Monitoring the efficiency of user jobs

CHEP 2009, Prague

James Casey, Daniel Filipe Rocha Da Cunha Rodrigues, Ulrich Schwickerath

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

User job efficiency monitoring: why and how ?

limited available computing resources:

- sites have an interest in high box usage
- experiments get only limited resources. Efficient use of these is in their interest

Increase machine usage

- additional job slots per machine
- backfilling of jobs to idle machines
  - respect job requirements
  - respect fair share
  - don't overload boxes
- mix CPU and I/O bound jobs
  - requires knowledge about jobs
  - respect memory and disk space requiremets

Improve job efficiency

- **monitor it !**
- identify where time is spent
  - allows to improve code
- categorize jobs
  - earmark them
- distinguish and separate I/O and CPU bound jobs

poster session

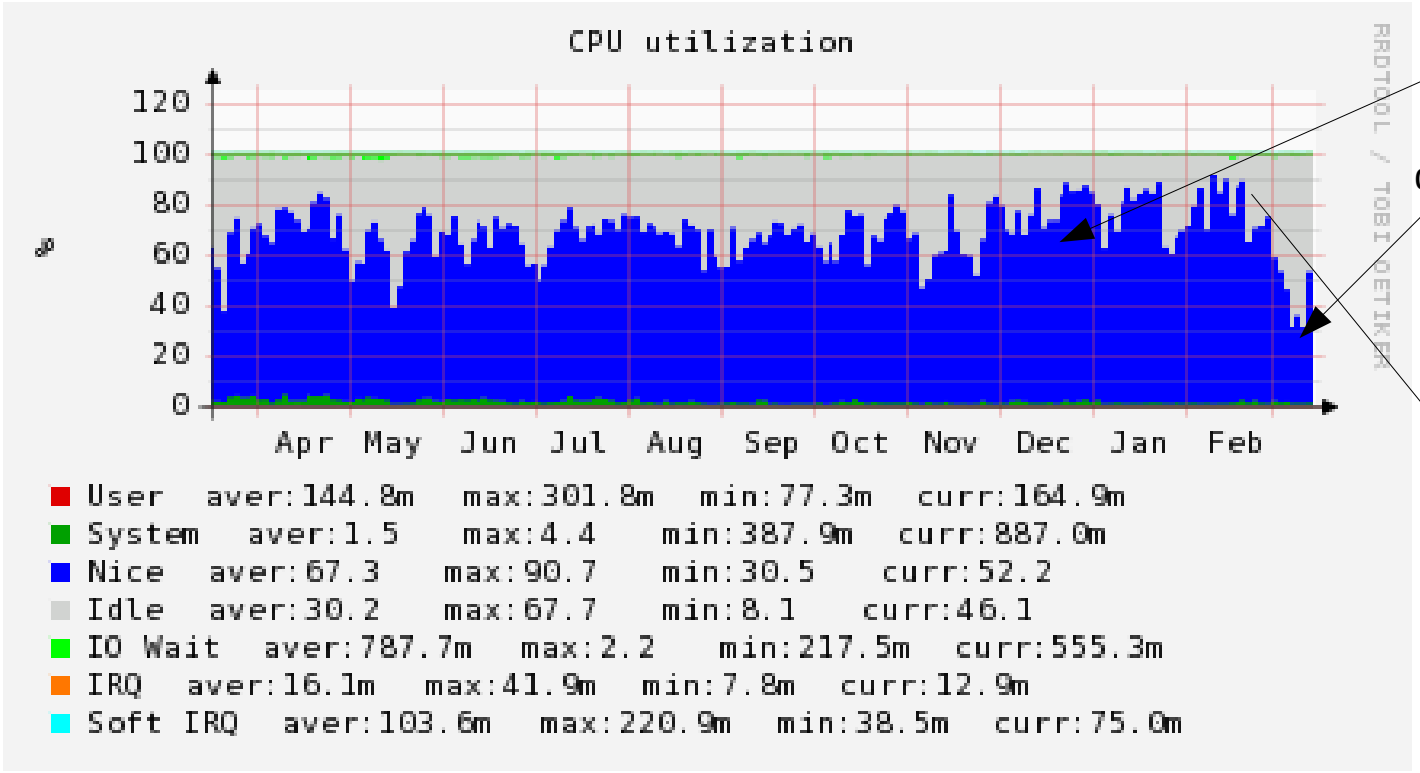
http://indico.cern.ch/contributionDisplay.py?contribId=171&confId=35523

This presentation!

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

CPU usage: situation at CERN

CERN public batch usage (last year)



start job back-filling (poster session)

draining for intervention

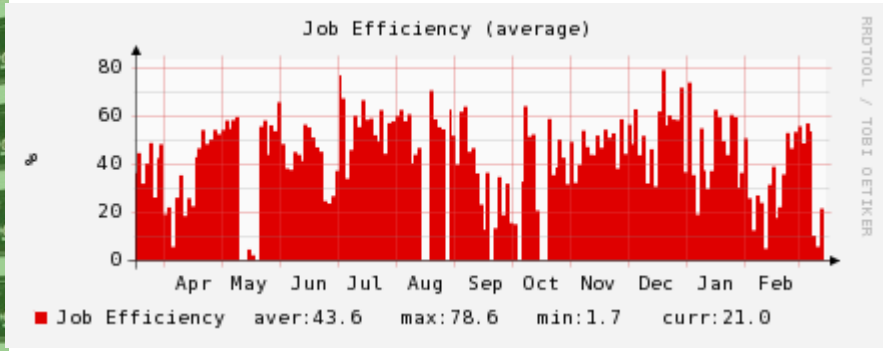
with some tricks up to 97% usage

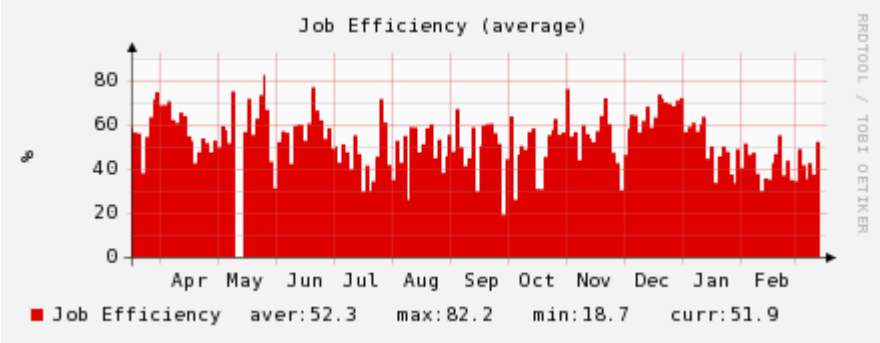
average usage of ~70% usage
Mixture of all kind of jobs: I/O and CPU bound jobs

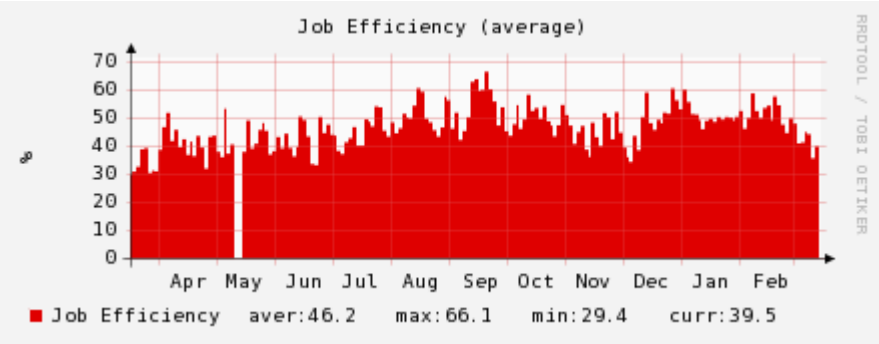
CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

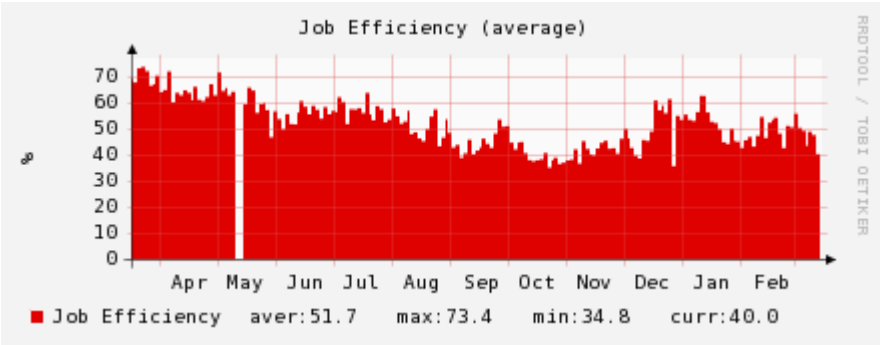
Introduction

User job efficiency: situation at CERN (grid + local, last year)









Where is the time spend ?
Waiting for Data ?
Inefficient code ?
Different classes of jobs ?

Need more details about the jobs!

Need instrumentation of jobs

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

MSG – Messaging System for the Grid

Implements a MOM (Messaging Oriented Middleware) approach

- Information is sent as Messages (Header + Body)
- Messages are sent to / consumed from a logical destination
  - Queue (Point to Point)
  - Topics (Multicast)
- Suitable for Asynchronous environments
  - All components are independent!
  - A message is guaranteed to be at a certain location at a given time
    - Meaning a consumer may be down, but recover later
    - Message is not lost, and publisher doesn't worry about sending again.

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

MSG – Messaging System for the Grid

MSG consists of

- Broker
  - Apache ActiveMQ open source implementation
    - Supports many different protocols (STOMP, Openwire(JMS), HTTP)
    - Reliable Delivery (persistence of messages)
    - High Availability (failover brokers)
    - Network of Brokers
- Clients
  - msg-consume2oracle, msg-simple-publish
    - Lightweight, minimal dependencies scripts.
  - Java, C++, perl, python bindings
- Messages Specifications
  - Key-value pairs
  - Topic
  - Designed for allowing batching
  - Based on Grid Probe Specification

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

MSG for Job Monitoring

MOM approach is suitable for Job Monitoring!

Remarks:

- Messages can be sent asynchronously from anywhere (WN's, RB's, WMS, JobWrapper, …)
- Into a logical destination instead of physical locations!
- Meaning new consumers may be added on the fly
- Using msg-publish-simple through msg-tag
- Sending information from each job
- And batched from a batch system
- MSG enables a flexible, reliable way of sending information
- Simple instrumentation using msg-tag
- Flexible : stomp (recommended) or http (if behind a firewall!)
- Reliable: messages persisted coupled with high availability

More information:

http://indico.cern.ch/contributionDisplay.py?contribId=136&sessionId=9&confId=40435

Job instrumentation with MSG

Status of the project

Automatically uploaded data for all jobs at CERN:

- job start record (just before user payload is started)
- job end record (directly after the user payload returns) (*)
- full view of the job from the batch system perspective (LSOF record)

Specifically: LSF records

- upload once per day for all finished jobs in the last 24h
- uses LSF accounting records as input

```
context:LSOF
state:JOBEND
localjobid:25666848
ownerdn:schwicke
wnhostname:lxb8186
```

→ job identification, matching with local/experiment monitoring

```
cpufactor:2.500000
cpuUsage:1332.000000
walltime:1458
memoryUsage:885
```

→ job resource usage and efficiency

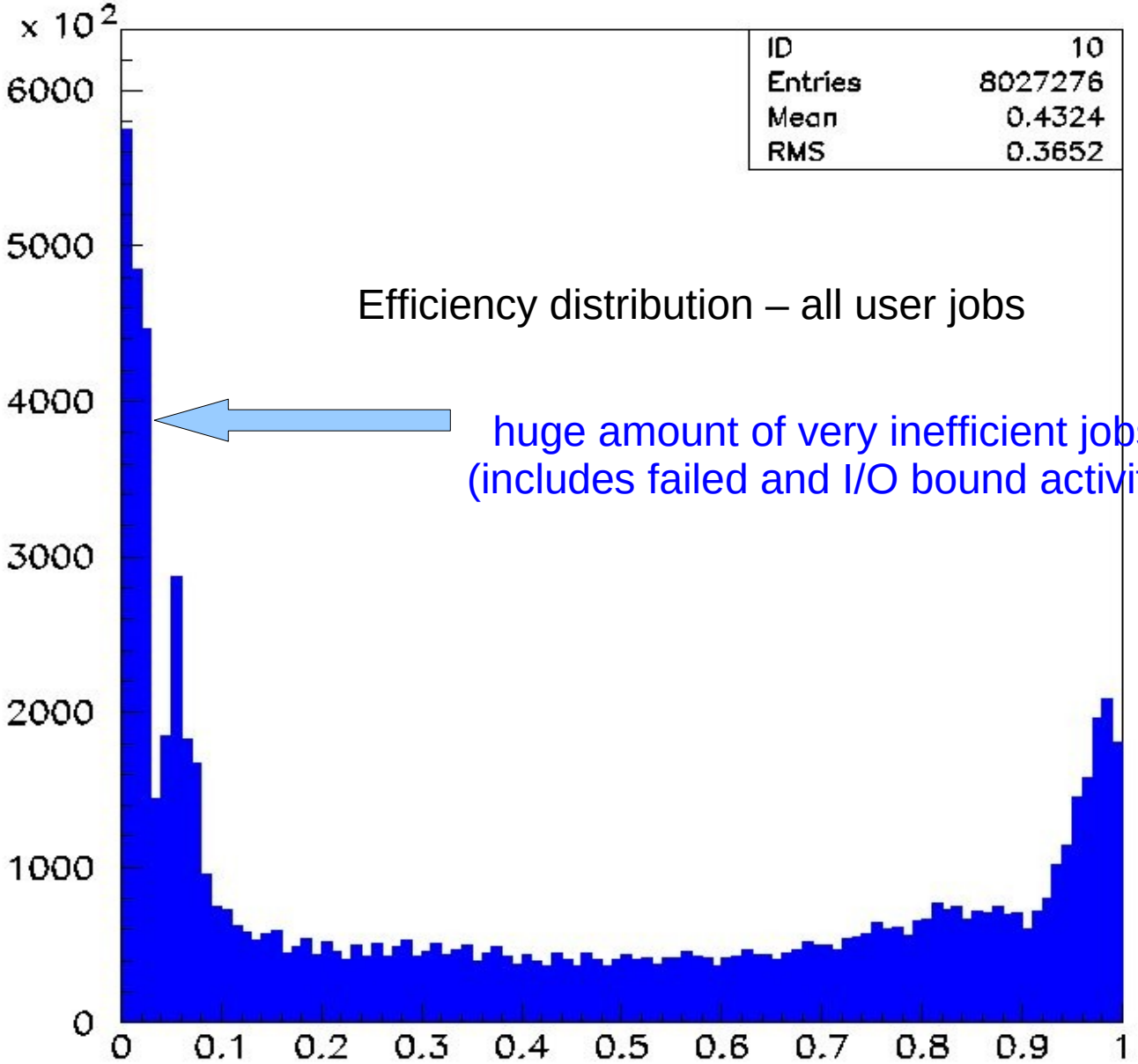
```
exitcode:0
submittime:1236779980
finishtime:1236819184
```

→ total waiting time and success rate

(*) can be missing if the job crashed or got terminated

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

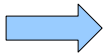
Example: data from LSF messages only



CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Instrumenting experiments frameworks

How to send a message to the system ?



```
/usr/bin/msg-tag --context=<string>
                 [--state=<string>]
                 --cputime=<cputime in seconds>
```

- check if this tool exists, and if so call it. That's all!
- works if your job ends up at CERN
- portable; could be used at other sites as well
- supports PBS/Torque and LSF

Optional arguments:

- --walltime
- --vojobid=<string> (useful for GRID jobs)
- --vosite=<string> (useful for GRID jobs)
- --cpufactor=<decimal number> (filled automatically for CERN)

Caveats:

- you need to keep track of the CPU time yourself. That can be tricky ...
- it currently only exists at CERN

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Example instrumentation

Example: instrumenting DELPHI (LEP) physics code

- 2 classes of jobs run by the same user
- distinguished by the 'context' field in the messages
- context and state strings
  - currently free text fields
  - find common tags between experiments for similar activity ?

Analysis (4-Jet Higgs search)

```
context: *hqq*
state  : start
         prepare
         ypatchy
         compile
         link
         run
         store
         end
```

MonteCarlo production

```
context: 'delmc'
state:    run
          delsim
          delana
          sdst
          store
          finish
```

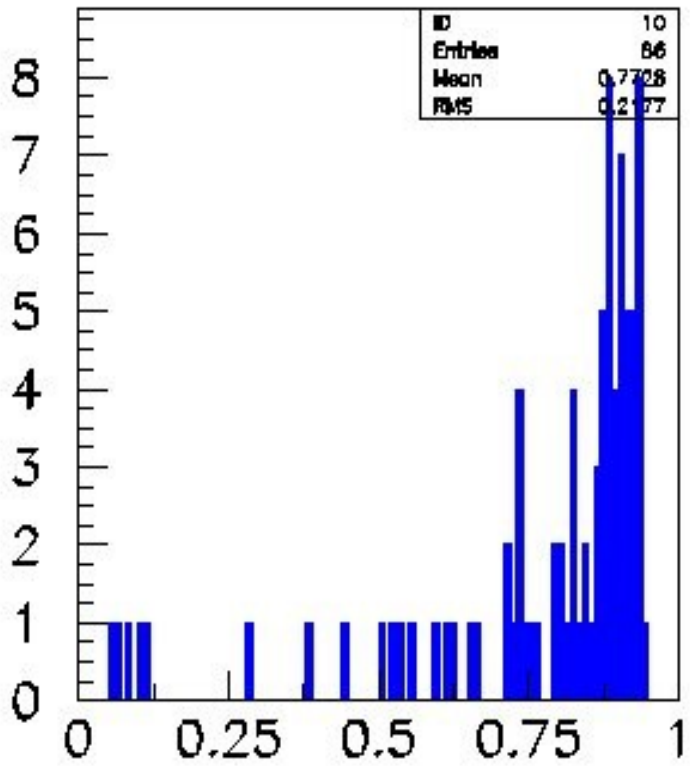
Remarks:
- start of generation step (run)
- start of detector simulation step (delsim)
- start of reconstruction step (delana)
- start of post processing (sdst)

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

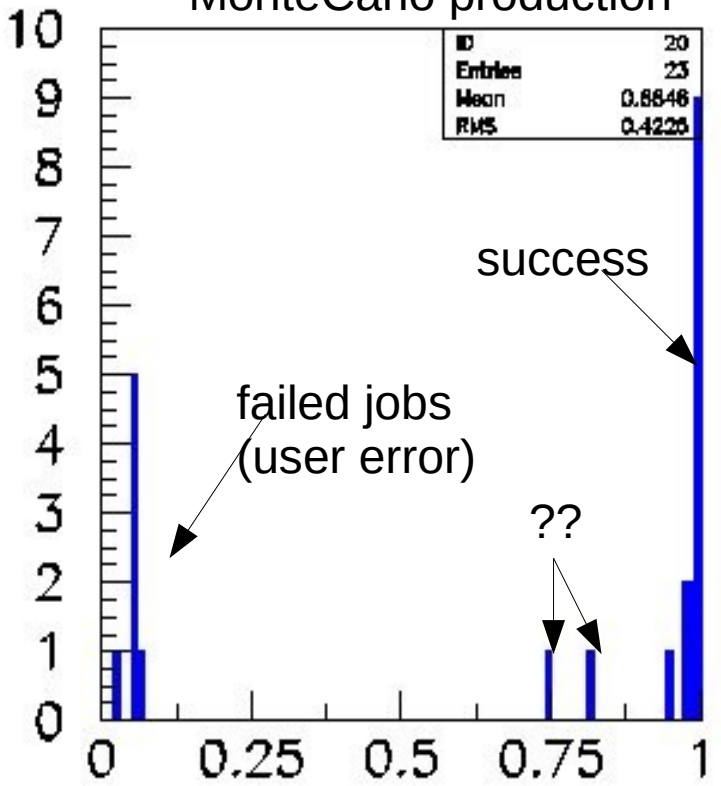
Example instrumentation

Job efficiency seen by LSF

Higgs analysis



MonteCarlo production

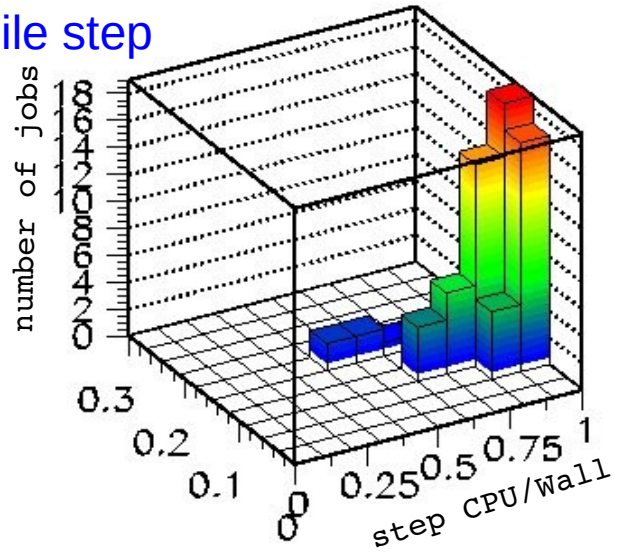


CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

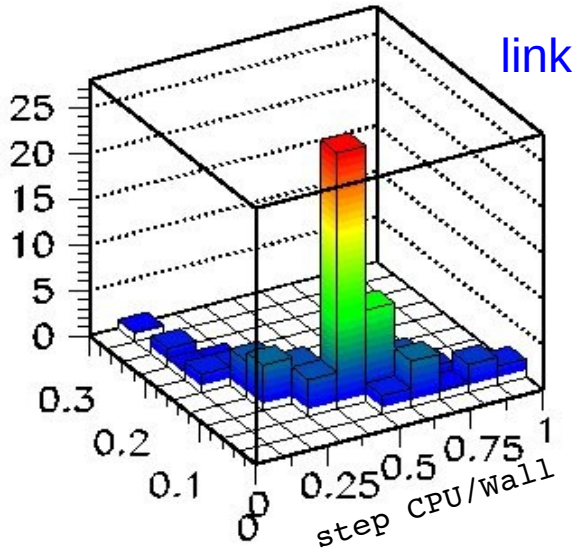
Example instrumentation

Analysis of User provided data : example of Higgs analysis

compile step

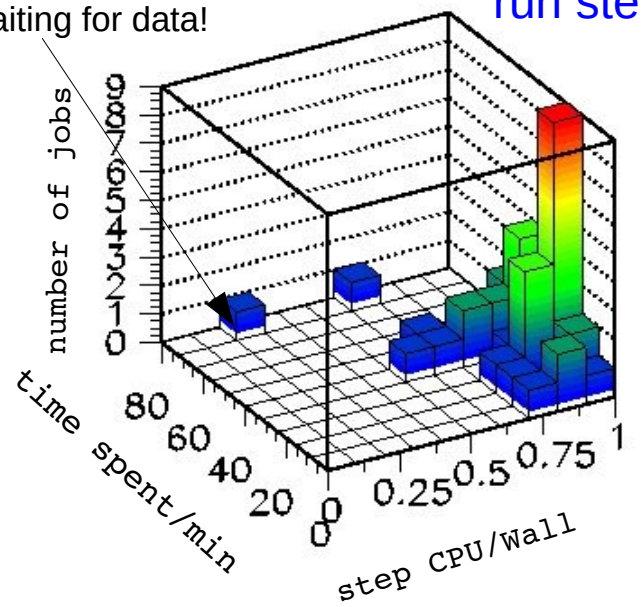


link step

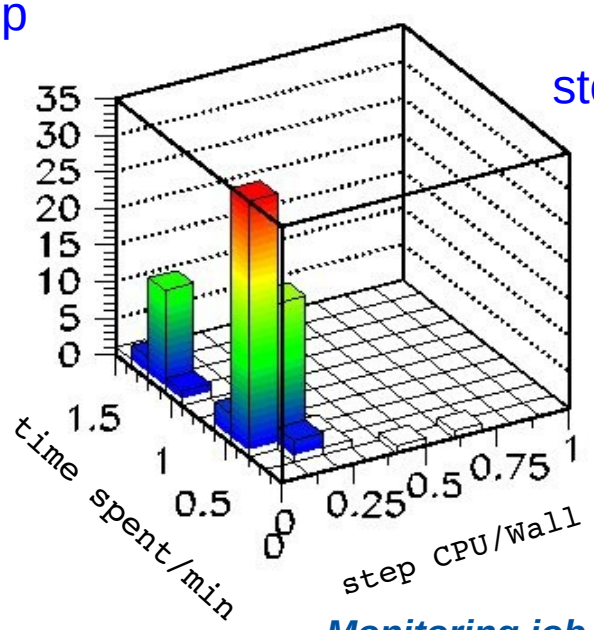


run step

waiting for data!



store step, I/O



CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Improving job efficiency ...

... **can ONLY be done by the experiments/users**

IT can help to monitor what people are doing
... if the jobs are instrumented.

Proposal:

- Check and consume the IT provided data
- Earmark your jobs
  - use the MSG framework
  - send a message at the beginning of the job to categorize it
  - use the context field for this purpose
  - match with IT provided data
- Context fields should be synchronized between experiments

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it

Conclusions

MSG is a powerful tool which can be used to check jobs for inefficiencies and thus help to improve job efficiency

It's currently in place at CERN already, and can be used by jobs running at CERN.

It's made to be portable, so other sides can support it as well.

An interface to existing monitoring systems (eg. dashboard) will be nice !

See also:
http://indico.cern.ch/materialDisplay.py?contribId=136&sessionId=9&materialId=slides&confId=40435

CERN IT Department
CH-1211 Genève 23
Switzerland
www.cern.ch/it