COOL

verification

Reconstruction stress testing

It works well!
Doesn’t it?

David Front, Weizmann Institute

Contents

- COOL verification
- Atlas COOL reconstruction requirements
- Compare Atlas requirements to LHCB’s
- Scalability testing
- Testing technique
- Results
- Plans
- Summary

COOL verification

- Strong unit testing is done by developers
- Past findings
  - Learn requirements:
    - Payload queries vs. R/O SQL access
    - Define workload
  - COOL issues
    - COOL support of bulk insertion is important
    - Memory leaks at RAL and COOL
    - Computing IOV tables statistics is essential

Atlas COOL reconstruction workload

Expected arrival rate of Atlas COOL data:

- 'IOV average duration'
  (frequency of updating DCS data)
  is **5**-10 minutes
- Avg. data in each IOV duration is **10**-100 MB
  - # COOL folders: **1000**-3000
  - # channelIds per folder: **100**-300
  - # bytes per IOV: **100**
    Payload: usually some (~20) numbers

Atlas COOL reconstruction workload

Reconstruction processes:

- 1000 Reconstruction processes may run in parallel, on 100 machines (up to 10 processes/machine)
- A process starts every 5 seconds
- Reconstruction is done for data that is an order of two days old
- Reconstruction jobs are spawned in sequential time, preserving the order of data taking
- At start, each process reads all COOL data of its IOV,
- Afterwards, processes do reconstruction for up to 5000 sec (1000 processes * 5 sec)
- 60-120 processes read the same COOL data (Since a process starts every 5 seconds, and IOV duration is 5-10 minutes)

Atlas COOL reconstruction workload: Goal

Each reconstruction process uses an average of no more than 5 seconds of the DB server ‘exclusive’ time

This means that on the long run, if in average *X* clients are reading from COOL simultaneously, read duration (of 1 IOV of all channelIds of all folders) should not be longer than: *5 seconds*X*

Note: *X* is much smaller than the number of reconstruction processes (1000), because most of the time, each process does reconstruction work rather than reading COOL data

Compare Atlas reconstruction to LHCB

	Atlas	LHCB	Ratio
data per IOV [MB]	10-100	2-40	2.5-5
Jobs /day	17280 (Once per 5 secs)	1400	12
distribution	All at CERN	7 sites	7
Atlas compared to LHCB reconstruction, loads its COOL DB server (~4*12*7=336) a few orders of magnitude more			

Scalability testing (Luca)

Explore response of the system to the variation of a tested parameter, such as #users

Plot response time seen by one user for a reference query, as a function of the #users simultaneously accessing the server

Present the variation of server host and DB resources (CPU, memory, I/O, network) as a function of #users (on the server)

Present the variation of client node response as a function of #users on each client

Testing technique

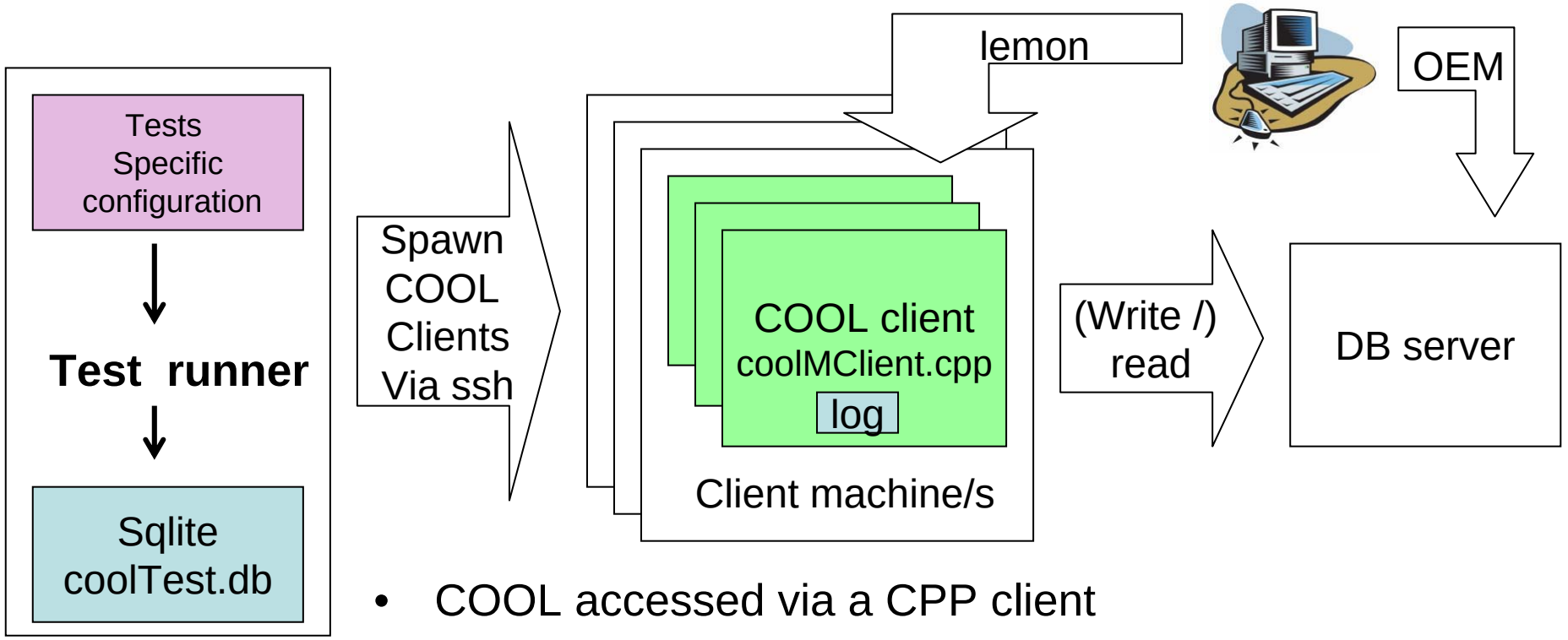
- Statistics was collected for each IOV table before testing
- COOL repository:
  - 100 Folders
    - 10th'of expected minimum
  - Each with 100 ChannelIds
    - minimum
  - Each with 1000 IOVs
    - equivalent to half a week
  - COOL payload: 10 floats
- DB server: non RAC Oracle cooldev (400MB cache)
- COOL clients: lxb and lxplus nodes

Testing compromises

Simulation was only partial:

- 100 Folders: Only 10th'of expected minimum
- 1000 IOVs per folder: equivalent to half a week
- DB server: non RAC Oracle
- Small number of client machines < 10
- DB Server IO behavior was not fully simulated
  - Should be: 60 clients read the same data, then another 60 clients read next IOV and so on
  - Was: Same IOV repeatedly read

COOL verification infrastructure



- COOL accessed via a CPP client
  Client does COOL reads(writes) according to configuration and logs cool and host performance
- Python scripts
  - Per-tests-suite python configuration
  - Run multiple clients
  - Create sqlite database per tests suite

Per tests suite configuration

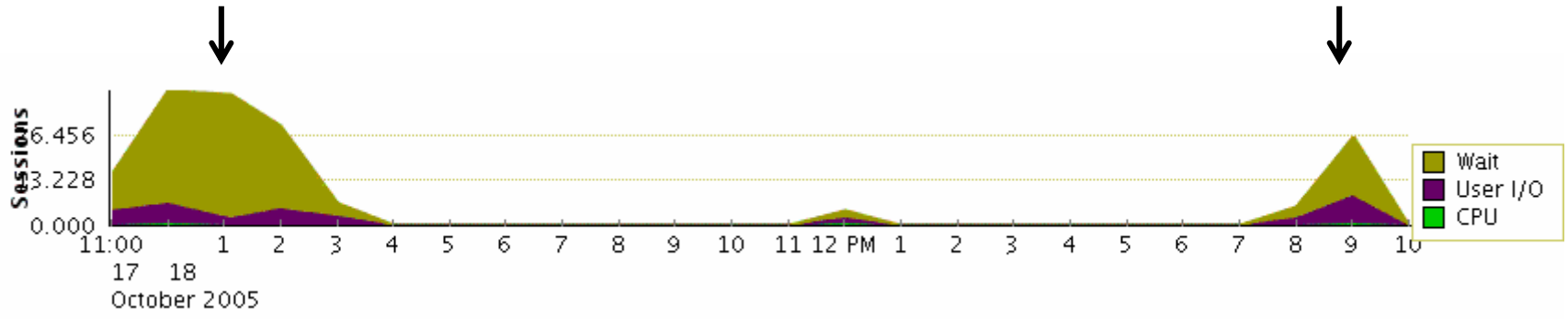
```
optsDict = {
"CoolVersion" :  "COOL_1_2_4"
"hosts"       : ['lxb0682', 'lxplus056'],
"command"     : 'read',              # read or write
"clientOpts"  : {                    # Options that go to clients
   "numIOVs"                 : 1,          # 1 IOV for reconstruction
   "totNumFolders"           : 100,
   "channelsPerFolder"       : 100,
   "firstSince"              : -1,         # Since of last IOV
   "numClients"              : [10,20,30], # A test for each value
   "numFieldsFlt"            : 10,         # Defines the payload
   "testDuration"            : 1,
   "dbKind"                  : "oracle",   # DB  options….
    …
}}
```

Sqlite test result database

```
CREATE TABLE coolTest (
    time TEXT,
    testValue TEXT,
    runNum INTEGER,
    isDefault INTEGER,
    duration INTEGER,
    ExpectedWoR REAL,
    numPerSecWoR INTEGER,
    expectedMBpMinWoR REAL,
    clientsCPUperc INTEGER,
    passed TEXT,
    testName TEXT,
    client INTEGER,
    report INTEGER,
    elapsed INTEGER,
    ActualWoR REAL,
    percWoR INTEGER,
    MBpMinWoR REAL,
    load5 REAL,
    clientMemPerc INTEGER,
    hostname TEXT);
```

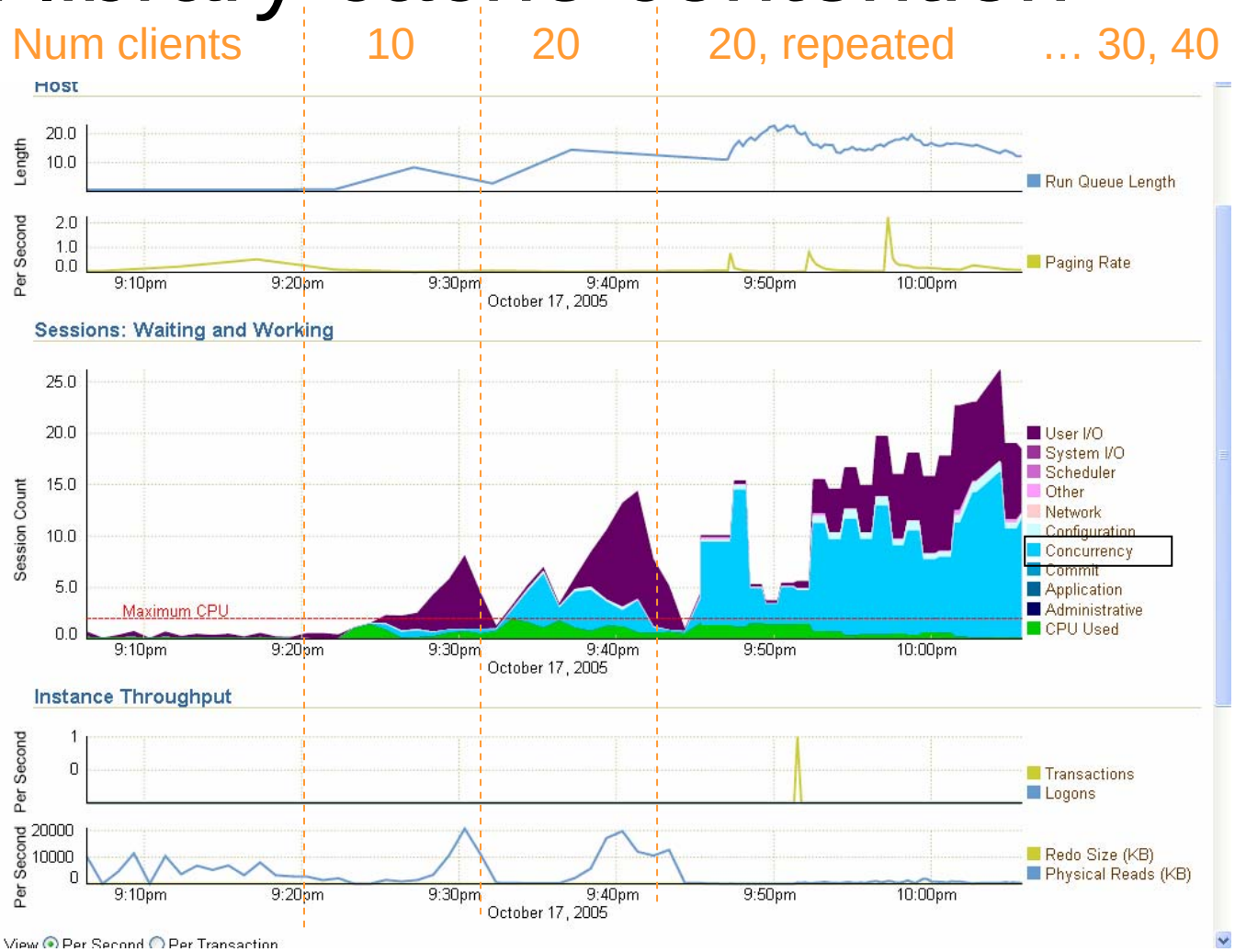
Server is occupied waiting

- Testing with ‘usual’ parameters:
  Once every 5 seconds a job is spawned that reads 100 records (one per channelId) from 100 COOL tables
- The number of clients varies between 1 and 50
- Cache contention appears with 20 clients
- Tests done for IOV number 1000, and repeated for IOV number 1, in order to avoid the effect of: #2223Performance of IOV retrieval is not uniform in [-inf, +inf]:
  Each client reads the same data IOV number:
  IOV number 1000 or IOV number 1



‘concurrency’ is ‘latch library cache contention’

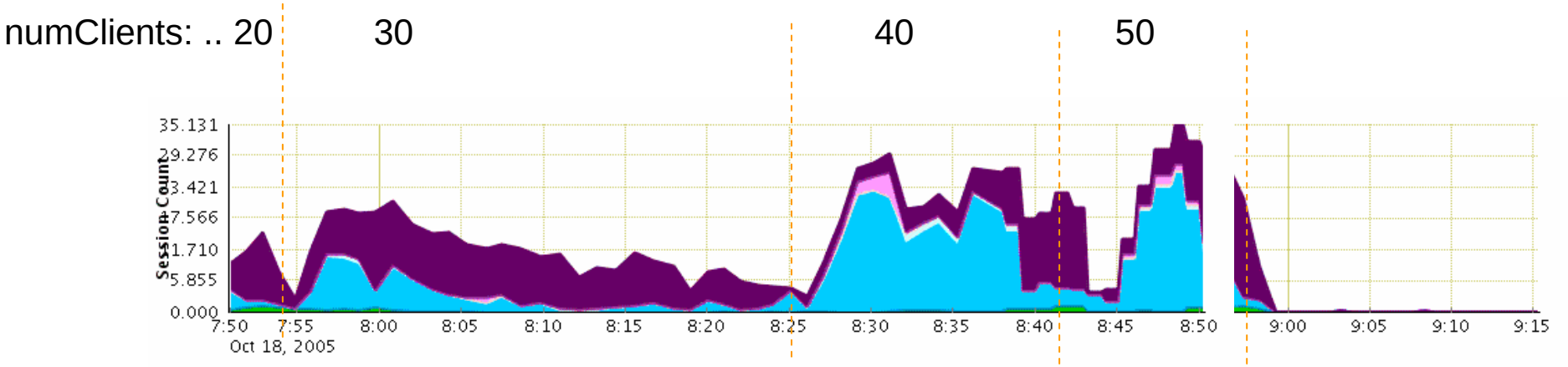
IOV number = 1000
~ half a week of work



Repeating pattern for IOV 1000

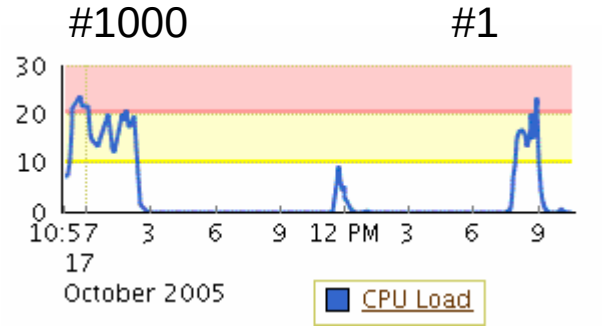
- When spawning a number of tasks, a period of ‘latch library cache’ with no IO
- Is followed by a period where the contention goes lower and IO happens
- The pattern becomes more prominent, at each testing iteration

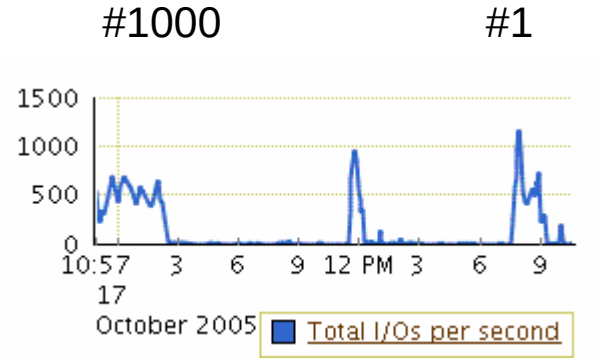
1-50 clients, read from IOV#1

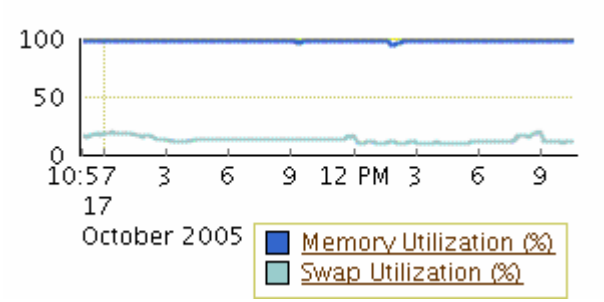


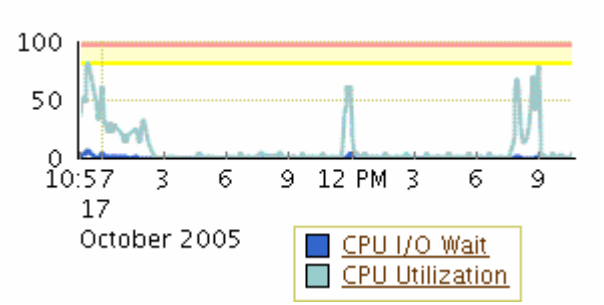
Legend:
Green: CPU, blue: concurrency, purple: IO, pink: other

Host monitoring









For IOV number 1000 and for number 1:

- Server: Host memory is ok, I/O goes high, and hence CPU load
- Clients: No high stress observed

For IOV number 1000 the stress is longer than for number 1

Elapsed time and client monitoring

IOV # 1000

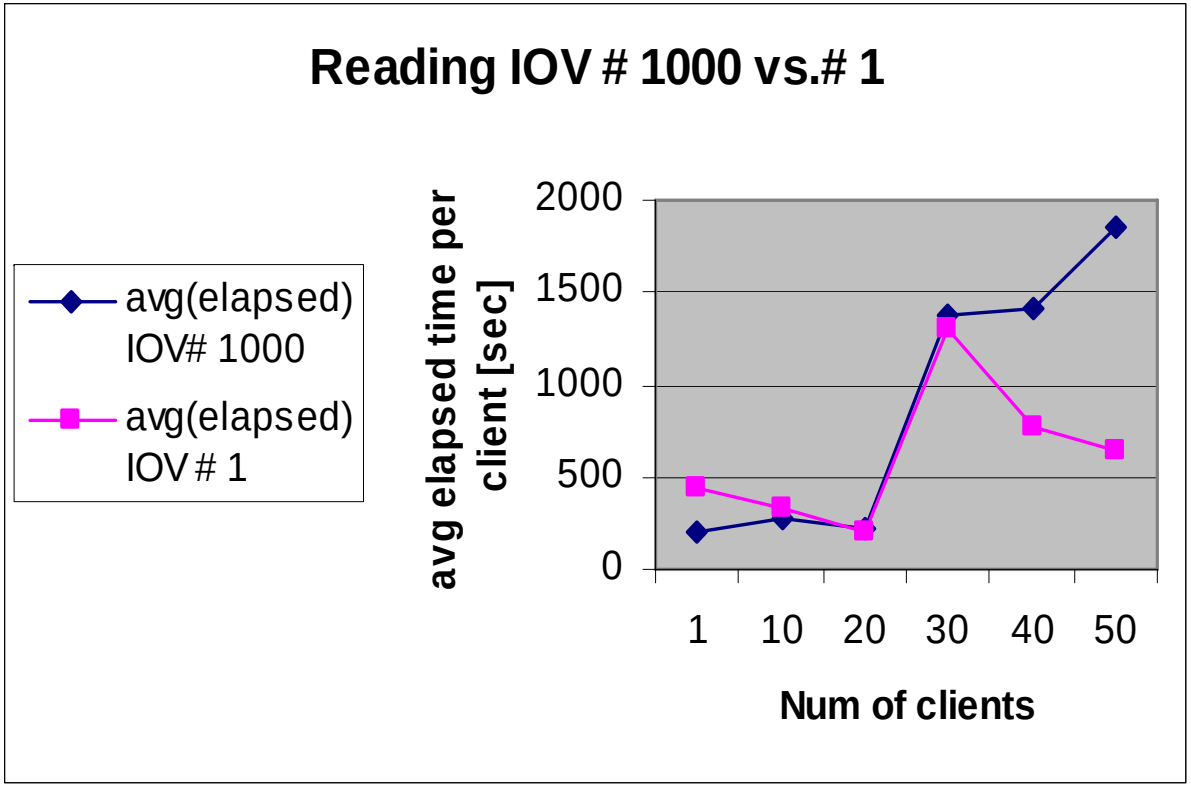
time 2005-10-17	testName	testValue	avg(elapsed)	min(elapsed)	max(elapsed)	avg(load5)	clientsCPUperc	clientMemPerc
21:01	numClients	1	201	201	201	3	11	0.7
21:30	numClients	10	270	116	363	1	8	0.7
21:41	numClients	20	224	150	287	0	7.3	0.7
22:03	numClients	30	1370	719	2496	0	3.7	0.7
23:46	numClients	40	1410	1032	1488	0	1.4	0.7
00:22	numClients	50	1858	1281	2375	0	2.3	0.7

IOV# 1

Time 2005-10-18	testName	testValue	avg(elapsed)	avg('serverTime') = avg(elapsed) /numClients	min (elapsed)	max (elapsed)	avg (load5)	clientsCPUperc	clientMemPerc
19:34	numClients	1	448	448	448	448	0	5.8	2.9
19:42	numClients	10	333	33	309	362	0	7.2	2.9
19:48	numClients	20	198	10	159	241	0	10.9	3
19:53	numClients	30	1298	43	348	1758	0	7.2	3
20:24	numClients	40	766	19	641	1038	0	2.5	3
20:42 20:55	numClients	50	639	13	532	785	0	3.3	3

IOV # 1000 ~/dfront/public/cool/coolLogs/tests_reconstruction/read/05-10-17_20-53_lxplus8Machines
IOV # 1 ~/dfront/public/cool/coolLogs/tests_reconstruction/read/05-10-18_19_34_Since1ManyClients

Avg elapsed, IOV # 1000 vs. # 1

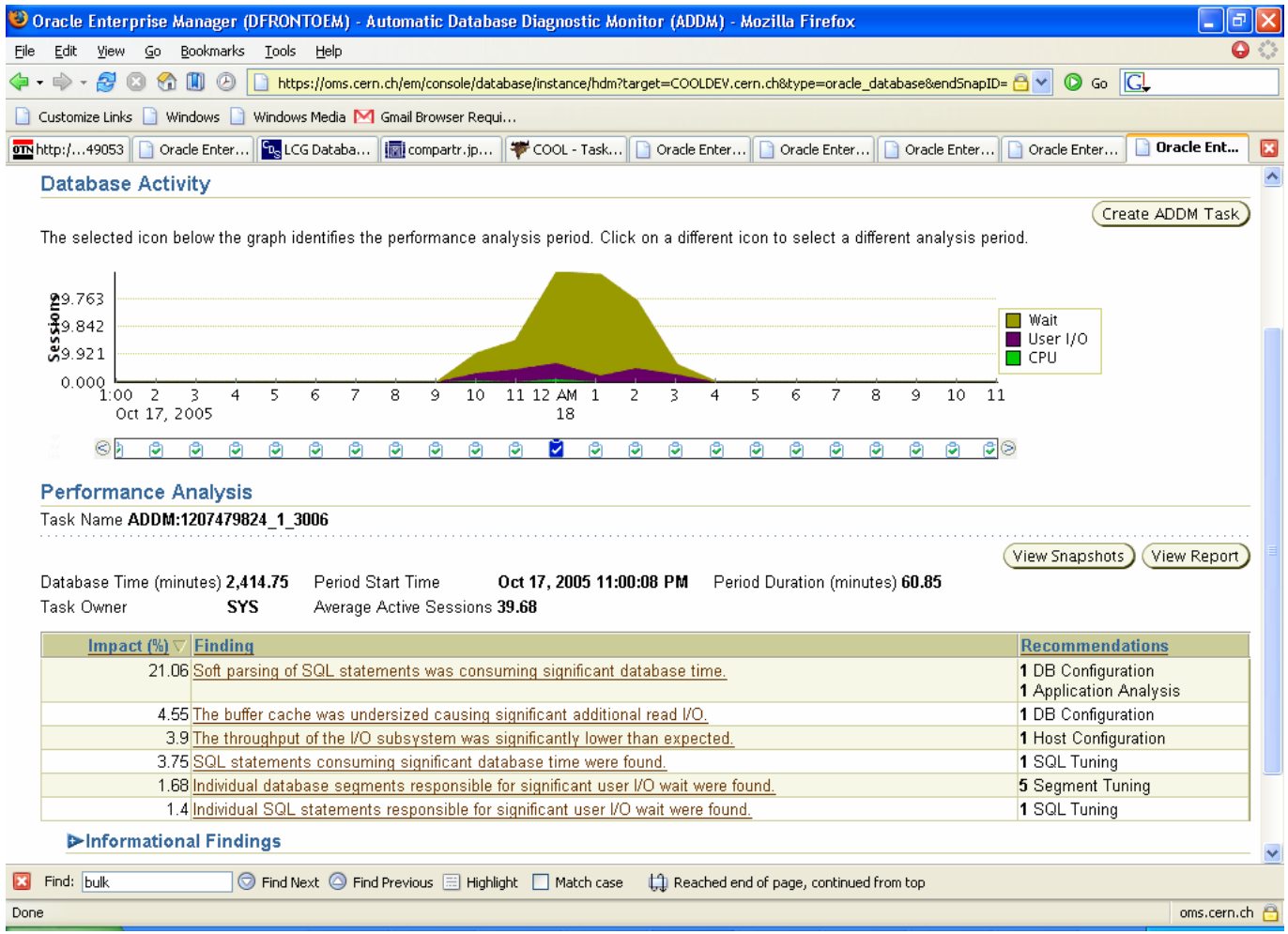


IOV #1000

IOV #1

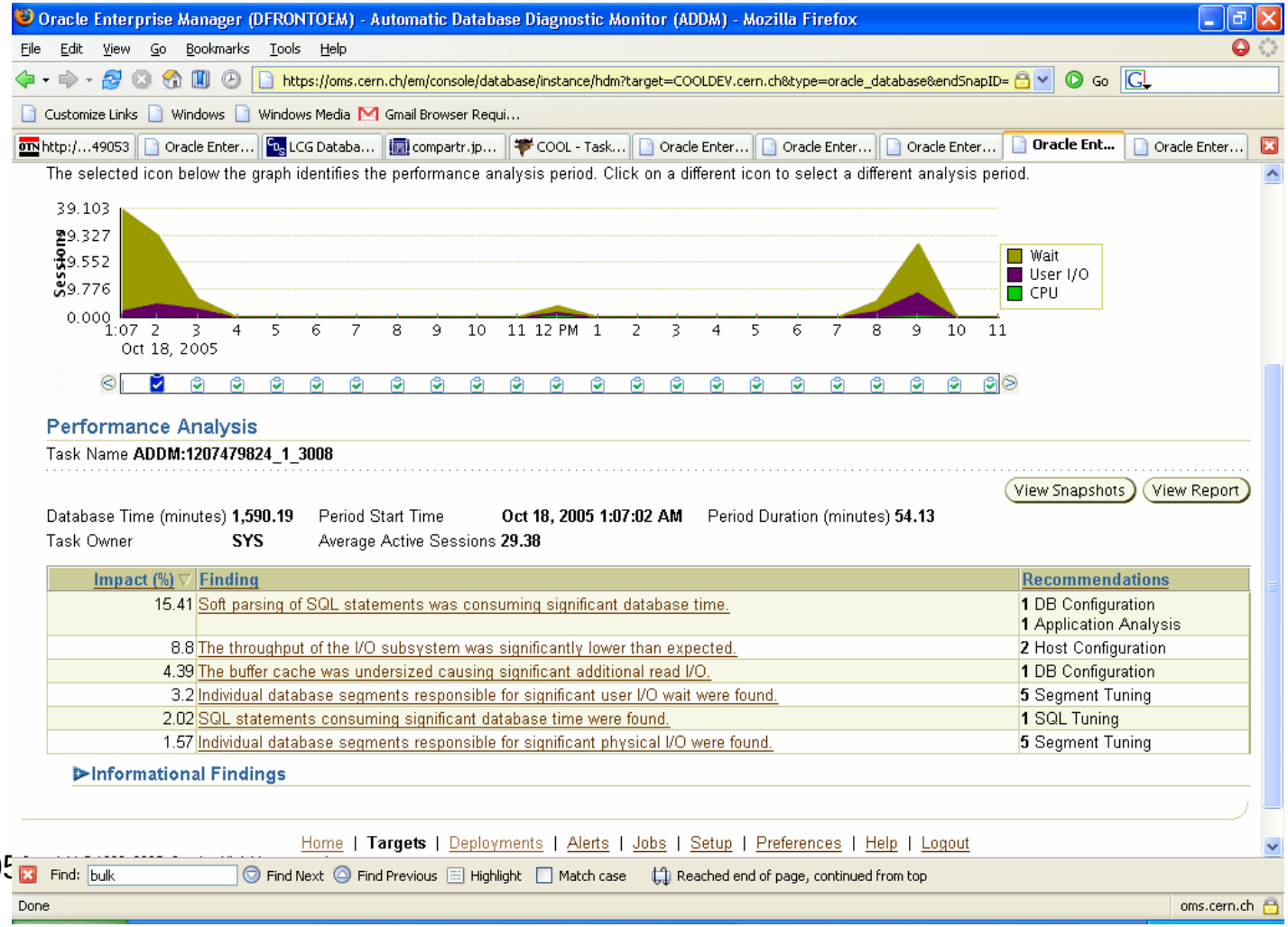
OEM recommendations IOV #1000

https://oms.cern.ch/em/console/database/instance/hdm?target=COOLDEV.cern.ch&type=oracle_database&endSnapID=3006&startSnapID=3005&task_id=3043



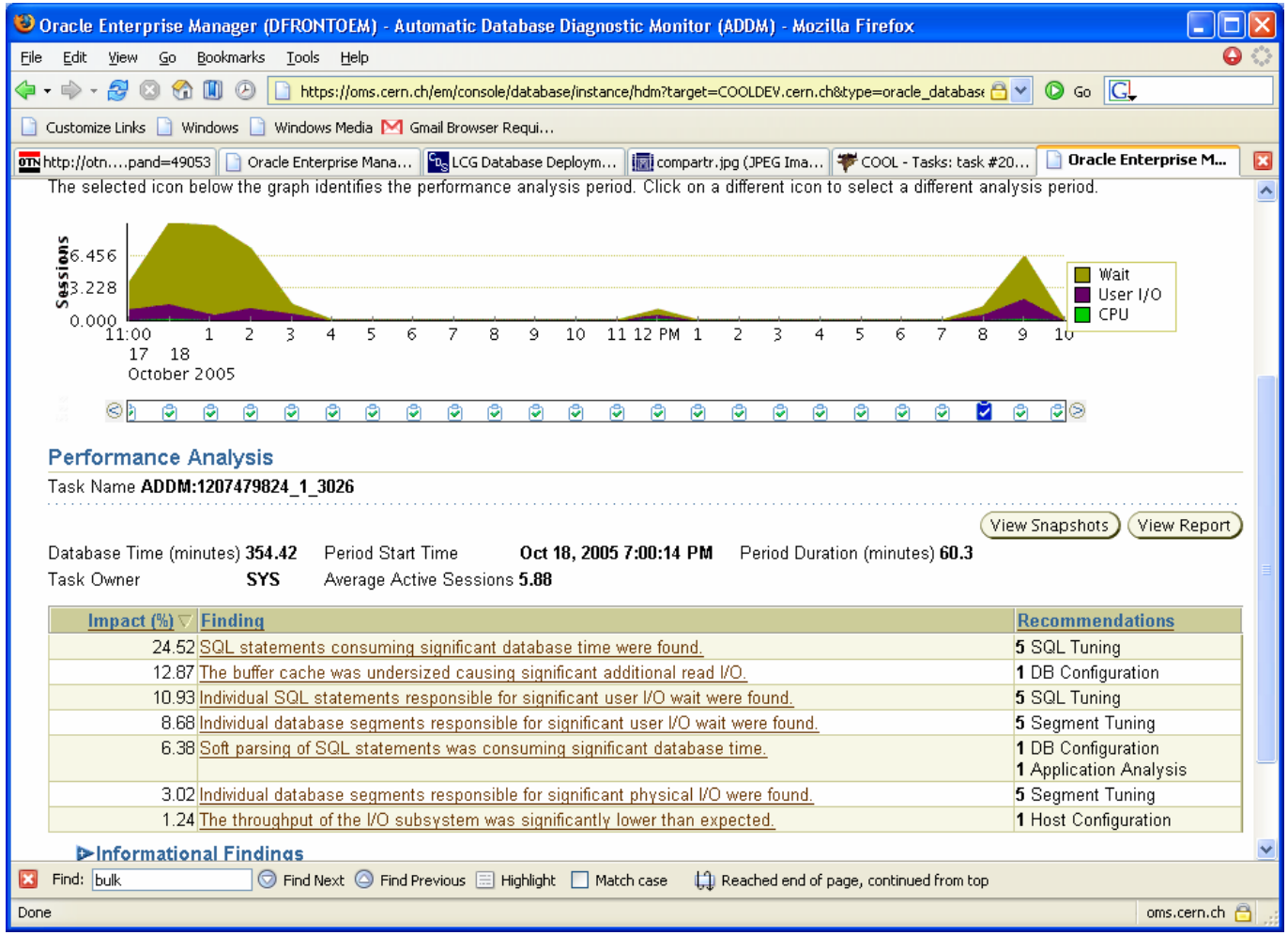
OEM recommendations IOV #1000

https://oms.cern.ch/em/console/database/instance/hdm?target=COOLDEV.cern.ch&type=oracle_database&endSnapID=3008&startSnapID=3007&task_id=3045



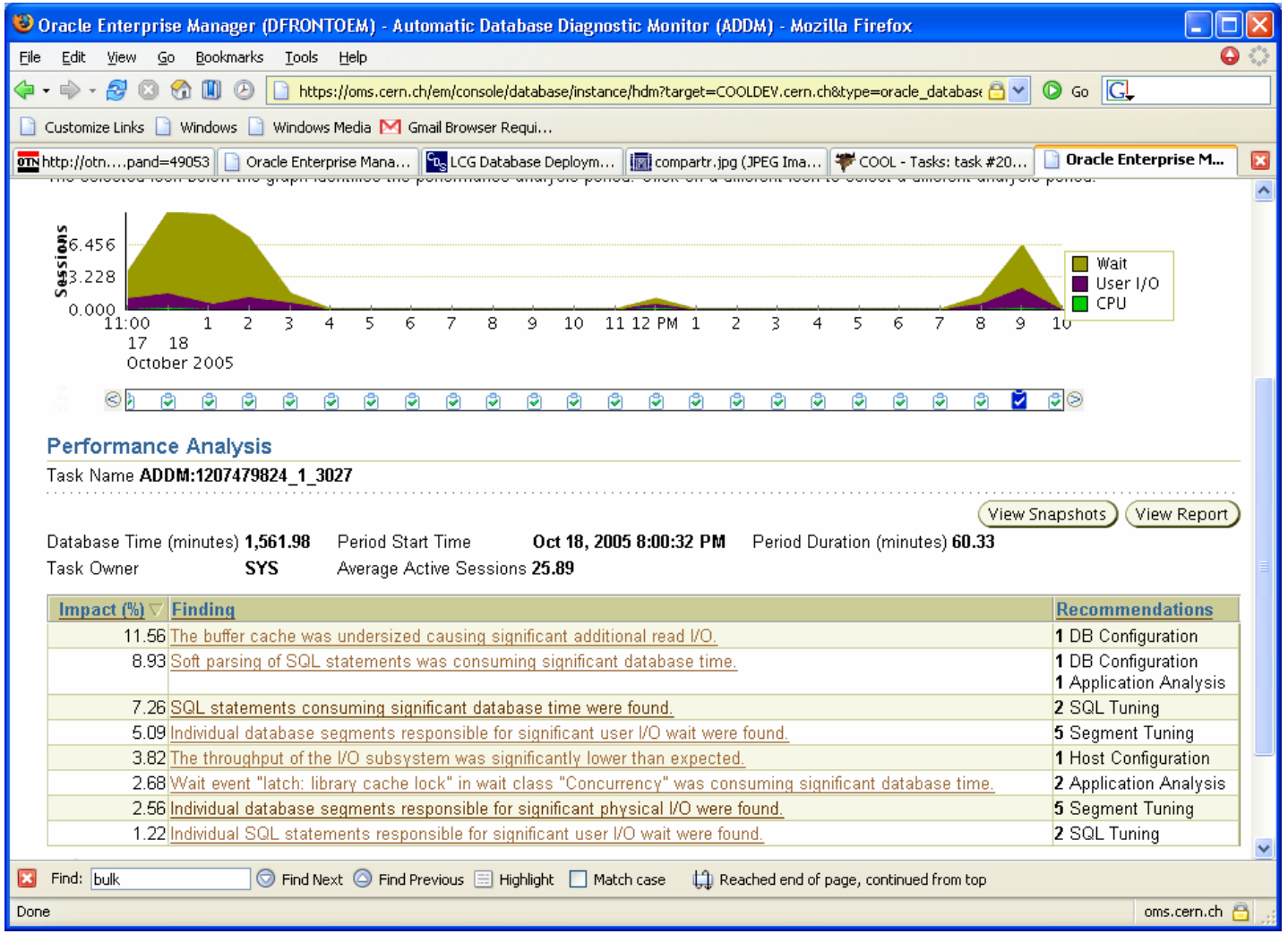
OEM recommendations IOV #1

https://oms.cern.ch/em/console/database/instance/hdm?target=COOLDEV.cern.ch&type=oracle_database&endSnapID=3026&startSnapID=3025&task_id=3063



OEM recommendations IOV #1

https://oms.cern.ch/em/console/database/instance/hdm?target=COOLDEV.cern.ch&type=oracle_database&endSnapID=3026&startSnapID=3025&task_id=3064&event=view_result



Enhancing the simulation: sustained insertion

- My testing was done only for one IOV (simulating 5 minutes)
- This was done, assuming that the data of each IOV is independent to each other
- Actually, the arrival of reading jobs will be sustained
- If Oracle server benefits from reading one IOV, in order to read the next IOV faster/ more easily, one could expect better results
- I did not test this yet, but as you will see, Andrea did

Testing results

- My testing so far failed to show that COOL 1_2_4 does comply with the Atlas reconstruction workload requirements
- While reading IOV # 1, results are generally better than for IOV #1000, but in both cases the server is too stressed and performance is too low
- The (soon) planned fix of issue
  #2223Performance of IOV retrieval is not uniform in [-inf, +inf]
  should enhance compliance with Atlas reconstruction requirements
- Because of server stress, it does not make sense to extrapolate the performance of full workload from my current (partial) numerical results
- LHCB has considerably less demanding reconstruction requirements and hence may probably be met by COOL 1_2_4
- Actions:
  - Run sustained insertion for a while (on RAC server), to learn if this betters results
  - Learn OEM (and TBD Oracle support) recommendations
  - Learn from Andrea's more optimistic testing

Verification plans/ wish list

- Repeat insertion and reconstruction testing when following issues are fixed
  - #2223Performance of IOV retrieval is not uniform in [-inf, +inf]
  - #2009Simultaneous bulk insertion to more than one channel at a time.
- Test closer to realistic workload:
  - Tables with 1 year of data (rather than a few days)…
- Enhance definition of the reconstruction workload
  - Assuming that Richard H. has fresh input for this
- Enhance testing environment
  - Use lam at my framework
- Wish: Access OEM data programmatically
- Suggestions?

Lessons

- Once testing diverts considerably from ‘normal’ conditions (client or server is highly stressed), rather than driving (wrong) conclusions from results, it is better to locate and fix the cause reasons
- Closer work and better communication with Databases Oracle support and Andrea could enhance that to allow future testing to focus faster than past testing
- Interesting problems tend to ‘hide’ as side effects
- I ‘reinvented’ a ‘lam’-like testing framework ☹

Links

- Relevant savannah Items:
  Verification tasks:
  #2398Scalability tests for COOL data retrieval performance.
  #2409Atlas workload tests for COOL data retrieval
  #2566Comparison of LHCb and Atlas reference workloads
  Gating issues:
  #2223Performance of IOV retrieval is not uniform in [-inf, +inf]
  #2009Simultaneous bulk insertion to more than one channel at a time.
- cvs code: Cool/contrib/VerificationClient/
- Log directories, per tests suite
  ~dfront/public/www/cool/coolLogs/tests_reconstruction/read/*
- Per test configuration script:
  At each log directory: tests_reconstruction.py
- Result sqlite DB files:
  At each log directory: coolTest.db

A happy new year!

A blessing
for a (Jewish)
happy new year
by a young friend