

QoS-Based Dissemination of Content in Grids

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 proposes a bio-inspired approach tailored to the construction of a Grid information system in which content, specifically metadata descriptors that specify the characteristics of Grid resources, is disseminated and logically reorganized on the Grid. Content management is achieved through the collective activity of a large number of ant-like agents, which perform simple operations at the local level, but together engender an advanced form of “swarm intelligence” at the global level. Agents travel the Grid through P2P interconnections and use probability functions to: (i) replicate resource descriptors and (ii) collect similar descriptors in nearby Grid hosts. Simulation analysis shows that the proposed ARMAP protocol (Ant-based Replication and MApping Protocol) is capable of reducing the entropy of the system and efficiently propagating content. Furthermore, ARMAP permits to enforce the dissemination of descriptors related to high Quality of Service (QoS) resources. The Grid information system built with the proposed protocol enables the use of a semi-informed discovery algorithm which drives query messages towards “representative peers” that maintain information about high QoS resources having the required characteristics.

1. Introduction

Content Networks have been introduced in [15] to generalize the functionalities and objectives of Content Distribution Networks (CDN). While CDNs are specifically designed for the efficient delivery of content to Web users, essentially through replication and caching of content at locations as close to users as possible, CNs are more generically designed for the management of metadata content. Descriptors of Web resources but also metadata information about format and location of resources, provide the means for efficient resource placement and retrieval.

These issues have much in common with the topic of resource management in Grids. In particular, the information system of a Grid [16] supports discovery and handling of Grid resources through an appropriate management of metadata. Grid resources can be both hardware facilities and software tools that are needed by users to perform their computations. In current Grids, such as the Globus Toolkit [8], resources are usually provided to clients in the form of Grid-aware Web services.

Before accessing a Grid resource, a user should know what resources/services are available and where they are located. Indeed the query of users can be satisfied by delivering them metadata documents which describe potentially useful resources. In the following, a metadata document is also referred to as *descriptor*, which can be composed of a syntactical description of a Grid service (e.g. a Web Services Description Language (WSDL) document) and/or a semantic description of the capabilities of the service.

As metadata management is a widely considered issue in Grids, Grid techniques for the dissemination and management of metadata information can be successfully applied in Content Networks. Indeed, a Grid can provide components and services that efficiently support the functionalities of Content Networks, specifically content replication, caching and delivery. It may even be argued that certain types of Grids, i.e., those dedicated to the distributed management of data and content (as opposed to Grids specifically devoted to massive and parallel computation) *are* actually Content Networks, since they are used to manage content and deliver it to users.

This paper discusses a novel approach for the construction of a Grid information system in which resource descriptors are replicated, disseminated and reorganized in order to improve their management and increase the performance of discovery operations.

This approach, proposed in [6] in its basic version, exploits the features of (i) epidemic mechanisms

tailored to the dissemination of information in distributed computer systems [14] and of (ii) self-organizing biological systems which feature “swarm intelligence” characteristics.

The proposed ARMAP protocol (*Ant-based Replication and Mapping Protocol*) is specifically inspired by the behavior of ants and termites which cluster and map corpses within their environment [5]. A number of ARMAP agents travel the Grid through P2P interconnections, disseminate resource descriptors, and map descriptors according to the semantic classification of resources: they accumulate descriptors of resources belonging to different classes in different regions of the Grid. Each host can maintain descriptors of local resources, which are never removed, as well as descriptors of resources published by other hosts, which can be picked up and discarded by agents. In the following, when distinction is relevant, such descriptors will respectively be referred to as *local* and *remote* descriptors.

Each ARMAP agent uses simple probability functions to decide whether or not to *pick* descriptors from or *drop* descriptors into the current Grid host. Reorganization of descriptors emerges, in a swarm intelligence fashion, from pick and drop operations performed by a large number of agents. The ARMAP protocol is used to build a Grid information system in which (i) descriptors are replicated and disseminated and (ii) the overall spatial entropy is reduced. A balance between these two functionalities is achieved by regulating the activity of agents through a *pheromone* mechanism, which relies exclusively on local information, so as to guarantee the scalability of this approach.

This approach inherits interesting and highly beneficial properties from its biological counterpart (ant colonies), namely: (i) self-organization, since decisions are based on local information, i.e., without any central coordinator; (ii) adaptivity, since agents can flexibly react to the ever-changing environment and (iii) stigmergy awareness [9], since agents are able to interact and cooperate through the modifications of the environment that are induced by their operations. Such properties are able to guarantee scalability, while currently used algorithms for building Grid information systems are centralized and hierarchical, hence not well scalable [18].

The ARMAP protocol is specifically tailored to the dissemination of descriptors of resources having high Quality of Service (QoS) features. In fact, after issuing their queries, users generally discover information concerning a number of resources, and then they can choose the resources which are the most appropriate for their purposes. Selection of resources is based on the evaluation of their QoS. The QoS of a Grid

resource is given by a number of features and characteristics that are able to satisfy the user needs, among which: scalability, capacity, performance, reliability, availability, robustness, accuracy, security [17]. However, if a query returns information about a lot of candidate resources, the selection process can become difficult and time-consuming. The ARMAP protocol facilitates this process by fostering the replication and dissemination of descriptors related to high QoS resources.

A semi-informed discovery protocol can exploit the logical resource organization achieved by ARMAP. Indeed, whenever a large number of descriptors of a specific class are accumulated in a restricted region of the Grid, it becomes convenient to drive query messages (issued by peers to search for descriptors of that class) towards that region, in order to maximize the number of discovered descriptors and minimize the response time. While this paper focuses on enhancements and performance of the ARMAP protocol, the discovery protocol, namely ARDIP (*Ant-Based Resource Discovery Protocol*) is extensively discussed in [7].

The remainder of the paper is organized as follows. Section 2 introduces the ARMAP protocol, in particular the basic pick and drop probability functions, the pheromone mechanism that steers agents’ activity, and the enhanced (or *QoS-aware*) pick and drop functions defined to cope with the QoS of resources. Section 3 analyzes the performance of the ARMAP protocol, evaluated with an event-driven simulator, and briefly describes the semi-informed discovery protocol that exploits the work of ARMAP. Section 4 discussed related work and Section 5 concludes the paper.

2. The ARMAP protocol

The aim of the ARMAP protocol [6] is the construction of a Grid information system in which resource descriptors are replicated and spatially mapped on the Grid according to semantic classification of resources. It is assumed that the resources have been previously classified into a number of classes N_c , according to their semantics and functionalities (see [3]).

The ARMAP protocol has been analyzed in a Grid in which each host is connected to a number of neighbor hosts through P2P interconnections. The Grid has a dynamic nature, and hosts can, more or less frequently, disconnect and rejoin the network. When connecting to the Grid, a host generates a number of agents given by a discrete Gamma stochastic function, with average N_{gen} , 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 past activity. This mechanism allows for the control of the number of agents that operate on the Grid and assures a regular turnover of such agents. Indeed the overall number of agents is maintained to a value which is about N_{gen} times the number of active peers.

Each ARMAP agent offers its contribution to the reorganization of descriptors. 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 descriptors from one peer to another. More specifically, at each host an agent must decide whether or not to *pick* the descriptors of a given class, and then carry them in its successive movements, or to *drop* descriptors that it has previously picked from another host. In their basic definition, pick and drop probability functions depend on the number and class of descriptors that are maintained in the local region of the Grid, as explained in the following.

2.1 Basic Pick and Drop Functions

The basic pick and drop probability functions are inspired by the pick and drop functions introduced in [5], and later elaborated and discussed in [2] and [13], which allow to emulate the behavior of some species of ants that build cemeteries by aggregating corpses in clusters. This approach is adapted to our purposes, by the following main modifications: (i) descriptors are not only aggregated, as in the mentioned papers, but also *replicated*, in order to achieve information dissemination; (ii) descriptors are spatially mapped, i.e., they are accumulated in separate clusters according to the class to which they belong.

Pick operation. Whenever an ARMAP agent hops to a Grid host, it must decide, for each resource class, whether or not to *pick* the descriptors of that class which are managed by the current host, unless it already carries descriptors 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 descriptors of a given class decreases as the local region of the Grid accumulates such descriptors and vice versa. This assures that as soon as the equilibrium condition is broken (i.e., descriptors belonging to different classes begin to be accumulated in different regions), a further reorganization of descriptors is favored.

The basic *Ppick* probability function is shown in formula (1). The agent evaluates this function, generates a real number between 0 and 1, from a uniform random distribution, and then it performs the pick operation if the generated number is lower than

the value of the *Ppick* function. This function is the product of two factors, which take into account, respectively, the *absolute* accumulation of descriptors of a given class and their *relative* accumulation (i.e., with respect to other classes). While the absolute factor is inspired by the pick probability defined in [2], the relative factor was introduced here to achieve the spatial separation of descriptors.

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

The *fa* fraction is computed as the number of *local* descriptors of the class of interest, maintained by the hosts located in the *visibility region*, out of the overall number of descriptors of the class of interest (including both *local* and *remote*) that are maintained by the same hosts. The value of *fa* is comprised between 0 and 1, as well as the value of *fr*, defined below. The *visibility region* includes all the hosts that are reachable from the current host with a given number of hops, i.e. within the *visibility radius*, which is an algorithm parameter. As more *remote* descriptors of a class are accumulated in the local region, *fa* decreases, the first factor of the pick function decreases as well, and the probability that the agent picks the descriptors becomes lower. This facilitates the formation of accumulation regions. Conversely, the pick probability is large if a small number of descriptors are located in the local region of the Grid.

The *fr* fraction is computed as the number of descriptors of the class of interest, accumulated in the hosts located in the visibility region, divided by the overall number of descriptors (of all classes) that are accumulated in the same region. As the local region accumulates more descriptors of a class, with respect to other classes, the value of the pick probability for this class becomes lower, and vice versa. This facilitates the spatial separation of descriptors of different classes.

The constants *k1*, *k2* are set to the value 0.1, as in analogous formulas defined in [2]. However, we found that the behavior of the ARMAP protocol does not qualitatively differ if the values of these constants vary within a given range, i.e., the range [0,05-0,5] for *k1*, and [0,05-0,9] for *k2*. In fact these values can have an impact on the velocity and duration of the transient phase of the replication process, but do not alter the global behavior of the protocol in steady conditions.

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 descriptors 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 descriptors, it removes them from the current host (except for the *local* descriptors, which cannot be removed), thus preventing an excessive proliferation of replicas.

Drop operation. As well as the pick function, the drop function is first used to break the initial equilibrium and then to strengthen the mapping of descriptors belonging to different classes in different Grid regions. Whenever an agent gets to a new Grid host, it must decide, if it is carrying descriptors of a given class, whether or not to *drop* such descriptors in the current host. As opposed to the pick operation, the drop probability function P_{drop} , shown in formula (2), is directly proportional to the relative accumulation of descriptors of the class of interest in the visibility region. The value of the constant $k3$ is set to 0.3, as in [2], but tests showed that any value comprised in the range [0,2-0,9] can be selected.

Note that the drop function does not contain a factor related to the absolute accumulation of descriptors, because it has been seen that in the long term this factor could start an avalanche mechanism, that occurs whenever a region accumulates too many descriptors belonging to two or more different classes. In such a case, drop operations would be further facilitated for all such classes (instead of only one), thus weakening the spatial reorganization of descriptors. This phenomenon does not occur with the mere use of a relative accumulation factor, as in formula (2).

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

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 E_p gives an estimation of the extent to which the descriptors have been mapped within the visibility region centered in p . E_p 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 descriptors 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 descriptors belonging to one specific class, thus contributing to the spatial mapping of descriptors.

As shown in formula (3), the overall entropy E is defined as the average of the entropy values E_p computed at all the Grid hosts. In (3), $fr(i)$ is the fraction of descriptors of class C_i that are located in

the visibility region with respect to the overall number of descriptors located in the same region.

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

In [6] 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, an agent *copies* the descriptors 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* descriptors 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 descriptors of all classes, thus weakening the efficacy of spatial mapping. The protocol is effective only if agents, after replicating a number of descriptors, switch from *copy* to *move*. A self-organization approach based on ants' pheromone 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 descriptors are properly reorganized. Each agent maintains an amount of pheromone and increases it when its activeness tends to decrease; then it switches to the *move* mode as soon as the pheromone level exceeds a defined threshold T_f .

In more details, at given time intervals, i.e. every 2,000 seconds, each agent counts up the number of times that it has evaluated the 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 1 minus the ratio between the number of performed operations and the overall 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 Φ_i is computed as in formula (4).

$$(4) \Phi_i = E \cdot \Phi_{i-1} + \varphi$$

The evaporation rate E_v is set to 0.9, and φ is the pheromone amount added in the last time interval. The value of the pheromone amount is always comprised between 0 and 10, and a pheromone threshold T_f is used to control the switch of agents from *copy* to *move*. Whenever the pheromone level exceeds T_f , 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 value of T_f , which must be set in the range [0-10], can be used to tune the number of agents that work in the *copy* mode and are therefore able to create new replicas. The effect of this tuning is discussed in Section 3.2.

2.3 QoS-Aware Pick and Drop Functions

The pick and drop functions discussed in Section 2.1 allow to replicate and reorganize descriptors regardless of the quality of service of the corresponding resources. A better achievement would be obtained if descriptors of resources characterized by high QoS characteristics were replicated and disseminated more effectively than other descriptors. This way a user that issues a query would find more useful resources and his/her satisfaction would increase [17].

For the sake of simplicity, in this paper it is assumed that the QoS of a resource can be expressed by a non-negative real value [23], and that admitted QoS values are comprised between 0 to 10, with higher values corresponding to better quality. This implies that for each resource class a method is defined to compute the QoS value of a resource belonging to this class; for example, the QoS of a Grid host may be calculated by combining different features such as CPU power, number of processors, main memory etc., whereas a data mining software may be evaluated on the basis of users' assessment.

In the following, an expression like "the QoS of a descriptor" will be used as an abbreviation for "the QoS of the resource described by a descriptor". To achieve a selective dissemination of high QoS descriptors, pick and drop functions have been enhanced through the definition of additional "QoS-aware" factors. More specifically, the pick probability function defined in formula (1) is multiplied by the factor QoS_{pick} defined in formula (5).

$$(5) \ QoS_{pick} = \frac{k4}{k4 + \frac{QoS_{resource} - QoS_{peer}}{QoS_{resource}}}$$

This factor is evaluated by an ARMAP agent each time it moves to a new host. In formula (5), $QoS_{resource}$ is the average QoS value of a generic descriptor belonging to the class of interest. In a Grid, each agent can estimate $QoS_{resource}$ by averaging the QoS values of the descriptors maintained by the peers that it visits. However, here it is assumed that this value is known and equal to 5. QoS_{peer} is the average QoS value of the descriptors that are maintained by the current host. The parameter $k4$ is set to a real value not lower than 2.

It can be observed that the average value of QoS_{pick} is equal to 1. This means that the average value of the QoS-aware pick function is not altered, and thus the overall mapping of descriptors (regardless of their QoS values) is not biased by the new factor. However, if the current peer maintains descriptors characterized by high quality of service, the value of QoS_{pick} is 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 if the peer maintains low QoS descriptors.

Analogously, the drop probability function defined in formula (2) is multiplied by a new factor QoS_{drop} , reported in formula (6).

$$(6) \ QoS_{drop} = \frac{k4}{k4 + \frac{QoS_{peer} - QoS_{agent}}{QoS_{agent}}}$$

In formula (6), QoS_{agent} is the average QoS value of the descriptors of the class of interest which are carried by an agent. It results that the drop operation is favored when an agent carries descriptors that have QoS values higher than the descriptors located in the current peer, thus fostering the dissemination of information pertaining to high QoS resources.

3. Analysis of the ARMAP Protocol

3.1 Simulation Environment

The performance of the ARMAP protocol with both basic and QoS-aware pick/drop functions has been evaluated with an event-based simulator written in Java. Simulation *objects* are used to emulate Grid peers and ARMAP agents. Each object reacts to external *events* according to a finite state automaton and responds by performing specific operations and/or by generating new messages/events to be delivered to other objects. For example, a peer visited by an agent gives it information about the descriptors that this peer maintains; afterwards the agent uses probability functions to decide whether or not to *pick* descriptors from or *drop* descriptors into the peer. Finally, the agent sets the simulation time in which it will perform its next movement on the Grid and creates a related event that will be delivered at the specified time to the peer to which the agent will move. Events are ordered in a general queue by a simulation *manager*

^(*) with the use of the new factor, in some cases the pick probability can assume values higher than 1; in these cases the probability is truncated to 1. It corresponds to a 100% probability of picking descriptors which have a very high QoS. Analogous considerations hold for the drop probability.

component, and they are delivered to corresponding destination objects according to their expiration time, so that peers and agents can operate concurrently along the simulation time. Moreover, the simulation environment exploits the visual facilities and libraries offered by the Swarm environment [19].

Table 1 reports the main simulation parameters used in our analysis. The number of peers Np , or Grid size, ranges from 900 to 4900, and each peer is connected to at most 8 neighbor peers. The connection time of a specific peer 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 the 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 [11]). Grid resources are assumed to be classified in a number of classes Nc equal to 5. The mean number of agents that travel the Grid is about $Np/2$: 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 is set to 60 s, and the maximum number of hops in a single agent movement, $Hmax$, is set to 3. The visibility radius Rv , defined in Section 2 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.

Table 1. Simulation parameters

Parameter	Value
Grid size (number of peer), Np	900 to 4900
Mean peer connection time, $PlifeTime$	100,000 s
Mean number of resources published by a peer	15
Number of classes of resources, Nc	5
Mean number of agents generated by a peer, $Ngen$	0.5
Mean 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

Simulation analysis proved that ARMAP protocol is robust with respect to variations of the above mentioned parameters, as shown in [6]. For example, the number of resources published a peer, the number

of resource classes, the number of agents and the value of $Hmax$ can affect the rapidity of the process, and in some cases can slightly influence the steady values of performance indices, but the qualitative behavior of ARMAP is always preserved.

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 descriptors. The dissemination index $Nrpp$ is defined as the mean number of descriptor replicas that are available on the network for each resource. Since new replicas are only generated by ARMAP agents that work in the *copy* mode, the number of such agents, $Ncopy$, is another interesting performance index. The processing load L is defined as the number of agents per second that are received and processed by a single peer.

Finally, $QoSavg$ is the average QoS value of all the descriptors that are present in the Grid at a given time.

3.2 Performance and Tuning of ARMAP

Prior to the numerical analysis, a graphical description of the behavior of the replication algorithm is given in Figure 1. For the sake of clearness, here the number of classes is set to 3, and the peers are arranged in a 2-dimensional mesh. The figure shows only a portion of the Grid, though the simulation was performed on a network with 2500 hosts.

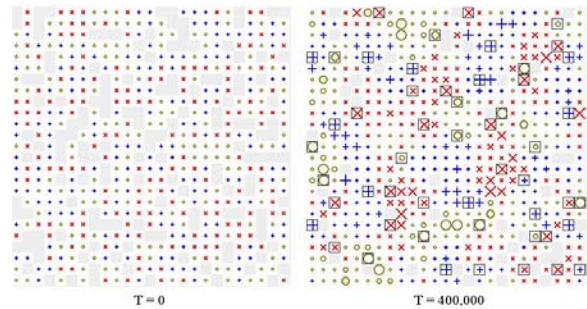


Fig. 1. Accumulation and reorganization of resource descriptors, belonging to 3 resources classes, in a region of the Grid.

Different symbols are associated with the three classes. Each peer is visualized by the symbol that corresponds to the class to which the largest number of descriptors, maintained by this peer, belong. Furthermore, the symbol size is proportional to the number of descriptors of the dominant class. Two snapshots of the network are depicted: the first is taken when the ARMAP process is initiated, at time 0, the

second is taken 400,000 seconds later, in a quite steady situation. This figure shows that descriptors are initially distributed in a completely random fashion, but subsequently they are accumulated and reorganized by agents in separate regions of the Grid, according to their class. The peers with a marked border are *representative* peers that work as attractors for query messages, as briefly described in Section 3.4.

A set of simulation runs have been performed to evaluate the performance of the basic pick and drop probability functions, described in Section 2.1, and the effectiveness of the pheromone mechanism that drives the mode switch of agents, explained in Section 2.2. These simulations were performed with a Grid size N_p equal to 2500 and a number of classes N_c equal to 5. Figure 2 reports the number of agents that work in *copy* mode, N_{copy} (also called *copy agents* in the following), versus time, for different values of the pheromone threshold T_f . When ARMAP is initiated, all the 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 T_f . This corresponds to the sudden drop of curves that can be observed in Figure 2. This drop does not occur if T_f is equal to 10 because this value can never be reached by the pheromone (see formula (4)); hence with $T_f=10$ all agents remain in *copy* along all their lives.

After the first phase of the ARMAP process, an equilibrium is reached because the number of new agents which are generated by hosts (such agents always set off in *copy* mode) and the number of agents that switch from *copy* to *move* get balanced. Moreover, if the pheromone threshold T_f is increased, the average interval of time in which an agent works in *copy* becomes longer, because the pheromone level takes more time to reach this threshold; therefore the average value of N_{copy} at the equilibrium becomes larger. In conclusion, the setting of T_f is a simple and efficient method to tune the number of *copy* agents, and therefore the rate of generation of new replicas.

Figure 3 shows the mean number of replicas per resource and confirms that descriptor dissemination is more intense if the pheromone threshold is increased, because a larger number of *copy* agents operate on the network. However, a more intense dissemination is not always associated to a better reorganization, i.e. to a more effective spatial separation of descriptors belonging to different classes. Figure 4 shows that when the number of *copy* agents is increased, which is achieved by increasing the threshold T_f , the reduction of the overall entropy is lower. For example, with $T_f=3$, the value of the overall entropy decreases from the initial value of about 1 (maximum disorder) to less than 0.72. With $T_f=9$, however, the entropy is only

reduced to about 0.87. As an extreme case, virtually no entropy decrease is observed if all the agents operate in *copy* ($T_f=10$), which confirms that the presence of *move* agents is necessary to perform an effective descriptor reorganization.

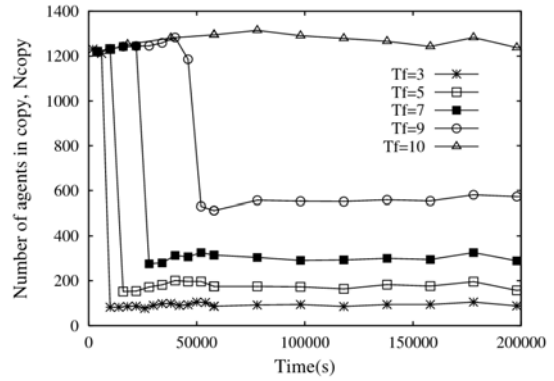


Fig. 2. Number of agents in *copy* mode for different values of the pheromone threshold T_f .

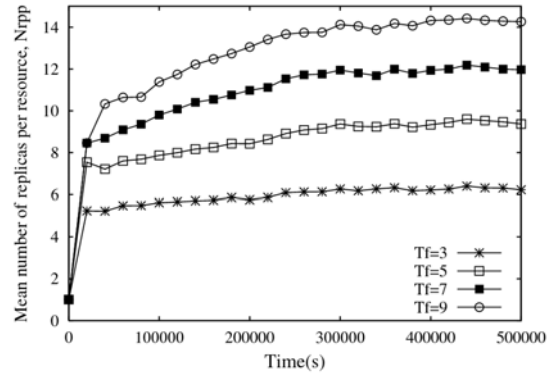


Fig. 3. Mean number of replicas per resource for different values of the pheromone threshold T_f .

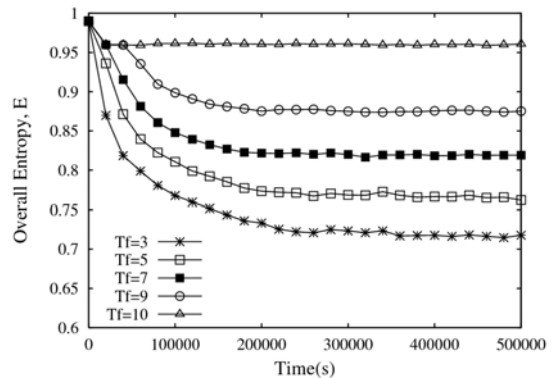


Fig. 4. Overall system entropy for different values of the pheromone threshold T_f .

It can be concluded that *copy* agents are useful to replicate and disseminate descriptors but it is the *move* agents that actually perform the descriptor reorganization and are able to create Grid regions specialized in specific classes of resources. A balance between the two features (replication and reorganization) can be performed by appropriately tuning the pheromone threshold.

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. Recalling that the number of agents N_a is approximately equal to the number of peers N_p times the mean number of agents generated by a peer, N_{gen} , L can be obtained as follows:

$$(7) L = \frac{N_a}{T_{mov} \cdot N_p} \approx \frac{N_{gen}}{T_{mov}}$$

With the parameter setting reported in Table 1, 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 and self-organizing nature, since each agent operates and selects its mode only according to local information collected by the hosts that it visits. The scalable nature is proved by results obtained with different Grid sizes. Results shown in Figure 5 are obtained with $Tf=5$ and show that in all the tested Grids, with N_p ranging from 900 to 4900, the overall entropy is reduced in a similar fashion. The values of other performance indices are also hardly modified by variations in the Grid size.

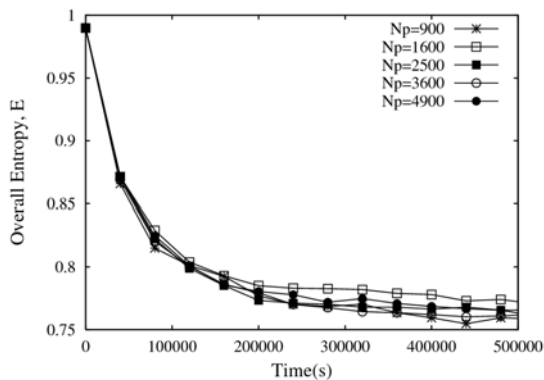


Fig. 5. Overall system entropy for different values of the Grid size N_p , with $Tf=5$.

3.3 Performance of QoS-Aware Pick and Drop Functions

Enhanced, or QoS-aware, pick and drop functions have been introduced, and discussed in Section 2.3, to foster the dissemination of high QoS descriptors and enhance the quality of results returned to users. In fact agents are biased to copy and move high QoS descriptors more likely than low QoS descriptors.

The effectiveness of these functions was evaluated by comparing the results obtained with the *simple* pick and drop probability functions defined in formulas (1) and (2), and with the *enhanced* functions obtained by using the additional “QoS-aware” factors, QoS_{pick} and QoS_{drop} , defined in formulas (5) and (6). Furthermore, different values of the parameter $k4$, used in the QoS-aware factors, were tested. This parameter can steer the QoS-based dissemination of descriptors. In fact, the average value of the QoS-aware factors is always equal to 1 (which means that the overall replication process is not altered), but their variability depends on the value of $k4$. In fact, with a high value of $k4$, the values of QoS_{pick} and QoS_{drop} are always very close to the average, i.e., to 1. Conversely, the variability of these factors can be increased by reducing the value of $k4$: this permits to better differentiate the dissemination of high and low QoS descriptors.

The effect of the QoS-aware pick and drop functions is shown in Figures 6 and 7.

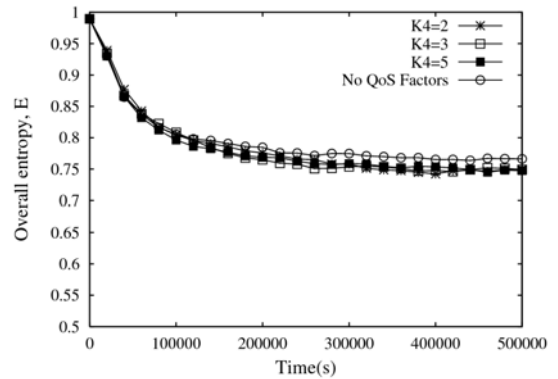


Fig. 6. Overall system entropy for different values of the parameter $k4$.

Figure 6 confirms that the overall entropy is not significantly modified by the use of the new factors, whatever is the value of $k4$. The QoS-aware factors have no influence on the overall dissemination and reorganization of descriptors, which is the looked-for behavior.

On the other hand, Figure 7 shows that the QoS-aware pick and drop functions significantly improve

the average QoS level of the descriptors disseminated on the Grid, referred to as QoS_{avg} . In fact, under the usage of simple pick and drop functions, QoS_{avg} always remains very close to 5, which is the average QoS value of a resource. Conversely, with QoS-aware pick and drop functions, QoS_{avg} gets to higher values, because agents pick and drop high QoS descriptors with a higher probability. It is also noticed that the steady value of QoS_{avg} is inversely proportional to the value of $k4$. For example, with $k4$ equal to 2, QoS_{avg} rapidly increases in the first phase of the ARMAP process, as a result of the activity of a very large number of *copy* agents (see Figure 2), and subsequently gets stabilized at a value of about 6.1, which is more than 20% higher than the average QoS value of a single resource.

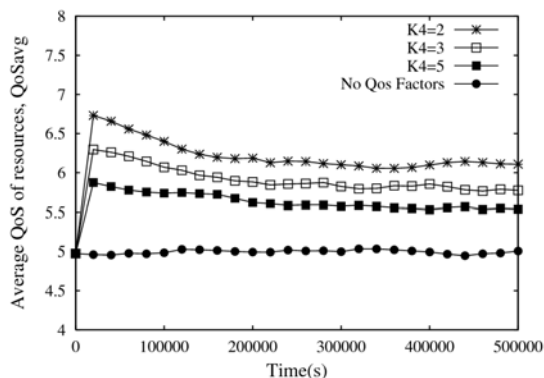


Fig. 7. Average QoS level of descriptors for different values of the parameter $k4$.

3.4 A Semi-Informed Discovery Protocol

The logical resource organization achieved by ARMAP can be exploited through a semi-informed discovery protocol, as mentioned in Section 1, in order to maximize the number of discovered resources and minimize the response time. Such a discovery protocol, namely ARDIP (Ant-Based Resource Discovery Protocol), is based on the notion of *representative peers*. In a Content Network or a Data Grid, a representative peer can be given the responsibility of managing metadata documents of resources of a given class, and offering this information to users. More specifically, as descriptors of a class are accumulated in a Grid region, the peer that, within this region, collects the largest number of descriptors is elected as a representative peer for this class. The objective of a discovery operation is to direct a query message to a representative peer as fast as possible, since such a

peer likely maintains a large number of useful descriptors.

The key features of the ARDIP protocol are discussed in the following, while a detailed discussion can be found in [7]. With ARDIP, a discovery operation is performed in two phases, a *blind* one and an *informed* one. In the *blind* phase, the *random walks* technique is adopted [12]: a number of query messages are issued by the requesting peer and travel the Grid in a *blind* (random) fashion. Whenever a query gets close enough to a representative peer, the search becomes *informed* and the query is driven towards this representative peer. When the query arrives at the representative peer, a queryHit message is generated and gets back to the requesting peer by following the same path in the opposite direction. The proximity of a representative peer is detected through another kind of pheromone mechanism, which is inspired by the behavior of ants that search for a food source. When a query message first discovers a representative peer in its random path, the queryHit message leaves an amount of pheromone in the first peers of its return journey, and this pheromone can be later recognized by other peers that casually approach the representative peer.

The semi-informed protocol aims to combine the benefits of both blind and informed resource discovery approaches which are currently used in P2P networks [21]. In fact, a pure blind approach (e.g. using flooding or random walks techniques) is simple to implement but has limited performance and can cause an excessive network load, whereas a pure informed approach generally requires a very structured resource organization which is impractical in a large, heterogeneous and dynamic Grid.

Results presented in [7] are related to the case in which ARMAP agents use *basic* (i.e., not QoS-aware) pick and drop probability functions. Current work focuses on the analysis of ARDIP under the usage of QoS-aware pick and drop functions, in order to evaluate the possible improvement in the quality of discovered results.

4. Related Work

The main purpose of the protocol presented in this paper is the dissemination of metadata documents and their intelligent reorganization. These two objectives are correlated, since an intelligent dissemination of information can increase efficiency management and facilitate discovery, as discussed in [7].

Information dissemination is a fundamental and frequently occurring problem in large, dynamic and distributed systems whose main purpose is the

management and delivery of content. In [10] it is proposed to disseminate information selectively to groups of users with common interests, so that data is sent only to where it is wanted. In our paper, instead of classifying users, the proposal is to exploit the classification of resources: resource metadata documents are replicated and disseminated with the purpose of creating regions of the network that are specialized in specific classes of resources. In [1] information dissemination is combined with the issue of effective replica placement, since the main interest is to place replicas in the proximity of requesting clients by taking into account changing demand patterns. Specifically, a metadata document is replicated if its demand is higher than a defined threshold and each replica is placed according to a multicast mechanism that aims to discover the data server which is the closest to demanding clients.

The ARMAP protocol, proposed in this paper, is inspired by biological mechanisms, and in particular exploits the self-organizing and swarm intelligence characteristics of several biological systems, ranging from ant colonies to wasp swarms and bird flocks. In these systems, a number of small and autonomous entities perform very simple operations driven by local information, but from the combination of such operations a complex and intelligent behavior emerges [4]: for example, ants are able to establish the shortest path towards a food source; birds travel in large flocks and rapidly adapt their movements to the ever changing characteristics of the environment.

These biological systems can be quite naturally emulated in a Grid through the multi-agent paradigm [20]: the behavior of insects and birds can be imitated by mobile agents which travel through the hosts of a Grid and perform their operations. Multi-agent computer systems can inherit interesting and highly beneficial properties from their biological counterparts, namely: (i) self-organization, since decisions are based on local information, i.e., without any central coordinator; (ii) adaptivity, since agents can flexibly react to the ever-changing environment; (iii) stigmergy awareness [9], since agents are able to interact and cooperate through the modifications of the environment that are induced by their operations.

The ARMAP protocol is specifically inspired to ant algorithms, a class of agent systems which aim to solve very complex problems by imitating the behavior of some species of ants [2]. A technique based on ant pheromone is exploited to tune the behavior of ARMAP agents, making them able to autonomously switch from the *copy* to the *move* mode. This kind of approach is discussed in [22], where a decentralized scheme, inspired by insect pheromone, is used to

control the activity of a single agent and better accomplish the system goal.

The ARMAP pick and drop probability functions are inspired by probability functions that were introduced in [5] (and later elaborated and discussed in [2] and [13]) to emulate the behavior of some species of ants that build cemeteries by aggregating corpses in clusters. In ARMAP, such functions have been adapted through the following main modifications: (i) descriptors are not only moved and aggregated, as are corpses in the mentioned papers, but also *replicated*; (ii) descriptors are reorganized according to the class to which they belong; (iii) additional “QoS-aware” factors are defined to foster the dissemination of descriptors related to high QoS resources.

5. Conclusions

This paper discusses and analyzes an approach for the construction of a Grid information system which disseminates and reorganizes content, specifically metadata documents that describe Grid resources, according to the characteristics of these resources. This permits to improve the efficiency of content management and better respond to discovery requests issued by users.

The proposed ARMAP protocol relies on the combined use of novel distributed paradigms, i.e., multi-agent and P2P techniques. In fact, management of content is performed by a number of ant-like agents that travel the Grid through P2P interconnections among hosts. Agents disseminate resource descriptors on the Grid, and aggregate descriptors related to similar resources in neighbor Grid nodes, so contributing to decrease the overall system entropy. A specific enhancement of ARMAP is tailored to the selective dissemination of descriptors related to high quality resources. This can increase the quality of results that are discovered by users with their queries.

Simulation results show that the ARMAP protocol is able to achieve the mentioned objectives, and is inherently scalable, as agents’ operations are driven by self-organization and fully decentralized mechanisms, and no information is required about the global state of the system.

References

- [1] M. S. Aktas, G. C. Fox, M. Pierce: Fault tolerant high performance Information Services for dynamic collections of Grid and Web services, *Future Generation Computer Systems*, Vol. 23, No. 3, Elsevier Science (2007) 317-337.
- [2] E. Bonabeau, M. Dorigo, G. Theraulaz, G. Swarm Intelligence: From Natural to Artificial Systems, Oxford

- University Press, Santa Fe Institute Studies in the Sciences of Complexity (1999).
- [3] A. Crespo, H. Garcia-Molina: Routing indices for peer-to-peer systems. Proc. of the 22nd International Conference on Distributed Computing Systems ICDCS'02, Wien, Austria (2002) 23-33.
 - [4] P. Dasgupta: Intelligent Agent Enabled P2P Search Using Ant Algorithms, Proc. of the 8th International Conference on Artificial Intelligence, Las Vegas, NV, USA (2004) 751-757.
 - [5] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, L. Chretien: The dynamics of collective sorting robot-like ants and ant-like robots, Proc. of the 1st International Conference on Simulation of Adaptive Behavior on From Animals to Animats. MIT Press, Cambridge, MA, USA (1990) 356-363.
 - [6] A. Forestiero, C. Mastroianni, G. Spezzano: Construction of a Peer-to-Peer Information System in Grids, *Frontiers in Artificial Intelligence and Applications*, Vol. 135, Self-Organization and Autonomic Informatics (I), IOS Press (2005) 220-236.
 - [7] A. Forestiero, C. Mastroianni, G. Spezzano: An Agent-Based Semi-Informed Protocol for Resource Discovery in Grids, Proc. of the 2006 International Conference on Computational Science ICCS, Reading, United Kingdom, Springer-Verlag LNCS 3994 (2006) 1047-1054.
 - [8] I. Foster: Globus Toolkit Version 4: Software for Service-Oriented Systems, Proc. of the International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779 (2005) 2-13.
 - [9] P. Grassè: La reconstruction du nid et les coordinations inter-individuelles chez *bellicositermes natalensis* et *cubitermes* sp. la theorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs, *Insectes Sociaux* 6 (1959) 41-84.
 - [10] A. Iamnitchi, I. Foster: Interest-aware information dissemination in small-world communities, Proc. of the 14th IEEE International Symposium on High Performance Distributed Computing, HPDC 2005. Research Triangle Park, NC, USA (2005).
 - [11] A. Iamnitchi, I. Foster: A Peer-to-Peer Approach to Resource Location in Grid Environments, in *Grid Resource Management*, eds. J. Weglarz, J. Nabrzyski, J. Schopf, and M. Stroinski, Kluwer Publishing (2003).
 - [12] C. Lv, P. Cao, E. Cohen, L. Li, S. Shenker: Search and replication in unstructured peer-to-peer networks, *ACM, Sigmetrics* (2002).
 - [13] M. Martin, B. Chopard, P. Albuquerque: Formation of an ant cemetery: swarm intelligence or statistical accident?, *Future Generation Computer Systems*, Vol. 18, No. 7, Elsevier Science (2002) 951-959.
 - [14] K. Petersen, M. Spreitzer, D. Terry, M. Theimer, A. Demers: Flexible Update Propagation for Weakly Consistent Replication, Proc. of the ACM 16th Symposium on Operating System Principles (1997) 288-301.
 - [15] T. Plagemann, V. Goebel, A. Mauthe, L. Mathy, T. Turletti, G. Urvoy-Keller: From content distribution networks to content networks - issues and challenges, *Computer Communications*, Vol. 29, No. 5, Elsevier Science (2006) 551-562.
 - [16] B. Plale, P. Dinda, G. Von Laszewski: Key concepts and services of a grid information service, Proc. of the 15th International Conference on Parallel and Distributed Computing Systems, PDCS (2002).
 - [17] S. Ran: A Model for Web Services Discovery With QoS, *ACM SIGecom Exchanges*, Vol. 4, No. 1 (2003).
 - [18] J. M. Schopf, M. D'Arcy, N. Miller, L. Pearlman, I. Foster, C. Kesselman: Monitoring and discovery in a web services framework: Functionality and performance of the globus toolkit's mds4, Technical Report ANL/MCS-P1248-0405, Argonne National Laboratory (2005).
 - [19] The Swarm Development Group of Santa Fe: The Swarm environment, University of New Mexico, USA, <http://www.swarm.org>.
 - [20] K. Sycara: Multiagent systems, *Artificial Intelligence Magazine*, Association for the Advancement of Artificial Intelligence, Vol. 10, No. 2 (1998) 79-93.
 - [21] D. Tsoumakos, N. Roussopoulos: A Comparison of Peer-to-Peer Search Methods, Proc. of the 6th International Workshop on the Web and Databases WebDB, San Diego, CA, USA (2003) 61-66.
 - [22] H. Van Dyke Parunak, S. Brueckner, R. Matthews, J. Sauter: Pheromone learning for self-organizing agents, *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, Vol. 35, No. 3 (2005) 316-326.
 - [23] L. Vu, M. Hauswirth, K. Aberer: QoS-based Service Selection and Ranking with Trust and Reputation Management, Proc. of the 13th Conference on Cooperative Information Systems CoopIS 2005, Agia Napa, Cyprus (2005) 466-483.