Learning Reliability Models of Grid Resource Supplying

Cyril Briquet and Pierre-Arnoul de Marneffe

Department of Electrical Engineering & Computer Science, University of Liège,
Montefiore Institute, B37, B-4000 Liège, Belgium
email: {C.Briquet,PA.deMarneffe}@ulg.ac.be

Abstract

Resource exchange between Grid participants is at the core of Grid computing. Distributed bartering is a distributed and moneyless method of resource exchange. Recent work related to distributed bartering has mainly dealt with resource supplying. However, Grid participants still face an unstable resource environment due to the partial and intermittent nature of the exchanged resources. The problem considered in this paper is the unreliability of resource supplying. Though it cannot be totally avoided, a proactive stance may lower its impact in the long run. We propose to explore the reduction of performance variability by improving resource consumption. The goal is to enable Grid participants to identify and avoid unreliable resource suppliers by learning reliability models of resource supplying. A Machine Learning problem is defined and the generated models are applied to select more reliable resources in the hope of improving resource consumption.

1 Introduction

Transient exchange of resources within Virtual Organizations is a key feature of Grid computing [1]. Grid participants naturally require incentives to exchange resources with one another. In this context, so-called market-based methods and the Grid economy [2] constitute an important and promising approach. The Grid economy views access to resources (e.g. computing time) as a commodity that can be bought and sold on a resource market. Its main objective is to attain a market equilibrium between resource supply and resource demand.

Fundamental issues related to the decision making aspects of resource exchange have not fully been explored [1, 3]. Most market-based methods seek to reach market equilibrium. However, they yet have to investigate possible behaviours when an optimal price level has been reached. More research is thus needed. For example, in a market with perfect atomicity, quality of service (e.g. reliability and predictability) could potentially help to differentiate multiple suppliers all offering comparable resources at, or near, some optimal price. Indeed, Schopf & al. pointed out that Grid users *"want not only fast execution times from their applications, but predictable behaviour, and would be willing to sacrifice some performance in order to have reliable run times"* [4]. Taking

into account factors that cannot easily be fitted into resource pricing is, in our opinion as well, a promising idea for future research on resource negotiation.

Most Grid market-based resource exchange methods so far studied, even decentralized ones, rely on some kind of central bank or global currency management organization [5]. We recognize the importance of this approach, especially in Application Service Provider scenarios. At the same time, we argue that there are many noncommercial exchange of resources that are penalized by this dependence upon a central component. Some examples include, but are not limited to: academic departments, subsidiaries of a global corporation, home users willing to pool their resources. Unlike participants in commercial exchange of resources, where sellers and buyers are clearly separated, these Grid participants may act both as resource suppliers (i.e. *sellers*) and resource consumers (i.e. *buyers*).

Such Grid participants do not seek immediate financial profit from the supplying of their own resources. Their main objective is to offer good service to *their own* users, meaning computing jobs under classical constraints (temporal deadlines, makespan, utilization). To reach this goal, they may want to consume resources from other participants when their own resources cannot handle their job load. They may not be willing or able to pay money for this consumption. They may however be willing to supply their own resources when they don't use them. In this sense, there is no absolute necessity for money-based transactions. We argue that both the overhead induced by a centralized banking component and the unwillingness to pay money call for distributed, moneyless exchange of resources. Such a form of commerce or resource exchange may be referred to [2] as distributed bartering.

Distributed bartering middlewares are beginning to emerge but their decision-making aspects have not yet been fully explored. In this paper, we propose to extend a successful middleware, `OurGrid` [6], with learning capabilities to allow Grid participants to consume more reliable resources. Improved resource consumption will in turn lead to a better servicing of computing requests.

The rest of the paper is structured as follows: Section 2 offers references to related work, Section 3 presents the Network of Favors model on which `OurGrid` relies for resource exchange, Section 4 defines the Machine Learning problem applied to generate reliability models and Section 5 concludes and discusses future work.

2 Related Work

A set of Grid participants exchanging resources with one another may create a cooperative environment [2], but we contend that they may also generate a market-based, competitive environment. In fact, there is no reason for any exchange of resource to be restricted to a cycle-stealing model such as the well-known `Folding@home` [7] project. In this context, most Grid participants simply donate resources. In other situations, Grid participants will have to be motivated [2, 5] to contribute resources and may want to compete to consume the best (or most appropriate) resource.

Resource negotiation may be performed with different levels of dynamicity: (1) equal priority (identical resource supplying to/consumption from all Grid participants), (2) statically assigned priority (*out of band* contract between Grid participants), (3) dynamically negotiated priority (explicit or implicit contract between Grid participants). Requirements for scalability and autonomicity call for most dynamic resource negotiation.

Bartering is a form of dynamic, implicit resource negotiation that has recently been gaining more attention. We define bartering as a distributed, moneyless form of resource exchange, which may be competitive. All proposed bartering architectures are not fully distributed. For example, the Faucets middleware [8] is a centralized, zero-sum cluster bartering architecture which relies upon a database located on a central server. Credit is granted to the computing peers to consume resources. A bidding system allows the peers to compete for resource consumption.

Some authors [9] restrict the definition of bartering as a supplying of resource *immediately* followed by a reciprocal supplying. Our understanding of bartering is closer to their concept of *community pattern*, which however relates to a local - rather than a global - computing environment.

`OurGrid` [6], with a focus on a global computing environment, matches pretty well our understanding of bartering. It is a P2P system based on the so-called Network of Favors model. Peers exchange resources with one another and keep their own accounting [10] of resource supplying and consumption. They donate (supply) computational resources (= make favors) in the hope of reciprocal behavior on behalf of the resource consumers.

Another interesting P2P architecture [11], built on the `SHARP` system [12] (secure highly available resource peering), envisions a *"bartering economy as providing the basis for decentralized growth"*. Resource discovery and creation of a secure P2P overlay are thoroughly considered. A simple Tit-for-Tat strategy incites resource exchange. It must be noted that the establishment of trust is explicit and requires the deployment of the `SHARP` system, which is not lightweight. A very interesting fact about this P2P architecture is that bartering is seen as a first step of the evolution of a Grid economy towards currency-based commerce involving the most reliable peers.

`GridIS` [13] is yet another incentive-based P2P resource exchange system which seeks to optimise job scheduling. It provides incentives to the Grid participants to continue consuming/supplying resources but its focus is mainly on supplier decision making. Resource suppliers may be configured to adopt an aggressive (more potential rewards) or conservative (less risk) job acceptation policy. With respect to job acceptation policy, `GridIS` may be related to another effort [14] towards better Grid scheduling. This system proposes a resource exchange market with a focus on job acceptation/admission control policy. The idea is to balance risk and reward by considering the opportunity cost of accepting new jobs.

An important issue that is beyond the scope of this paper is resource discovery (more precisely, the matchmaking of resource supply and consumption in the

bartering system). This aspect of bartering systems (which has been coined the *"double coincidence of wants"*) has been recently studied [15] within a Grid context from a theoretical and graph theory perspective. Most of the bartering middlewares actually being P2P systems (`OurGrid`, `SHARP`-based architecture, ...), they take care of resource discovery by creating overlay networks of peers.

Finally, classical Machine Learning (ML) algorithms (k-NN, Decision Trees, ...) have successfully been used in Grid resource management systems [3]. For example, instance-based learning algorithms have been applied to generate performance estimators based on task input parameters [16] in the well-known `Punch` middleware. In a more recent work, reinforcement learning algorithms have been applied to generate performance estimators [17], with a focus on resource consumption. In the latter case, reliability of resource supplying is not taken into account for consumption and a simple FCFS resource supplying policy is used.

3 Network of Favors Model

As stated in the previous section, `OurGrid` [6] is a P2P system based on the so-called Network of Favors model. The peers (Grid participants) supply their non-busy resources (= make favors) to each other in the hope of future reciprocal behaviour on the behalf of the resource consumers.

Each peer acts as both a supplier and a consumer of resources and maintains its own private bookkeeping of exchanged resources (i.e. accounting of the given and received favors). To avoid ID-changing attacks, the favor balances of each peer are always nonnegative. The main goal of favor accounting is to prioritize resource supply in case of conflicting supply requests: priority in supplying is given to the peers who have contributed the most resources in the past. It should be noted that two accounting models [10] are currently considered: (1) a simple, time-based accounting model (biased towards slower resources), where a favor equals the supply time of the given resource and (2) a more robust accounting model, which weighs the supply time with the estimated relative computing power between consumer and supplier.

Whatever the selected accounting model, there is a known problem of possibly asymmetrical accounting. Under certain circumstances, the supplier updates its favor balance while the consumer does not. Over time, biased accounting will result in degraded resource exchange as there will be a skewed perception of reciprocity.

This divergence of accounting will occur in case of task execution failure. Task failure may happen for multiple reasons: (1) supplied resource preemption for local use, (2) supplied resource failure, (3) temporary departure of supplier from the Grid, ... In these situations, the supplier estimates (rightly so) that it has supplied a resource and lost its use for some time before task execution is cancelled. At the same time, the consumer estimates (again rightly so) that the failed task execution has not been of any benefit. Task execution failures obviously degrade performance.

There is explicitly no assumption of QoS or performance guarantees in `OurGrid`.

However, task replication techniques are proposed to sustain some level of performance. That requires a higher level of resource consumption. An interesting observation [18] is that *"adding capacity and reducing variability are, in some sense, interchangeable options"*. We thereby propose to explore variability reduction as a complementary approach to improve performance stability in bartering middlewares, particularly `OurGrid`.

4 Towards Better Resource Consumption

The core of the considered resource accounting problem is a lack of reliability in resource supplying. Resource unreliability leads to task incompleteness, which causes asymmetrical accounting and delays the delivery of value to resource consumers. The Network of Favors is a satisfactory model, mainly concentrating on resource supplying. However, we argue that accounting for completed work is not enough. The quality of resource supplying should be taken into consideration and modelled so that peers avoid as much as possible to contract with unreliable peers. In other words, even if there is no assumption of QoS, we want to establish a way for peers to preferably contract with other peers that will give them uninterrupted service. This would support the idea of encoding the negotiated quality of experience [1]. We propose to extend the resource exchange model and reduce variability in task execution by better consuming resources. The goal is threefold: (1) to define resource unreliability, (2) to allow each peer to identify unreliable resources/peers and (3) to make each peer choose (when there is a choice) to consume the most reliable resource (which of course depends on the perspective of each peer). The requirements of avoiding explicit negotiation schemes and centralized organization are taken into account by taking advantage of the independence of peers in the P2P system.

The context of a P2P system with independent peers and without trust frames the available management data to what can be independently observed by the peers. The most important information is the outcome of remote task execution: successful, cancelled, rejected. It should be noted that we do not distinguish between task failure and task preemption. Another important information is the time spent between task submission and outcome. This information is important when the task completes successfully (= execution time) as well as when it does not (= failure time).

The negotiation state of a (consumer) peer when submitting a given task is also available: favor balance with every other peer, total resource consumption and supplying with every other peer, success ratio (task execution success count divided by task submission count) of each individual peer. A short temporal window representing recent history may be generated for each of these attributes: resource consumption or supplying time, success ratio, ... for the last X task submissions to/execution for every other peer. The mean favor balance, the mean consumption or supplying time, ... during the last X task submissions to/execution for every other peer may also be computed. The global state of the peer network may also be taken into account by computing means of some

of these variables across all peers (mean recent success ratio, ...). This would allow to estimate the global load of the peer network and distinguish abnormal task execution failure from more regularly unreliable supplying behaviour.

Of course, task-related information such as input parameters and static performance benchmarks may also be used, when available.

All this data may be stored by each peer in a database, with a vector of attributes stored for each task. For a given peer, let

- p_i = another peer
- t_j = a task to execute remotely
- $a(o)$ = vector of attributes of task t_j run on p_i

where o (an object of the database) represents a task t_j executed on the peer p_i and $a(o)$ the vector storing all related data. The database objects may be represented as lines and the attributes as columns.

$a_0(o_1)$	$a_1(o_1)$	$a_2(o_1)$	...	$a_{N-1}(o_1)$	$c(o_1)$	$y(o_1)$
$a_0(o_2)$	$a_1(o_2)$	$a_2(o_2)$	...	$a_{N-1}(o_2)$	$c(o_2)$	$y(o_2)$
$a_0(o_3)$	$a_1(o_3)$	$a_2(o_3)$	...	$a_{N-1}(o_3)$	$c(o_3)$	$y(o_3)$
...						

How should we define and identify unreliability from this data ? We want to predict the outcome of a task submitted to a given peer and its execution time when the outcome is successful. We may define $c(.)$ as the reliability classification function from the perspective of the consumer peer and $y(.)$ as the runtime regression function. They both are functions of the objects of the database (i.e a task), which is adequately projected to exclude the function output from the vectors of attributes. For a given peer, let

- $c(a(o))$ = { successful, cancelled, rejected }
- $y(a(o))$ = task execution time

We define the identification of unreliable resource supplying as a Machine Learning problem:

- given a finite set of examples $(a(o), c(a(o)))$, find a decision model $d(a(o))$ as close as possible to $c(a(o))$.

We may then define an estimator of execution performance, also as a Machine Learning problem:

- given a finite set of examples $(a(o), y(a(o)))$, find a regression model $r(a(o))$ as close as possible to $y(a(o))$.

As the reliability of peers evolves over time, the models must be periodically refreshed or rebuilt.

To generate the decision and regression models, we propose to use classical ML algorithms. With regards to the use of ML algorithms in resource management, what distinguishes our work from past research is the combination of (1) using distributed bartering, (2) taking into account the supplying reliability and (3) using data collected only from independent observation rather than from a

resource monitoring system (local or system-wide). Moreover, the Grid environment considered in this paper is much more dynamic than previously considered cluster environments.

We then propose that each peer prioritizes other peers based on their reliability first and then given their expected performance. The coupling of reliability and performance estimation effectively enables a peer to consume resources from the peers perceived as the most reliable. This makes peers avoid consumption of fast but unreliable resources, which would probably fail during task execution and therefore lead to delays due to task re-execution.

Besides, the absence of assumption of QoS precludes the enforcement of hard scheduling deadlines. However, with our proposal it is possible to estimate if such a deadline will be met if a given task is executed on a given peer.

5 Conclusions & Future Work

This paper has thoroughly presented Grid resource exchange based on distributed bartering, which we define as a distributed, moneyless form of resource exchange. The state of the art has been presented along with a known problem yet to be solved, the asymmetry of resource accounting. This problem is the result of supplying unreliability, in other words, the failure of task execution by supplied resources.

Most resource exchange related work has focused on resource supplying. In particular, prioritizing resource supplying has been considered from various perspectives. Our work focuses on resource consumption and its prioritizing. Related work shows that using Machine Learning algorithms to generate performance estimators is a valid idea. We have therefore proposed to use ML algorithms to generate reliability models of resource supplying. These models may then be used to prioritize resource consumption so as to avoid peers who show good performance but lack supplying reliability.

To conclude, our proposal of using ML algorithms to improve resource exchange combines several interesting features: distributed bartering, the consideration of supplying reliability to improve consumption and the use of management data collected without support of a resource monitoring system.

In our future work, we will explore the performance and precision of different combinations of learning algorithms and attribute vectors. We will also investigate further application of reliability modelling. In particular, it would be interesting for peers to build models of their own supplying reliability and consumption patterns. This would thereby allow us to study the linking of consumption and supplying models and perhaps eventually lead to improved resource exchange performance.

Acknowledgements. We want to thank Krzysztof Rzadca for his support and for pointing out excellent references. We also want to thank Sébastien Jodogne for his support and insights. Finally, we want to thank Claire Kopacz and Elisabeth Spilman for the linguistic review.

References

1. I. Foster, N.R. Jennings & C. Kesselman, Brain Meets Brawn: Why Grids and Agents Need Each Other, *Proc. Int. Conference on Autonomous Agents and Multi-Agent Systems*, New York, USA, 2004.
2. R. Buyya, D. Abramson & S. Venugopal, *The Grid Economy*, Proc. of the IEEE, Special Issue on Grid Computing, 93 (3), New York, USA, 2005.
3. C. Briquet & P.-A. de Marneffe, Grid Resource Negotiation: Survey with A Machine Learning Perspective, *Proc. Parallel and Distributed Computing and Networks*, Innsbruck, Austria, 2006. To appear.
4. J.M. Schopf & B. Nitzberg. *Grids: The Top Ten Questions.* Scientific Programming Journal, 2002.
5. R. Buyya, *Economic-based Distributed Resource Management and Scheduling for Grid Computing*, PhD Thesis, Monash University, Melbourne, Australia, 2002.
6. N. Andrade, W. Cirne & F. Brasileiro. OurGrid: An Approach to Easily Assemble Grids with Equitable Resource Sharing. *Proc. 9th Workshop on Job Scheduling Strategies for Parallel Processing*, Seattle, USA, 2003.
7. `Folding@home` project. `http://folding.stanford.edu/`
8. L.V. Kale, S. Kumar, J. DeSouza, M. Potnuru & S. Bandhakavi, Faucets: Efficient Resource Allocation on the Computational Grid, *Proc. International Conference on Parallel Processing*, Montreal, Canada, 2004.
9. P. Obreiter & J. Nimis, A Taxonomy of Incentive Patterns - the Design Space of Incentives for Cooperation, *Proc. International Workshop on Agents and Peer-to-Peer Computing*, Melbourne, Australia, 2003.
10. R. Santos, A. Andrade, F. Brasileiro, W. Cirne & N. Andrade, Accurate Autonomous Accounting in Peer-to-Peer Grids, *Proc. 3rd Int. Workshop on Middleware for Grid Computing*, Grenoble, France, 2005.
11. B.N. Chun, Y. Fu & A. Vahdat, Bootstrapping a Distributed Computational Economy with Peer-to-Peer Bartering, *Proc. Workshop on Economics of Peer-toPeer Systems*, Berkeley, 2003.
12. Y. Fu, J. Chase, B. Chun, S. Schwab & A. Vahdat, SHARP: An Architecture for Secure Resource Peering, *Proc. ACM Symposium on Operating Systems Principles*, New York, USA, 2003.
13. L. Xiao, Y. Zhu, L.M. Ni & Z. Xu, GridIS: an Incentive-based Grid Scheduling, *Proc. Int. Parallel and Distributed Processing Symposium*, Denver, 2005.
14. D. Irwin, J. Chase & L. Grit, Balancing Risk and Reward in a Market-based Task Service, *Proc. Int. Symposium on High Performance Distributed Computing, HPDC-13*, Honolulu, USA, 2004.
15. C. Ozturan, Resource Bartering in Grids, *Proc. Concurrent Information Processing And Computing*, Sinaia, Romania, 2003.
16. N.H. Kapadia, J.A.B. Fortes & C.E. Brodley, Predictive Application-Performance Modeling in a Computational Grid Environment, *Proc. 8th IEEE Int. Symposium on High Performance Distributed Computing*, Redondo Beach, California, USA, 1999.
17. A. Galstyan, K. Czajkowski & K. Lerman, Resource Allocation in the Grid Using Reinforcement Learning, *Proc. Int. Conference on Autonomous Agents and Multi-Agent Systems*, New York, USA, 2004.
18. R. Leus, *The generation of stable project plans*, PhD Thesis, Catholic University of Leuven, Leuven, Belgium, 2003.