Deploying the Large Scale Air Pollution Model as a Grid Service

Cihan Sahin, Ashish Thandavan, Vassil Alexandrov

Centre for Advanced Computing and Emerging Technologies (ACET)
School of Systems Engineering
University of Reading, UK
Who are we?
OGSA Testbed Project
Danish Eulerian Air Pollution Model (DEAP Model)
Porting DEAP Model to our IBM resource
System Design
System Components
Conclusions & Future Work
• The Centre for Advanced Computing and Emerging Technologies (ACET) located at School of Systems Engineering, University of Reading.

• One of ACET’s computing resources consists of a four-node IBM p-655 cluster. Each node has eight 1.5 GHz POWER4 processors with 16 GB RAM and 72 GB internal storage.

• OS is AIX 5.2

• IBM’s LoadLeveler and Parallel Operating Environment
OGSA Testbed Project

• One year e-Science testbed project funded by EPSRC
• Install and evaluate middleware based on OGSI on partners’ resources forming a UK-wide testbed
• Document experiences with deploying applications on this testbed.
• Partners – Universities of Portsmouth, Westminster, Manchester, Southampton & Reading, Daresbury Laboratories and MTA-SZTAKI
• The University of Reading’s contribution – the IBM pSeries cluster and the DEAP Model
DEAP Model (Physics)

Physical and Chemical Processes

System of Partial Differential Equations

Splitting procedure

Horizontal Transport

Horizontal Diffusion

Chemistry & Emissions

Deposition

Vertical Exchange

Horizontal Transport & Horizontal Diffusion

Vertical Exchange

Chemical Reactions & Emissions & Deposition
DEAP Model Details

- Space domain is 4800x4800 km, Europe at the centre.
- Coarse resolution: 96x96 grids, Fine resolution: 480x480 grids.
- 2-D: One vertical layer, 3-D: 10 vertical layers.
- Time step: 900 sec for transport sub-models & 150 sec for chemical sub-model.
- Parallelisation: MPI and OpenMP
- A configuration file “initial_input” defines number of grids on X,Y,Z coordinates and the location of the input data.
• MPI deadlock problem

**BEFORE (Sun cluster)**

```fortran
  call MPI_WAITALL(4, request, status, ierror)
  if(ierror.ne.MPI_SUCCESS) then
    ierror = -13
    return
  endif
```

**AFTER (IBM resource)**

```fortran
if (nproc > 1) then
  bottom(1)=request(1)
  bottom(2)=request(2)
  top(1)=request(3)
  top(2)=request(4)
  if (me .eq. 0) then
    call MPI_WAITALL(2, bottom, status, ierror)
  else if (me .eq. nproc-1) then
    call MPI_WAITALL(2, top, status, ierror)
  else
    call MPI_WAITALL(4, request, status, ierror)
  endif
endif
if(ierror.ne.MPI_SUCCESS) then
  ierror = -13
  return
endif
```
Porting DEAP Model to IBM resource –II

- DEAP Model runs on our IBM resource and scales almost linearly.

<table>
<thead>
<tr>
<th>SUB-MODEL</th>
<th>N=1</th>
<th>N=2</th>
<th>N=4</th>
<th>N=8</th>
<th>N=16</th>
</tr>
</thead>
<tbody>
<tr>
<td>WIND+SINKS</td>
<td>120.6</td>
<td>61.5</td>
<td>34.4</td>
<td>18.1</td>
<td>19.8</td>
</tr>
<tr>
<td>ADVECTION</td>
<td>1362.5</td>
<td>654.1</td>
<td>317.8</td>
<td>165.7</td>
<td>84.9</td>
</tr>
<tr>
<td>CHEMISTRY</td>
<td>5884</td>
<td>2971.2</td>
<td>1483.7</td>
<td>778.1</td>
<td>411.4</td>
</tr>
<tr>
<td>OUTSO2</td>
<td>375.9</td>
<td>305.2</td>
<td>280.2</td>
<td>319.3</td>
<td>343.2</td>
</tr>
<tr>
<td>COMM</td>
<td>0.08</td>
<td>51.9</td>
<td>45.1</td>
<td>105.1</td>
<td>283.9</td>
</tr>
<tr>
<td>TOTAL</td>
<td>7749.4</td>
<td>4051</td>
<td>2168.2</td>
<td>1388.5</td>
<td>1143.2</td>
</tr>
</tbody>
</table>

Run results for 2-D coarse grain DEAP model over a year.
Ideal Case Design

- Globus Toolkit v3 is installed on the IBM resource.
- Clients invoke the grid service deployed on the IBM resource.
- The grid service invokes WS-GRAM service which interfaces with Loadleveler.

**BUT**

Globus Toolkit v3 could not be installed!
Alternative Design

• The Globus Toolkit v3 is installed on a machine between clients and the IBM resource.

• Globus machine handles essential functionality like security.

What needs to be developed?

• A resource management mechanism
• A grid service deployed on the globus machine
Alternative Design Components

1. DEAP Model Grid Service
2. Resource Management
3. Interface to Loadleveler
4. Client Application
DEAP Model Grid Service

- Remote method specifications and implementation
- Submitted jobs’ status retrieval (use of notifications)
- Authentication of the users by proxy certificates
- Authorization of the users by mapping to local usernames (grid-map file)
- Message level security. All RMIs are encrypted.
Resource Management

- Client / Server pairs for job control and job status retrieval.
  - A: Job Control Client
  - B: Job Control Server
  - C: Job Status Client
  - D: Job Status Server
  - 2-7: Represents a job control data / message flow.
  - 9-13: Represents a job status data / message flow.

- A, B: Forwards job control requirements and local scheduler messages.
- C, D: Handles job status retrieval
A Perl script wrapped in $B$ and $C$ of the resource management programs.

- Parses the job specification file and prepares LoadLeveler job submission file
- Uses a mapping file to map specifications to LoadLeveler commands.
- Submits to LoadLeveler and returns job identification number
- Or LoadLeveler message in case cancel, hold or release are used.
Client Application

Client Interface for Grid Service (v1.0)

Grid Service Application Interface

Connection | Job Control
---|---
Job Id: acet-blue 1172 | Cancel job
Job Id: acet-blue 1183 | Hold Job

Status | Output | Info | Handle
---|---|---|---
N/A | Available | hash-15592594-110555712...
N/A | Available | hash-1105930-1105557269...

Grid Service Application Interface

Job Specification

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Arg Partition Model on Reading University HPC resource</td>
</tr>
<tr>
<td>Package</td>
<td>remote demo modeler</td>
</tr>
<tr>
<td>Executable</td>
<td>main/mult.model</td>
</tr>
<tr>
<td>Parameter</td>
<td>loadleveler</td>
</tr>
<tr>
<td>Scheduler</td>
<td>parallel</td>
</tr>
<tr>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>Processor Number</td>
<td>8</td>
</tr>
<tr>
<td>Input Data</td>
<td>initial_input.mgn</td>
</tr>
<tr>
<td>Output Data</td>
<td>output_data/kdem_model_output.tar</td>
</tr>
</tbody>
</table>

Create Job Specification File

Reset Values

Log:

- Notifier: 2009-08-28T11:19:48Z: Starting notification collector ...
- Notification Generator is NULL -> Starting now ...
- Disconnecting: Connection to remote service is closed
- Notification generator is ON, Turning it OFF
- Listener is already RUNNING, BYPASS
- Notification Generator is OFF, Turning it ON
Conclusions & Future Work

- Development of a resource management system, can easily be extended for other architectures and schedulers.
- Proposed design is generic enough to deploy other applications as well.
- Client application can be extended for generic use or replaced by a portal interface.

- **OGSI is dead, Long Live WSRF!**
  OGSI is dead which means WSRF replaces the grid middleware with Web Services.

  **BUT**

  The functionality of all high level components stays the same!
Thank you!

Any Questions?

[c.sahin,a.thandavan,v.n.alexandrov]@rdg.ac.uk