Multiagent and Grid Systems – An International Journal 2 (2006) 1–9
IOS Press

Distributed holonic multi-agent system for resource discovery in grids

M. Bakhouya, J. Gaber* and A. Koukam
Laboratoire Systèmes et Transports (SeT), Université de Technologie de Belfort-Montbéliard (UTBM), Rue Thierry Mieg 90010 Belfort Cedex, France

Received 22 June 2005
Revised 14 September 2005; 6 October 2005
Accepted 6 October 2005

Abstract. Large scale networks such as computational Grid is a distributed computing infrastructure that can provide globally available network resources. Their size and complexity continue to increase and permit an almost ubiquitous availability of resources; users become able to access network resources irrespective to their location. Therefore, new resource management and access models are required and need to be highly flexible with self-organizing capabilities in order to cope with a dynamically changing environment. In this paper, a distributed and adaptive holonic multi-agent system for resource discovery in Grids is presented. Simulations are presented to demonstrate that structuring servers into holons improves the performance of request resolution processes in dynamic Grid environment.

Keywords: Grids, resource discovery, holonic multi-agent system, mobile agent, random walk, affinity relationships

1. Introduction

The evolution of computing and networking has gone successively from mainframes, to minicomputers and later to personal computers. The rapid growth in personal computing and commodity networking has created much larger distributed-computing environments such as computational Grids [16]. These environments have the potential ability to integrate large-scale computing resources, on demand. Trends today are towards networking resources. User ability to compute will no longer be limited to the resources he has currently at hand or those localized statically on a set of hosts known a priori. For example, a user with a diskless machine can load a word-processor for a punctual need. He no longer needs necessarily a machine with a disk and his own word-processing software installed on it. In addition, he may not know in advance from which host he get the resource throughout his current session. Also, users as well as resources can be mobile or partially connected. The availability of the host and its resource depends on the dynamic behaviour of the network (e.g., failure or network disconnection). When the user submits its request, eventually with QoS requirements, a resource discovery system will process it according to the network status at that instant, and in order to satisfy the QoS constraints [26]. More precisely, resource discovery system should locate and return a set of servers' addresses that match the description of the requested resources. Resources can be divided into two basic categories [16,25]: system resources and application resources. System resources are bound to specific hosts, representing hardware devices (e.g. disk) or logical system objects. Application resources are software entities managed by an application.

Typical resource discovery architecture consists of resource providers that create and publish resources, resource brokers that maintain a repository of published

*Correspondending author: Dr. J. Gaber, Laboratoire SeT, Departement Genie Informatique, Université de Technologie de Belfort-Montbeliard (UTBM). Tel.: +33 3 84 58 32 52; E-mail: gaber@utbm.fr; jaafar.gaber@utbm.fr.

ISSN 1574-1702/06/$17.00

resources to support their discovery, and resources requesters that search the resource broker's repositories. Repositories have traditionally a centralized architecture consisting of multiple repositories that synchronize periodically [8].

In a large-scale network, centralized architecture cannot meet the requirements of both scalability and adaptability simultaneously. The way in which they have typically been constructed is often very inflexible due to the risk of bottlenecks and the difficulty of repositories updating [1,6,8,16]. Also, Grid is a dynamic environment where the location and availability of resources are constantly changing. More precisely, some resources could be disconnected from the network and new ones may join it at any time. Therefore, a new operational model for resource discovery that provide scalability and adaptability issues is required [26,27]. Many efforts are now developed to propose decentralized approaches for resource management [1,6,7,9,27]. Issues are that network resources must be able to scale to billions of nodes and users, able to adapt to dynamic conditions in the network, must be highly available, secure and should require minimal human configuration and management.

In this paper, an autonomous structure of holonic multi-agent system as a model of resource discovery in Grids is presented. Holonic multi-agent systems are considered to be a powerful high-level abstraction approach for complex software systems modelling [4, 5,14].

The rest of the paper is organized as follows. Section 2 presents the related work. In Section 3, we present the holonic multi-agent approach to the grid resource discovery. Section 4 presents the simulation results. Conclusion is given in Section 5.

2. Related work

Several solutions have been proposed to address some issues of resource discovery in Grids. Krauter et al. [16] describe a comprehensive taxonomy for resource management architectures. According to [16], most current resource discovery systems in Grids are based on the classical client/server approaches and support hierarchical or federated organizations. For example, Cao et al. [12] have proposed an agent-based approach for resource management in metacomputing environment using a hierarchy of agents that act as a router between a request and a service. Three types of agents are used. The broker is an agent that heads the hierarchy. A coordinator is an agent that heads a sub-hierarchy. The third agent, termed standard agent, is a leaf-node in the hierarchy. An agent has one connection to an agent higher in the hierarchy to register with and to be registered with many lower level agents. Unfortunately, some important issues related to the proposed hierarchy infrastructure were not addressed in this paper. In particular, the paper do not figure out how new agents that join the system choose their heads in the hierarchy. Also, when a coordinator or the broker agents leave the hierarchy, a question arises regarding how their lower level agents will elect their new heads in the hierarchy.

Today, since Grids increase in sizes and complexity, resource discovery mechanism should be decentralized to avoid bottlenecks and guarantee scalability and adaptability. Several works adopt Peer-to-Peer models to implement non-hierarchical decentralized Grid infrastructures. For example, Iamnitchi et al. [1] have proposed a Peer-to-Peer approach for resource discovery in Grid environment. Each peer acts independently, based on possibly partial or outdated information about the rest of the system and forwards the requests it can not solve to another peer in its neighbourhood. A learning-based strategy for request forwarding is proposed. Peers learn from experience by recording the requests answered by other peers. By this learning strategy, each peer maintains information about other peers in the network. The drawback of this strategy is that the information about peers leaving the network has to be flushed from other peers in the overlay network. This issue however is not addressed in this paper. Wang in [7] has proposed another learning based strategy approach inspired by ant colony and by using mobile agents. In this approach, the learning strategy is based on the use of a migration policy to discover routes between peers with available resources. These routes are then used to resolve user requests. As pointed out by the author from the experiments [7], the main difficulty of this approach is how to design and set up the migration policy parameters to handle different user requests. Sultanik and Regli [9] have proposed another approach based on random walk to maintain and propagate service location information. The autonomous decentralized and self-organizing approach proposed in this paper for resource discovery in Grids is quite different from these works. Peer-to-peer principle embraced by holonic multi-agent system is shown to be the appropriate approach to provide scalable and adaptive resource discovery services.

3. Holonic multi-agent approach to the grid resource discovery

The term holon was originally proposed by Arthur Koestler [2] to model self-organization in biological systems. It is based on the Greek word "holos" for "whole" and the suffix "-on" that denotes "part". This term reflects the tendencies of holons to act as autonomous entities, and yet as cooperative parts that form an apparently self-organizing system. As biological examples, the human being consists of organs which in turn consist of tissues that can be further decomposed into cells. Also, the human being is part of a family and a society [18–20].

The concept of holonic agent or holon was primarily introduced in MAS to model capabilities that emerge from the composition of several agents' activities. More precisely, a holon may have an emergent functionality that none of its body agents could perform for alone. In addition, body agents may be holonic agents themselves. Hence, holonic multiagent systems provide terminology and theory that transfer modularity and recursion to agent paradigm [18,19]. Primarily, this concept has been exploited in manufacturing and transportation domains to define and build structures called holarchies or holonic organizations [11,15]. In these domains, goals can be recursively decomposed into subtasks that can be assigned to individual holons in the holarchie [20]. The main issue is the specification of a holonic organization for the multi-agent system at design-time. In dynamic setting, the challenge becomes how to define a self-organizing mechanism that determine the most appropriate organizational structure for the system at run-time and according to environment changes [26,27].

According to Gerber [5], three organizational structures for modelling holonic multi-agent systems have being proposed. The first organizational structure considers a holon as a federation of autonomous agents. In this case no agent has to give up its autonomy, and the holon is emerged through decentralized cooperation among the agents. In the second organizational structure, a holon is considered as a moderated group of agents. In this case, agents give up only part of their autonomy to the super-holon (i.e. the representative of the holon). In the third organizational structure, a holon is represented by a flexible group of agents. The realization of this approach assumes procedures of splitting and merging agents that lead to the creation of new ones

In the context of resource discovery, the Grid is considered as a collection of geographically distributed servers that may join and leave the network in unpredictable manner at any time. These servers are organized into decentralized holonic multi-agent systems as depicted in Fig. 1. More precisely, a holon represents a service. A service is composed by a set of hardware or software resources that users need to discover and select. These resources could be interfaced by web services. Web services are applications that permit to describe software components; in particular they define identification and accessing methods that enable the discovery and the use of these components (i.e., the resources).

For example, for a punctual need, a user who would like to open video files or create video CD might need the following resources: a video player, format transcoding software, the MPEG-4 codec (for his wireless laptop), video effect or edge detection algorithms etc... Since he does not know from which hosts he should get these resources throughout his current session, the user submits his multimedia composed service request, eventually with QoS requirements, to the resource discovery system that will process it according to the network status at that instant.

To address scalability and adaptability issues, it is more appropriate to use the holonic organization based on a federation of autonomous agents. This organizational structure is required to be self-organizing with inherent support for scalability and adaptability to user requirements and network changes. To address scalability, server agents (Sagent) that represent the resources of their corresponding servers in the network are organized into communities. Each community is an holon. More precisely, server agents establish relationships between them based on their affinity. Affinity corresponds to the adequacy with which two server agents could bind to create a composed service or to point out a similar service. Adequacy could be implemented based on keywords or objects in common describing a service or resources. To address adaptability issue, affinity relationships between server agents are dynamic; the affinity values can be adjusted at run-time to cope with changes in the network.

The resource discovery system or middleware for a service will proceed as follows. It will first locate an appropriate holon and then resolve the request within this holon. Within a holon, each server agent represents the resources of its corresponding server in the network. Each Sagent can store and provide information about its set of resources and learns about a subset of other server

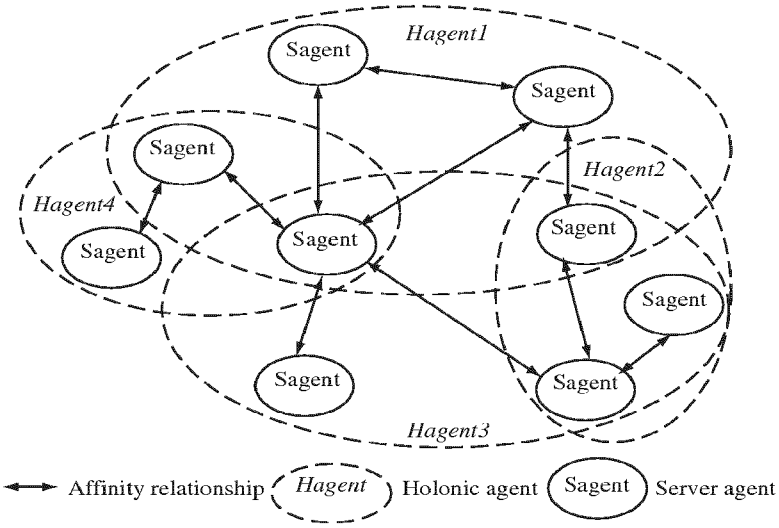

Fig. 1. Server agent contains affinity relationships to other server agents. Server agents together with their affinity relationships as a whole form holons on which requests resolution are performed.

agents, together with their available resources. In other words, each server agent has affinity relationships to, and information about, several other server agents in the network (e.g., their IP address). These affinity relationships exist only between the different server agents that provide the resources needed to compose the service represented by the holon.

The proposed holonic based organization assumes that Sagents are autonomous agents while holon agents (Hagents) are emerged structures that are not represented explicitly, but they exist through the affinity relationships between Sagents. In other words, Sagent cooperate equally rather than being assigned subordinate and supervisory relationships. It should be noted that an individual Sagent on its own can also be viewed as an individual holon. Recall that a holon in this proposed approach represents a composed service that can be provided by a set of Sagents together (i.e., a servers group). It is worth noting also that this multi-organization based on dynamic affinities supported by relationships provides a highly decentralized system while remaining adaptive in a dynamic environment. More precisely, this decentralized architecture offers a high degree of resilience against servers leaving the network. For example, when a server leaves the network, all the peer relationships with other servers are removed without additional overhead since it does not rely on any overlay control structure. Also, a leaving server or a communication failure does not have impact on the resolution of user requests.

It should be noted that hierarchical organisational structure using super-holon cannot meet the requirements of both scalability and adaptability simultaneously for resource discovery in Grid due to the difficulty of updating information about services, the risk of bottlenecks and the presence of a single point of failure. In addition, each arriving and leaving servers need to actively notify the super-holons in the hierarchy.

In this proposed approach, service discovery consists of two processes: resource dissemination process and user request resolution process. The resource dissemination is a mechanism for server agents to learn about the existence of each other in order to create affinity relationships. The request resolution process is the process by which the user or an application will be provided by the required resources information. These two processes of the resource discovery approach will be described in the next section.

3.1. The resource dissemination process

By the resource dissemination process, Sagents should interact and establish affinity relationships between them in order to form Hagents. This interaction is carried out in the following manner. When a server joins the network, it creates an agent, called disseminator agent, to publish its data. More precisely, this disseminator agent is a mobile agent that initiates a random walk and moves randomly in the network. A random walk is a stochastic process that evolves in the fol-

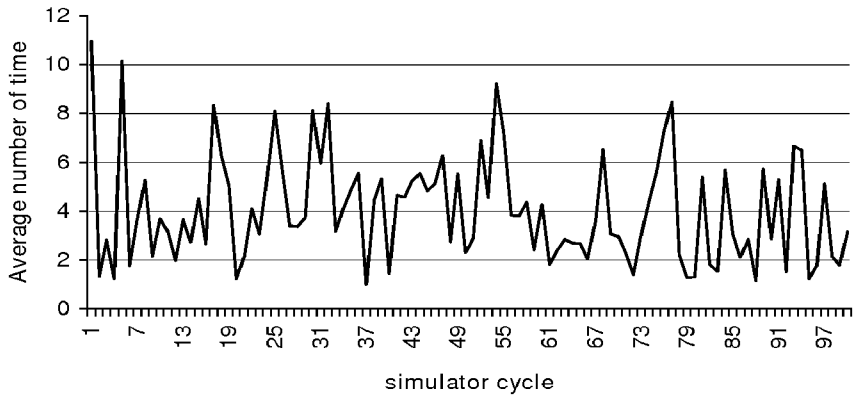

Fig. 2. Average number of time to resolve the user requests in the case of resource discovery without dissemination.

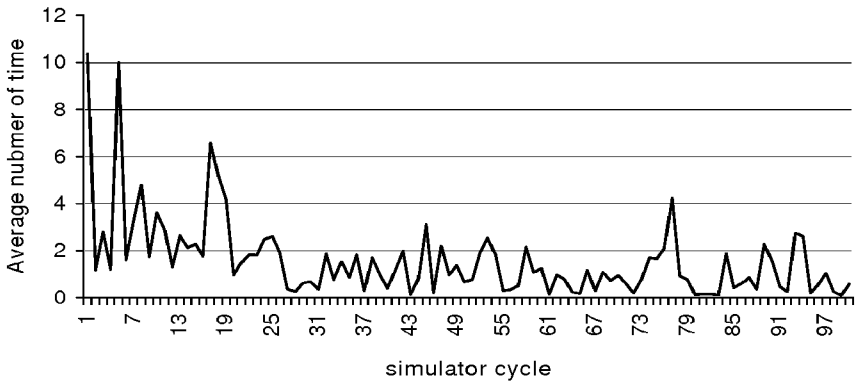

Fig. 3. Average number of time to resolve the user requests in the case of resource discovery with dissemination.

lowing way. At each step, a mobile agent chooses one of the nearest neighbours of its current location at random and steps to that neighbour. The process continues this way by taking random steps that are independent of all the previous moves. By cloning, disseminator agents can create replications of themselves. It is well known that parallel random walks in a connected network will cover the entire network asymptotically [3, 10,17]. This implies in turn that asymptotically, all Sagents will learn about the existence of the all other Sagents in the network. In particular, when two Sagents detect an interest in common, a relationship is established and the two Sagents will group themselves into the same Hagents. More generally, two Sagents have interest in common if their respective resources allow the creation of a composed service or if the Sagents holds a similar resource or service. Also, when a server leaves the network, all the peer relationships with other servers are removed.

Regarding the agent cloning operation, the distributed algorithm proposed by Amin and Mikler in [13] is used to regulate and control dynamically the number of clones spawned in the network.

3.2. The request resolution process

The request resolution process is the process by which the user will be provided by the required service. To locate a service, the user creates a mobile agent, called request agent. This agent initiates a random walk in the network until it meets an appropriate holon that can resolves the request (i.e. an initial entry point of a holon). In this holon, the request agent uses and adjusts affinity relationships in two stages: request forwarding stage and result backtracking stage. During the forwarding stage, the request agent guided by the affinity relationships seeks server agents from inside the holon that can provide the required resources. In

order to avoid loops, the request agent stores the list of the visited servers. By the affinity adjustments during this stage, a selected path from within the holon graph will emerge as a response to the request. When the all required resources are discovered, the request agent starts the backtracking phase. During this phase, the path computed between an end point in the holon and the initial entry point will be reinforced globally by secondary affinity adjustments. This phase makes the holon more adapted to future requests. Therefore, the resource discovery system will be able to process the most frequent requests more efficiently. The rest of this section presents the request forwarding and result backtracking phases.

3.2.1. Request forwarding

Within a holon, each server agent contains affinity relationships to other server agents. Each server agent provides at least one resource. A service can be represented by a set of resources $(r_1, r_2, \ldots r_n)$ and called it a service path. To locate a service, the user creates a request agent and gives it resources to be located, IP user address and the TTL (time-to-live). The TTL is the maximum number of server agents that can be visited by a request agent before it returns if the required service is not found. The request agent moves randomly in the network until it meets a server agent matching one of required resources (i.e. an entry point to a holon) and a request forwarding process starts through this holon. This server agent has several relationships to other servers. Consequently, there is more than one service path in the holon that can resolve the request. Indeed, the request agent needs to make local decision to determine which relationships matching one of searched resources that are not found yet. The greedy search strategy with reinforcement of affinity relationships can be applied by request agent to find a required service path. More precisely, the request agent is guided by the relationships to server agents providing resources that belongs the required service path. If there is more than one server, the request agent selects a relationship to one among them that has higher affinity value. Affinity is a real variable that is adjusted or reinforced by request agent satisfaction as in [21]. This satisfaction, called happiness in [21], is used to measure the degree of end user satisfaction with a service. According to this work, the strength of relationships between entities is modified by applying the Steepest Decent Method. In the proposed approach, the request agent satisfaction is described by required resources according to resources offered by server agents during request forwarding phase. The affinity variation for a particular request r is determined as follows:

$$\Delta m_{ij}(r) = \mu(LocSat_{ij}(r) - f(m_{ij})) \quad (1)$$

m_{ij} is the value of the affinity relationship between a server s_i and a server s_j regarding the holon service. $LocSat_{ij}(r)$ is the satisfaction value of the request agent, residing at server s_i, by the resource provided by server s_j to resolve the request r. More precisely, if the request agent residing at server s_i searches a resource which is provided by server s_j to resolve the request r, then the request agent is satisfied and $LocSat_{ij}(r)$ is equal to 1 and 0 otherwise. Consequently, the request agent adjusts the affinity of relationships m_{ij} using Eq. (1) where μ is a positive value between 0 and 1. The value of affinity is mapped to a value between 0 and 1 by using the logistic equation:

$$f(m_{ij}) = \frac{1}{1 + e^{-m_{ij}}} \quad (2)$$

With this Eq. (2), the affinity value m_{ij} increases quickly when it is near 0 and satisfaction is equal to 1. Also, the affinity value m_{ij} decreases quickly when the satisfaction is equal to 0.

Let consider that request agent searches the service r composed of three resources r_a, r_b and r_c. A request agent residing at server s_1 found the resource r_a in s_1 and would like to search the resource r_b and r_c. The request agent perceives that the server s_1 has relationships to servers s_2, s_3, and s_4. The server s_2 and s_3 provide both the resource r_b, and the server s_4 provides a resource r_d. In this case, servers s_2 and server s_3 can satisfy the request and the server s_4 can not. Consequently, the value $LocSat_{12}(r)$ and $LocSat_{13}(r)$ are set to 1, but $LocSat_{14}(r)$ is set to 0. After the affinity adjustment, using the Eq. (1), the request agent will have the choice of keeping one of two relationships (server s_2 or server s_3). The request agent chooses the relationship that has higher affinity value. Let consider that the affinity value m_{12} is higher compared to affinity m_{13}. In this case, server s_2 is considered to be the most suitable server by the request agent to resolve the request r. The request agent then decrements its TTL and moves to server s_2. A request agent repeats the same process until it find the resource r_c or its TTL is expired (i.e., TTL becomes equal to 0).

3.2.2. Result backtracking

During the result backtracking phase, the request agent goes back from the end point of the holon, via the intermediate server agents on the founded service path,

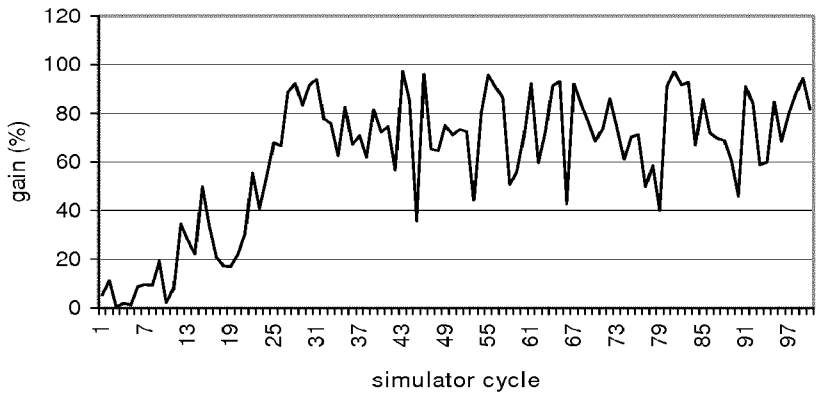

Fig. 4. The gain (%) in term of average number of time to locate required resources in the case of resource discovery with dissemination compared to average number of time to locate the required resources in the case of resource discovery without dissemination.

to the initial point of the holon. During this phase, the request agent adjusts the affinity relationships between the server agents, based on its global satisfaction. The affinity variation for a particular request r between a server s_i and a server s_j is determined as follows:

$$\Delta m_{ij}(r) = \mu(GloSat(r) - f(m_{ij})) \quad (3)$$

$GloSat(r)$ is the global satisfaction value of the request agent regarding the provided service. More precisely, when the request agent is on the server s_j, it moves back to its predecessor server s_i and adjusts the affinity relationship m_{ij} using Eq. (3), wherein $GlobSat(r)$ is set to 1. The request agent repeats the same process until it reaches the initial point of the holon. If the time to live (TTL) of request agent has expired or the required resources are not found, the value of $GlobSat(r)$ is set to 0. In other words, affinity relationships are reinforced with positive satisfaction if resources are found and with negative satisfaction if the time to live is expired and or if the request is not satisfied. Recall that μ is a positive value between 0 and 1, and affinity values are mapped to a value between 0 and 1 using Eq. (2).

For example, let us suppose a particular request r composed of four resources r_a, r_b, r_c and r_d, the request agent has visited four servers s_1, s_2, s_3, and s_4. Let consider that the request agent is now located in server s_4 and that all the required resources are found. The request agent starts then the result backtracking phase and moves back from server s_4, via server s_3 and s_2, to server s_1. During this phase, the request agent reinforces the links m_{34}, m_{23} and m_{12} using the Eq. (3), wherein $GloSal(r)$ is set to 1.

4. Simulation results

The proposed discovery approach is evaluated by a simulator implemented with NS2 [22,24]. A network of 100 servers is generated randomly with BRITE generator [23]. Each server provides one resource of ten kinds of resources. The simulation abstracts any considerations about networking issues such as bandwidth constraints and time processing. The code and state size of mobile agent are constant and do not change during the simulation. Recall that the objective of the resource discovery system is to discover and select server groups (i.e., Holon) that can resolve the requests.

Two strategies are compared. In the first strategy, servers' data dissemination is not performed and thus server agents have no knowledge about resources provided by others in the network. In other words, server agents do not have affinity relationships with other server agents and there are no holons. At each simulation step, 10 request agents created at some servers selected randomly are asking for different kinds of resources generated randomly between 1 and 5. Request agents walk randomly in the network until to meet servers with the required resources and resolve their request.

In the second strategy, at the beginning of the simulation, each server agent creates a disseminator agent that initiates a random walk in the network to disseminate its data. By cloning, disseminator agents can create replications to themselves. To control and regulate the number of clones spawned in the network, the AM algorithm due to Amin and Mikler [13] is used. At each simulation step, 10 request agents created at some servers selected randomly are asking for different kinds

of resources generated randomly between 1 and 5. Request agents walk randomly to seek server agents with the required resources. As the simulation progress, request agents start move through established affinity relationships inside holons. By the affinity adjustments during the request forwarding and backtracking request processes, for a particular request, a selected path from within an appropriate holon graph will emerge as a response to that request.

Figure 2 shows the average number of time to resolve the request in the case of resource discovery without dissemination. For each request resolution, this represents the average number of time used to request agents forwarding and a result backtracking. This result shows that without resource dissemination each server agent has no knowledge of the resources provided by other server agents. Consequently, the request agent may traverse a large number of server agents. Figure 3 shows the average number of time to resolve the request in the case of resource discovery with dissemination. At the beginning of the simulation, there are no holons, and request resolution performs poorly. As more simulator time elapses, server agents create many affinity relationships and there are many holons, leading to improve performance in request resolution. Each created affinity relationship is initialised with 0, and this value is updated during resolution request process using Eqs (1) and (3).

Figure 4 shows the gain (%) of the average number of time to resolve the request in the case of resource discovery with dissemination compared to average number of time to resolve request in the case of resource discovery without dissemination. Comparison of these results demonstrates that the creation of holons based or resource dissemination and affinity adjustment is useful for improving performance in request resolution. At the beginning of the simulation, the gain is negligible. As more simulator time elapses, server agents create many affinity relationships and the gain become important (approximatively 80%).

5. Conclusion

In this paper, an autonomous decentralized approach for resource discovery in Grids is presented. In this approach, peer-to-peer principle embraced by holonic multi-agent system is shown to be the appropriate approach to provide a scalable and adaptive resource discovery system. Server agents are organized into communities. Each community is an holon. More precisely, server agents establish relationships between them based on their affinity. Affinity corresponds to the adequacy with which two server agents could bind to create a composed service or to point out a similar service. These affinities are adjusted or reinforced by the request agent satisfaction during request forwarding and result backtracking phases. Simulations demonstrate that, compared to simple random walk, structuring servers into holons improves the performance of request resolution processes in a dynamically changing environment such as the Grid. Future work will address the specification issue of the proposed approach together with additional simulations with ns2 to evaluate the approach performance when variations of random walk schemes are considered.

Acknowledgements

The authors would like to thank the referees for their careful reviews and suggestions in the improvement of the paper.

References

[1] A. Iamnitchi and I. Foster, A peer-to-peer approach to resource location in grid environments, in: *Grid Resource Management, Kluwer Publishing*, J. Weglarz, J. Nabrzyski, J. Schopf and M. Stroinski, eds, 2003, http://www.cs.duke.edu/~anda/.

[2] A. Koestler, *The ghost in the machine*, hutchinson, 1967.

[3] A.Z. Broder, A.R. Karlin, P. Raghavan and E. Upfal, Trading space for time in undirected s-t connectivity, *ACM Symposium on Theory of Computing* (1989), 543–549.

[4] B. Horling and V. Lesser, *A survey of multi-Agent organizational paradigms, Knowledge Engineering Review*, 2005, To appear. An earlier version is available as UMass Computer Science Technical Report 04-45, http://dis.cs.umass.edu/~bhorling/papers/horling-paradigms.pdf.

[5] C. Gerber, J. Siekmann and G. Vierke, *Holonic multi-agent systems*, Technical Report R-99-03, DFKI GmbH, ISSN 094-008X, 1999.

[6] D. Talia and P. Trunfio, *Web services for peer-to-peer resource discovery on the Grid*, DELOS Workshop: Digital Library Architectures 2004, 73–84.

[7] D. Wang, *A resource discovery model based on multi-agent technology in P2P system*, Intelligent Agent Technology (IAT'04), IEEE/WIC/ACM International Conference, 2004, 548–551.

[8] D. Xu, K. Nahrstedt and D. Wichadakul, *QoS-aware discovery of wide-area distributed services*, in First IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid), 2001, 92–99.

[9] E. Sultanik and W. Regli, *Service discovery on dynamic peer-to-peer networks using mobile agents*, Third International Workshop on Agents and Peer-to-Peer Computing (AP2PC 2004), http://www.sultanik.com/publications/.

[10] H. Baala, O. Flauzac, J. Gaber, M. Buid and T. El-Ghazawi, A self-stabilizing distributed algorithm for spanning tree construction in wireless ad hoc networks, *Journal of Parallel and Distributed Computing* **63**(1) (January 2003), 97–104.

[11] H.J. Bûrckert, K. Fischer and G. Vierke, Holonic transport scheduling with TeleTruck, *Journal of Applied Artificial Intelligence* **14** (Taylor & Francis 2000), 697–725.

[12] J. Cao, D.J. Kerbyson and G.R. Nudd, *Use of agent-based service discovery for resource management in metacomputing environment*, Euro-Par'01, 2001, 882–886.

[13] K.A. Amin and A.R. Mikler, *Dynamic agent population in agent-based distance vector routing*, Second International Workshop on Intelligent Systems Design and Applications, Atlanta, USA, August, 2002, 195–200.

[14] K. Fischer, M. Schillo and J. Siekmann, *Holonic multiagent systems: The foundation for the organization of multiagent systems*, Proceedings of the First International Conference on Applications of Holonic and Multiagent Systems (HoloMAS'03), LNAI 2744:71–80, Springer-Verlag 2003.

[15] K. Fischer, Agent-based design of holonic manufacturing systems, *Journal of Robotics and Autonomous Systems* **27:1-2:3-13** (1999), Elsevier Science B.V.

[16] K. Krauter, R. Buyya and M. Maheswaran, A taxonomy and survey of Grid resource management systems for distributed computing, *Software Practice and Experience* **32**(2) (2002), 135–164.

[17] M. Bui, S.K. Das, A.K. Datta and D.T. Nguyen, Randomized agent based routing in wireless networks, *International Journal of Foundations of Computer Science* **12**(3) (2001), 365–384.

[18] M. Schillo and K. Fischer, *Holonic multiagent systems*, KI Zeitschrift, 4:54, arendtap 2003, http://www.virtosphere.de/data/publications/articles/SchilloFischer.HoloMAS.pdf.

[19] M. Schillo, B. Fley, M. Florian, F. Hillebrandt and D. Hinck, *Self-organization in multiagent systems: from agent interaction to agent organization*, in Proceedings of the 3rd International Workshop on Modeling Artificial Societies and Hybrid Organizations (MASHO'02), Workshop at KI2002, the 25th German Conference on Artificial Intelligence Aachen, 47–56.

[20] M. Ulieru, R. Brennan and S. Walker, The holonic enterprise A model for Internet-enabled global supply chain and workflow management, *Int. Journal of Integrated Manufacturing Systems* (13/8) (2002), ISSN 0957–6061.

[21] T. Itao, T. Nakamura, M. Matsuo, T. Suda and T. Aoyama, Service emergence based on cooperative interaction of self-organizing entities, *Proc. of the IEEE SAINT* (2002), 194–203.

[22] The network simulator NS-2, available at the Information Sciences Institute web site: http://www.isi.edu/nsnam/ns/.

[23] The BRITE generator, available at the Computer Sciences Department of Boston University web site: http://www.cs.bu.edu/brite/.

[24] The NS-2 patch, available at the Faculty of Information Technology, Mathematics and Electrical Engineering, Department of Telematics web site: http://www.item.ntnu.no/~wittner/ns/index.html.

[25] Y. Gidron, O. Holder I. Ben-Shaul and Y. Aridor, Dynamic configuration of access control for mobile components in Fargo, *Concurency and Computation* **13**(1) (January 2001), 5–21.

[26] J. Gaber, New paradigms for ubiquitous and pervasive computing, Internal research report Université de Technologies de Belfort-Montbéliard (UTBM), September 2000.

[27] M. Bakhouya, Self-adaptive approach based on mobile agent and inspired by human immune systrm for service discovery in large scale networks, PhD Thesis N°034, Université de Technologies de Belfort-Montbéliard, 2005.

Authors' Bios

Mohamed Bakhouya received the B.S. degree in computer science engineering from the Faculty of Sciences and Technologies (FST), Errachidia, Morocco in 1998, and the Master's degree from the "Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes" ENSIAS, Rabat, Morocco in 2001. He is pursuing his Ph.D. in computer science at the University of Technology of Belfort-Montbéliard (UTBM), France. His research interests include distributed algorithms, mobile computing, artificial intelligence and grid computing.

Jaafar Gaber received the Ph.D. degrees in 1998 from University of Science and Technology of Lille (France) in Computer Science. He is currently an Associate Professor of Computational Sciences and Computer Engineering at the University of Technology of Belfort-Montbéliard UTBM (France). Prior to joining UTBM, he was a research scientist at the Institute of Computational Sciences and Informatics (CSI) in George Mason University in Fairfax (Virginia, USA). His research interests include high-performance computing, distributed data mining, Biocomputing, Distributed algorithms and mobile computing, computer networks and communications.

Abderrafiaa Koukam received the Ph.D. degree in computer science from University of Nancy I, Nancy, France, in 1990. From 1986 to 1989, he served as an Assistant Professor at the University of Nancy I and Researcher with the "Centre de Recherche en Informatique de Nancy (CRIN)" from 1985 to 1990. In 1999, he received the enabling to direct the research in computer science from University of Bourgogne, Bourgogne, France. Presently, he is a Professor of Computer Science at the Université de Technologie de Belfort – Montbeliard (UTBM), Belfort, France. He was elected Vice President of the Scientific Council of UTBM. He heads research activities at the Systems and Transportation (SeT) Laboratory on modelling and analysis of complex systems, including software engineering, multiagent systems, and optimisation.