Subject: Notes for further performance tests on the GSI Tier2
Posted by Silvia Masciocchi on Tue, 01 Jul 2008 07:50:43 GMT
View Forum Message <> Reply to Message

Here come some scattered thoughts, summarizing a few things which still need to be done as performance tests on the GSI Tier2 farm.

INFO (code and data)
==========
- the train used for the reference tests, in June 2008, can be found in /u/sma/train/v3/. This version will not be further modified in order to stay compatible with the large scale tests.
- root is 519-04, par files are from AliRoot v4-13-Release
- the train is run using the macro AnalysisTrain.C
- the script train.sh is used to submit jobs to the batch farm
- the data used are the 6.5 million events produced by me at GSI in the past months. They are both from aliroot v4-10-Release and v4-13 (Release and Rev-01). Due to their different size, data lists are produced (by Anton) by complete mixture of the different event types. Lists can be found in /u/sma/data/mc/speed/. Each list contains 100,000 events.
- more info about the data themselves is at http://alice-wiki.gsi.de/cgi-bin/view/Data/WebHome
- CPU and memory consumption are also monitored with the AliSysInfo by Marian. The output is stored in a syswatch.log file. The information can be plotted using my macro (preliminary version) /u/sma/train/v3/RESULTS/plots/mem.C

TO DO
=====
1. Measure the EFFECTIVE volume of the data read in, by the train. Suspicion: The sum of AliESDs.root, galice.root, Kinematics.root, TrackRefs.root is about 20 MBytes. Compared to that, reading the traffic through the 1 Gbit link (node-switch) indicates a higher volume of data. Check.
Specifically we can compare the effective volume read in when the files are zipped or not (when zipped, other root files are included)

2. I did a preliminary test with read ahead, which showed no sizeble advantage. But in my test the read ahead was implemented ONLY for the AliESDs.root file, while now we read also the MC truth and therefore it has to be implemented also for the other 3 files. Please, see my preliminary modifications in
/lustre_alpha/alice/sma/soft64/AliRoot/v4-13-Release/ANALYSIS/AliAnalysi sManager.cxx
This can then be tested on the single machine (e.g. lxb281) running 1, 2, 4, 6, 8 jobs in parallel and compare with my results.

3. Since lxb363 had problems during my tests, I do not have reliable results reading from 4 separate, individual disks (in my tests disks 1 and 3 resulted to be faster than 2 and 4). Redo this point completely, without and with read ahead.

4. Several ideas from Volker Lindenstruth (check with him)
4.1. check the difference between data loading and data processing (e.g. load one event, process again to compare times without loading in between. Or flush cache between events)
4.2. measure average processing time per event
4.3. measure difference between nodes with 2 or 4 GBytes memory per core (to decide what to buy next in terms of memory)

5. Ideas proposed by Carsten (from his email, June 24, 2008)

5.1. One question to answer is, if the limiting factors on the Lustre side are the server them self or the GBit links. To answer this I've asked Thomas for configuring one of the Lustre nodes with link truncation (2 GBit/s then).
5.2. The other question concerning the switches is, how much data the switches can handle. Therefore your test should be redone with using only a part of the nodes at a time, first lxb281-lxb304 and then lxb327-lxb363. This will then show how much real data we can transfer (90-95% of 1GBit/s) and if the "bad" switch is still bad.

6. Measure performance when reading from the Storage Element (to estimate the length of the analysis when run as GRID job in AliEn)

7. This train could be used as monitoring tool for the batch farm and its network infrustructure. We could prepare a reference job (this train with a given data sample) and run it regularly on a random machine of the batch farm. By looking at its performance (e.g. real time needed, could be more), and saving all details of the run (e.g. which node, and consequently which switch, which 10GBit link, which way through the backbone), we can monitor if something in the farm is wrong (e.g. machines overloaded and out of optimal working point, switches or links having problems, and so on). It would be a higher level monitoring, but would avoid the user to have to find out that machines are out of good shape since days, to wait much longer for a given processing and so on.