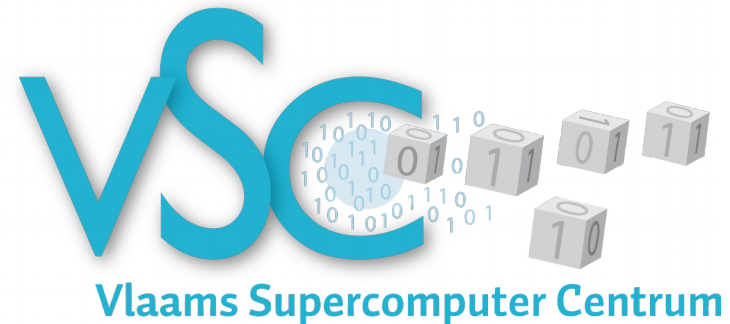




High Performance Computing
Empowering reseachers



Hanythingondemand – Hadoop clusters on HPC clusters

FOSDEM, 31 January 2016

Ewan Higgs

DICT - UGent, VSC

ewan.higgs@ugent.be

<http://www.ugent.be/hpc> - <http://www.vscentrum.be>



About Me

Ewan Higgs

Big Data Coordinator for HPC

Ghent University

- What is hanythingondemand?
- Why have we made hanythingondemand?
- Wade into hod (no deep dives)
- Use cases
- Developer things

HOD – Hanythingondemand

- <https://github.com/hpcugent/hanythingondemand>

Run a Hadoop cluster in our HPC clusters

Extensive good documentation:

- <https://hod.readthedocs.org>

`hod create` – Create a new cluster.

`hod connect` – Connect to your cluster.

`hod batch` – create a new cluster to run a script.

`hod list` – list your clusters.

Create a new cluster

```
$ hod create --label mycluster -n 4 --dist  
Hadoop-2.6.0-cdh5.4.5-native
```

Connect to the cluster

```
$ hod connect mycluster
```

Run jobs on the cluster:

```
$ yarn jar wordcount.jar WordCount  
wordcount/input wordcount/output
```

Create a new cluster and run a script

```
$ hod batch --dist Hadoop-2.6.0-cdh5.4.5-  
native -n16 --script myscript.sh
```

Elide arguments with environment variables:

```
export HOD_BATCH_DIST=Hadoop-2.6.0-cdh5.4.5-native  
hod batch -n16 --script myscript.sh
```

Create an IPython Notebook

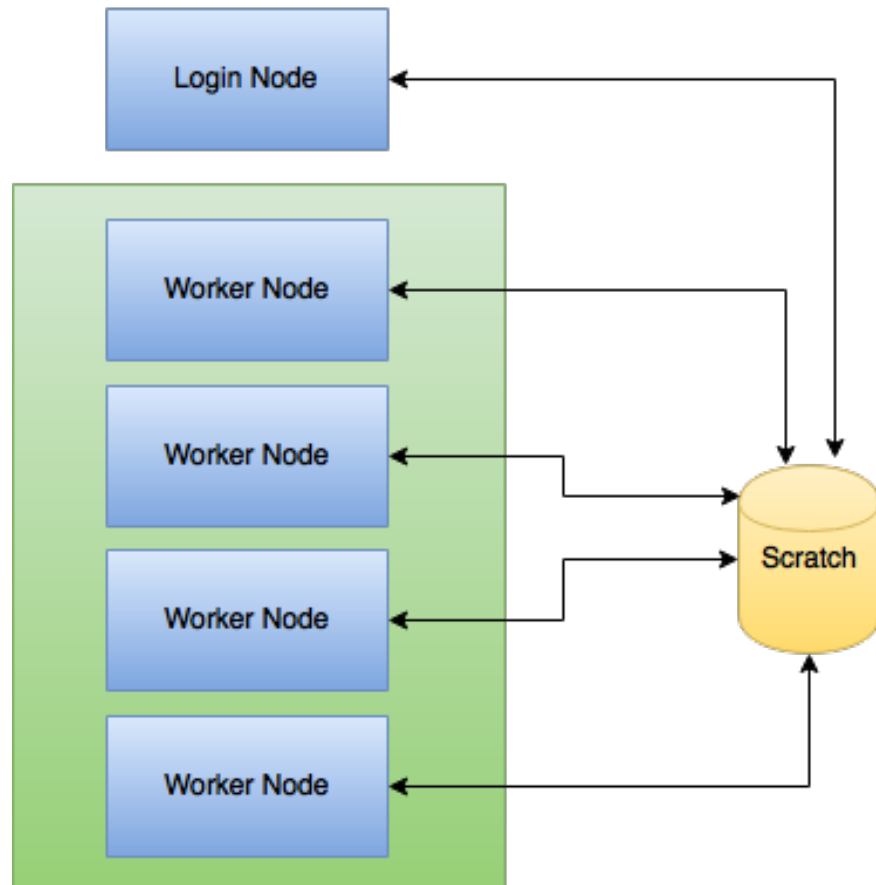
```
$ hod create --dist IPython-notebook-3.2.3  
-n2
```

To use, make an SSH tunnel to the head node
and set a proxy.

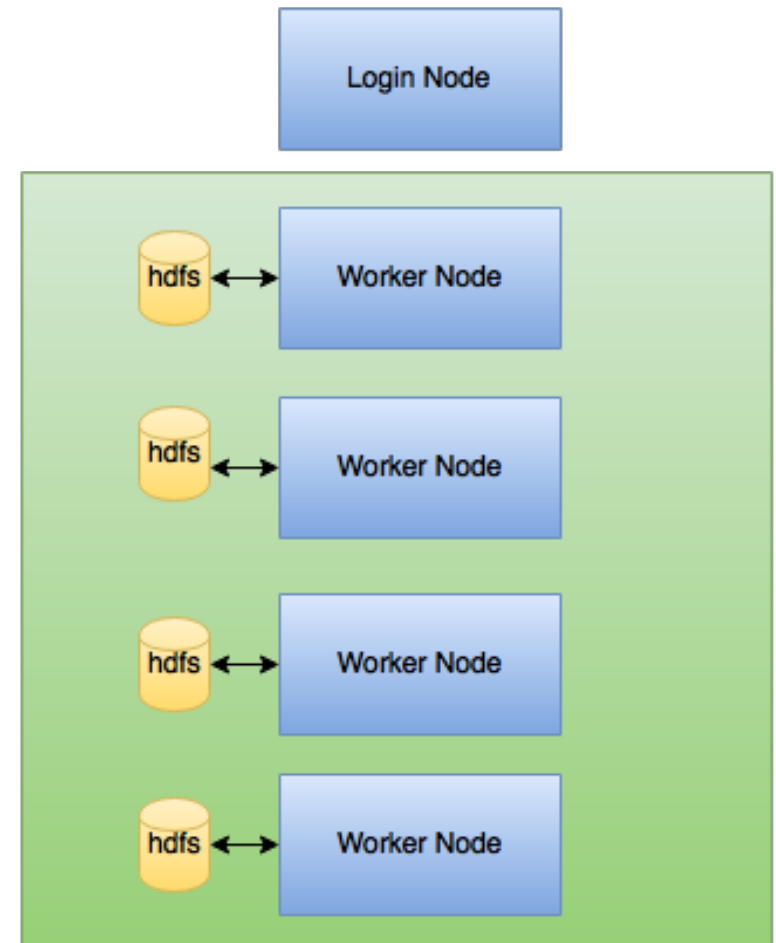
Why not just buy a big data system?

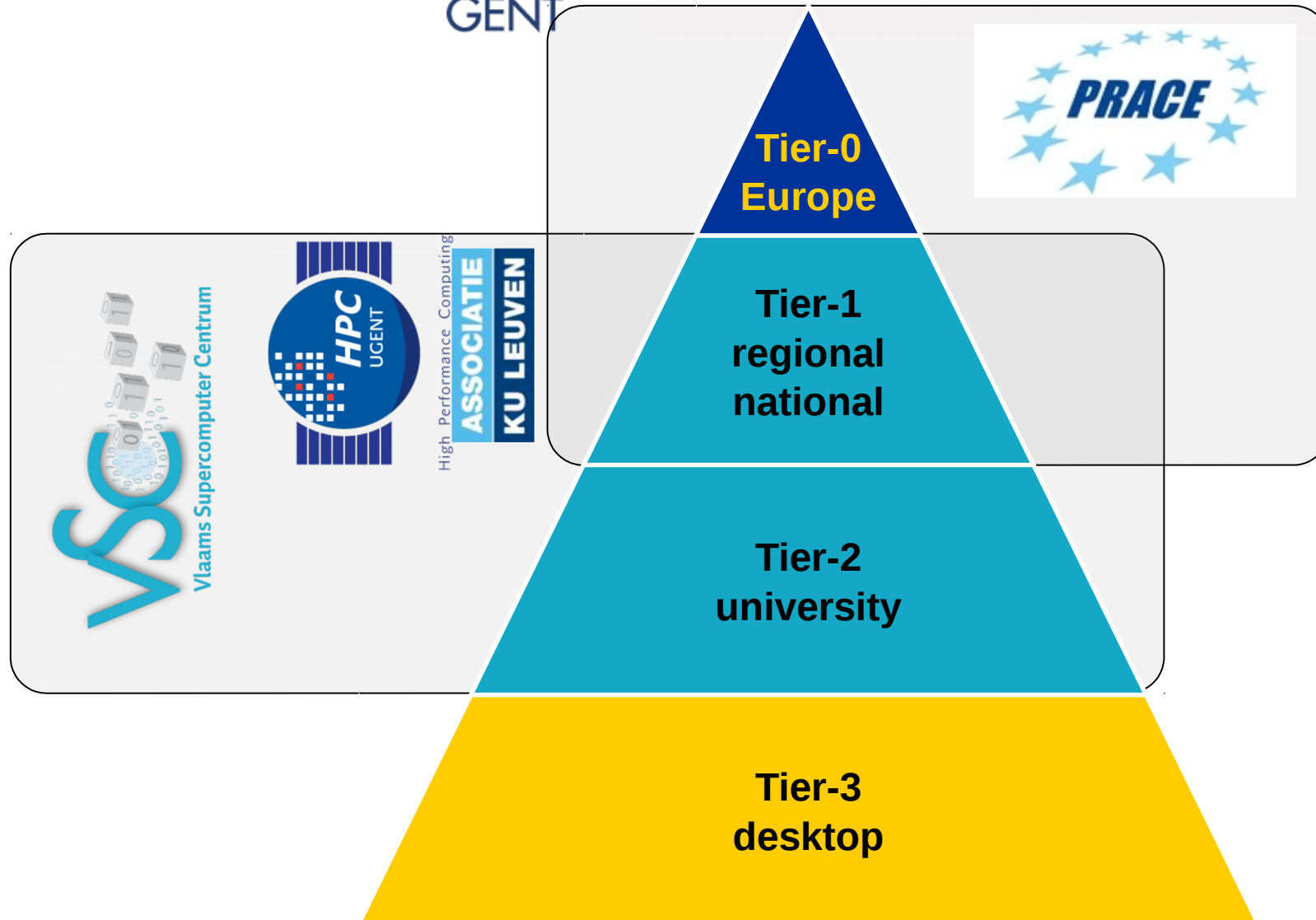
Why not just go cloud?

HPC



Big Data





Pokemon clusters

HPC-UGent:
STEVIN
infrastructure



	raichu	delcatty	phanpy	golett	swalot
# nodes	64	158	16	200	128
# cores	1024	2528	384	4800	2560
Interconnect	Ethernet	Infiniband FDR	Infiniband FDR	Infiniband FDR-10	Infiniband FDR
CPU	Intel Xeon Sandy Bridge	Intel Xeon Sandy Bridge	Intel Xeon Haswell	Intel Xeon Haswell	Intel Xeon Haswell
Clock (GHz)	2.6	2.6	2.5	2.5	2.6
Memory per node (GiB)	32	64	512	64	128
Installed	2012	2013	2015	2015	2016



	muk (Ghent)	Tier1b (Leuven)	Swalot (Tier2)
# nodes	528	580	128
# cores	8448	16240	2560
Interconnect	Infiniband FDR	Infiniband EDR	Infiniband FDR
CPU	Intel Xeon Sandy Bridge	Intel Xeon Broadwell	Intel Xeon Haswell
Clock (GHz)	2.6	3.2	2.6
Memory per node (GiB)	64	128/256	128
Installed	2013	2016	2016

Applications Run Natively **IN** Hadoop

BATCH
(MapReduce)

INTERACTIVE
(Tez)

ONLINE
(HBase)

STREAMING
(Storm, S4,...)

GRAPH
(Giraph)

IN-MEMORY
(Spark)

HPC MPI
(OpenMPI)

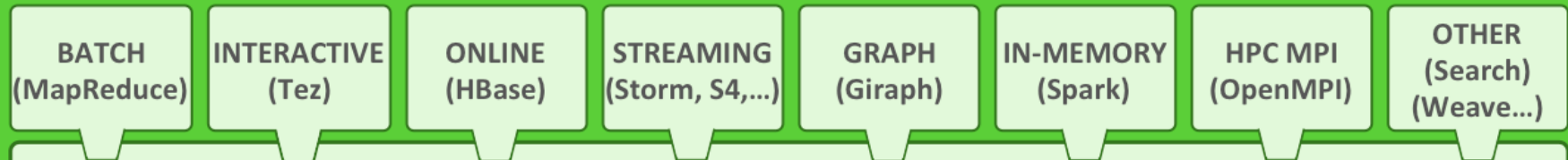
OTHER
(Search)
(Weave...)

YARN (Cluster Resource Management)

HDFS2 (Redundant, Reliable Storage)



Applications Run Natively **IN** Hadoop



YARN (Cluster Resource Management)

GPFS

PBS



Disk-Locality in Datacenter Computing Considered Irrelevant

Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, Ion Stoica
University of California, Berkeley
{*ganessa, alig, shenker, istoica*}@cs.berkeley.edu

1 Introduction

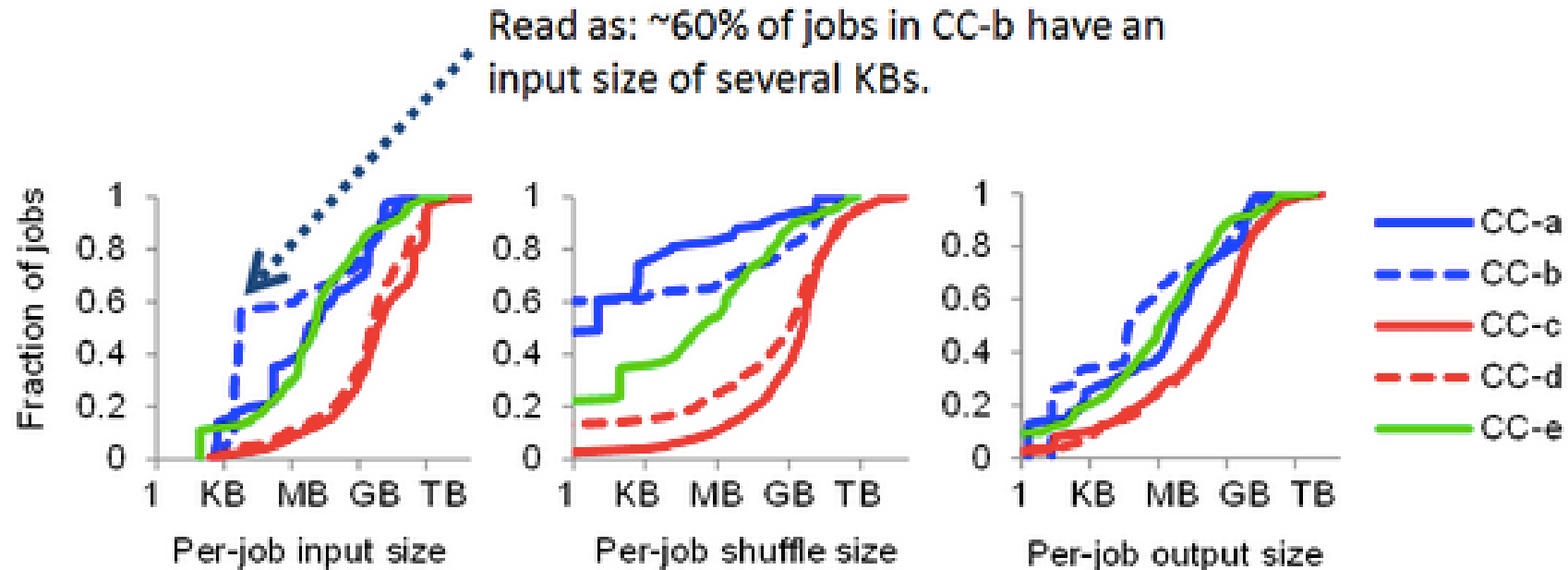
Data center computing is becoming pervasive in many organizations. Computing frameworks such as MapReduce [17], Hadoop [6] and Dryad [25], split jobs into small tasks that are run on the cluster's compute nodes. Through these frameworks, computation can be performed on large datasets in a fault-tolerant way, while hiding the complexities of the distributed nature of the cluster. For these reasons, a considerable work has been done to improve the efficiency of these frameworks.

over-subscription. Such topologies have already been adopted in several datacenters [12]. This ensures that bandwidth across racks will be equal to the bandwidth within a rack.

Another trend that strengthens our thesis is the need to save more and more data in clusters. The need for storage space outweighs affordable storage, and the gap is projected to continue to expand [2]. The ever-increasing demand for storage not only makes solid state devices (SSDs) economically infeasible to deploy as a primary storage medium [5, 9, 16], but more importantly has led

But locality of data...

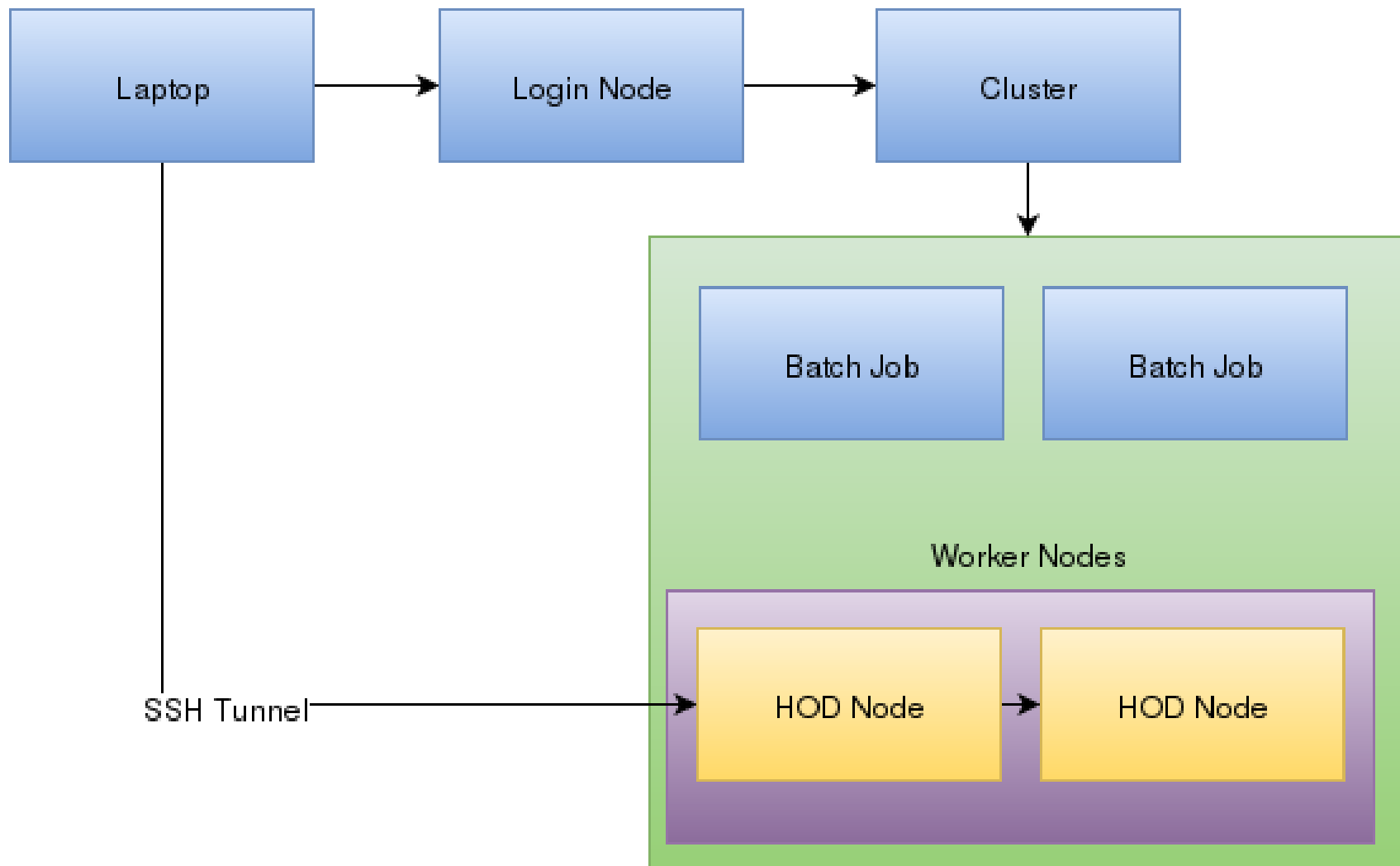
Figure 1. Per-job input, shuffle, and output data sizes



...as much as 90% most jobs fit on a single node based on a report from Cloudera's (n.b.: data from 2012).

Wade into HOD

HOD - Overview



What's a dist?

```
$ ls etc/hod/IPython-notebook-3.2.3/  
hod.conf      nodemanager.conf      screen.conf  
ipython.conf  resource manager.conf  start-notebook.sh
```



```
#!/*- cfg -*-  
# vim: ft=cfg  
[Meta]  
version=1  
  
[Config]  
# The start-notebook.sh requires IPython, matplotlib, and Spark to be loaded.  
# These are set here:  
modules=Hadoop/2.6.0-cdh5.4.5-native,Spark/1.6.0,IPython/3.2.3-intel-2015b-  
Python-2.7.10,matplotlib/1.4.3-intel-2015b-Python-2.7.10  
master_env=HADOOP_HOME,EBROOTHADOOP,JAVA_HOME,PYTHONPATH  
services=resourcemanager.conf,nodemanager.conf,ipython.conf,screen.conf  
config_writer=hod.config.writer.hadoop_xml  
# Point the workdir to a path on the parallel file system using the command  
# line named argument: --config-workdir=...  
#workdir=  
autogen=ipython_notebook  
directories=$localworkdir/dfs/name,$localworkdir/dfs/data  
~  
~
```

```
#-*- cfg -*-  
# vim: ft=cfg  
[Unit]  
Name=nodemanager  
RunsOn=all  
  
[Service]  
# note: The format is not a daemon since we wait for it to complete.  
ExecStart=$$EBR00THAD00P/sbin/yarn-daemon.sh start nodemanager  
ExecStop=$$EBR00THAD00P/sbin/yarn-daemon.sh stop nodemanager  
  
[Environment]  
YARN_NICENESS=1 /usr/bin/ionice -c2 -n0  
HADOOP_CONF_DIR=$localworkdir/conf  
YARN_LOG_DIR=$localworkdir/log  
YARN_PID_DIR=$localworkdir/pid  
~  
~
```

Auto generated configurations

Config Overrides

```
#!/*- cfg -*/  
# vim: ft=cfg  
[Meta]  
version=1  
  
[Config]  
modules=Hadoop/2.6.0-cdh5.4.5-native,Spark/1.5.0,IPython/3.2.1-intel-2015a-Python-2.7.10,matplotlib/1.4.3-intel-2015a-Python-2.7.10  
master_env=HADOOP_HOME,EBROOTHADOOP,JAVA_HOME,PYTHONPATH  
services=resourcemanager.conf,nodemanager.conf,ipython.conf,screen.conf  
config_writer=hod.config.writer.hadoop_xml  
# Point the workdir to a path on the parallel file system using the command  
# line named argument: --config-workdir=...  
#workdir=  
#autogen=ipython_notebook  
autogen=ipython_notebook  
directories=$localworkdir/dfs/name,$localworkdir/dfs/data  
  
[yarn-site.xml]  
yarn.app.mapreduce.am.resource.cpu-vcores=1  
yarn.scheduler.maximum-allocation-vcores=16  
yarn.scheduler.minimum-allocation-vcores=1  
yarn.nodemanager.resource.cpu-vcores=16  
~  
~
```

User Stories

Halvade: scalable sequence analysis with MapReduce

Dries Decap^{1,5}, Joke Reumers^{2,5}, Charlotte Herzeel^{3,5}, Pascal Costanza^{4,5}
and Jan Fostier^{1,5,*}

¹Department of Information Technology, Ghent University - iMinds, Gaston Crommenlaan 8 bus 201, 9050 Ghent, Belgium, ²Janssen Research & Development, a division of Janssen Pharmaceutica N.V., 2340 Beerse, Belgium, ³Imec, Kapeldreef 75, 3001 Leuven, Belgium, ⁴Intel Corporation Belgium and ⁵ExaScience Life Lab, Kapeldreef 75, 3001 Leuven, Belgium.

Associate Editor: Prof. Gunnar Ratsch

ABSTRACT

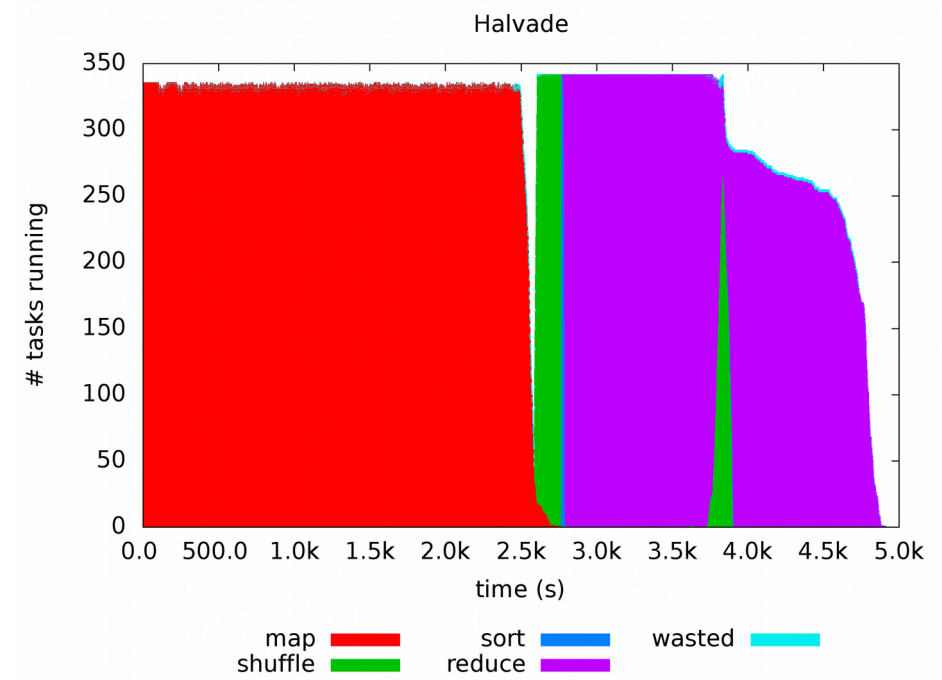
Motivation: Post-sequencing DNA analysis typically consists of read mapping followed by variant calling. Especially for whole genome sequencing, this computational step is very time-consuming, even when using multithreading on a multi-core machine.

Results: We present Halvade, a framework that enables sequencing pipelines to be executed in parallel on a multi-node and/or multi-

for an overview. Especially for whole genome sequencing, applying such tools is a computational bottleneck. To illustrate this, we consider the recently proposed Best Practices pipeline for DNA sequencing analysis (Van der Auwera *et al.*, 2013) that consists of the Burrow-Wheeler Aligner (BWA) (Li and Durbin, 2009) for the alignment step, Picard (<http://picard.sourceforge.net>) for data preparation and the Genome Analysis Toolkit (GATK) (McKenna

Table 2. Runtime as a function of the number of parallel tasks (mappers/reducers) on the Intel Big Data cluster and Amazon EMR. The time for uploading data to S3 over the internet is not included in the runtimes for Amazon EMR.

Cluster	# worker nodes	# parallel tasks	# CPU cores	runtime
Intel Big Data cluster	1	3	18	47h 59min
	4	15	90	9h 54min
	8	31	186	4h 50min
	15	59	354	2h 39min
Amazon EMR	1	4	32	38h 38min
	2	8	64	20h 19min
	4	16	128	10h 20min
	8	32	256	5h 13min
	16	64	512	2h 44min





Code

Limitations

Community

<https://github.com/hpcugent/hanythingondemand>

- Python 2.7
- GPL v2
- ~80% code coverage
- Jenkins builds

- Only PBS/Torque
- Server coding in Python2 and without twisted

Would you like to use this at your site?

Are there any tools you need?

Do you need slurm or Grid Engine?

- HOD lets HPC users use Hadoop ecosystem
- Auto generated configurations
- Being used for actual research
- HPC clusters can make good Big Data clusters
- Check it out!

Thanks

Questions?

Further contact:

ewan.higgs@ugent.be