Building a Peer-to-Peer Information System in Grids via Self-Organizing Agents

Agostino Forestiero, Carlo Mastroianni, Giandomenico Spezzano
ICAR-CNR 87036 Rende (CS), Italy
{forestiero,mastroianni,spezzano}@icar.cnr.it
Tel: +39-0984-831725 Fax: +39-0984-839054

Abstract. A Grid information system should rely upon two basic features: the replication and dissemination of information about Grid services and resources, and the distribution of such information among Grid hosts. This paper examines an approach based on ant systems to replicate and map Grid services information on Grid hosts according to the semantic classification of such services. The Ant-based Replication and MApping Protocol (ARMAP) is used to disseminate resource information by a decentralized mechanism, and its effectiveness is evaluated by means of an entropy index. Information is disseminated by agents – ants - that traverse the Grid by exploiting P2P interconnections among Grid hosts. A mechanism inspired by real ants' pheromone is used by each agent to autonomously drive its behavior on the basis of its interaction with the environment. "Swarm Intelligence" emerges from the activity of a high number of ants. The ARMAP protocol enables the use of a semi-informed search algorithm which can drive query messages towards a cluster of peers having information about resources belonging to the requested class. A simulation analysis has been performed to evaluate the performance of ARMAP.

Keywords: ant-inspired systems; Grid; information system; multi agent system; P2P; resource mapping; spatial entropy.

1 Introduction

Grid and peer-to-peer (P2P) computing models share several features, and their integration could bring benefits in both fields. In particular, the use of P2P protocols is expected to improve the efficiency and scalability of information services in large-scale Grid systems [11, 19]. This integration trend can facilitate the deployment and management of a pillar component of Grids: the information system. The Grid information system provides resource discovery and browsing services which are invoked by Grid clients when they need to use hardware or software resources matching given criteria and characteristics. In currently deployed Grid systems, e.g. in the Web Services Resource Framework (WSRF) [23]), the information system is handled through a centralized or hierarchical approach. Nowadays, the Grid community agrees that in large and highly heterogeneous Grids it is more efficient to devise scalable Grid information services based on a distributed approach [11].

In this paper, it is assumed that the resources offered by a service-oriented Grid are Grid services which are deployed and published by Grid Virtual Organizations; for example, in the WSRF framework, published Grid services are Web services having enriched functionalities such as state management. Furthermore, it is assumed that such Grid services are semantically classified according to their features: a *class of resources* is defined as a set of Grid services matching specified properties. Generally, a query is not issued to search a single resource, but to collect information about resources belonging to a given class [6, 13]. After receiving a number of responses, a user or client can choose the resource which is the most appropriate for their purposes.

This paper proposes a novel approach for the construction and management of a Grid information system which allows for an efficient search of resources. It is assumed that an underlying P2P infrastructure interconnects Grid nodes and can be used to explore the Grid. The proposed approach exploits the features of (i) epidemic mechanisms tailored to the dissemination of information in distributed systems [15, 5] and (ii) self adaptive systems in which "Swarm

Intelligence" emerges from the behavior of a high number of agents which interact with the environment [3].

The rationale of using a replication approach is the following: even if a Grid service is provided by a particular Grid host, a number of information documents describing this service should be distributed on the Grid in order to facilitate discovery operations. An information document is usually composed of a description of the service (i.e. a WSDL document and possibly an ontology description) and an URL reference. In the following an information document describing a Grid service or resource will be referred to as a *resource descriptor*.

The ARMAP protocol (*Ant-based Replication and MApping Protocol*), proposed in this paper, aims to disseminate information in a controlled way, in order to maximize the benefit of the replication mechanism and facilitate discovery operations. Replicas are spatially mapped on the Grid so that resource descriptors belonging to the same class are placed in nearby Grid hosts. The mapping of resource descriptors is managed through a multi agent approach, inspired by the model that was introduced in [8] to emulate the behavior of ants which cluster and map items within their environment. This paper proposes a variant of that model, in which items (in our case the resource descriptors) are both replicated and mapped. A number of agents traverse the Grid via the underlying P2P interconnections and copy or move resource descriptors from one host to another, by means of appropriate *pick* and *drop* random functions. In particular, each agent is tailored to pick resource descriptors of a given class from a region in which that class of resources is scarcely present, and drop them in a region where those resources are already being accumulated. A spatial entropy function is defined to evaluate the effectiveness of the ARMAP protocol in the logical reorganization of resources.

Each agent can operate in either the *copy* modality or the *move* modality. If the *copy* modality is used, the agent, when executing a *pick* operation, leaves the resource descriptors on the current host, generates a replica of them, and carries the replicas until it will drop them in another host. Conversely, with the *move* modality, as an agent picks the resource descriptors, it removes them from the current host, thus preventing an excessive proliferation of replicas. In a first phase, the *copy* modality is used to generate an adequate number of resource descriptor replicas on the network. However, the *copy* modality cannot be maintained for a long time, since eventually every host would have a huge number of resource descriptors of all classes, thus weakening the efficacy of resource mapping. Therefore, each generated agent must switch from the *copy* to the *move* modality, and a mechanism must be defined to determine the correct time at which this modality switch must be performed. The analysis of the entropy trend allowed for the definition of a decentralized self-organizing mechanism through which each agent can tune its modality by itself, by analyzing its activeness, i.e. the frequency of *pick* and *drop* operations it performs. This decentralized mechanism, inspired on ants' pheromone [3], guarantees that the ARMAP protocol is fully scalable and fault-tolerant, since agents can tune their behavior without having a global knowledge of the state of the system.

A semi-informed discovery protocol can efficiently exploit the logical reorganization of resources performed by ARMAP: if a number of resource descriptors of the same class are accumulated in a restricted region of Grid hosts, queries for such resources can be driven towards that region, in order to maximize the number of useful responses. Indeed, a discovery operation can be performed in two phases. In the first phase, a query is forwarded through a *blind* mechanism. In the second phase, whenever a query gets close enough to a Grid region specialized in the needed class of resources, the search becomes *informed*: the query is driven towards the specialized Grid region and will easily discover a large number of useful resource descriptors. In reference [9], it is shown that such resource reorganization allows for a discovery of a large number of resources in a limited amount of time.

This paper focuses on the rationale, behavior and performance evaluation of the ARMAP protocol and shows that ARMAP can be effectively used to build a Grid information system in which resource descriptors are properly replicated while keeping the overall entropy as low as possible. In particular, Section 2 introduces ARMAP, discusses the *pick* and *drop* random functions that drive the behavior of mobile agents, and describes the pheromone mechanism used by agents to tune their activeness. Section 3 reports a performance evaluation of ARMAP, achieved by means of an event-driven simulation framework built upon the SWARM simulation environment [17]. Section 4 discusses related work and compares our approach with several others proposed recently. Section 5 concludes the paper.

2 A Multi-Agent Protocol for Mapping Resources on the Grid

This Section introduces and discusses the ARMAP protocol. ARMAP aims to disseminate Grid resource descriptors and spatially map them on the Grid according to their semantic classification, in order to gather a consistent number of resource descriptors of the same class in a restricted region of the Grid. It is assumed that the resources have been previously classified into a number of classes Nc, according to their semantics and functionalities (see [6] and [13]).

ARMAP exploits the random movements and operations of a number of mobile agents that travel the Grid using P2P interconnections. This approach is inspired by biological systems, in particular by ant systems [3, 7, 8], in which swarm intelligence emerges from the collective behavior of very simple mobile agents (ants), and a complex overall objective is achieved.

In ARMAP, each mobile agent can pick a number of resource descriptors on a Grid host, carry such descriptors while moving form host to host, and deposit them on another Grid host. Initially, it is assumed that each agent is “class-specific”, i.e. it manages the resource descriptors of only one class. This assumption will be released later. Section 2.1 describes the basic features of the ARMAP protocol (agent movements and *pick* and *drop* operations), whereas the approach used to tackle the dynamic nature of the system is discussed in Section 2.2. Section 2.3 introduces the entropy function used to evaluate the effectiveness of ARMAP, and explains the advantage of switching the ARMAP modality from *copy* to *move*. Furthermore, Section 2.3 discusses a decentralized approach, based on ants’ pheromone, that is used by each agent to evaluate the correct time at which it must operate the modality switch. Finally, Section 2.4 discusses the role of the ARMAP protocol in the design of a Grid information system.

2.1 ARMAP basic operations

Agent Movement

Each agent travels over the Grid through P2P interconnections among Grid hosts. The ARMAP protocol has been analyzed in a P2P network in which peers are arranged in a mesh topology, as in the SWARM simulator [17], and each peer is connected to at most 8 neighbor peers, including horizontal, vertical and diagonal neighbors. The mesh topology was chosen to achieve a more intuitive and immediate graphical representation of the system evolution, which helps to understand the involved dynamics. However, it is also assumed that peers frequently leave and re-join the network, as discussed later, which assures that at a specific time a peer is actually connected to a random number of active peers (at most 8), so relaxing the rigid mesh assumption.

At random times, each agent makes a random number of hops along the P2P network (the maximum number of hops `Hmax` is a protocol parameter), executes the agent's algorithm specified by the ARMAP protocol, and possibly performs a *pick* or *drop* operation.

Pick operation

Once an agent specialized in a class `Ci` gets to a Grid host, if it is currently unloaded, it must decide whether or not to pick the resource descriptors of class `Ci` that are managed by that host. In order to achieve the replication and mapping functionalities, a *pick* random function `Ppick` is defined with the intention that the probability of picking the resource descriptors of a given class decreases as the local region of the Grid accumulates such descriptors. This way resource mapping is further facilitated.

The Ppick random function, defined in formula (1), is the product of two factors, which take into account, respectively, the relative accumulation of resource descriptors of a given class (with respect to the other classes), and their absolute accumulation (with respect to the initial number of resource descriptors of that class).

$$(1)\ P_{pick} = \left(\frac{k_1}{k_1 + f_r}\right)^2 \cdot \left(\frac{(f_a)^2}{k_2 + (f_a)^2}\right)^2$$

In particular, the fr fraction is computed as the number of resource descriptors of class Ci accumulated in the peers located in the *visibility region* divided by the overall number of resource descriptors that are accumulated in the same region. The visibility region includes all the peers that are reachable from the current peer with a given number of hops (i.e. within the *visibility radius*). Here it is assumed that the visibility radius is equal to one, so that the visibility region is composed of at most 9 hosts, the current one included. It is assumed that each host is informed, through a soft state mechanism, about the resource descriptors that are maintained by the hosts located within the visibility region.

The fa fraction is computed as the number of resource descriptors of class Ci that are *owned* by the hosts located in the visibility region out of the number of resource descriptors that are *presently* maintained by such hosts, including the descriptors deposited by the agents. The inverse of fa gives an estimation of the extent to which the hosts under interest have accumulated resource descriptors of class Ci so far. k1 and k2 are threshold constants which are both set to 0.1.

If the ARMAP protocol works in the *copy* modality, when an agent picks the resource descriptors of class Ci, it leaves a copy of them in the current host; conversely, if the *move* modality is assumed, such resource descriptors are removed from the current host. In the latter case, the current host will only maintain the descriptors of class Ci that it owns, but loses all the information about the descriptors of class Ci which have been deposited by the agents.

Drop operation

Whenever an agent specialized in a class `Ci` gets to a new Grid host, it must decide whether or not to drop the resource descriptors of class `Ci`, in the case that it is carrying any of them. As opposed to the *pick* operation, the dropping probability is directly proportional to the relative and absolute accumulation of resource descriptors of class `Ci` in the visibility region. The `Pdrop` function is shown below.

$$(2)\ P_{drop} = \left(\frac{f_r}{k_3 + f_r}\right)^2 \cdot \left(\frac{k_4}{k_4 + (f_a)^2}\right)^2$$

In (2), the threshold constants `k3` and `k4` are set to 0.3 and 0.1, respectively.

A high-level description of the ARMAP algorithm executed by each agent is given in Fig. 1: the role of the protocol modality (*copy* or *move*) is highlighted.

```
// Na = number of agents: each one is specialized in a class of resources
// Hmax = max number of P2P hops that an agent can perform between two
//        successive operations
// mod = ARMAP modality (copy or move)
For each agent a (specialized in class Ci) do forever {
  Compute integer number h between 1 and Hmax;
  a makes h P2P hops;
  if (a is unladen) {
    compute Ppick;
    draw random real number r between 0 and 1;
    if (r<=Ppick) then {
      pick resource descriptors of class Ci from current host;
      if (mod == move)
        remove resource descriptors of class Ci from current host;
    }
  }
  else {
    compute Pdrop;
    draw random real number r between 0 and 1;
    if (r<=Pdrop) then
      drop resource descriptors of class Ci into current host;
  }
}
```

Fig. 1. High-level description of the ARMAP algorithm

So far, only class-specific agents were considered: with this assumption, each peer casually selects the class of resources in which the generated agent will be specialized. However, to improve performance, it is also possible to generate *generic* agents, which are able to pick and drop resource descriptors belonging to all the resource classes or a subset of them. In such a case, the algorithm shown in Fig.1 is slightly modified: the agent computes the *pick* and *drop* random functions separately for each class it can manage. This way an agent may pick the resource descriptors of class `Ci` from a Grid host, and drop the descriptors of another class `Cj` into the same host. The performance increase obtained with generic agents will be shown in Section 3.3.

2.2 Managing a dynamic Grid with ARMAP

In a dynamic Grid, peers can, more or less frequently, go down and reconnect again. As a consequence of this dynamic nature, two different and opposite issues must be tackled. The first is related to the management of new resources provided by new or reconnected hosts: once all the agents have switched to the *move* modality, it becomes impossible to replicate and disseminate descriptors of new resources; hence agents cannot live forever, and must be gradually be replaced by new agents that set off in the *copy* modality. At the same time, the system must deal with obsolete resources, i.e. with resources, provided by peers that has left the system, that are no more exploitable.

To tackle these two issues, it is necessary to manage the lifecycle and the gradual turnover of agents, and control the overall number of agents that travel the Grid. The proposed solution is to correlate the lifecycle of agents to the lifecycle of peers. The average connection time of a specific peer is generated according to a Gamma probability function, with an average value set to a parameter `PlifeTime`, which is equal to 100,000 seconds. Use of the Gamma distribution assures that the Grid contains very dynamic hosts, which frequently disconnect and rejoin the network, as well as much more stable hosts.

When joining the Grid, a peer generates a number of agents given by a discrete Gamma stochastic function, with average `Ngen`, and sets the life time of these new agents to the average connection time of this peer. This setting assures that (i) the relation between the number of peers and the number of agents is maintained over the time (more specifically, the overall number of agents is approximately equal to the number of active peers times `Ngen`) and (ii) a proper turnover of agents is achieved, which allows for the dissemination of new resource descriptors, since new agents start with the *copy* modality.

Every time a peer disconnects from the Grid, it loses all the resource descriptors previously deposited by agents, thus contributing to the removal of obsolete information. Furthermore, a soft state approach [16] is adopted to avoid, especially in stable nodes, the accumulation of descriptors to obsolete or non existent resources. Periodically, each node contacts the nodes which own the original resources and refresh the content of the related descriptors: if this refreshment fails for a resource, the corresponding descriptor will be removed. In general, the refreshment rate should depend on the amount of memory available on the node, on the amount of traffic which is tolerable on the network, and on the characteristics of the resources, i.e. on their more or less dynamic nature. In this paper, it is assumed that the value of the refreshment period is twice the average connection time `PlifeTime`.

The dynamic nature of resources is another important issue that must be tackled in a heterogeneous system, since discovery of dynamic resources (e.g., amount of free main memory in a host) is more critical than discovery of static resources (e.g., read-only documents). The rate at which resources are updated has a significant impact on the availability of dynamic resources, as discussed in [4]. The ARMAP protocol can be enhanced to cope with this issue, with the purpose of disseminating dynamic resources more rapidly and efficiently than static resources. To achieve this, pick and drop probabilities are modified through appropriate time-dependent factors. In particular, the pick probability, defined in formula (1), is multiplied by an additional factor, `TFpick`, which is reported in formula (3).

$$(3)\ TF_{pick} = \frac{k_5}{k_5 + \frac{U_{peer} - U_{resource}}{U_{resource}}}$$

In this factor, the parameter `Uresource`, maintained by each agent, is the estimated “age” of Grid resources, where the age is defined as the time elapsed since the last time that the resource has been updated. The parameter `Uresource` is computed by averaging the age of the resources maintained by the hosts that an agent has visited along its life. On the other hand, `Upeer` is the average age of the resources maintained by the peer that an agent is currently visiting. Finally, the value of the parameter `k5`, not lower than 5, guarantees that the pick probability never exceeds a value of 1.

It can be observed that the average value of `TFpick` is equal to 1, which assures that the overall dissemination of resources (regardless of their age) is not biased by the new factor. However, if the current peer maintains resources which have been updated very recently (i.e., `Uresource` is higher than `Upeer`), the value of `TFpick` will be higher than 1; hence the overall pick probability increases, and the pick operation is favored. Conversely, the value of the new factor is lower than 1 for static or dated resources.

Analogously, the drop probability function defined in formula (2) is multiplied by a new factor `TFdrop`, reported in formula (4). Preliminary studies show that the use of these factors efficiently fosters the dissemination of up-to-date/dynamic resources. Moreover, the parameter `k5` can be used to tune the ARMAP behavior. In fact, a more differentiated management of dynamic and static resources is obtained with lower values of `k5`, and vice versa. Current work is tailored to achieve a detailed performance analysis of time-aware pick and drop probability functions.

$$(4)\ TF_{drop} = \frac{k_5}{k_5 + \frac{U_{resource} - U_{peer}}{U_{resource}}}$$

It is worth mentioning that the described approach for handling a dynamic Grid implicitly manages any unexpected peer fault, because this occurrence is processed in exactly the same way as a peer disconnection. Indeed, the two events are indistinguishable, since (i) a peer does not have to perform any procedure before leaving the system, and (ii) in both cases the resource descriptors that the peer has accumulated so far are removed.

2.3 Spatial entropy and pheromone mechanism

A spatial entropy function is defined to evaluate the effectiveness of the ARMAP protocol, as shown in formula (5). For each peer `p`, the entropy `Ep` gives an estimation of the extent to which the visibility region centered in `p` has accumulated resource descriptors belonging to one class. This function, inspired by the Shannon entropy formula, is constructed upon the value of `fr(i)`, defined as the fraction of resource descriptors of class `Ci` within the visibility region. The `Ep` formula is normalized, so that possible values are comprised between 0 (when all the resource descriptors in the visibility region belong to just one class) and 1 (when the resource descriptors are equally distributed among the different classes). The overall spatial entropy `E` of the network is defined as the average of the entropy values `Ep` computed at all Grid hosts.

$$(5)\ Ep = \frac{\sum_{i=1..Nc} fr(i) \cdot \log_2 \frac{1}{fr(i)}}{\log_2 Nc}, \quad E = \frac{\sum_{p \varepsilon Grid} Ep}{Np}$$

Simulation runs were executed to evaluate the correct time at which the modality switch (from *copy* to *move*) should be performed in order to minimize the spatial entropy. Results are given in Section 3.2. The assumption here is that each agent knows the value of the overall entropy at every instant of time. However, in the real world an agent maintains only a local view of the environment, and cannot determine its behavior on the basis of global system parameters such as the overall system entropy. Therefore, a method is introduced with the purpose of enabling a single agent to perform the modality switch only on the basis of local information. Such a method is based on the observation that an increase in the overall entropy value corresponds to a significant decrease in the activity of agents, i.e. in the frequency of *pick* and *drop* operations that are performed by agents. Indeed a low agent activity is a clue that a high degree of resource reorganization has already been achieved.

Accordingly, a single agent can evaluate its own activeness by using a pheromone mechanism [22]. In particular, at given time intervals, i.e. every 2000 seconds, each agent counts up the number of successful and unsuccessful *pick* and *drop* operations (a *pick* or *drop* operation attempt is considered *successful* when the operation actually takes place – i.e. when the random number extracted is lower than the value of the operation probability function, see Fig.1). At the end of each time interval, the agent makes a deposit into its pheromone base, by adding a pheromone amount equal to the fraction of unsuccessful operations with respect to the total number of operation attempts. An evaporation mechanism is used to give a higher weigh to the recent behavior of the agent. In more details, at the end of the i-th time interval, the pheromone level Φi is computed with formula (6).

(6) $\Phi i = Ev \cdot \Phi i - 1 + \varphi i$

The evaporation rate `Ev` is set to 0.9, and φi is the fraction of unsuccessful operations performed in the last time interval. As soon as the pheromone level exceeds a given threshold `Tf`, the agent realizes that the frequency of *pick* and *drop* operations has remarkably reduced, and switches its protocol modality from *copy* to *move*. The value of `Tf` is set by observing the global system behavior, as explained in Section 3.2. The choice of updating the pheromone level every time interval, instead of every single operation, has been made to fuse multiple observations into a single variable, so giving a higher strength to agents' decisions.

2.4 Role of the ARMAP protocol for the design of P2P information systems in Grids

We believe that the ARMAP protocol can be a significant step towards the efficient design and implementation of a P2P-based information system in a Grid environment. However, to better understand its role, it is necessary to discuss how ARMAP can be related to the overall information system design process, which could be composed of the following three components/steps:

1. classification of Grid resources;
2. replication and mapping of resource descriptors with the ARMAP protocol;
3. construction of a discovery service.

The first component allows users to identify the features and functionalities of the resources they need (i.e. a particular resource class). Classification of resources can be performed with different techniques, as discussed in Section 4.

The second component, ARMAP, is the subject of this paper. The third component, the discovery service, assumes that the Grid resources have been logically reorganized through the ARMAP protocol, or at least that ARMAP is working while discovery requests are being forwarded. The use of ARMAP permits to handle discovery requests by combining the flexible and scalable features of a *blind* approach with the efficiency and fastness of an *informed* approach. A discovery protocol that takes full advantage of the ARMAP work is briefly described in the following and is analyzed in [9]. Query messages first travel the Grid network with a *blind* mechanism, according to the random walk technique [6]; however, the search procedure is turned into an *informed* one as soon as a query message approaches a region which has gathered resource descriptors belonging to the requested class. A number of peers, i.e. the peers that collect a large number of resource descriptors belonging to a specific class, are elected as *representative peers*, and assume the role of attractors for query messages. This way, a query can be routed towards the nearest representative peer of the class under consideration, and there it will easily find a large number of useful results.

3 Simulation Analysis

In this section we introduce and discuss the parameters and performance indices used to evaluate the ARMAP protocol (in Section 3.1), than we report and discuss simulation results which demonstrate the protocol effectiveness in a Grid environment. In particular, the trend of the overall system entropy (Section 3.2) confirms the advantage of using the ARMAP protocol with two modalities, *copy* and *move*. In Section 3.2, it is also shown how agents can autonomously choose the protocol modality with a pheromone mechanism. Section 3.3 analyzes the performance of ARMAP achieved by varying the number, type and mobility of agents. Finally, Section 3.4 investigates the impact of the number of resource classes and the Grid size on ARMAP behavior.

3.1 Simulation Parameters and Performance Indices

Simulation runs were performed by exploiting the software architecture and the visual facilities offered by the SWARM environment [17]. SWARM is a software package for multi-agent simulation of complex systems, developed at the Santa Fe Institute.

Table 1 and Table 2 report, respectively, the simulation parameters and the performance indices used in our analysis. The number of peers Np (or Grid size) was varied from 225 (a 15x15 grid) to 10000 (a 100x100 grid). The number of resources (Grid services) owned and published by a single peer is determined through a gamma stochastic function having an average value equal to 15 (see [11]). Grid resources are classified in a number of classes Nc varying from 3 to 9; the class to which each resource belongs is selected by the simulator with an uniform random function.

When an agent moves to a destination peer, it performs the algorithm shown in Fig. 1, possibly picks and/or drops a number of resource descriptors, and finally moves to another peer. Each peer generates a random number of agents with average equal to Pgen: by modulating this parameter, the overall number of agents Na was varied from Np/4 to 2Np. Both class-specific and generic agents were considered in the simulation analysis. The average connection time of a peer

(PlifeTime) is set to 100,000 sec. Each time a peer joins the Grid, it generates its current connection time by using a stochastic Gamma function with average PlifeTime. The average time between two successive agent movements (i.e. between two successive evaluations of the *pick* and *drop* functions) is set to 60 sec. To move towards a remote host, an agent exploits P2P interconnections among Grid hosts. The number of hops performed within a single agent movement is a random function: the maximum number of hops, Hmax, is varied from 1 to D/2, where D is the square root of Np. Finally, the visibility radius Rv, defined in Section 2.1, is set to 1.

Among the performance indices, the overall entropy E, defined in Section 2.3, is the most important one, since it is used to estimate the effectiveness of the ARMAP protocol in the logical reorganization of resources. The Nrep index is defined as the mean number of replicas per resource that are available (i.e. that are maintained by active peers) on the Grid. Fop is the frequency of successful operations (*pick* and *drop*) that are performed by each agent; this index gives an estimation of agents' activeness and system stability, since successful operations are less frequent when a low level of entropy has been achieved. Finally, the traffic load L is defined as the number of hops per second that are performed by all the active agents.

Table 1. Simulation parameters

Parameter	**Value**
Grid size (number of peer), **Np**	225 to 10000
Mean number of resources published by a peer	15
Number of classes of resources, **Nc**	3 to 9
Number of agents (class-specific or generic), **Na**	Np/4 to 2Np
Mean life time of a peer, **Plifetime**	100,000 s
Mean amount of time between two successive movements of an agent	60 s
Maximum number of hops, **Hmax**	1 to D/2
Visibility radius, **Rv**	1

Table 2. Performance indices

Performance Index	**Definition**
Mean Spatial Entropy, **`E`**	Defined in Section 2.3
Mean number of replicas, **`Nrep`**	Mean number of replicas per resource (included the original copy) which are available in the Grid
Mean frequency of successful operations, **`Fop`**	Mean number of successful operations - *pick* or *drop* - that are performed by a single agent per unit time (operations/s)
Traffic load, **`L`**	Mean number of hops that are performed by all the agents of the Grid per unit time (hops/s)

3.2 Protocol modality and pheromone mechanism

The results shown in this section and in Section 3.3 are relative to simulations performed for a network with 2,500 (50x50) hosts that provide resources belonging to 5 different classes, unless otherwise stated. In particular, results shown in this section are obtained by setting the number of agents `Na` to half the number of peers `Np` (`Pgen` is set to 0.5), and the maximum number of hops `Hmax` to 3. All agents are *generic*, therefore they can pick and drop resource descriptors belonging to every class. A comparison with class-specific agents is shown in Section 3.3.

Graphs of performance measures, reported versus time, illustrate the gradual effect of the ARMAP protocol in the reorganization and mapping of resource descriptors. Figure 2 shows that the exclusive use of the *copy* modality is not effective: the overall system entropy decreases very fast in a first phase, but increases again when the agents create an excessive number of replicas (Figure 3). Indeed, if agents continue to create new replicas, eventually all peers will possess a huge number of resource descriptors of all classes, thus completely undoing the mapping work. The curve labeled as “copy/move” in Figure 2 is obtained by switching the protocol modality, from *copy* to *move*, when it is observed that the entropy function (calculated every 2000 seconds) increases two times in succession. The effect of this switch is very interesting: the system entropy not only stops increasing but decreases to much lower values. This behavior is the effect of the action of the agents, which do not generate further replicas, as shown in Figure 3, but continue their work in creating low-entropy regions specialized in particular classes of resources. In these tests a *global* parameter of the system – the value of the overall system entropy - is assumed to be known by all the agents. In the following, this approach will be referred to as the *global* one.

Figure 2 and 4 show that, with the *copy* modality, at approximately the same time as the system entropy begins to increase, the frequency of operations performed by a single agent stops increasing and begins to decrease. This experimental result can be exploited to define a *local* approach that allows agents to perform the modality switch on their own. This is achieved through the pheromone mechanism explained in Section 2.3. However, it is necessary to set a proper value of the pheromone threshold `Tf`, that is used by agents to realize when the modality switch must be performed. This value was set with the following method: by running simulations with the *global* approach, we calculated the mean pheromone value of a generic agent at the time at which the system entropy begins to increase. This value was used as the pheromone threshold in the *local* approach. Figure 5 and 6 compare the curves of system entropy and average number of replicas obtained with the *global* and *local* approaches. It is seen that the local approach is effective, since it approximates the global one very strictly.

Figure 7 gives a graphical description of the mapping process: the 50x50 square grid represents the simulated network, and each cell represents a peer. To facilitate the comprehension of the process, the number of resource classes is set to 3. The *local* approach is assumed. Each cell is marked with a circle, a square or a cross: such symbols correspond to class `C1`, `C2` and `C3`, respectively. The presence of a circle means that in the corresponding node the number of resource descriptors of class `C1` exceeds the numbers of descriptors belonging to the other 2 classes; the presence of a square or a cross has an analogous meaning. Furthermore, the thickness of the symbol represents how much the peer is specialized in one class, i.e. it is proportional to the difference between the number of descriptors belonging to the most numerous class and the number of descriptors belonging to the second most numerous class. To depict the gradual accumulation of resource descriptors, four snapshots of the network are shown. Such snapshots are taken at time 0 (when the ARMAP protocol is started), and after 25,000, 50,000 and 100,000 seconds, respectively. The sequence of snapshots confirms the effectiveness of the accumulation process, which is very fast in the first phase, thus allowing for a notable performance enhancement of discovery operations after a short amount of time (see [9]).

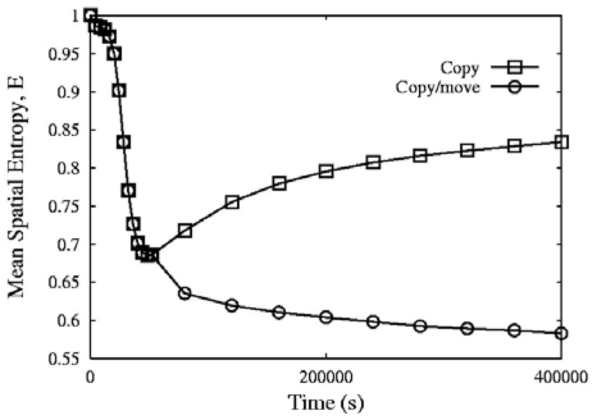

Fig. 2. Mean spatial entropy vs. time; comparison between ARMAP used with the copy modality and ARMAP with the copy/move modality switch.

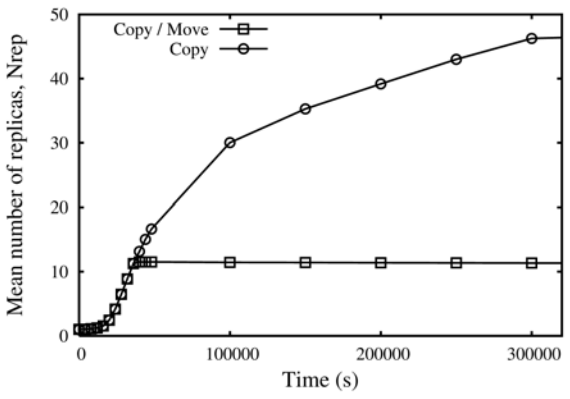

Fig. 3. Mean number of replicas vs. time; comparison between ARMAP used with the copy modality and ARMAP with the copy/move modality switch.

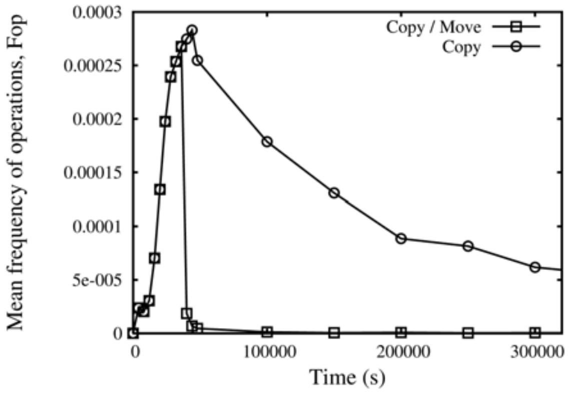

Fig. 4. Mean frequency of operations vs. time; comparison between ARMAP used with the copy modality and ARMAP with the copy/move modality switch.

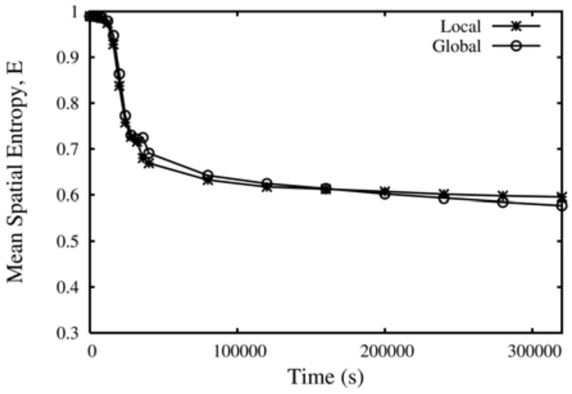

Fig. 5. Mean spatial entropy vs. time; comparison between global and local approaches.

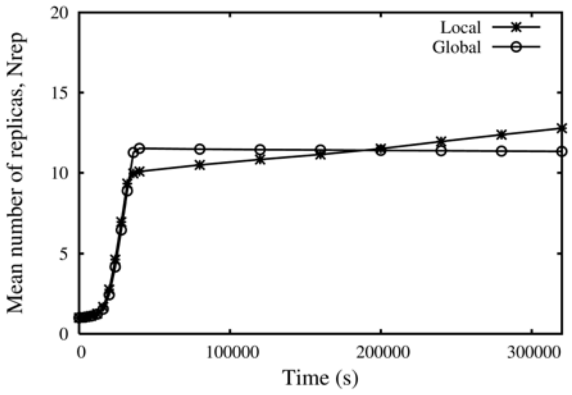

Fig. 6. Mean number of replicas vs. time; comparison between global and local approaches.

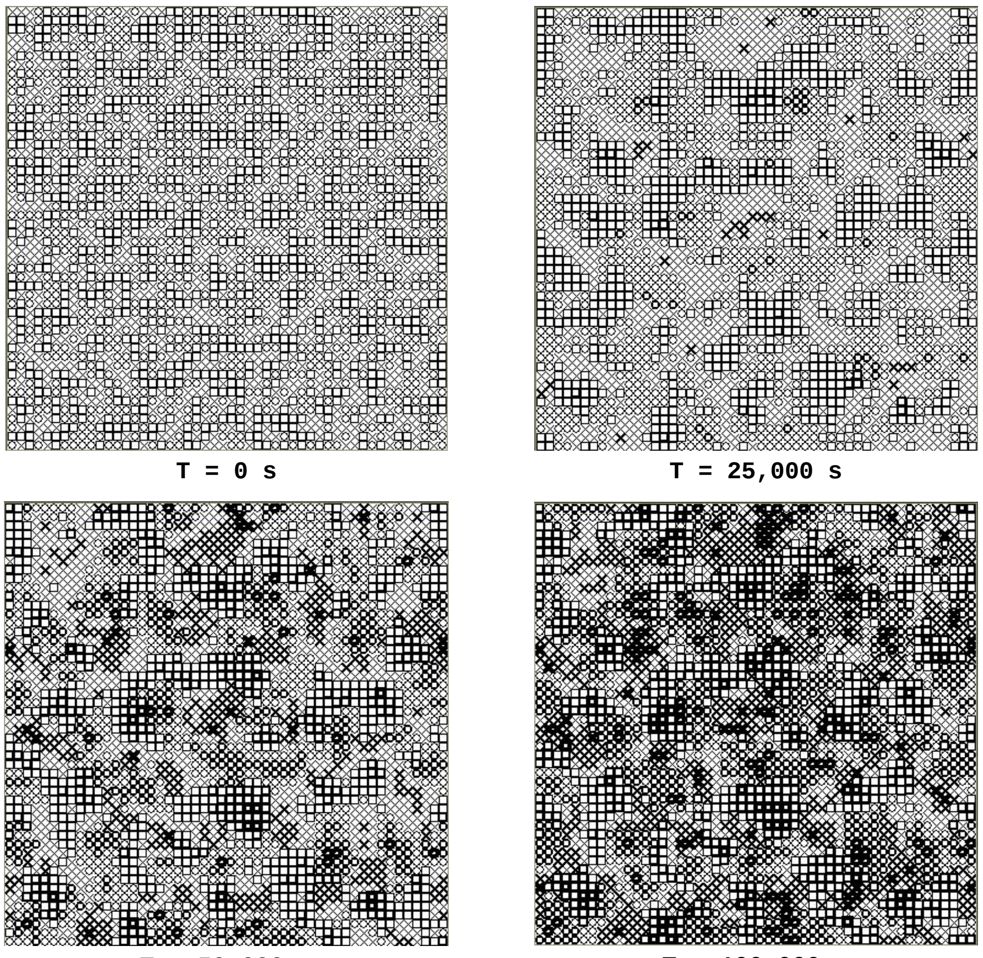

Fig. 7. Gradual mapping of resources in a network with 2,500 peers and 3 resource classes. Each peer contains a symbol (circle, square or cross) that corresponds to the class of resources which is the most numerous in such a peer. The symbol thickness represents the level of specialization of the peer.

3.3 Number and mobility of agents

In Section 3.2 the number of agents `Na` was set to `Np`/2, and the parameter `Hmax` was set to 3. The aim of this section is to analyze how the performance of ARMAP is affected by the number, type and mobility of agents. A first consideration is that the value of the pheromone threshold `Tf` depends on the number of agents per peer, i.e. on the ratio `Na`/`Np`. Table 3 shows, for different values of the ratio `Na`/`Np`, the corresponding instants of time at which the switch modality should be performed (calculated with the *global* approach), and the mean pheromone levels reached by a generic agent at those instants. It can be observed that as the number of agents increases, the time interval after which such agents should stop creating new replicas becomes shorter, since the overall activity of agents is higher. As a consequence, the pheromone threshold `Tf` decreases as well: a single agent will reach the threshold after a shorter interval of time.

Figure 8 reports the trend of the overall system entropy obtained with different numbers of agents; the *local* approach is used and the pheromone thresholds are set to the corresponding values shown in Table 3. An increase in the number of agents makes the system entropy decrease faster and reach lower values. However, a higher activity of agents also causes an increase in the traffic load (Figure 9). A correct setting of the ratio `Na`/`Np` should take into account the trend of these performance indices and in general should depend on system features and requirements, for example on the system capacity of sustaining a high traffic load.

Figure 10 compares the trend of the overall system entropy obtained with generic and specialized (class-specific) agents: in both cases the number of agents is set to `Np`/2. The performance increase achieved with the use of generic agents is confirmed, since they permit to obtain much lower values of the system entropy.

Finally, the effect of the parameter `Hmax` is analyzed in Figures 11 - 13. Here, the number of (generic) agents `Na` is set to `Np`/2. An increase in `Hmax` speeds up the mapping of resources (Figure 11), since an agent can move resource descriptors between more distant peers, and causes an increase in the mean number of replicas (Figure 12) and in the traffic load (Figure 13). Again, a correct setting of `Hmax` should take into consideration the network requirements. We chose to set `Hmax` to 3, because this is the minimum value that permits to move an agent between two visibility regions that are completely disjoint, given that the visibility radius is set to 1.

Table 3. Modality switch time and pheromone level at switch time with different numbers of agents.

Number of agents Na	Modality switch time (s)	Pheromone level at switch time
Np/4	64,000	8.33
Np/2	42,000	7.73
Np	22,000	6.17
2Np	16,000	5.20

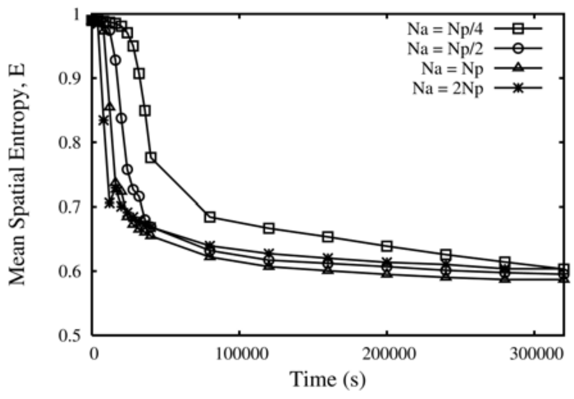

Fig. 8. Mean spatial entropy vs. time, with different numbers of agents.

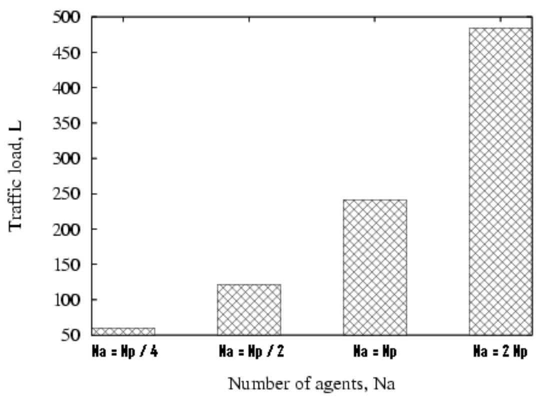

Fig. 9. Mean traffic load (hops/s), with different numbers of agents.

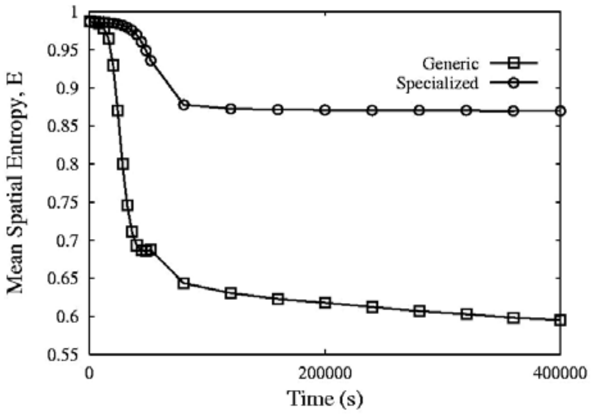

Fig. 10. Mean spatial entropy vs. time. Comparison between generic and specialized agents. The number of agents Na is set to Np/2.

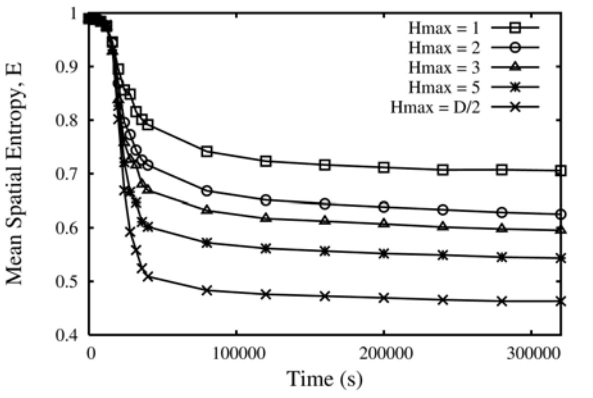

Fig. 11. Mean spatial entropy vs. time, with different values of Hmax.

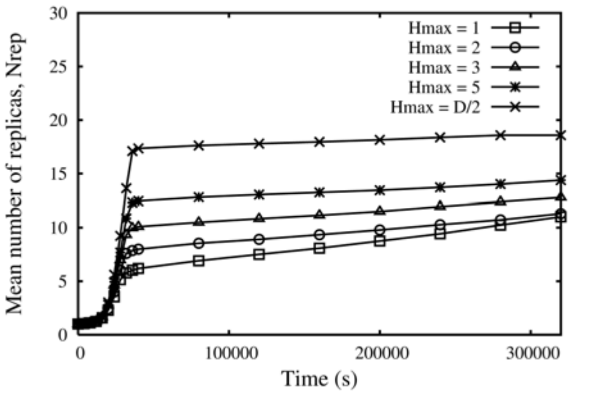

Fig. 12. Mean number of replicas vs. time, with different values of Hmax.

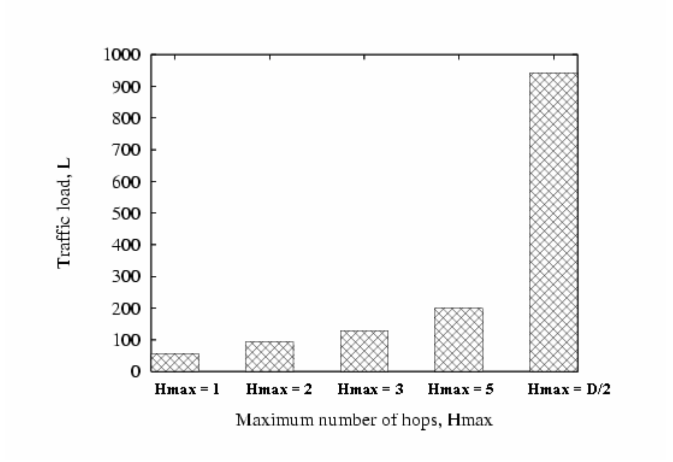

Fig. 13. Mean traffic load (hops/s), with different values of Hmax.

3.4 Number of classes and network size

So far, it was assumed that the resources are pre-classified in exactly 5 classes, and that the network is composed of 2,500 (50x50) peers. The aim of this section is to investigate how the number of classes Nc and the Grid size Np affect the behavior of the ARMAP protocol. The results shown in this section are obtained with Na=Np/2 and Hmax=3.

The results shown in Table 4 were obtained by adopting the *global* approach with different values of Nc, and Np equal to 2,500. As the number of classes increases, the optimum modality switch time increases. In fact, when the number of classes is higher, the agents have to replicate a larger number of resource descriptors to achieve a significant reduction of the system entropy. As a consequence, the pheromone threshold must be higher, as also appears from Table 4.

Figure 14 shows the trend of the overall entropy achieved with the *local* approach and the pheromone threshold values reported in Table 4. It is interesting to note that initially the entropy function decreases more rapidly when the number of classes is small; the reason is that in the first phase it is easier to separate resource descriptors belonging to 3 classes rather than 7 or 9. However, in the long term the agents' work permits to reach entropy values that are smaller and smaller as the number of classes increases. Indeed, the entropy curves cross and completely invert their order within a time interval time comprised between 25,000 and 50,000 seconds from the start of the simulation.

A set of simulation runs was performed to evaluate the effect of the network size on performance results. Figures 15 and 16 show, respectively, the overall entropy and the mean number of replicas for different values of the number of peers `Np`. The number of classes is set to 5. It can be seen that the effect of the Grid size is almost irrelevant: we can deduce that a single peer does not need to have any knowledge about the size of the network to regulate its behavior. Other simulation results, not reported here, show that also the mean number of resources published by each peer has little effect on the behavior of the ARMAP protocol.

As a conclusion, an effective tuning of ARMAP can be obtained by setting the pheromone threshold to a proper value, as shown in Tables 3 and 4. This value essentially depends on two parameters: the number of agents per peer, i.e. the ratio `Na/Np`, and the number of classes `Nc`. The former parameter is known because it is equal to `Pgen`, the average number of agents that a peer generates when joining the Grid. The latter parameter is also known if it is assumed that the resources are categorized in a predetermined number of classes.

Table 4. Modality switch time and pheromone level at switch time with different numbers of classes of resources.

Number of classes of resource (`Nc`)	Modality switch time (s)	Pheromone level at switch time
3	40,000	7.69
5	42,000	7.73
7	48,000	7.97
9	56,000	8.18

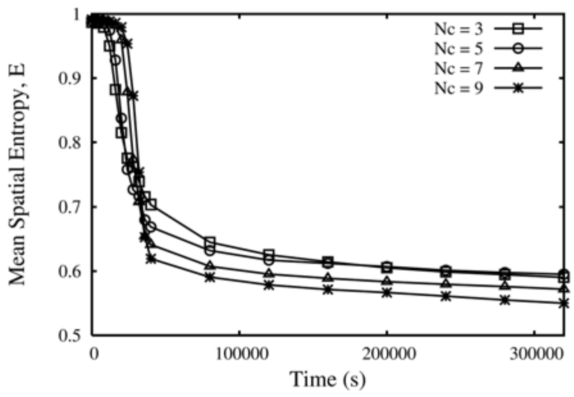

Fig. 14. Mean spatial entropy vs. time, with different values of the number of classes `Nc`.

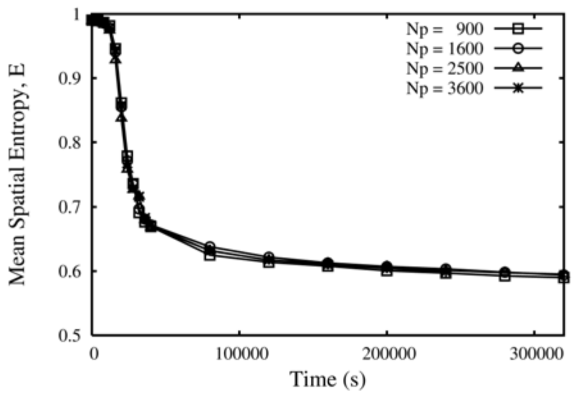

Fig. 15. Mean spatial entropy vs. time, with different values of the network size `Np`.

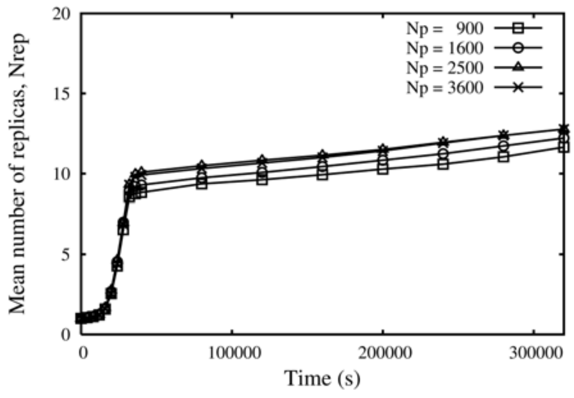

Fig. 16. Mean number of replicas vs. time, with different values of the network size `Np`.

4 Related Work

Since Grid hosts provide a large set of distributed and heterogeneous resources, an efficient Grid information service is a pillar component of a Grid. Current Grid information services offer centralized or hierarchical information services, but this kind of approach is going to be replaced by a decentralized one, supported by P2P interconnection among Grid hosts.

In the last years, a number of P2P techniques and protocols have been proposed to deploy Grid information services: for example, super-peer networks [13, 24] achieve a balance between the inherent efficiency of centralized search, and the autonomy, load balancing and fault-tolerant features offered by distributed search.

P2P search methods can be categorized as *structured* or *unstructured*. The structured approach assumes that hosts and resources are made available on the network with a global overlay planning. In Grids, users do not usually search for single resources (e.g. MP3 or MPEG files), but for software or hardware resources that match an extensible set of features. Accordingly, while structured protocols, based on highly structured overlays and Distributed Hash Tables, are usually very efficient in file sharing P2P networks, unstructured or hybrid protocols seem to be preferable in largely heterogeneous Grids. Unstructured search methods can be further classified as *blind* or *informed* [20]. In a blind search (e.g. using flooding or random walks [12]), nodes hold no information that relates to resource locations, while in informed methods (e.g. routing indices [6] and adaptive probabilistic search [21]), there exists a centralized or distributed information service that drives the search for the requested resources. As discussed in Section 2.4, the approach presented in this paper aims to combine the flexible and scalable features of a blind approach with the efficiency and rapidity of an informed approach.

The ARMAP protocol introduced in this paper is based on the features of Multi-Agent Systems (MAS), and in particular of ant-based algorithms. A MAS can be defined as a loosely coupled network of problem solvers (agents) that interact to solve problems that are beyond the individual capabilities or knowledge of each problem solver [18]. Research in MASs is concerned with the study, behavior, and construction of a collection of autonomous agents that interact with each other and the environment. Ant-based algorithms are a class of agent systems which aim to solve very complex problems by imitating the behavior of some species of ants [3].

The Anthill system [2] is a framework that supports the design, implementation and evaluation of P2P applications based on multi-agent and evolutionary programming. In Anthill, societies of adaptive agents travel through the network, interacting with nodes and cooperating with other agents in order to solve complex problems. Reference [7] introduces an approach based on ant behavior and genetic algorithms to search resources on a P2P network. A sub-optimal route of query messages is learnt by using positive and negative pheromone feedbacks and a genetic method that combines and improves the routes discovered by different ants. Whereas in [7] the approach is tailored to improve search routes with a given distribution of resources in the network, this paper proposes to logically reorganize and replicate the resources in order to decrease the intrinsic complexity of discovery operations. Instead of directly using ant-based algorithms to search resources, the ARMAP protocol exploits an ant-based replication and mapping algorithm to replicate and reorganize resource descriptors according to their categorization.

Our protocol is a variant of the basic sorting algorithm proposed in [8]. However, the latter assumes that only one item can be placed in a cell of a toroidal grid, and items can only be moved form one cell to another, without the possibility of performing any replication. Conversely, the ARMAP protocol assumes that a cell (i.e. a Grid host) can store several items (i.e. resource descriptors) and agents can create many replicas of the same item.

The problem of driving the behavior of a single agent, which should autonomously be able to take actions without having an overall view of the system, is discussed in [22]. There, a decentralized scheme, inspired by insect pheromone, is used to limit the activity of a single agent when it is no more concurring to accomplish the system goal. In this paper, a similar approach is used to drive the behavior of an agent, in particular to evaluate the correct time at which it should switch from the *copy* to the *move* modality.

Information dissemination is a fundamental and frequently occurring problem in large, dynamic, distributed systems, since it consents to lower query response times and increase system reliability. Reference [10] proposes to disseminate information selectively to groups of users with common interests, so that data is sent only to where it is wanted. In this paper, instead of classifying users, it is proposed to exploit a given classification of resources: resource descriptors are replicated and disseminated with the purpose of creating low-entropy regions of the network that are specialized in a specific class of resources. The so obtained information system allows for the definition and usage of a semi-informed search method, as explained in Section 2.4.

The ARMAP protocol assumes that a classification of resources has already been performed. This assumption is common in similar works: in [6], performance of a discovery technique is evaluated by assuming that resources have been previously classified in 4 disjoint classes. Classification can be done by characterizing the resources with a set of parameters that can have discrete or continuous values. Classes can be determined with the use of space filling curves that represent different parameters on one dimension [1]. Space filling curves are also used by [14] to support a self-organized grouping method for efficient Grid resource discovery. A group is formed by a collection of nodes with similar characteristics, i.e., with similar values of a space filling function based on multiple attributes. A discovery query is first directed to the leader of a group with the pre-specified characteristics similar to the request made in the query. Then the leader node will forward the query to the nodes that maintain useful resources, or otherwise to another leader that is likely to satisfy the query.

5 Conclusions

This paper introduces an approach based on multi agent systems to manage resources and construct an efficient information system in Grids. The presented ARMAP protocol accomplishes two main purposes: it replicates resource information on Grid hosts, and gathers information related to similar resources in restricted regions of the Grid. The work is performed by ant-like agents which travel over the network by exploiting P2P connections and move resource information between Grid hosts. A pheromone mechanism is used to control the behavior of a single agent and, from a global point of view, to control the dissemination of resource information and keep the overall system entropy as low as possible. A simulation study allowed for a deep analysis of the ARMAP protocol for different values of network and protocol parameters. In particular, simulations proved the effectiveness of the decentralized mechanism based on agents' pheromone. The controlled dissemination of information assured by ARMAP guarantees a high availability of resources and favors an efficient management of the information system, since different clusters of peers can become specialized in different resource classes. Furthermore, with the use of ARMAP it is possible to devise a semi-informed discovery protocol that maximizes the success of a query request by routing it towards a cluster of peers which are specialized in the resource class specified in the query.

Acknowledgements

This work has been partially supported by Italian MIUR project "OpenKnowTech: Laboratorio di Tecnologie per la Integrazione" and by the project KMS-Plus funded by the Italian Ministry of Productive Activities.

[1] And. Andrzejak, A., Xu, Z.: Scalable, efficient range queries for grid information services, Proceedings of the Second IEEE International Conference on Peer-to-Peer Computing, Linkping University, Sweden (2002).

[2] Bab. Babaoglu, O., Meling, H., Montresor, A.: Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems, Proc. of the 22th International Conference on Distributed Computing Systems, ICDCS '02, Vienna, Austria (2002).

[3] Bon. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, Santa Fe Institute Studies in the Sciences of Complexity (1999).

[4] Che. Cheema, A.S. Muhammad, M. Gupta, I.: Peer-to-peer discovery of computational resources for Grid applications, Proc. of the 6th IEEE/ACM International Workshop on Grid Computing (2005).

[5] Coh. Cohen, E., Shenker, S.: Replication strategies in unstructured peer-to-peer networks, ACM SIGCOMM '02 Conference (2002).

[6] Cre. Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. In: 22 nd International Conference on Distributed Computing Systems, ICDCS '02, Vienna, Austria (2002), pp.23-33.

[7] Das. Dasgupta, P.: Intelligent Agent Enabled P2P Search Using Ant Algorithms, Proceedings of the 8th International Conference on Artificial Intelligence, Las Vegas, NV (2004), pp. 751-757.

[8] Den. Deneubourg, J. L., Goss, S., Franks, S., Sendova-Franks, A., Detrain, C., Chrétien, L.: The dynamics of collective sorting: robot-like ants and ant-like robots, Proceedings of the first international conference on simulation of adaptive behaviour, From animals to animats, Paris, France (1991), pp. 356-365.

[9] For. Forestiero, A., Mastroianni, C., Spezzano, G.: An Agent Based Semi-Informed Protocol for Resource Discovery in Grids, Proc. of International Conference on Computational Science, ICCS, Reading, United Kingdom (2006).

[10] Ia1. Iamnitchi, A., Foster, I.: Interest-Aware Information Dissemination in Small-World Communities, IEEE International Symposium on High Performance Distributed Computing, HPDC 2005, Research Triangle Park, NC (2005).

[11] Ia2. Iamnitchi, A., Foster, I., Weglarz, J., Nabrzyski, J., Schopf, J., Stroinski, M.: A Peer-to-Peer Approach to Resource Location in Grid Environments, eds. Grid Resource Management, Kluwer Publishing (2003).

[12] Lv. Lv, C., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer networks, ACM, Sigmetrics (2002).

[13] Mas. Mastroianni, C., Talia, D., Verta, O.: A Super-Peer Model for Resource Discovery Services in Large-Scale Grids, Future Generation Computer Systems, Elsevier Science, Vol. 21, No. 8 (2005), pp. 1235-1456.

[14] Pad. Padmanabhan, A. Shaowen Wang Ghosh, S. Briggs, R.: A self-organized grouping (SOG) method for efficient Grid resource discovery, Proc. of the 6th IEEE/ACM International Workshop on Grid Computing (2005).

[15] Pet. Petersen, K., Spreitzer, M., Terry, D., Theimer, M., Demers, A.: Flexible Update Propagation for Weakly Consistent Replication, Proc. of the 16th Symposium on Operating System Principles, ACM, (1997), pp. 288-301.

[16] Sha. Sharma, P., Estrin, D., Floyd, S., Jacobson, V.: Scalable timers for soft state protocols, Proc. of the IEEE Conference on Computer Communications INFOCOM, Kobe, Japan (1997).

[17] Swa. SwarmWiki environment, Center for the Study of Complex Systems, the University of Michigan, http://www.swarm.org/wiki.

[18] Syc. Sycara, K.: Multiagent systems, Artificial Inteligence Magazine, vol. 19, no. 2 (1998), pp. 79–92.

[19] Tal. Talia, D., Trunfio, P.: Towards a Synergy between P2P and Grids, IEEE Internet Computing 7(4) (2003), pp. 94-96.

[20] Ts1. Tsoumakos, D., Roussopoulos, N.: A Comparison of Peer-to-Peer Search Methods. Proc. of the Sixth International Workshop on the Web and Databases, WebDB, San Diego, CA (2003), pp.61-66.

[21] Ts2. Tsoumakos, D., Roussopoulos, N.: Adaptive probabilistic search for peer-to-peer networks. In: Third International Conference on Peer-to-Peer Computing, P2P '03, Linkoping, Sweden (2003), pp. 102-110.

[22] Van. Van Dyke Parunak, H., Brueckner, S. A., Matthews, R., Sauter, J.: Pheromone Learning for Self-Organizing Agents, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, vol. 35, no. 3 (2005).

[23] Web. Web Services Resource Framework, http://www.globus.org/wsrf/.

[24] Yan. Yang, B., Garcia-Molina, H.: Designing a Super-Peer Network, 19th Int'l Conf. on Data Engineering, IEEE Computer Society Press, Los Alamitos, CA, USA (2003).