Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Controlling Information Dissemination in Grids with an Epidemic Algorithm

A.Forestiero, C. Mastroianni, G. Spezzano

RT-ICAR-CS-06-03 **Settembre 2006**

Consiglio Nazionale delle Ricerche – Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) – Sedi di Cosenza (*Via P. Bucci 41C, 87036 Rende, Italy*), di Napoli (*Via P. Castellino 111, 80131 Napoli*) e Palermo (*Viale delle Scienze, Parco D'Orleans, 90128 Palermo*)– URL: *www.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Controlling Information Dissemination in Grids with an Epidemic Algorithm

A.Forestiero, C. Mastroianni, G. Spezzano

Rapporto Tecnico N.:
RT-ICAR-CS-06-03

Data:
Settembre 2006

I rapporti tecnici dell'ICAR-CNR sono pubblicati dall'Istituto di Calcolo e Reti ad Alte Prestazioni del Consiglio Nazionale delle Ricerche. Tali rapporti, approntati sotto l'esclusiva responsabilità scientifica degli autori, descrivono attività di ricerca del personale e dei collaboratori dell'ICAR, in alcuni casi in un formato preliminare prima della pubblicazione definitiva in altra sede.

Controlling Information Dissemination in Grids with an Epidemic Algorithm

Agostino Forestiero, Carlo Mastroianni, Giandomenico Spezzano

ICAR-CNR
Via P. Bucci cubo 41C, 87036 Rende (CS), Italy
e-mail: {forestiero,mastroianni,spezzano} @icar.cnr.it

Abstract

This paper examines a multi agent approach to build a Grid information system in which metadata related to Grid resources is disseminated and logically reorganized on Grid hosts. Agents, whose behavior is inspired by ant systems, replicate and distribute resource descriptors according to their features and construct accumulation regions. This facilitates resource discovery operations: another type of agents, i.e., discovery agents, are attracted towards Grid hosts which provide information about a large number of resources having the required characteristics. Both kinds of agents use pheromone-like mechanisms which guarantee self-organization and decentralization, since operations are performed only on the basis of local information. In particular, logical replication and organization of resources can be tuned, according to users' needs, by modulating the behavior of agents, which are able to replicate information in the first phase of their lives, but subsequently only move it. This switch is regulated by the value of a pheromone threshold, and an epidemic mechanism is used to communicate this parameter to hosts and agents of the Grid. Simulation analysis confirms the effectiveness of the proposed resource reorganization and resource discovery protocols and of the epidemic tuning mechanism.

1 Introduction

Grid systems are becoming more and more popular, but their effective usage is made problematic both by their ever increasing size and by the heterogeneity of hosts and resources of which they are composed. In this scenario, an urgent issue is the deployment of an information system featuring decentralized characteristics, opposed to the centralized or hierarchical information systems which are today provided by most Grid platforms, such as those based on the Web Services Resource Framework [7] .

This paper discusses a novel approach for the construction of a Grid information system which allows for an efficient management and discovery of resources. The approach, introduced in [5] and [6] in its basic version, exploits the features of (i) epidemic mechanisms tailored to the dissemination of information in distributed systems [11] and (ii) self adaptive systems in which "swarm intelligence" emerges from the behavior of a large number of agents which interact with the environment [1, 3]. This approach is partly inspired by biological systems, such as ant and termite colonies, which boast self-organizing, decentralized and scalable features that can profitably be exploited in distributed computer systems and specifically in Grids.

The proposed ARMAP protocol (*Ant-based Replication and MApping Protocol*) disseminates Grid resource descriptors (i.e., metadata documents) in a controlled way, by spatially sorting (or *mapping*) such descriptors according to their semantic classification, so achieving a logical reorganization of resources. A metadata document can be composed of a syntactical description of the service (i.e. a WSDL document) and/or an ontology description of service capabilities. For the sake of simplicity, in the following an information document describing a Grid resource will be simply referred to as a *resource*.

Each ARMAP agent travels the Grid through peer-to-peer (P2P) interconnections among Grid hosts, and uses simple probability functions to decide whether or not to *pick* resources from or *drop* resources into the current Grid host. Resource reorganization results from pick and drop operations performed by a large number of agents. A self-organization approach based on ants' pheromone [14] enables each agent to regulate its activity, i.e. its *operation mode*, only on the basis of local information. Indeed, each agent initially works in the *copy* mode (it can generate new resource replicas and disseminate them on the Grid). Subsequently, it switches to the *move* mode, i.e., it only moves resources from one host to another, without generating new replicas. This switch is performed when the level of a pheromone variable, which depends on agent's activity, exceeds a given threshold.

The ARMAP protocol can effectively be used to build a Grid information system in which (i) resources are properly replicated and (ii) the overall entropy is reduced. A balance between these two features can be achieved by regulating the pheromone threshold, i.e. by shortening or extending the time interval in which agents operate under the *copy* mode. Tuning of the pheromone mechanism can be *static* or *dynamic*. In the case of static tuning, the threshold is set before ARMAP protocol is started, whereas, in the case of dynamic tuning, the threshold can be tuned by a supervisor agent while ARMAP is running, depending on users' needs. This introduces a twofold control mechanism: each agent uses local information to self-regulate its activity, whereas a supervisor agent dynamically sets a global system parameter, i.e., the pheromone threshold, and propagates the value of this parameter via an epidemic mechanism. The supervisor initially communicates a new value of the pheromone threshold only to the peer in which such agent resides. Since this moment, each agent which visits this infected peer will be "infected" and its own pheromone threshold will be changed. In turn, whenever an infected agent visits a non-infected peer, the latter will be contaminated and will subsequently infect other agents. So, in a short time, all or most agents will be infected with the new value of the pheromone threshold.

A semi-informed discovery protocol (namely ARDIP, Ant-based Resource DIscovery Protocol) exploits the logical resource reorganization achieved by ARMAP. The rationale is the following: if a large number of resources of a specific class are accumulated in a restricted region of the Grid, it is convenient to drive search requests (issued by hosts to search for resources of that class) towards that region, in order to maximize the number of discovered resources and minimize the response time. An ARDIP discovery operation is performed in two phases. In the first phase a *blind* mechanism, specifically the random walks technique [9], is adopted: a number of query messages are issued by the requesting host and travel the Grid through the P2P interconnections among Grid hosts. In the second phase, whenever a query gets close enough to a Grid region which is collecting the needed class of resources, the search becomes *informed*: the query is driven towards this Grid region and will easily discover a large number of useful resources.

The semi-informed ARDIP protocol aims to combine the benefits of both blind and informed resource discovery approaches which are currently used in P2P networks [13]. In fact, a pure blind approach (e.g. using flooding or random walks techniques) is very simple and scalable but has limited performance and can cause an excessive network load, whereas a pure informed approach (e.g. based on routing indices [2] or adaptive probabilistic search [12]) generally requires a very structured resource organization which is impractical in a large, heterogeneous and dynamic Grid.

Simulation analysis shows that ARMAP and ARDIP protocols, if used together, allow for achieving a very high effectiveness in discovery operations.

The remainder of the paper is organized as follows. Section 2 describes the ARMAP protocol and Section 3 analyzes its performance, in particular by using the epidemic mechanism that enables dynamic tuning. Subsequently, Section 4 and 5 introduce and analyze the ARDIP protocol for resource discovery. Conclusions are given in Section 6.

2. The ARMAP Protocol

The aim of the ARMAP protocol [5] is to achieve a logical organization of Grid resources by spatially sorting them on the Grid according to their semantic classification. It is assumed that the resources have been previously classified into a number of classes *Nc*, according to their semantics and functionalities (see [10]).

The ARMAP protocol has been analyzed in a P2P Grid in which hosts are arranged in a 2-dimension toroidal space, and each host is connected to at most 8 neighbor peers. Such a mesh topology is chosen to obtain an intuitive and immediate graphical representation of system evolution, which helps to understand the protocol behavior. However, the Grid has a dynamic nature and hosts can, more or less frequently, disconnect and rejoin the network. Therefore at a specific time a peer is actually connected to a random number of active peers (at most 8), so relaxing the mesh assumption.

When connecting to the Grid, a host generates a number of agents given by a discrete Gamma stochastic function, with average *Ngen*, and sets the life time of these agents to *PlifeTime*, which is the value of the mean connection time of the peer, calculated on the basis of the peer's past activity. This mechanism allows for controlling the number of agents that operate on the Grid and assures a regular turnover of such agents. Indeed the number of agents is maintained to a value which is about *Ngen* times the number of peers.

2.1 Pick and Drop Functions

Each ARMAP agent offers its contribution to the reorganization of resources. Periodically an agent sets off from the current peer and performs a number of hops through the P2P links that interconnect the

Grid hosts. Then the agent uses appropriate *pick* and *drop* functions in order to replicate and move resources from one peer to another. More specifically, at each host an agent must decide whether or not to *pick* the resources of a given class, and then carry them in its successive movements, or to *drop* resources that it has previously picked from another host. Pick and drop probability functions depend on the number and the class of resources that are maintained in the local region of the Grid, as explained in the following.

Whenever an ARMAP agent hops to a Grid host, it must decide, for each resource class, whether or not to *pick* the resources of that class which are managed by the current host, unless it already carries resources of that class**.** In order to achieve replication and mapping functionalities, a pick random function is defined with the intention that the probability of picking the resources of a given class decreases as the local region of the Grid accumulates such resources and vice versa. This assures that as soon as the equilibrium condition is broken (i.e., resources belonging to different classes begin to be accumulated in different regions), the further reorganization of resources is more and more pushed.

The *Ppick* random function, defined in formula (1), is the product of two factors, which take into account, respectively, the relative accumulation of resources of a given class (with respect to the other classes), and their absolute accumulation (with respect to the initial number of resources of that class). In particular, the *fr* fraction is computed as the number of resources of the class of interest, accumulated in the peers located in the visibility region, divided by the overall number of resources that are accumulated in the same region.

$$(1)\quad P_{pick} = \left(\frac{k1}{k1 + fr}\right)^2 \cdot \left(\frac{(fa)^2}{k2 + (fa)^2}\right)^2$$

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 *Rv*. It is assumed that the visibility radius is equal to one, so that the visibility region is composed of at most 9 hosts (if all the neighbor peers are active), the current one included. The *fa* fraction is computed as the number of resources owned by the hosts located in the visibility region out of the overall number of resources that are maintained by such hosts, including the resources owned by other hosts and here deposited by the agents. The inverse of *fa* gives an estimation of the extent to which such hosts have accumulated resources of the class of interest. *k1* and *k2* are non-negative constants which are both set to 0.1. The *pick* operation can be performed with two different modes. If the *copy* mode is used, the agent, when executing a pick operation, leaves the resources on the current host, generates a replica of them, and carries such replicas until it will drop them in another host. Conversely, with the *move* mode, as an agent picks the resources, it removes them from the current host (except those owned by this host), thus preventing an excessive proliferation of replicas.

As well as the pick function, the *drop* function is first used to break the initial equilibrium and then to strengthen the mapping of resources belonging to different classes in different Grid regions. Whenever an agent gets to a new Grid host, it must decide, if it is carrying some resources of a given class, whether or not to drop such resources in the current host.

As opposed to the pick operation, the drop probability function *Pdrop*, shown in formula (2), is proportional to the relative accumulation of resources of the class of interest in the visibility region. In (2) the threshold constant *k3* is set to 0.3.

$$(2)\quad P_{drop} = \left(\frac{fr}{k3 + fr}\right)^2$$

2.2 System Entropy and Pheromone Mechanism

A spatial entropy function, based on the well known Shannon's formula for the calculation of information content, is defined to evaluate the effectiveness of the ARMAP protocol.

For each peer *p*, the local entropy *Ep* gives an estimation of the extent to which the resources have already been mapped within the visibility region centered in *p*. *Ep* has been normalized, so that its value is comprised between 0 and 1. In particular, an entropy equal to 1 corresponds to the presence of comparable numbers of resources belonging to all the different classes, whereas a low entropy value is obtained when the region centered in *p* has accumulated a large number of resources belonging to one specific class, thus contributing to the spatial ordering of resources. As shown in formula (3), the overall entropy *E* is defined as the average of the entropy values *Ep* computed at all the Grid hosts. In (3), *fr(i)* is the fraction of resources of class *Ci* that are located in the visibility region with respect to the overall number of resources located in the same region.

$$(3)\quad E_p = \frac{\sum_{i=1..Nc} fr(i) \cdot \log_2 \frac{1}{fr(i)}}{\log_2 Nc}, \quad E = \frac{\sum_{p \varepsilon Grid} Ep}{Np}$$

In [4] it was shown that the overall spatial entropy can be minimized if each agent exploits both the ARMAP modes, i.e. *copy* and *move*. In the first phase, the agent *copies* the resources that it picks from a Grid host, but when it realizes from its own activeness that the mapping process is at an advanced stage, it begins simply to *move* resources from one host to another, without creating new replicas. In fact, the copy mode cannot be maintained for a long time, since eventually every host would have a very large number of resources of all classes, thus weakening the efficacy of resource mapping. The protocol is effective only if agents, after replicating a number of resources, switch from *copy* to *move*.

A self-organization approach based on ants' pheromone mechanism enables each agent to perform this mode switch only on the basis of local information. This approach is inspired by the observation that agents perform more operations when the system entropy is high, but operation frequency gradually decreases as resources are properly reorganized. Therefore each agent increases its pheromone level when its activeness tends to decrease, and switches to the move mode as soon as the pheromone level exceeds a defined threshold *Tf*. In particular, at given time intervals, i.e. every 2,000 seconds, each agent counts up the number of times that it has evaluated the pick and drop probability functions, and the number of times that it has actually performed pick and drop operations. At the end of each time interval, the agent makes a deposit into its pheromone base, by adding a pheromone amount equal to the ratio between the number of "unsuccessful" operations and the total number of operation attempts.

An evaporation mechanism is used to give a higher weigh to recent behavior of the agent. Specifically, at the end of the i-th time interval, the pheromone level Φi is computed with formula (4).

$$(4)\quad \Phi i = E_{ARM} \cdot \Phi i-1 + \varphi i$$

The evaporation rate E_{ARM} 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 *Tf*, the agent realizes that the frequency of pick and drop operations has remarkably reduced, so it switches its protocol mode from copy to move. The choice of updating the pheromone level every time interval, instead of every single agent's operation, has been made to fuse multiple observations into a single variable, so giving a higher statistical relevance to the decisions of the agent. The value of *Tf* can be used to tune the number of agents that work in copy mode and are therefore able to create new resource replicas. Tuning can be *static* or *dynamic*, as discussed, respectively, in Section 3.2 and 3.3. Beforehand. parameters and performance indices are introduced in Section 3.1.

3. Performance of the ARMAP Protocol

3.1 Parameters and Performance Indices

Table 1 reports the main simulation parameters used in our analysis. The number of peers *Np*, or Grid size, is set to 2500 (corresponding to a 50x50 toroidal grid of peers). The average connection time of a specific peer, *PlifeTime*, is generated according to a Gamma distribution function, with an average value set to 100,000 seconds. The use of the Gamma function assures that the Grid contains very dynamic hosts, that frequently disconnect and rejoin the network, as well as much more stable hosts. Every time a peer disconnects from the Grid, it loses all resource descriptors previously deposited by agents, thus contributing to the removal of obsolete information. The number of Grid resources owned and published by a single peer is determined with a Gamma stochastic function that has an average value equal to 15 (see [8]). Grid resources are assumed to be classified in a number of classes *Nc*, which is set to 5.

The mean number of agents that travel the Grid is set to *Np*/2: this is accomplished, as explained in Section 2, by setting the mean number of agents generated by a peer, *Ngen*, to 0.5. The average time *Tmov* between two successive agent movements (i.e. between two successive evaluations of *pick* and *drop* functions) is set to 60 s. The maximum number of P2P hops that are performed within a single agent movement, *Hmax*, is set to 3. The visibility radius *Rv*, defined in Section 2.1 and used for the evaluation of pick and drop functions, is set to 1. Finally, the pheromone threshold *Tf*, defined in Section 2.2, ranges from 3 to 10.

A set of performance indices are defined for the performance evaluation of ARMAP. The overall entropy *E*, defined in Section 2.2, is used to estimate the effectiveness of the ARMAP protocol in the reorganization of resources. The *Nrpp* index is defined as the mean number of replicas that are generated for each resource. The processing load *L* is defined as the number of agents per second that are received and processed by a single peer.

Table 1. ARMAP parameters

Parameter	Value
Grid size (number of peer), *Np*	2500
Mean peer connection time, *PlifeTime*	100,000 s
Mean number of resources published by a peer	15
Number of classes of resources, *Nc*	5
Number of agents, *Na*	*Np*/2
Mean time interval between two successive movements of an agent, *Tmov*	60 s
Maximum number of hops, *Hmax*	3
Visibility radius, *Rv*	1
Pheromone threshold, *Tf*	3 to 10

3.2 Static Tuning

A first set of simulation runs have been performed to evaluate the performance of the ARMAP protocol and investigate the effect of static tuning. Static tuning is obtained by setting the pheromone threshold *before* the ARMAP protocol is initiated. When ARMAP is initiated, all agents (about 1250, half the number of peers) are generated in the *copy* mode, but subsequently several agents switch to *move*, as soon as their pheromone value exceeds the threshold *Tf*. If the pheromone threshold *Tf* is increased, the average interval of time in which agents work in *copy* mode becomes longer. As a consequence, the average number of agents that work in *copy* (also called *copy agents* in the following) is larger, so such agents are able to create more resource replicas. Hence, a proper setting of the pheromone threshold is a very efficient method to enforce or reduce the generation of new replicas and the velocity and intensity of resource dissemination. However, a more intense dissemination is not always associated to a better resource reorganization, i.e. to a more effective spatial separation of resources belonging to different classes, and therefore to a lower overall entropy.

Figure 1 shows that lower values of the overall entropy are achieved with lower values of the pheromone threshold. For example, with *Tf*=3, the value of the overall entropy decreases from the initial value of about 1 (maximum disorder) to less than 0.72. As an extreme case, virtually no entropy decrease is observed if all the agents operate in *copy* (*Tf*=10), which confirms that the mode switch is strictly necessary to perform an effective resource reorganization.

Figure 2 shows the mean number of replicas generated per resource, and confirms that resource dissemination is more intense if the pheromone threshold is increased, because a larger number of *copy* agents operate on the network. It can be concluded that *copy* agents are useful to replicate and disseminate resources, but it is the *move* agents that actually perform resource reorganization and are able to create Grid regions specialized in specific classes of resources.

As opposed to the indices described so far, the processing load *L*, i.e., the average number of agents per second that are processed by a peer, does not depend on the pheromone threshold, but on the number of agents and on the frequency of their movements across the Grid. *L* is obtained as shown in formula (5):

$$(5)\quad L = \frac{N_a}{T_{mov} \cdot N_p} = \frac{P_{gen}}{T_{mov}}$$

In the described scenario, each peer receives and processes about one agent every 120 seconds, which can be considered an acceptable load. Even though analysis is focused on a Grid with 2500 hosts, the ARMAP protocol is intrinsically scalable due to its decentralized nature, since each agent performs pick and drop operations, and tunes its operating mode, only according to local information collected by the hosts that it visits.

3.3 Dynamic Tuning and Epidemic Mechanism

Whereas the previous subsection analyzes static tuning, since the value of *Tf* is set before ARMAP is initiated, this section introduces dynamic tuning, the purpose of which is to regulate *Tf* while ARMAP is running. The value of *Tf* should be increased if more replicas are needed, while it should be reduced if a better spatial mapping of resources is desired.

Dynamic tuning can be achieved by a few *supervisor agents* that, according to the needs and the level of satisfaction of users, communicate to ARMAP agents a new value of the pheromone threshold, so to enforce or reduce the activity of agents. In this paper, an *epidemic mechanism* is proposed to transmit information to all agents. The epidemic approach is an effective solution for disseminating information in peer to peer systems [4]. This type of mechanism mimics the spread of a contagious disease in which infected entities contaminate other "healthy" entities.

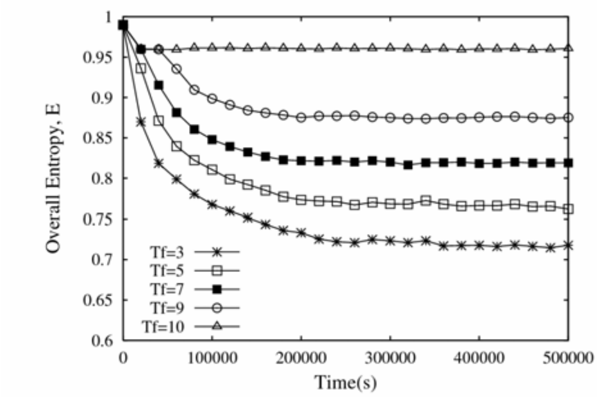

Fig. 1. Static tuning. Overall system entropy, for different values of the pheromone threshold *Tf*.

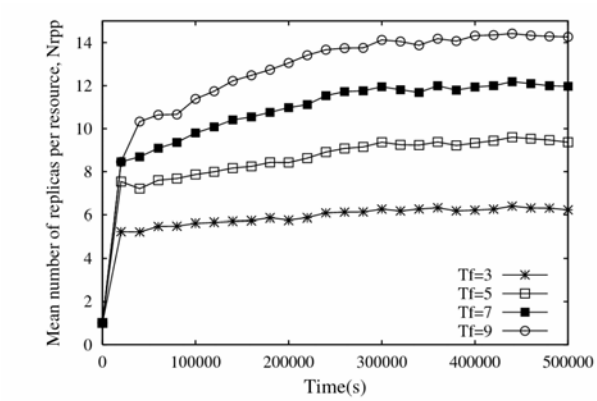

Fig. 2. Static tuning. Mean number of replicas per resource, for different values of the pheromone threshold *Tf*.

ARMAP dynamic tuning works as follows. When a supervisor agent decides to regulate resource replication, on the basis of users' needs, it initially communicates a new value of the pheromone threshold only to the peer in which such agent resides: infection will then spread from this peer. Since this moment, each agent which visits this infected peer is contaminated and its own pheromone threshold is changed. In turn, whenever an infected agent visits a non-infected peer, the latter is contaminated and will subsequently infect other agents. So, in a short time, all or most agents are "*infected*" with the new value of the pheromone threshold.

Figure 3 shows the trend of the number of replicas per resource *Nrpp* in the case of dynamic tuning. In this figure, dotted curves are related to the values of *Nrpp* obtained, under static tuning, with *Tf*=5 and *Tf*=9. Continuous curves report the performance achieved with dynamic tuning. The initial threshold *Tf* is set to 5, but at time=400,000 seconds, *Tf* is changed to 9 by a supervisor agent located in one of the Grid host. The continuous line labelled with squares corresponds to an ideal scenario in which there exists a *global control* mechanism which is able to *immediately* communicate the new pheromone threshold to all agents. On the other hand, the continuous line labelled with circles is achieved by exploiting the above described epidemic mechanism, which is initiated by the mentioned supervisor agent. It is noted that in both cases the trend of *Nrpp*, after a transition phase in which agents are induced to increase their activeness, converges to the curve obtained with static tuning and *Tf*=9, so confirming the effectiveness of dynamic tuning. The transition phase experienced with epidemic control is slower than that measured with global control due to the time necessary to propagate information to a significant number of agents. Anyhow, the trend obtained with epidemic mechanism is very close to the trend achieved with the ideal global control.

Analogous observations can be made about the overall entropy. Indeed, Figure 4 depicts the values of entropy obtained under static tuning and dynamic tuning, and compares epidemic and global control. The overall entropy increases from a value of about 0.75 with *Tf*=5, to about 0.88 with *Tf*=9, both with global control and epidemic control. As observed for the number of replicas, the trend of the overall entropy obtained with global control approaches a steady value more quickly than with epidemic control, due to the time necessary, with the epidemic mechanism, to inform ARMAP agents about the change in the pheromone threshold. However, the additional delay experienced when exploiting the epidemic mechanism is tolerable.

Overall, the epidemic mechanism is effective and requires no extra message load, since information is carried at no cost by ARMAP agents which travel the Grid. Conversely, global control requires an onerous and well synchronized mechanism to quickly pass information to all agents.

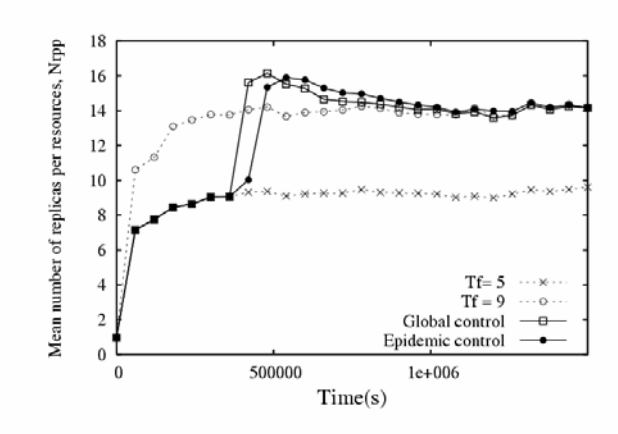

Fig. 3. Dynamic tuning. Mean number of replicas per resource *Nrpp*, when the pheromone threshold is changed from *Tf*=5 to *Tf*=9. Comparison of global (ideal) and epidemic control.

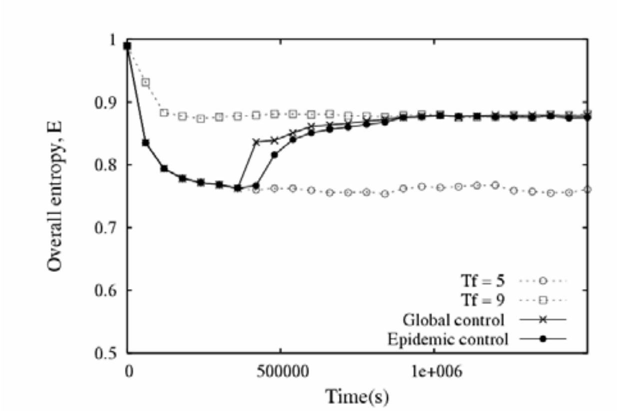

Fig. 4. Dynamic tuning. Overall system entropy *E*, when the pheromone threshold is changed from *Tf*=5 to *Tf*=9. Comparison of global (ideal) and epidemic control.

4 The ARDIP protocol

The ARDIP (*Ant-based Resource DIscovery Protocol*) protocol is used by clients to discover Grid resources belonging to a given class. The objective is to drive an ARDIP agent, i.e. a query message, towards a region of the Grid in which the needed class of resources is being accumulating. Because ARDIP fully exploits the replication and spatial sorting of resources achieved by ARMAP, the two protocols should be used together: as ARMAP agents perform the logical reorganization of resources and build the Grid information system, it is more and more likely that ARDIP agents can find a remarkable number of useful resources in a small amount of time.

The ARDIP protocol is based upon three modules: (a) a module for the identification of representative peers which work as attractors for query messages; (b) a module which defines the semi-informed search algorithm; (c) a stigmergy mechanism that allow query messages to take advantage of the positive outcome of previous search requests. These modules are described in the following.

Identification of representative peers.
As a class of resources is accumulated in a Grid region, the peer that, within this region, collects the maximum number of resources belonging to that class is elected as a *representative peer*. The objective of a search operation is to let a query message get to a representative peer, since such a peer, as well as its neighbors, certainly manages a large number of useful resources. The ARDIP protocol assumes that a peer *p* is a representative peer of class *Ci* if at least one of the two following conditions are verified: (a) the peer *p* maintains a number of resource descriptors of class *Ci* that exceeds *f1* times the mean number of actual resources of the same class which are offered by a generic peer; (b) the peer *p* maintains a number of resource descriptors of class *Ci* that exceeds *f2* times (with *f2*<*f1*) the number of resource descriptors of the same class maintained by its adjacent neighbors.

To limit the number of representative peers in the same region, each representative peer periodically checks if other representative peers are present in its neighborhood, within the *comparison radius Rc*: two neighbor representative peers must compare the number of resources they maintain, and the “loser” will be downgraded to a simple peer.

Semi-informed search.
When a user needs to discover resources belonging to a given class, it initiates a *search request* procedure, i.e., it issues a number of *query messages*. The semi-informed search algorithm includes a *blind* search phase and an *informed* search phase. The random walks paradigm is used during the blind search phase: the query messages travel the Grid through P2P interconnections by following a random path. The network load is limited with the use of a TTL parameter, which is equivalent to the maximum number of hops that can be performed by a query message before being discarded.

A *blind* search procedure is switched to an *informed* one as soon as one of the issued query messages approaches a representative peer, i.e. when such a message is delivered to a peer which knows the existence of a representative peer and knows a route to it (see the description of the stigmergy module below).

During the informed search phase, a query message is driven towards the representative peer, and the TTL parameter is ignored so that the query cannot be discarded until it actually reaches the representative peer. Therefore, the semi-informed walk of a query message ends in one of two cases: (i) when the TTL is decremented to 0 during the blind phase; (ii) when the query reaches a representative peer. In both cases a queryHit message is created, and all the resource descriptors of class of interest, which are found in the current peer, are put in this message. The queryHit follows the same path back to the requesting peer and, along the way, collects all the useful resource descriptors that are managed by the peers through which it passes.

Stigmergy mechanism.
The stigmergy mechanism is a mechanism, often observed in biological systems, through which elementary entities exploit the environment to communicate with each other. For example, in ant colonies, an ant that finds a food source leaves a *pheromone* along its way back to the nest, and such a pheromone will alert other ants to the presence of the food source. The ARDIP protocol exploits a similar mechanism: when a query message accidentally gets to a representative peer for the first time, the returning queryHit will deposit an amount of pheromone in the peers that it encounters as it retreats from the representative peer. In this paper, it is assumed the pheromone is deposited only in the first two peers of the queryHit path.

When a query message gets to a peer during its blind search, it checks the amount of pheromone which has been deposited there; if the pheromone exceeds a threshold T_{ARD}, it means that a representative peer is close and search becomes informed. An evaporation mechanism assures that the pheromone deposited on a peer does not drive queryHits to ex-representative peers. The pheromone level at each peer is computed every time interval of 5 minutes. The amount of pheromone Φi, computed after the i-th time interval, is given by formula (6).

$$(6)\quad \Phi_i = E_{ARD} \cdot \Phi_{i-1} + \varphi_i$$

The evaporation rate E_{ARD} is set to 0.9; φi is equal to 1 if a pheromone deposit has been made in the last time interval by at least one agent, otherwise it is equal to 0. The threshold T_{ARD} is set to 2.

5 Performance of the ARDIP protocol

This section evaluates the performance of the ARDIP protocol and shows that resource discovery operations are more and more effective as resources are reorganized by ARMAP agents. Table 2 reports the values assumed by main ARDIP parameters, which have been defined in Section 4, whereas performance indices are defined and explained in Table 3.

The index *Fsq* is essential to evaluate how many search requests are able to reach a representative peer. Figure 5 proves the valuable effect caused by the combined use of ARMAP and ARDIP protocols. In fact, after a very small amount of time, the logical reorganization of resources produces a significant increase in *Fsq*. Moreover, *Fsq* increases with the TTL value, since a search request extends the blind phase and has more chances to get to a representative peer.

The most important performance measure is *Nres*, the mean number of results that are discovered after a search request. Indeed, it is generally argued that the satisfaction of the request depends on the number of discovered resources returned to the user [15]. The trend of *Nres* is depicted in Figure 6, which shows that the mean number of results is larger and larger as resources are being organized by ARMAP. Furthermore, even when the probability of reaching a representative peer begins to become stable (Figure 5), the number of results continues to increase, meaning that representative peers are able to collect more and more resources of the class in which they are specialized. It is also noted that the requests which successfully get to a representative peer can discover considerably more results than the rest of the requests, and such a difference increases with the value of the TTL parameter.

The semi-informed discovery protocol not only increases the number of results, but also allows users to discover them in a shorter amount of time. Figure 7 shows that the response time decreases as the ARMAP work proceeds, and also that it is notably smaller if the search request reaches a representative peer; in this case, in fact, the discovery operation is stopped even if the TTL value is still greater than 0, and a queryHit is immediately issued (see Section 4). However, this performance improvement is achieved only with TTL equal to 5, whereas it is not evident for smaller values of TTL, i.e., with TTL equal to 3.

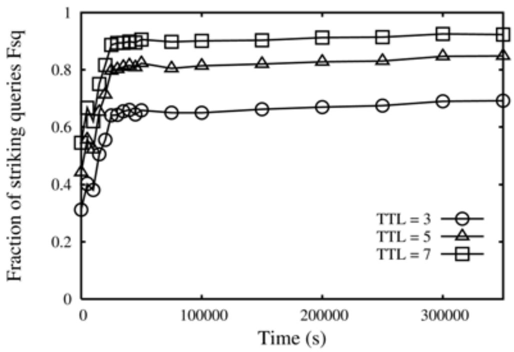

Fig. 5. Fraction of search requests that are successfully driven to a representative peer, for different values of TTL.

Table 2. ARDIP parameters

Parameter	Value
Number of query messages issued by the requesting peer	4
Time to live, *TTL*	3-7
Factor *f1*, for the identification of representative peers	5
Factor *f2*, for the identification of representative peers	2.5
Comparison radius, *Rc*	2
Mean frequency at which a Grid peer issues query messages	1/300 (1/s)
Mean message elaboration time at a Grid peer	100 ms

Table 3. ARDIP performance indices

Performance Index	Definition
Fraction of striking requests, *Fsq*	Fraction of search requests that are successfully driven towards a representative peer
Mean number of results, *Nres*, *Nres(rep)*, *Nres(norep)*	Mean number of resources that a node discovers after its search request (computed for all the requests, for the requests that are actually delivered to a representative peer, for the requests that are not delivered to a representative peer)
Response times, *Tr*, *Tr(rep)*, *Tr(norep)*	Mean amount of time (s) that elapses between the generation of a request and the reception of a corresponding queryHit (computed for all the requests, for the requests that are actually delivered to a representative peer, for the requests that are not delivered to a representative peer)

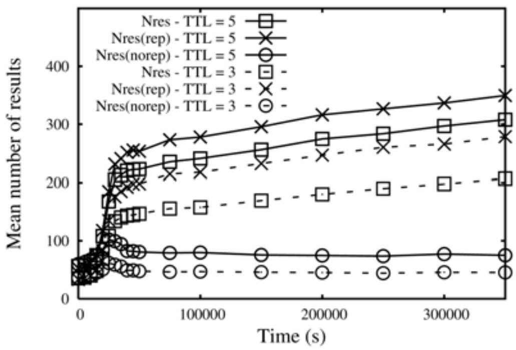

Fig. 6. Average number of results, with different values of TTL, reported for requests that reach a representative peers, for requests that do not reach a representative peers, and for all the requests.

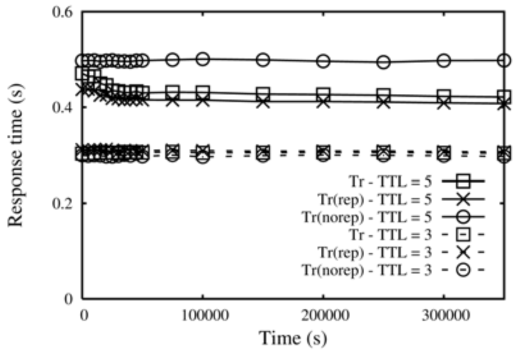

Fig. 7. Response time, with different values of TTL, reported for requests that reach a representative peers, for requests that do not reach a representative peers, and for all the requests.

6 Conclusions

This paper introduces an approach based on the multi agent paradigm, and inspired by biological systems such as ant and termite colonies, for building an efficient information system in Grids. The approach is exploitable in very large networks, since it is fully decentralized and self-organizing. Two complex objectives, specifically reorganization and discovery of resources, are achieved through simple operations performed by two different kinds of agents. Some agents are in charge of replicating and moving resource descriptors so to create accumulation regions specialized in different classes of resources. Other agents guide query messages toward the core of such regions, which allows to discover a large number of resources belonging to a given class.

Along with the performance analysis of the protocols which drive agents' operations, this paper evaluates an epidemic approach used to tune the activeness of agents. In fact replication and reorganization of resources can be modulated by enlarging or shortening the first phase of the life of reorganizing agents, in which agents are allowed to create new resource replicas. This is achieved by tuning the value of a *pheromone* parameter. A supervisor agent, on the basis of users' requirements, is enabled to change the pheromone value and an epidemic mechanism is used to spread infection, i.e. let information pass from peers to agents and from agents to peer. Simulation shows that the proposed epidemic mechanism is equally effective and nearly efficient as an ideal mechanism through which all agents are immediately informed regarding the updated pheromone value.

References

[1] 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)

[2] Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. Proc. of the 2th International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria (2002), pp. 23-33

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

[4] Eugster, P., Guerraoui, R., Kermarres, A.M., Massoulieacute, L.: Epidemic Information Dissemination in Distributed System, Computer, IEEE Computer Society, vol. 37, No. 5 (2004), pp. 60-67.

[5] Forestiero, A., Mastroianni, C., Spezzano, G.: A Multi Agent Approach for the Construction of a Peer-to-Peer Information System in Grids, Proc. of the 2005 International Conference on Self-Organization and Adaptation of Multi-agent and Grid Systems SOAS'05, Glasgow, Scotland (2005)

[6] 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. LNCS vol. 3994 (2006), pp. 1047–1054

[7] Globus Alliance and IBM: Web Services Resource Framework, http://www.globus/org/wsrf/

[8] 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).

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

[10] 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

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

[12] 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

[13] 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

[14] 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)

[15] Yang, B., Garcia-Molina, H.: Efficient search in peer-to-peer networks, Proc. of ICDCS, Wien, Austria (2002).