

A Pre-Coalition Protocol for Minimizing Message Passing and State Information Updates in the A^{3p}viGrid System

Avinash Shankaranarayanan¹, Frank Dehne², Andrew Lewis³

^{1,3}*School of ICT, Institute for Integrated and Intelligent Systems [IIIS],
Griffith University, Australia.*

²*School of Computer Science, Carleton University, Canada*

Avinash.Shankaranarayanan@student.griffith.edu.au, frank@dehne.net, A.Lewis@griffith.edu.au

Abstract:

The primary problem that affects the use of efficient grid or cluster computing systems are the discovery and usage of parallel and high performance grid computing applications and related services. The primary objective of this paper is to build on a protocol that minimizes the number of messages exchanged between the various peers by applying new grid based pre-coalition concepts to the peer to peer grid services model A^{3p}viGrid. Lists of agents are maintained in the form of categorized listings that help in the formation of potential coalitions and negotiations among the agents. By developing a new pre-coalition protocol we are able to simulate the number of messages passed inside the A^{3p}viGrid system that uses predetermined coalition formations to utilize coalition leaders/agents for communication on behalf of a clan or society of agents or peers.

Keywords: Static Coalition, P2P, A^{3p}viGrid, agents, leaders.

1 Introduction

Grid computing has taken a major step forward in gluing together the business and academic worlds with the concept of grid based services. The Web services and peer-to-peer infrastructures have gone through a lot of modifications and the standardization of these technologies is still far from reality. This is primarily because of the role played by data in

application processing. Although XML based technologies and other distributed systems such as peer-to-peer [1] and web services [2] seem to simplify the sharing and perception of data, it has been simply too hard to make use of from an end-user point of view. This situation has produced a new race of technologies each independent of others while mutually beneficial to one another. A trend is apparent of the people collaborating on large projects in the form of virtual groups, organizations and societies with a corresponding trend toward the use of distributed systems. Peer-to-Peer technology is one form of distributed systems that can help layout a decentralized topology for a disjoint set of agents/peers such as that on the Internet. Another technology that can help overcome common grid related problems are multi-agent-based grid systems which allow programs to act intelligently given a set number of environment variables to react to within a given restricted environment.

Automated negotiation systems with self-interested agents have been an important component of distributed problems solving [3, 4] in the field of Multi-agent systems [5, 6, 7]. The primary reason for this is the growth of standards in communication technologies such as KQML [8], Java, etc, on the Internet. These technologies affect the negotiation capabilities of an agent when it comes to negotiating and communicating using different languages and protocols. On the other hand technologies such as web/grid services (OGSA and WSRF) try to deliver a service oriented execution environments under the

de-centralized hierarchy of peer-to-peer computing. Multi-Agent technology provides automation of coalition formation techniques based on operative decision making. Grid technologies have wide-spread use in areas of high computational usage and requirements, such as life sciences and quantum physics. The standardization of these technologies have led to the usage of grid services and their respective specifications such as OGSA[9] and WSRF[10] through the Global grid forum (GGF) [11] where scientists and researchers collaborate and discuss standardization issues for grid computing. In this paper we show how applying grid based coalition formation helps to create a peer-to-peer environment where agents play the vital role of negotiating and minimizing state information updates (message passing) as opposed to existing grid architectures such as Condor [12], and Chinook [13]. This paper is organized as follows. Section 2 talks about coalition formation (Static or Pre-coalition) and how it becomes a vital solution to grid computing; Section 3 talks about the design of the A^{3p}viGrid architecture and its various components; section 4 deals with the pre-coalition formation protocol which defines the way a set of agents would react and negotiate under a specific networked environment and try to provide a framework for job processing using optimal coalitions; section 5 shows us a working example of how many message would be passed when the a real-time job is applied to the protocol structure; and finally section 6 conclude the paper limitations and future directions.

2 Coalition formation in Grids

A coalition, in the context of agent-based systems, is usually defined to be a group of agents that come together to solve a common task or achieve a common objective. Coalition formation has its roots in game theory where a set of agents having similar characteristics, goals or capabilities tend to form a group based on mutual interest to solve a particular problem in a given problem space. In general, with respect to agent based systems and game theory, coalition formation occurs on the fly where agents

tend to form groups to achieve a common goal such as job processing or maximizing their utility value.

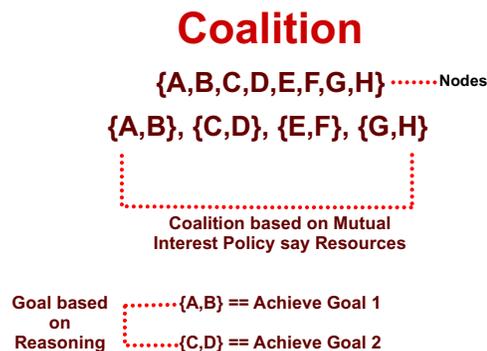


Figure 1: Represents an Example of Coalition in peers A, B, C, D, E, F, G, H

The problem environment has a number of known or unknown constraints or variables which affect the performance and decision making abilities of the agents. Interactions of mutual/self interested agents have been a widely researched area in the fields of microeconomics and game theory [14]. It is possible to define two forms of coalitions, namely *pre-coalition* (Static) and *post-coalition* (Dynamic) with respect to grid computing applied to problems in these fields.. This paper tries to prove that pre-coalitions are better at solving optimization problems and giving better coalition formation solutions to minimize message passing with respect to grid computing.

Pre-coalitions are typically formed on the basis of more persistent common goals and tasks, and are less likely to change from problem to problem.

Post-coalitions, on the other hand, are groupings that are formed to address the needs of a specific task or common objective. Once these tasks are completed, or the common objectives met, dynamic coalitions tend to disband, and re-form in different ways.

3 Functions of the A^{3p}viGrid System

The A^{3p}viGrid Architecture is primarily focused on providing a peer-to-peer based Ad-hoc Multi Agent environment that enables users to remotely join the A^{3p}viGrid system to search for new parallel

programs or services and submit jobs for job processing. Peer-to-peer systems share a common interest in files that are distributed across a wide area network such as the Internet. A centralized directory service is used for the discovery of these remote peers and their respective file structure. A similar concept is utilized here except each peer or a host of peers, tends to provide grid based services such as unique parallel processing environments and programs for serial or parallel execution of programs. In Figure 2 the A^{3p}viGrid system runs on a commonality of services and negotiations based on mutual interests. Coalitions are formed and registered in the APM. In Figure 2 there are two coalition formations namely C1 (A, B, C, D) and C2 (E, F, G, H), based on commonality of services and self-interests of the agents. Let us assume that a service called parallel addition or padder.exe is available in nodes A, B, C, D, E, F, G and H. Each node is

functioning as autonomous peers where a bunch of agents do routine job processing. Based on the commonality of services which here would be the padder.exe service and self interested agents a coalition is formed based on the locality of the peers in a local domain space having very minimal network threshold or turnaround time. That is the systems are located in a local domain space where the nodes can provide Quality of services {QoS} to every peer available in its domain. The primary goal of the A^{3p}viGrid system is to minimize overall communication between nodes; minimize state information updates and the removal of a resource discovery model based on agent based peer manager directory services that tend to provide service and resources discovery between remote peer using the authentication of the peers.

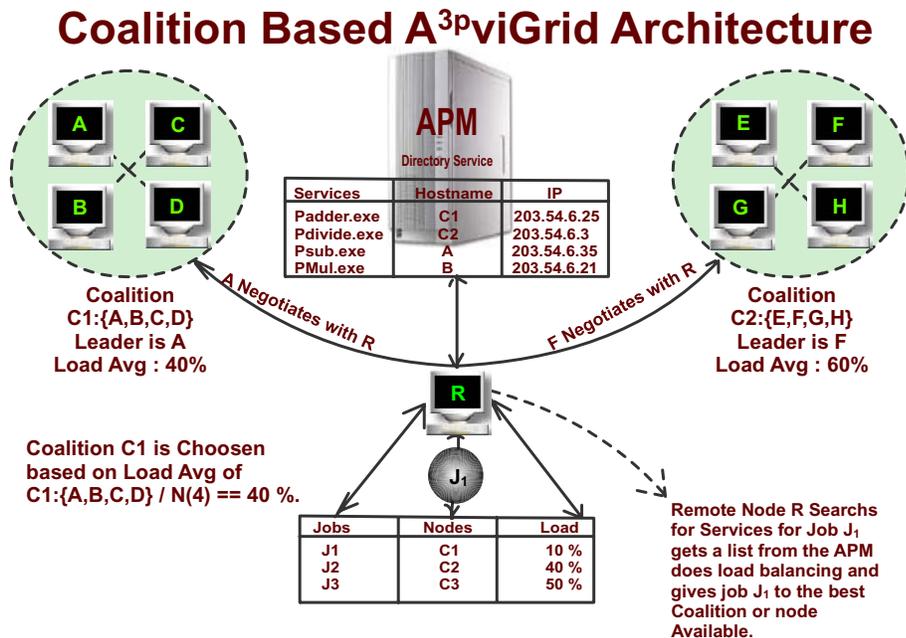


Figure 2: Blue print of the A^{3p}viGrid Architecture

Here the coalitions C1 and C2 are registered in the APM directory service with a commonality of service namely padder.exe. A remote node R has a padder.exe service requirement for a job called J1 based on the load in R. The remote node R then authenticates and searches for peers having the padder.exe services for offloading of the process

J1. A list of coalition and peers are then downloaded into R. The node then starts to negotiate with coalitions C1 and C2 and mutual agreement is negotiated based on the payoff or utility value of the job to be processed. Before negotiations with the coalitions C1 and C2, R tends to do local load balancing based on the load averages of the coalitions and

also the turnaround time or closeness of the coalition domain is calculated and a decision is taken for the offloading of J1 based on the turnaround time and the payoff value offered during negotiations. Once the agents agree upon a template system, jobs are compared with the templates available to do job processing in the closest matching template. Once a satisfactory match is obtained, based on the distance calculated the coalition sends an ACK and job processing takes place followed by the results sent back to remote node R the originator of the job. Thus the need for a resource discovery model is minimized as it is noted that services offered by remote nodes becomes a proof of concept for the satisfying resource requirements of the services offered by the remote peer. The coalition formation in agents tends to minimize the number of messages passed between the nodes and self interested agents form coalitions based on the payoff value. To minimize the state information message flow the load balancing [15] is done on the fly on the local peers with the help of agents. Each agent is assigned a specific task to which it adheres to, with respect to its environment. These agents act upon useful information and logs details such as success and failure of jobs; the average throughput during a particular time of the day etc, about remote nodes used for processing jobs. agents are also used for searching for new programs on local nodes and information is stored locally as well as on the peer manager Server which is called an agent based peer manager or simply APM. The A³viGrid Architecture tends to address the problem of resource discovery and application delivery effectively by applying coalition formation techniques to grid computing.

4 Pre-Coalition Protocol

From Figure 3 let us now see how the protocol works.
 Step 1: Let us assume that sets of agents Say (A, B, C, D) and (E, F, G, H) form a predetermined coalition based on the commonality of interests say padder.exe service.

Step 2: As soon as the coalition is determined the agents starts to poll for a leader who then represents the coalition on behalf of the coalition members.

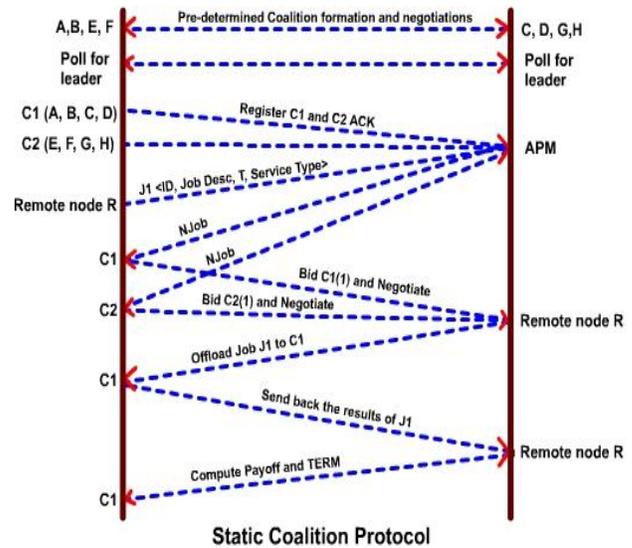


Figure 3: Functioning of the Pre-coalition protocol

Step 3: The leader registers the coalition formation into the APM along with the services offered by the coalition.

Step 4: The APM registers the coalitions or individual agents into the respective service tables for potential job processing.

Step 5: Let us now take R as a remote node that needs to offload its job say J1 from its Job queue.

Step 6: R sends a Job Description message say <ID, Job desc, Service Type> to the APM for the job J1.

Step 7: The APM in turn sends a message (Njob) to all the local coalitions or agents listed based on its service table for the requested service.

Step 8: The coalition leader computes individual pay off values with respect to the job at hand and then assesses the job to determine if it can be performed by the coalition or not.

Step 9: Local load balancing takes place by utilizing algorithms such as 4(2) and 4(2)i where the average load of the coalition is computed and sent to the remote node R.

Step 10: Based on the assessment the coalition leaders then bid for the job in this case J1, and quote their payoff for J1 to the remote node R.

Step 11: Now R decides which local coalitions are most suitable for J1 with respect to payoff, load and turnaround time.

Step 12: Remote load balancing takes place in R by which R determines the best/better node for offloading the Job J1.

Step 13: After negotiations R decides to offload its Job J1 to say coalition C1 and so sends it an ACK.

Step 14: C1 sends R an ACK and does the processing and returns the results back to R.

Step 15: R calculates payoff and sends ACK and TERM to C1.

Step 16: Continue back to Step 1 for processing new Jobs.

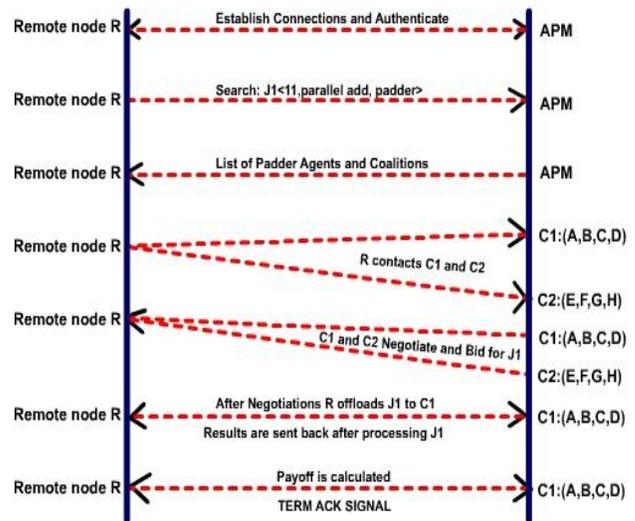
5 Calculating messages passed by example using static coalition protocol

Figure 4 demonstrates how the protocol works for static coalitions.

Step 1: Messages (2): Establish connections between R and the APM

Step 2: Messages (2): Remote node Authentication.

[**Note:** It assumed that the remote node R is already registered with the APM. If it is a new node then more messages might be needed for registering the node]



Static By Example: <11, Parallel add, Padder.exe

Figure 4: Using Padder.exe service as Example

Step 3: Messages (4): Search: Job J1<Job ID, Parallel Addition, Padder.exe> (J1<Job ID, Job Description, Program name>)

Step 4: Messages (8): The APM Sends R a list of agents and Coalitions available for Padder.exe service namely C1 :(A, B, C D) and C2 :(E, F, G, H).

Step 5: Messages (2): R contacts selected coalitions or agents from the List.

Step 6: Message (2): C1 and C2 are two coalitions that are capable of doing job J1 and hence they bid for J1.

Step 7: Message (4): After negotiations with respect to payoffs R offloads J1 to C1 for Job processing.

Step 8: Message (8): After job processing C1 sends the results back to R.

Step 9: Message (4): R calculates pay for C1 and terminated with TERM ACK.

[Note: Here it is assumed that negotiations, lists, data file and results are taken as 8 messages and in real time can be less or more depending on the size of the files under usage. TMsgSize == Number of Messages * two Bytes. (I.e.) TMsgSize = 36 * 2 == 72 Bytes.]

6 Limitations and Conclusion

Any architecture has its limitations which are of no exception pertaining to the A^{3p}viGrid system. The following are some aspects of the system:

- Fault tolerance in centralized directory service APM
- Agent communication languages
- Pre-Coalition based on the similar jobs.

Thus pre-coalition formation seems to be an efficient approach to implementing optimal message passing in grid systems.

7 References

[1] Yatin Chawathe, Sylvia Ratnasamy, Lee Breslau, Nick Lanham, Scott Shenker, Making Gnutella-like P2P Systems Scalable

[2] Open LDAP directory service protocol - Source: www.openldap.org, Last Accessed: 24/6/2004

[3] A.H. Bond and L.Gasser, Readings in Distributed Artificial Intelligence. Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[4] E.Durfee, V. Lesser and D.Corkill. Cooperative distributed problem solving. In A.Barr, P.Cohan, and E.Feigenbaum editors,

The Handbook of Artificial Intelligence, volume IV, Addison Wesley, 1989.

[5] S.Franklin and A.Graesser, "Is it an Agent, or just a Program?" A Taxonomy for Autonomous Agents". In Proceedings of the third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996.

[6] Avinash Shankar, Chattrakul Sombatheera, Aneesh Krishna, Dr.Aditya Ghose, Dr.Philip Ogunbona, "Dynamic Coalition in Agent Aware Adhoc Virtual P2P Interconnect Grid Computing System - A3pviGrid.", ICE-IS 05, June 21st - 24th 2005,

[7] N. Griffiths and M. Luck. Coalition formation through motivation and trust. In Proc. of the Second International Joint Conference on Autonomous Agents and Multiagent Systems 2003 (AAMAS03), pages 17–24, Melbourne, Australia, 2003.

[8] T.Finin, R.Fritzson and D.McKay, A language and protocol to support intelligent agent interoperability. In Proc. Of the CE & CALS Washington '92 Conference, June 1992.

[9] Foster, I., Kesselman, C., Nick, J. and Tucke S. 2002. The Physiology of the Grid. An Open Grid Service Architecture for Distributed Systems Integration. <http://www.globus.org/ogsa/>

[10] WSRF -<http://www.cs.virginia.edu/~gsw2c/wsrp.net.html>, Last accessed – 25 Oct 2005.

[11] GGF - <http://www.gridforum.org/> last accessed 25th Oct 2005, last accessed – 25 Oct 2005.

[12] Condor - <http://www.cs.wisc.edu/condor/>, last accessed – 25 Oct 2005.

[13] Stephen B Montgomery, Tony Fu, Jun Guan, Keven Lin & Steven J M Jones, An application of peer-to-peer technology to the discovery, use and assessment of bioinformatics programs, Nature Methods Journal, 2005, p563

[14] Agentlink.org - European Coordination Action for Agent-Based Computing, University of Liverpool and University of Southampton, <http://www.agentlink.org/>, Last visited 15/4/2004

[15] Avinash Shankar, PDPTA 2005, June 27th – 30th 2005, "Applying Coalition Concepts to Service Oriented Multi-Agent Load Balancing Systems – A^{3p}viLoad", PDPTA 2005.