

Scalable Resource Annotation in Peer-to-Peer Grids

Nazareno Andrade^{1,2}

Elizeu Santos-Neto²

Francisco Brasileiro¹

¹Universidade Federal de Campina Grande

²University of British Columbia

{nazareno,fubica}@lsd.ufcg.edu.br,elizeus@ece.ubc.ca

Abstract

Peer-to-peer grids are large-scale, dynamic environments where autonomous sites share computing resources. Producing and maintaining relevant and up-to-date resource information in such environments is a challenging problem, due to the grid scale, the resource heterogeneity, and the variety of user demand. This work proposes a peer-to-peer annotation approach where users can freely annotate available resources as a solution to this problem. We advocate that the proposed approach (i) is scalable, as the job of updating the resource information is divided among users; (ii) will improve resources' utilization, by reducing the amount of resources which are allocated to users without matching their applications constraints; and (iii) will allow resource allocators to increase users' utility, leveraging access to more detailed preference descriptions. The paper also discusses the challenges in implementing and deploying such approach and present solutions to tackle these challenges.

1 Introduction

A grid is a federation of sites sharing computational resources. A peer-to-peer grid is a free-to-join grid formed of potentially unknown and untrusted participants. By lowering the joining cost, these systems seek to attain large scales and to cater for the computational demands of small and medium-sized institutions which do not participate in large-scale grid deployment efforts [9]. Some examples of systems implementing peer-to-peer grid computing are Our-Grid [5], Cluster Computing On the Fly [12], ParCop [7] and the Self-organizing Flock of Condors [8].

In each site of a peer-to-peer grid, resource *administrators* operate a *resource scheduler*, which allocates the site's resources among grid users. Each *user* requests resources from these schedulers by submitting applications to her *grid broker*.

An application submission carries its *constraints* (e.g. operating system, minimum amount of memory) and *preferences* (e.g. the more memory the better) for resources. Grid brokers use this information to communicate the user demand to the resource schedulers. Upon receiving a new request, a resource scheduler executes a matchmaking between the *constraints* of all received requests and the attributes of the resources it controls. After this matchmaking, the resource scheduler divides the available resources among their matching requests accordingly to its policies and users' *preferences*.

Typically, both resource characteristics and the attributes users resort to specify constraints and preferences change over time. For example, resources might have new software deployed, operating system patched or hardware updated. The users, in turn, have constraints and preferences which depend on the application they are running during a given time frame. In this scenario, inaccurate resource information may lead to two undesired side effects: first, it may reduce resource utilization, as resources which actually do not match applications' constraints might be allocated to them and will be unable to perform useful work; and second, it may impair users' utility, as resources might be allocated to applications without taking into account users' preferences.

Relying on resource administrators to maintain a catalog that accurately describes resources in such a dynamic environment is neither scalable nor agile enough: the number of resource administrators which must take action for an information about all resources in the system to be available or updated is proportional to the size of the grid.

This paper proposes a collaborative resource annotation mechanism where users can annotate resources with arbitrary attributes and values and share this information. The rationale behind the proposed approach is to leverage user annotations to feed site resource schedulers with more precise information. Also, by sharing their annotations, users make the annotation process more efficient. We argue this approach is scalable and provides valuable information to resource schedulers to maximize resource utilization and

users' utility. Additionally, we claim that enabling users to annotate grid resources opens two new possibilities: i) it allows administrators to view grid resources from the perspective users care about; and ii) it provides users with more information to understand unexpected resource behavior, making failures easier to diagnose.

In the next section, we discuss in detail our approach and contrast it with previous uses of user-generated information. Although enabling users to share their resource annotations has the potential of enabling scalable and agile resource information management, it also unveils a series of challenges, which we discuss together with alternatives for tackling them in Section 3. Section 4 presents efforts related to our approach and Section 5 our final remarks.

2 Peer-to-peer resource annotation

The problem of classifying a large set of items to satisfy different users' requirements and preferences is not new. Large content-sharing Web systems such as Flickr [3], YouTube [6] and Delicious [2] tackle the issue of classifying items by putting the users in charge of describing the items they produce or consume. In these systems, users classify resources according to arbitrary characteristics and share the metadata they generate. The emergent classification is useful to group items and to navigate through the large set of items.

Similar to collaborative classification systems, we propose enabling users to annotate grid resources and to share such annotations. The annotation is performed by the user's grid broker, automating the process, and shared through a grid information system, which is already deployed on the grid. Resource schedulers, in turn, use this information to optimize resource allocation.

To illustrate the process of annotating resources and the usage of such annotations, let us use an example (Figure 1). Consider a user, *Sarah*, whose molecular dynamics application, similar to Folding@Home [1], requires at least a given version of the *R* software and runs faster on computing resources which have a GPU (Graphics Processing Unit) that supports CUDA [4]. Assume also that resource administrators did not specify whether the resources in the grid have these characteristics, because they did not anticipate the need for such specific characteristics.

In our proposed approach, Sarah informs her constraints and preferences to her resource broker. The grid broker requests resources from the grid that satisfies the constraints and preferences passed by Sarah and resource schedulers use the attributes specified in Sarah's request to perform resource allocation. An important point is that resource schedulers use this information *opportunistically*: the information about the attributes in Sarah's requests might not be available for all resources a scheduler controls. In the cases

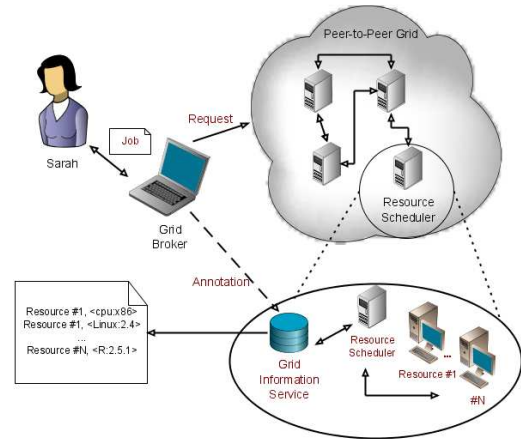


Figure 1. Peer-to-Peer Grid where each site is represented by a resource scheduler (peer)

where information is not available, the scheduler must perform as it does today: allocate resources without taking into account Sarah's constraints or preferences for which there is no information available.

Upon receiving the allocated resources, Sarah's broker starts to use resources which have the attributes that specify a match to Sarah's constraints defined. For the remaining resources, the broker will first try to obtain the relevant information and annotate them. Those which are found to match the constraints will then be used.

Note that missing resource information is counterproductive for both resource schedulers and users. From the resource scheduler standpoint, it might decrease system utilization, as resources which do not match Sarah's constraints might be allocated to her, and will demand re-allocations when Sarah discovers the mismatch. From the user's point of view, resources need first to be annotated to then potentially be used to run the application, what introduces overhead in the application execution. The benefits of available information observation motivate *annotations sharing* among users: writing their annotations in the grid information service, users can minimize the number of resources each of them needs to annotate.

In fact, because resource schedulers benefit from information availability, resource administrators have an incentive to act as annotators and share their annotations as well. Note, however, that the system does not depend on resource administrators to produce information; it can work solely with the peer-to-peer information produced on the edges of the system.

Our example so far illustrated the three main benefits that the proposed approach offers:

1. *Scalability*: peer-to-peer annotation divides the cost of information production, which was previously put on resource administrators, among users. This labor division makes resource annotation highly scalable: information producers are added with every addition of information consumers.
2. *Increased utilization*: increasing users' expressiveness when specifying constraints allows the system to reduce the proportion of resource allocations which do not match users' constraints.
3. *Increased user utility*: increasing users' expressiveness when specifying preferences allows them to provide otherwise unavailable information for resource schedulers. This information could be used to rank resources and maximize the utility of competing users.

Additionally, we envision two positive side effects of collaborative resource annotation in peer-to-peer grids:

4. *Improved system visualization*: the classification of resources according to users' criteria provides a new view of the grid which might better inform users and administrators. Users might use this view to better adapt their applications to the grid (e.g. porting an application to a more popular platform). On the other hand, administrators can use the user-generated information and usage logs to understand what resource characteristics are most valued by users.
5. *Easier fault diagnosis*: failures are common and hard to diagnose in complex distributed systems such as grids [10, 14, 20]; providing users with more information about resources on which an application fails allows them to better contrast these with resources on which the application runs correctly. This facilitates fault diagnosis either by the user or by automatic diagnosis methods (e.g. [22]).

Finally, our proposed approach can be implemented with minimal modifications to the existing infrastructure of peer-to-peer grids. The necessary modifications are that, first, the grid information service allows users to publish information, and second, that resource schedulers consider resource information opportunistically, as some resources might not have been annotated. We note there are already resource schedulers capable of using both users' constraints and preferences available for peer-to-peers grid (e.g. [16]).

3 Challenges

Although collaborative resource annotation brings multiple advantages, the characteristics of peer-to-peer grids pose challenges in deploying such solution, among which we

highlight: (i) the potential high number of unknown items a user might deal with simultaneously, (ii) the dynamic nature of the resources being classified, and (iii) the incentive structure of the system.

First, the scale of grids complicate matters because users may have to deal with hundreds or thousands of resources at once. This renders manual annotation unfeasible; users must be able to annotate resource collections in batch. Hopefully, the grid broker is already enabled to run operations on all resources a user obtains in parallel. A simple solution is that *users express to their brokers relevant resource characteristics as automated tasks* to be run on resources. The specification of such tasks in a common language would also ease sharing the semantics of annotations. Additionally, a user could inform the broker how the outcome of running his application provides information about the resources on which it ran: if the application fails with some exit values, it means that the environment was not set for the application, while succeeding in running the application implies that the resource has some characteristics.

A second issue is that grid resource characteristics are dynamic at some timescale. Over time, the software installed on a resource or its hardware configuration might change, making the resource information inaccurate. Such dynamic changes impact the usefulness of our approach, as stale annotations make the resource description inconsistent. *Enabling brokers to take into account the age of information* is an approach to address this challenge. Thus, brokers have to confirm available information periodically, re-annotating resources, and inform resource schedulers about their tolerance regarding the age of available annotations. We note that even with this counter measure, our approach is only suitable for resource characteristics which change in a timescale that is large enough for users to benefit from annotations. Complementary mechanisms are needed if highly dynamic information such as CPU load or number of open sockets in a resource are vital to popular applications.

Third, grid users compete for shared resources. Although users benefit from shared information, this competition might encourage users to try to make resources less contended by not sharing or producing misleading annotations. *Annotation sharing can be encouraged by a reciprocity mechanism*: the system can enforce that a user can only access information from other users to whom she makes her information available. To deal with possible misleading annotations, *users must take into account their trust on the authors of shared information*. This can be addressed by a reputation mechanism. A user could build trust on his peers based on the similarity between his opinion and theirs (c.f. Walsh and Sireer [21]). Additionally, offline ties already present in the users' scientific collaborations could be used to gain trust in shared information.

4 Related Work

Several studies consider the problem of matchmaking between requests and grid resources either comparing user requirements and resource information (e.g. [16, 11]) or using semantic mechanisms (e.g. [19, 13]). However, these studies assume resource information will be available. Our work is complementary to them, as it focuses on providing accurate information for matchmaking.

Similarly, our peer-to-peer resource annotation approach complements the Semantic Grid vision [17], as it projects the realization of a necessary component in this ecosystem. We have not discussed the use of more structured information, such as ontologies, however our approach could be used for users to provide more structured information with minimal changes.

An approach similar to ours in a different context is Tribler, a BitTorrent client that allows users to tag content. Tribler leverages the tags to connect users with similar interests [15]. A peer-to-peer grid using our approach could explore a similar idea, introducing users which have similar interests, so that they can take advantage of off-system communication.

We also explore file annotations with custom metadata to enable communication across layers in storage systems [18]. At a higher level, resource annotation in peer-to-peer grids and cross-layer communication through custom metadata in storage systems target the same opportunity: enabling better communication of resource characteristics between users and infrastructure allows the system to optimize for the requirements which matter most to users.

5 Final remarks

In this paper, we proposed an approach for peer-to-peer resource annotation on peer-to-peer grids inspired by collaborative classification Web communities. The anticipated benefits of this approach are (i) scalability, (ii) increased resource utilization, (iii) increased users' utility, (iv) improved system visualization and (v) easier fault diagnosis. We also discussed a number of challenges in implementing this approach and presented solutions to tackle such challenges. We believe the ideas discussed here can be of great use to a number of systems which have been realizing the peer-to-peer grid vision.

A natural next step of this work is a quantitative analysis of the benefits of our approach in a deployed peer-to-peer grid. Another important question which arises is the definition of the applicability scope of this approach. Studies analyzing whether this approach can complement the mechanisms already in place in service-oriented grids or in platforms which are centrally administered such as PlanetLab

could shed light on this issue, putting in perspective the limits of the usefulness of the proposed approach.

References

- [1] Folding at home. <http://folding.stanford.edu>.
- [2] del.icio.us. <http://www.del.icio.us>, 2008.
- [3] Flickr. <http://www.flickr.com>, 2008.
- [4] NVidia CUDA. <http://www.nvidia.com/cuda>, 2008.
- [5] Ourgrid. <http://www.ourgrid.org>, 2008.
- [6] Youtube. <http://www.youtube.com>, 2008.
- [7] N. A. Al-Dmour and W. J. Teahan. Parcop: A decentralized peer-to-peer computing system. In *ISPDC/HeteroPar*, pages 162–168, Cork, Ireland, July 2004.
- [8] A. R. Butt, R. Zhang, and Y. C. Hu. A self-organizing flock of condors. *J. Par. Dist. Comput.*, 66(1):145–161, 2006.
- [9] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, and M. Mowbray. Labs of the world, unite!!! *Journal of Grid Computing*, 4(3):225–246, 2006.
- [10] G. Kola, T. Kosar, and M. Livny. Faults in large distributed systems and what we can do about them. In *Euro-Par*, pages 442–453, Lisbon, Portugal, August 2005.
- [11] C. Liu and I. T. Foster. A constraint language approach to matchmaking. In *RIDE*, pages 7–14, March 2004.
- [12] V. Lo, D. Zappala, D. Zhou, Y. Liu, and S. Zhao. Cluster computing on the fly: P2P scheduling of idle cycles in the internet. In *IPTPS' 04*, pages 227–236, 2004.
- [13] S. A. Ludwig and S. M. S. Reyhani. Semantic approach to service discovery in a grid environment. *J. Web Sem.*, 4(1):1–13, 2006.
- [14] R. Medeiros, W. Cirne, F. V. Brasileiro, and J. P. Sauvé. Faults in grids: Why are they so bad and what can be done about it? In *GRID 2003*, pages 18–24, 2003.
- [15] J. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. van Steen, and H. Sips. Tribler: A social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 2007.
- [16] R. Raman, M. Livny, and M. H. Solomon. Matchmaking: An extensible framework for distributed resource management. *Cluster Computing*, 2(2):129–138, 1999.
- [17] D. D. Roure, Y. Gil, and J. Hendler, editors. *Guest editors' Introduction: EScience*, volume 19. IEEE, 2004.
- [18] E. Santos-Neto, S. Al-Kiswany, N. Andrade, S. Gopalakrishnan, and M. Ripeanu. Enabling cross-layer optimizations in storage systems with custom metadata. In *17th HPDC*, June 2008.
- [19] H. Tangmunarunkit, S. Decker, and C. Kesselman. Ontology-based resource matching in the grid - the grid meets the semantic web. In *International Semantic Web Conference*, pages 706–721, October 2003.
- [20] D. Thain and M. Livny. The ethernet approach to grid computing. In *HPDC*, pages 138–151, June 2003.
- [21] K. Walsh and E. G. Sirer. Experience with an object reputation system for peer-to-peer filesharing (awarded best paper). In *NSDI*, San Jose, California, USA, 2006.
- [22] H. J. Wang, J. C. Platt, Y. Chen, R. Zhang, and Y.-M. Wang. Automatic misconfiguration troubleshooting with peerpressure. In *OSDI*, pages 245–258, 2004.