Multi-Agent Systems for Adaptive and Efficient Job Scheduling Service in Grids

Pooja Sapra
Lecturer (CSE)
Apeejay College of Engineering
Vill. Silani, Distt. Sohna, Gurgaon

Minakshi Memoria
Lecturer (CSE)
Apeejay College of Engineering
Vill. Silani, Distt. Sohna, Gurgaon

Sunaina
Lecturer (CSE)
Apeejay College of Engineering
Vill. Silani, Distt. Sohna, Gurgaon

ABSTRACT
In this paper we propose an adaptive efficient job scheduling service model on Grids using multi agent systems and a market like Service level Agreement (SLA) negotiation protocol based on the Contract Net model. This job scheduling service model involves four types of agents: service user agents, service provider agents, local scheduler agents and inter-grid agents. Service provider agents provide services to service user agents by allocating resources using local scheduler agents. Service provider agents provide services to service user agents by allocating resources using local scheduler agents. The service provider agents may contact the inter-grid agents if enough resources are not available in their own grid. Inter-grid agents provide resources from the neighboring grid. The service provider agent may adapt the dedicated service according to its interactions with service user agent.

The SLA negotiation protocol is a hierarchical bidding mechanism involving negotiations between the four agents. In this protocol, the agents exchange SLA announcements, SLA-bid, and SLA-awards to negotiate the schedule of jobs on Grid Compute resources. To deal with the presence of uncertainties, re-negotiation is used to allow the agents to re-negotiate the SLA in failure

Categories and Subject Descriptors
[**Computer Network]**: General – Sharing, Agents, Scheduling

General Terms
Networking, Grid Computing, Multi-Agent

Keywords
Scheduling, SLA, Multi-Agent, Grid Computing

1. INTRODUCTION
One of the challenges for a service Grid is to efficiently process users' requests to Grid services in large numbers. This is essentially a problem of optimal scheduling of the Grid resources to complete the requested services within a given time slot through effective job scheduling. Scheduling jobs in a Grid computing environment is a complex problem as resources are geographically distributed having different usage policies and may exhibit highly non-uniform performance characteristics, heterogeneous in nature, and have varying loads and availability.

In recent years, Multi-agent systems and the Contract Net Protocol [6] have achieved successful results in solving the scheduling problem in a wide range of applications such as flexible manufacturing systems [5, 6], e-commerce, railway scheduling, healthcare etc. In this paper we propose the extention of fundamental infrastructure for efficient job scheduling on Grids based on multi-agent systems, and a Service Level Agreement (SLA) negotiation protocol based on the Contract Net Protocol proposed by D. Ouelhadj et al. [15]. The proposed grid infrastructure is adaptive and inter-grid services can also be provided.

Service Level Agreements (SLAs) [1, 2, 3] are emerging as the standard concept by which work on the Grid can be arranged and coordinated. An SLA is a bilateral agreement, typically between a service provider and a service consumer. An SLA can be designed to include the agreed constraints for individual jobs such as acceptable start and end time bounds and a simple description of resource requirements.

A multi-agent system is a network of agents that work together to solve problems that are beyond their individual capabilities [7]. Multi-agent systems are distributed and autonomous systems made up of autonomous agents that support reactivity, and are robust against failures locally and globally [8]. Due to the highly heterogeneous, distributed, dynamic, and complex Grid computing environments, multi-agent systems appear to be a suitable approach to solve the Grid scheduling problem.

2. RELATED WORK
A number of initiatives to apply agents in computational Grids have appeared and they are still in their early development. Abramson et al. [9] developed a resource management system for scheduling computations on distributed resources (Nimrod-G). In Nimrod-G resource agents manage the execution of jobs on resources, and a central resource broker performs resource discovery, trading and discovery. Cao et al. [10, 11] developed an agent-based resource management system (ARMS) for Grid computing, where each agent represents a local Grid resource and acts as a service provider of high performance computing power. Agents cooperate with each other using a technique of service advertisement and discovery. Rana and Walker [12] proposed an agent-based approach to integrate services and resources in which service and resource agents contain behavioral rules, and can modify these rules based on their interaction with other agents and with the environment in which

they operate. Frey et al. (2002) developed CONDOR-G, a computation management agent for Grid scheduling. The scheduling is centralized and performed by a remote agent. Buyya et al. [13] discussed economic models for resource allocation and for regulating supply and demand in Grid computing environments.

Most of these agent based Grid scheduling systems are centralized as scheduling is performed by a Grid high-performance scheduler (broker), and resource agents manage only the execution of jobs on resources. In a Grid environment resources are geographically distributed and owned by different individuals. It is not practical that a single point in the virtual system retains entire Grid's information that can be used for job scheduling. Therefore, distributed scheduling would be more efficient and robust.

3. THE PROPOSED SLA BASED EFFICIENT AND ADAPTIVE GRID SCHEDULING MULTI-AGENT SYSTEM INFRASTRUCTURE

Fig.1. shows the multi-agent infrastructure for SLA based Grid scheduling. The infrastructure involves four types of agents: Service User Agents, Local Scheduler (LS) Agents, Service Provider (SS) Agents and Inter-grid Agents. Four databases are also used in this architecture to store SLAs information, LSA agents' information, resource description and Inter-Grid agents' information.

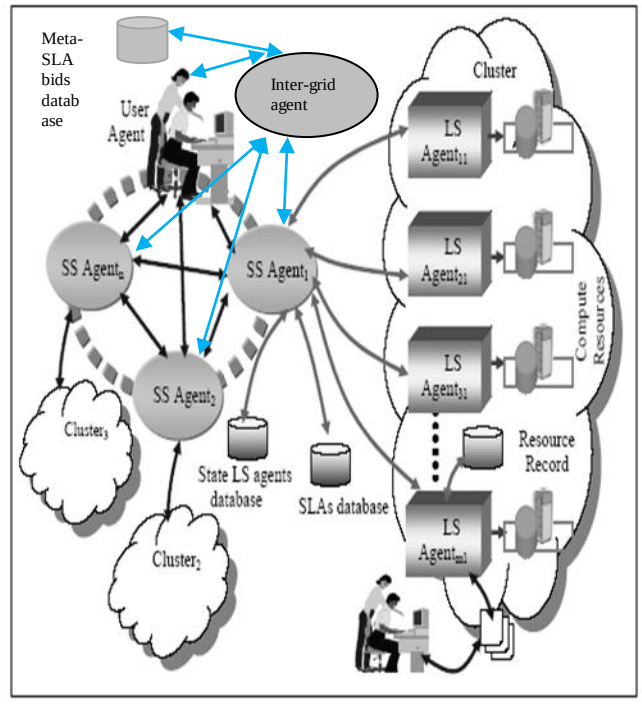

Fig.1. SLA based Grid Scheduling System Infrastructure

The proposed infrastructure is adaptive. The service provider agents may change the dedicated service according to its interactions with service user agent or may offer another service to change or adapt the original service (meta-level) or may use dynamic intelligent reflection rules to change the service it is currently providing.

3.1 Service User Agent

The service user agent requests the execution of services on the Grid and negotiates SLAs with the SS agents. The user agent can also submit services locally to SS agents.

3.2 Local Scheduler Agent

Each Computer resource within each institution is assigned to a LS agent. The set of Compute resources and their corresponding LS agents constitute a cluster. The LS agents are responsible for scheduling jobs, usually assigned by the SS agents, on Compute resources within the cluster.

3.3 Service Provider Agent

SS agents are distributed in the Grid, one at each institution, and act as mediators between the service user agents and the LS agents. Each cluster of LS agents of the same institution is coordinated by one SS agent. The user agent usually submits jobs to the SS agents. SS agents negotiate SLAs with the user agent and the LS agents to schedule the jobs on Compute resources.

3.4 Inter Grid Agent

Inter grid agent is one per grid and interact with service user agents and service provider agents. Inter grid agent is used for job scheduling among different LS agent clusters when none of the cluster is having enough resources for performing the job submitted by user agents. They can also be used for inter grid job scheduling.

3.5 Databases

The databases used in the architecture are the following:

Resource record: holds information about the Compute resources on which the jobs are executed by a LS agent.

SLA storage: stores the description of the SLAs.

State info about LS agents: holds the status of LS agents and the loading/usage information about their local resources which is used by the SS agents.

Inter-Grid Agent Information: is collection of meta-SLA bids submitted by various SS agents. This information is stored at Inter grid agent.

4. SLA NEGOTIATION PROTOCOL

We propose an SLA negotiation protocol based on the Contract Net Protocol to allow the SS agents to negotiate the schedule of jobs on the distributed Computer resources with the user agents and the LS agents. The Contract Net Protocol is a high level protocol for achieving efficient cooperation introduced by Smith [6] based on a market-like protocol. Smith took his inspiration from the way that companies organize the process of putting contracts out to tender in public markets. It is the most common and best-studied mechanism for distributed task allocation in

agent-based systems [8, 14]. In this protocol, a decentralized market structure is assumed and agents can take on two roles: a manager and a contractor. The manager agent advertises the task by a task announcement to other agents in the net. In response, contracting agents evaluate the task with respect to their abilities and engagements and submit bids. A bid indicates the capabilities of the bidder that are relevant to the execution of the announced task. The manager agent evaluates the submitted bids, and selects the most appropriate bidder to execute the task, which leads to awarding a contract to the contractor with the most appropriate bid. The contractor assumes the responsibility for the execution of the task. After a task has been completed, a report is sent to the manager. The advantages of the contract net protocol include the following: dynamic task allocation via self-bidding which leads to better agreements; it provides natural load balancing (as busy agents need not bid); agents can be introduced and removed dynamically; and it is a reliable mechanism for distributed control and failure recovery.

The proposed SLA negotiation protocol is a hierarchical bidding mechanism. Due to the dynamic nature of Grid Computing where desired tasks and available resources may be continuously changing, a commitment duration is attached to the negotiation messages, to allow the agents to specify the time windows by which the agents must respond to a given negotiation message. The SLA negotiation protocol involves two different negotiation levels:

- **Meta-SLA negotiation**
- **Sub-SLA negotiation**

4.1 Meta-SLA contents

The meta-SLA contents under negotiation are initiated using information and constraints provided by the user. During meta-SLA negotiation these values may be refined, and information may be added in the final agreed SLA such as proposed compensation package.

The possible user inputs are:

- Required resources over a time period.
- Time constraint: job start time and end time.
- Cost constraint: the cost the user is ready to pay.
- Required execution host information**.**

A meta-SLA may have the following information:

Meta-SLA identifier: an identity tag assigned by the SS agent in order for the system to keep track of SLAs and is later used for accessing SLA repository for information.

Resource information: the required capability of the resource for the application. It is usually very high level description of a resource metric containing resource details such as machine information and the range of processors.

Estimated due date: the time by which the job should be finished.

Estimated cost: it is the maximum cost limit pre-set by the user for any given job execution request. The system will use this parameter to set a limit on its search for resources.

Required execution host information [Optional]: users may choose preferred execution host for their jobs.

Book-keeping information: miscellaneous information.

4.2 Sub-SLA contents

In this level the SS agents negotiate sub-SLA with the LS agents. The SS agents decompose the meta-SLA into its low level resource attributes, sub-SLAs which contain low level raw resource description such as processes, memory processors, etc.

A sub-SLA may have the following information:

- Low level raw resource descriptions such as number of nodes, architecture, memory, disk size, CPU, network, OS, application (if any needed).
- Time constraint.
- Cost constraint.
- Meta-SLA identifier to keep truck of the meta-SLA.
- Storage Requirement - e.g.: 1GB may be used during execution and the output to be stored at Temp directory.
- User with whom the meta-SLA is made.
- LS agent with whom the sub-SLA is agreed.

4.3 Steps of the SLA Negotiation Protocol

The steps of the SLA negotiation protocol are described below (Fig. 2):

a) User agent-SS agents Meta-SLA negotiation steps:

a1) Meta-SLA-request: the user agent submits a meta-SLA to the nearest SS agent to request the execution of a job.

a2) Meta-SLA-announcement: upon receiving the meta-SLA request from the user agent, the SS agent advertises the meta-SLA to the neighboring SS agents in the net by sending a meta-SLA announcement.

a3) Meta-SLA-bidding: in response to the meta-SLA-announcement, each SS agent queries the LS agents' state information database to check the availability of required resources for the job response time to identify suitable set of resources. Cost is then evaluated for each potential resource. The SS agents select the best resources which satisfy the availability and cost constraints of the job. Each SS agent submits a meta-SLA-bid to the user agent and the inter-grid agent. The meta-SLA-bid describes the new estimated response time and cost, and other additional information such as proposed compensation package.

a4) Meta-SLA-award: the user agent, upon receiving the meta-SLA-bids, selects the most appropriate SS agent with the best meta-SLA-bid and sends a notification agreement to its. The selected SS agent stores the agreed meta-SLA in SLAs database. The user now has an agreement with the Grid provider to use its resources**.**

It may happen that none of the LS agents cluster has enough resources for performing the job. Then the user agent can contact the inter-grid agent. Inter-grid agent has collection of meta-SLA bids information submitted by the SS agent of each institution. Inter grid agent makes different combinations of resources from different clusters and then evaluates the cost for each potential combination. After that the Inter-grid agent offers the user agent the best possible combination of meta-SLA bids. If the user agent agrees, it sends a notification agreement to this. . The selected SS agents store the agreed meta-SLA in SLAs database.

b) SS agents-LS agents Sub-SLA negotiation steps:

b1) Sub-SLA-announcement: now the SS agent awarded agreement begins the process of decomposing the meta-SLA by generating one or more sub-SLAs. The sub-SLAs will have low level raw resource descriptions. The SS agent sends the sub-SLA-announcements to the identified suitable LS agents within the cluster under its control.
b2) Sub-SLA-bidding: the LS agents, upon receiving the sub-SLA-announcements, query the resource record for information such as load average, CPU speed, memory size, to identify suitable resources and submit bids to the SS agent.

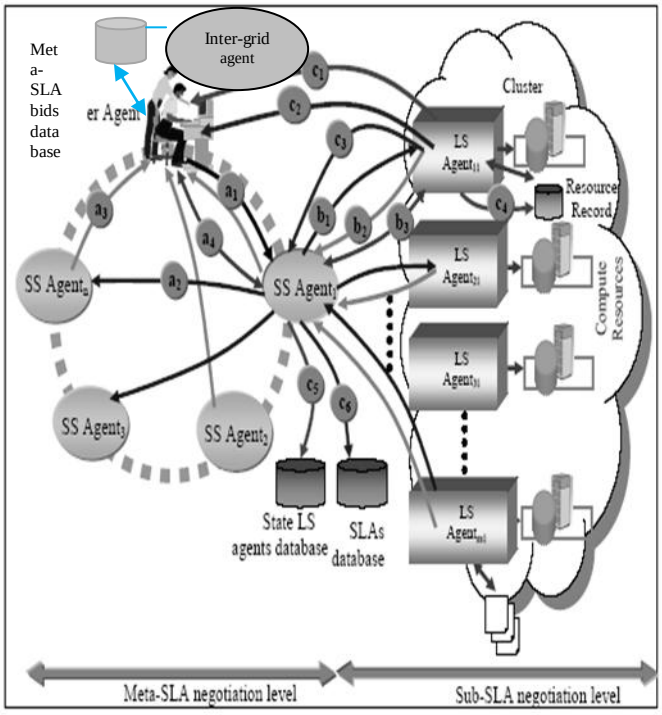

Fig.2. Steps of the SLA Negotiation Protocol

b3) Sub-SLA-award: the SS agent evaluates the best sub-SLA in accordance to time and cost criteria and sends a notification agreement to the most appropriate LS agent with the best resources. The SS agent stores the sub-SLA in the SLAs database and updates the LS agents.

d) Task execution steps

The basic steps for task execution are the following:
d1) The LS agent sends a notification of initiation of job execution to the user agent.
d2) At the end of job execution, a final report including the output file details is sent to the user agent.
d3) LS agent sends a notification end job execution to the SS agent.
d4) LS agent updates information held in the Resource Record database.
d5) SS agent updates LS agents' state information database.
d6) SS agent updates the status of the SLAs in the SLAs database.

5. SLA RE-NEGOTIATION IN THE PRESENCE OF UNCERTAINTIES

Even once a SLA has been agreed, re-negotiation is required for multiple reasons: to extend the running time of an increasingly interesting simulation; to increase the computational resources dedicated to either the simulation, thereby accelerating the experiment, or to the visualization, in order to improve the resolution of the rendering; resources might also be given back when the improved speed or picture quality was no longer required. Also, more generally, resources may fail unpredictably, high-priority jobs may be submitted, etc. In a busy Grid environment, SLAs would be constantly being added, altered or withdrawn, and hence scheduling would need to be a continual, dynamic and uncertain process. The introduction of re-negotiation, permitted during job execution (as well as before it commences) makes the schedule more dynamic, requiring more frequent rebuilding of the schedule.
In the presence of failures, the SS agents re-negotiate the SLAs in failure at the local, inter-cluster and global levels in order to find alternative LS agents to execute the jobs in failure. The basic steps of failure recovery are the following, as shown in Fig.3.

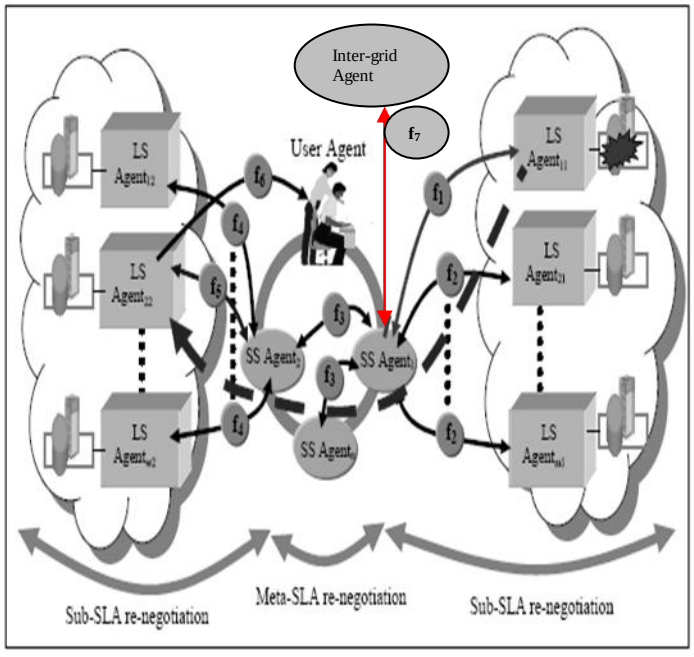

Fig.3. Re-negotiation in the presence of resource failure

f1) LS agent$_{11}$ notifies SS agent$_1$ of the failure.
f2) SS agent$_1$ re-negotiates the sub-SLAs in failure to find alternative LS agents locally within the same cluster by initiating a Sub-SLA negotiation session with the suitable LS agents.

f3) If it cannot manage to do so, the SS $agent_1$ re-negotiates the meta-SLAs with the neighboring SS agents by initiating a Meta-SLA negotiation session with SS $agent_2$ to execute the job in failure.
f4) SS $agent_2$, …, SS $agent_n$ re-negotiate the sub-SLAs in failure to find alternative LS agents.
f5) SS $agent_2$ located LS $agent_{22}$ to execute the job in failure. There are two possible cases for continuing the task execution:
The job may be restarted with the last saved status. Otherwise the job restarts from the beginning.
f6) At the end of task execution, LS $agent_{22}$ sends a final report including the output file details to the user agent.
f7) In case the SS $agent_1$ could not find alternative LS agents at the local and inter-cluster levels, the SS agent1 sends an alert message to the inter-grid agent to inform that the meta-SLA cannot be fulfilled within the current grid. Then Inter-grid agent interacts with other neighboring grids to find the available resources to execute the job.

6. CONCLUSION

In this paper we have proposed a new infrastructure for adaptive and efficient job scheduling on the Grid using multi-agent systems and a SLA negotiation protocol based on the Contract Net Protocol. The SLA negotiation protocol proposed based on the Contract Net Protocol allows the service user agent a flexible negotiation with service provider agent for execution of jobs on the Grid. The SLA negotiation protocol is a hierarchical bidding mechanism which yields many advantages relevant to Grid scheduling, such as dynamic task allocation, natural load-balancing, dynamic introduction and removal of resources, and robustness against failures. Re-negotiation is used to allow the agents to re-negotiate the SLAs in failure.

7. REFERENCES

[1] Choi, H. R., Kim, H. S., Park, B. J., Park, Y. J., Whinston, A.B.: An agent for selecting optimal order set in EC marketplace. Decision Support Systems, 53(2003) 39-58.

[2] Czajkowski, K., Dan, A., Rofrano, J., Tuecke, S., Xu, M.: Agreement-based Service Management (WS-Agreement). Draft Global Grid Forum Recommendation Document (2003).

[3] Keller, A., Kar, G., Ludwig, H., Dan, A., Hellerstein, J.L.: Managing Dynamic Services: A Contract Based Approach to a Conceptual Architecture. Proceedings of the 8th IEEE/IFIP Network Operations and Management Symposium (2002) 513–528.

[4] Ouelhadj, D., Hanachi, C., Bouzouia, B.: Multi-agent architecture for distributed monitoring in flexible manufacturing systems (FMS). Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, USA (2000) 1120-1126.

[5] Ouelhadj, D., Cowling, P., Petrovic, S.: Contract net protocol for cooperative optimisation and dynamic scheduling of steel production. In: Ajith, Ibraham, Katrin, Franke and Mario, Koppen, (eds.): Intelligent Systems Design and Applications, Springer-Verlag (2003) 457-470.

[6] Smith, R.: The contract net protocol: high level communication and control in distributed problem solver. IEEE Transactions on Computers, 29 (1980) 1104-1113.

[7] O'Hare, G., Jennings, N. (Eds.): Foundations of Distributed Artificial Intelligence, Wiley, New York (1996).

[8] Shen, W., Norrie, D., Barthes, J. (eds.): Multi-agent systems for concurrent intelligent design and manufacturing, Taylor & Francis, London (2001).

[9] Abramson, D., Buyya, R., Giddy, J.: A computational economy for Grid computing and its implementation in the Nimrod-G resource broker. Future generation Computer Systems, 18 (2002) 1061-1074.

[10] Cao, J., Kerbyson, D., Nudd, G.: Performance evaluation of an agent-based resource management infrastructure for Grid computing. Proceedings of the First IEEE/ACM International Symposium on Cluster Computing and the Grid (2001) 311-318.

[11] Cao, J., Jarvis, S.: ARMS: An agent-based resource management system for Grid computing. Scientific Programming, 10(2002) 135-148.

[12] Rana, O. and Walker, D.: The Agent Grid: Agent-based resource integration in PSEs. Proceedings of the 16th IMACS World Congress on Scientific Computing, Applied Mathematics and Simulation, Lausanne, Switzerland (2000).

[13] Buyya, R., Abramson, D., Giddy, J., Stocking, H.: Economic models for resource management and scheduling in Grid computing. Concurrency and Computation: Practice and Experience, 14 (2002) 1507-1542.

[14] Wooldridge, M. (eds.): An introduction to multi-agent systems. John Wiley & Sons, Ltd., Chichester, England (2002).

[15] "A Multi-agent Infrastructure and a Service Level agreementNegotiation Protocol for Robust Scheduling in Grid Computing" by 1 D. Ouelhadj, 2J. Garibaldi, 3J. MacLaren, 4R. Sakellariou, 5K. Krishnakumar, 6Amnon Meisels