CDF/DOC/COMP_UPG/PUBLIC/xxxx

# CDF Task Force on Computing Usage Interim Report (31 August 2004)

**Stefano Belforte (Chair), Patrizia Azzi, Ray Culberston, Ashutosh Kotwal, Song Ming Wang, Matt Herndon, Konstantin Anikeev, Elliot Lipeless, Igor Sfiligoi**

## 1 Introduction

This Task Force was formed by the CDF spokespersons on May 26 2004 to investigate the level of optimization (or lack of) in usage of computing resources. The obvious goal is to maximize the experiment physics output from one side by demonstrating funding agencies that we are using computers effectively, therefore our need estimates are sound, from the other by indicating ways to use more effectively the resources we have. We recognize the difficulty of the task and tackle it with humility. We have no hope to put a final word on this, and will only try to contribute knowledge and thought to what has to be a continuous work of managing machines and humans in coordination.

## 2 charge

```
Charge for the CDF-Grid Resource Usage Task Force
```

This TF is charged with reporting on the current degree of optimization in using CDF computing resources for analysis and on how to improve it. The TF will take into account resources both at FNAL and in the CDF-Grid and will develop and deploy monitoring and accounting tools as needed to study the situation. An additional product of this TF will be the accounting of usage of computing resources off site, as requested by CDF International Finance Committee.

As a guideline to the kind of result we expect from the TF, an itemized list of topics follows:

A: Identify existing usage patterns with a goal of optimizing resources
- A1: Are the physics groups centralizing usage (e.g.. producing ntuples/skims for everyone's use)?
- A2: What fraction of the physics group usage is centralized?
- A3: Will these same patterns of usage work for 2fb-1?

B: Look in detail at whether central and non-central usage is optimized
- B1: Identify heavily used common tools that would improve performance if sped up.
- B2: Prescribe a systematic way to improve the usage

C: Breakdown usage on each CAF system
- C1: define and implement a strategy of what to monitor so to have results by end of summer on
- C2: usage by country, by institutions, by physics group
- C3: usage for data analysis vs. MC vs. central vs. non-central usage

Please prepare a short written report in time for the next IFC meeting, which will be in the fall. You should get up to speed quickly enough in order to monitor the large usage for ICHEP preparations.

## 3 Task Force organization

We decided early on to proceed as follow

- Concentrate on CPU usage. There seems to be little to say about disks other then a general encouragement to keep event size small. Of course we expect that need for large disk resident samples and/or large disk caches in front of the tape robot will be reduced as users use more ntuples and less DST, but since this topic is addressed by our CPU usage optimization effort, we do not make a separate discussion of disk sizes.

- Survey usage patterns as from physics groups Representatives in task force and/or conveners could report. We met 2 times for this and wrote minutes and slides. More work was done vie e-mail given the geographically distributed nature of the Task Force.

- Define a procedure for monitoring user's CAF usage over the summer to verify that and make it quantitative. Data was defined by end of June, but pressing issues with CAF operation and other delayed implementation until end of July on Condor CAF and mid of August on "normal CAF".

- Go back to discussion once we have the monitor result, in September.

## 4 This report

This interim report reflects our reaction to initial monitor findings and a first assessment of whether the initial picture described by physics groups was reasonable. Mostly we have been looking for a quantitative measurement of the impact on CPU usage.

Several important notions have been learned in this process, that makes us less optimistic to have a "conclusion" by end of September:

1. Setting up a proper monitor is difficult and we have only partially succeeded so far.

2. Introducing good monitoring requires changing the core of the CAF system, which is difficult to do without affecting operations. Also only newer version of CDF code report informations like I/O usage.

3. The current Condor CAF "analyze" page

   `http://cdfcaf.fnal.gov/condorcaf/analyze/index.html`

   that was setup by this Task Force members is proving extermely valuable, but still it has to be extended to normal-CAF and to offsite farms, and we need to do a much better job at tracking dataset usage and at classifying the work as Analyisis or MonteCarlo (to begin with)

4. Usage patterns varies considerably during the year and monitoring over a long period is required

5. Our analysis farms are heavily used by many people doing many different things and the overall load is a sum of many small pieces, with no way to single out a few "bottlenecks" to tackle.

6. Even once we (will) have good monitoring in place, analysis, digestion and data reduction of those information is a daunting task

# 5 Physics Groups Survey

Here you will write a short paragraph for each group with a summary of what you reported to our June meetings. I will fill the QCD part.

## 5.1 Top

### 5.1.1 Datasets

The Top Physics group oversees about 40 different analyses. Since most of these analyses rely on the same final state signature the overall number of datasets for the signale is fairly "small" compared to other groups. However, several other datasets are needed for background determination, b-tagging efficiency measurement and energy scale definition. In more detail:

- Signal samples: High $p_T$ electron and muon, multijet
- Other samples: low $p_T$ electron and muon, inclusive jets (20,50,70,100), photon plus jets, and $Z \to b\bar{b}$

The stripping of these samples is done by a responsible nominated by the Top Group and is saved in official datasets in the DFC.

On the Montecarlo side instead the number different processes that needs to be generated, simulated and reconstructed for background, efficiency and energy scale measurement is large. One of the first effort of the Top Group has been to centralize and organize the generation and production of MC datasets common to most analyses and possibly also to other physics groups (the role of "Top MC coordinator"has been created, currently D.Wthitesons `http://www-cdf.fnal.gov/ danielw/topmc5/`). This coordinator allows to minimize the duplication of the effort for large scale common datasets, an efficient interaction with the simulation group, and also an easier coordination with the other physics groups. A web page is mantained: `http://www-cdf.fnal.gov/d̃anielw/topmc5/`. The production of this official MC goes through the available farms (Toronto, McGill, Karlsruhe). Clearly given the number of analysis, it is clear that there is always a residual need for MC datasets that is analysis specific: in these case the Top Group makes available the instruction for the "official" processing so that the "custom made" samples can be as close as possible to the official ones and the user does not have to revalidate the whole procedure (simplyfing test jobs). These custom MC are typically produced on the CAF.

### 5.1.2 Analysis Tools and Ntuples

The Top Group has developed a main analysis module (`TopEventModule`) used to define the official variables definition and event selection for the different final states signatures, this insures coherence of the analysis and ease of validation for new releases. This module provides an object in the output event record that can subsequently be read out for further analysis or ntuplizing. The most used ntuplizing module is TopNtuple that is the one developed by the same people and in conjunction with TopEventModule. It is by far the most used, however there are a few other analysis ntuples used by some analyses (STNtuple, UCNtuple, modified TopNtuple). For this reason the production of TopNtuple on the official and most used datasets is organized by the group and is done by a few people (current responsible: Sal Rappoccio, Ben Kilminster) and stored on CAF disks. Some numbers (as of June 2004) of what the analyses are using:

- Official TopNtuple: 12

- modified TopNtuple: 5
- Rochester Ntuple: 3
- Duke Ntuple: 1
- UCNtuple: 1
- STNtuple: 4

Most of the ntuplizing effort happens when there is a new "physics" release, meaning a new production code but also a new tagger and or new prescriptions for event selection and refitting. Typically the set of instructions to be used for publication is decided by the group and the code "frozen" in a set of CVS tags described on a web page.

### 5.1.3 Analysis work details

Most of the analysis work for top physics is actually done at the ntuple level. The ntuples used are fairly inclusive and flexible to allow further analysis in a creative fashion (not complete black box). The platform used are mostly the desktop using rootd to access the ntuples on the CAF disks, however for large datasets, root jobs are sent over the CAF. The overall size of the official data ntuple (high pt lepton) is about 12 GB(and a single ntuple size is about 22kB/event (data) and 33kB/event(MC)). The ntupliziong jobs are not very time consuming. Having available queues on the CAF the whole high pt lepton datasets can be ntuplized in a few days. The whole reprocessing of the data instead can take much longer depending of what goes into the job itself (the worst case example is the "REMAKE" experience of 2003 where we had to run manually almost a production job).

## 5.2 Exotics

Ray and/or Ming

## 5.3 Bottom

Matt. note that I have some info from S.Giagu about ntuple usage in Bottom.

### 5.4 QCD

Stefano

### 5.5 EWK

Ashutosh

## 6 tools

Here Igor and Elliot will describe the monitoring tools they setup.

## 7 A couple of examples

We did attempt to look at CPU usage inside some "typical" user's job, to see if there is some obvious cpu-eating module that can be attacked, optimized, and then bring benefit back.

Unfortunately even typical jobs are made of many different moduls, and often man yinstances of the same with differnt paramters cuts that show very different times. So while it is clear e.g. that when CVTMFT is used for finding 4-trakck vertexes it is awfully slow, in most other cases its load is not different from other modules. Very unlikely we can come up with a silver bullet recipy.

#### 7.0.1 topNtuple creation: 1 sec/event

Kindly contributed by Sal Rappoccio:

```
this is with 5.3.1

 During Event Processing:
 ========================
 Module name:         # Calls:     Mean cpu time:        Mean clk time:      Total Cpu:
 ------------------------------------------------------------------------------------
 CalibrationManager      59391   0.000006+/-0.000001   0.000011+/-0.000002   0.360
 ConfigManager           59391   0.000005+/-0.000001   0.000007+/-0.000000   0.310
 DHInput                 120119  0.016141+/-0.000164   0.037326+/-0.000709   1938.820
 ErrorLoggerManager      59391   0.000005+/-0.000001   0.000008+/-0.000001   0.320
```

```
FileOutput               118791  0.000030+/-0.000005   0.000051+/-0.000010   3.530
GeometryManager          59391   0.000007+/-0.000001   0.000008+/-0.000001   0.400
JetProbModule            59391   0.093883+/-0.000482   0.135950+/-0.000720   5575.820
PuffModule               59391   0.009744+/-0.000024   0.013123+/-0.000067   578.730
SecVtxModule             59391   0.043092+/-0.000113   0.062450+/-0.000206   2559.250
SignalManager            59391   0.000002+/-0.000001   0.000008+/-0.000001   0.140
SimInitManager           59391   0.000004+/-0.000001   0.000007+/-0.000000   0.210
TopEventModule-Lepton    59391   0.123256+/-0.000203   0.177688+/-0.000336   7320.290
TopEventModule-WZ        59391   0.124028+/-0.000200   0.171331+/-0.000335   7366.120
TopFilter-BTag           59391   0.000287+/-0.000007   0.000408+/-0.000012   17.050
TopFilter-Lepton         59391   0.000298+/-0.000007   0.000429+/-0.000012   17.670
TopFilter-W              59391   0.000227+/-0.000006   0.000342+/-0.000012   13.460
TopFilter-Z-loose        59391   0.000200+/-0.000006   0.000282+/-0.000010   11.890
TopFilter-Z-tight        59391   0.000203+/-0.000006   0.000319+/-0.000011   12.070
TopNtuple-BTag           58880   0.314194+/-0.000937   0.452971+/-0.001387   18499.750
TopNtuple-Inclusive      59391   0.288126+/-0.000878   0.415451+/-0.001304   17112.080
TopNtuple-Lepton         1870    0.235610+/-0.003938   0.346086+/-0.005890   440.590
TopNtuple-W              484     0.137500+/-0.003909   0.198738+/-0.006085   66.550
TopNtuple-Z-loose        125     0.207840+/-0.010778   0.302853+/-0.016343   25.980
TopNtuple-Z-tight        17      0.204706+/-0.017597   0.312602+/-0.027761   3.480
--------------------------------------------------------------------------------
Mean Cpu total = 1.799393; Mean Clock total = 2.628447
Sum total Cpu = 106867.724197; Sum total Clock = 156106.080010
```

The machine where this ran was not indicated, some "generic" CAF node, so let's assume a 2GHz P4. The actual TopNtuple part takes 0.5 sec/event including BTag, and the last 2 lines from AC++ report makes no sense since the sum of the numbers in the last column is 61564 cpu-seconds. For 59391 events the average is 1.03 sec/event.

### 7.0.2 Hadronic B skimming for Bs cadidates: 4 sec/event

Kindly contributed by Donatella Lucchesi:

```
*** Execution Times for all Modules Run so Far ****

During Event Processing:
========================
Module name:         # Calls:     Mean cpu time:        Mean clk time:   Total Cpu:
----------------------------------------------------------------------------------
```

```
BAllHad-BD0k3pi         83      0.001084+/-0.000343   0.001092+/-0.000052   0.090
BAllHad-BD0kk           20      0.001000+/-0.000688   0.000996+/-0.000114   0.020
BAllHad-BD0kpi          41      0.000488+/-0.000341   0.000920+/-0.000085   0.020
BAllHad-BD0pipi         19      0.000000+/-0.000000   0.001026+/-0.000106   0.000
BAllHad-BDpl            77      0.000909+/-0.000330   0.001034+/-0.000051   0.070
BAllHad-BDs3pi          75      0.000267+/-0.000187   0.000956+/-0.000044   0.020
BAllHad-BDsPhi          17      0.000000+/-0.000000   0.001021+/-0.000095   0.000
CalibrationManager      1101    0.000009+/-0.000009   0.000009+/-0.000000   0.010
D1VertRecoModule-BD0kk  1096    0.012491+/-0.000311   0.012612+/-0.000289   13.690
D1VertRecoModule-BD0kpi 1096    0.020648+/-0.000501   0.021051+/-0.000510   22.630
D1VertRecoModul-BD0pipi 1096    0.010675+/-0.000288   0.010806+/-0.000264   11.700
D1VertRecoModule-BDk3pi 1096    3.293449+/-0.228879   3.314961+/-0.230339   3609.620
D1VertRecoModule-BDpl   1096    0.166195+/-0.007217   0.167284+/-0.007259   182.150
D1VertRecoModule-BDs3pi 1096    0.160821+/-0.007075   0.161874+/-0.007114   176.260
D1VertRecoModule-BDsPhi 1096    0.237509+/-0.011150   0.238844+/-0.011223   260.310
D2VertRecoModule-BD0kk  323     0.005263+/-0.000362   0.005078+/-0.000270   1.700
D2VertRecoModule-BD0kpi 495     0.004505+/-0.000275   0.004513+/-0.000186   2.230
D2VertRecoModul-BD0pipi 243     0.005021+/-0.000388   0.004882+/-0.000242   1.220
D2VertRecoModule-BDk3pi 352     0.015455+/-0.001623   0.015775+/-0.001601   5.440
D2VertRecoModule-BDpl   539     0.006327+/-0.000370   0.006611+/-0.000326   3.410
D2VertRecoModule-BDs3pi 578     0.006401+/-0.000359   0.006356+/-0.000315   3.700
D2VertRecoModule-BDsPhi 165     0.006000+/-0.000549   0.005344+/-0.000411   0.990
DBeamModule             1101    0.000100+/-0.000030   0.000085+/-0.000000   0.110
DHInput                 2209    0.012345+/-0.000330   0.012485+/-0.000331   27.270
ErrorLoggerManager      1101    0.000000+/-0.000000   0.000009+/-0.000000   0.000
FileOutput              2204    0.013466+/-0.001137   0.013692+/-0.001151   29.680
GeometryManager         1101    0.000009+/-0.000009   0.000008+/-0.000000   0.010
HepRootManager          1101    0.000018+/-0.000013   0.000115+/-0.000043   0.020
PuffModule              1101    0.020363+/-0.000263   0.020302+/-0.000231   22.420
SignalManager           1101    0.000018+/-0.000013   0.000008+/-0.000000   0.020
TrackAssocModule        1096    0.019206+/-0.000548   0.019366+/-0.000533   21.050
TrackSelectorModule     1101    0.042761+/-0.000599   0.042988+/-0.000603   47.080
*********************************************************************************************
I'm using CharmMods to reconstruct several decay channels:

BD0kk   --> Bu->D0pi   D0->KK
BD0pipi --> Bu->D0pi   D0->pipi
BD0kpi  --> Bu->D0pi   D0->Kpi
BDk3pi  --> Bu->D0pi   D0->K3pi
BDpl    --> Bd->D+pi-  D+->Kpipi
```

```
BDs3pi  --> Bs->Dspi   Ds->3pi
BDsPhi  --> Bs->Dspi   Ds->phipi phi->kk

D1VertRecoModule-XXXX: Reconstruct the lowest particle in the decay
  (i.e D0->KK D0->pipi ... Ds->phipi)
CharmMods fits the particles to one vertex using CTVMFT. I do not apply
 any constraints. When I have a sub-resonance like phi I just cut around the
 resonance mass I do not do an additional fit.
You see here that the execution times depend on the number of particles
 in the final state, i.e. on combinatorial.

D2VertRecoModule-XXXX: Reconstruct the final particle (i.e. Bu, Bd, Bs).
 The decays are almost the same a this level since we attach one track (pi)
 to the D and the execution times are almost equal.

DBeamModule: find the primary vertex (in this version the Beam Line)

TrackAssocModule: aasociate Def track with SVT in my case

TrackSelectorModule: make selection using number of hits and other track parameters.

The tracks are refitted using the latest rescaling of the covariance matrix.
```

This was ran on a 2.6 GHz P4 on CNAFCAF (INFN dCAF in Bologna) This time the totals are 1101 events processed in 4442.9 cpu-seconds, i.e. 4.0 seconds per event

# 8 findings

Here I write something provocative for you to react

- Usage patterns are complex
- Usage patterns are not what expected nor what we used to specify the system
- Up to half of analysis farm has been used for MC at some times
- top/exo/qcd/ewk claims most analysis work done on ntuple off the CAF

  - still users in those groups made large usage of CAF even before 5.3.3 tools are released

- lot of works appear to be done outside "organized framework", like user MC, tests of various kinds, we just do not know what

- there is general perception, and a lot of anecdotal evidence, of large "waste" due to repeating large tasks because of more or less simple mistakes, but no way to measure this that we could find

- CPU utilization in a single job ranges widely from a fraction of a second per event to many seconds, with no outstanding culprits.

- The bulk of the CPU usage is scattered among many users who hit the farm (heavily somtimes) for a time, and then disappear for long time. Nor a few heavy customers.

- This scenario is very difficult to "optimize".

# 9 recommendations

- Do more "at production time"
  will reduce CPU needs later on and reduce mistakes

- Make "official documented ntuples available
  will help new users get some idea of the data with little CPU usage, may again save on errors/mistakes

- Make data available for analysis earlier, prevent big pre-conference rush. On the same line, find ways for users to look at data as they are (re)processed, not just wait fo the full sample to be done.

- Divide farms in 2

  1. Official work: put here most of resources, control by phys. groups.
  2. Users works: everybody's playground, little resources, learning field for making mistakes etc.

  This can be done in lots of ways, does not need to be felt as punitive or limiting by users

- Speeding up CVTMFT is possible, and it should be done since a large fraction of very cpu-intensive jobs goes there, but there is no evidence that this will make a big impact on the overall cpu needs. Same for speeding up ntuplizing jobs.