

Provisioning and Scheduling Resources for World-Wide Data-Sharing Services

Alexandru Iosup, Paweł Garbacki, D.H.J. Epema

Faculty of Electrical Engineering, Mathematics, and Computer Science, Delft University of Technology

Mekelweg 4, 2628 CD, Delft, The Netherlands

Email: {A.Iosup, P.Garbacki, D.H.J.Epema}@tudelft.nl

Members of the Virtual Institute on Grid Resource Management Scheduling of CoreGRID

Abstract

Grid computing is becoming the natural way to aggregate and share large and heterogeneous sets of resources. However, grid development and acceptance hinge on proving that grids reliably support large communities of users, and their real applications. In this paper we assess the ability of existing grid infrastructures to provision resources for a class of applications with numerous potential users, namely the class of world-wide data-sharing services. For this purpose, we first analyze the requirements of this class of applications, and match them against the existing spare capacity in three existing large-scale grid environments, namely OSG/Grid3, NorduGrid, and CERN LCG. We then address the need to allocate insufficient resources to world-wide data-sharing services by introducing and assessing through trace-based simulation five domain-specific scheduling policies. Our findings support the idea that grid technology could be leveraged with great success for existing and future world-wide data-sharing services, without impacting the level of service for the currently existing load.

1 Introduction

Grid computing's long term goal is to become the standard way to share heterogeneous resources, and to aggregate them into virtual platforms, used by multiple organizations and independent users [7]. With the grid infrastructure starting to meet the requirements of such an ambitious goal [1, 19, 23], the current evolution of grids undoubtedly depends on proving that it can run real applications, that are used by large communities of users. In this work we focus on a class of such applications, namely the class of dynamic world-wide data-sharing applications.

World-wide data-sharing applications are currently encompassing millions of concurrent users worldwide¹ [13]. Simple applications operating on top of the network layer ensure that popular data are being shared across large communities of people [18]. However, with the dynamic problem of users joining and leaving the collaborative network at relatively short intervals (*churn*) growing with the collaborative network size [2], it is becoming increasingly difficult to guarantee a reliable infrastructure. Moreover, the quality of service demanded by the collaborative applications rises yearly, e.g., the average network bandwidth de-

¹Sllyck.com lists size information for many such applications.

manded by these applications has risen from 240 Kbps to 500 Kbps in just two years [21, 13]. Thus, an important research question gains weight at the same pace with the remarkable growth of these applications: "How can world-scale data-sharing services be reliably provided?". In this paper we seek an answer to this question, and find it as a combination of two issues: the SuperPeer concept, and the resource provisioning and scheduling mechanisms. Our main contribution is three-fold:

- We identify the resource provisioning requirements of a class of SuperPeer-based world-wide data-sharing applications, and assess the ability of currently deployed grid infrastructures to accommodate their provisioning needs, *without* adverse impacts on the level of service provided by the grids for their currently existing load;
- Having shown evidence that the current and future needs of world-wide data-sharing applications exceed the available infrastructure, we devise five domain-specific resource allocation (scheduling) policies;
- We assess through trace-based simulations the capabilities of the five scheduling policies, if they were to be deployed in three major cluster-based grid infrastructures, namely OSG/Grid3, NorduGrid, and CERN's LCG, or in an all-encompassing grid meta-infrastructure.

2 World-Wide Data-Sharing Services

In this section we present a brief overview of the world-wide data-sharing services as viewed through the lens of a motivating scenario, the concept of SuperPeers (SP), and a service model that will be used throughout this work.

2.1 Motivating Scenario: Large-Scale P2P File-Sharing Environments

World-scale P2P file-sharing environments comprise a large number of users collaborating in data sharing [18]; the data is present usually in the form of system-independent files. The P2P file-sharing applications (P2P-FS-Apps) offer three main mechanisms to their users: connecting to the network (from hereon used interchangeably with *bootstrap*), delivering data, and searching for data.

Most P2P-FS-Apps are deployed such that they provide bootstrap services on some dedicated resources, with

high availability. The users, also called peers, connect to the network through these services; with connections taking each a very short time, few dedicated resources can handle very high demand. The bootstrap connection is usually quickly dropped in favor of other peers; still, the P2P-FS-Apps display poor quality of service during failures of the bootstrapping services [21].

Following bootstrapping, peers spend most of their collaborative network participation time engaged in data transfers. Here, the main problem preventing the effective use of the transfer infrastructure is the *free-riding* phenomenon, that is, the fact that the majority of peers attempt to obtain data without further sharing it [24].

Searching for data can prove an important time factor for peers only in networks where the content is not reliable: the phenomenon affects on average from 0% [21] to 50% [15] of the shared data in real environments, depending on the application anti-pollution mechanisms. Note that there exist deployed solutions which guarantee simultaneously 0% pollution rate, low or none free-riding, and have proven capable to serve millions of users [21, 13] daily.

2.2 The Role of SuperPeers

The performance, reliability and scalability of the services provided by wide-area cooperative systems is determined by the characteristics of the participating nodes. Node properties such as churn rate, hardware characteristics, resource sharing policy, uptime distribution, and connectivity directly affect the quality of service offered by the system [21].

A well known mechanism for improving the quality of service offered by the system is based on exploiting high capacity, high availability, nodes, commonly called SuperPeers [27], by assigning them additional responsibilities in the system. SuperPeer-based approaches have been successfully used to improve the search performance [16] and content availability [26] in content sharing networks, as well as to improve the network topology maintenance and fault tolerance [8].

As increasing the population of SuperPeers consequently improves the overall system performance, *the main issue in providing a good level of service for SuperPeer-based systems is to provision enough resources for the needed population of SuperPeers.*

3 Resource Provisioning and Scheduling

In this section, we present and discuss resource provisioning and scheduling mechanisms, and metrics for assessing their performance.

3.1 Resource Provisioning

In Section 2.2 we have argued for the importance of provisioning resources for SuperPeer-based applications. We

Table 1. The service model parameters. CC, FC, and SP stand for collaborative community, flashcrowd, and super-peer, resp.

Param.	Description	Typical Values
λ_{CC}	CC creation rate	50-100 CCs/day
σ_{CC}	Average CC size	0.5-2 KPeers
δ_{CC}	CC duration	15-30 days
Π_{CC}	Capacity required by a CC (depends on σ_{CC} ; α_{CC} is a service dispatch rate constant)	σ_{CC}/α_{CC} SP $\alpha_{CC} = 1000$ non-exclusive
λ_{FC}	FC creation rate	1-10 FCs/day
σ_{FC}	Average FC size	10-100 KPeers
δ_{FC}	FC duration	3-7 days
Π_{FC}	Capacity required by an FC (depends on σ_{FC} ; α_{FC} is a service dispatch rate constant)	σ_{FC}/α_{FC} SP $\alpha_{FC} = 20,000$ exclusive

now address the question *"How many resources should be provisioned for a given world-wide data-sharing service?"*.

To answer the question, we first model the services as multiclass M/G/1 queues, with two job classes: the collaborative environments class, and the flashcrowds class. The flashcrowd class corresponds to the high demands imposed by a large community appearing suddenly (interest peak). Each job is a SuperPeer application that consumes computational and bandwidth resources in an exclusive (space shared) or non-exclusive (time-shared) mode.

The collaborative environments class is characterized by the rate with which new environments need to be created, by the size of the collaborative community, by the duration of the collaborative community, and by the capacity required by a collaborative community. The flashcrowds (large-scale collaborative models) class is characterized by the same parameter types. In our service model, a collaborative environment of maximum size and duration is generated at the end of each flashcrowd, to emulate the long tail in which the flashcrowds' number of users distribution typically degenerates. This approach is consistent with the mid- and end-phases of the analytical flashcrowd model by Massoulié and Vojnović [17], and with real scale observed in flashcrowd measurements [20, 21]. Similarly to the real-life situation, all service jobs have a fixed lifetime, which represents the span of the users interest. Table 1 summarizes the parameters of our service model, and their typical values, as observed in previous measurement studies of large-scale P2P networks [21, 10, 13] and large-scale multimedia streaming networks [20]. The service dispatch constants (α_{CC} and α_{FC}) show how well the obtained capacity is used by the SuperPeer to dispatch requests, and are much lower for collaborative communities (shared resources), than for flashcrowds (exclusive resource access).

We can now give an answer to our provisioning question: given a predicted size of the data-sharing community, and a class to which this community belongs, e.g., normal, or flashcrowd, the number of SuperPeers, and from it the number of needed resources, can be computed. However, the number of resources that can actually be obtained depends also on the number of free resources in the system, for which there also will not exist demand, such that the level of service offered by the grid systems to their already existing users is not affected.

3.2 Scheduling Policies

Consider the motivating scenario (see Section 2): many collaborative communities compete to obtain the needed resources, on which to deploy SuperPeers. When the number of available resources is insufficient, a scheduling policy is used to select the communities that will receive service. Below we describe five scheduling policies, addressing the domain-specific issue of balancing the number of communities and of peers that do not receive service. We also discuss the relative merits of a random scheduling policy.

Random Assignment Policy The random policy (Random-Pol) randomly assigns resources to communities. The expected outcome is that the resources are fairly allocated: no community obtains more resources than another on the basis of its properties. An expected advantage is the relative ease of enforcing this policy: the complexity of the assignment is $O(1)$. The expected downside of this policy is that the random assignment may leave highly populated communities without resources.

Communities First Policy The communities first policy (CommF-Pol) assigns resources such that the highest possible number of communities receive the resources needed to operate. The CommF-Pol policy maximizes therefore the *number of serviced communities*. This policy operates in $O(C)$ time, with C the number of communities, which, given realistic assumptions (see Section 4.2), means that the policy is scalable. The main downside of this policy is that highly populated communities, which consequently require many resources to operate, will remain unserved in case of resource shortages.

Peers First Policy The peers first policy (PeerF-Pol) assigns resources such that the highest possible number of peers receive the resources needed to operate. This policy maximizes therefore the *number of serviced peers*. The PeerF-Pol policy operates in $O(C)$ time, with C the number of communities; similarly to CommF-Pol, the PeerF-Pol policy is scalable. The main downside of this policy is that un-populated communities are *unattractive* for this policy, and will remain unserved in case of resource shortages.

Oldest Community First Policy The oldest community first policy (OldF-Pol) assigns resources such that the

oldest communities receive the resources needed to operate first. This policy rewards customer fidelity; the complement of the OldF-Pol, the **Newest Community First Policy** (NewF-Pol) can be adopted to attract new users; from this work's perspective, it has identical statistical properties to the OldF-Pol, and will not be independently investigated. The OldF-Pol policy operates in $O(C)$ time, with C the number of communities; similarly to CommF-Pol, the OldF-Pol policy is scalable. The main downside of this policy is that new customers will remain unserved in case of resource shortages.

Non-Exclusive Resources First Policy This policy (NonExclF-Pol) relates to exclusive resource allocation: when communities may have the option to share allocated resources, allocating resources to communities actually willing to share may ensure that a higher number of communities are serviced. The policy operates in $O(\max\{C, R\})$ time, with C the number of communities, and R the number of aggregated resources; for multi-cluster grids, R , the number of clusters, is in realistic cases on the order of hundreds or thousands, and the policy is scalable.

3.3 Service Metrics

For the performance evaluation of various systems addressing the world-wide data-sharing services we define the following domain-specific service metrics.

The Community Coverage metric (CommCov) is defined as $CommCov = |C_s| / |C| \times 100$, where C is the overall set of collaborative communities, and C_s is the set of successfully serviced collaborative communities. The CommCov metric evaluates the coverage of the service provided to the collaborative communities.

The Peer Coverage metric (PeerCov) is defined as $PeerCov = |P_s| / |P| \times 100$, where $P = \{p, p \in C\}$ is the overall set of peers from all collaborative communities, and P_s is its counterpart for C_s . The PeerCov metric evaluates the coverage of the service provided to the *members* of the collaborative communities (peers). Note that PeerCov can widely differ from CommCov, e.g., when resources are in shortage and the resource allocation mechanism prioritizes large communities; then, a large number of peers, but a small number of communities will be served.

The Overall Service Coverage metric (AllCov) is defined as $AllCov = \min\{CommCov, PeerCov\}$, that is, it evaluates the percentage of time that all service requests were fully satisfied.

The Extra Service Cost metric (XCost) is defined as $XCost = XRes + \alpha^{xc} \times |SysQRes|$, where $XRes$ is the number of extra resources needed to fulfill all pending service requests, $SysQRes$ is the number of resources already reserved by other users of the grid system, over the existing number of resources, and α^{xc} is a constant quantifying the penalty of requesting resources that are already

reserved by other users. The $XCost$ quantifies the cost required to provision more resources than normally available, with a system occupancy penalty.

4 Experimental Setup

In this section we present our experimental setup.

4.1 Simulated environment

We have built a custom cycle-based simulator, with the simulation step of 1 day. We consider that resources are provisioned from already deployed grid systems, without interfering with the existing user's load. We consider that all resources within a grid can be considered as a single pool of resources; this corresponds to the conservative assumption that all resources can accommodate the requirements of their running SuperPeers (note that the main resource consumption for SuperPeers concerns bandwidth, and thread and socket descriptors, rather than computational power).

During each cycle, we simulate the creation (arrival) of several world-wide data-sharing services, and assess whether the simulated grid system can provide the needed resources; if the system cannot provide sufficient resources, different scheduling policies yield different values for the performance metrics. We use the tools provided within the GRENCHMARK framework [12] to guarantee that *the same workload is generated for the evaluation of each of the six scheduling policies*.

The reason behind this rather coarse granularity is many-fold. First, this granularity is suitable for our service model (see Section 3.1); note that for SuperPeers with uptimes of up to a few hours there is no need for external resource provisioning, as they already exist in real data-sharing communities [21]. Second, as can be seen in Section 4.2, this is a level at which the outliers in our data can be smoothed out. Third, this allows us to keep our service model simple, and to not take into account service migration costs.

4.2 Experimental data

To extract the experimental data needed for describing the deployed grids' sizes and their existing user's load, we use three existing grid traces, and a fourth *fabricated* trace. To smooth out resource availability of only a couple of minutes/hours we consider only the number of free resources over a whole day.

While we build on the tools with which we have analyzed one of the grid traces in our previous work [11], the type of grid data required for this article has not been analyzed elsewhere (to the best of the authors' knowledge). Using newly developed automatic analysis tools, we have analyzed the resource availability exhibited by each of the grid traces.

Grid traces The first grid trace comes from the OSG/Grid3 [11]. We have analyzed traces recorded by the grid-level scheduler corresponding to one of the largest

Virtual Organizations (from hereon, VO), the USATLAS, for over one year (i.e., from 06/2004 to 01/2006). Since there are three major VOs in the system — the others being iVDgL and USCMS—, and a small number of minor VOs, we have conservatively inferred the total utilization as $3.5 \times U$, where U is the observed utilization generated by the USATLAS VO.

The second trace originates from NorduGrid [4]. We have obtained full traces of NorduGrid for a period spanning from 2003 to 2006. Since the NorduGrid's size (number of resources) has varied greatly over time, we use the value of 3100 processors as the average system size, and process the trace accordingly.

The third trace is CERN's Large Hadron Collider Computing Project Grid (LCG) [11]. The official LCG web site publishes activity statistics daily; we have built an automatic data gathering tool, and have obtained data spanning over one year (i.e., from 02/2005 to 04/2006).

For the fourth trace, we consider a hypothetical meta-infrastructure encompassing the three infrastructures OSG/Grid3, NorduGrid, and CERN LCG. We assume that this infrastructure, which we name *MetaGrid*, will first provision resources to its community of users, effectively a combination of the communities of users of its three composing grids, and only then offer spare capacity to external communities; we refer from hereon to this assumption as the *internal-use-first* assumption.

Figure 1 shows the number of available resource over time, for the production environments Grid3, NorduGrid, and LCG. It can be observed that the larger the grid, the higher the number of available resources, e.g., the average number of free resources is over 800, 2000, and 6500 in Grid3, NorduGrid, and LCG, respectively.

Since most of the jobs that currently run in the considered grid environments are single-CPU jobs [11], we can consider the free resources as available for the first single-CPU job that arrives; with parallel jobs, some jobs might have been in the queue waiting for resources to be freed, effectively preventing newly-arrived jobs from starting. Without parallel jobs, we can conservatively consider the free resources as available for provisioning world-wide collaborative applications.

World-wide data-sharing service parameters We take the values of the world-wide collaboration model's parameters (see Section 3.1) from our previous work [21, 13], and from related large-scale models and measurements [20, 17].

For our simulations, we define five workload types: *Low Load*, *Normal Load*, *High Load*, *Very High Load*, and *Extreme Load*. Table 2 shows the values of the service model parameters, for the five types of workloads. The data for the *Normal Load* were extracted directly from our previous data [13]; the values for the *High Load* and *Very High Load* workloads were inferred from the trends reported from our recent measurements [21, 13].

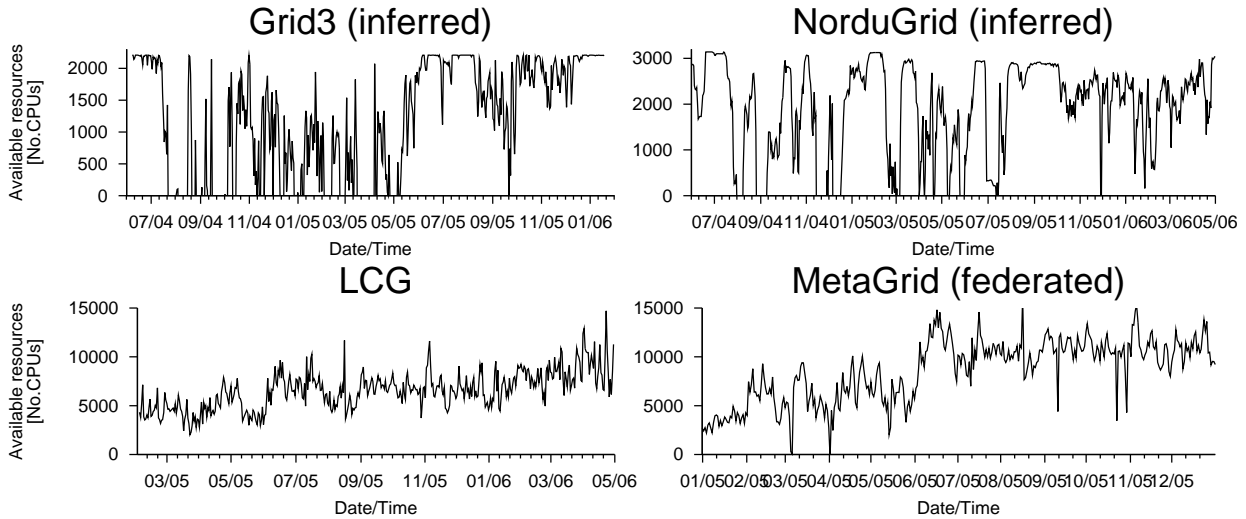


Figure 1. Number of available resource over time for the OSG/Grid3, the NorduGrid, CERN LCG, and the federated MetaGrid.

Table 2. The used values for the service model parameters.

Parameter name	Low Load	Normal Load	High Load	Very High Load	Extreme Load
λ_{CC} [CCs/day]	1,5,10	10,20,50	20,50,100	100,200,500	100,500,1000
σ_{CC} [peers]	20,50,100	20,50,100	50,100,200	100,200,500	200,500,1000
δ_{CC} [days]	7,15,30	7,15,30	15,30,90	15,30,90	15,30,90
λ_{FC} [FSs/day]	0,1	0,1,2	1,2,5	2,5,10	2,5,10
σ_{FC} [Kpeers]	5,10	5,10,20	20,50,100	100,500,1000	10,500,1000
δ_{FC} [days]	2,3	3,5	3,5,7	3,5,7	5,7,15

5 The Results

In this section we discuss our simulation results. We focus on answering two questions: "Can currently existing grids provide enough resources for the load of worldwide data-sharing services, without disturbing their existing users?", and "What is the delivered performance of our scheduling policies and their delivered performance, and what is the best policy for a given load size?".

5.1 Resource Provisioning

Figure 2 shows the evolution of the *CommCov* and *PeerCov* metrics over time for the OSG/Grid3, NorduGrid, CERN LCG, and MetaGrid environments, under normal load. We observe that even for normal load, only the largest current grid, namely CERN LCG, can ensure continuous coverage both for communities and peers, but that other grid systems may also provide more than 50% coverage. Note that the MetaGrid environment has lower service capabilities than one of his composing environments, the CERN LCG (!). This is due to the fact that the federated environment pays the price of executing the waiting load of some of its component clusters on other, free, resources.

Table 3 shows the best performance per system, given the load class. Each value represents the best achieved performance of any of the scheduling policies, for each metric considered separately. As expected, the observed values for the real environments – OSG/Grid3, NorduGrid, and CERN

LCG – show a decrease in performance with the increase in load. Note that by design the *AllCov* metric is much more susceptible to system capacity drops than the *CommCov* and the *PeerCov* metrics. Notably, for very high load on NorduGrid the *AllCov* metric value drops to 0% while the values of the *CommCov* and the *PeerCov* metrics remain over 75% each.

By comparing the values presented in Table 3 with the optimal system size, as depicted in Figure 3, we observe that the general system size of the considered systems is at most half the optimum system size, for the extreme load (the potential high load of tomorrow). Assuming that the current grid load will grow at the same pace as the resource provisioning requirements, we infer that the optimal grid size to accustom such combined load is of around 100,000 resources (double the maximum size of today's CERN LCG grid). However, we raise a question mark on the feasibility of enlarging grids to such sizes based solely on existing infrastructure. Besides the inevitable logistical problem, potentially solved by future research, it is the fact that federated resources may bring with them pending load, which will be dispatched on the newly available resources, lowering therefore the perceived global system throughput (see Figure 2 for anecdotal evidence !). We conclude that current grid systems should double their capacity to accommodate the extreme loads incurred by the future's world-wide data-

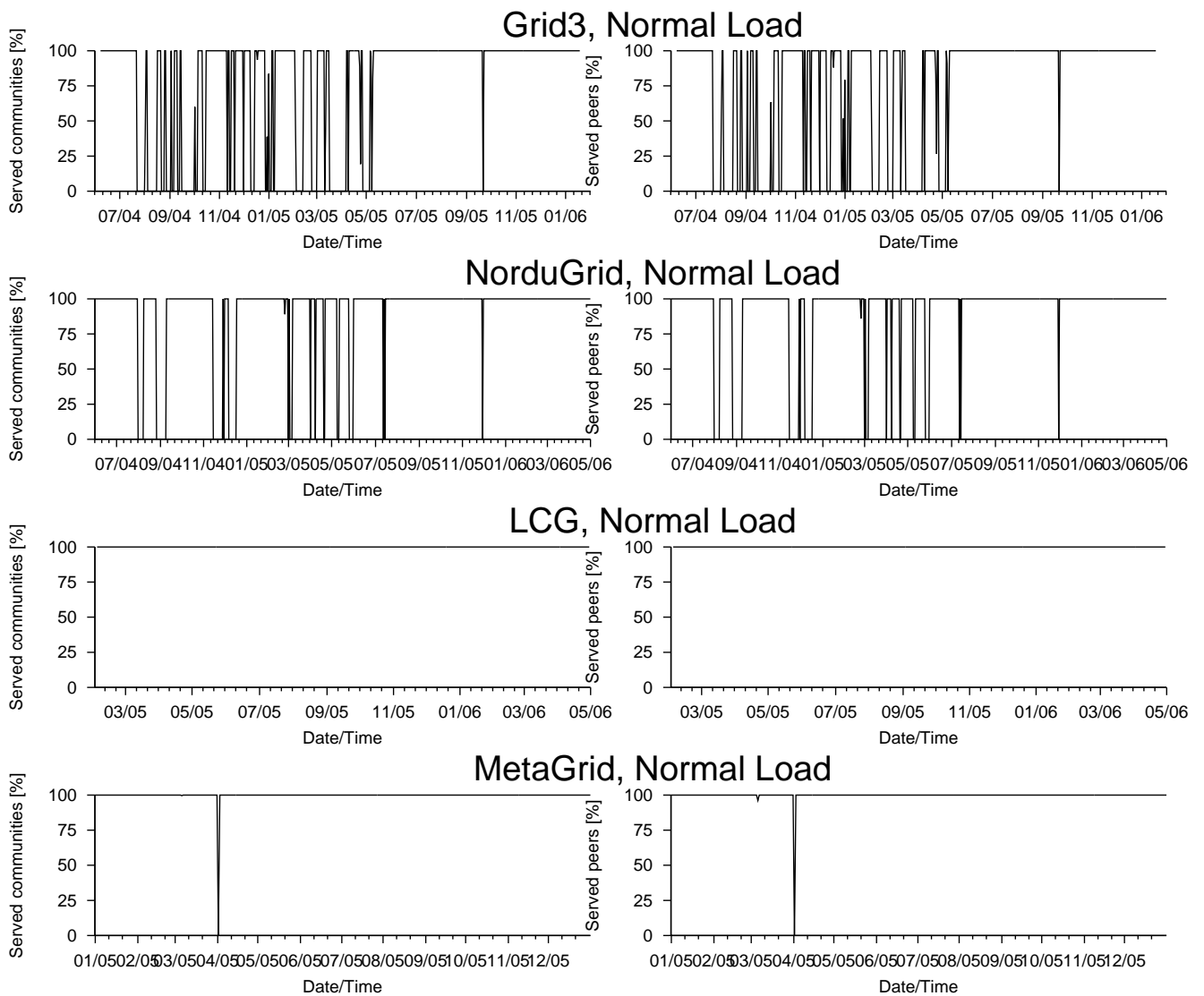


Figure 2. Communities and peers coverage over time for the OSG/Grid3, NorduGrid, CERN LCG, and MetaGrid environments, under normal load.

Table 3. The best performance per system, given the load class.

System	OSG/Grid3				NorduGrid			
	AllCov	CommCov	PeerCov	XCost	AllCov	CommCov	PeerCov	XCost
Low	78%	78%	78%	24.21	94%	94%	94%	7.65
Normal	76%	76%	77%	37.71	94%	94%	94%	11.1
High	71%	75%	76%	132.24	91%	93%	94%	38.85
Very High	0%	5%	5%	3198.48	0%	76%	77%	1996.46
Extreme	0%	1%	2%	12771.48	0%	1%	1%	10553.791

System	CERN LCG				MetaGrid			
	AllCov	CommCov	PeerCov	XCost	AllCov	CommCov	PeerCov	XCost
Low	100%	100%	100%	0.12	100%	100%	100%	0.12
Normal	100%	100%	100%	0.33	100%	100%	100%	0.33
High	100%	100%	100%	2.17	99%	100%	100%	2.17
Very High	95%	100%	99%	74.68	95%	99%	99%	74.68
Extreme	0%	7%	8%	9099.29	0%	7%	8%	8509.85

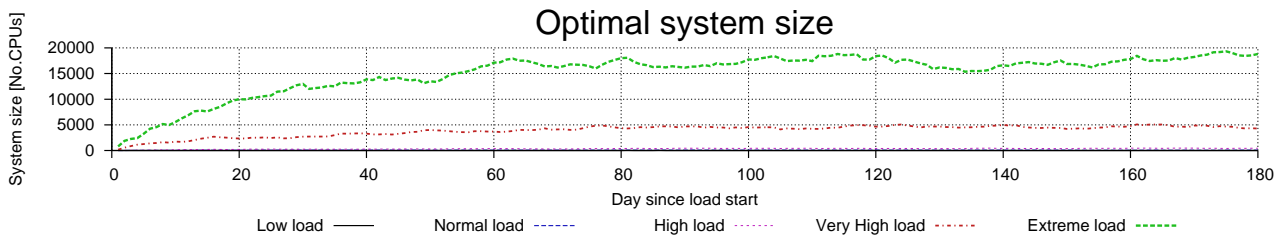


Figure 3. The optimal system size for the first six months, under all loads sizes.

sharing services, but that *The larger the better!* syntagma needs adequate usage policies.

5.2 Scheduling Policies

We now turn our attention to the validation and comparison of the resource allocation mechanisms proposed in Section 3.2. Having observed a similar scheduling policy behavior, regardless of the load size, we present in this section only the results obtained for very high load.

Figure 4 depicts and the percentages of served communities and peers for the five resource allocation mechanisms operating in the Grid3, NorduGrid, CERN LCG, and Meta-Grid, under very high load; only the first three months of service are shown for each system-policy combination; the policies performed similarly throughout the whole duration of the traces. The `Random-Pol` achieves a remarkably good balance between peer and community service. As expected, the `CommF-Pol` and the `PeerF-Pol` maximize the percentage of served communities and peers, respectively. According to their goals, the `OldF-Pol` and the `NonExclF-Pol` both act as trade-offs between maximizing the percentage of served communities, and maximizing the percentage of served peers. We conclude that the proposed resource allocation policies perform according to their design.

To compare the policies, we evaluate for each load size the performance of the policy in all the experimental environments. We consider a policy to be the best for a certain load if it performs on average at least 5% better than the next best policy, on all the experimental environments. We observe that only the `CommF-Pol` and the `PeerF-Pol` are the best in the `CommCov` and `PeerCov` categories, respectively, but only for the very high and the extreme loads. For all other cases, there are at least two very close contenders to the title of best policy. We conclude that the policy design space was completely covered by the five selected policies, and that the `CommF-Pol` and the `PeerF-Pol` fulfill their design promises.

6 Related Work

To the best of our knowledge, ours is the first approach to combine on-demand resource provisioning from dynamic environments (grids) with world-wide data-sharing services. Thus, we relate our work to research in world-wide data-sharing services, and to two separated (until now)

areas of research: generic on-demand resource provisioning, and large-scale collaborative environments.

There are several approaches for resource provisioning for world-wide data-sharing services from static environments [9, 14] (but none for the dynamic case, e.g., grids). Closest in goals to our work is the Media-On-Demand architecture [9]. The authors take a hierarchical approach to this problem, and work in the context of a privately owned, albeit large, infrastructure, with static load.

Alternatively, there are many approaches for generic on-demand resource provisioning with multiple resource owners [5, 1, 6, 23]. In Epema *et al.* [5], applications may use the free resources situated in different clusters situated in different geographical locations. Amongst other achievements, the resource utilization mechanism (called *flocking*) enables setting usage restrictions (a precursor of usage SLAs). Several economic approaches address the issue of generic resource provisioning in multi-cluster grids [6, 23]. Finally, a number of approaches deal with the underlying mechanisms of collaborative environments [20, 25], with the focus being the application control and service, as opposed to our system-oriented view.

7 Conclusions and Ongoing Work

This work has addressed the problem of resource provisioning for world-wide data-sharing services using grid infrastructures. We have analyzed the requirements of these applications, and have identified the concept of SuperPeers as a critical requirement. We have further assessed the capacity of already deployed grids to provision resources for SuperPeer-based services. Our trace-based simulations indicate that three currently existing production grid infrastructures, namely Grid3, LCG, and NorduGrid, can provide the necessary services for normal loads, but that a doubling of grid capacity is required to satisfy the requirements of extreme loads. However, our experiments have also shown that simply federating more resources to a common world-wide grid infrastructure could lead to a degradation of performance, and that new federation mechanisms are required. To cope with the case of resource shortage during high load, we have devised, assessed, and compared five domain-specific scheduling policies.

For the future, we plan to extend this research with a study of the needed mechanisms for usage SLAs [3] and for failure handling.

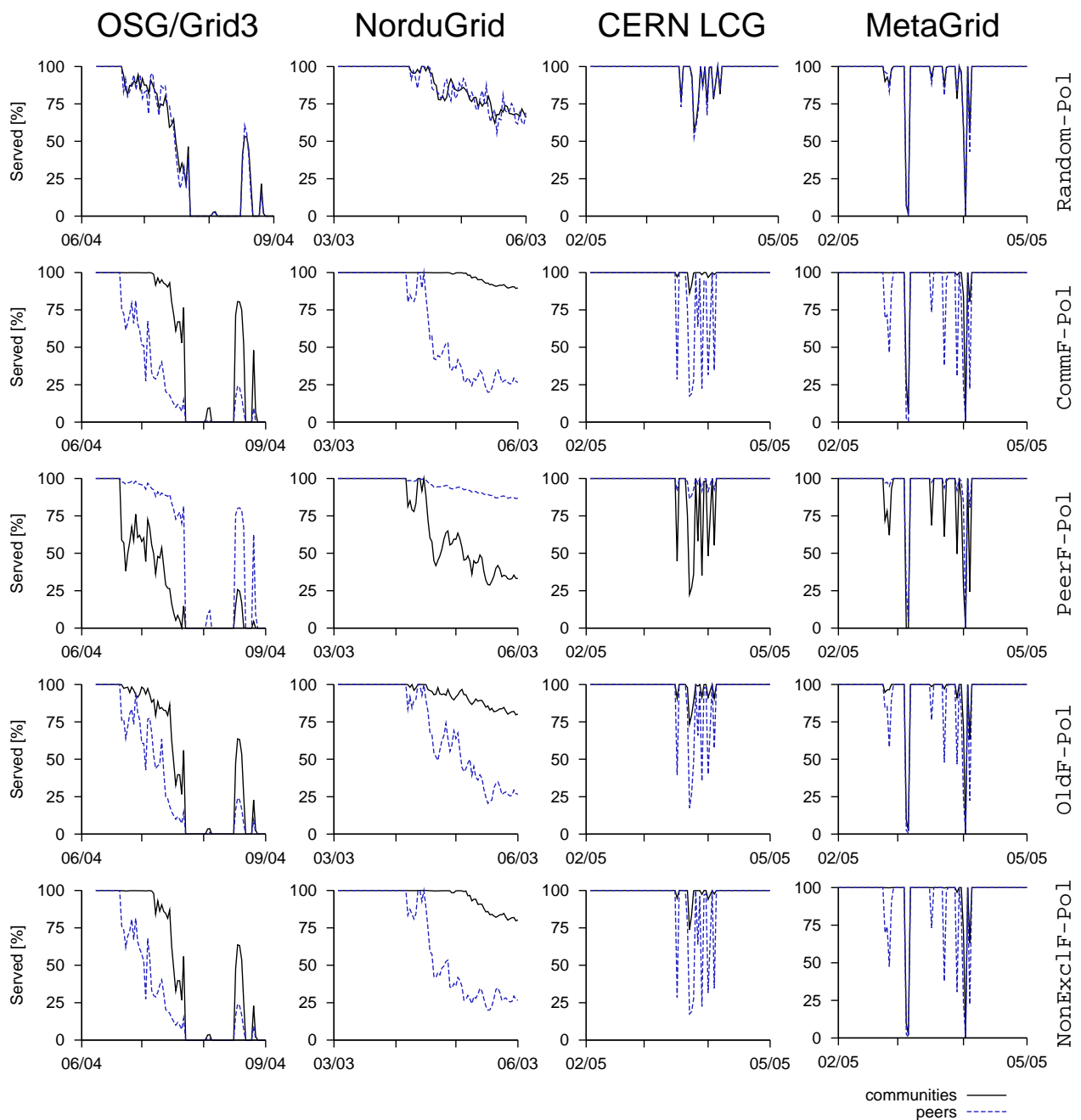


Figure 4. Comparative display of the percentages of served communities and peers over the first three months for the five resource allocation mechanisms operating in the Grid3, NorduGrid, CERN LCG, and MetaGrid, under very high load.

References

- [1] F. Berman, A. Hey, and G. Fox. *Grid Computing: Making The Global Infrastructure a Reality*. Wiley Publishing House, 2003. ISBN: 0-470-85319-0.
- [2] R. Bhagwan, S. Savage, and G. M. Voelker. Understanding availability. In *IPTPS*, pages 256–267, 2003.
- [3] A. Dan, C. Dumitrescu, K. Ranganathan, and M. Ripeanu. A Layered Framework for Connecting Client Objectives and Resource Capabilities. *International Journal of Cooperative Communication Systems*, 2006. To appear.
- [4] P. Eerola, B. Kónya et al. The nordugrid production grid infrastructure, status and plans. In H. Stockinger, editor, *GRID*, pages 158–165. IEEE Computer Society, 2003.
- [5] D. H. J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne. A worldwide flock of Condors: Load sharing among workstation clusters. 12(1):53–65, May 1996.
- [6] C. Ernemann, V. Hamscher, and R. Yahyapour. Economic scheduling in grid computing. In D. G. Feitelson, L. Rudolph, and U. Schwiegelshohn, editors, *Proceedings*

- of the 8th Job Scheduling Strategies for Parallel Processing, volume 2537 of *LNCS*, pages 128–152. Springer, 2002.
- [7] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *Intl. J. Supercomputer Applications*, 15(3), 2001.
- [8] P. Garbacki, D. Epema, and M. van Steen. Two-level semantic caching scheme for super-peer networks. In *IEEE Tenth International Workshop on Web Content Caching and Distribution*, Sophia Antipolis, France, Sep 2005.
- [9] S. Graupner, W. Kalfa, and C. Reimann. Modeling and simulation of Media-On-Demand services - evaluating a digital media Grid architecture. Technical Report HPL-2002-192, HP, Jul 2002.
- [10] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In *Proc of the Internet Measurement Conference (IMC)*, Berkeley, CA, USA, 19-21 Oct 2005.
- [11] A. Iosup, C. Dumitrescu, D. H. Epema, H. Li, and L. Wolters. How are real grids used? the analysis of four grid traces and its implications. In *The 7th IEEE/ACM International Conference on Grid Computing (Grid)*, Barcelona, ES, Sep 28-29 2006. (accepted).
- [12] A. Iosup and D. H. J. Epema. Grenchmark: A framework for analyzing, testing, and comparing grids. In *CCGRID*, pages 313–320. IEEE Computer Society, 2006.
- [13] A. Iosup, P. Garbacki, J. Pouwelse, and D. H. J. Epema. Correlating topology and path characteristics of overlay networks and the Internet. In *6th Int'l Workshop on Global and Peer-to-Peer Computing (GP2PC)*, held in conjunction with the *IEEE/ACM CCGrid'06*, 2006.
- [14] V. Kalogeraki, A. Delis, and D. Gunopulos. Handling multimedia objects in peer-to-peer networks. In *CCGRID*, pages 438–439. IEEE Computer Society, 2002.
- [15] J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in P2P file sharing systems. In *Proc. of the IEEE Infocom*, 2005.
- [16] A. Loser, M. Wolpers, W. Siberski, and W. Nejdl. Semantic overlay clusters within super-peer networks. In *International Workshop on Databases, Information Systems, and P2P Computing, colocated with 29th International Conference on Very Large Databases (VLDB2003)*, Berlin, Germany, september 2003.
- [17] L. Massoulié and M. Vojnovic. Coupon replication systems. In D. L. Eager, C. L. Williamson, S. C. Borst, and J. C. S. Lui, editors, *SIGMETRICS*, pages 2–13. ACM, 2005.
- [18] D. Milojevic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. Technical Report HPL-2002-57, HP Labs, 2002.
- [19] H. Mohamed and D. Epema. Experiences with the koala co-allocating scheduler in multiclusters. In *Proc. of the 5th IEEE/ACM Int'l Symp. on Cluster Computing and the GRID (CCGrid2005)*, Cardiff, UK, May 2005.
- [20] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai. Distributing streaming media content using cooperative networking. In *NOSSDAV*, pages 177–186. ACM, 2002.
- [21] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. In M. Castro and R. van Renesse, editors, *IPTPS*, volume 3640 of *LNCS*, pages 205–216. Springer, 2005.
- [22] J. Pouwelse, P. Garbacki, A. Iosup et al. Tribler: A Social-Based Peer-to-Peer System. In *Proc. of the 5th International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, CA, USA, 27-28 Feb 2006.
- [23] R. Ranjan, R. Buyya, and A. Harwood. A case for cooperative and incentive-based coupling of distributed clusters. In *Proceedings of the 7th IEEE Int'l. Conference on Cluster Computing (CLUSTER)*, Boston, MA, USA, Sep 27–30, 2005.
- [24] S. Sen and J. Wang. Analyzing peer-to-peer traffic across large networks. *IEEE/ACM Trans. Netw.*, 12(2):219–232, 2004.
- [25] T. Stading, P. Maniatis, and M. Baker. Peer-to-peer caching schemes to address flash crowds. In P. Druschel, M. F. Kaashoek, and A. I. T. Rowstron, editors, *IPTPS*, volume 2429 of *LNCS*, pages 203–213. Springer, 2002.
- [26] Z. Xu and Y. Hu. Sbarc: A supernode based peer-to-peer file sharing system. In *Eighth IEEE International Symposium on Computers and Communications*, Kemer-Antalya, Turkey, jun-july 2003.
- [27] B. Yang and H. Garcia-Molina. Designing a super-peer network. In *IEEE International Conference on Data Engineering*, Bangalore, India, March 2003.