Towards a robust and user-friendly MPI functionality on the EGEE Grid

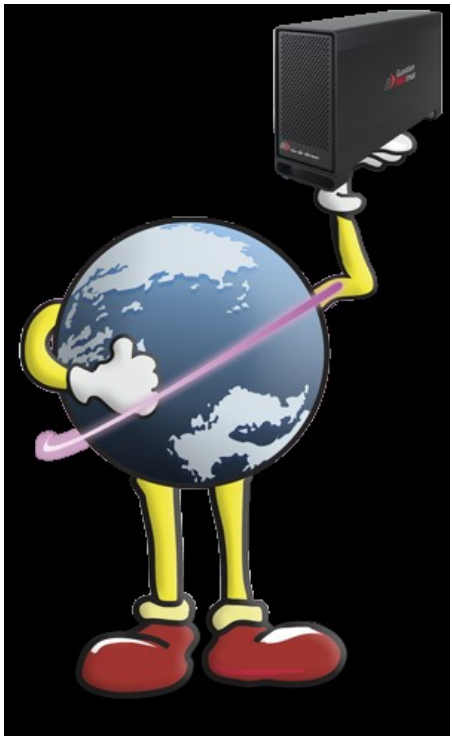
Jeroen Engelberts
SARA Reken- en Netwerkdiensten, Amsterdam
jeroene@sara.nl

Outline

- **Outline**
  - Introduction
  - The MPI Working Group
    - Background
    - History
    - The MPI Task Force
  - Towards user-friendly MPI functionality
    - Current status
    - Complete nodes
    - Packing of tasks on nodes
  - Conclusions

Introduction

- **The Grid/HPC world we live in:**
  - LCG/gLite initially designed for serial job-farming
  - EGEE expanded beyond High-Energy Physics
  - Grid infrastructure heterogeneous -
    - Several schedulers are used:
      - *PBS, LSF, Sun Grid Engine*
    - Several hardware network types exist:
      - *GigE, Infiniband, Myrinet, etc*
    - Several popular MPI flavors exist:
      - *LAM, OpenMPI, MPICH*
  - Scheduling balance serial / parallel jobs



First MPI Working Group

- **MPI was implemented, but was rather limited**
  - First MPI Working Group was setup
  - Their achievements/recommendations were
    - MPI implementation should require few middleware modifications
    - Info systems advertise the available capabilities/flavors
    - Environment variables set in user space
    - Easy setup guide for site admins
    - Jobtype = “MPICH” → “Normal” with setting of Cpunumber = *n*
  - Still, MPI functionality not satisfactory, resulting in:
    - Local, site-specific MPI implementations
    - Sites not installing MPI (too hasslesome)
    - Users not using (often broken) MPI installation

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP
- **MPI Task Force started after EGEE'09**
  - Consists of two “champions”: John Walsh & Isabel Campos
  - See previous presentation (John Walsh) for achievements

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - **→ Find out why only a handful of sites install MPI ←**
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - **→ Deploying MPI support for users and site administrators ←**
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - **→ Reduce flavors to OpenMPI and MPICH2 ←**
    - Revive MPI SAM test
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - **→ Revive MPI SAM test ←**
    - Future support of MPI
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - **→ Future support of MPI ←**
    - Enable requests for all cores in one node
    - Enable specification of process packing over nodes
    - Broaden to other parallel methods like OpenMP

(New) MPI Working Group

- **New MPI WG to address remaining and new issues**
  - Write a recommendation document describing:
    - Find out why only a handful of sites install MPI
    - Deploying MPI support for users and site administrators
    - Reduce flavors to OpenMPI and MPICH2
    - Revive MPI SAM test
    - Future support of MPI
    - **→ Enable requests for all cores in one node ←**
    - **→ Enable specification of process packing over nodes ←**
    - **→ Broaden to other parallel methods like OpenMP ←**

Currently available

Example JDL-fragment

```
…
Jobtype = “Normal”;
Cpunumber = 8;
…
```

User gets: 8 cores (job slots) in one cluster

- Not suitable for some jobs of type:
  - Shared memory
  - Auto multi-threaded applications
  - Memory intensive applications
  - I/O intensive
  - **Not** I/O intensive

New JDL parameters

- **Three new parameters**
  - WholeNodes – type: boolean – default: False
    (whether or not full nodes will be reserved, no other users or jobs)
  - NodeNumber – type: integer – default: 1
    (number of nodes requested)
  - SMPGranularity – type: integer – default: 1
    (minimum number of cores per node)

- **Already existing JDL parameter**
  - CPUNumber – type: integer – default: 1
    (used as number of cores or job slots)

Constraint: CPUNumber <= NodeNumber * SMPGranularity

Example 1

- **A pure multithreaded application with 4 threads**

```
…
Jobtype = “Normal”;
WholeNodes = “True”;
SMPGranularity = “4”;
…
```

  - NodeNumber defaults to 1
  - CPUNumber defaults to 1, but is not used here

Example 2

- **An MPI job spread over many nodes**

```
…
Jobtype = “Normal”;
NodeNumber = “16”;
CPUNumber = “16”;
…
```

  - SMPGranularity defaults to 1, but is not used here
  - WholeNodes defaults to False, resulting in sharing the nodes

    Could be set to True for single user use. If billed by #nodes#wallclockhours, this can be (very) expensive*

Example 3

- **A hybrid MPI/OpenMP job**

```
…
Jobtype = “Normal”;
WholeNodes = “True”;
NodeNumber = “4”;
SMPGranularity = “4”;
…
```

  - CPUNumber defaults to 1, but is not used here

Example 4

- **Just one machine**

```
…
Jobtype = “Normal”;
WholeNodes = “True”;
…
```

  - NodeNumber defaults to 1
  - SMPGranularity defaults to 1, but is not used here
  - CPUNumber defaults to 1, but is not used here

Well suited for Matlab jobs, which by default automatically claim all CPU and memory available in one node.

Scheduling

The perfect policy does not exist (all users happy)

The perfect policy will not last (job mix will change)

Don't let scheduling issues interfere with implementation

The dimensioning of subclusters is interpreted differently at sites (cores are averaged sometimes)

Make system administrators aware that such definitions could make MPI scheduling impossible

Conclusions and Outlook

- **MPI support should continue within EGI**
  - Centralized/Standardized MPI support is preferred
  - An MPI, or parallel Task Force is demanded
- **A JDL-implementation to handle requests for whole nodes becomes more important**
  - Increasing cores/node – automatic parallelization (eg Matlab)
  - Upgrade of 32 to 64 bits OS
- **User ability to steer packing of processes gives him/her the possibility to optimize his/her tasks**

Acknowledgements

MPI Working Group members

Roberto Alfieri (INFN, I)
Roberto Barbera (INFN, I)
Ugo Becciani (INAF, I)
Stefano Cozzini (Democritos, I)
Dennis van Dok (Nikhef, NL)
Fokke Dijkstra (Groningen University, NL)
Karolis Eigilis (Baltic Grid)
Francesco De Giorgi (Democritos, I)
Oliver Keeble (CERN)
Vangelis Koustis (PhD student, Greece)
Alvaro Lopez (CSIC, ES)
Salvotore Montforte (INFN, I)
John Ryan (Trinity College, Dublin, IRL)
Mehdi Sheikhalishahi (PhD student, Iran)
Steve Traylen (CERN)

MPI Task Force members

John Walsh (Trinity College, Dublin, IRL)
Isabel Campos (CSIC, ES)
Enol Fernández (CSIC, ES)