print 3.14 * (42 / 11) + 4 - 2
11.42
>>> print 1 + 2 * 3 - 4 / 5.0
6.2
>>> print (1 + 2) * (3 - 4) / 5.0
-0.6
```

10.11.5 Types and Using the REPL

We have so far seen a few examples of types: **string**s, **bool**s, **int**s, and **float**s. A **type** indicates that values of that type support a certain set of operations. For instance, how would you exponentiate a string? If you ask the interpreter, this results in an error:

```
>>> "hello"**3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'int'
```

There are many different types beyond what we have seen so far, such as **dictionaries**s, **list**s, **set**s. One handy way of using the interactive python is to get the type of a value using `type()`:

```
  >>> type(42)
  <type 'int'>
  >>> type(hello)

<type 'str'>
  >>> type(3.14)
  <type 'float'>
```

You can also ask for help about something using `help()`:

```
>>> help(int)
>>> help(list)
>>> help(str)
```

Tip: Using `help()` opens up 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.

10.12 Dict

One of the very important datastructures in python is a dictionary also refered to as *dict*. It represents a key value store:

```
person = {'Name': 'Albert', 'Age': 100, 'Class': 'Scientist'}

print ("person['Name']: ", person['Name'])
print ("person['Age']: ", person['Age'])
```

You can delete elements with the following commands:

```
del person['Name']; # remove entry with key 'Name'
person.clear();     # remove all entries in dict
person dict ;        # delete entire dictionary
```

You can iterate ofer a dict:

```
for item in person:
  print (item, person[item])
```

10.13 Lists

see: https://www.tutorialspoint.com/python/python_lists.htm

10.13.1 Control Statements

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:

```
x = int(input("Tell X"))
if x == 4:
    print('You guessed correctly!')
print('End of program.')
```

When you execute this program it will always print ‘End of program’, but the text ‘You guessed correctly!’ will only be printed if the variable x equals to four (see table above). Python can also execute a block of code if x does not equal to 4. The else keyword is used for that.

```
x = int(input("What is the value of  X"))

if x == 4:
    print('You guessed correctly!')
else:
    print('Wrong guess')

print('End of program.')
```

10.13.2 Iterations

To repeat code, the for keyword can be used. To execute a line of code 10 times we can do:

```
for i in range(1,11):
    print(i)
```

The last number (11) is not included. This will output the numbers 1 to 10. Python itself starts counting from 0, so this code will also work:

```
for i in range(0,10):
    print(i)
```

but will output 0 to 9.

The code is repeated while the condition is True. In this case the condition is: i < 10. Every iteration (round), the variable i is updated.Nested loops Loops can be combined:

```
for i in range(0,10):
    for j in range(0,10):
        print(i,' ',j)
```

In this case we have a multidimensional loops. It will iterate over the entire coordinate range (0,0) to (9,9)

10.13.3 Functions

To repeat lines of code, you can use a function. A function has a unique distinct name in the program. Once you call a function it will execute one or more lines of codes, which we will call a code block.

```
import math

def computePower(a):
    value = math.pow(a,2)
    print(value)
```

```

computePower(3)
```

We call the function with parameter a = 3 . A function can be called several times with varying parameters. There is no limit to the number of function calls.

The def keyword tells Python we define a function. Always use four spaces to indent the code block, using another number of spaces will throw a syntax error.

It is also possible to store the output of a function in a variable. To do so, we use the keyword return.

```
import math

def computePower(a):
    value = math.pow(a,2)
    return value

result = computePower(3)
print(result)
```

10.13.4 Classes

A class is a way to take a grouping of functions and data and place them inside a container, so you can access them with the . (dot) operator.

```
    class Fruit(object):

    def __init__(self):
        self.tangerine = "are organge-colored citrus fruit, which is closely related␣
↪to a mandarin organge"

    def apple(self):
        print "Apples are rich in antioxidants, flavanoids, and dietary fiber!"

thing = Fruit()
thing.apple()
print thing.tangerine
```

10.14 Data base access

see: https://www.tutorialspoint.com/python/python_database_access.htm

10.15 Writing and Saving Programs

Make sure you are no longer in the interactive interpreter. If you are you can type `quit()` and press Enter to exit.

You can save your programs to files which the interpreter can then execute. This has the benefit of allowing you to track changes made to your programs and sharing them with other people.

Start by opening a new file `hello.py`:

```
$ nano hello.py
```

Now enter write a simple program and save:

```
print "Hello world!"
```

As a check, make sure the file contains the expected contents:

```
$ cat hello.py
print "Hello world!"
```

To execute your program pass the file as a parameter to the `python` command:

```
$ python hello.py
Hello world!
```

Congratulations, you have written a Python **module**. Files in which Python directives are stored are called **module**s

You can make this programs more interesting as well. Let's write a program that asks the user to enter a number, *n*, and prints out the *n*-th number in the Fibonacci sequence:

```
$ emacs print_fibs.py
```

```
import sys

def fib(n):
    """
    Return the nth fibonacci number

    The nth fibonacci number is defined as follows:
    Fn = Fn-1 + Fn-2
    F2 = 1
    F1 = 1
    F0 = 0
    """

    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)


if __name__ == '__main__':
    n = int(sys.argv[1])
    print fib(n)
```

We can now run this like so:

```
$ python print_fibs.py 5
5
```

Let break this down a bit. The first part:

```
python print_fibs.py 5
```

can be translated to say:

> The Python interpreter `python` should run the `print_fibs.py` program and pass it the parameter `5`.

The interpreter then looks at the `print_fibs.py` file and begins to execute it. The first line it encounters is:

```
import sys
```

This line consists of the `import` keyword. Here `import` attempts to load the `sys` module, which has several useful items.

Next the interpreter sees the `def` keyword. The begins the definition of a function, called `fib` here. Our `fib` function takes a single argument, named `n` within the function definition.

Next we begin a multi-line string between the triple double-quotes. Python can take this string and create documentation from it.

The `fib` function returns the *n*-th number in the Fibonacci sequence. This sequence is mathematically defined as (where *n* is subscripted):

$$
\begin{aligned}
F_0 &= 0 \\
F_1 &= 1 \\
F_n &= F_{n-1} + F_{n-2}
\end{aligned}
$$

This translates to Python as:

```
 if n == 0:
   return 0
 elif n == 1:
return 1
 else:
   return fib(n-1) + fib(n-2)
```

Next we have the block:

```
if __name__ == '__main__':
```

If the interpreter is running this module then there will be a variable `__name__` whose value is `__main__`. This **if statement** checks for this condition and executes this block if the check passed.

Tip: Try removing the `if __name__ == '__main__'` block and run the program. How does it behave differently? What about if you replace with something like:

```
print fib(5)
print fib(10)
```

The next line:

```
n = int(sys.argv[1])
```

does three different things. First it gets the value in the `sys.argv` array at index 1. This was the parameter *5* we originally passed to our program:

```
$ python print_fibs.py 5
```

Substituting the parameter in, the line can be rewritten as:

```
n = int("5")
```

We see that the `5` is represented as a string. However, we need to use integers for the `fib` function. We can use `int` to convert `"5"` to `5`

We now have:

```
n = 5
```

which assigns the value 5 to the variable n. We can now call fib(n) and print the result.

10.16 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 virtualenv command, and installing and uninstalling PyPi packages using pip.

10.16.1 Virtual Environments

Often when you use shared computing resources, such as india.futuresystems.org you will not have permission to install applications in the default global location.

Let's see where grep is located:

```
$ which grep
/bin/grep
```

It seems that there are many programs installed in /bin such as mkdir and pwd:

```
$ ls /bin
alsacard     dbus-cleanup-sockets  env             hostname         mailx          pwd
alsaunmute   dbus-daemon           ex              igawk            mkdir          raw
...
```

If we wished to add a new program it seems like putting it in /bin is the place to start. Let's create an empty file /bin/hello-$PORTALNAME:

```
$ touch /bin/hello-$(whoami)
touch: cannot touch `/bin/hello-albert': Permission denied
```

Tip: Recall that $PORTALNAME is your username on FutureSystems, which can also be obtained using the whoami shell command. t seems that this is not possible. Since india is a shared resources not all users should be allowed to make changes that could affect everyone else. Only a small number of users, the administrators, have the ability to globally modify the system.

We can still create our program in our home directory:

```
$ touch ~/hello-$(whoami)
```

but this becomes cumbersome very quickly if we have a large number of programs to install. Additionally, it is not a good idea to modify the global environment of one's computing system as this can lead to instability and bizarre errors.

A virtual environment is a way of encapsulating and automating the creation and use of a computing environment that is consistent and self-contained.

The tool we use with Python to accomplish this is called `virtualenv`.

Let's try it out. Start by cleaning up our test earlier and going into the home directory:

```
$ rm ~/hello-$(whoami)
$ cd ~
```

Now lets create a virtual env:

```
$ virtualenv ENV
PYTHONHOME is set.  You *must* activate the virtualenv before using it
New python executable in ENV/bin/python
Installing setuptools............done.
Installing pip...............done.
```

When using `virtualenv` you pass the directory where you which to create the virtual environment, in this case `ENV` in the current (home) directory. We are then told that we must activate the virtual environment before using it and that the python program, setuptools, and pip are installed.

Let's see what we have:

```
$ ls ENV/bin
activate  activate.csh  activate.fish  activate_this.py  easy_install
easy_install-2.7  pip  pip-2.7  python  python2  python2.7
```

It seems that there are several programs installed. Let's see where our current `python` is and what happens after activating this environment:

```
$ which python
/N/soft/python/2.7/bin/python
$ source ENV/bin/activate
(ENV) $ which python
~/ENV/bin/python
```

Important: As virtualenv stated, you **must** activate the virtual environment before it can be used.

Tip: Notice how the shell prompt changed upon activation.

10.16.2 Fixing Bad Code

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.

I'm going to demonstrate creating a user python environment, installing a couple packages from pypi, and use them to examine some code.

First, get the bad code like so:

```
$ wget --no-check-certificate http://git.io/pXqb -O bad_code_example.py
```

Let’s examine the code:

```
$ nano 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.

10.16.3 Using pip to install packages

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.

Note: You can skip ‘–trusted-host pypi.python.org’ option if you have a patch on urllib3 on Python 2.7.9.

10.16.4 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’s 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 of passes
$ autopep8 -i file.py # update in place
```

10.17 Further Learning

There is much more to python than what we have covered here:

- conditional expression (`if`, `if...then`,‘‘if..elif..then‘‘)
- function definition(`def`)
- class definition (`class`)
- function positional arguments and keyword arguments
- lambda expression

- iterators
- generators
- loops
- docopts
- humanize

Note: you can receive extra credit if you contribute such a section of your choice addressing the above topics

10.18 Writing python 3 compatible code

see: http://python-future.org/compatible_idioms.html

10.19 Exercises

10.19.1 Lab - Python - FizzBuzz

Write a python program called fizzbuzz.py that accepts an integer n from the command line. Pass this integer to a function called fizzbuzz.

The fizzbuzz function should then iterate from 1 to n. If the ith number is a multiple of three, print "fizz", if a multiple of 5 print "buzz", if a multiple of both print "fizzbuzz", else print the value.

10.19.2 Lab - Python - Setup for FutureSystems

1. Create a 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 FizzBuzz program.
5. Demonstrate the program works and submit the code and output.

10.20 Ecosystem

10.20.1 virtualenv

Often you have your own computer and you do not like to change its environment to keep it in prestine condition. Python comes with mnay libraries that could for example conflict with libraries that you have installed. To avoid this it is bets to work in an isolated python environment while using virtualenv,. Documentation about it can be found at:

```
* http://virtualenv.readthedocs.org/
```

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 n your bashrc or bash_profile command so you do not forget to activate it.

10.20.2 Autoenv: Directory-based Environments

> **Warning:** We do not recommend that you use autoenv. INstead we recommend that you use pyenv. For this class neither is important.

If a directory contains a `.env` file, it will automatically be executed when you `cd` into it. It's easy to use and install.

This is great for...

- auto-activating virtualenvs
- project-specific environment variables

Here is how to use it. Add the ENV you created with virtualenv into `.env` file within your project directory:

```
$ echo "source ~/ENV/bin/activate" > yourproject/.env
$ echo "echo 'whoa'" > yourproject/.env
$ cd project
whoa
```

Here is how to install. Mac OS X Using Homebrew:

```
$ brew install autoenv
$ echo "source $(brew --prefix autoenv)/activate.sh" >> ~/.bash_profile
```

Using pip:

```
$ pip install autoenv
$ echo "source `which activate.sh`" >> ~/.bashrc
```

Using git:

```
$ git clone git://github.com/kennethreitz/autoenv.git ~/.autoenv
$ echo 'source ~/.autoenv/activate.sh' >> ~/.bashrc
```

Before sourcing activate.sh, you can set the following variables:

- `AUTOENV_AUTH_FILE`: Authorized env files, defaults to `~/.autoenv_authorized`
- `AUTOENV_ENV_FILENAME`: Name of the `.env` file, defaults to `.env`

- AUTOENV_LOWER_FIRST: Set this variable to flip the order of .env files executed

Autoenv overrides cd. If you already do this, invoke autoenv_init within your custom cd after sourcing activate.sh.

Autoenv can be disabled via unset cd if you experience I/O issues with certain file systems, particularly those that are FUSE-based (such as smbnetfs).

10.20.3 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 think about pipy 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 pagage_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.

10.20.4 Links

Useful ecosystem Links:

- 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/

CHAPTER

ELEVEN

PYTHON FOR BIG DATA

11.1 Managing Data

11.1.1 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 pacakes and will prbably sufice for your projects in case you use Python.

11.1.2 Pandas

- http://pandas.pydata.org/

According to the Pandas Web page, “Pandas is a library 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 conduction 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

11.2 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

11.3 Graphics Libraries

11.3.1 MatplotLib

- http://matplotlib.org/

According the 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."

11.3.2 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.

11.3.3 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

11.3.4 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

11.3.5 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.

- URL: http://pygal.org/

11.4 Network and Graphs

- igraph: http://www.pythonforsocialscientists.org/t_igraph.html
- networkx: https://networkx.github.io/

11.5 REST

- django REST FRamework 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

11.6 Examples

- *Fingerprint Analysis*

CHAPTER

TWELVE

PYTHON FINGERPRINT EXAMPLE

Python is an easy-to-use language for running data analysis. To demonstrate this, we will implement one of the NIST Big Data Working Group case studies: matching fingerprints between sets of probe and gallery images.

In order for this to run, you'll need to have installed the NIST Biometric Image Software (NBIS) and Sqlite3. You'll also need the Python libraries `numpy`, `scipy`, `matplotlib`.

The general application works like so:

1. Download the dataset and unpack it
2. Define the sets of probe and gallery images
3. Preprocess the images with the `mindtct` command from NBIS
4. Use the NBIS command `bozorth3` to match the gallery images to each probe image, obtaining an matching score
5. Write the results to an sqlite database

To begin with, we'll define our imports.

First off, we use the print function to be compatible with Python 3

```
from __future__ import print_function
```

Next, we'll be downloading the datasets from NIST so we need these libraries to make this easier:

```
import urllib
import zipfile
import hashlib
```

We'll be interacting with the operating systems and manipulating files and their pathnames.

```
import os.path
import os
import sys
import shutil
import tempfile
```

Some general usefull utilities

```
import itertools
import functools
import types
```

Using the `attrs` library provides some nice shortcuts to define featurefull objects

```
import attr
```

We'll be randomly dividing the entire dataset, based on user input, into the probe and gallery sets

```
import random
```

We'll need these to call out to the NBIS software. We'll also be using multiple processes to take advantage of all the cores on our machine.

```
import subprocess
import multiprocessing
```

As for plotting, we'll use `matplotlib`, though there are many other alternatives you may choose from.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

Finally, we'll write the results to a database

```
import sqlite3
```

12.1 Utility functions

Next we'll define some utility functions.

```
def take(n, iterable):
    "Returns a generator of the first **n** elements of an iterable"
    return itertools.islice(iterable, n )


def zipWith(function, *iterables):
    "Zip a set of **iterables** together and apply **function** to each tuple"
    for group in itertools.izip(*iterables):
        yield function(*group)


def uncurry(function):
    "Transforms an N-arry **function** so that it accepts a single parameter of an N-
↪tuple"
    @functools.wraps(function)
    def wrapper(args):
        return function(*args)
    return wrapper


def fetch_url(url, sha256, prefix='.', checksum_blocksize=2**20, dryRun=False):
    """Download a url.

    :param url: the url to the file on the web
    :param sha256: the SHA-256 checksum. Used to determine if the file was previously␣
↪downloaded.
    :param prefix: directory to save the file
    :param checksum_blocksize: blocksize to used when computing the checksum
    :param dryRun: boolean indicating that calling this function should do nothing
    :returns: the local path to the downloaded file
    :rtype:

    """

    if not os.path.exists(prefix):
        os.makedirs(prefix)

    local = os.path.join(prefix, os.path.basename(url))

    if dryRun: return local

    if os.path.exists(local):
        print ('Verifying checksum')
        chk = hashlib.sha256()
        with open(local, 'rb') as fd:
            while True:
                bits = fd.read(checksum_blocksize)
                if not bits: break
                chk.update(bits)
        if sha256 -- chk.hexdigest():
            return local

    print ('Downloading', url)

    def report(sofar, blocksize, totalsize):
        msg = '{}%\r'.format(100 * sofar * blocksize / totalsize, 100)
        sys.stderr.write(msg)

    urllib.urlretrieve(url, local, report)

    return local
```

12.2 Dataset

We'll now define some global parameters.

First, the fingerprint dataset.

```
DATASET_URL = 'https://s3.amazonaws.com/nist-srd/SD4/
↪NISTSpecialDatabase4GrayScaleImagesofFIGS.zip'
DATASET_SHA256 = '4db6a8f3f9dc14c504180cbf67cdf35167a109280f121c901be37a80ac13c449'
```

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.

```
def prepare_dataset(url=None, sha256=None, prefix='.', skip=False):
    url = url or DATASET_URL
    sha256 = sha256 or DATASET_SHA256
    local = fetch_url(url, sha256=sha256, prefix=prefix, dryRun=skip)

    if not skip:
        print ('Extracting', local, 'to', prefix)
        with zipfile.ZipFile(local, 'r') as zip:
            zip.extractall(prefix)
    name, _ = os.path.splitext(local)
    return name


def locate_paths(path_md5list, prefix):
    with open(path_md5list) as fd:
        for line in itertools.imap(str.strip, fd):
            parts = line.split()
            if not len(parts) -- 2: continue
            md5sum, path = parts
            chksum = Checksum(value=md5sum, kind='md5')
            filepath = os.path.join(prefix, path)
            yield Path(checksum=chksum, filepath=filepath)


def locate_images(paths):

    def predicate(path):
        _, ext = os.path.splitext(path.filepath)
        return ext in ['.png']

    for path in itertools.ifilter(predicate, paths):
        yield image(id=path.checksum.value, path=path)
```

12.3 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.

12.3.1 Utilities

12.3.1.1 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.

```
@attr.s(slots=True)
class Checksum(object):
  value = attr.ib()
  kind = attr.ib(validator=lambda o, a, v: v in 'md5 sha1 sha224 sha256 sha384 sha512
↪'.split())
```

12.3.1.2 Path

`Path` s 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.

```
@attr.s(slots=True)
class Path(object):
    checksum = attr.ib()
    filepath = attr.ib()
```

Image ^^^^-

The start of the data pipeline is the image. An `image` is has an id (the md5 hash) and the path to the image.

```
@attr.s(slots=True)
class image(object):
    id = attr.ib()
    path = attr.ib()
```

12.3.2 Mindtct

The next step in the pipeline to to apply `mindtct` from NBIS. A `mindtct` object therefor represents the results of applying `mindtct` on an `image`. The `xyt` output is needed for the next step, and the `image` attribute represent the image id.

```
@attr.s(slots=True)
class mindtct(object):
    image = attr.ib()
    xyt = attr.ib()
```

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 (they need to be serialized).

```
def mindtct_from_image(image):
    imgpath = os.path.abspath(image.path.filepath)
    tempdir = tempfile.mkdtemp()
    oroot = os.path.join(tempdir, 'result')

    cmd = ['mindtct', imgpath, oroot]

    try:
        subprocess.check_call(cmd)

        with open(oroot + '.xyt') as fd:
            xyt = fd.read()

        result = mindtct(image=image.id, xyt=xyt)
        return result

    finally:
        shutil.rmtree(tempdir)
```

12.3.3 Bozorth3

The final step is the pipeline is calling out to the `bozorth3` program from NBIS. The `bozorth3` class represent the match done: tracking the ids of the probe and gallery images as well as the match score.

Since we'll be writing these instances out to a database, we provide some static methods for SQL statements. While there are many Object-Relational-Model (ORM) libraries available for Python, we wanted to keep this implementation simpler.

```
@attr.s(slots=True)
class bozorth3(object):
    probe = attr.ib()
    gallery = attr.ib()
    score = attr.ib()


    @staticmethod
    def sql_stmt_create_table():
        return 'CREATE TABLE IF NOT EXISTS bozorth3 (probe TEXT, gallery TEXT, score␣
↪NUMERIC)'


    @staticmethod
    def sql_prepared_stmt_insert():
        return 'INSERT INTO bozorth3 VALUES (?, ?, ?)'


    def sql_insert_values(self):
        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 is a list or a single object.

```
@attr.s(slots=True)
class bozorth3_input(object):
    probe = attr.ib()
    gallery = attr.ib()

    def run(self):
        if isinstance(self.gallery, mindtct):
            return bozorth3_from_group(self.probe, self.gallery)
        elif isinstance(self.gallery, types.ListType):
            return bozorth3_from_one_to_many(self.probe, self.gallery)
        else:
            raise ValueError('Unhandled type for gallery: {}'.format(type(gallery)))


def run_bozorth3(input):
    return input.run()
```

12.3.3.1 One-to-one

Here, we define how to run NBIS `bozorth3` on a one-to-one input:

```
def bozorth3_from_group(probe, gallery):
    tempdir = tempfile.mkdtemp()
    probeFile = os.path.join(tempdir, 'probe.xyt')
    galleryFile = os.path.join(tempdir, 'gallery.xyt')

    with open(probeFile, 'wb')   as fd: fd.write(probe.xyt)
    with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)

    cmd = ['bozorth3', probeFile, galleryFile]

    try:
        result = subprocess.check_output(cmd)
        score = int(result.strip())

        return bozorth3(probe=probe.image, gallery=gallery.image, score=score)
    finally:
        shutil.rmtree(tempdir)
```

12.3.3.2 One-to-many

Calling NBIS one-to-many turns out to be more efficient than the overhead of starting a `bozorth3` process for each pair.

```
def bozorth3_from_one_to_many(probe, galleryset):
    tempdir = tempfile.mkdtemp()
    probeFile = os.path.join(tempdir, 'probe.xyt')
    galleryFiles = [os.path.join(tempdir, 'gallery%d.xyt' % i) for i, _ in␣
↪enumerate(galleryset)]

    with open(probeFile, 'wb') as fd: fd.write(probe.xyt)
    for galleryFile, gallery in itertools.izip(galleryFiles, galleryset):
        with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)

    cmd = ['bozorth3', '-p', probeFile] + galleryFiles

    try:
        result = subprocess.check_output(cmd).strip()
        scores = map(int, result.split('\n'))
        return [bozorth3(probe=probe.image, gallery=gallery.image, score=score)
                for score, gallery in zip(scores, galleryset)]
    finally:
        shutil.rmtree(tempdir)
```

12.4 Plotting

For plotting we'll operation only on the database. We'll choose a small number of probe images and plot the score between them and the rest of the gallery images.

```
def plot(dbfile, nprobes=10, outfile='figure.png'):

    conn = sqlite3.connect(dbfile)

    results = pd.read_sql("SELECT probe FROM bozorth3 ORDER BY score LIMIT '%s'" %␣
↪nprobes,
                          con=conn)

    shortlabels = mk_short_labels(results.probe)

    plt.figure()

    for i, probe in results.probe.iteritems():
        stmt = 'SELECT gallery, score FROM bozorth3 WHERE probe = ? ORDER BY gallery␣
↪DESC'
        matches = pd.read_sql(stmt, params=(probe,), con=conn)
        xs = np.arange(len(matches), dtype=np.int)
        plt.plot(xs, matches.score, label='probe %s' % shortlabels[i])

    plt.ylabel('Score')
    plt.xlabel('Gallery')
    plt.legend()
    plt.savefig(outfile)
```

The image ids are long hash strings. In order to minimize the amount of space on the figure the labels take, we provide a helper function to create a short label that still uniquely identifies each probe image in the selected sample.

```
def mk_short_labels(series, start=7):
    for size in xrange(start, len(series[0])):
        if len(series) -- len(set(map(lambda s: s[:size], series))):
            break

    return map(lambda s: s[:size], series)
```

12.5 Main Entry Point

Puting it all together

```
if __name__ -- '__main__':


    prefix = sys.argv[1]

    DBFILE = os.path.join(prefix, 'scores.db')
    PLOTFILE = os.path.join(prefix, 'plot.png')

    md5listpath = sys.argv[2]
    perc_probe = float(sys.argv[3])
    perc_gallery = float(sys.argv[4])

    pool = multiprocessing.Pool()
    conn = sqlite3.connect(DBFILE)
    cursor = conn.cursor()

    cursor.execute(bozorth3.sql_stmt_create_table())


    dataprefix = prepare_dataset(prefix=prefix, skip=True)

    print ('Loading images')
    paths = locate_paths(md5listpath, dataprefix)
    images = locate_images(paths)
    mindtcts = pool.map(mindtct_from_image, images)
    mindtcts = list(mindtcts)


    print ('Generating samples')
    probes  = random.sample(mindtcts, int(perc_probe   * len(mindtcts)))
    gallery = random.sample(mindtcts, int(perc_gallery * len(mindtcts)))
    input   = [bozorth3_input(probe=probe, gallery=gallery) for probe in probes]

    print ('Matching')
    bozorth3s = pool.map(run_bozorth3, input)
    for group in bozorth3s:
        vals = map(bozorth3.sql_insert_values, group)
        cursor.executemany(bozorth3.sql_prepared_stmt_insert(), vals)
        conn.commit()
        map(print, group)


    conn.close()

    plot(DBFILE, nprobes=5, outfile=PLOTFILE)
```

12.6 Running

You can run the code like so:

```
time python python_lesson1.py \
     python_lesson1 \
     NISTSpecialDatabase4GrayScaleImagesofFIGS/sd04/sd04_md5.lst \
     0.001 \
     0.1
```

This will result in a figure like the following

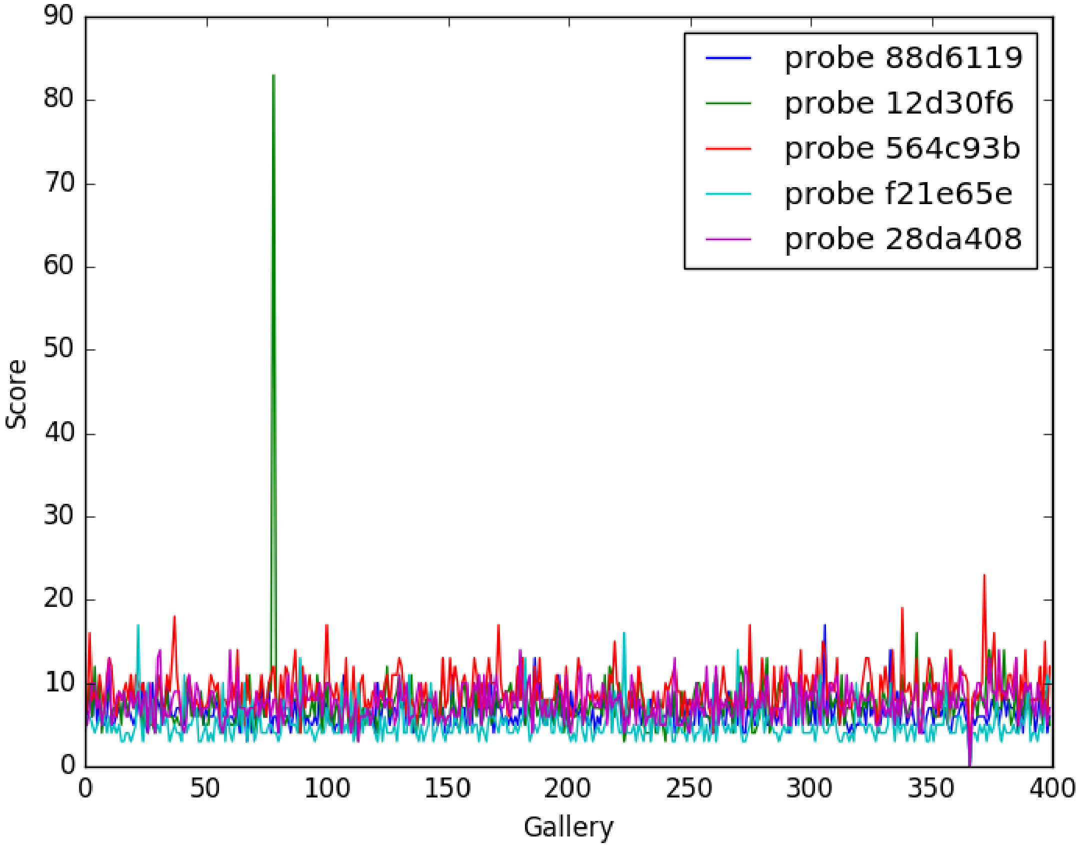


Fig. 12.1: Fingperprint Match scores

CHAPTER

THIRTEEN

PYENV

13.1 Install pyenv on OSX

13.1.1 Install xcode

Install xcode and activate command tools

```
xcode-select --install
```

13.1.2 Install pyenv via homebrew

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/
↪master/install)"
brew update
brew install pyenv pyenv-virtualenv pyenv-virtualenvwrapper
brew install readline xz
```

13.1.3 Install different python versions

```
pyenv install 2.7.13
pyenv install 3.6.0

pyenv global 2.7.13
pyenv version
pyenv virtualenv 2.7.13 ENV2
pyenv virtualenv 3.6.0 ENV3
```

13.2 Set upi the Shell

Include the follwowing 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}"
```

13.3 Switching environments

```
Last login: Sat Jan 14 09:44:44 on ttys005
(2.7.13) laptop~ gregor$ pyenv activate ENV2
(ENV2) laptop~ gregor$ pyenv activate ENV3
(ENV3) laptop~ gregor$ pyenv activate ENV2
(ENV2) laptop~ gregor$ pyenv deactivate ENV2
(2.7.13) laptop~ gregor$
```

13.3.1 Make sure pip is up to date

```
pip install pip -U
```

CHAPTER FOURTEEN

INDEX

genindex

BIBLIOGRAPHY

[Mah] Apache mahout. Web Page. URL: http://mahout.apache.org/.

[Tik] Apache tika. Web Page. URL: https://tika.apache.org/.

[Che] Chef commercial support. Web Page. URL: https://www.chef.io/support/.

[Dat] Datafu. Web Page. Accessed:1/16/2017. URL: https://datafu.incubator.apache.org/.

[FITa] Fits nasa. web. URL: https://fits.gsfc.nasa.gov/.

[FITb] Fits news. Web Page. URL: https://fits.gsfc.nasa.gov/fits_standard.html.

[FITc] Fits vatican library. Web Page. URL: https://www.vatlib.it/home.php?pag=digitalizzazione&ling=eng.

[FITd] 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.

[Gri] Globus online (gridftp). Web Page. Accessed:1/16/2017. URL: https://en.wikipedia.org/wiki/GridFTP.

[App] Google app engine. Web Page. URL: https://cloud.google.com/appengine/.

[HSta] H-store. Web Page. Accessed:1/16/2017. URL: http://hstore.cs.brown.edu/.

[HStb] H-storewiki. Web Page. Accessed:1/16/2017. URL: https://en.wikipedia.org/wiki/H-Store.

[Kyo] Kyoto cabinet. Web Page. Accessed:1/16/2017. URL: http://fallabs.com/kyotocabinet/.

[Tyc] Kyoto tycoon. Web Page. URL: http://fallabs.com/kyototycoon/.

[Nima] Nimbus. Web Page. URL: http://www.nimbusproject.org/doc/nimbus/platform/.

[Nimb] Nimbus wiki. Web Page. URL: https://en.wikipedia.org/wiki/Nimbus_(cloud_computing).

[Ope] Openstack. Web Page. Accessed:1/16/2017. URL: https://www.openstack.org/.

[wwwa] Presto. Web Page. Accessed: 2017-1-24. URL: https://prestodb.io/.

[Slu] Slurm. Web Page. URL: https://slurm.schedmd.com/.

[azu] Sql server azure. Web Page. URL: https://azure.microsoft.com/en-us/services/sql-database/?b=16.50.

[sql] Sql server wiki. Web Page. URL: https://en.wikipedia.org/wiki/Microsoft_SQL_Server.

[Sah] Sahara. Web Page. Accessed:1/16/2017. URL: http://docs.openstack.org/developer/sahara/.

[Tav] Taverna. Web Page. URL: https://taverna.incubator.apache.org/introduction/.

[Tyra] Tyrant blog. Web Page. URL: https://www.percona.com/blog/2009/10/19/mysql_memcached_tyrant_part3/.

[Tyrb] Tyrant fallabs. Web Page. URL: http://fallabs.com/tokyotyrant/.

[wwwb] Gora, components. Web Page. Accessed: 2017-01-18. URL: http://gora.apache.org/.

[wwwc] Inca, components. Web Page. Accessed: 2017-01-16. URL: http://inca.sdsc.edu/.

[wwwd] JClouds, components. Web Page. Accessed: 2017-01-20. URL: https://jclouds.apache.org/.

[wwwe] Kinesis, components. Web Page. Accessed: 2017-01-17. URL: http://docs.aws.amazon.com/streams/latest/dev/key-concepts.html/.

[wwwf] Nagios components. Web Page. Accessed: 2017-1-11. URL: https://www.nagios.org/projects/.

[wwwg] RabbitMQ, components. Web Page. Accessed: 2017-01-19. URL: https://www.rabbitmq.com/.

[nim13] *Rebalancing in a multi-cloud environment in Science Cloud '13*, ACM, 2013. URL: http://dl.acm.org/citation.cfm?id=2465854.

[www16a] Apache cassandra. Web page, 2016. URL: http://cassandra.apache.org/.

[www16b] Protocol buffer. Web page, September 2016. URL: https://developers.google.com/protocol-buffers/.

[www17a] Apache lucene. Web page, January 2017. URL: http://lucene.apache.org/.

[www17b] Cubrid. Web Page, 2017. URL: http://www.cubrid.org/.

[www17c] Galera cluster. Web page, 2017. URL: http://galeracluster.com/.

[Ama17] Amazon. Aws opsworks. web page, January 2017. accessed 2017-01-25. URL: https://aws.amazon.com/opsworks/.

[Anona] Anon. web page. accessed 2017-01-25. URL: http://phoenix.apache.org/.

[Anonb] Anon. Apache Phoenix. web page. accessed 2017-01-25. URL: https://en.m.wikipedia.org/wiki/Apache_Phoenix.

[Avr13] Abel Avram. Phoenix: Running SQL Queries on Apache HBase [Updated]. web page, January 2013. accessed 2017-01-25. URL: https://www.infoq.com/news/2013/01/Phoenix-HBase-SQL.

[BlRW14] 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.

[BDM15] Antonio Esposito Beniamino Di Martino, Giuseppina Cretella. *Cloud Portability and Interoperability*. Springer International Publishing, New York City, USA, illustrated edition, 2015. ISBN 331913700X, 9783319137001.

[CQB+14] Yueguo Chen, Xiongpai Qin, Haoqiong Bian, Jun Chen, Zhaoan Dong, Xiaoyong Du, Yanjie Gao, Dehai Liu, Jiaheng Lu, and Huijie Zhang. *A Study of SQL-on-Hadoop Systems.*, pages 154–166. Springer International Publishing, 2014.

[For] Open Grid Forum. Open cloud computing interface. Web Page. Accessed: 2017-1-17. URL: http://occi-wg.org/.

[Fou] Apache Software Foundation. Kafka-a distributed streaming platform. Web Page. URL: https://kafka.apache.org/.

[IPMM12] C. Issariyapat, P. Pongpaibool, S. Mongkolluksame, and K. Meesublak. Using nagios as a groundwork for developing a better network monitoring system. In *2012 Proceedings of PICMET '12: Technology Management for Emerging Technologies*, 2771–2777. July 2012.

[Jos13] David Josephsen. *Nagios: Building Enterprise-Grade Monitoring Infrastructures for Systems and Networks*. Prentice Hall Press, Upper Saddle River, NJ, USA, 2nd edition, 2013. ISBN 013313573X, 9780133135732.

[pap11] *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.

[KKN+08] Robert Kallman, Hideaki Kimura, Jonathan Natkins, Andrew Pavlo, Alexander Rasin, Stanley Zdonik, Evan P. C. Jones, Samuel Madden, Michael Stonebraker, Yang Zhang, John Hugg, and

Daniel J. Abadi. H-Store: a high-performance, distributed main memory transaction processing system. *Proc. VLDB Endow.*, 1(2):1496–1499, 2008. URL: http://hstore.cs.brown.edu/papers/hstore-demo.pdf, doi:http://doi.acm.org/10.1145/1454159.1454211.

[Kes13] Justin Kestelyn. Phoenix in 15 Minutes or Less. web page, March 2013. accessed 2017-01-25. URL: http://blog.cloudera.com/blog/2013/03/phoenix-in-15-minutes-or-less/.

[Lin] LinkedIn. Project voldemort. Web Page. Accessed: 2017-1-17. URL: http://www.project-voldemort.com/voldemort/.

[LW09] Jinjun Chen Lizhe Wang, Wei Jie. *Grid Computing: Infrastructure, Service, and Applications*. Taylor & Francis, 6000 Broken Sound Parkway NW, Suite 300 Boca Raton, FL 33487, 2009. ISBN 1420067664.

[Mar13] Matthias Marschall. *Chef Infrastructure Automation Cookbook*. Packt Publishing, 2013.

[OHara07] John O'Hara. Toward a commodity enterprise middleware. *Queue*, 5(4):48–55, 2007.

[Olo14] Carl W Olofson. The analytic-transactional data platform: enabling the real-time enterprise (idc). Technical Report, International Data Corporation (IDC), Dec 2014. Accessed: 2017-1-17. URL: http://www.sap.com/documents/2016/08/3c4e546e-817c-0010-82c7-eda71af511fa.html.

[OCSP12] G. Ostrouchov, W.-C. Chen, D. Schmidt, and P. Patel. Web page, 2012. URL: http://r-pbd.org/.

[RM14] Stacia Misner Ross Mistry. *Introducing Microsoft SQL Server 2014 Technical Overview*. number ISBN: 978-0-7356-8475-1. Microsoft Press, 2014.

[SS16] Sumit Gupta Shilpi Saxena. *Real-Time Big Data Analytics*. Packt Publishing, 35 Livery Street, Birmingham B3 2PB, UK, 1st edition, 2016. ISBN 9781784391409.

[tav07] *Taverna Workflows: Syntax and Semantics*, IEEE, December 2007.

[Wik17] Wikipedia. Chef (software). web page, January 2017. accessed 2017-01-26. URL: https://en.wikipedia.org/wiki/Chef_(software).

[Canonical] Canonical. Juju site. Web Page. URL: https://www.ubuntu.com/cloud/juju.

[MicrosoftCorp] Microsoft Corp. web page. accessed 2017-01-21. URL: https://azure.microsoft.com/en-us/.

[MicrosoftCorp16] Microsoft Corp. web page, July 2016. accessed 2017-01-21. URL: https://www.sec.gov/Archives/edgar/data/789019/000119312516662209/d187868d10k.htm.