

# 1 Processing Farm

The CDF Production Farm provides the experiment with production-level reconstruction processing of all raw data using a cluster of about 200 dual Pentium nodes, or about 700 GHz of CPU at present. The farm fulfills two goals: to provide fully reconstructed data using a standard executable and calibrations within a few days of data taking, and to re-process some or all of the data with updated executables or calibrations as dictated by the needs of the physics program.

In this section, we first describe the transition of farm architecture from a central cluster to a SAM based facility. The upgrade of the farm system is intended to better optimize resource utilization and increase the flexibility of the processing model. It is using SAM to track file metadata and to assist in managing job submission in an optimal way.

## 1.1 The FBS Farm Architecture

The CDF production farm is a collection of dual CPU PCs running Linux, interconnected gigabit ethernet. The hardware architectures are cost-effective. The quantities and types of worker nodes currently in the farm are listed in Table 1.

The challenge in building and operating the production farm is in managing the large flow of data through the computing units. The first production farm constructed for Run II data process is a centralized cluster having direct access to the Enstore tape library. The headnode server, `cdffarm0` (a Dell 6650 server), provides the core FBS batch system and a MySQL database used for job and file tracking. It runs control daemons for resource management and job control for each data stream. The disk space within the farm, which is a collection of IDE drives on all worker nodes, is virtualized through a “dfarm” file system hosted on `cdffarm0`. An additional server machine, `fnpcc`, hosts a java server and a web server that run the user interface and other monitoring operations, as well as interactive services for farm operators.

In this centralized farm system, 16 workers are configured as I/O nodes. Eight of I/O nodes stage data from tape and distributes input files to the processing nodes. The output nodes collect files from processing nodes, concatenates them as needed into files with a size appropriate for tape storage and then write them to tape. The I/O node configuration includes a `pnfs` filesystem that provides direct access to the Enstore data archive. All I/O nodes are 2.6 GHz dual Intel P4 machines and

Year	Numbers	Type	P3 equivalent GHz
2002	32	P3/1.26 duals	81
2003	64	P4/2.6 duals	253*
2003	80	P4/3.0 duals	356*

Table 1: Past production farm procurements. These are the nodes currently in use. (\* scaled by 1.35 to Intel P3 equivalent).

utilize optical Gigabit network connections in order to saturate the I/O channel to tape. All the data processing workers are dedicated to running the reconstruction programs, and are not to be accessed by any other purpose.

The control software is a complete chain of data processing from the retrieval of data from the Enstore tape archive to the storage of reconstructed data in the tape archive. Farm processes are defined by the specified input dataset. The workflow for each dataset is handled by a “farmlet” that has a series of daemons that perform data and job management functions, and that track file status and job history through the MySQL database.

Processing jobs are dispatched in units of a “file-set”, a `pnfs` sub-directory of 10 files, each with a typical size of 1 GB. The input I/O nodes copy data from tape to a local scratch area, then into `dfarm`. The staged raw data files are first dispatched to workers running the reconstruction executable; the output files are written into `dfarm`. A raw data file can have multiple output files each containing data that satisfy different sets of online triggers. The output file sizes, consequently, vary from about 20 MB to 1 GB. The output I/O nodes, or “concatenators”, collect the products of each output dataset in `dfarm` and concatenate the files into the final output files. The history information in the database is used to ensure that all events in an output file form a contiguous time period during data taking. Exceptions are allowed for files that fail to process due to multiple abnormal terminations.

All farmlets work independently and share all farm resources on a first-come-first-served basis. This scheme can lead to unintended interferences, particularly with regard to tape drive utilization. Each of the farmlets, for instance, can access only a single tape drive for input data staging, and one for each output dataset. The Enstore queuing system does not distinguish between input datasets, however, it allow a single farmlet to create a large backlog in the dataset request queue, thereby blocking access to tape by other farmlets despite the availability of open tape drives. While not an issue for raw data processing, this feature can seriously limit throughput for data re-processing, where the input data volume is large, without occasionally substantial human intervention.

Experience with this system has revealed a number of deficiencies in the architecture of the processing software. Error may occur in any instant for having a few thousands of files being processed and created. The difficulty in operation is mainly caused by the rigid control on file tracking, disregards the fact that file may be lost due to crashes in reconstruction code and worker hardware failure.

- The control over resource allocation is not flexible to allow full utilization of CPU and I/O resources, particularly under heavy re-processing loads.
- The monitoring tools is not sufficient to reveal clogged processes due to program crasher or hardware failure. It requires expert checkup to realize presence of problems.
- The system design lacks error recovery protocols and is therefore not robust against many common errors. Job failures or re-submissions typically require a significant human effort in correcting MySQL records in order to restore

the state of job and file tracking. This issue is compounded by infrastructure failures that result from the poor scaling properties of the system under heavy loads.

- The control software is heavily chained tip to toe. The lack of modular service management framework makes it difficult to alter the processing model in response to changes in requirements.
- The demands upon the farm are episodic, driven by occasional periods of re-processing, when the load is extremely high, or machine shutdowns when no raw data is logged and the load is essentially zero. A significant fraction of the farm remains idle over the course of a year.
- The uniqueness of the farm architecture leads to scaling issues and significant under-utilization of the farm resources in some circumstances, and may limit further expansion of the farm.

As a result of these and similar issues, maximizing the throughput on the farm has required adapting the characteristics of the input data stream to the limitations of the farm rather than having the farm processing model evolve with the needs of the experiment. A current requirement for re-processing, for instance, is that the data be split and concatenated in advance. A change in the splitting scheme is not easily handled unless the new split is a strict subset of the existing split. There is no adaptation currently available that can allow more uniform utilization of farm CPU cycles.

In order to address these issues, we have undertaken a complete re-design and upgrade of the farm architecture to a CAF facility and the farm control to the SAM based data handling system. The goals of the upgrade are to improve the resource management capabilities of the system, provide robust error recovery and to migrate to an infrastructure that will allow production jobs to be inter-operable on other platforms and the production farm to be a computing element within a GRID computing model for CDF. The CAF infrastructure will in principle add the farm to the pool of computing available to the experiment for user analysis. We have demonstrated that the farm jobs are inter-operable on remote systems. Jobs were submitted to remote CAFs in Japan and Taiwan. After the production jobs were demonstrated to operate on a test SAM farm. We have converted most of the workers to the new system.

## 1.2 Upgrade to a SAM-Based Farm

The new production system is a SAM-based system with data handling and file metadata services provided by SAM. The daemons of the FBS farm that manage job status and work-flow are replaced by SAM projects and batch submission tools. Resource management is replaced by a cron job probing utilization of resource and services. Job submission is a periodic cron job checking on the resource status and is prohibited if any of the dependence is not sufficient. Input datasets for projects are

defined prior to submission using highly flexible database queries that can be applied to any cataloged data. Error recovery is simplified to job resubmission. False starts can simply be re-submitted as new projects.

The new farm system is developed on the CAF platform using Condor batch system. The production jobs on the FBS farm require little if any CDF-specific services, therefore the SAM farm is tailored with little need to rely on CAF or CDF-specific features, a fact that should simplify the task of GRID enabling production jobs. The farm control software is modularized and is single threaded to resource management and services in each step from preparation of input dataset to final storage. Each step is independent from the proceeding process, therefore it prevents the lengthy and labor intensive cleanup required to the FBS farm in restoring the farmlet daemons and MySQL records to an appropriate initial state.

There are several important benefits to the SAM farm upgrade:

- The CAF becomes the platform for testing of farm software and GRID development, thereby allowing these activities to proceed in parallel with on-going farm activities.
- The farm becomes available for opportunistic use by jobs submitted to the CAF, which will improve the utilization of farm resources.
- The virtual boundary established between the CAF and the farm allows the experiment to dynamically increase the size of the farm into the CAF should the need for re-processing exceed the capacity of the farm.
- Since SAM is a common tool across all Run II experiments, the farm system will benefit from efforts to extend SAM to the GRID.
- Since the CAF is ubiquitous across CDF, efforts to migrate either general CAF or production processing to the GRID will benefit the other.

The upgrade outlined above provides a flexible architecture in which to conduct farm operations. As importantly, the services provided by the CAF are essentially the same as those needed on the GRID. A proper structuring of the production system on the CAF infrastructure can therefore offer a number of simple development pathways by which the system can evolve to operate in a GRID environment.

### 1.2.1 SAM farm architecture

The SAM data handling system is organized around a set of servers communicating via CORBA to store and retrieve files and associated metadata. File information is stored in the SAM database as file metadata. A task for processing many files is launched as a SAM project. A project is organized for a user dataset, with a consumer process established to receive data files. File delivery is coordinated such that the events are read only once to all the analysis programs of the project.

Illustrated in Fig. 1 is the hardware architecture and applications for data production with SAM. With the input provided by SAM, disk space is only required for

output on durable cache, before concatenation and afterwards to be stored to SAM. The communication with SAM database is conducted by the farm servers configured as SAM stations. The CAF and durable storage are entities easily specified in the job submission, therefore it is flexible to use any facility accessible. To improve bandwidth and file usage, the SAM production farm is configured for direct access to the dCache file system where input files are located. Concatenated output files are transferred directly to Enstore.

Job submission is controlled by applications scheduled on a SAM station. The usage of file metadata is generalized for bookkeeping purpose. The tasks preparing input datasets and data processing in a CAF worker node are illustrated in Fig. 2. The tasks are:

- Preparation of input datasets :**  
 input data to be processed are selected by queries to online DFC records for data quality (good-run) and detector calibration. The input datasets are organized in run sequence of more than 20 files of one or multiple runs for a raw data stream.
- Start of SAM project, CAF submission :**  
 a SAM project is started for a dataset not fully consumed. It is submitted to a CAF. SAM establishes a consumer process to deliver files to CAF workers. From the CAF headnode, workers receive an archived (tar) file containing

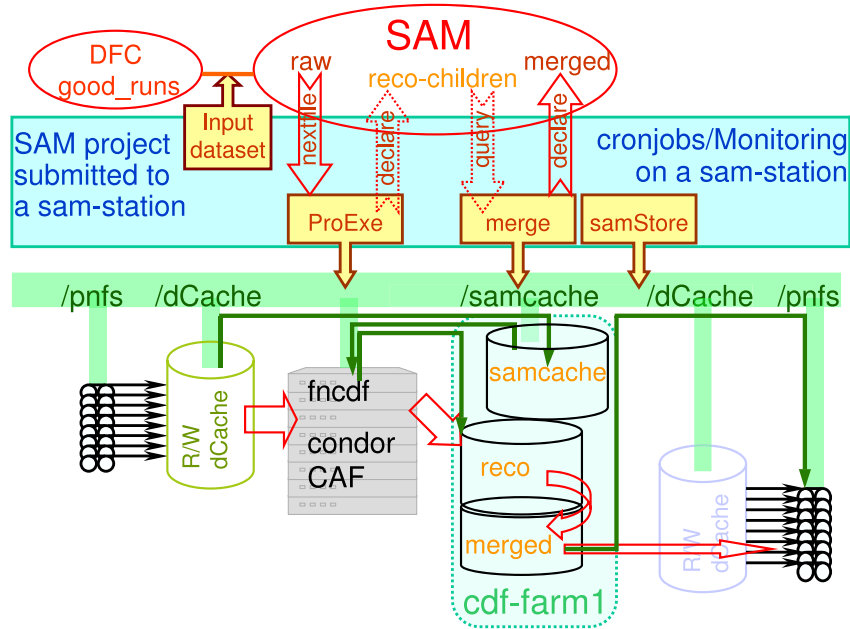


Figure 1: Data flow and job control of a SAM farm. Data are transported by SAM to a file cache accessible to the Condor CAF. Output is sent to a durable storage where concatenation is executed. Merged outputs are declared to SAM and stored to Enstore.

program binary, library and control TCL cards. Input files are copied to the local scratch area. Files are delivered according to the file consumption status, till all files are delivered. Output of the program are then copied to dedicated durable storage nodes, and the associated metadata are declared to SAM.

The dataset preparation and job submission are all issued periodically by cron jobs. A project monitoring graph on the consumption of data files are plotted in Fig. 3 To prevent exhaustion of computing recourses, permission is required by a monitoring template recording the latest resource status.

In comparison with the FBS farm, the SAM farm management is attending

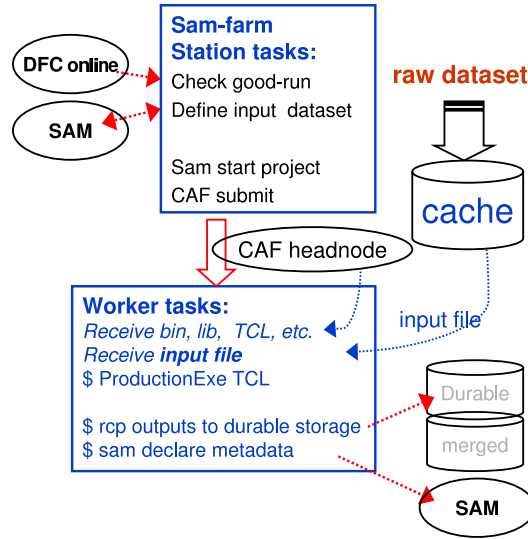


Figure 2: Task flow for a SAM project submitted to a CAF worker. A worker node receives the executable tarball, copies input data file, after processing outputs are copied to durable storage with metadata registered to SAM.

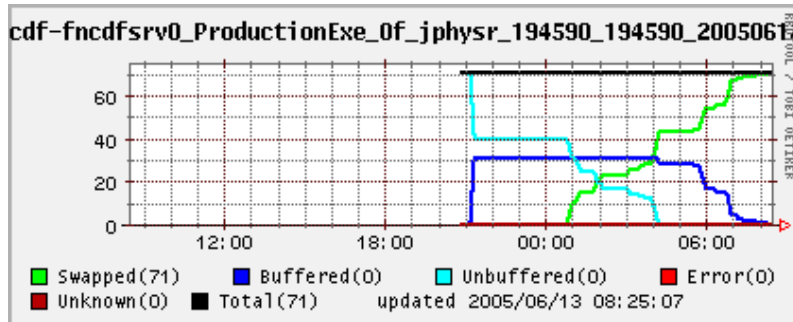


Figure 3: Consumption of files by a SAM project is plotted. The total of 71 files in a dataset were requested and quickly "buffered" to CAF workers. The CAF job is configured to use 30 CPU segments. After approximately 4 hours, consumed files are being "swapped". The project is terminated after all files are swapped.

processes of datasets. Tracking for individual file is taken care by the SAM consumer process. The operation is therefore reduced to detect incomplete projects and debug. The bookkeeping tasks is reduced from tracking thousands of files in an instance to a few dozens of projects. The monitoring is concentrated on the usage of durable storage, where outputs from CAF are checked and merged in the concatenation process.

### 1.2.2 Durable storage

Output of CAF jobs are transported to durable storages on 2 TByte file servers. A concatenation job is launched upon a threshold of total number of files, these files are then merged into output of size close to 1 GByte. Previously in the FBS farm, the output of concatenation is truncated into 1 GByte files of exact size, therefore an input file can be split into two concatenated files. This algorithm is changed to be flexible on the output file size to include a complete set of files. This change has the advantage on parentage of merged files for the unique correspondence of input and output. The concatenation procedure conducted on the durable storage node is illustrated in Fig. 4. The details are described in the following:

- **Durable cache :**

a durable cache is a directory on a large file server where CAF output of the same dataset are stored. In total 41 directories are used for all reconstructed datasets. The files are buffer to a threshold (for example 100 files). A cron job sort them into lists of files in sequence of data taking period. File size of a list is within the desired concatenation range. And the control TCL read by the executable (AC++Dump) is prepared to include these files.

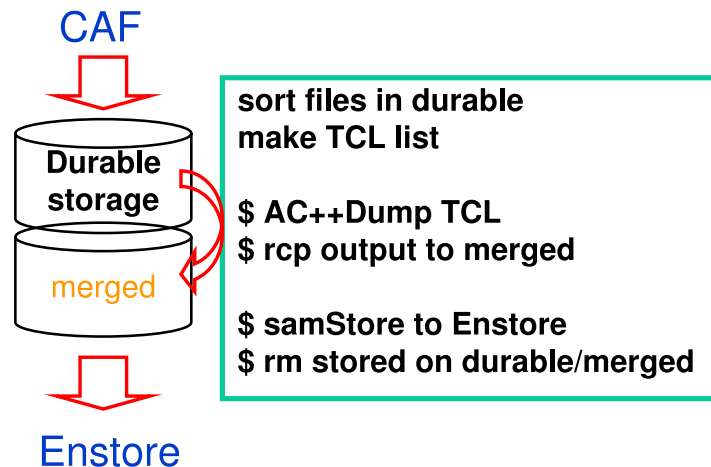


Figure 4: Files in a durable cache are sorted into lists in TCL cards read by the concatenation binary (AC++Dump). The merged files (of size close to 1 GB) are stored to SAM.

- **Concatenation :**  
concatenation is running on the file server where the durable storage resides. Output is transported to the "merged" directory ready to be stored to SAM.
- **SAM store :**  
merged files are scanned periodically for a threshold (for example 10 GByte) and the SAM store is conducted to copy files to Enstore and declare metadata. The threshold size is used to reduce Enstore operation cycles.

The executable doing concatenation job is just moving blocks on disks, therefore we chose to do it locally on the file server to prevent network hurdle and additional management load. The CPU time is roughly 3 minutes per GByte on a P3 2.6 GHz file server with 7200 rpm IDE hard drives. For all the operations on a durable storage, this is the slowest process and potentially the bottle neck that may be improve by newer file servers equipped with more and faster CPU's. The network giga-link speed is commonly running at 20 MByte/sec and the Enstore logging rate by a single mover can accomplish more than 1 TByte a day.

The new system is more tolerable to errors. We impose parentage in metadata listing input raw data parents and output children. With a SAM query we find files not yet processed. If a merged output should be reprocessed, we query its parents for preparation of recovery.

### 1.3 Scalability

The FBS farm uses dfarm file system which is a collection of IDE hard disks on the workers. With a total 200 workers, the chance of losing file is high due to hardware failure. The load on the MySQL database corresponds to thousands of queries in an instance, which requires a significant CPU power on a server. The architecture of the FBS farm is constraint to the data handling with direct access to Enstore. This had prevented usage other than the dedicated production operation.

The SAM production farm exploits the advantage of the data handling system with usage of file metadata for bookkeeping. The usage of durable cache for concatenation is a specific procedure for CDF data production. The software contains applications of SAM clients, therefore the constrain on hardware facilities is minimum. The CPU facility employed can be any of CAF facility. The durable storage file servers can also be located anywhere accessed by the CDF data handling system. With jobs submitted to CAF facilities in Japan and Taiwan, we were able to accomplish a few MByte/sec in bandwidth. The dedicated SAM production farm is constructed in the spring 2005 at Fermilab. It has gigabit network links with a CAF of 70 workers and four file servers each having 2 TB durable storage space. The data input is configured for direct copy from dCache read pool and SAM store to Enstore. Each file server can provide a 0.5 TByte throughput with two concatenation jobs running. This system has accomplished a stable operation for CDF data collected in 2005. Currently the SAM farm network is limited to two Gigabit links, and is CPU bounded.

## 1.4 Farm Capacity

Processing tasks on the production farm falls into three main categories: raw data processing during normal data taking, data re-processing and specialized tasks. The impact of each must be considered when discussing the capacity of the production farm.

During periods of normal data taking, the policy of the experiment is to process the raw data within three days of data taking. This requirement stems in part from the need to provide rapid feedback to the detector operations group for the purpose of monitoring data quality. Due to various latencies in providing calibrations and other routine operational delays, the farm must have sufficient capacity to process raw data at an average rate that significantly exceeds the peak logging rate of the experiment. We calculate the farm capacity required to keep up with data taking by multiplying the peak logging rate of the experiment by the CPU per event. A contingency factor of 50% is included in the CPU per event in order to allow for increases in processing time with luminosity or more complex versions of the executable. We further assume an 75% utilization efficiency for the farm (which includes routine operational delays).

Data re-processing is required from time to time because the quality of the reconstruction programs improves with time. Periodic re-processing allows the entire dataset or some fraction thereof to benefit from these improvements. The problem of re-processing differs from that of raw data taking in that the rate of data input to the farm is limited only by the I/O capacity to the tape archive, and far exceeds the rate at which the detector can generate data. During Run I, CDF re-processed each year about 30% of the available data toward the later portions of the run, with somewhat higher fractions near the beginning. For the current estimates, we assume a re-processing fraction of 30% in FY-05, and 20% for all subsequent years. The drop is justified based upon improved reconstruction and production management as the run progresses.

In previous years, we have estimated the total required farm CPU by assuming that some fraction of the entire dataset must be re-processed over the same period during which the raw data is logged. The same contingency and utilization factors are assumed for the re-processing. As we will discuss later, this model is not well matched to actual experience and is one topic that we hope to address with the production farm upgrade. We nonetheless use this model for the current budget estimates because it builds in some reserve capacity in the farm is in fact required in order to accommodate data re-processing, and because the cost of the farm in the end is relatively modest.

The final set of tasks performed by the farm include such items as the beam axis calculation and other calibration constants required by the reconstruction programs. Other tasks may include the processing of datasets for special purposes as requested by the physics groups that are too large to be efficiency conducted without the automation provided by the farm. The load from these sources is negligible and are typically easily accommodated without explicitly including them in the calculation.

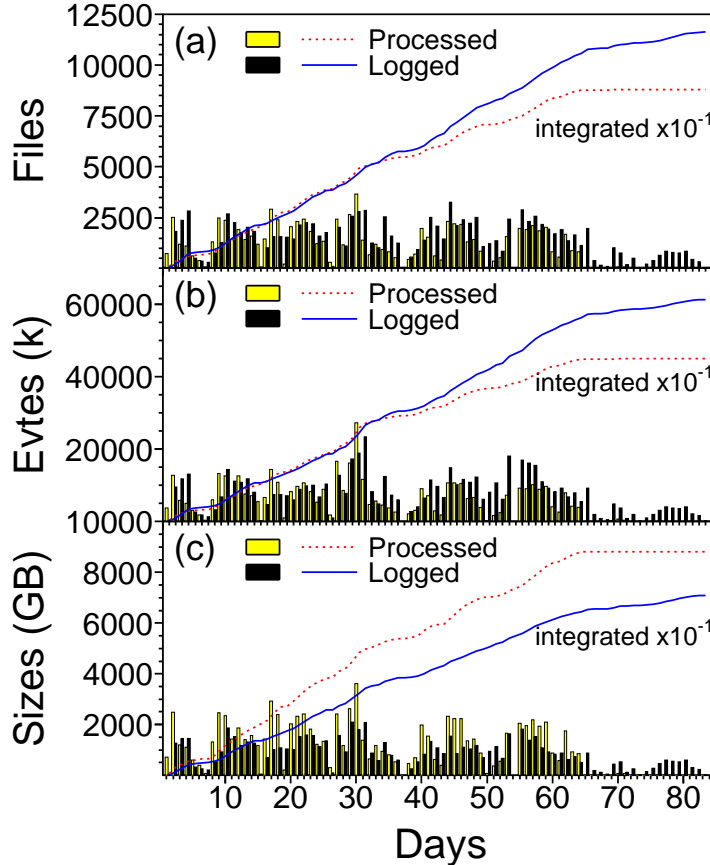


Figure 5: Daily processing rates are shown in histograms for (a) number of files, (b) number of events, and (c) data size. The integrated rates are shown in lines. Compressed outputs were created for selected data-sets (about a quarter of the total). Event size is reduced by about 30% and thus a net reduction in output storage.

Figure 5 shows the production farm capacity illustrated for the data re-processing conducted during winter 2004. On average the daily processing speed is about 10 M events, corresponding to 2 TByte through-put in data logging.

## 1.5 Farm Procurement Plan

Using the model outlined in the previous section, we estimate the total required capacity of the farm as a function of time. The results are shown in Table 2. To estimate the cost, we assume the purchase of Intel P4 equivalent dual processor machines at a unit cost of \$2.2k. The processor speed is assumed to double every 18 months, starting with a speed of 2.5 GHz P3 equivalent in FY-04. (The FY-04 speed includes the observed 10% increase in processor speed over the machines available in FY-03.) We then constrain purchases within the farm to increments of two racks, where we have assumed 40 U of usable rack space (the current standard for new purchases) and 1 U nodes. The final procurement plan is tabulated in Table 2 for

the next three fiscal years. An increase in the event logging rate in FY-05 drives an increase in the required CPU for the farm. The decreased re-processing fraction in FY-06 compensates for a much smaller relative increase in the logging rate in FY-06. The larger data volume then drives the requirement up again in FY-07.

FY	Need (THz)	New CPU (#)	Retire CPU (#)	CPU Speed (GHz)	Total (THz)	Total Cost (\$M)
03	480	2* 64	2* 73	2.2	525	0.19
04	1100	2* 80	2* 64	3.0	1100	0.24
05	1400	2* 80	2* 64	3.9	1500	0.18
06	1200	0	2* 64	6.2	1300	0
07	2600	2* 80	2* 64	9.9	2600	0.18

Table 2: Production farm procurement. Numbers in FY-03 to FY-04 are actual and FY-05 to FY-07 are estimates.