Computing

F. Bianchi
Torino

VI SuperB Collaboration Meeting
Frascati, December 12th, 2012

Outline

- Status of SuperB computing effort.
- A computing model for a tau-charm factory.
- Outlook.

Status of SuperB Computing Effort

Computing Chapter of Detector TDR

- Text is completed.
  - Reviewed by E. Luppi.

- Many thanks to all the people who contributed text.

- A special thank to S. Luitz for all is hard work in improving the content and the wording of this chapter.

Production System Upgrade (1)

- Book-keeping database (*sbk5*) modified to implement new production system features on FullSim session:
  - **Request**: new request and production request definition [TESTED]
  - **Jobs**: new fields and constraints to implement functionalities to re-submit failed job in bulk mode (**job re-submission**) [ONGOING]

- Job Wrapper (Severus):
  - **Submission**: changes to implement **job re-submission** functionality [ONGOING]
  - **Stage-in**: new fail-over mechanism for the stage-in phase implemented [TO BE TESTED]

Production System Upgrade (2)

- Web-UI code (PHP, JavaScript, Smarty) modified to handle new features and *sbk5* updates on FullSim session:
  - **Production:** new production creation interface with a robust software version and parameters management [TESTED]
  - **Production Requests:** possibility to create different production requests having the same physical parameters by changing the number of events per job [TESTED], new input mode for *job re-submission* [ONGOING]
  - **Expert Submission Interface:** "Job Details" form section updated keeping previous functionality for production initialization [TESTED]
  - **Shift Submission Interface:** minor changes [TESTED]
  - **Job Monitor:** minor changes [TESTED]
  - **Submission Monitor:** minor changes [TESTED]

Data Access Library

- R&D work for the development of a software library with an optimized data access management
- Features:
  - intelligent pre-fetching and buffering algorithms
  - logical file name map with different physical storage URL
  - possibility of supporting storage protocols not supported by ROOT
  - read-head buffer and caching mechanism in order to solve the overhead problem

G. Donvito, G. Marzulli, S. Pardi

Distributed HadoopFS

- INFN-Bari has developed and tested a new policy in order to use Hadoop file-system for an automatic data replication among different site in a Wide Area Network environment
- The test were performed successfully between Bari and Napoli
- We want to go on testing and stressing this solution to understand if it fits the requirements of a production usage

D. Diacono, G.Donvito, A. Fella, P. Franchino, E. Manoni, S. Pardi

Data access test

- Test goals:
  - measure the latency period due to the increase number of parallel read stream
  - measure the latency period due to the increase of round trip time elapsed between source and destination
  - support the development of a general, experiment wide, data access software layer
  - start the characterization of a concrete WAN scenario, including traffic impact, typical latency, network resource overloading
- Test layout definition:
  - 1, 5, 10, 50 and 100 parallel set of read streams
  - each stream reads a random files according to a trace file obtained from an analysis application
  - 250 compressed root files, 476 MB each
  - sources: INFN-T1 and INFN-Bari
  - destinations: INFN-T1, INFN-Napoli, GRIF and FNAL
  - measured the time of the cURL execution

DIRAC evaluation

- DIRAC: framework to manage and use a distributed computing infrastructure
  - Grid, cloud, Boinc, local farm, desktop computing
  - User mangement, grid certifcate, VOMS, workload and data management, FTS transfer, 2 File Catalog (LFC and DFC), monitoring, accounting, workflow
  - Pilot job paradigm
  - Largely adopted (not only) in HEP community
    - LHCb, BelleII, BESIII, ILC, CTA, etc.
- Supported and developed by
    - DIRAC developers team
    - DIRAC community

DIRAC for SuperB

- Manage both EGI and OSG sites
- Single administration point – minimize human effort
- Mass data transfer successfully tested
- Simulation Production successfully tested
  - Stagein, Stageout, Severus job wrapper
- Job priority policy defined by VO manager
- Web interface for every user type (physicist, shifter, manager, admin, etc.)
- Possibility to use Cloud Resources (Amazon, OpenStack, occi-compatible cloud)

SuperB DIRAC

- Extending DIRAC for SuperB needs
- Interaction between DIRAC and bookkeeping database (SBK5)
  - Load, add and modify
- Severus job wrapper porting in DIRAC
  - Use DIRAC capabilities where possibile to benefit of DIRAC advanced features
    - Stagein, stageout, software setup, SBK5 interactions
- Simulation Production
  - Porting WebUI functionalities in DIRAC
    - Site management
    - Session management
    - Job submission
    - Output management

SuperB Dirac project credits

- Marcin Chrzaszcz – Kracow
- Giacinto Donvito - Bari
- Armando Fella – Pisa
- Rafał Grzymkowski – Kracow
- Bruno Santeramo – Bari
- Miłosz Zdybał - Kracow

- Thanks to DIRAC developers, expecially to: Andrei Tsaregorodtsev, Federico Stagni, Matvey Sapunov, Krzysztof Daniel Ciba, Ricardo Graciani, Adrian Casajus

V. Ciaschini, M. Corvo, F. Giacomini, A. Gianelli, S. Longo, R. Stroili

Parallel Computing R&D Activities (1)

- We implemented a prototype, based on the BaBar FastSim framework, to exploit parallelism inside current analysis
- Using the Intel TBB flow-graph object we can realize parallelism not only at event but also at module level
  - This also give us the possibility for an algorithm level parallelism that can be explored in the future
- Measurements done on the prototype demostrate that:
  - the model can be used to reduce the total memory footprint
  - the scheduling schema may be employed to efficiently use systems with large number of cores

Parallel Computing R&D Activities (2)

- Some limitations for parallelism have been found in the current framework code:
  - use of Common Blocks in Fortran code
  - widespread usage of static objects
  - some module are not really OOP-compliant, in particular for what concern encapsulation
  - auxiliary data structures (e.g. Event) don't allow concurrent data access
- With the accumulated experience we are now ready:
  - to formalize specifications for analysis modules
  - to show an initial proposal design of a natively parallel architecture framework for experiment analysis

Bruno Multi-Thread (1)

- Full simulation software is not exploiting at all the many parallelization possibilities offered by modern computing
  - Not only SuperB, but the vast majority of existing HEP simulations run sequential single-thread programs
- Main limitation in the past has been that the main simulation toolkit (Geant4) was not designed to be run in any non-sequential mode
- Recently, things have changed, with the release of a prototype of G4 suitable for multi-thread applications
  - Result of several years of development, now ready for usage by "experts", even though most likely not yet for official deployment
- SuperB was in the very interesting situation of being a new project, with a brand new simulation software
  - The decision was taken to experiment with the new G4 prototype and adapt Bruno to use it

Bruno Multi-Thread (2)

- Geant4 MT is a variant of the stock Geant4 distribution, whose general architecture has been modified to allow event-by-event parallelism
- A master thread initializes the geometry and the "shareable" parts of the physics
  - Then it launches *n worker threads*
    - Within the same physical computing element
- Generated events are dispatched to the worker threads for processing
  - Every event is fully simulated by one thread
  - Every thread simulates only one event at a time
- Equivalent to splitting the generated events into subsamples and processing them through n independent processes
- Multi-thread saves most of the initialization time, and vastly reduces the memory footprint, as threads share the same memory

Bruno Multi-Thread (3)

- Migrating a simple application to Geant4-MT is just a matter of compiling against the new G4 and modifying the client code in a few well identified places
  - Procedure is well documented

- Unfortunately, Bruno is not a simple application, and no existing migration how-to could be effectively used

- Moreover, Bruno was not thought to be run in parallel, and some parts of its code had to be adapted/rewritten

Bruno Multi-Thread (4)

- Now we have a running, fully featured simulation, with the same functionalities as the existing Bruno release
  - This means that, as far as G4 is concerned, the migration is completed
- What we still miss is the persistency, i.e. the ability to write to file hits and MC truth
  - This requires dealing with ROOT, not with G4
  - Some parts of ROOT's I/O are not (meant to be) thread safe

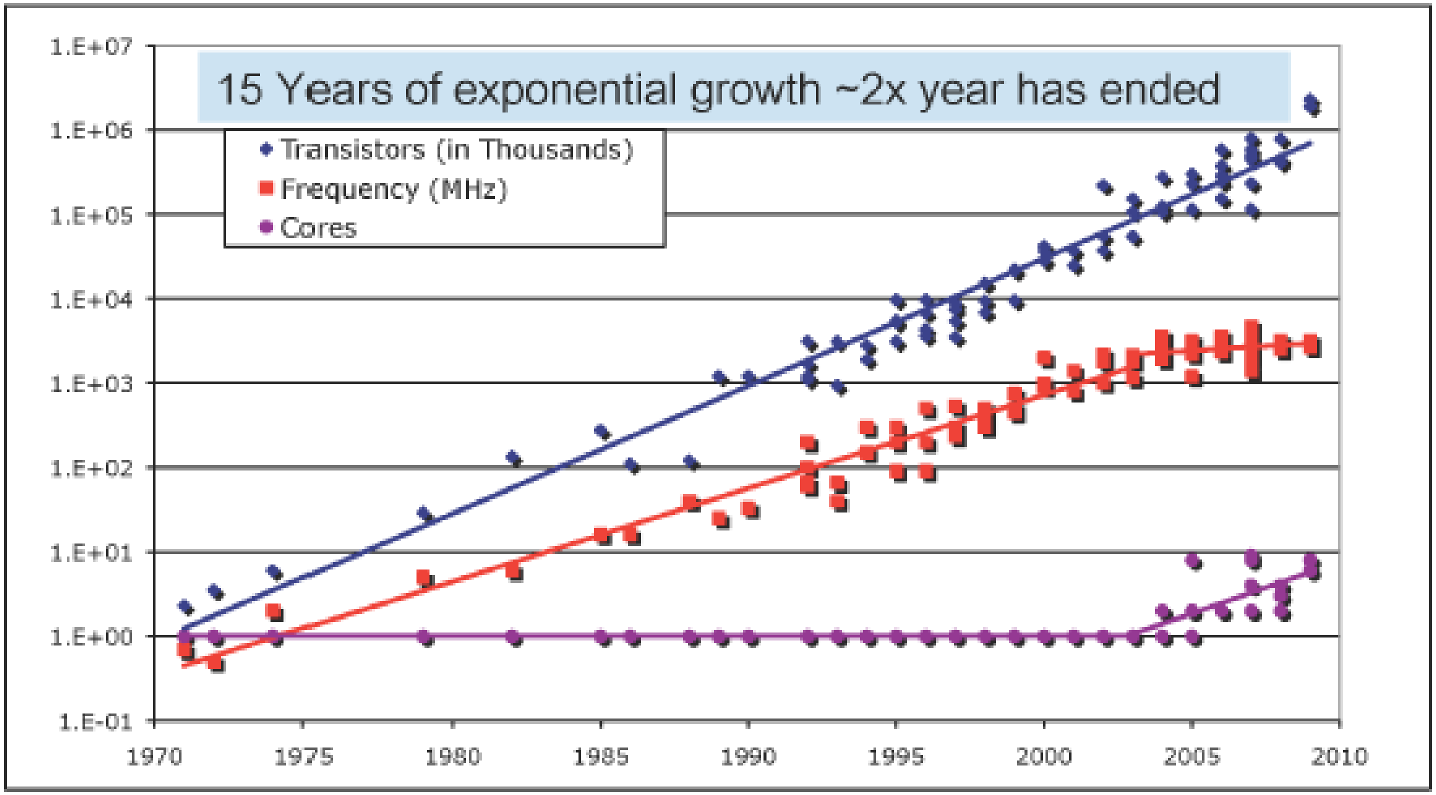
Towards a Computing Model for a tau-charm Factory

A Possible Computing Model

- “Raw data” from the detector will be permanently stored, and reconstructed in a two step process.
- Monte Carlo data will be processed in the same way.
- Selected subset of Detector and MC data, the “skims”, will be made available for different areas of physics analysis.
  - Very convenient for analysis.
  - Increase the storage requirement because the same events can be present in more than one skim.
- Improvements in constants, reconstruction code, or simulation may require reprocessing of the data or generation of new simulated data.
  - Require the capability of reprocessing in a given year all the data collected in previous years.
- An estimate of necessary resources can be made based upon a set of assumption (luminosity profile, event size, acquisition rate…).
  - Expect O(100 PB), O(5 MHEPSpec).

Impact of Architecture Evolution

Moore's law still live and well.
But scaling of clock frequency replaced by scaling of cores/chip.



Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanoviç

Moore's Law Reinterpreted

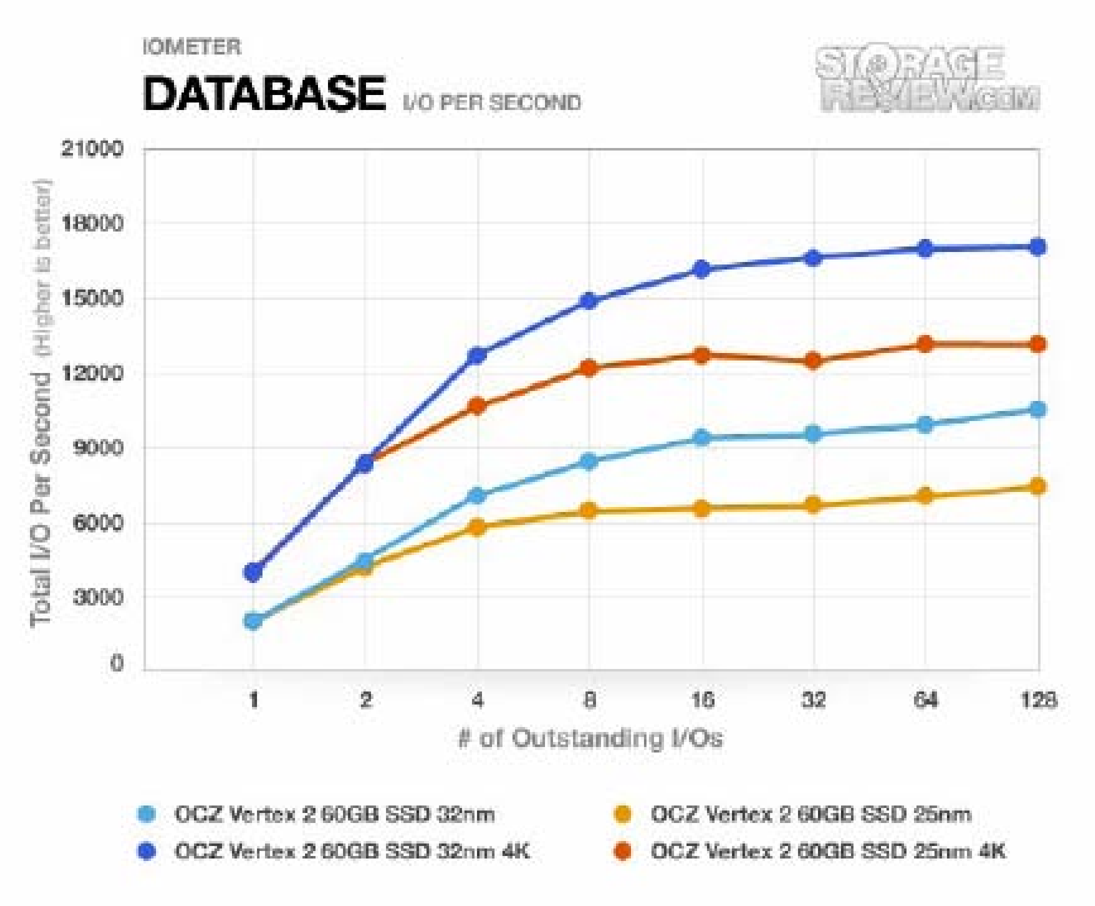
- Number of cores/chip will double every two years.
- Clock speed will not increase because of power.
- Need to deal with systems with millions of concurrent threads.
- Need to deal with inter-chip parallelism as well as intra-chip parallelism.

To Stay on Moore's Law

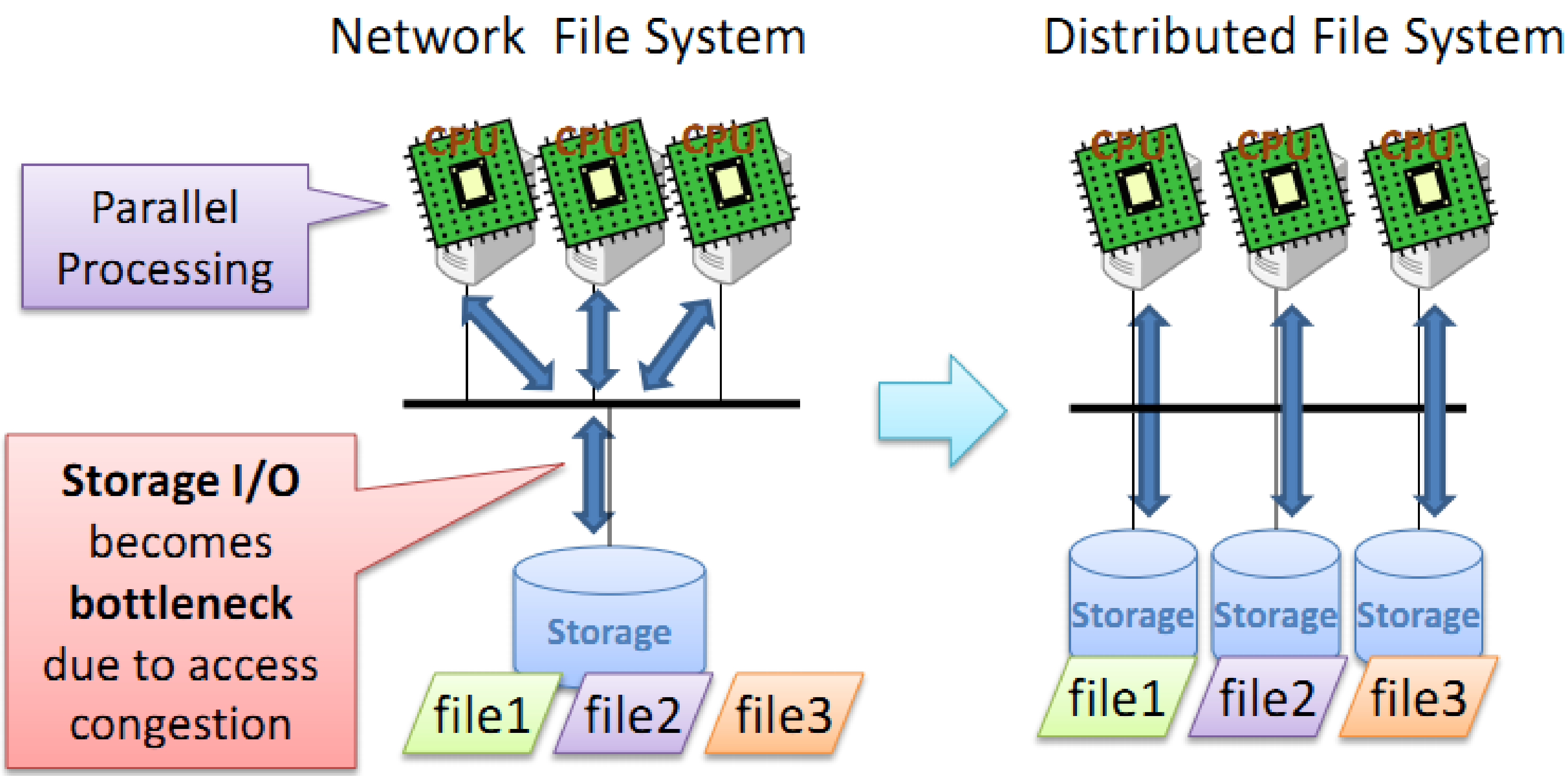
- We need to be able to exploit multi/many cores architecture with high efficiency.
- Efficient software will require a design that highlights parallelism.
  - Novel problem decomposition.
  - High granularity task.
- New programming paradigm.
  - Think local and parallel!
  - Decompose a problem vertically (parallel) first, then horizontally (sequentially)
  - Consider speculative computation in place of likely miss-predicted branches
  - Prefer deterministic algorithm to recursion, hit/miss
- The Event Processing Framework will have to enable such an approach
  - Task scheduling.
  - Memory Model & Data transformation.
  - Library of optimized algorithms.

Data Access & Distributed Storage

- Kryder's law ("Moore for storage"): disk storage density doubles every [year, or 18 months].
- Good. However, even if the number of bytes on a disk that can be bought for unit cost follows Moore's law, **the speed of disk access does not.**
- Need a strategy to avoid I/O bottlenecks



Needs for Distributed File System



Storage Questions

- How setup the storage in the sites and how share and replicate data between them?
- File/Replica in different location? Investigate on storage systems able to do it natively.
- Which data access services we want implement?
- Which file system is optimal for our application?
- Job Locality? Trying to understand if we can use a paradigm in which the job run as closer as possible to the data
- Catalogue and metadata system?

Grid Computing

- Wikipedia: **Grid computing is a term referring to the** combination of computer resources from multiple administrative domains to reach a common goal.
- In practice this is implemented using **Middleware** (Condor Toolkit, gLite, UNICORE, ARC,...) tools that provide access to Grid infrastructures.
- But it does not exclude that **HPC or Cloud resources** can be integrated when appropriated.

Evolution of Grids

- Middleware is moving towards better interoperability.
- Infrastructure is getting fragmented into multiple Grids.
- Special use cases require specialized resources.
- General purpose Grids will not solved all needs.
- Scientific communities are getting global:
  - Computing will be distributed

Issues with Grid Computing

- Dealing with heterogeneous resources
  - Various computing clusters, grids, etc
- Dealing with the intra-community policies
  - User groups, quotas and priorities
- Priorities of different activities
  - Dealing with a variety of applications
  - Massive data productions
  - Individual user applications, etc
- Overcome deficiencies of the standard middleware
- Inefficiencies, failures
  - Production managers can afford that, users can not
- Lacking specific functionality
- Alleviate the excessive burden from sites – resource providers – in supporting multiple VOs
  - Avoid complex VO specific configuration on sites
  - Avoid VO specific services on sites

Grid Resource Management Framework (1)

- The complexity of managing the workload resulted in specific software layer on top of the standard Grid middleware:
  - AliEn (Alice), PanDA (Atlas), GlideIn WMS (CMS), DIRAC (LHCb)

- Need a Distributed Resource Management Framework.

Grid Resource Management Framework (2)

- Workload Management:
  - Handling of computing tasks
  - Locate optimal resource for execution
  - Ensure proper execution
  - Retrieval of results

- Key aspects:
  - Global view of resources and needs (integration of all activities)
  - Provide interoperability by adding a common layer
  - Ready to integrate new domains

Grid Resource Management Framework (3)

- Data Management:
  - Handing of data to make it available were needed
  - Efficient use of resources (storage, network,..)
  - Flexible access: local, remote
  - Metadata

- Key issues:
  - Dynamic data placing (popularity)
  - Resource management
  - Data Integrity

Outlook

- Computing for a tau-charm factory has many similarities with computing for SuperB.
  - And any experiment with similar data volumes and CPU requirements.
- Tools developed for SuperB could be adapted to an experiment at a tau-charm factory.
- The SuperB R&D program adresses some of the questions that need to be adressed by an experiment at a tau-charm factory.