CDF/DOC/COMP_UPG/PUBLIC/5787
December 14, 2001
Version 1.5

Physics Analysis Computing needs assessment CDF CAF Review Fall 2001

B.Ashmanskas, S.Belforte, I.Bird, T.A.Davis, F.Donno, R.Hughes, P.T.Keener, A.Kreymer, P.Murat, A.Reichold, M.Shimojima, D.Waters, S.Wolbers, F.Würthwein, Ch.Young

Abstract

This is a work document of the CAF review committee contributing basic numbers for the final report. The CDF analysis needs are reviewed and translated into computing needs in order to be used in defining the upgrade directions for the Central Analysis Facility. The needs are estimated to be in the order of a thousand of today's fastest Linux processors and a hundred TB of disks.

1 Introduction

Whom we are, why we do this ...

This is a work document by the committee named by the offline management to review the CDF Central Analysis Facility (CAF) and advice on direction for CAF evolution in the future. The committe charge, report and work organisation are in the committe final report [1], toghether with many usefull links and references.

In order to carry out its task, the committee made yet another approssimation at assessing the computing needs for physics analysis in the light of the results of our benchmarking studies [2], of information gathered from the physics groups and of various presentations heard at the committee workshop on October 18-19 at Fermilab (see [1] Appendix D).

This document is our estimate of the physics goals and relative computing needs. In doing this we had to make assumptions about the way of working of the collaboration that may not prove to be correct (e.g. we assumed that primary data are mostly accessed only in a well defined and organised ways by the phyisics groups) so we tried to make it clear which those assumptions are.

2 Summary

Goals:

- Enable Physics Groups to process Primary Data Sets to create Secondary Data sets for users analysis (skim)
- Enable all CDF physicists to analyse data using those secondary Data Sets (analysis)
- Allow each user or Physics Group to make one pass on the data set chose in a matter of days

Conclusions:

- (Re)Tracking can only be one on the farm (of course)
- Primary to Secondary feasible using the entire upgraded fcdgsgi2 and some patience
- user's analysis has two options (in units of 1GHz processors):
  - run at O(10Hz), need O(1K) processors, each user needs 10 parallel jobs
  - run at O(100Hz), need O(100) processors, each user runs one job for each data set
- user's analysis will require O(70TB) of disk cache

Comments:

- the more we tried to pin-down our estimates, the more we are aware of their uncertainties. We believe the error bar on the CPU needs is at least a factor 2, but should not to be off by a factor 10.
- parallel jobs are a pain. User (groups) will look for formats that allow O(100Hz), they already exists (StNuple, optimised stripped PADs like MC and jets) and will be used, binary files were and will be popular also if they give performance. If everybody goes for a different solution, chaos may follow.

Recommendations (prioritised):

1. need to plan for hundreds of processors at least, scalable to a few thousands
2. need official prescription for fast access data set (thumbnails ?). This could be multi-branch PAD or something like "Standard Ntuple". All experiments but CDF have it. This format beyond being user's friendly can cut by a factor ten the needed number of CPUs.
3. need an integrated approach to hardware and software design because trade offs between speed of access and size of the data format impact heavily not only the user's job execution times, but also the hardware choices and the budget.
4. need to start looking into ways to automatize breaking long jobs in many parallel ones

3 Useful numbers

- unless explicitly stated we use 1day=10^5 seconds (vs. $60 * 60 * 24 = 86400$) this is off (on the optimistic side) by 15%.
- unless explicitly stated we measure CPU needs in units of "1GHz processor" i.e. 1000 Mips (see [2] Appendix C). This is about 4 times faster then current fcdfsgi2 [2] and is more or less one 1GHz P3 running CDF code on Linux.
- $10nb \times 1fb^{-1} = 10^7$ events $\times$100KByte/event = 1TB
- 5nb at L3 means 1×10^7 events for Run2a = 1 TB

4 Physics Needs

We assume the following:

1. full data sample for Run2a = 2 fb^{-1} = 2×10^9 events
2. data sample for Summer 2002, two possibilities:
   (a) 200 pb^{-1} = 1/10 of full set
   (b) 6 months @ 30Hz = 500×10^6 events = 1/4 full sample
3. data sample is organised in Primary Data Sets ad defined in CDF4718 "CDF Run-II Trigger Table and Datasets Plan", November 2001. See also CDF5565 "Datasets and Raw Datastream for Run II", February 2001. Both are evolving documents available from the Trigger and Datasets working group page:
   http://www-cdf.fnal.gov/upgrades/daq_trig/twg/twg.html

   Without replicating here the Data Set list from Section 8 of CDF 4718, it can be summarised in its present version as (see Figure 1):

   (a) 46 independent data sets
   (b) L3 σ ranges from 1 to 100 nb, average is 12, rms is 18, most frequent value is 5, only 10 data sets have more then 10nb, only 4 more then 40, average in the 0-20nb range is 5.5 , sum over all data sets is 556 nb.
   (c) we will take 5.5nb as "typical" and round it to 5nb = 10^7 events for all of Run2a.
   (d) we will take 556 nb total as total data sample size to analyse, and assume there are $556/5 \simeq 100$ (in rounding we are throw 10% of optimism) data sets instead of 48. Each is 1TB of data for Run2a. Total is 10^9 events.

4. in summary Primary Data sets is a collection of 100 Data Sets, each 10^7 events, in PAD format 100KB/event, 100TB total. This figure is $\sim 1/2$ of the "classic" estimate of 160TB. We do not know how to reconcile things at present. Some trigger is still missing from CDF4718, total L3 budget ranges from 375 to 750 nb from high to low instantaneous luminosity. There will be a big difference in the total number of logged events if there is an extended period of data taking with a larger L3 cross section. Most likely this means that for the full Run2a this estimate may be off by a factor of 2 in total number of events, on the other hand most data sets size is well defined by the "signal cross section" (i.e. all high-P_t ones) and should be constant for a given integrated luminosity.

5. As a cross check, current estimates by Physics Groups are $0.5 - 2.5$ TB per data set from the exotic group [12] [1], while the top/EWK group's compilation [5] of all their major samples shows that with exception of photons they are all $\leq 10^6$ events for $200pb^{-1}$.

6. in this "exercise" large data sets as $B_{(s)} \rightarrow D_{(s)}\pi \rightarrow \geq$hadrons (L3$\sigma = 100nb$) are handled as 10 "typical" ones and users who can not restrict to sub-samples will have to wait longer, or manage to split/parallelise their jobs more.

7. most user work (we call this process **"analysis"**) is done on Secondary Data Sets that are (see also CDF4568 "Data Handling Criteria : CDF Physics Groups", April 1998)

   - produced in official organised way by Physics Groups (we call this process **"skim"**)
   - in "fast access" format
   - 50-100 KByte/event
   - each derived from the corresponding primary data set with some tuning/clean up. The corresponding reduction in number of events is unknown at present, we will use the same size for Primary and Secondary Data Sets, but some common sense and comparison with Run 1 argument (like hight Pt lepton sample in Run1 was O(20GB), even times 20, it is much less then high Pt e-mu Primary Data Set) suggest that larger data sets will be somehow reduced removing background (something like the number of high-Pt signal events should be of the same order of magnitude regardless of signature). Hopefully this will compensate a possible underestimate on the number of events based on the L3 cross sections.
   - of the size of a few to several million events each (on the Run2a full scale).

8. there will be 200 active users, meaning 200 people running jobs on the 2ndary data sets at each time (the top/EWK group estimates 150, a half of them spinning on large data sets, the exotic group estimates 50 analysis to be based at Fermilab, from the CDF members list we imagine 400 physicists active on analysis, we assume they will actively run jobs about 50% of the time), this is a "steady state" number that does not include peak values before conferences etc.

[1]Ray has assumed 200KB/event so his list has 1-5 for Run 2a

5 What can be done with (upgraded) fcdgsgi2

Assessing where we stand is preliminary to any future estimate. We use the performances reported in the benchmark document [2], restrict to fcdfsgi2, use Mips*second to scale to 1GHZ processors (as discussed in that document) and here we are only concerned with picking the proper line(s) from the benchmark table.

For TrackTest and PhiKAna programs we take timing as measured running on ttbar simulation. For this we have to take some numbers reported in [2] only on cdfpca and rescale to fcdfsgi2 multiplying execution time by 3. The relevant numbers are:

```
           cdfpca      fcdfsgi2
input       60 msec     180 msec
TrkAna      24 msec      72 msec
StnMaker   400 msec      135 msec  (assuming no Puff)
DHoutput   131 msec     318 msec   (from StnMaker)
PhiKAna    347 msec    1041 msec
```

For "typical" skim and analysis jobs, we combine tasks as follows:

skim Use TrkAna as representative of the "user part", assume that a skim will also write ntuples and thus add in the StnMaker ntuple filling time (assume no time spent in Puff, i.e. take the sum of the lines: FillStntuple, InitStntuple, StntupleFilter and StntupleMaker from Table 1 of the benchmark report [2]), take input time as measured but assume DHoutput will be faster then what seen by StnMaker. We obtain:

Skim (Secondary Data Set creation)
= input + TrkAna + Stnmaker + output
= 180 + 72 + 135 + 318 = 705msec/ev = 1.4Hz on sgi2

Clearly there is plenty of room for more complex user operation (i.e. increase the 72 msec of TrkAna) before significantly affecting performance. Removing StnMaker or reducing DHoutput time to 100msec all bring to 500msec $\sim$ 2Hz. Doing both gives 3Hz. For comparison present stripping jobs run by the Physics Groups on September/October data (very simple event selection and no ntuple filling) run at 2.3 Hz on fcdfsgi2 [11]

analysis Use PhiKAna (which already includes a lot of histogramming and ntuple filling) and DHinput only. The processing time will be:

Analysis (user's processing of Secondary Data sets)
= input + PhiKAnala
= 0.180 + 1.041 = 1.22 sec/ev = 0.8 Hz

5.1 Summer 2001

In Summer 2002, assuming $200\mathrm{pb}^{-1}$ of integrated luminosity, most Data Sets will be of the order of 1Mevent [5], the time to process them is:

Frequency	Time to pass 1Mevents
3 Hz	3 days
2 Hz	5 days
1.4 Hz	7 days
0.8 Hz	13 days

In other terms: 1 fcdfsgi2 processor will allow to create one secondary data set in one week (or less), **or** to analyse it in two weeks.

The above times are acceptable [5], but there will be 100 data sets, 200 users, 100 processors (out of the 128 CPU's in the upgraded fcdfsgi2, 8 are dedicated to code build). User's can't expect to integrate 2 CPU-weeks in 2 weeks of calendar time.

It is easy to see that skim has a change to be done, and will also easily use a lot of the total resources, some processors will be used (as now!) for general purpose and miscellaneous tasks (code development, limited MonteCarlo, limited analysis), how much will be left for users analysis depends on how much skim will have to be iterated. Since at the beginning code and constant tend to change very often very little or nothing may be left for analysis. Assuming (optimistically) 50 CPU's to be available for users analysis, each of the 200 users will go through one data set in 2 months. This is way too long.

It is also possible that in Summer 2002 we do not have 1/10th of the data ($200\mathrm{pb}^-1$), but 1/4th (6 month of data taking at 30Hz average).

The bottom line is that the new fcdfsgi2 should be enough for the skims, but little or no resources will be available for users analysis, even if analysis programs were faster.

5.2 Full Run2a

Clearly there is no way that fcdfsgi2 can come close to the goal when data sets increase by one order of magnitude.

6 Computing Needs 1: Secondary Data Set Re-Creation

In this and the following section we look more at the global picture for full Run2, so assume some sort of steady state running were the rush to do everything at the same time has settled and resources can be estimated in “traditional”ways. We do not attempt to guess how much higher usage peaks (pre-conference e.g.) can be.

The first time Secondary Data sets can be created directly by the reconstruction farm, or in any case are created as data gets collected, at a slow rate, this means 20MBytes/sec average, we know from benchmarking that $10 \sim 20$ AC++ jobs can keep up with this, so that is not a problem. The issue is refreshing a data set once new code/constants are available, at that time one wants to process $O(10^7 \sim 10^8)$ events in "days".

Primary→Secondary data set is a coordinated activity, which:

1. is not done "routinely" so users will accept a longer time to have a new version of data set available, then the time they will need when accessing it: OK to wait one week to have the new J/ψ's, not-OK to spend one week to read them !
2. is a dedicated effort by an organised group, so the need to divide the task into many parallel jobs, each running on a fraction of the data, and collect the output together is manageable and the resulting effort acceptable

Secondary data sets are re-created from RawData (it does not matter if real RawData, or RawData uncompressed from PADS), writing more or less the same number of events (see above) in possibly more compact format and typically running one ore more of:

- re-tracking (i.e. full blown pattern reconstruction.)
- re-fitting (just refit the list of tracks as input)
- re-clustering (or similar fast reprocessing of some sub-detector)
- selection based on simple cuts

the needed CPU time is very uncertain given the enormous difference in execution time among those tasks. Optimistically we assume that re-tracking is done on the farms "transparently" i.e. smoothly, effortlessly, and with infinite speed, as part of periodical event reprocessing as Production gets better. It is easy to get convinced that re-running full tracking anywhere else then on the farm is not going to work.

Furthermore we assume that the other tasks proceed as "fast AC++", i.e. jobs more or less limited to how fast AC++ can read data (see benchmarking [2]: one needs a lot of user code to impact the CPU needed just for I/O), so O(10Hz) on 100KB/event raw data (1MByte/sec).

10Hz is a bit optimistic, but not unreasonable, should be the proper order of magnitude and not be off by more then a factor 2: we estimated earlier 2 Hz on present fcdfsgi2 for one possible skim job, this is $2 \times 4 = 8$Hz on 1GHZ processor and we expect at least some improvement in data format to speed up I/O.

As 10Hz gives about 10^6 events/day, small data sets can be dealt with in weeks (acceptable?), large ones in months. Running in parallel on fractions of each data set is thus absolutely mandatory for large data sets, but will probably have to be done in general, especially if the 10Hz estimate turns out to be too optimistic.

How many data sets will be re-created at the same time ? There are 100 data sets, assuming each requires a refresh once a month, since one refresh takes 10 days (10^7events at 10Hz), there are 10*100/30 = 33 CPU-days needed each day, so 33 1GHz processors. To avoid being carried away by rounding, notice that the more conservative 6Hz times the "simply correct" 86.400 sec/day gives 20 days per refresh, in this case one needs 66 processors.

Within these numbers (one refresh per month per data set proceeding at 10 Hz) the needed CPU is manageable (e.g. the upgraded fcdfsgi2 could provide 100 CPU's 300MHZ each, i.e. 25 of the 33 processors of the optimistic scenario) and the only issue is the needed bandwidth from/to tape. In this scenario is the bandwidth is about 1MBytes/sec $\times$ 30 processes $\times$ 2 (same amount of data in output as input [2]) $\simeq$ 60MBytes/sec. We assume that Primary Data sets are tape resident and in PAD format (100KB/event) if input were to be raw data (200KB/event), the input bandwidth doubles and the total is $\sim$ 100MBytes/sec. Note that the tape bandwidth is independent of processing time as long as one can put enough CPU to refresh each data set every month.

In conclusion dedicating the entire upgraded fcdfsgi2 to this activity will allow each of the 100 datasets to be refreshed once every 2 3 months, which we call marginal.

7 Computing Needs2: Users Analysis

We look at 4 different estimates of the needed CPU:

optimistic Data sets are large, computers are fast, users are smart. An aggressive opti-

[2]the new Secondary Data sets have to be archived even if targeted as disk-resident

mist [10] may conclude that the CDF community will find a way to look at data at least as fast as the best performance of all approaches tested in the benchmarking phase. There we found data input to need about 1msec/event for both StnTuple analysis using Root [8] and AC++ job reading bbar MonteCarlo. Also there are example of optimised PAD format for jet data that allow reading of several thousand events per seconds [7].

So we take 1msec/event for data input. For user's analysis we go from 0.1msec/event with simple ROOT loops [8] to 1msec/event for TrkAna [2] to 3.6 msec/event for PhiKAna [2].

We average these numbers to an estimate of 3msec i.e. 300Hz, and expect typical real-life cases in this scenario to range from 100Hz to 1KHz. This kind of speed can be obtained when the user's jobs only read a part of the 100KB event (multibranch I/O), some (small?) fraction of the jobs will need to access the full event (hit level information) and will run about 10 times slower.

300Hz allow to process 10^6evens/hour. Assuming each of the 200 users gets 4 hour of CPU each day, he/she can go through 4 million events, adequate for Summer 2002 even in case we write 500×10^6 events rather then 200pb^{-1}. Total need is then about 40 1GHz processors for Summer 2002, and ten times as many for Run2, i.e. on the full Run2a scale each user will have 2 1GHz CPU available for his/her "daily work". This number seems to be sort of a "constant of nature" (e.g. 8 months ago an estimate of the amount of CPU needed for the Italian CDF physicists resulted [9] in 2 CPU's each, in that work these CPU's were assumed to be fcdfsgi ones and some penalising limitation on the number of passes through a data sample had to be imposed). Definitely $1 \sim 2$ CPU per user is is a minimum as users tend to "have one job running" at all times and faster CPUs usually produce less optimised code/behaviour.

extrapolate present performance Basing on the performance on a typical nowadays processors, using present AC++ framework on production output files that contains calorimeter data in compact "PAD like" form, analysis will run at 10 Hz on one processor $\simeq 1M$event/day. The 10Hz is [6] from a $\sim 700MHz$ Linux system, we round to 10Hz on a 1GHz processor. Given the few Mevent sample of summer 2002, one user needs one CPU in order to be done in "days". Total need is then about 200 1GHz processor for summer, and for Run2, to keep the "day" from becoming 10, $1 \sim 2$K processors are needed.

On the Run2 time scale users will routinely need to work by submitting several parallel jobs ($5 \sim 10$) on corresponding fractions of the data set, an awkward and error prone operation that somehow defies the desire (CDF4568) of preserving the "natural way of work". Including as Physics Need the request to go through one data set in a day with one job, the 10Hz are simply unacceptable.

comparison to Run1 From the Benchmark report Appendix H [2] we conclude we need 10 times more CPU per event then typical Run1 jobs, on 20 times the data. Scaling from Run1 system (cdfsga = 24 150MHz processors) with the metric defined in the benchmark document (just count Mips) this yields $24 \times 150 \times 200 = 720 \times 10^3 \simeq 1000 \times 10^3$Mips i.e. 1000 1GHz processors. In principle this should be an overestimate since in Run1 cdfsga was used both for user's analysis and for Secondary Data set creation. In practise the particular example discussed in [2] may not be representative of the average.

BaBar Simply compare to the amount of resources needed for the user's analysis in the BaBar experiment. BaBar is now in a situation similar to where CDF will be one year from now: physics data steadily coming in, hundreds of people looking at data. While event reconstruction is likely very different (finding tracks at Y4S vs ppbar), the user analysis on the production output is likely very similar (cut on energies, loop on tracks, vertexing, invariant mass calculation...). If anything, CDF physicists will have to deal with a larger number of physics objects (more tracks, jets) while still doing the same physics. The big difference is in the cross section: BaBar (4nb $\times$ 30E-6 nb^{-1} $\simeq 10^8$ events) collects in one year about 1/10th of the CDF Run2 sample. BaBar computing is performed on a O(2000) processor farm at SLAC, about 20% of this ($\sim$ 400 processors) is for user's analysis [3]. Scaling to the larger CDF data set, one can estimate 4K processors needed at CDF.

The conclusion is the one reported at the beginning of this document, if user's analysis becomes at least a factor 10 faster then it is now, several hundreds of processors will be needed (overall one looks at a factor 10 increase in CPU power with respect to the upgraded fcdfsgi2). Otherwise the needed increase in number of processors is 100 and we look at an analysis farm of a few thousand nodes.

8 Disk space

Disk needs for skim is only what is needed for the staging in/out of data, skim does not dictate whether output or input data sets should be disk resident, and since there are relatively few skim jobs, even if all of them miss the disk cache and hit the tape, the overall bandwidth should not be a concern. We saw before that even assuming that one has to go back to raw data on tape at all times, the overall bandwidth is 100MBytes/sec. To be compared e.g. with the design specification for the original CAF design of 150MBytes/sec [13]sect.3.4.

To define the needs for analysis, we look at how much disk space is needed to keep the CPU running. We do not have as a goal to have all events on disk, but we'll find out that we could/should keep almost all of secondary data sets on disk. There are good possibilities to do this on data sets of the size expected for Summer 2002 with 10TB of disks [10]. Going to full Run2a this means 100TB of disk (i.e. the overall Secondary Data set volume assumed in this study), too close to the physical (and budgetary) limit of what is possible for simply planning on it. So here we will use a different approach: assume data stay on tape and are cached on disk "as needed", and find the order of magnitude for the cache. The disk cache may very well be organised with some data set fully disk resident permanently and some forced to more frequent migration, and for practical and political reasons this may be more convenient then dynamically managing all disk space, while slightly less efficient.

Either 400 or 4000 CPU, is only a matter of architecture, i.e. of connecting one or 10 CPUs to a given disk to allow one user to run 10 jobs in parallel. Since the goal is to allow each data set to be traversed in days (lets say 2 days), the bandwidth disk→cpu is the same. This is: 200 users*1TB/#days = 200/2 = 100TB/day, i.e. $100 \times 10^6/10^5$ = 1000MBytes/sec. To be compared with previous estimate of 1.5GBytes/sec [13]sect.3.3, that estimate included also the skim jobs etc. To keep all numbers round, we stay with 1GBytes/sec, so have to provide, in average, 5MB/sec bandwidth into the CPU for each of the 200 users.

In principle one could run with "no disk" (even the fastest analysis we considered runs at 300Hz times 100KB = 30MBytes/sec, i.e. within what the mass storage can deliver to a single processor) and retrieve the data from tape as needed. In practice a large disk cache is needed to reduce the overall bandwidth from tape. We size the disk cache trying to keep the latter to 100MBytes/sec, this allow for being off by a small factor, but is the right order of magnitude. At first approximation, we have to beat down the 1GBytes/sec by an order of magnitude, i.e. 90% of the requests must hit the disk cache. But ...
First of all not all the 200 users will look at different data sets, there are only 100 data sets all in all. Then, the "include dataset" feature of the DH input module will generate a "D0 freight train" effect so that additional users accessing a data set already being used will use files already on disk first. This means that we can calculate as if there was only one user per data set, the others affect the disk→cpu bandwidth not the tape→disk one [3]. Of course if the 200 users divide up evenly in every two people requesting a different data set at the same time, there is little to gain, and the system may just grind to halt! Some optimism is needed here and we assume on average only 1/2 data set being used at any time, i.e. in average there are 4 users on each data set, or in yet other words, there are 50 freight trains of 4 users each. Remember that this is average, is fine to have 7 people running on the top sample vs one accessing the high E_t photon. Then we look at the needs of the 50 freight trains as if they were 50 users needing data at 5MB/sec each, for a total of 250MBytes/sec that we round to 300MBytes/sec for a bit of safety. The needed bandwidth from tape to disk is then this 300MBytes/sec times the fraction of cache misses, which is the fraction of data not on disk. To keep this fraction at 30% (so that $300 \times 30\% \simeq$ 100MByte/sec) we need 70TB of disk on the full Run2a time scale. Not an impossible task. Here is where factors 2 make a big difference, reducing PAD size to 50KB would mean 35TB of disk, almost an "easy" number.

Anyhow the situation is not dramatic, keeping fixed 100MBytes/sec tape→disk overall and reducing the disk cache to 50TB means 50% of the data will have to be accessed at tape speed (100MBytes/sec) rather then desired "disk speed" (300MBytes/sec) so total processing times increase by a factor 4. (2 days becomes one week, a "soft failure").

Tuning the tape bandwidth, the disk cache size, and defining some data set priorities, should make it possible to keep the "few days" target for the majority of the users. Somehow this may even happen automatically as heavily requested files will always be on disk, the

[3]This reasoning implies a single common disk cache to the CAF, which may not be true, hopefully it will hold for orthogonal sub-sets of the full data sample, each having its own disk cache.

key is to keep the disk cache at least as large as $\sim 50\%$ of the total secondary data sample. Just to avoid confusion, the 100MBytes/sec tape$\rightarrow$disk talked about here is in addition to the 100MBytes/sec estimated for the skim at the beginning of this section.

In conclusion we foresee the need for a disk cache of $50 \sim 70$TB for secondary data sets. Good event compression will help a lot here, but it is important to preserve the capability to read in the analysis job only a fraction of the event data in a very efficient manner.

At the same time it appears like the total required bandwidth from tape to disks for skim and analysis (i.e. besides the farms) may easily reach a few hundreds of Mbytes/sec, and there is no guarantee that it will not be several.

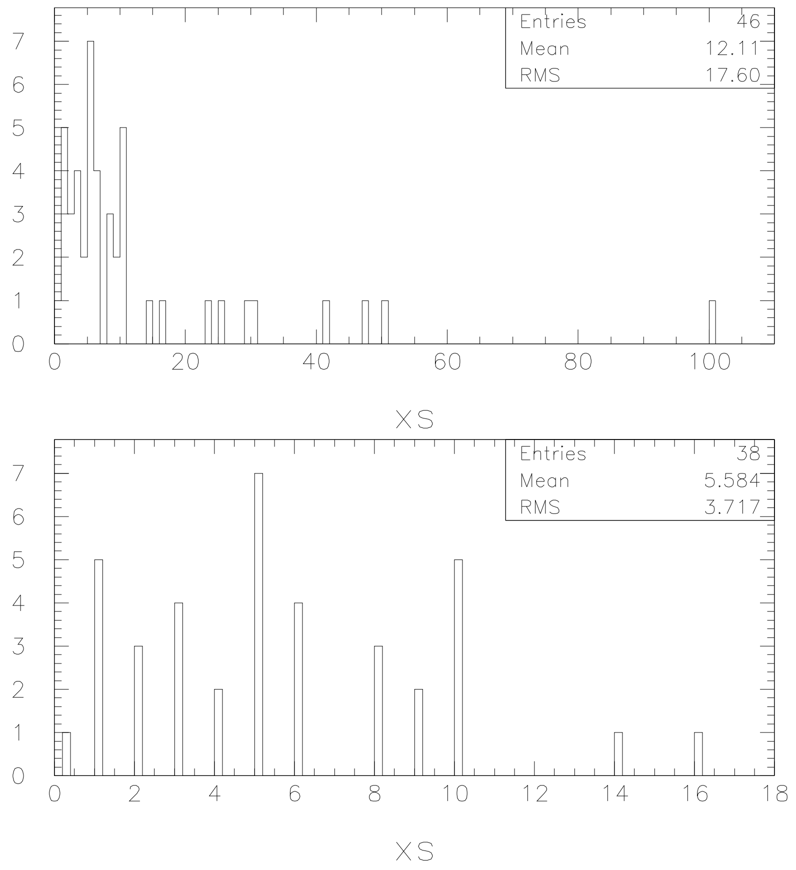

Figure 1: distribution of L3 cross sections for the 46 Data Sets defined in CDF4718 as of November 2001. Horizontal axis is the L3 cross section in nb, vertical is number of data sets. The bottom plot is a blow up of the region ($\sigma \leq 20$nb)

References

[1] Final Report CDF CAF Review Fall 2001; CDF note 5802

[2] Report on Benchmarking, CDF CAF Review Fall 2001; CDF note 5743

[3] Charlie Young, private communication, 25 October, 2001.

[4] CAF review Workshop, Fermilab, 18-19 October 2001.
http://www-cdf.fnal.gov/upgrades/computing/projects/central/workshop/index.html

[5] Top/EWK and QCD Group Perspective, talk by Pierre Savard at the CAF workshop
http://www-cdf.fnal.gov/upgrades/computing/projects/central/workshop/talks/savard.ps

[6] Pierre Savard's laptop

[7] Robert Harris and Pierre Savard, private communication, October 2001. This format has been setup by Pierre Savard and is currently used e.g. for di-jet balancing studies.

[8] Pas-ha Murat, private communication to the CAF committee, September 2001.

[9] The ICRB web page has details about estimating computing hardware needs for a group of 20 Italian users
http://www-cdf.fnal.gov/internal/upgrades/computing/icrb/ana_italy.html and
http://www-cdf.fnal.gov/internal/upgrades/computing/icrb/italian_FCC_ana.txt

[10] What is needed/possible, talk by Stefano Belforte at the CAF workshop
http://www-cdf.fnal.gov/upgrades/computing/projects/central/workshop/talks/belforte.pdf

[11] Beate Heinemann, private communication, 19 November 2001

[12] Exotics Analysis Disk Space, talk by Ray Culberston at the Joint Physic Groups meeting, 8 November, 2001
http://www-cdf.fnal.gov/internal/physics/joint_physics/docs/ray_081101.ps.gz

[13] CDF Run II Data Handling Central Analysis System Hardware Architecture, W.Brown, L.Buckley-Geer, S.Lammel, T.Watts, CDF note 4707