Context-aware distributed cloud computing using CloudScheduler

R. Seuster, C.R. Leavett-Brown, K. Casteels,
M. Murray, M. Paterson, D. Ring, R. Sobie, R. Taylor,
University of Victoria

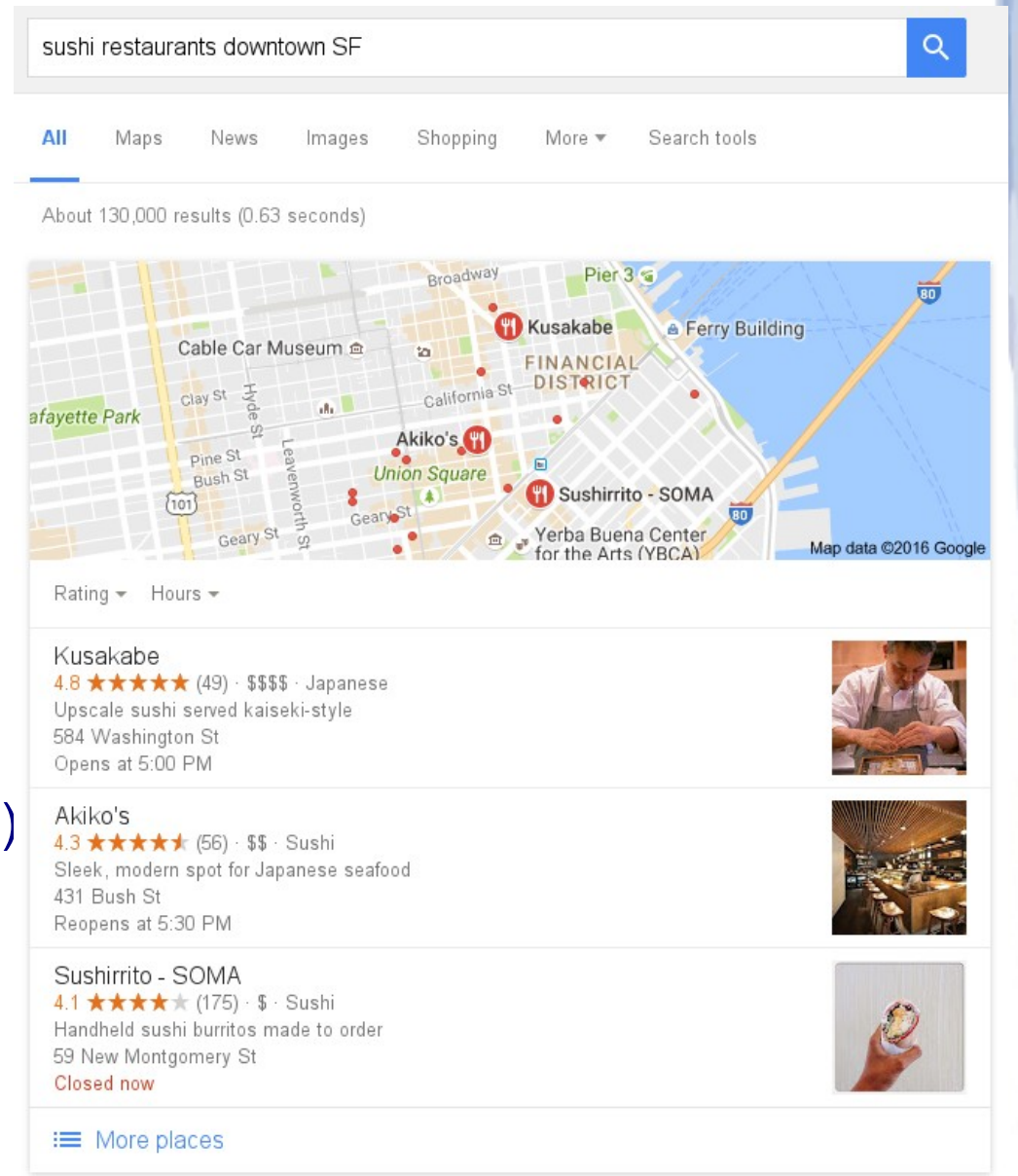
CHEP 2016
9.-15. Oct. 2016

What does context aware mean ?

- compare to search engines: personalized search results

  Search engine

- user looks for good restaurant close to his/her location
  'good Indian restaurant' or 'best sushi restaurant in downtown SF'
- search engine takes (GPS) location, user's taste and budget (from search string or from profile) into account and presents personalized prioritized list



What does context aware mean ?

- compare to search engines: personalized search results

Search engine

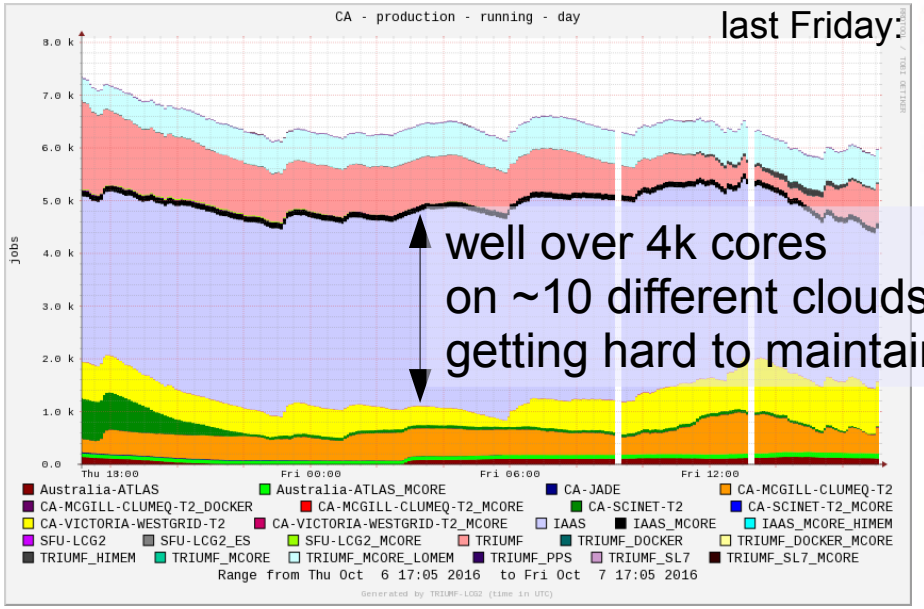
- user looks for good restaurant close to his/her location 'good Indian restaurant' or 'best sushi restaurant in downtown SF'
- search engine takes (GPS) location, user's taste and budget (from search string or from profile) into account and presents personalized prioritized list

CloudScheduler (CS) simple example:

- user submits certain type of job, e.g. high IO required and n MB of memory
- clouds X and Y provide this and have free resources, but X is closer to input data
- CS starts new VM on cloud X and job runs there

What can we gain ?

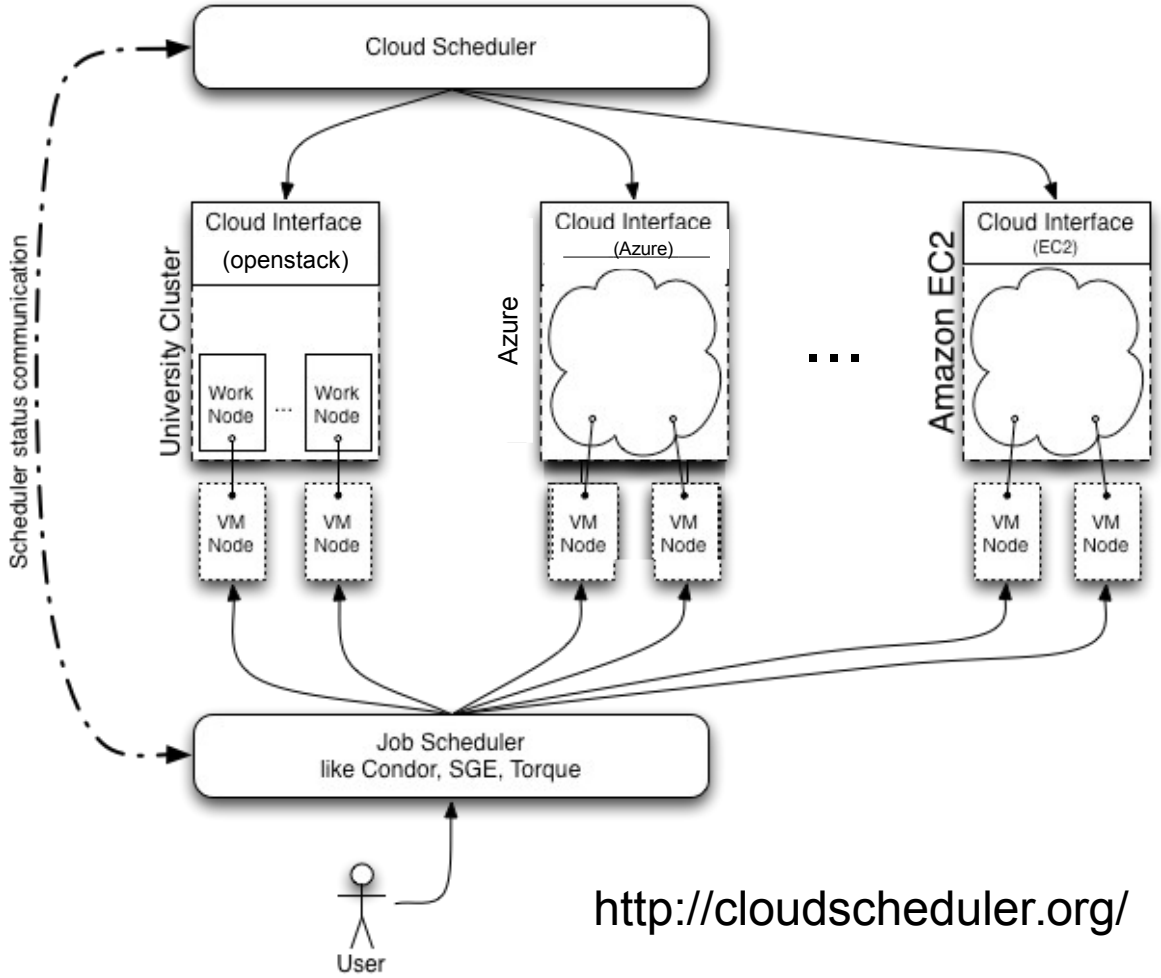
- Better utilize dynamic and opportunistic resources
  - let clouds and VMs auto-configure themselves
  - locate optimal SW caches and storage of data
- CS can more easily react to external, known problems
  - retire faulty or stressed VMs, disable unstable clouds, etc.



→ better resource utilization with less manual interaction
same team can operate more and larger clouds
easier adoption of new cloud resources worldwide

Current CloudScheduler

- CloudScheduler manages VMs in all clouds, starts and retires them as needed, use CERNVM images
- At startup VMs register with Job Scheduler, condor in our case
- Jobs are submitted by pilotfactory at TRIUMF for ATLAS and DIRAC for Belle II directly to queue
- We can run on multiple clouds: Amazon, Azure, our local development cluster, ComputeCanada clouds, several other clouds in Canada, US and Europe



http://cloudscheduler.org/

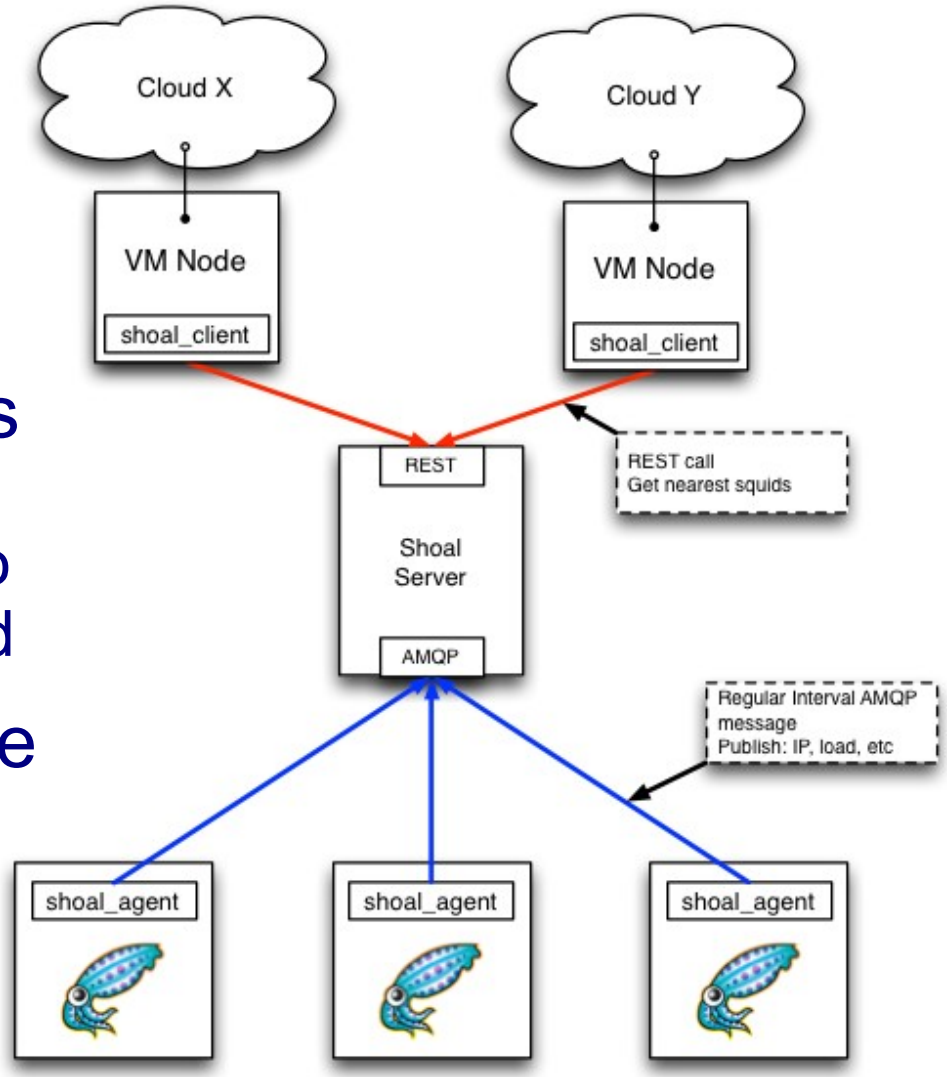
- Plan to extend to GCE and further cloud providers in Europe in near future

Auto-configure Squids: Shoal

- Shoal: 'group of squids', GeoIP based squid discovery service, basic wpad support in use since several years
  - Each cloud is configured to connect to its closest squid

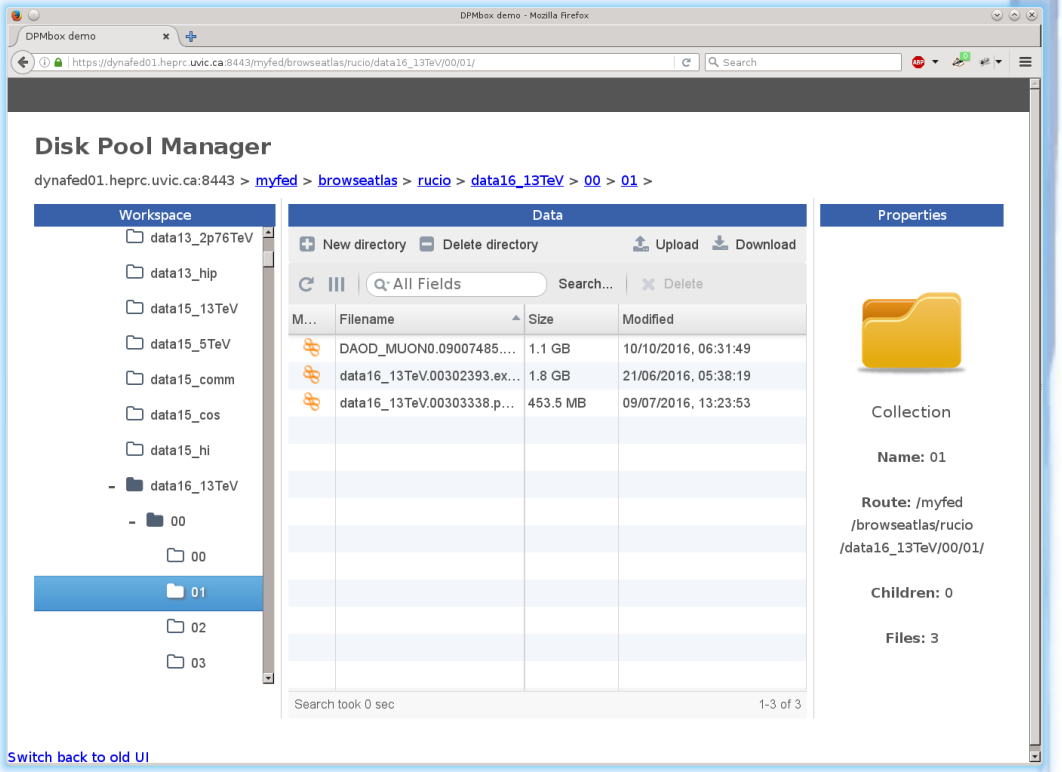
→ optimal access to code and conditions

https://github.com/hep-gc/shoal



Auto-configure Storage: DynaFed

- DynaFed: federation of storage elements using http or WebDAV protocol, to be used in near future
  - Each cloud gets data from nearest location
- Part of a WLCG demonstration project



→ optimal access also to data !
for now only for reading data, address writing later

http://lcgdm.web.cern.ch/dynafed-dynamic-federation-project

Monitoring

- Start with collecting detailed monitoring data
  - What data do we need to collect ? Start with simple things like monitor system performance and extend as necessary
    - # running and retiring jobs over time
    - # of jobs, estimate efficiencies from CPU usage
    - Process system logfiles
- Then feed back monitoring into operation
  - Learn to automatically identify 'old' problems and apply fix
  - Keep track that they don't happen too often
- Later feed back experience from monitoring and operation into auto-configuration of clouds and VMs
  - Take prior knowledge about cloud into account

New Monitoring Page

- Collecting information from different sources and display them on single page:

IAAS 04:02:47 06-Oct

amazon-east 20 | amazon-west 20 | azure 50 | beaver 6 | cc-east 92 | cc-west [illegible] | chameleon 57 | dair-ab 4 | dair-qc 6 | Jade 5

Cloud		CloudScheduler VMs: Starting	Running	Retiring	Error	Idle	Lost	HTCondor Slots: 1	2	3	4	5	6	7	8
amazon-east	atlas-worker	0	20	0	0	0		20	0	0	0	0	0	0	0
amazon-west	atlas-worker	0	20	0	0	2		18	1	0	0	0	0	0	0
azure	atlas-mcore-worker	0	46	0	0	0		39	8	0	0	0	0	0	0
beaver	atlas-worker	0	6	0	0	0		6	6	6	6	6	6	6	6
cc-east	atlas-worker	0	88	0	0	0		87	87	88	88	88	88	88	88
	atlas-mcore-worker	0	4	0	0	0		4	0	0	0	0	0	0	0
cc-west	atlas-worker	0	284	4	0	0		286	288	284	288	286	285	288	284
	atlas-mcore-worker	0	3	0	0	0		3	0	0	0	0	0	0	0

- Overall #jobs:

Jobs	HTCondor Jobs: Total	Idle	Running	Completed
All	3371	71	3283	17
1 Core	3188	21	3150	17
8 Core	133	25	108	0

Benchmarking
and health reporting

- Since recently utilize code from the HEPiX benchmarking working group here
  http://bmkwg.web.cern.ch/bmkwg/
  → poster during poster session A on Tuesday
- Runs at startup of VM and writes to database
- VM reports every 15mins of uptime, it cpu usage(s) and other information back to DB at UVic
  - Allows up to estimate VM's uptime etc. when we loose VM e.g. due to spot market pricing
- Now can calculate following information reported

Benchmarking

- Following info is calculated per cloud:

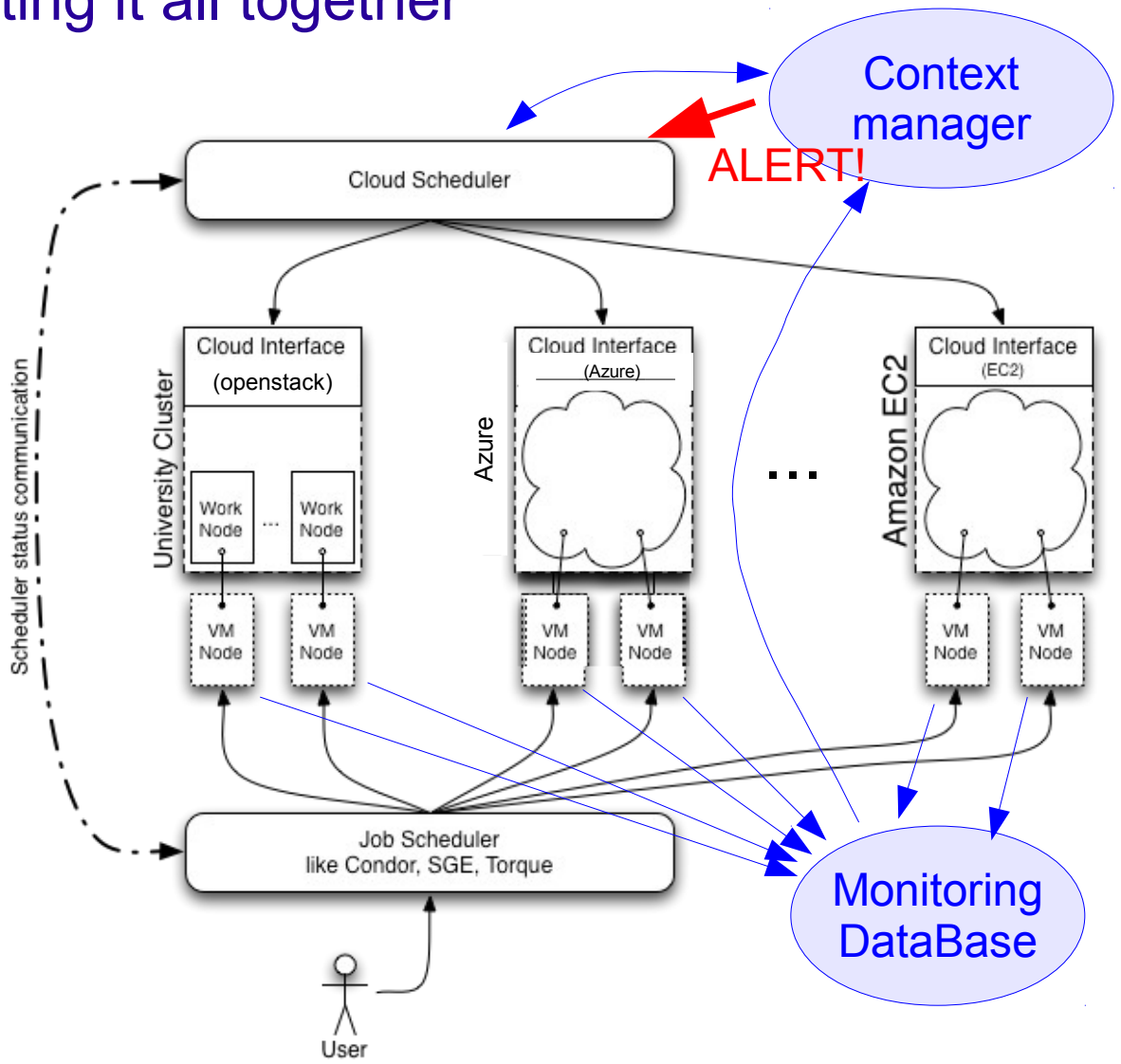
```
overlap: otype = 0 problem
Cloud          #    Bmk                 Total
                                 User    Total     Eff
cloud A       20   15.8          65.0     81.2    0.800
cloud B       53   19.9         179.7    247.1    0.727
cloud C       23   14.9          70.8     93.3    0.758
cloud D       16   21.0           4.5     11.3    0.397
cloud E       18   12.2          29.8     41.0    0.726
cloud F       18   11.9          48.0     64.0    0.750
cloud G      308    21.5         899.9   1503.1     0.599
cloud H       45    9.8          34.3    155.5    0.221
Total                           1331.9   2196.6
Monday October 10  17:00:02
```

Might not use all cpus on VM due to memory requirements of payload

A Future CloudScheduler

putting it all together

- VMs report all relevant information back into Monitoring DataBase
- ContextManager enquires DB to discover any urgent action and alerts CloudScheduler
  - CS then executes them at a 'convenient' time
- CloudScheduler consults with ContextManager about next actions, if VM needs to be started (and where!), retired, ...



Some Recent Problems

and how some could be fixed automatically

- detached 100% java process, kill -9 <pid> did not work – either delete this process (and all its threads) or retire VM after all jobs finish
- jobs which lost heartbeats – carefully monitor network to find root cause and meanwhile discard VM if too many jobs lost on it
- application failures – interface with PanDa and DIRAC and notify human if too many problems
  - no experience so far with automated detection of problems of the application and their cure

Acknowledgements

We thank for the continuing support from Amazon, Azure, Cameleon, Canarie and ComputeCanada without which none of this would have been possible

Summary & Conclusions

- Making progress on building a context-aware CloudScheduler
  - Aim is to reduce maintenance burden on operators and enable automated checking and error fixing
- Access to code, conditions and data (soon) already optimized by Shoal and DynaFed
  - Still, adding new clouds is labour intensive, thinking abut how to reduce that
- Extending monitoring as needed as we go along
  - Will probably soon start to auto-heal from problems soon