Classes of Service for Scheduling in Lambda Grids using Ant Colony Optimization

Gustavo Sousa Pavani[1], Rogério L. Iope[2], Helio Waldman[1] and Líria M. Sato[2]

[1] Federal University of ABC (UFABC)
Rua Catequese, 242. Santo André–SP, Brazil. CEP: 09090-400.

[2]University of São Paulo (USP)
Av. Prof. Luciano Gualberto, Trav. 3, No. 158. São Paulo–SP, Brazil. CEP: 05508-900.

Abstract. *Scheduling in Grid computing environments is a complex task because of the unique characteristics and demands exhibited by those systems. This work proposes the use of classes of service to obtain a more efficient way for Grid resource discovery and allocation. By using ants that continuously wander in the network probing for Grid resources, we designed a system capable of a joint optimization of lightpath routing and scheduling in Lambda Grid systems, without the need of a resource broker, considering the heterogeneity of resources with multiple users competing for them. The results show that the proposed ant-based system can effectively be used as a Grid scheduler over optical networks.*

Resumo. *O escalonamento de recursos em ambientes de Grids computacionais é uma tarefa complexa, devido às características e demandas dinâmicas de tais sistemas. Este trabalho propõe o uso de classes de serviço com o objetivo de obter um meio mais eficiente para a descoberta e a alocação de recursos em Grids. Através do uso de agentes móveis que continuamente vagueiam pela rede em busca de recursos computacionais, desenvolvemos um sistema capaz de executar uma otimização conjunta de roteamento de caminhos ópticos e de escalonamento em sistemas de Grid construídos sobre redes ópticas WDM, conhecidos como 'lambda grids', sem a necessidade de uso de um agente alocador de recursos, e que leva em consideração a heterogeneidade dos recursos, com múltiplos usuários competindo por eles. Os resultados obtidos mostram que o sistema multi-agentes proposto pode efetivamente ser usado como um escalonador de recursos de Grids sobre redes ópticas WDM.*

1. Introduction

The concept of computational Grid environments is not new, but mainly built on ideas first exposed in 1960s. The term "Grid" was introduced only in the middle of the 1990s by the pioneers Foster and Kesselman, to describe their proposal for a distributed computing infrastructure for advanced science and engineering [Foster and Kesselman 1999], inspired in other kinds of distributed infrastructures assembled to deliver a certain class of services to the final user, like the electric power grid. A Grid computing infrastructure can be defined as "a distributed system that integrates and coordinates resources not subject to centralized control, using standard and open protocols and interfaces to deliver nontrivial qualities of service" [Foster 2002].

A critical aspect of Grid computing environments is the network support. The rapid evolution and adoption of Grid technologies are occurring mainly due to the significant increase of the bandwidth and speed of networks. However, current Grid systems are still relying on the commodity Internet to resource integration, but this is severely restricting the deployment of Grid in wide-area scales. The commodity Internet, based on a best-effort delivery model, is too slow to allow efficient –nearly real-time– use of remote resources and to handle the huge masses of data being generated in emerging e-Science applications [Boutaba et al. 2003]. Also, it is simply not possible to schedule an underlying best-effort network.

A promising solution to these limitations is optical networking. Recent advances on Wavelength Division Multiplexing (WDM) technologies can effectively use the available bandwidth in a single strand of fiber, so building a widespread Grid infrastructure over an optical networking core is a very attractive proposition [Simeonidou and Nejabati 2004]. In typical WDM networks, optical connections between source and destination nodes are dynamically set up by lightpaths that may traverse several intermediate nodes under the control of a series of optical switches.

Emerging lightpath-based technologies on WDM networks are appealing to the Grid community because they allow the inclusion of the network infrastructure as a schedulable resource under the control of a Grid scheduler, just like it is currently done with computing and storage resources. Recent developments on integrated control planes such as Generalized Multi-Protocol Label Switching (GMPLS) [Mannie 2004] are enabling a more effective and standardized way of managing the network resources on optical cores.

In this paper, our main goal is to describe a system based on Ant Colony Optimization (ACO) metaheuristics [Dorigo and Stützle 2004] that works as a Grid scheduler for systems built around a GMPLS-controlled WDM optical network, where lightweight ant-agents act on behalf of the Grid users to make scheduling decisions based on the users' need of processing power, taking into account the available path on the optical infrastructure from each user to an available resource, thus balancing the Grid workload.

Scheduling is a core function of any resource management system. A scheduler acts as the interface between the resource consumers, or users, and the underlying resources. We are considering a distributed environment where a set of computational resources are interconnected by optical links, and a pair of users who want to submit jobs for execution on the set of resources. The scheduling algorithm should be able to matchmaking the set of users' jobs and the set of resources.

This paper proposes an extension to a previous work [Pavani and Waldman 2006b], in which it was considered only one user searching for a single type of resource on the entire network. For the present work, we are considering more than one user competing for heterogeneous resources with different application needs.

Moreover, this work introduces the concept of classes of service when performing resource discovery and resource allocation. Indeed, in traditional Grid systems, when selecting a computational resource to be assigned to a job, an exhaustive search is generally done, looking for resources that matches the minimum requirements of each particular

job. This process, however, can be enhanced if the Grid users and resource providers make an agreement between themselves to define classes of service to fulfill the applications requirements, thus establishing a Service Level Agreement (SLA) [Sandholm 2005] for different classes of applications. Such classes can be suitable for e.g. applications that need huge amounts of RAM memory for executing.

By using lightweight agents (ants) to carry the information about availability for each class of service in each administrative domain, we propose an architecture that can be used for a joint optimization of routing and resource discovery and allocation in Grids systems. As the provisioning of lightpaths is handled in an on demand basis, a Routing and Wavelength Assignment (RWA) [Zang et al. 2000] algorithm must be integrated to the scheduler to provide it with resource discovery capabilities. These capabilities emerge naturally from the way this scheduler is proposed.

The remaining of the paper is organized as follows. In Section 2, we briefly discuss the problem of scheduling in Grid systems. The proposed algorithm and node architecture are described in Section 3. In Section 4, we detail the simulation studies carried out to properly characterize the algorithm behavior. Results obtained are presented and discussed in Section 5. Finally, in Section 6, conclusions are drawn.

2. Grid Scheduling

Grid scheduling is the process of making scheduling decisions involving distributed resources that span multiple administrative domains [Schopf 2004]. A resource is defined as anything that can be scheduled to participate in a computational task: CPU-time, storage space, network bandwidth, data acquisition instruments, and so on. A job is the software entity that runs on the Grid infrastructure and which keeps requesting resources during its entire lifetime, competing with other jobs for shared resources.

Scheduling in Grid environments is a complex task because of the unique characteristics and demands exhibited by those systems, such as resource and application heterogeneity, and resource division in distinct and well defined administrative domains. Inappropriate scheduling of jobs is a serious drawback in such environments, as it prevents exploiting the full potential of the Grid infrastructure and may lead to excessive communication overhead or under-utilization of usually expensive resources. The functionality and performance of a Grid scheduler ultimately determine the users' experience of the environment, which can be a determining factor for a wide user acceptance or rejection of the system as a whole.

The problem of scheduling heterogeneous jobs onto heterogeneous resources, also known as the task allocation problem, is known to be an NP-hard problem. In such a class of problems, exact solution methods are intractable for most instances, and solutions must be searched using heuristic approaches.

For our work, we propose the use of the ACO metaheuristics for Grid scheduling, which is detailed in the following Section.

3. Ant Colony Optimization (ACO) metaheuristics

ACO is a recently proposed metaheuristics approach for solving hard combinatorial problems or problems that need distributed control. It was inspired in the process of foraging

for food of real ant colonies, which efficiently organize their collective behavior using chemical pheromone trails deposited on the ground by each individual as a means of indirect communication between the colony members. ACO algorithms have since been widely applied to find solutions for many NP-hard combinatorial optimization problems, not necessarily related to computing systems.

The (artificial) ants act as lightweight mobile agents that perform an adaptive greedy search on the solution space of the problem, by moving throughout the edges on the graph that represents the problem. The greedy search takes into account information provided by artificial "pheromone trails" which change dynamically at run-time to reflect the agents' acquired search experience.

ACO metaheuristics is explicitly modeled in terms of computational, generally mobile, agents, and has an important feature that is explored in our work: it is especially well-suited for routing in telecommunications networks [Bonabeau et al. 2000]. ACO agents can be defined as computational entities that act autonomously and can react to external stimuli, make decisions according to the environment, and cooperate with other agents. Mobile agents are agents that are capable to migrate autonomously in a network.

By means of both iterative and parallel processes, each agent, or ant, builds a solution using two types of local information: specific problem information and information added by other ants during previous iterations of the algorithm that are embodied in a single variable per network element and destination, which is called the pheromone level. Indeed, while building the solution, each ant gathers information about characteristics of the problem to be solved and about its own performance and uses this information to modify the representation of the problem, i.e., the pheromone at each node, as seen locally by other ants. The representation of the problem is thus modified in such a way that the information contained in previous solutions can be distributively explored to obtain better solutions.

In fact, our previous work [Pavani and Waldman 2006b] showed how to integrate an ACO-based RWA algorithm into a Grid scheduler. The rationale behind this new application is that ant-based algorithms seem to be very appropriate to be used as a basis for scheduling decisions in Grid environments, as they naturally combine the solutions to the discovery and routing problems: when a computational resource is discovered, the pheromone levels produced by the search activity are already an appropriate metric for good routing. In ACO based algorithms, discovery and routing emerges naturally as the system evolves towards the best solution.

The AntNet [Di Caro and Dorigo 1998] framework is an ACO-based algorithm designed for solving the problem of routing packets in telecommunication networks, and this work was largely inspired on it.

The original AntNet framework uses the delay introduced by each hop as the metric for routing. However, this is not applicable in circuit-switching networks, where the main metric is generally the number of hops. The minimization of the number of hops of a connection following some criteria is a very good heuristics to reduce the overall blocking probability in a network. Then, the ant used in the proposed algorithm will only carry on its memory the node identification of each node where it passes by.

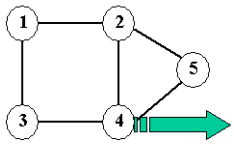
At each intermediate node i, the following data structures have to be maintained

in AntNet, which are slightly adapted in our algorithm in order to use the number of hops in lieu of the delay:

1. Pheromone-routing table $\mathcal{T}^i$: It is a matrix containing a row for each destination d of the network and a column for each neighbor node n, for storing the pheromone values. The sum of each row must be equal to 1, i.e., $\sum_{n \in \mathcal{N}_i} \tau^i_{dn} = 1$, where $\mathcal{N}_i$ represents the set of neighbors of node i. τ^i_{dn} defines the value of pheromone in the link $l_{i \to n}$ for the specified destination d and estimates the probability of reaching d given that n is selected as the next hop. In this way, we have only one type of pheromone, but an ant that travels to a specific destination only senses/modifies the pheromone levels associated with its destination (row) in the routing table. Figure 1(a) depicts an example of pheromone routing table of an example network.
2. Statistical parametric model $\mathcal{M}^i$: It is a matrix containing the triplet $<\mu_d, \sigma_d, E_d>$ for each destination d of the network, where μ_d represents the average length of the paths followed by the ant from the current node to destination d, σ_d the standard deviation for these path lengths and E_d the best value of path length found for this destination within the non-sliding window of w observations [Di Caro and Dorigo 1998]. Non-sliding means that when the number of observations reaches $w + 1$, all the accumulated values are reset, i.e., $E_d = \mu_d =$ (path length of the $(w + 1)$-st observation), $\sigma_d = 0$ and the observation counter receives 1.
   The values of μ^i_d and σ^i_d are updated using the following exponential model [Jacobson and Karels 1990]:
   $$\mu^i_d \leftarrow \mu^i_d + \eta(o_{i \to d} - \mu^i_d) \quad (1)$$
   $$\sigma^i_d \leftarrow \sigma^i_d + \eta(|o_{i \to d} - \mu^i_d| - \sigma^i_d)\ , \quad (2)$$
   where $o_{i \to d}$ is the new observation of the distance between nodes i and d, and η is the factor of the exponential model, which weighs the number of the most recent observations that will influence the mean. The number of effective observations is approximately equal to $5/\eta$ [Di Caro and Dorigo 1998].
   The number of values in the window is calculated as $w = 5(c/\eta)$, where $c \in (0, 1]$ is a reduction factor. Therefore, the window is updated in a smaller interval than the one used for the mean and standard deviation estimates, in such a way that the value of E^i_d and these estimates refer to the same set of observations [Di Caro and Dorigo 1998].
   Figure 1(b) depicts an example of the statistical parametric model of an example network.
3. Availability vector $\mathcal{A}^i$: It is a vector containing availability metrics for each destination d of the network.

$\mathcal{M}$ and $\mathcal{T}$ can be seen as local memories for the nodes, capturing different aspects of the network dynamics. The $\mathcal{M}$ model maintains distance estimates to all nodes, while the pheromone routing table gives the relative goodness of the next hop to reach a given destination.

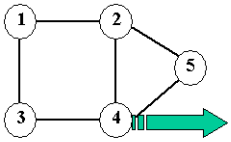
The AntNet algorithm is composed by two phases [Dorigo and Stützle 2004]: solution construction and updating of the data structures, which are detailed in the next sub-sections.



Neighbor nodes

Destination nodes	3	2	5
1	0.1	0.5	0.4
2	0.2	0.3	0.5
3	0.9	0.1	0.0
5	0.5	0.25	0.25

(a) Example of pheromone routing table.



Destination nodes	μ_d	σ_d	E_d
1	2.2	0.5	2.0
2	1.3	0.3	1.0
3	1.2	0.4	1.0
5	1.4	0.5	1.0

(b) Example of statistical parametric model.

Figure 1. Example of pheromone routing table and statistical parametric model of node 4.

3.1. Solution construction

The algorithm starts with the initialization of the pheromone routing tables of each optical node. To speed up the convergence of the algorithms, we used the intelligent initialization of routing tables as described in [Barán and Sosa 2001].

Before starting the arrival of requests, only ants are launched to explore the network and populate the routing tables with topology information. In practice, this allows for the configuration of the routing tables with the shortest path in the absence of congestion. After a small amount of time I_{warmup}, the lightpath requests start to arrive randomly at the network nodes.

At regular intervals $(1/R_{ants})$, a forward ant $F_{s\rightarrow d}$ is launched from a random source node s to another random destination node d. On its trip from s to d, the forward ant selects the next hop $(i+1)$ using a random scheme that accounts for the path selection probabilities, given by the pheromone levels τ_{dn} in each neighbor link, and for a heuristics value h_n, calculated from the availability of the neighbor links.

The availability of each neighbor link is calculated as follows:

$$h_n = \frac{l_n^a}{W}, \tag{3}$$

where l_n^a is the number of available wavelengths on neighbor n and W is the total number of wavelengths deployed on the link.

During its trip, the forward ant gathers the label of each node where it passes by, putting it in its memory $V_{s\rightarrow i}$, which also servers as tabu list [Glover and Laguna 1997].

If among the neighbor nodes of the node that is processing the ant, there are any not visited yet, the choice of the next hop is done using a random scheme and the probability for each candidate node n to be the next hop $(i+1)$ are given by the following expression [Di Caro and Dorigo 1998]:

$$p_n^d = \begin{cases} \left(\frac{1}{1+\alpha}\right)\frac{\tau_{dn}}{\sum_{k\in T}\tau_{dk}} + \left(\frac{\alpha}{1+\alpha}\right)\frac{h_n}{\sum_{k\in T}h_k}, \forall k \in T \\ 0 \text{ , otherwise,} \end{cases} \tag{4}$$

where α gives the emphasis between pheromone level (long-term memory) and instantaneous availability state (short-term memory)), and $T = \mathcal{N}_i \setminus V_{s\rightarrow i}$.

However, if all neighbor nodes have already been visited (T is empty), this indicates that the ant entered a loop. We ignore the heuristic correction given by the link

congestion, choosing the next node in a random way, where the probability of each candidate node n to be the next hop $(i+1)$ is given by the following expression:

$$p_n^d = \begin{cases} \dfrac{\tau_{dn}}{\sum_{k \in T'} \tau_{dk}} & , \forall k \in T', \text{ if } |\mathcal{N}_i| > 1 \\ 1 & , \text{ if } \mathcal{N}_i = \{v_{i-1}\} \\ 0 & , \text{ otherwise,} \end{cases} \tag{5}$$

where v_{i-1} represents the last visited node and $T' = \mathcal{N}_i - \{v_{i-1}\}$.

In this case, after the selection of the next hop, all labels that belong to nodes of the cycle are removed from the ant's memory.

If the ant does not reach its destination node in a number of pre-established hops it is dropped. This avoids lost ants circulating forever in the network.

3.2. Updating of data structures

When the forward ant arrives at d, it becomes a backward ant $B_{d \to s}$ and returns to s using the same path followed by the forward ant, but in the opposite direction. At each intermediate node i, it updates the local parametric model $\mathcal{M}^i$ and, after it, the pheromone routing table, for all entries relative to d.

Moreover, this update is also made for all nodes $d' \in V_{i \to d}$, $d' \neq d$ in the sub-paths $(i \to d')$ traversed by the forward ant $F_{s \to d}$ after visiting i. If this sub-path is statistically good, then the entries of $\mathcal{M}^i$ and $\mathcal{T}^i$ relative to d' are also updated. This allows for the updating of good paths found by ants that were not intended to those destinations.

A sub-path is considered statistically good if $dist(V_{i \to d'}) < I_{sup}^{d'}$ [Di Caro and Dorigo 1998], where $dist()$ is a function that gives the distance, in terms of number of hops, of the path followed by the ant and I_{sup} is a superior estimate calculated from Tchebycheff's inequalities, which permit the definition of a confidence interval of a random variable that follows any kind of distribution. The inferior estimate is equal to $E_{d'}$. The superior estimate can be expressed by the following formula:

$$I_{sup}^{d'} = \mu_{d'} + \frac{1}{\sqrt{(1-\gamma)}} \frac{\sigma_{d'}}{\sqrt{w}} , \tag{6}$$

where γ is the confidence level coefficient.

Thus, the local parametric model is updated using Equations 1 and 2, where $o_{i \to d} = dist(V_{i \to d})$. If $o_{i \to d} < E_d^i$, then $E_d^i \leftarrow o_{i \to d}$. The same process is repeated for all d', whose sub-paths were considered statistically good.

After the updating of the parametric model, an adaptive reinforcement r_d is calculated for the updating of the routing table [Di Caro and Dorigo 1998]:

$$r_d = c_1 \left(\frac{E_d}{dist(V_{i \to d})} \right) + c_2 \left(\frac{I_{sup}^d - E_d}{(I_{sup}^d - E_d) + (dist(V_{i \to d}) - E_d)} \right) \tag{7}$$

The first term of Eq. 7 simply evaluates the ratio between the best route within the non-sliding observation window and the distance traversed by the ant. The second term

evaluates how far is this distance from the confidence interval. It is important to note that the second term must be considered equal to zero when $dist(V_{i \to d}) = I^d_{sup} = E_d$. The c_1 and c_2 coefficients weigh the importance of each term.

The obtained r is limited to 0.9 to avoid stagnation and its value is "squashed" using the following expression:

$$r_d = \frac{s(r)}{s(1)}, \text{ where } s(x) = \left(1 + \exp\left(\frac{a}{x|\mathcal{N}_i|}\right)\right)^{-1}, \tag{8}$$

where a is an amplifier coefficient.

Now, if the neighbor node m is on the path, then it receives a positive reinforcement:

$$\tau_{dm} \leftarrow \tau_{dm} + r_d(1 - \tau_{dm}) \tag{9}$$

On the other hand, the other nodes receive a negative reinforcement:

$$\tau_{dn} \leftarrow \tau_{dn} - r_d\tau_{dn}, \forall n \in \mathcal{N}_i, n \neq m. \tag{10}$$

As already done for the parametric model, the updating process is also repeated for all d' considered statistically good.

Although the parametric model does not have link availability information, the effect of a higher number of forward ants that choose the path with a link less loaded results in a higher reinforcement of the paths with lesser blocking probability.

Finally, when the backward ant returns to source node s, the position d of the availability vector $\mathcal{A}^s$ is updated with the value of availability value gathered by the ant at destination d.

3.3. Node Architecture

The basic difference in the ACO algorithm as it is used in this work relative to [Pavani and Waldman 2006b] is the availability vector, used to store the availability information for each resource node, which was enhanced in the present work. In fact, the availability vector in the previous work was too limited in the sense that it was capable of being used by only one user with the limiting assumption that all machines in the Grid are homogeneous.

By using classes of service, the availability vector now stores a metric related to each class. Therefore, for each destination of the availability vector we have an entry for each class of service. These entries are updated by the backward ants, as in the previous work. In fact, classes of service facilitates the process of discovering available resources in the Grid.

Classes of service can be defined by an agreement between all different administrative domains of the Grid. Any kind of protocol can be used to define these classes, by negotiating the service level expected for each class of service. In this way, it is possible that the users state their preferences and/or application requirements in a very simple and scalable manner without the need for an exhaustive search of all available resources in the Grid.

It is very difficult to establish a generic set of classes of service, because it is usually an application-specific problem. There can be any amount of classes of service as needed.

For each node, we have defined an SLA Manager, similar to the one proposed in [Sandholm 2005], which is responsible for establishing the classes of service in the Grid, determining the common requirements and/or preferences for the negotiated service level.

We have also defined three different data structures for implementing the ACO routing. The first two, pheromone routing table and statistical parametric model, were proposed in [Di Caro and Dorigo 1998] for the AntNet algorithm, and adapted for the RWA problem in [Pavani and Waldman 2006a]. The last one is the availability vector [Pavani and Waldman 2006b], which stores an availability metric for each destination of the network. This structure allows for the load-balancing of the Grid workload, since the node with the highest availability (as indicated by the availability vector) is assigned to process a job. Note that ants travel along the network through the separated control channel, i.e., they do not rely on the lightpaths used for transporting job data.

Also, we have GMPLS data structures (RSVP-TE and LMP databases) in each node, but no Link-State Database is being used, since the routing is done by using the ACO data structures.

The Path Computation Element (PCE) [Farrel et al. 2006] uses the information stored in the pheromone routing table for calculating the path in a hop-by-hop basis. For instance, the neighbor entry with the highest level of pheromone is chosen as the next hop of the RSVP Path message [Pavani and Waldman 2006a]. The wavelength can be assigned using a first-fit approach, since this scheme performs well in terms of blocking probability and fairness, and have a low computational cost [Zang et al. 2000]. Resilience to network failures can be straightforwardly implemented using the technique proposed in [Pavani and Waldman 2008].

We are considering that all the Grid participants are directly attached to both the optical data network core and the control plane signaling infrastructure - or at least, there exists a gatekeeper responsible for handling the computing resources behind it at each optical network node. This integration allows for the full control of the network while being able to directly interact with the Grid infrastructure. In this case, there is no need for Grid middleware or resource broker to manage the resources of the optical networking domain.

4. Simulation

For testing the proposed algorithm, we used the NSFNet backbone network, shown in Figure 2. It is a 14-node network with 21 bidirectional links and it is well-balanced, with an average shortest-path length between all pairs of nodes equal to 2.2.

We have considered a homogeneous, Poissonian traffic with uniform spatial profile. Nodes 1 and 2 compete for resources in the Grid, which are all located on the other nodes. The period for the execution of each job follows an exponential execution time with an average value of 1500 seconds (on a machine with a SpecINT2000 [SPECInt2000 Benchmark] score equal to 1000).

For evaluating the proposed algorithm, we have defined two different classes of

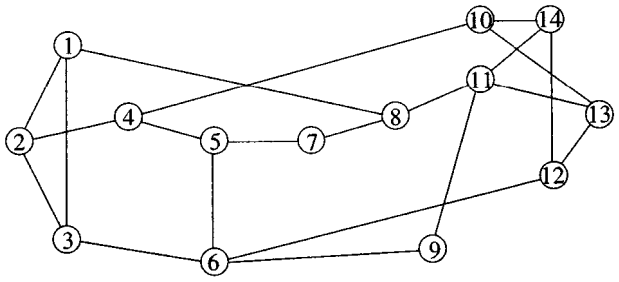


Figure 2. NSFNet backbone network.

service related to the processing power (defined by the SpecINT2000 scores) as available in the Grid, as shown in Table 1:

Table 1. Classes of Service description.

Class	SPECInt Score
Standard	500
High-perf	2000

We have considered also three different scenarios:

- **Scenario 1:** Node 1 requests resources on the standard class and node 2 on the high-performance class.
- **Scenario 2:** Both nodes 1 and 2 request resources on the high-performance class.
- **Scenario 3:** Both nodes 1 and 2 request resources on the standard class.

The focus of this work is on CPU-intensive tasks. We are considering that the execution time is much higher than the time it takes to send the input and executable files from users to resources and the time to send the output files back to the users when execution is finished.

Moreover, we are considering that there is enough storage space on the resources for handling any number of processing requests. Since we are only interested in the steady-state evaluation of the system, we assume that the system does not allow advance reservation of resources. This consideration is also linked to the fact that the GMPLS standard does not support advance reservation. In addition, adding reservation increases the wait time of queued applications in almost all cases [Smith et al. 2000].

Without loss of generality, we consider that the blocking is only due to the lack of a destination node to handle the job, i.e., the optical network has enough wavelengths to handle processing requests and their returning results. Indeed, the trade-offs between number of wavelengths and processing resources were extensively studied in [Pavani and Waldman 2006b], thus the same behavior would be reproduced in this work if the same simulation scenarios were considered. Note that this simulation assumes no wavelength conversion capability in the networks.

The parameters used in the simulations are depicted in Table 2:

We have considered that each node has the same number of processors of the same class. For the standard class, we have 80 processors per node and for the high-performance class, 20 processors per node. Therefore, both classes have the same total processing power despite the different number of processors.

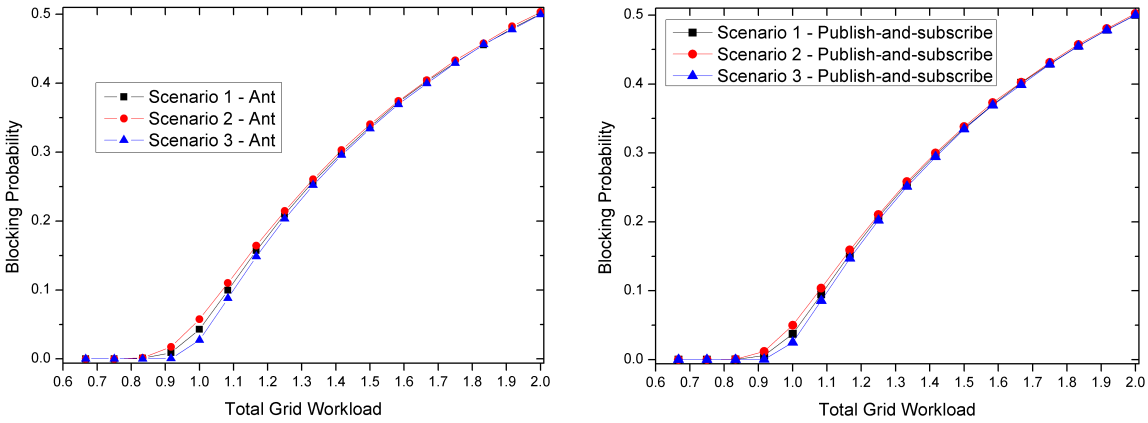
Table 2. Parameters used in the simulations.

Parameter	Symbol	Value
Rate for launching forward ants	R_{ants}	48 ants/s
Interval to start the lightpath requests	$I_{requests}$	1000 s
Correction for routing of forward ants	α	0.6
Weight of the window's exponential model	η	0.005
Reduction for parametric model's window	c	0.3
First weight of the adaptive reinforcement	c_1	0.6
Second weight of the adaptive reinforcement	c_2	0.4
Confidence level for reinforcement	γ	0.65
Amplifier of the squash function	a	5

Finally, for comparison purposes, we implemented a distributed publish-and-subscribe Grid system that routes the lightpaths using a shortest-path scheme and allocates the jobs on the least-loaded node.

5. Numerical Results

First of all, we evaluated both our algorithm and the publish-and-subscribe system under different loads. For sake of clarity, the results for the different approaches are depicted in Figures 3(a) and 3(b), respectively. There is almost no difference in terms of blocking probability performance between the two approaches. Thus, the rest of the paper is focused on the proposed ant-based algorithm.



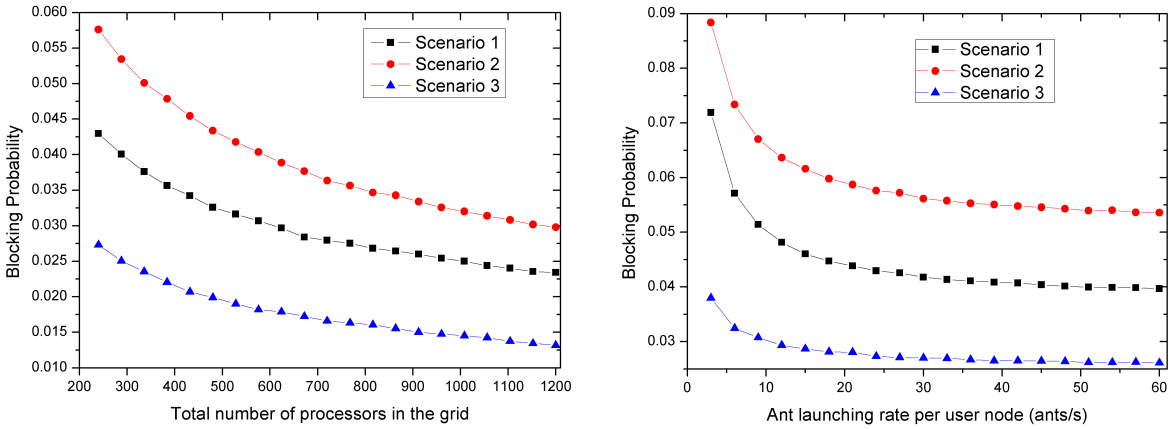
(a) Ant-based algorithm (b) Publish-and-subscribe

Figure 3. Blocking probability under different Grid workloads

We can observe that the blocking probability in Scenario 2 is higher than in Scenario 1, which in turn has a higher blocking probability than in Scenario 3. This can be explained by the fact that Scenario 3, followed by Scenario 1, has a higher number of processors per node for the same workload of jobs.

At a 100% Grid workload, just a small percentage of jobs is blocked. At the extreme case shown, when we force a workload with a value that is the double that the Grid can process, half of the jobs are blocked, i.e., it can handle all the the supported workload and the excess load is blocked as expected.

We have also verified the influence on the number of processors in the Grid upon the blocking probability. The results are depicted in Figure 4(a). We have used a 100% Grid workload.



(a) Number of processors in the Grid (b) Ant launching rate

Figure 4. Blocking probability for the ant-based algorithm.

Looking at Figure 4(a), we can see that the greater the number of processors available in the Grid, the easier it is to find an available processor in a fully loaded Grid.

In other words, the algorithm becomes more efficient when the number of processors per node increases. Note that, in a conventional publish-and-subscribe communication system like the Grid system we are considering, the greater the number of processors in a node, the more complex is the process of best matching the request and the resource, since an exhaustive search is needed.

It is also a good indication that the proposed ACO algorithm scales well with the Grid size.

Then, we evaluated the influence of the global ant launching rate over the blocking probability for a Grid workload equal to 100%. The results are shown in Figure 4(b).

Increasing the ant rate improves the performance of the system, but this improvement tends to stability after a certain value. This parameter is very important to obtain a good performance of the ACO algorithm. However, the value of this parameter is intrinsically dependent of the system and it is usually tuned up using a trial and error approach.

The extra overhead due to the ants on the control channels are very dependent on the implementation of the ant packets. Let's suppose that the ant is implemented like a datagram in a IPv6-capable network. In the simplest case, we have a 40-byte header and each hop contributes 16 bytes to the ant's payload. Also, the backward ant has two extra 8-byte fields for storing the availability information, i.e., one 8-byte field for each class implemented.

Figure 5 depicts the overhead caused by the ant packets averaged over all links for different loads, where the bars indicate the standard error of the mean.

As can be depicted from Figure 5, the overhead introduced by the ants into the control channels is very small, when compared to traditional routing architectures, and

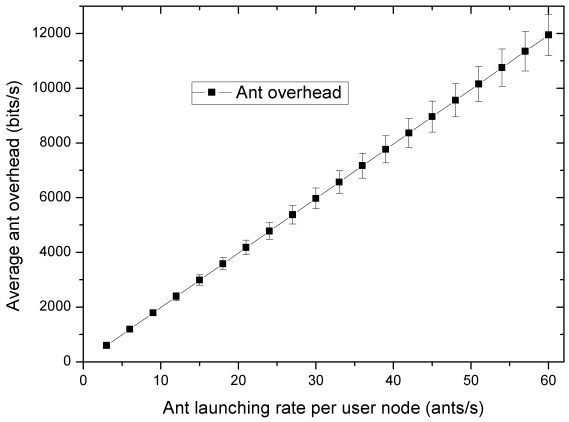


Figure 5. Overhead of the ant packets on the control channels.

scales linearly with the ant launching rate. Since the routing and the resource discovery are combined, the resource discovery also has a very little impact on the control channels.

6. Conclusions

In this work, we proposed the use of classes of service to simplify and to speedup the process of discovery, selection and ranking of resources in a Lambda Grid environment. By using an ACO-based algorithm, we have demonstrated that this approach can effectively be used to tackle the problems related to the heterogeneity of resources and the distributed nature of Grid systems.

The proposed algorithm is capable of completely integrating the lightpath routing and Grid resource management in a single framework, introducing a very small communication overhead. Also, it balances the Grid workload by assigning the jobs to the least-loaded node [Pavani and Waldman 2006b] and tends to be more scalable than the traditional publish-and-subscribe system.

6.1. Future Work

It would be interesting to introduce some kind of advance reservation in this architecture to evaluate its impact. However, since the proposal for a Grid GMPLS standard is in its infancy, some assumptions should be made about it for implementing the reservation of resources.

Acknowledgments

The authors wish to thank Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Brazil for supporting this research.

References

Barán, B. and Sosa, R. (2001). AntNet – routing algorithm for data networks based on mobile agents. *Revista Iberoamericana de Inteligencia Artificial*, (12):75–84.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (2000). Inspiration for optimization from social insect behaviour. *Nature*, 406:39–42.

Boutaba, R., Golab, W., Iraqi, Y., Li, T., and St. Arnaud, B. (2003). Grid-controlled lightpaths for high performance grid applications. *Journal of Grid Computing*, 1(4):387–394.

Di Caro, G. and Dorigo, M. (1998). AntNet: distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. MIT Press.

Farrel, A., Vasseur, J.-P., and Ash, J. (2006). A Path Computation Element (PCE)-Based Architecture. RFC 4655 (Informational).

Foster, I. (2002). What is the grid? a three point checklist. *Grid Today*.

Foster, I. and Kesselman, C., editors (1999). *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers.

Jacobson, V. and Karels, M. (1990). Congestion avoidance and control. *ACM Computer Communication Review*, 18(4):314–329.

Mannie, E. (2004). Generalized Multi-Protocol Label Switching (GMPLS) Architecture. RFC 3945 (Proposed Standard).

Pavani, G. S. and Waldman, H. (2006a). Evaluation of an ant-based architecture for all-optical networks. In *10th Conference on Optical Network Design and Modelling (ONDM'06)*.

Pavani, G. S. and Waldman, H. (2006b). Grid resource management by means of ant colony optimization. In *Third International Conference on Broadband Communications, Network and Systems (BroadNets 2006)*, San Jose, CA.

Pavani, G. S. and Waldman, H. (2008). Restoration in wavelength-routed optical networks by means of ant colony optimization (to appear). *Photonic Network Communications*.

Sandholm, T. (2005). Service level agreement requirements of an accounting-driven computational grid. Technical Report TRITA-NA-0533, Royal Institute of Technology, Stockholm, Sweden. http://www.pdc.kth.se/ sandholm/trita/TRITA-SLA.pdf.

Schopf, J. M. (2004). Ten actions when grid scheduling: the user as a grid scheduler. In *Grid resource management: state of the art and future trends*, pages 15–23. Kluwer Academic Publishers.

Simeonidou, D. and Nejabati, R. (2004). Photonic infrasctructure for grid enabling networks. *6th International Conference on Transparent Optical Networks: ICTON 2004*.

Smith, W., Foster, I., and Taylor, V. (2000). Scheduling with advanced reservations. In *Proceedings of the 14th International Symposium on Parallel and Distributed Processing*, pages 127–132.

SPECInt2000 Benchmark. http://www.specs.org/.

Zang, H., Jue, J., and Mukherjee, B. (2000). A review of routing and wavelength assignment aproaches for wavelength-routed optical WDM networks. *Optical Networks Magazine*, 1(1):47–60.