OGST: an Opportunistic Grid Simulation Tool

Gilberto Cunha Filho and Francisco José da Silva e Silva
Latin American Grid (LAGrid) workshop

Grupo de Engenharia de Sistemas e Mobilidade (SisMo)
Programa de Pós-Graduação em Engenharia de Eletricidade
Universidade Federal do Maranhão (UFMA)
http://www.sismo.deinf.ufma.br

October - 2008

Introduction

- Computational grids have become an attractive alternative for execution of applications that demand huge computational power and have been used to solve problems in varied areas of scientific, enterprise, and industrial activities;
- Several research groups have been addressing the complexity of building the Grid software infrastructure, facing challenges such as the support for:
  - Huge resource heterogeneity;
  - High scalability of distributed resources;
  - Efficient resource allocation and management;
  - Dynamic resource scheduling;
  - Distributed fault-tolerance.

Introduction

- Simulation tools play a fundamental role on the development of Grid middlewares since:
  a) Researchers often do not have access to huge Grid testbed environments, limiting the capacity for evaluating situations that demand high amount of resources;
  b) It is difficult to explore in large scale application and resources scenarios involving several users in a repetitive and controlled way, due to the dynamic nature of Grid environments;
  c) Real Grid applications usually consume great amount of time, ranging from a few hours to even weeks.

Introduction

- Recognizing that the challenges of the grid, we are currently involved on developing adaptive scheduling approaches and dynamic re-scheduling of applications on InteGrade (regarding self-optimization);
- In order to be considered successful, our work must provide answers to fundamental questions, such as:
  - Is it worth to perform dynamic adaptations to the Grid scheduling policy?
  - What are the costs/benefits involved?
  - When adaptive actions should be applied?
  - What are the adaptive actions that should be considered?
- For finding answers to those questions, we developed an Opportunistic Grid Simulation Tool (OGST).

Opportunistic Grid Simulation Tool (OGST)

- OGST (Opportunistic Grid Simulation Tool) is a simulation utility whose main objective is to assist developers of opportunistic Grid middlewares on validating new concepts and implementations under different execution environment conditions and scenarios;
- It allows the simulation of large scale applications and resources scenarios involving several users in a repetitive and controlled way;
- It was developed in the context of the InteGrade project, but was designed to allow the simulation of generic opportunistic Grids in order to be applied by other Grid middleware research projects.

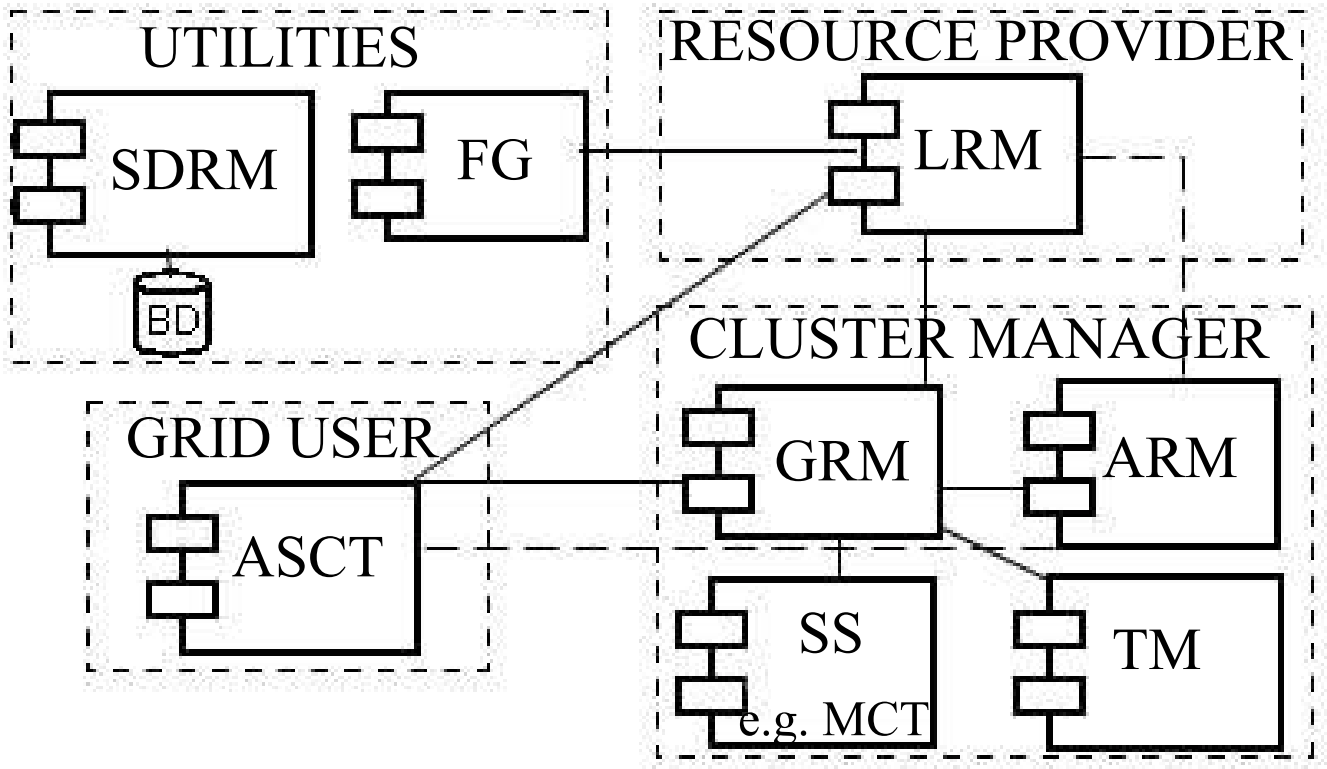
OGST design principles

- OGST design principles in accordance with the characteristics of opportunistic Grids, establishing the following requirements for its development:
  1. it must provide support for defining Grid environments that exhibits high heterogeneity of machines and network links;
  2. it must allow the definition of heterogeneous applications, ranging from regular to bag-of-tasks and parallel applications;
  3. it must allow the simulation of frequent join and leave of nodes;
  4. it must allow node and link fault injection;
  5. it must provide support for simulating application fault tolerance mechanisms commonly applied on opportunistic Grids (e.g. checkpointing, and replication).

OGST design principles

6. it must allow the simulation of variant availability of each Grid node, considering different usage periods;
7. it must allow the scheduling algorithm to obtain informations concerning the applications and the execution environment;
8. it must allow the dynamic replacement of the scheduling algorithm and/or dynamic adjustment of scheduling parameters;
9. it must allow the use of traces collected from real environments (such as node availability) in addition to synthetic data.

OGST Architecture (*Main Components*)



OGST Architecture (*Main Components*)

- `Feature Generator` (FG) is a utility used for defining the simulated Grid environment (nodes and network links) and applications with their arrival rate;
- `Application Submission and Control Tool` (ASCT) represents the Grid user and is responsible for application submission and receiving notification about its conclusion;
- `Global Resource Manager` (GRM) receives application submissions from the ASCT and runs the `Scheduling Strategy` (SS), for schedule each application task for execution on a specific Grid node;
- `Trader Manager` (TM) provide data about the availability of Grid resources, and it is also responsible for simulating node failure and recovery.

OGST Architecture (*Main Components*)

- `Local Resource Manager` (LRM), responsible for the instantiation and execution of application tasks scheduled to the node, maintaining a list of tasks waiting for execution. It is also responsible for varying the local resource load;
- `SimulationDataRecordManager` (SDRM) uses a relational database for storing the simulation collected data (e.g. conclusion timestamps) and allows the automatic generation of graphs (e.g. the average application completion time as a function of the mean time between node failures or as a function of the arrival rates);
- `Application Replication Manager` (ARM) is responsible replica management, in order to circumvent eventual node failures.

Implementation highlights

- OGST was written in Java as an object-oriented system and is an extension of the GridSim toolkit;
- GridSim provides a base class called `GridSim`, that provides the communication between OGST components based on an event-driven approach;
- OGST allows the automatic creation of simulated Grid environments composed of a large amount of highly heterogeneous nodes, according to an uniform distribution;
- A node processing capacity is defined in MIPS (Millions Instruction Per Second) as per SPEC (Standard Performance Evaluation Corporation) CPU (INT) 2000 benchmark rating.

Implementation highlights

- OGST explicitly implements two application execution models: regular and Bag-Of-Tasks. The generated tasks length, defined in MI (Million Instructions), follows an uniform distribution (to simulate task heterogeneity);
- Tasks arrival time follows a Poisson distribution;
- Node failure and its recovery are generated during the simulation execution and follow a exponential distribution;
- OGST extends GridSim fault tolerance mechanism by providing support for simulating the use of checkpoint and replication, besides of an automatic restart of failed tasks;
- OGST provides a library of scheduling algorithms composed of four scheduling heuristics: InteGrade, OLB, MCT, and Min-min and provide the support for implementing other algorithms.

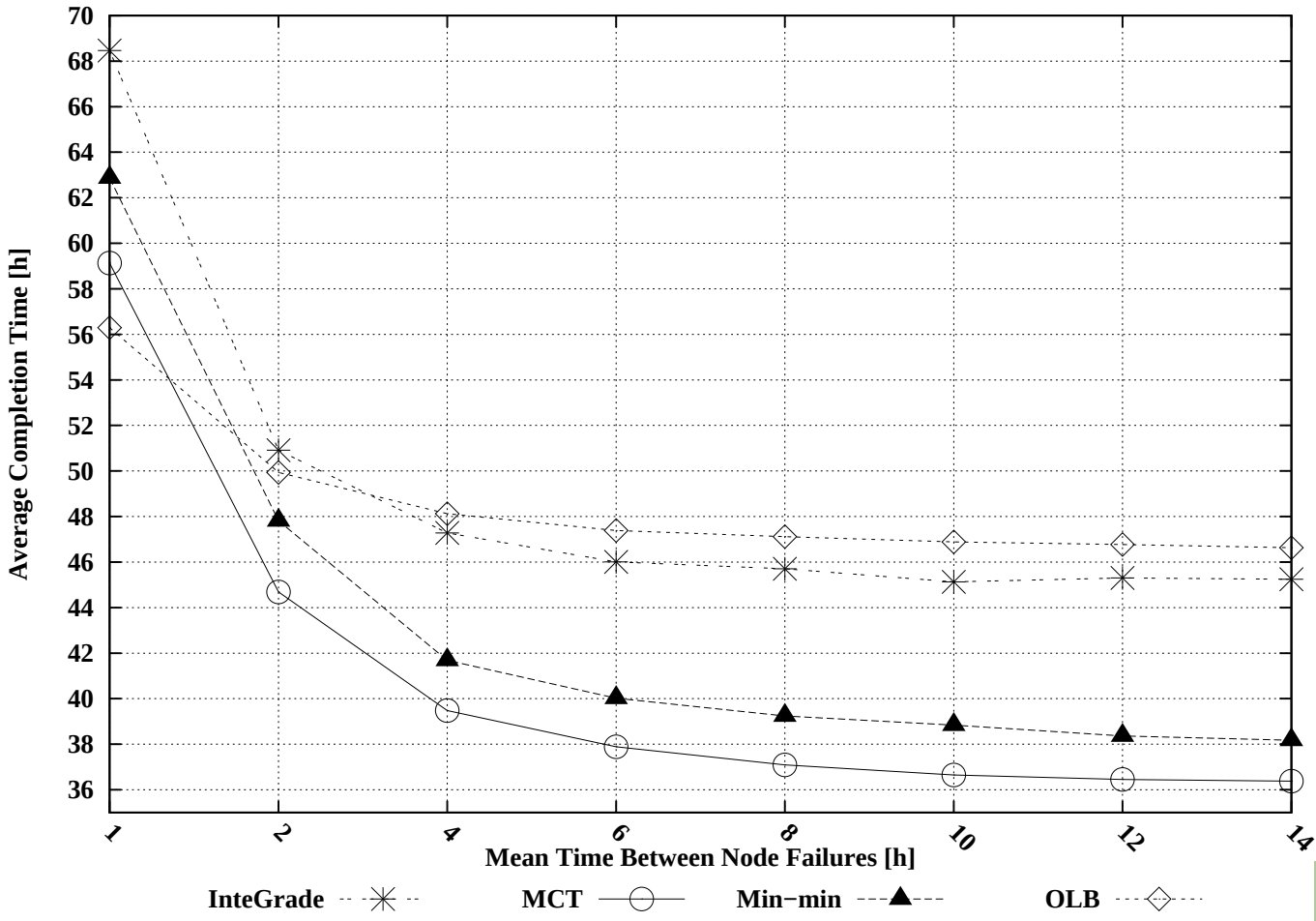
Simulations

- We performed simulations using OGST with the aim of analysing the performance of different scheduling algorithms (MCT, Min-min, OLB, InteGrade) under different execution environment conditions, considering the objective of minimizing the applications average completion time;
- For each combination of scheduling algorithm, arrival rate (0.025 (low) and 0.25 (high)), mean time between node failures (1 to 14 hours) and the use of two different fault tolerance mechanisms(restart and checkpointing), leading to a total of 128 different simulations;
- Each simulation was repeated 30 times, resulting on 3840 experiments.

Grid environment and applications

- The simulated Grid environment was composed of 100 machines with an average processing power equivalent to a Pentium 4 with 2.8 GHz (1000 MIPS), which was considered a representative value for regular personal computers;
- In order to take into consideration the environment heterogeneity, Grid nodes were generated according to an uniform distribution $U(222, 1776)$ MIPS, having the fastest machine a processing power 8 times higher than the slowest one;
- We simulated the execution of 1500 regular applications, synthetically generated with a variant length (in millions of instructions) through an uniform distribution, which takes approximately 5 to 50 hours of execution on an Pentium 4 with 2.8 GHz.

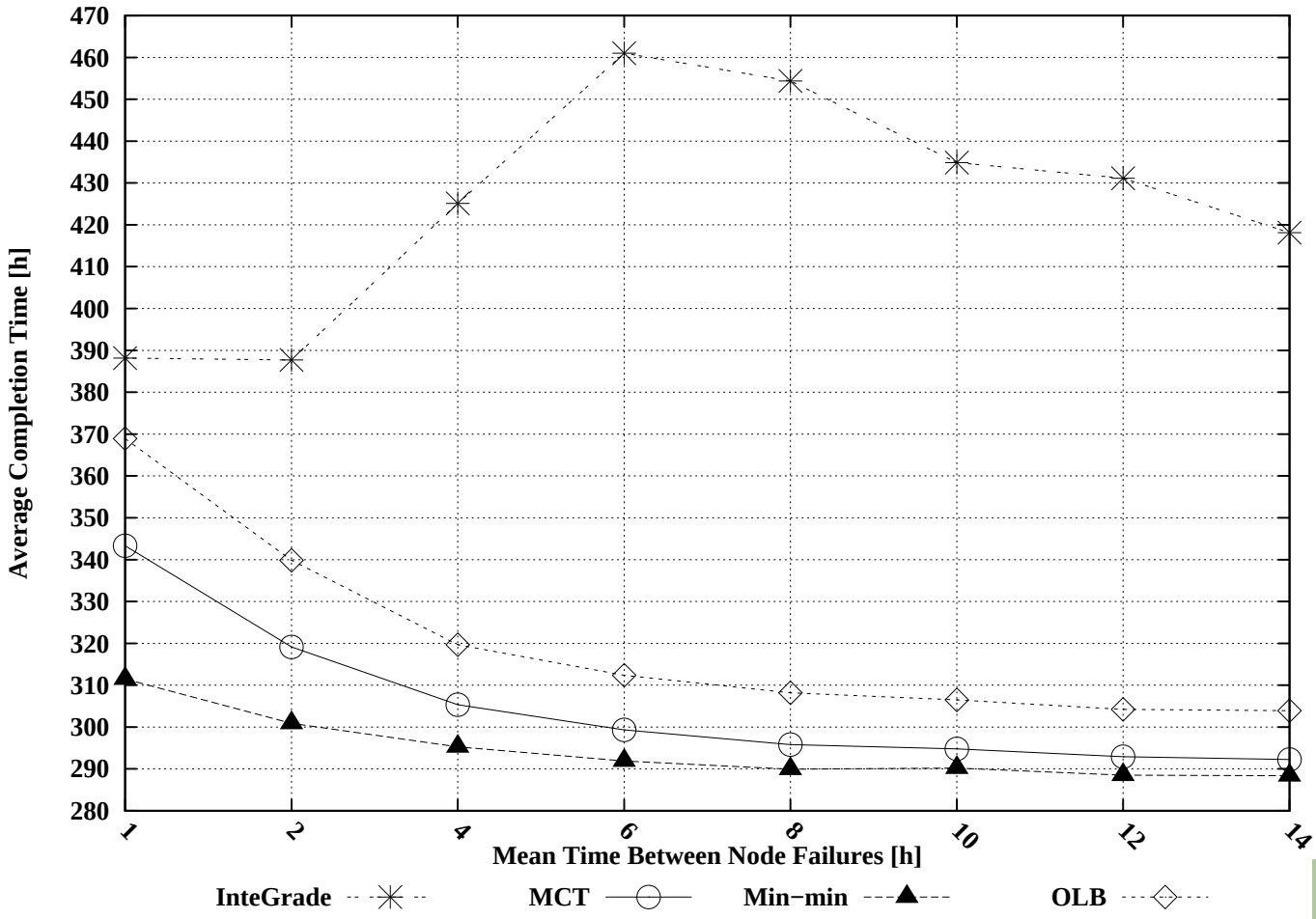
Average application completion time with restart failed tasks and an arrival rate of 0.025 applications per minute



1º Conclusions

- With a short interval between failures (1 hour), OLB achieved the lower average completion time per application, since it only schedules one task per machine reducing the required number of tasks resubmissions in case of a node failure;
- As the failure rate becomes lower (2 to 14 hours), MCT and Min-min performed better than OLB, by using information about the estimated task completion time;
- MCT performed better than Min-min, since MCT uses an on-line approach, scheduling the tasks as soon as they arrive to the scheduler, while Min-min uses a batch approach;
- Others simulations, when checkpoint is used for restarting failed tasks, even when the failure rate is high (1 hour), MCT and Min-min performed better than OLB, since the lower number of tasks resubmissions becomes less relevant, due to the fact that the tasks do not need to be restarted from scratch.

Average application completion time with checkpointing and an arrival rate of 0.25 applications per minute



2º Conclusions

When the environment exhibits a high arrival rate (0.25 applications per minute) and checkpoint or restart is used for task recovery, the Min-min obtained the best results, since the higher number of tasks maintains the Grid resources busy and more applications compose the mapping event set, allowing the comparison of their estimated completion time, which leads to a better task mapping.

Conclusion and future work

- OGST was carefully designed to take into consideration the dynamics of opportunistic Grids, providing a set of features that hasten the development of simulations that takes into consideration the dynamism of the execution environment.
- This work shows a first set of simulations performed that strengthened the benefits that can be achieved by switching from different scheduling heuristics considering a variant execution environment.
- Ongoing work includes:
  - the use of traces that will allow the use of real data concerning resource availability;
  - the support for dynamic replacement of scheduling heuristics;
  - the simulation of predictions of resource availability based on resource usage patterns.

Acknowledgments

This work, as part of the InteGrade project, is supported by the Brazilian Federal Research Agency, CNPq, grant No.55.0895/2007-8.