University of
Zurich^UZH

S3IT

IT infrastructure for research: an ongoing journey

Riccardo Murri
Services and Support for Science IT, University of Zurich
riccardo.murri@gmail.com

2018-11-13

Two disclaimers

Opinions and views expressed here are mine only, and may not reflect the official stance of UZH, its IT services, or my colleagues.

Although I have tried to report on scientific research accurately, there can still be errors and inaccuracies. They are all my faults.

What is Research IT?

. . . in the words of those who do it

> "S3IT supports UZH researchers
> in using IT to empower their research,
> from consultancy to application support
> and access to cutting-edge cloud, cluster
> and supercomputing systems."
>
> (source: https://www.s3it.uzh.ch/)

. . . in the words of those who *use* it

From: some.one@uzh.ch
Subject: computing power

Dear Madam/Sir,

I have been invited to submit a revision of the attached paper. There are some missing numbers in Table 1, since **I did not have enough computing power on my office computer to carry out these computations.** A referee has asked us for them, **therefore I need access to a supercomputer.**

Many thanks,
Some One

Traditional options for scientific computing

- Personal workstations
  - Limited: how much computing power can fit under your desk?
  - More freedom, more responsibilities
- Large shared batch-queuing systems
  - Centrally provided and administered
  - Typically a GNU/Linux cluster nowadays.

Large, shared, batch-queuing clusters

Large

Shared

Batch-queuing

Can allow much larger degree of parallelism compared to own workstation.

Limit is institutional money.
(Dept. / University / Computing Centre)

Large, shared, batch-queuing clusters

Large

Shared

Batch-queuing

In particular: *same* OS and *same* set of installed software for all, *same* scheduler configuration for all, *same* filesystem(s) for all . . .

So, installed software and usage is subject to **policies**.

Large, shared, batch-queuing clusters

Large

Shared

Batch-queuing clusters

"Cluster" is the architecture:

- standard ("commodity") servers as compute nodes
- high-performance network interconnecting them
- shared filesystem(s)
- job scheduler to allocate resources

Reference: D. Becker, Th. Sterling, et al.: *BEOWULF: A parallel workstation for scientific computation*, in: Proceedings, International Conference on Parallel Processing vol. 95, (1995). http://www.phy.duke.edu/~rgb/brahma/Resources/beowulf/papers/ICPP95/icpp95.html

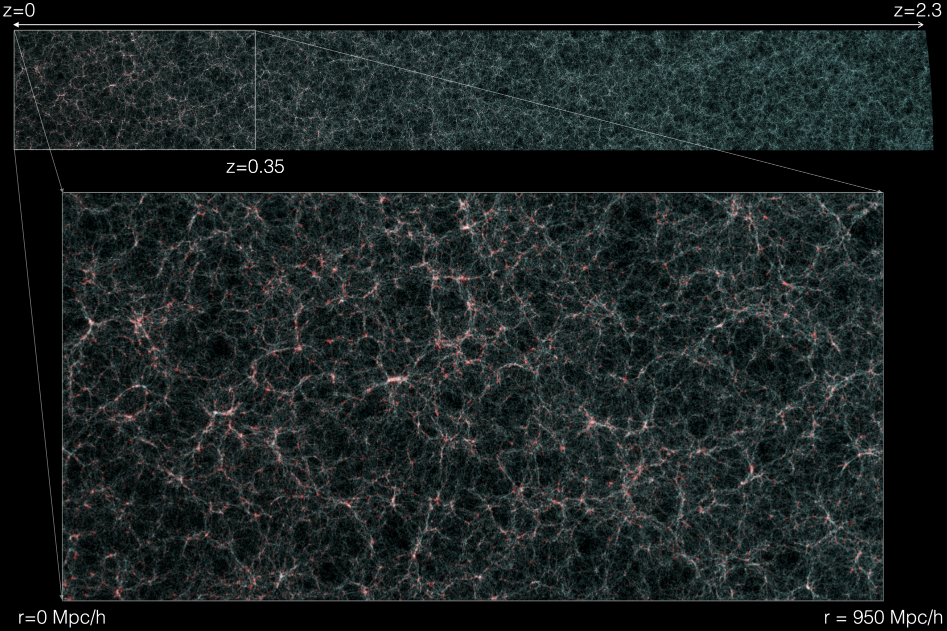
PKDGRAV3

Large *N*-body simulation code.

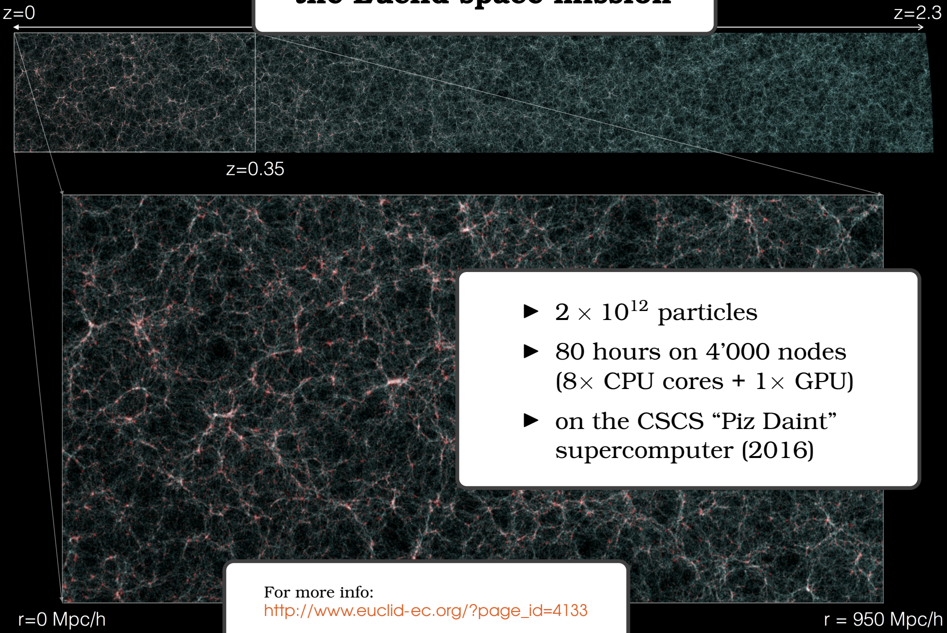
Written by Joachim Stadel, Doug Potter, and collaborators at UZH.

PKDGRAV3: beyond trillion particle cosmological simulations for the next era of galaxy surveys
D. Potter, J. Stadel, R. Teyssier - Computational Astrophysics and Cosmology, 2017

Flagship mock galaxy catalog



Create test dataset for the Euclid space mission



- 2×10^{12} particles
- 80 hours on 4’000 nodes (8× CPU cores + 1× GPU)
- on the CSCS “Piz Daint” supercomputer (2016)

For more info:
http://www.euclid-ec.org/?page_id=4133

PKDGRAV3: computation and communication

- Fast Multipole Method: $O(N)$
- Communication overlaps with computation
  - one CPU core dedicated to MPI communication
  - latency is more important than bandwidth!
  - supported by Cray's custom "Aries" interconnect

PKDGRAV3: checkpointing and filesystem I/O

- Checkpoints: 20× 48 TB spread over 20× 28'000 files.
  - *Synchronous:* calculation must stop and wait until file is dumped
  - approx. 2GB per file
  - 1 file per computing thread

PKDGRAV3: checkpointing and filesystem I/O

- Checkpoints: 20× 48 TB spread over 20× 28’000 files.
  - *Synchronous:* calculation must stop and wait until file is dumped
  - approx. 2GB per file
  - 1 file per computing thread

Checkpoints are *needed* to overcome the 24h max runtime policy!

TissueMAPS

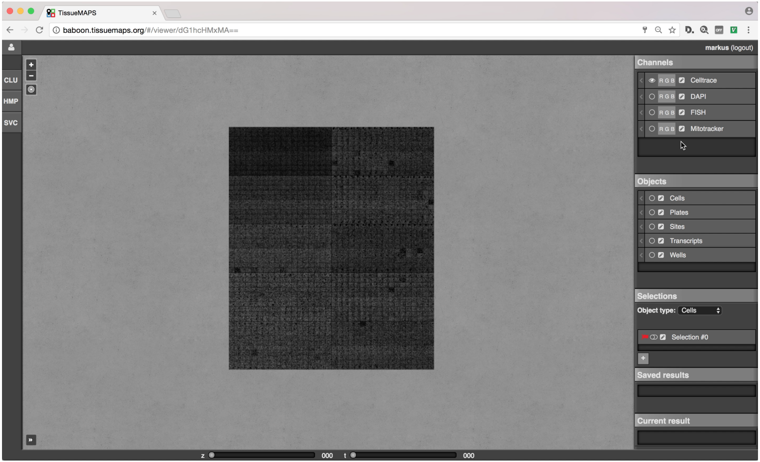
Scalable platform for image analysis of microscopy images.

- Developed for image-based cell profiling
- Automated workflow for microscopy image processing
- Browser-based client to explore results and command further analysis

Reference: “Computational Methods and Tools for Reproducible and Scalable Bioimage Analysis”

— M. D. Herrmann, Ph.D. Thesis, Univ. of Zurich (2017).

TissueMAPS: Demo of “Transcriptomics” data

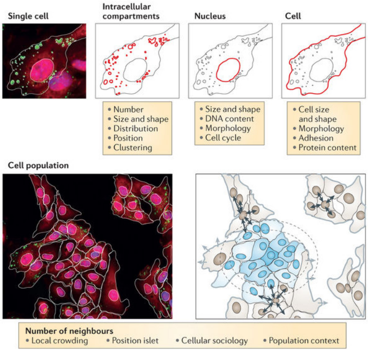


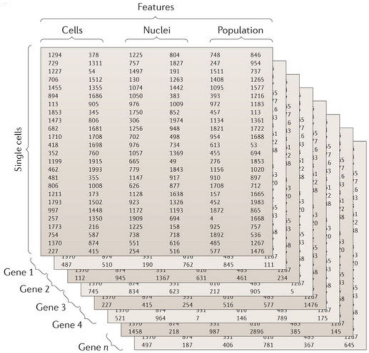
baboon.tissuemaps.org/#/viewer/dG1hcHMxMA==

Object type: Cells

https://youtu.be/Qmqf0ysDrx0

Image-based Cell Profiling





Cells		Nuclei		Population	
1294	378	1225	804	748	846
729	1311	757	1827	247	954
1227	54	1497	191	1511	737
706	1512	130	1263	1408	1265
1455	1355	1074	1442	1095	1577
894	1686	1050	383	393	1216
113	905	976	1009	972	1183
1853	345	1750	852	457	113
1473	806	306	1974	1134	1361
682	1681	1256	948	1821	1722
1710	1708	702	498	954	1688
418	1698	976	734	613	53
352	760	1057	1369	455	694
1199	1915	665	49	276	1853
462	1993	779	1843	1156	1020
481	355	1147	917	910	897
806	1008	626	877	1708	712
1211	173	1128	1638	157	1665
1793	1502	923	1326	452	1983
997	1448	1172	1193	1872	865
257	1350	1909	694	4	1668
1773	216	1225	158	925	757
754	587	738	718	1892	536
1370	874	551	616	485	1267
227	415	254	516	577	1476

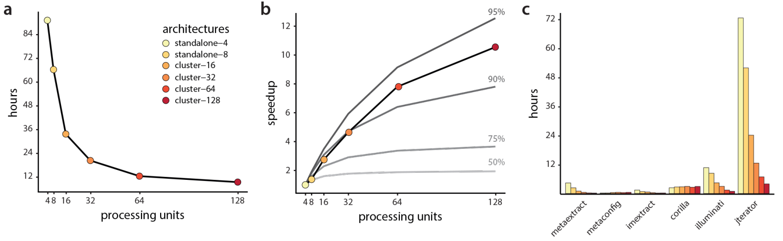
Reference: "Single-cell and multivariate approaches in genetic perturbation screens"

— P. Liberali, B. Snijder, L. Pelkmans, Nat. Rev. Genet., 16:18–32 (2015)

TissueMAPS: Scalability

Time for processing 35’280 microscope images on clusters of varying size.

- Almost perfectly scalable (10× speedup)
  - see figure b — in gray, theoretical speedup for different levels of parallelization
- The “image analysis” step benefits the most from larger resources



TissueMAPS: computational features

Pure “embarassingly parallel” workload:

- many short-lived independent jobs
- *very many* small files
- Sharded DB used to store and process real-time visualization data

For instance, in the “transcriptomics” data set:

- input microscope images: $352'800$ images, a few MBs each
- pyramid tiles: $41'231'720$, a few kB each
- DB table for object features: 650M rows

Conflicting requirements!

PKDGRAV3	**TissueMAPS**
Single large MPI job.	Huge swarm of short-lived jobs
Low-latency communication.	No communication across tasks.
Relatively small number of large files.	Huge number of small to tiny files.
Adapted to (high-end) cluster computing environment.	Requires setup of custom DB and web-service endpoints.

Conflict on Scheduling Policies

From: unhappy.user@uzh.ch
Subject: cluster priorities

Dear all,

despite my occasional complaints, it has never been explained that group *X* has a default higher priority on the cluster.

It leads to user *Y* being able to use 3120 cores at the time of writing with all(!) other users combining for 824 cores despite those users having eligible jobs in the queue.

My feeling is that the policy seems outdated and (nowadays) inappropriate.

Cheers + thanks, *Z.*

How do Tweets affect the Movie Box Office?

WARREN BEATTY is
THE FLY
Twentieth Century Fox Presents
A Robert H. Solo Production
A Philip Kaufman Film "The Fly"
Screenplay by W.D. Richter
Based on the story by George Langelaan
Music by Denny Zeitlin
Produced by Robert H. Solo
Directed by Philip Kaufman
DOLBY STEREO
R
ALSO STARRING CAROL LYNLEY AND STERLING HAYDEN
WITH ONE TOUCH THIS MAN WILL GAIN ALL YOUR SECRETS AND KNOW HOW YOU WILL DIE! STAY AWAY IF YOU DARE!
WILLIAM PETERSEN
MARY STUART MASTERSON
ZODIA
MAX VON SYDOW
as SIGMUND FREUD
PAMELA GRIER
IS BACK
IN
But she fell into something even more horrific...
A prison planet full of the most dangerous men across the universe
-in-

Images Copyright © 2015 Peter Stults
https://www.behance.net/gallery/25965817/What-If-Movie-Posters-Vol-V

"The Impact of Twitter on New Product Performance"
(work in progress)
— L. Deer, P. Chintagunta, and G. S. Crawford,
http://lachlandeer.github.io/pages/research.html

How does Word-of-Mouth on Twitter affect a movie's performance at the box office?

How do Tweets affect the Movie Box Office?

Try and isolate mechanisms by which Twitter is influencing demand — a computational experiment.

- Get the data:
  - Twitter stream dump
    - 300 movies
    - ± 6 months from release date
  - Box Office performance
- Analyze & Model
  - 85% of Tweets are in the English language — **Filter** out the rest!
  - **Categorize** each Tweet
    - advertisement, buzz, review — each category may affect the dynamics differently
  - **Compute** sentiment score of tweets
  - **Correlate** to Box Office timeseries data

How do Tweets affect the Movie Box Office?

Try and isolate mechanisms by which Twitter is influencing demand — a computational experiment.

- Get the data:
  - Twitter stream dump
    - 300 movies
    - ± 6 months from release date
  - Box Office perfor
- Analyze & Model
  - 85% of Tweets ar
    — **Filter** out the
  - **Categorize** each Tweet
    - advertisement, buzz, review
      — each category may affect the dynamics differently
  - **Compute** sentiment score of tweets
  - **Correlate** to Box Office timeseries data

Classical data science workflow!

Spark/Hadoop are the go-to tools.

How do Tweets affect the Movie Box Office?

Try and isolate mechanisms by which Twitter is influencing demand — a computational experiment.

- Get the data:
  - Twitter stream dump
    - 300 movies
    - ± 6 months from release date
  - Box Office perfor
- Analyze & Model
  - 85% of Tweets ar
    — **Filter** out the
  - **Categorize** each
    - advertisemen
      — each categ
      differently
  - **Compute** sentiment score of tweets
  - **Correlate** to Box Office timeseries data

Classical data science workflow!

Spark/Hadoop are the go-to tools.

Oh, wait. . . Do we have a Spark/Hadoop cluster here?

“There’s no problem we cannot solve
by adding one more layer of indirection.”
— *Fundamental Theorem of Software Engineering*

Abstract away the Infrastructure Layer!

Use *Infrastructure-as-a-Service* as a base for providing compute infrastructure.

We can create and setup ad-hoc computing infrastructures:

- *dedicated*: no sharing, exactly the software and policies you want
- *ephemeral*: create when idea comes, dispose when experiment is over

IaaS cloud computing

1. Provision *virtual* resources:
   - virtual machines (VM)
   - block and object storage
   - software-defined networking
2. Pay per use
   - No upfront investment in HW
3. Network-accessible API for control
   - allows *scripting* the set-up and tear-down of infrastructure
   - “infrastructure as code”

Advantages of “Infrastructure as Code”

1. Reproducibility
   - You can re-create the exact same infrastructure at a later time.
2. Version Control
3. Easy to clone/adapt
4. Readability

Advantages of "Infrastructure as Code"

1. Reproducibility
2. Version Control
   - can easily roll back changes!
   - precise log of how the infrastructure evolved over time
   - ... plus all niceties that we have from coding environments
3. Easy to clone/adapt
4. Readability

Advantages of “Infrastructure as Code”

1. Reproducibility
2. Version Control
3. Easy to clone/adapt
   - It’s just text files!
   - Good configuration/deployment tools have a programming languages: functions allow defining *“parametric infrastructure”*
4. Readability

Advantages of “Infrastructure as Code”

1. Reproducibility
2. Version Control
3. Easy to clone/adapt
4. Readability
    - Well-written code counts as documentation of the infrastructure setup

“Software-defined Sysadmin”

However, there are infrastructure setup chores:

- e.g., software installation and configuration
- now you must do these yourself!

ElastiCluster is our solution for automation of basic sysadmin tasks: provisioning and initial setup of a computing infrastructure.

What is ElastiCluster

ElastiCluster provides a **command line tool** and a Python API to **create, set up and resize** computing clusters hosted on IaaS cloud infrastructures.

Main function is to get a compute cluster up and running with a single command.

Effectively, a wrapper around **Ansible** which provides:

- idempotent configuration playbooks
- no-bootstrap remote actions via SSH

ElastiCluster features (1)

Computational clusters supported:

- Batch-queuing systems:
  - SLURM
  - GridEngine
  - Torque+MAUI
  - HTCondor
- Kubernetes
- Mesos
- Spark / Hadoop

Distributed storage:

- CephFS
- GlusterFS
- HDFS
- OrangeFS/PVFS

Optional add-ons:

- Ganglia
- JupyterHub
- EasyBuild

(Grayed out items have not been tested in a while. . .)

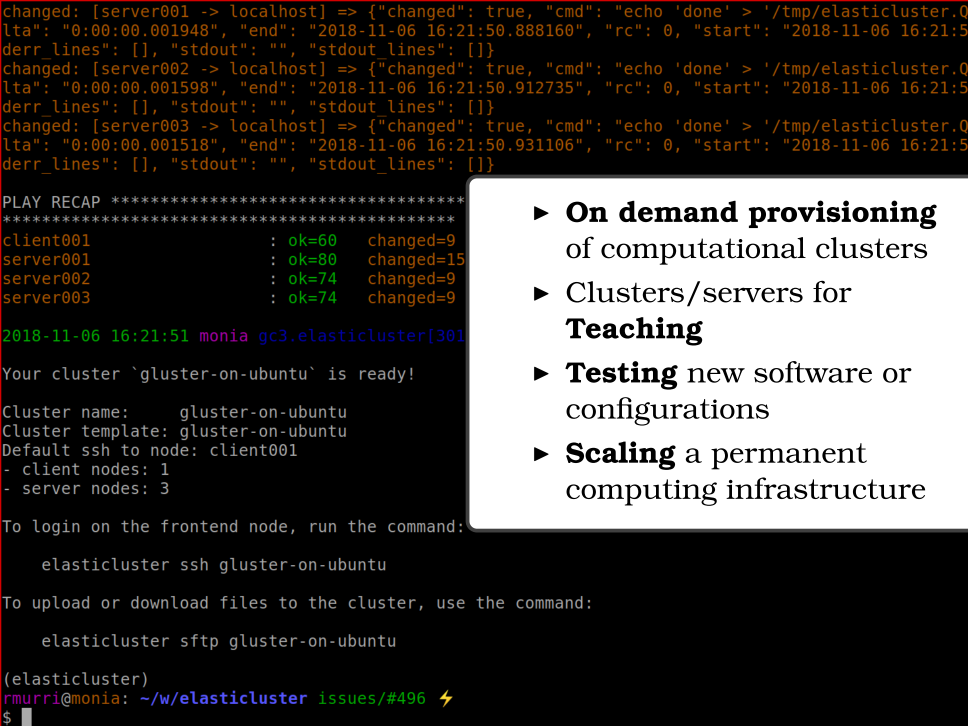
ElastiCluster features (2)

Run on multiple clouds:

- Amazon EC2
- Google Compute Engine
- OpenStack
- MS Azure
- . . . and anything supported by LibCloud

Supports several distros as base OS:

- Debian 9.x *(stretch)*, 8.x *(jessie)*
- Ubuntu 18.04 *(bionic)*, 16.04 *(xenial)*, 14.04 *(trusty)*,
- CentOS / Scientific Linux 7.x



```
changed: [server001 -> localhost] => {"changed": true, "cmd": "echo 'done' > '/tmp/elasticluster.Q
lta": "0:00:00.001948", "end": "2018-11-06 16:21:50.888160", "rc": 0, "start": "2018-11-06 16:21:5
derr_lines": [], "stdout": "", "stdout_lines": []}
changed: [server002 -> localhost] => {"changed": true, "cmd": "echo 'done' > '/tmp/elasticluster.Q
lta": "0:00:00.001598", "end": "2018-11-06 16:21:50.912735", "rc": 0, "start": "2018-11-06 16:21:5
derr_lines": [], "stdout": "", "stdout_lines": []}
changed: [server003 -> localhost] => {"changed": true, "cmd": "echo 'done' > '/tmp/elasticluster.Q
lta": "0:00:00.001518", "end": "2018-11-06 16:21:50.931106", "rc": 0, "start": "2018-11-06 16:21:5
derr_lines": [], "stdout": "", "stdout_lines": []}

PLAY RECAP *********************************
*********************************************
client001                  : ok=60   changed=9
server001                  : ok=80   changed=15
server002                  : ok=74   changed=9
server003                  : ok=74   changed=9

2018-11-06 16:21:51 monia gc3.elasticluster[301

Your cluster `gluster-on-ubuntu` is ready!

Cluster name:     gluster-on-ubuntu
Cluster template: gluster-on-ubuntu
Default ssh to node: client001
- client nodes: 1
- server nodes: 3

To login on the frontend node, run the command:

    elasticluster ssh gluster-on-ubuntu

To upload or download files to the cluster, use the command:

    elasticluster sftp gluster-on-ubuntu

(elasticluster)
rmurri@monia: ~/w/elasticluster issues/#496 ⚡
$ 
```

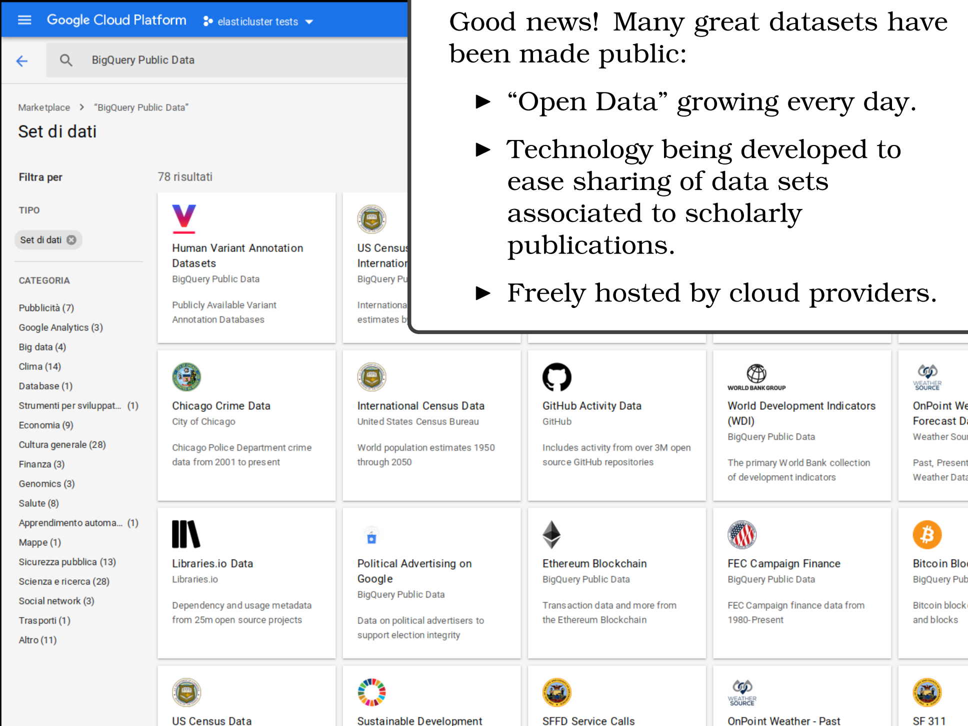
- **On demand provisioning** of computational clusters
- Clusters/servers for **Teaching**
- **Testing** new software or configurations
- **Scaling** a permanent computing infrastructure

No Compute without Data

Unless you're modeling something from first principles, you need data to base your computations on.

Good news! Many great datasets have been made public:

- ▶ “Open Data” growing every day.
- ▶ Technology being developed to ease sharing of data sets associated to scholarly publications.
- ▶ Freely hosted by cloud providers.



Marketplace > “BigQuery Public Data”

Set di dati

Filtra per

78 risultati

TIPO

Set di dati

CATEGORIA

Pubblicità (7)
Google Analytics (3)
Big data (4)
Clima (14)
Database (1)
Strumenti per sviluppat... (1)
Economia (9)
Cultura generale (28)
Finanza (3)
Genomics (3)
Salute (8)
Apprendimento automa... (1)
Mappe (1)
Sicurezza pubblica (13)
Scienza e ricerca (28)
Social network (3)
Trasporti (1)
Altro (11)

Human Variant Annotation Datasets
BigQuery Public Data
Publicly Available Variant Annotation Databases

US Census
Internation
BigQuery Pu
Internationa
estimates b

Chicago Crime Data
City of Chicago
Chicago Police Department crime data from 2001 to present

International Census Data
United States Census Bureau
World population estimates 1950 through 2050

GitHub Activity Data
GitHub
Includes activity from over 3M open source GitHub repositories

WORLD BANK GROUP
World Development Indicators (WDI)
BigQuery Public Data
The primary World Bank collection of development indicators

WEATHER SOURCE
OnPoint We
Forecast D
Weather Sou
Past, Present
Weather Data

Libraries.io Data
Libraries.io
Dependency and usage metadata from 25m open source projects

Political Advertising on Google
BigQuery Public Data
Data on political advertisers to support election integrity

Ethereum Blockchain
BigQuery Public Data
Transaction data and more from the Ethereum Blockchain

FEC Campaign Finance
BigQuery Public Data
FEC Campaign finance data from 1980-Present

Bitcoin Blo
BigQuery Pub
Bitcoin block
and blocks

US Census Data

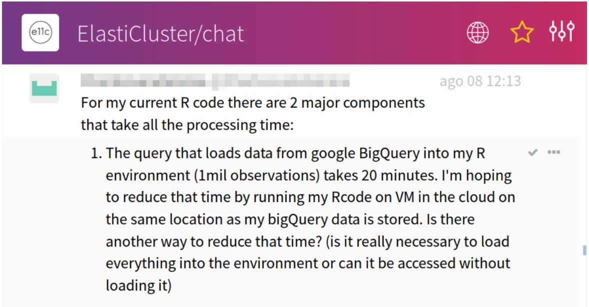
Sustainable Development

SFFD Service Calls

WEATHER SOURCE
OnPoint Weather - Past

SF 311

Access speed still an issue?



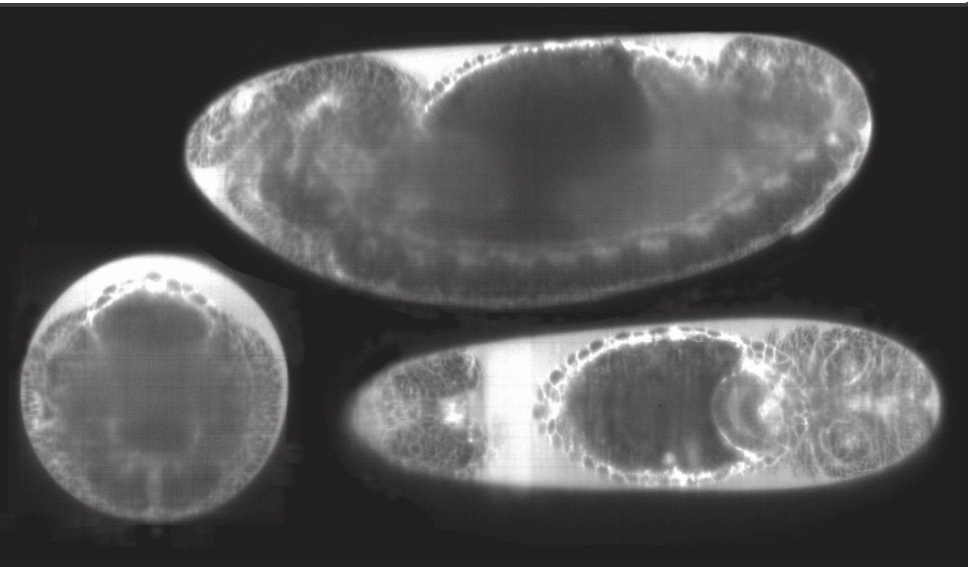
[redacted] ago 08 12:13

For my current R code there are 2 major components that take all the processing time:

1. The query that loads data from google BigQuery into my R environment (1mil observations) takes 20 minutes. I'm hoping to reduce that time by running my Rcode on VM in the cloud on the same location as my bigQuery data is stored. Is there another way to reduce that time? (is it really necessary to load everything into the environment or can it be accessed without loading it)

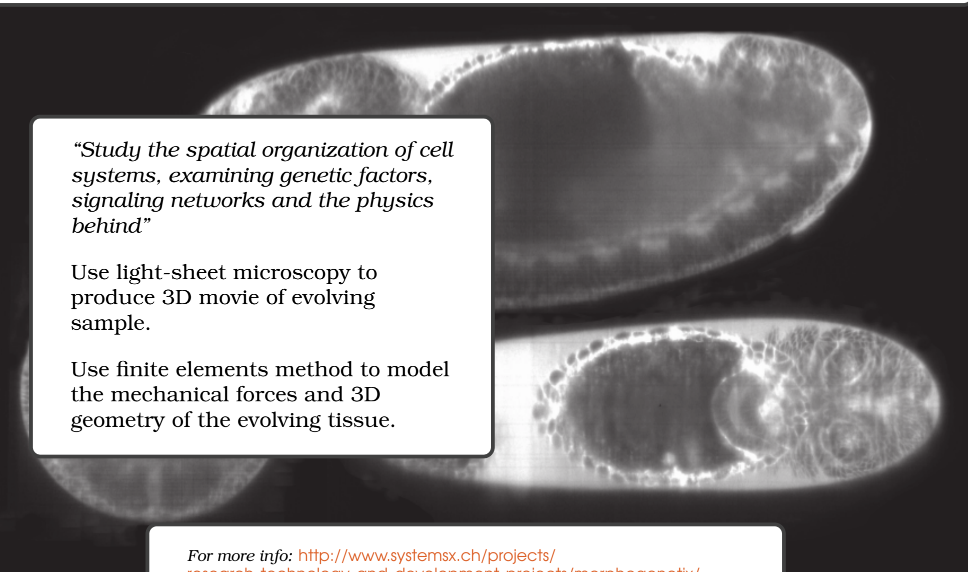
"The query that loads data from BigQuery into my R environment *[. . .]* takes 20 minutes."

MorphogenetiX: Modelling 3-D Shaping of Tissues



Single sections of a fly embryo imaged in 3D with light sheet microscopy. Clockwise from top: side view, top view, frontal view. © Damian Brunner

MorphogenetiX: Modelling 3-D Shaping of Tissues



"Study the spatial organization of cell systems, examining genetic factors, signaling networks and the physics behind"

Use light-sheet microscopy to produce 3D movie of evolving sample.

Use finite elements method to model the mechanical forces and 3D geometry of the evolving tissue.

For more info: http://www.systemsx.ch/projects/research-technology-and-development-projects/morphogenetix/

Single sect nicroscopy.
Clockwise from top: side view, top view, frontal view. © Damian Brunner

MorphogenetiX: Modelling 3-D Shaping of Tissues

Use light-sheet microscopy to produce 3D movie of evolving sample.

- Up to 8TB of data every 4 hours.

Post-process images to generate discretized model and connectivity information.

From FEM model run simulation of embryo development.

- Again, large production of data.

MorphogenetiX: Modelling 3-D Shaping of Tissues

Use light-sheet microscopy to produce 3D movie of evolving sample.

- Up to 8TB of data every 4 hours.

Post-process i
connectivity i

From FEM mo
development.

- Again, lar

Bandwidth to data center ≈ 1Gbit/s.

16 hours to copy 8TB.

4× times more than to produce it!

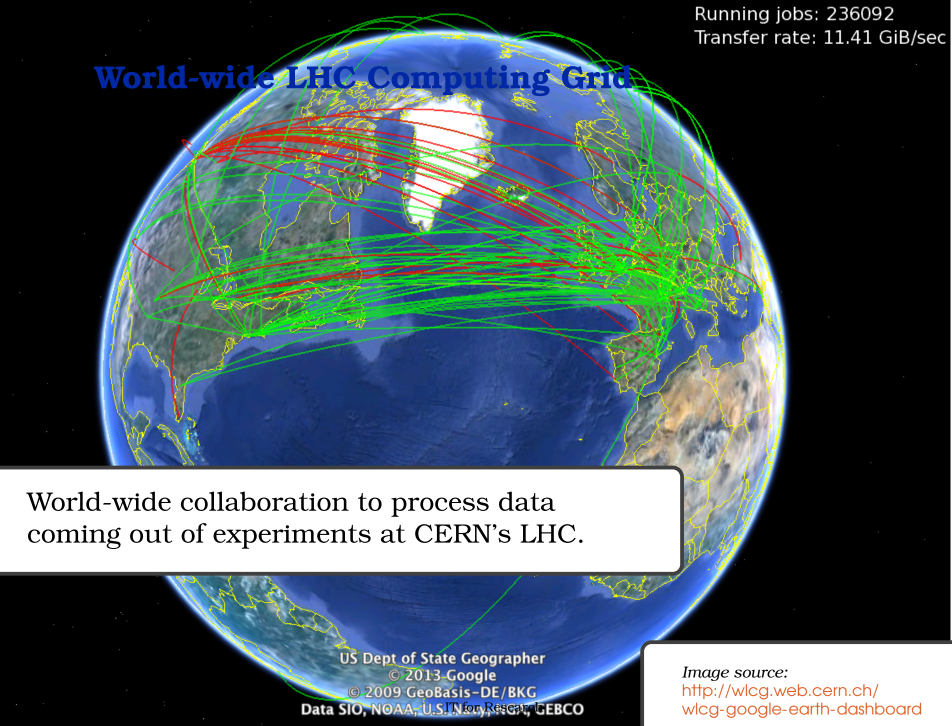
Need to filter data at source.

Not getting better short-term

“a 10x *(network)* speed increase over 15 years
is far slower than the 2x speed per 1.5 years
typically cited for Moore’s law.”
— https://en.wikipedia.org/wiki/100_Gigabit_Ethernet

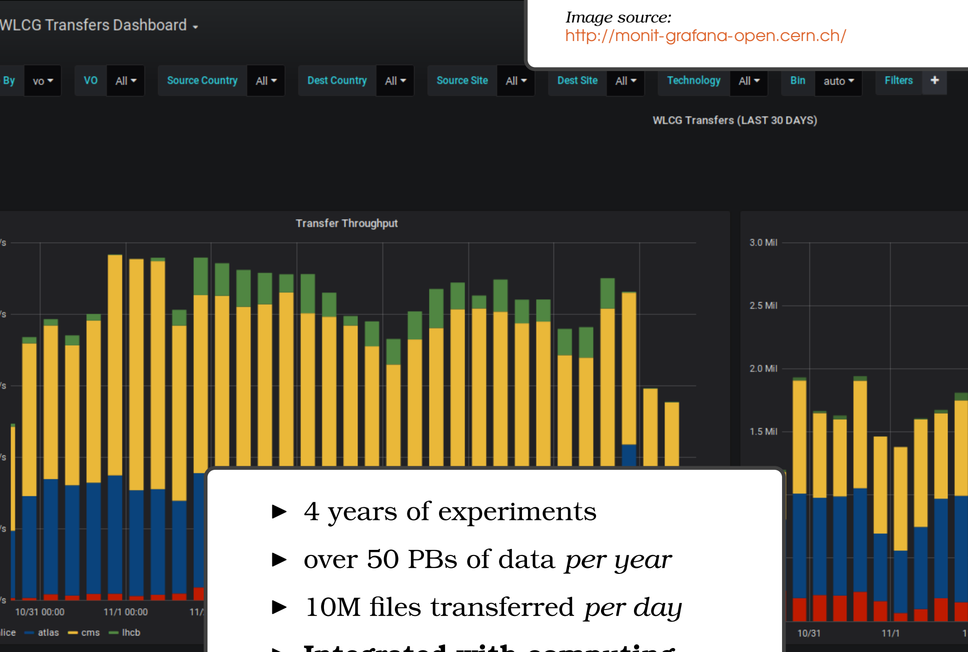
“Recent growth in *(genome)* sequencing technology
eclipses Moore”
— https://blog.acolyer.org/dna-storage-fig-1/

World-wide LHC Computing Grid


Running jobs: 236092
Transfer rate: 11.41 GiB/sec
US Dept of State Geographer
© 2013 Google
© 2009 GeoBasis-DE/BKG
Data SIO, NOAA, U.S. Navy, NGA, GEBCO

World-wide collaboration to process data coming out of experiments at CERN’s LHC.

Image source: http://wlcg.web.cern.ch/wlcg-google-earth-dashboard



WLCG Transfers Dashboard

By: vo | VO: All | Source Country: All | Dest Country: All | Source Site: All | Dest Site: All | Technology: All | Bin: auto | Filters +

Image source:
http://monit-grafana-open.cern.ch/

- 4 years of experiments
- over 50 PBs of data *per year*
- 10M files transferred *per day*
- **Integrated with computing grid!**

Data is *the* problem for experimental science

- Data can be produced faster than it can be moved.
    - HEP model: few large experiment sites, well-connected to high-speed Internet backbone.
- Some data comes with strict legal requirements attached!

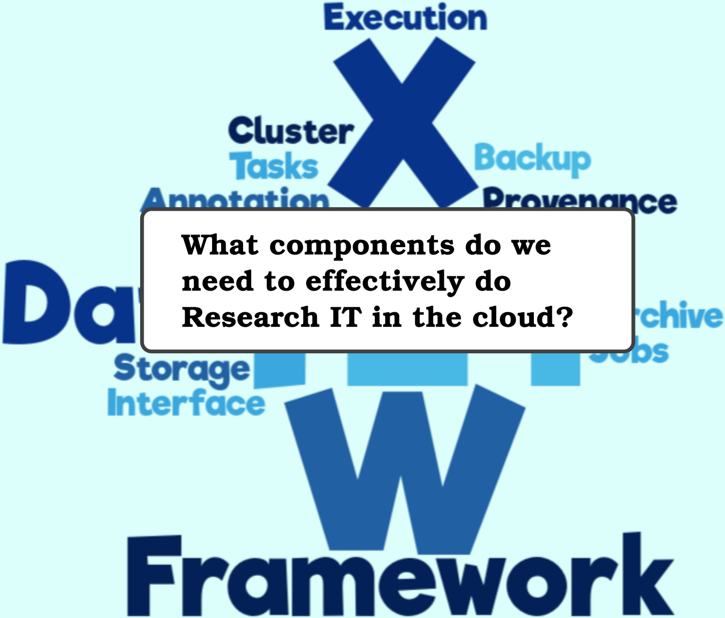
Is that all?

70% of requests to access to the UZH GPU cluster are for running neural nets code.

Do users *really* have to go through batch-queuing commands to run a TensorFlow model?

```
sbatch --mem=48g --time=1-0:00:00 \
  --gres=gpu:3 --qos=16MemPerGpu [...]
```

Towards a Science Cloud?



What components do we need to effectively do Research IT in the cloud?

Recall the traditional “layer model” of cloud services:

SaaS	End-user applications
PaaS	Resources and tools to create apps
IaaS	Virtualization of physical resources

A layers model for a Science Cloud?

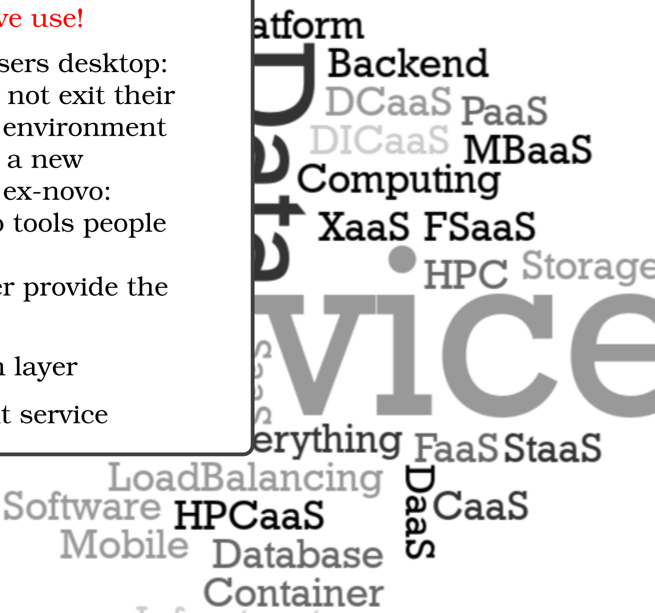
Interface	TissueMAPS, JupyterHub, CJ
Framework	Spark, TensorFlow, MPI, GC3Pie
Execution	(Elasti)Clusters, Hadoop, Mesos, Kubernetes
Infrastructure	public and on-prem cloud

What will a Science Cloud look like?

- Support interactive use!
- Flexible execution layer
- Data management service

What will a Science Cloud look like?

- Support interactive use!
  - Extends to users desktop: users should not exit their development environment
  - Cannot write a new environment ex-novo: integrate into tools people already use
  - Can container provide the bridge?
- Flexible execution layer
- Data management service



What will a Science Cloud look like?

- Support interactive use!
- Flexible execution layer
  - Need to scale!
  - Reconfigurable mix of batch-queueing and other paradigms
- Data management service

What will a Science Cloud look like?

- Support interactive use!
- Flexible execution layer
- Data management service
  - Manages data life cycle: from production, to consumption, to archival
  - Not necessarily a filesystem
  - Data format aware: can slice, filter, pre-process . . .

. . . but in the end, it’s a people’s thing

- IT support moving closer to researchers and away from infrastructure
- Interdisciplinary teams will be key

Special thanks go to . . .

. . . to the ElastiCluster fellow devs:
Antonio Messina, Nicolas Bär

. . . to my colleagues at GC3/S3IT:
Sergio Maffioletti, Tyanko Aleksiev

. . . to the Scientists who contributed:
Lachlan Deer, David Dreher, Markus D. Herrmann,
Lucas Pelkmans, Doug Potter, Joachim Stadel

Thanks!

(Any questions?)

Appendix

ElastiCluster

On-demand provisioning of compute clusters

TissueMAPS

- Deploy on cloud: compute cluster + parallel DB + web front-end

WLCG

- Deploy compute cluster with SL6.*x*

"Twitter Effect on Movies" experiment

- Deploy Spark + JupyterHub

PKDGRAV3

- Still need a real HPC cluster!

On-demand provisioning of compute clusters

:-) TissueMAPS
  - Deploy on cloud: compute cluster + parallel DB + web front-end

WLCG
  - Deploy compute cluster with SL6.*x*

"Twitter Effect on Movies" experiment
  - Deploy Spark + JupyterHub

PKDGRAV3
  - Still need a real HPC cluster!

On-demand provisioning of compute clusters

- :-) TissueMAPS
  - Deploy on cloud: compute cluster + parallel DB + web front-end
- :-) WLCG
  - Deploy compute cluster with SL6.*x*
- “Twitter Effect on Movies” experiment
  - Deploy Spark + JupyterHub
- PKDGRAV3
  - Still need a real HPC cluster!

On-demand provisioning of compute clusters

- :-) TissueMAPS
  - Deploy on cloud: compute cluster + parallel DB + web front-end
- :-) WLCG
  - Deploy compute cluster with SL6.*x*
- :-) "Twitter Effect on Movies" experiment
  - Deploy Spark + JupyterHub
- PKDGRAV3
  - Still need a real HPC cluster!

On-demand provisioning of compute clusters

:-) TissueMAPS

- Deploy on cloud: compute cluster + parallel DB + web front-end

:-) WLCG

- Deploy compute cluster with SL6.*x*

:-) “Twitter Effect on Movies” experiment

- Deploy Spark + JupyterHub

:-(PKDGRAV3

- Still need a real HPC cluster!

Clusters for teaching

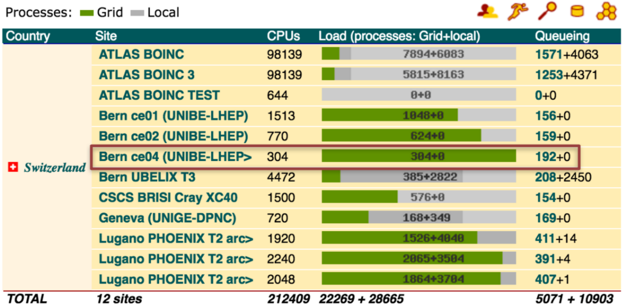
Example: JupyterHub+Spark clusters

- for teaching courses (e.g., data science), or
- for short-lived events (e.g., workshops).

Key ingredient is the ability to apply custom Ansible playbooks on top of the standard ones, to make per-event customizations.

Scaling permanent clusters

Example: additional WLCG cluster for ATLAS analysis hosted on SWITCHengines



Processes: 🟩 Grid ▨ Local

Country	Site	CPUs	Load (processes: Grid+local)	Queueing
Switzerland	ATLAS BOINC	98139	7894+6083	1571+4063
	ATLAS BOINC 3	98139	5815+8163	1253+4371
	ATLAS BOINC TEST	644	0+0	0+0
	Bern ce01 (UNIBE-LHEP)	1513	1048+0	156+0
	Bern ce02 (UNIBE-LHEP)	770	624+0	159+0
	Bern ce04 (UNIBE-LHEP>	304	304+0	192+0
	Bern UBELIX T3	4472	385+2822	208+2450
	CSCS BRISI Cray XC40	1500	576+0	154+0
	Geneva (UNIGE-DPNC)	720	168+349	169+0
	Lugano PHOENIX T2 arc>	1920	1526+4040	411+14
	Lugano PHOENIX T2 arc>	2240	2065+3504	391+4
	Lugano PHOENIX T2 arc>	2048	1864+3704	407+1
TOTAL	***12 sites***	***212409***	***22269 + 28665***	***5071 + 10903***

Reference: S. Haug and G. F. Sciacca,

“ATLAS computing on Swiss Cloud SWITCHengines”, CHEP 2016

Scaling permanent clusters

Example: additional WLCG cluster for ATLAS analysis hosted on SWITCHengines

> *"A 304 virtual CPU core Slurm cluster was then started with one command on the command line. This process took about one hour. A few post-launch steps were needed before the cluster was production ready. However, a skilled system administrator can setup a 1000 core elastic Slurm cluster on the SWITCHengines within half a day.* ***As a result the cluster becomes a transient or non-critical component. In case of failure one can just start a new one, within the time it would take to get a hard disk exchanged.****"*

Reference: S. Haug and G. F. Sciacca,

"ATLAS computing on Swiss Cloud SWITCHengines", CHEP 2016

Example: SLURM cluster

Cluster definition is done in a INI-format text file.

```
[cluster/slurm]
cloud=openstack
login=ubuntu
setup=slurm
frontend_nodes=1
compute_nodes=4
ssh_to=frontend
security_group=default
image_id=...
flavor=4cpu-16ram-hpc

[setup/slurm]
frontend_groups=slurm_master
compute_groups=slurm_worker
```

```
[cloud/openstack]
provider=openstack
auth_url=http://...
username=***
password=***
project_name=***

[login/ubuntu]
image_user=ubuntu
image_user_sudo=root
image_sudo=yes
user_key_name=elasticluster
user_key_private=
    ~/.ssh/id_rsa
user_key_public=
    ~/.ssh/id_rsa.pub
```

More examples: https://github.com/gc3-uzh-ch/elasticluster/tree/master/examples

Ansible

Ansible for Software Setup (1)

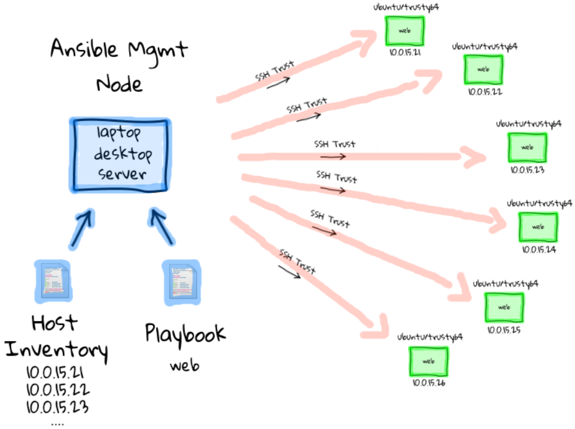


Image Copyright © 2013-2017 Sysadmin Casts - Justin Weissig
https://sysadmincasts.com/episodes/43-19-minutes-with-ansible-part-1-4

Ansible runs on a single node, and connects to all hosts under control via SSH.

No preparation is necessary on the target host, except for SSH access and Python 2.4+

Ansible for Software Setup (2)

Each *playbook* is a sequence of tasks.

All tasks are idempotent, hence all playbooks are idempotent.

Looping and conditional constructs allow (some) flexibility.

```
- name: Install required packages
  package:
    name: ’{{item}}’
    state: ’latest’
  become: yes
  with_items:
   - auctex
   - emacs
   - evince
   - git

- name: Enable hibernation
  template:
    src: files/90-hibernate.conf
    dest: /etc/polkit-1/localauthori

- name: Make `apt-file` cache
  command: |
    apt-file update
  become: yes
```

◂ Back to “What is ElastiCluster”