HPC Applications on the Sicilian Grid Infrastructure

Marco Fargetta (Marco.Fargetta@ct.infn.it)

Consorzio COMETA

Workshop finale dei Progetti Grid del PON "Ricerca" 2000-2006 - Avviso 1575

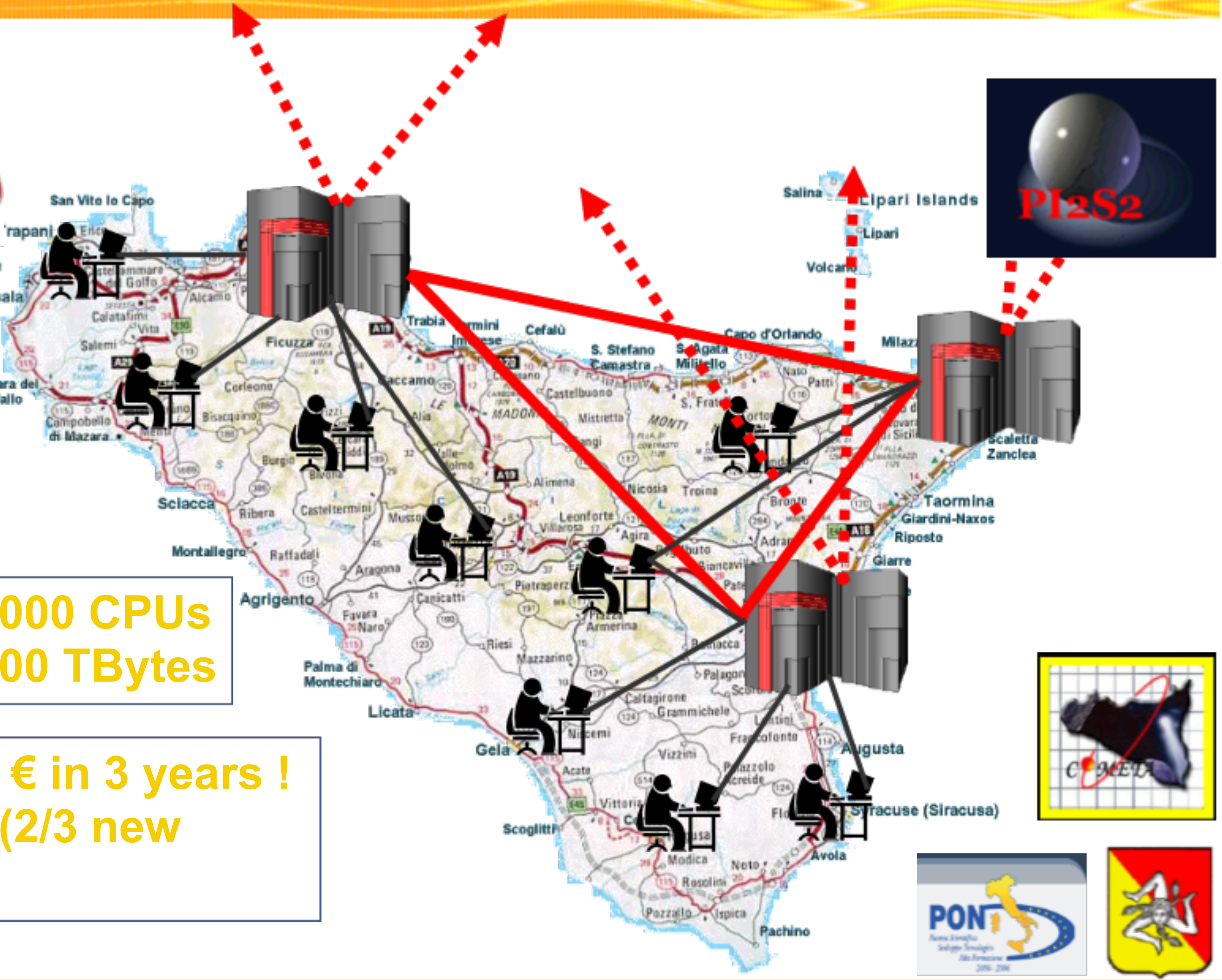
Catania, 10-12.02.2009

Outline

- Overview
- Sicilian Infrastructure
- Grid & HPC
- Middleware
  - MPI support
- Scheduling Policies
- License Server
- Use Cases
  - FLUENT, FLASH, OpenFOAM, etc...

The Sicilian E-Infrastructure

Trinacria Grid
Virtual Laboratory



More than 2000 CPUs
More than 300 TBytes

~15.000.000 € in 3 years !
~350 FTEs! (2/3 new employees)

COMETA

PON

Grid vs HPC

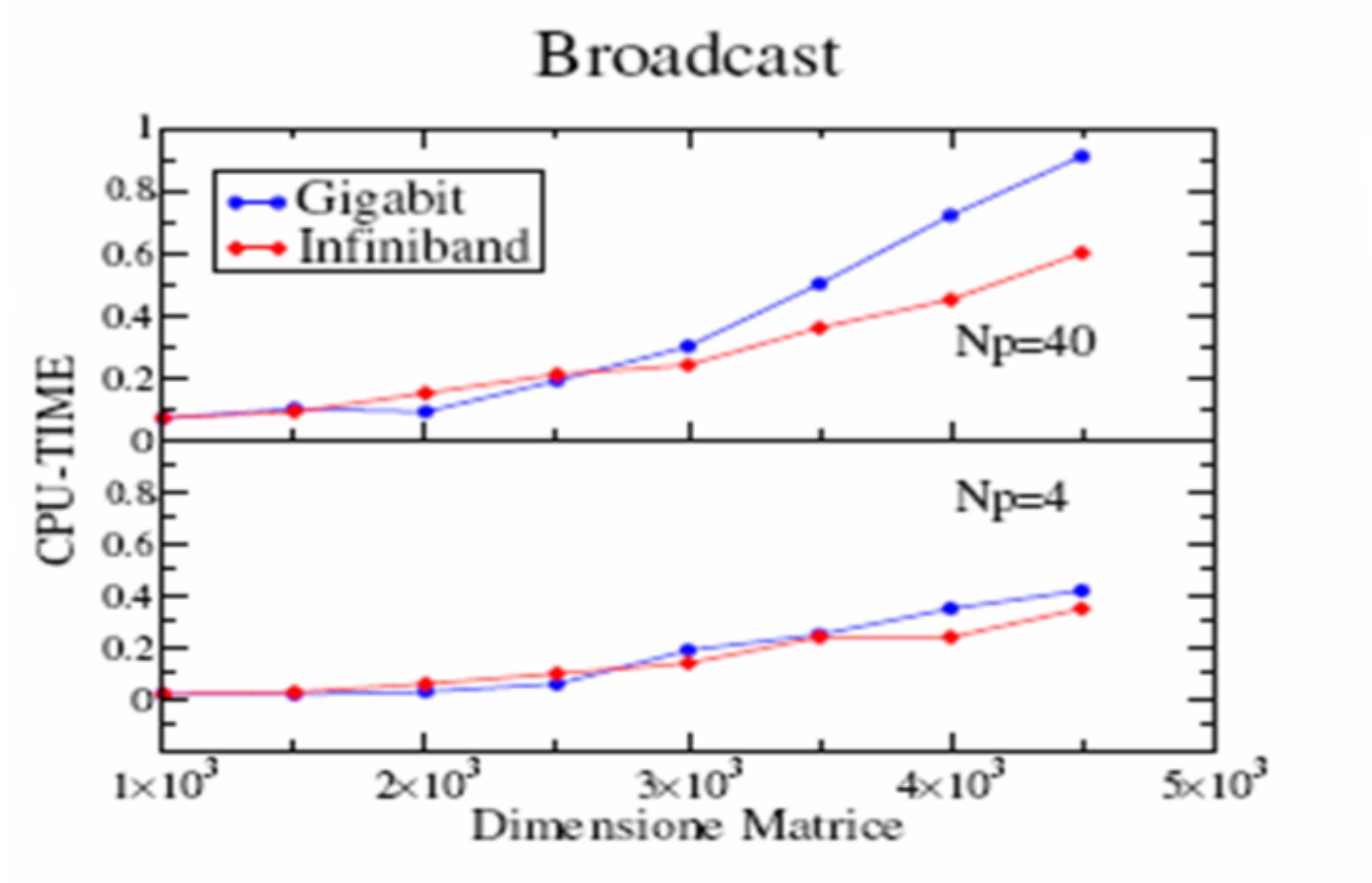
- **GRID**
  - Heterogeneous
    - Resources
    - Jobs
    - Users
  - Common network infrastructure (generally GigaBit Ethernet)
  - Focus on the overall infrastructure performance
    - Number of jobs
  - Allow the resources sharing
    - Reduce the ownership of the infrastructure

- **HPC**
  - Clusters dedicated to a single user (or community)
    - One user at a time
  - Components optimised for one application
  - Ad-hoc network technologies
  - Cutting edge components to get the best performance
  - Too expensive for many small companies and/or research groups

HPC on Grid

- **Provide a Grid supporting HPC is important to involve new communities**
  - Many HPC users will be happy to move on Grid and so reducing the the costs if they have comparable performance
- **PI2S2 has pushed a big effort on enable its infrastructure to the HPC users**
  - Choosing infrastructure components satisfying the requirements of many HPC application
    - es. server with multicore CPU, InfiniBand for nodes interconnection
  - Developing new solutions to integrate in the Grid middleware so to better manage HPC applications
  - Developing a license server to run on Grid so to allow the execution of commercial parallel applications
    - many users required commercial application like FLUENT and other

GigaBit vs InfiniBand



The advantage of using a low – latency network becomes more evident the greater the number of nodes

MPI & Grid

- **An MPI program is an application including calls to MPI library functions that synchronise the execution flow among cooperating nodes**
- **Official gLite supports only two MPI implementations:**
  - MPICH
  - MPICH2
- **Several patches enable MPI on “old” GigaBit Ethernet and “new” low-latency InfiniBand nets**

  NEW

  - In Cometa infrastructure MPI jobs run on either GigaBit (MPICH, MPICH2) or InfiniBand (MVAPICH, MVAPICH2)
- **Currently, MPI parallel jobs can run only inside a single Computing Elements (CE)**
  - The possibility of executing parallel jobs spread on different CEs is under investigation

Submission of an MPI application

- MPI jobs are specified by setting the JDL JobType attribute to **MPICH**
- The implementation variant is specified with the attribute MPIType
  - <MPIvariant>_<compiler>, e.g. `MVAPICH2_PGI706`
- The NodeNumber defines the number of required cores
- Es.

```
Type = "Job";
JobType = "MPICH";
MPIType = "MVAPICH_gcc4";
NodeNumber = 12;
Executable = "mergesort-ib1-gcc4";
........
```

Matchmaking: the Resource Broker (RB) chooses a CE (if any!) with enough free Processing Elements (PE = CPU cores)

e.g.: free PE# ≥ NodeNumber

Matchmaking

- When these attributes are included in a JDL script the following requirements are automatically added by the WMS:

```
(other.GlueCEInfoTotalCPUs >= NodeNumber) &&
Member (<MPIType>,other.GlueHostApplicationSoftwareRunTimeEnvironment)
```

Replaced with the real value

- These requirements allow the WMS to find out the best resource where the job can be executed

Additional elements

- **Executable** specifies the MPI executable
- **Arguments** specifies the WN command line
  - Executable + Arguments form the command line on the WN
- **mpi.pre.sh** is a special script file that is sourced before launching MPI executable

NEW

- **mpi.post.sh** is a special script file that is sourced after MPI executable termination
  - warning: they run only on the master node
- The **mpirun** command is issued by the middleware (…what if a proprietary script/bin?)
- **MPIGranularity** is a JDL attribute specifying the CPU core distribution

Execution

- **The WMS Job wrapper copies all the files indicated in the InputSandbox on the master and ALL of the “slave” nodes**

host based ssh authentication MUST BE well configured between all the WNs

- **If additional environment variables are needed ONLY on the “master” node, they can be set by the mpi.pre.sh**
  - If required ON ALL THE NODES a static installation is the only method (middleware extension is under consideration)
- **The WMS start the execution on the master**

mpi.jdl

```
[
Type = "Job";
JobType = "MPICH";
MPIType = "MPICH_GCC4";
Executable = “MPIparallel_exec”;
NodeNumber = 2;
Arguments = “arg1 arg2 arg3";
StdOutput = "test.out";
StdError = "test.err";
InputSandbox = {“mpi.pre.sh”,“mpi.post.sh”,
“MPIparallel_exec”};
OutputSandbox = {“test.err”, “test.out”, “
out”};
Requirements =
other.GlueCEInfoLRMSType == "PBS"
|| other.GlueCEInfoLRMSType == "LSF";
]
```

NEW

Executable

Pre e Post Processing Scripts

Local Resource Manager (LRMS) = PBS/LSF only

Scheduling Policy

- **Miscellaneous jobs running in the same infrastructure make Grids different from dedicated clusters**
  - Resources should be effectively shared among users having diverse constraint
- **PI2S2 sites have queues for different jobs duration**
  - increasing priority for shorter jobs
  - their policy is very complex due to the variety of requirements
- **Several type of jobs are identified**
  - emergency jobs need absolute priority so they perform pre-emption interrupting the execution of short jobs
    - emergency jobs are not very long, so pre-emption is acceptable
- **HPC jobs are based on resources reservation so they collect unloaded cores up to needed amount**
  - This policy is reasonably effective as the number of HPC jobs is far lower compared to the number of short jobs

License Server

- **Many HPC programs require a software license**
- **Unique license must be shared among the users on different and geographically distant sites**
  - Currently licenses are generally connected to a user name, or a physical net address
    - Both these solution do not fit the distributed environment of Grid infrastructures
- **A float license delivered to remote sites is required**
- **Several tools allow the management of floating license**
  - Flexlm is the most used free software license server
  - The applications must know the address of the server and they ask at run-time for an execution license
- **Current license servers do not identify Grid users so it is not easy to define usage policy for commercial software**

GridLM: Grid License Manager

- **To avoid problems with commercial software GridLM has been developed by COMETA and INFN Catania**
- **Execution permission is granted only for users belong to an authorised group inside a VO**
  - Users have to specify their group during the proxy creation
    - es. `voms-proxy-init --voms cometa:/cometa/example_sw`
  - Additionally, the software has to be specified in the JDL
- **The communication among sites use the Grid security mechanism**
  - Every message is encrypted with the proxy certificates
- **A further development aims to associate the license with delegated user proxy linked to a job**

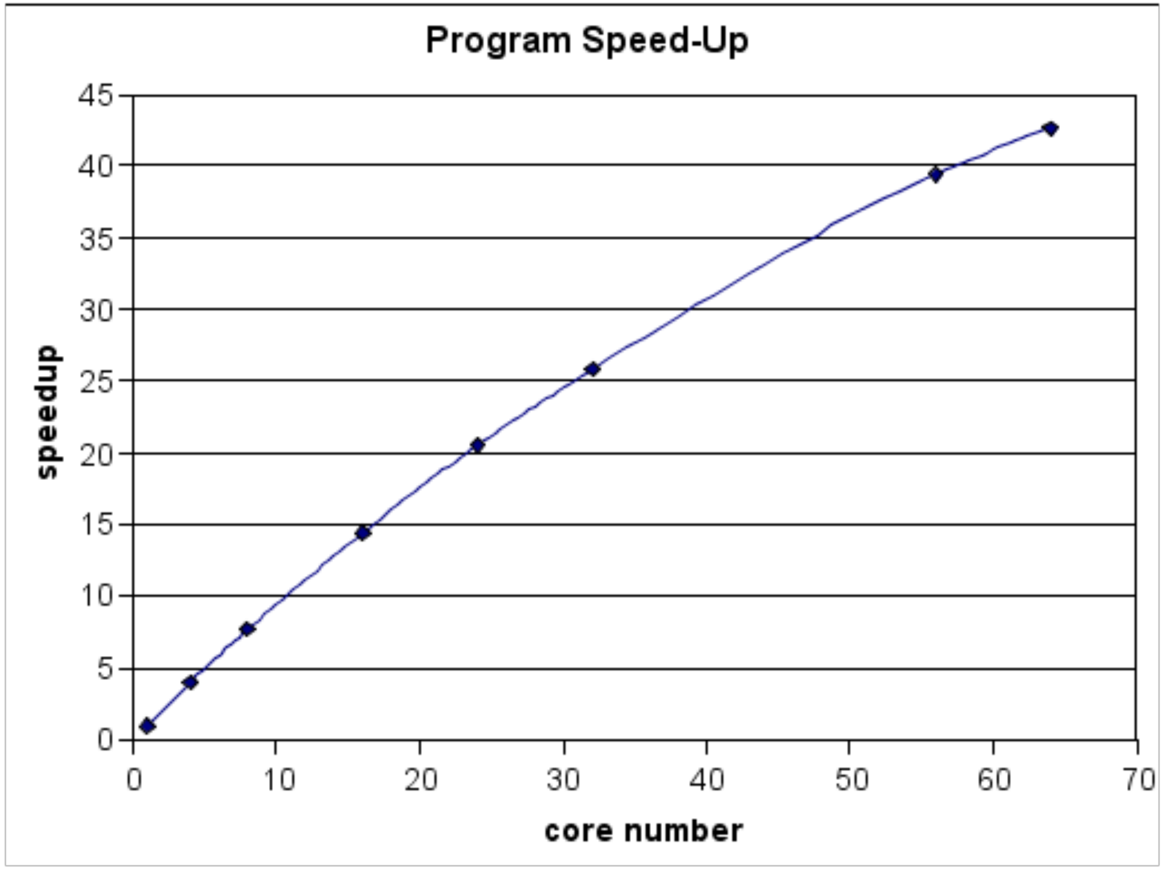
Use Case: FLUENT

- **FLUENT is a commercial software for Computer Fluid Dynamics (CFD)**
  - mainly used for for flow modelling, and heat and mass transfer simulations
- **Current distribution include the support for many MPI cluster configurations including the MPICH on InfiniBand**
- **No big effort required to execute FLUENT on Grid**
  - The MPI wrapper was by-passed since FLUENT uses its own wrapper
  - The pre-processing script is responsible to run the application
  - The other JDL parameter are used to find the execution resources

Use Case: FLASH

- **FLASH is a 3D astrophysical hydrodynamic code for supercomputers**
  - used in current astrophysical research
- **Compiled with PGI compiler and optimised for COMETA infrastructure**
- **In production with job using up to 64 cores and running for several days**
- **Some tests on performance enhancement after the optimisation have been executed**

Use Case: FLASH



Use Case: OpenFOAM

- **OpenFOAM is a free simulation environment**
  - commonly used as an equation solver in CFD problems
- **Very complex to execute and integration with Grid is not yet complete**
  - a special set-up is required in any execution node but Grid mechanisms allow access only to the master
  - the MPICH support requires to recompile the code with a no supported compiler
    - this is not a real limitation but need an extra effort to test the application
  - writing a solver requires to recompile the entire software so a mechanism to support new developer has to be investigated

MPI on the web...

- https://edms.cern.ch/file/454439/LCG-2-UserGuide.pdf
- http://oscinfo.osc.edu/training/
- http://www.netlib.org/mpi/index.html
- http://www-unix.mcs.anl.gov/mpi/learning.html
- http://www.ncsa.uiuc.edu/UserInfo/Training

Thank you for your kind attention !

Any questions ?