Decentralized Semantic Service Discovery in MAS

E. del Val, M. Rebollo, and V. Botti*

Grupo de Tecnología Informática - Inteligencia Artificial
Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Camino de Vera S/N 46022 Valencia (Spain)
{edelval,mrebollo,vbotti}@dsic.upv.es

Abstract. Service-Oriented Multi-Agent Systems (SOMAS) are dynamic systems populated by heterogeneous agents which can enter or leave the system dynamically. As a consequence of the heterogeneity feature, organizational concepts such as roles are useful to facilitate the coordination in the system. In addition, agent functionality should be modelled as services in order to allow heterogeneous agents or other entities to interact in a standardized way. Furthermore, due to the large-scale and the adaptive needs of the system, the traditional directory facilitators or middle-agents are not suitable for the management of the agents services. In this paper we present a distributed approach for agent service management in SOMAS based on social networks. The proposal provides a fully decentralized structure and allows agents to locate services using only local information. The system is enhanced using semantic information in the generation of the system structure and also in the search process.

1 Introduction

Service-Oriented Multi-Agent Systems (SOMAS) can be described as open and dynamic systems, where agents provide basic functionality through services and new agents can enter to the system and existing ones leave. An important issue that has raised great interest in the research community in the latest years is service discovery. In open systems where there is a large number of agents and the available agents change dynamically, finding the appropriate agent which offers the service required is not an easy task. Conventional approaches to locate agents with certain functionality in SOMAS, such as registries or middle-agents, are centralized approaches which are not always appropriated for large-scale and highly dynamic environments. These proposals present some weakness such as bottlenecks, complexity or the huge amount of memory needed to keep all the information about the agent's functionality when the system scales. Distributed approaches, such as agent coalitions or federations of registries, have been proposed to solve some of these problems but the required coordination effort to create the coalitions and to maintain data consistency between distributed registries makes these proposals not suitable for highly dynamic environments.

* This paper is a reviewed version of a paper named "Semantic Service Discovery in MAS Using Social Networks" which has been presented at WAT (Workshop on Agreement Technologies), in November 2010.

An alternative for traditional proposals is the use of social networks[19][22]. Human beings create social structures in a decentralized way which allows to locate other individual in a few steps considering only local information. This fact was observed by Milgram in the well known experiment of 'six degrees of separation'[18]. The results of this experiment arose two questions: how is the structure of these social networks and how an effective search of individuals is carried on only with local information. In the wake of this experiment, several works started to pay attention on the analysis of the underlying structures in human societies and the properties of these structures.

As a result of that, several models based on mathematical functions have been proposed to simulate the structure in real social networks. These models try to reflect how social links are established between individuals to form a network which can guide the search. These networks such as small-world or preferential attachment network, are called *navigable social networks* and it can be ensured that short paths between two random individuals can be found using only local information. How an effective search is carried on only with local information is the other important aspect. Which criteria should be follow by the individuals in order to guide the search towards the target? This depends on the structure of the network. There are some strategies that have better results depending of the underlying structure where it is applied. For instance in small-world networks similarity or geographical distance can be considered a good parameters to consider while strategies guided by degree are not so suitable.

In this work, we propose the use of a social network model as the underlying structure of a service discovery system for agents. The structure relies on a social feature present in many social networks called *homophily*[9]. *Homophily* expresses the idea that similar people interact with higher frequency than dissimilar people. Therefore, in our system agents with similar roles and services have more probability to be linked. The system provides a fully decentralized structure and allows agents to locate services using only local information. The system is enhanced using semantic information in the generation of the system structure and also in the search process.

2 Related Work

Open and dynamic environments where the scalability and the workload are low make use of middleagents to facilitate service discovery [8][16][17]. The matchmakers could provide an optimal matching due to they consider all the registered services in the system. Unfortunately, this kind of agents could be a bottleneck when the workload increases. Other drawbacks are their complexity, the huge amount of memory needed to keep service advertisements and the cost of service composition as the number of services grows significantly. Different approaches have been suggested to overcome the above mentioned problems. *Peer-to-peer* approaches [6][2][14] broadcast a query using local knowledge The drawback of this approach to service discovery is that the communication among agents is essential and the overall communication traffic overhead may be large. Another distributed way to locate distributed services is to form *coalitions* or clusters[13][12][10]. Nevertheless, the choice of what coalitions are going to be formed is a difficult task. This entails recursively to calculate the values of the coalitions and later selecting the coalition with the best result. A third way for agents to discover ser-

vices in efficiently is the distribution of the *middleagents* or *facilitators* [15][11][7]. These proposals suggest to split the function of the service facilitator among a group of agents. The system designer assigns a local matchmaker to each host or segment of the system, which provides matchmaking services to agents in its vicinity (its segment). In systems with very large segments the problems of scalability are only marginally relieved by this approach because the large segments become overloaded systems which have local bottlenecks. Another case in which this approach is not useful is in systems with many cross-links between segments. In this case the overhead of coordinating tasks among local matchmakers might be greater than the benefit obtained from their distribution.

3 Proposed System and Definitions

The proposal that we present here tries to overcome drawbacks of current discovery approaches in open SOMAS through a completely distributed approach, considering semantic information about organizational roles and services. This approach is based on social networks as underlying structure. The advantages and contributions of this proposal compared to others are:

- System which integrates services and agents. Agents have social and proactive capabilities which provide more flexibility and adaptability to the system. Services facilitate the reusability and interoperability. Here, agents functionality is described in terms of services, therefore we obtain the advantages of both technologies.
- System structure that guarantees, in general, that a service if it exits is going to be found in a bounded number of steps.
- Service discovery strategy that only needs local information to navigate the network in order to reach the required service.
- The use of semantic information to create the system structure and to lead the service search.
- Inclusion of organizational information in the service discovery process.

DEFINITION **1** *(Agent-Service Discovery System). An SDS is defined as $\mathcal{SDS}$ = ($\mathcal{A}$, $\mathcal{L}$), where $\mathcal{A}$ is the set of agents that are part of the $\mathcal{SDS}$ (nodes): $\mathcal{A}$= $\{a_1,...,a_n\}$, and each link ℓ=(a_i, a_j)$\in\mathcal{L}$ indicates the existence of a knowledge or communication relationship between agent a_i and a_j in the system (undirected links).*

Agents are social entities which have local knowledge about its immediate neighbors, including their identity, degree, organizational information and the semantic description of the services they offer, but it is unaware of the rest of the agents present in the system.

DEFINITION **2** *(Agent). An agent a_i=($\mathcal{R}$,$\mathcal{N}$) $|a_i \in \mathcal{A}$ is a social entity which can play several roles in different organizational units $\mathcal{R}$=$\{r_1,\ldots,r_n\}$:$|\mathcal{R}| > 0$, has a neighborhood $\mathcal{N}$=$\{a_k,...,a_m\}| \; a_k \in \mathcal{N}$, $\exists$ (a_i,a_k)$\in\mathcal{L}$, $|\mathcal{N}| > 0$.*

The agent role determines the kind of services an agent offers. It is used to create the structure of the system. Roles are defined inside an organization unit ou. The organization unit establishes a set of policies responsible of the structure of the system. These policies are related to basic system operations (*join*, *leave*, *discover*...).

DEFINITION **3** *(Role). A role in our system is defined as* $r = (\phi, \{s_1, \ldots s_n\}, ou) \in \mathcal{R}$ *where* ϕ *is a semantic concept for the role,* ou *is the organizational unit where the agent plays the role* r *and* $\{s_1, \ldots s_p\}$ *is the set of services offered by the agent.*

Each *service* s_i is a semantic service defined by the tuple: $s_i = (\mathcal{I}, \mathcal{O})$, where $\mathcal{I}$ denotes the set of inputs and $\mathcal{O}$ denotes the set of outputs. The $\mathcal{I}$ and $\mathcal{O}$ of the service are semantic concepts defined in a common ontology. To simplify the notation of the system, we are going to consider that each agent plays one role and offers one service.

The agent-service discovery system that we present relies on a property present in many real social networks: homophily. This word expresses the idea that similar people tend to interact and establish links with higher probability than dissimilar people. There are two types of homophily[9]:

- *choice homophily*, where patterns of interaction are driven by preferences for similarity. This kind of homophily has two forms: *status homophily*, where the individuals are considered similar if they share a cultural background, and *value homophily*, where individuals are considered similar on the basis of shared values, attitudes, and believes.
- *induced homophily*, emerges not from individual choice, but from influence dynamics that make individuals more similar over time.

In this work we focus on choice homophily and its two forms. In general, homophily has demonstrated that is one of the most pronounced features in social networks[3][21]. Due to the efficiency of the social networks with this feature, we consider important to consider this property in our system.

DEFINITION **4** *(Agent homophily). In our* $\mathcal{SDS}$ *the homophily between two agents is based on the status homophily and the value homophily:*

- *value homophily (*$\mathcal{H}_v(a_i, a_j)$*) is defined over the agent's services and it is considered as the semantic similarity between the services offered by the agents.*
- *status homophily (*$\mathcal{H}_s(a_i, a_j)$*) is defined over the agent's role and it is considered as the semantic similarity between the roles played by the agents*

Therefore, the homophily between two agents is defined as the linear combination of value and status homophily:

$$\mathcal{H} = \alpha * \mathcal{H}_v + (1 - \alpha) * \mathcal{H}_s \tag{1}$$

We are going to describe with more detail how are calculated each kind of homophily. The homophily function $\mathcal{H}_s(a_i, a_j)$, means the degree of match dom (*exact*, *subsumes*, *plug-in*, *fail*) between the semantic concept of the roles played by the agents.

$$\mathcal{H}_s(a_i, a_j) = role\_match(\phi_i, \phi_j) \tag{2}$$

where *role_match* is the function which calculates the semantic similarity between ϕ_i and ϕ_j. The homophily function $\mathcal{H}_s(a_i,a_j)$, means the degree of match between the services offered by the agents (s_i and s_j).

$$\mathcal{H}_v(a_i, a_j) = \beta * match(\mathcal{I}_i, \mathcal{I}_j) + (1 - \beta) * match(\mathcal{O}_i, \mathcal{O}_j), \tag{3}$$

where function $match$ solves a *bipartite matching problem* between semantic services. Before explaining the bipartite matching problem, we define a bipartite graph. In our case we have two bipartite graphs, one where the vertexes are the inputs of the services and the other where the vertexes are the outputs. In the case of matching between the inputs of the services (the process is the same for the outputs) the bipartite graph G=($\mathcal{I}_i \cup \mathcal{I}_j$, E) has a set of vertexes with service s_i inputs ($\mathcal{I}_i$) and the other set with s_j inputs ($\mathcal{I}_j$).

DEFINITION **5** *(Full connected Weighted Bipartite Graph for Service Inputs) A Service Inputs bipartite graph G=($\mathcal{I}$,E) is a graph whose vertexes can be divided into two disjoint subsets $\mathcal{I}_i \in s_i$ and $\mathcal{I}_j \in s_j$: $\mathcal{I}_i \cup \mathcal{I}_j = I \wedge \mathcal{I}_i \cap \mathcal{I}_j = \emptyset$. Each vertex from one of the subsets is connected to other vertex in the other subset e_k=(in_i,in_j) $\in E$, $in_i \in \mathcal{I}_i \wedge in_j \in I_j$. The weight of each edge ω_k is established with dom between service input concepts (in_i and in_j).*

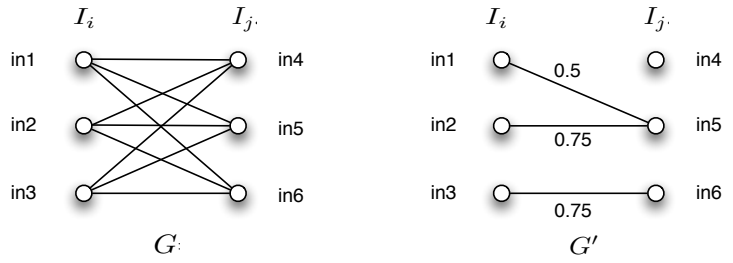
DEFINITION **6** *(Relaxed Weighted Matching Bipartite Graph for Service Inputs) Given a Service Inputs bipartite graph G=($\mathcal{I}_i \cup \mathcal{I}_j$,$E$), its matching G'=($\mathcal{I}_i \cup \mathcal{I}_j$,$E'$) $E' \subseteq E$ is a graph where all the vertexes of one set are connected with the other set of vertexes only with one edge. In this graph it we allow that edges share a vertex to give more flexibility to the matching. The sum of weights ($W_{G'}$) of the edges in the matching is maximized:*

$$W_{G'} = \frac{\sum_{\forall e_k \in E'} \omega_k}{|I_i|} \tag{4}$$

4 System Operations

4.1 Join

The process that an agent should follow to get into the $\mathcal{SDS}$ is as following (see Alg. 1): the agent a_i tries to establish a set of connections with other agents already present in the system. The number of connections that the agent is going to establish is generated by a random function which follows an exponential distribution. The idea is to generate a system with an exponential degree distribution to achieve the structure of a *preferential attachment network*[1]. A preferential attachment network it is characterized by a degree distribution which follows a power-law degree distribution, p(dg) $\propto dg^{\lambda}$, where $p(dg)$ indicates the probability to be connected to a node with degree dg. This means that there are some nodes have a high degree and the majority has a low degree. This structure ensures that the diameter of the network is $ln\ |\mathcal{A}|$, where $|\mathcal{A}|$ is the number of agents in the $\mathcal{SDS}$ [4]. This model is present in many *'online communities'* such



	in4	in5	in6
in1	0.25	0.5	0.25
in2	0	0.75	0
in3	0.5	0.25	0.75

$\text{match}(I_i,I_j) = (0.5+0.5+0.75)/3 = 0.58$

Fig. 1: Full connected Weighted Bipartite Graph for Service Inputs and Relaxed Weighted Matching Bipartite Graph for Service Inputs

as *WWW*, electronic mail or citation graphs [20]. These networks are the result of a growth process in which new nodes that join the system prefer to be connected to well connected nodes.

Once the agent knows the number of connections, it should decide which agents are going to be its neighbors. The probability of an agent a_i to establish a connection with agent a_j is directly proportional to the homophily degree between the agents, if the agents are more similar, they have more probability to be connected. This condition allows a new agent not only to establish 'short connections' between agents with similar roles and semantic services, but also between agents that are not similar ('long connections'). The idea of 'long connections' is to create short paths between groups of agents that do not offer similar services and reduce the number of hops needed to discover services. The probability to establish a connection between two agents Π_{a_i,a_j} in the system is based on the homophily degree between them:

$$\Pi_{a_i,a_j} = \mathcal{H}(a_i, a_j) \tag{5}$$

4.2 Leave

When an agent leave the system it could be for random failure in the network or deliberate attack or 'sabotage'. Periodically an agent sends a *keep alive* message to its neighbors. The agent will notice that one of its links is broken whether after sending a message, the time to receive an answer from the neighbor expires. In that case, the agent deletes the neighbor from its neighbor list and establishes a new link with other agent in the network to keep their degree (see Alg. 2 and Alg. 3).

Algorithm 1 Join, where a is the new agent and S the system

function Join(a, S)
$connections \leftarrow ExpRandom(\lambda)$
$connected \leftarrow False$
$dg \leftarrow 0$
while $\neg connected \wedge dg \leq connections$ **do**
  $a_r \leftarrow random(S)$
  if $\mathcal{H}(a, a_r) \geq UniRandom(0,1)$ **then**
    $\ell(a,a_r)$
    $a.dg \leftarrow a.dg+1$
    $updateNeighbors(a, a_r)$
    $connected \leftarrow True$
  end if
end while
end function

Algorithm 2 Leave, where a is the agent and S the system

function Leave(a, S)
local
for $a_i \in a.N$ **do**
  $removeLink(a, a_i)$
  $newLink(a_i,S)$
end for
end function

Algorithm 3 newLink, where a is the agent and S the system

function Link(a, S)
$connected \leftarrow False$
while $\neg connected$ **do**
  $a_r \leftarrow random(S)$
  if $sim(a, a_r) \geq UniRandom(0, 1)$ **then**
    $\ell(a,a_r)$
    $a_r.dg \leftarrow a_r.dg+1$
    $updateNeighbors(a, a_r)$
    $connected \leftarrow True$
  end if
end while
end function

4.3 Search

DEFINITION **7** *(Service discovery problem) Given a set of agents $\mathcal{A}$ situated in a $\mathcal{SDS}$= ($\mathcal{A}$,$\mathcal{L}$), the **service discovery problem** is defined as a probabilistic decision-making task in which an agent $a_i \in \mathcal{A}$ is looking for an agent $a_j \in \mathcal{A}$, which offers the required service s_t.*

The search process in the system is based only on the agent local knowledge. When the agent a_i is looking for an agent a_j which offers the required service s_t, a_i selects

which of its neighbors is the most appropriated to redirect the query instead of broadcast the query to all the neighborhood. In many networks which reflects power-law characteristics, the search is suggested to be based on degree. However, this makes that highly connected nodes could be overloaded with requests. In our proposal the selection of the most suitable agent is based on two criteria: *agent degree* and the *semantic similarity* between agents services and roles. This proposal is based on the algorithm presented in [5]. Until the target agent a_j is found, all future agents involve in the discovery process will make their decision similarly (see Alg. 4).

$$\Pi^{s_t}_{a_i a_j} = 1 - (1 - \Pi_{a_i a_j})^{dg} \tag{6}$$

Algorithm 4 Search where a_s is the source agent, s_t is the required service and S the system and Θ is the similarity threshold

```
function Search(a_s, s_t, S, Θ)
s←getService(a_s)
a←a_s
steps←0
while sim(s, s_t) ≥ Θ ∧ steps≤TTL do
  p_max←0
  for a_i ∈ a.𝒩 do
    dg ← a.dg
    s ← a.s
    p ← 1 − (1 − ℋ(a, a_i))^dg
    if p > p_max then
      p_max ← p
      a ← a_i
    end if
  end for
end while
return a
end function
```

5 Simulation Results

The test can be divided in two groups. The first group compares the performance of typical distributed search strategies (*degree*, *similarity*, *random*) to the proposal presented in this paper. The second test evaluates the fault tolerance of the $\mathcal{SDS}$ when relationships between agents in the system are broken randomly (an agent leaves the system) or following some patterns (which corresponds to deliberate failures 'sabotage').

5.1 System Characterization

The experiments have been done in a set of networks that simulate the $\mathcal{SDS}$ structure. These networks are preferential attachment networks and have generated as the result

of the *join* operation of agents. We have implemented two kind of networks: A where the agents have not roles and Network B where each agent plays a role. We consider 10 types of different roles. Each network is composed of 1000 agents with one semantic service each one. The services and roles have been assigned to the agents using a uniform distribution.

5.2 Performance

In this section we evaluate the *search* operation in our $\mathcal{SDS}$. Due to the similarities of our system and p2p systems, we compared the *search operation* to other typical search strategies used in p2p systems: *random*, *degree*, *similarity*, *similarity and degree*. We have analyzed the behavior of each strategy in 5000 searches in networks A and B.

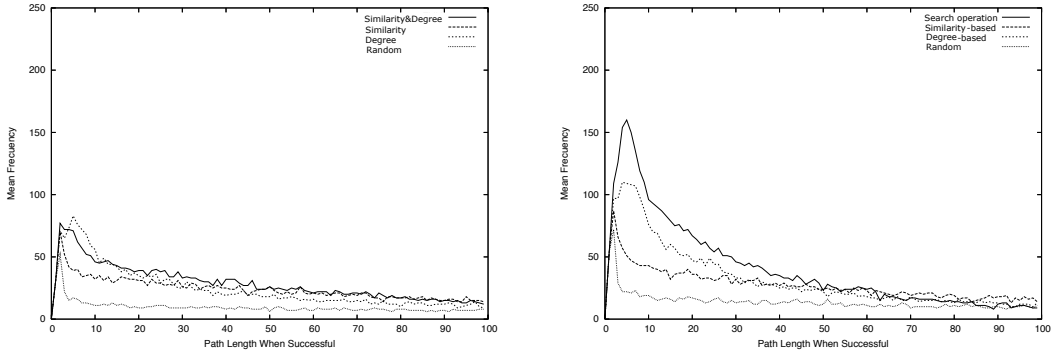
In figures 2a and 2b, the results obtained after the service search process are presented. We see that in general the strategies in a network with organizational information have a better performance than the same strategies in a network without this information. That shows that organizational information in the system can guide the search process better than the systems that only provide information related to the degree and services. Between all the strategies, the *search* operation that we present in this paper has a better performance than the others. This is because it considers, apart form the degree and semantic service information, the roles that agents play. This information reduces the set of possible agents suitable to offer the service.

An important parameter to consider in $\mathcal{SDS}$ is the number of steps to reach the target agent. Figure 3a shows the mean path length obtained with each strategy in networks with role information (B). In general, all the strategies return paths with more steps as the number of agents in the network grows. When the size of the network is over 700 agents, the path length does not increase significantly. This shows that the structure of the $\mathcal{SDS}$ is suitable for large-scale systems.

In Figure 3b the success rate of each search strategy in $\mathcal{SDS}$ is depicted. An obvious result is that as the system scale increases, the percentage successful searches decreases. The *search* operation presented here is the algorithm less influenced by the number of agents in the system. In general, the *search* operation in the 80% of searches finds a path between the source agent to the target agent.

5.3 Fault Tolerance

The last and very important check is the behavior of the $\mathcal{SDS}$ under failures. The problem appears when a broken link splits the system into two isolated parts, since some agents will no longer be reachable. To analyze it, agent failures have been modelled as a failure of all its connections. When some links are broken, an alternative path has to be found. For random failures (see Fig.4a and Fig.4b), it can be observed that when the number of deleted agents is from 10% to 30%, the path length increases, due to there are alternative paths (with more steps) to find the agent with the required service. When the number of deleted agents ranges from 30% to 50%, the network is divided in several isolated parts. Only the searches inside the isle will success, so the number of successful searches decreases and the path length decreases because the isle diameter are smaller.



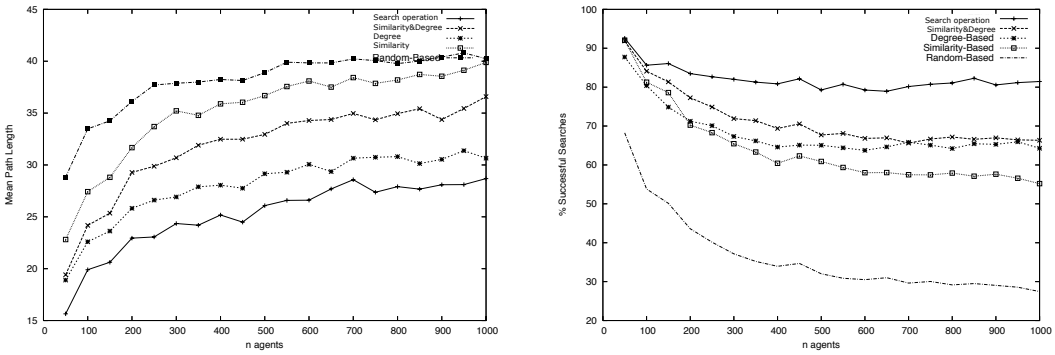
(a) (b)

Fig. 2: Search performance without/with role information

An interesting case is what happens when a deliberate failure is provoked. In the case of systems that follow a power-law, the worst case occurs when agents with highest degree (hubs) are disconnected. Figure 5a and 5b shows how *'sabotage'* affects the performance of the search process. In this case, the path length increases due to only a few highly connected hubs have been deleted and an alternative path exists. The performance attending the number of successful searches decreases considerably as the number of deleted hub increases.

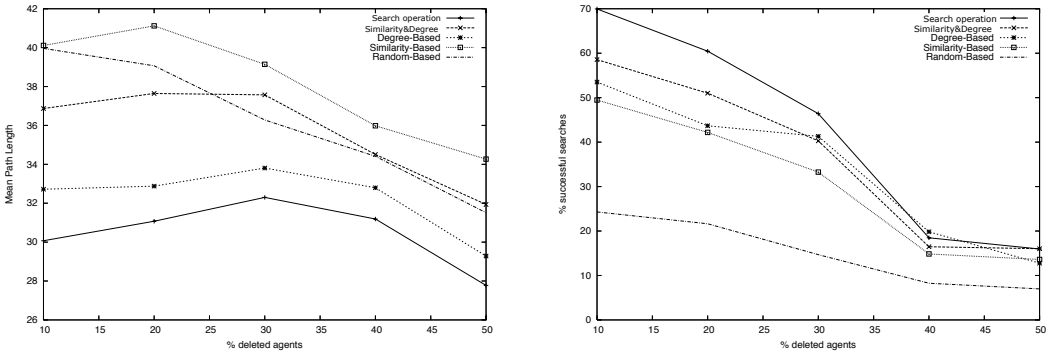
6 Conclusions

The aim of this work is to provide an alternative to traditional approaches that deal with the service discovery task in large-scale open SOMAS. Our proposal tries to overcome drawbacks present in other centralized (bottlenecks, complexity, huge amount of memory needed, global knowledge) and distributed (network traffic, congestion, coordination effort, data consistency between distributed registries, update data) discovery approaches. We consider that structures used in social networks facilitate the task of locating agent services in a few steps using only local information. For that reason we investigate the use of social networks as underlying structure of a service discovery system. This structure is based on the concept of similarity between individuals, considering organization role and services, and uses semantics to calculate this similarity. Furthermore, we provide several operations for the agents to be part of the system. An evaluation of the search functionality compared to other traditional p2p strategies is also provided. The behavior of the system under failure and 'sabotage' circumstances have been also evaluated. The results of the experiments show that the system is robust under failure and that the search functionality performs well.



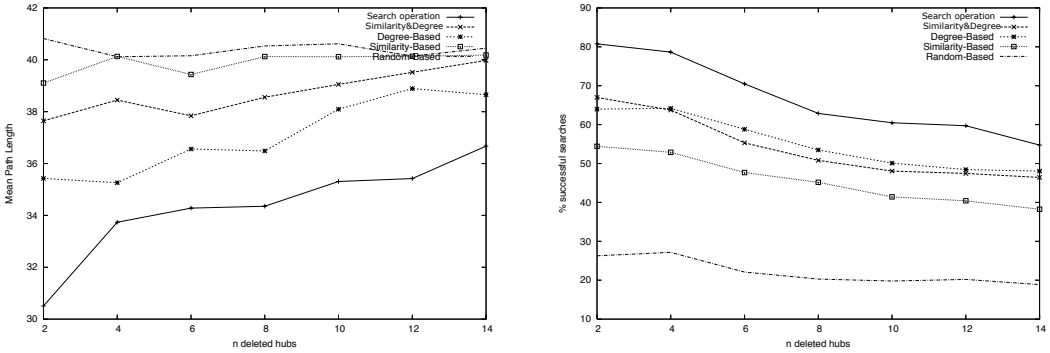
(a) (b)

Fig. 3: Mean path length and success



(a) (b)

Fig. 4: Mean path length and success with random failures



(a) (b)

Fig. 5: Mean path length and success under 'sabotage' conditions

Acknowledgment

This work is supported by TIN2009-13839-C03-01 and TIN2008-04446 projects, CONSOLIDER-INGENIO 2010 under grant CSD2007-00022, FPU grant AP-2008-00601 awarded to E. del Val.

References

1. A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.
2. E. Bircher and T. Braun. *An Agent-Based Architecture for Service Discovery and Negotiation in Wireless Networks*. 2004.
3. Basit Chaudhry, Chris Marton, and Hana Shepherd. Homophily and structure in multiplex networks.
4. Reuven Cohen and Shlomo Havlin. Scale-free networks are ultrasmall. *Phys. Rev. Lett.*, 90(5):058701, Feb 2003.
5. Özgür Şimşek and Jensen. Navigating networks by using homophily and degree. *Proceedings of the National Academy of Sciences*, 2008.
6. J. Dang and M. Hungs. *Concurrent Multiple-Issue Negotiation for Internet-Based Services*. 2006.
7. S. Jha, P. Chalasani, O. Shehory, and K. Sycara. A formal treatment of distributed matchmaking. In *Proc. of the 2nd Int. Conference on Autonomous Agents*, number Vol.3, pages 457–458, 1998.
8. M. Klusch, B. Fries, and K. Sycara. Automated semantic web service discovery with owls-mx. In *Proceedings of 5th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Hakodate, Japan, 2006.
9. M McPherson, Lynn Smith-Lovin, and James Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 2001.

10. M. Moore and T. Suda. A decentralized and self-organizing discovery mechanism. In *AINS*, 2002.
11. S. Mullender and P. Vitanyi. *Distributed Match-Making*. 1988.
12. E. Ogston and S. Vassiliadis. Local distributed agent matchmaking. In *Proceedings of the 9th International Conference on Cooperative Information Systems*, 2001.
13. E. Ogston and S. Vassiliadis. Matchmaking among minimal agents without a facilitator. In *AAMAS*, 2001.
14. A. Ouksel, Y. Babad, and T. Tesch. Matchmaking software agents in b2b markets. In *HICSS'04*, 2004.
15. K. Sigdel, K. Bertels, B. Pourebrahimi, S. Vassiliadis, and L.S. Shuai. A framework for adaptive matchmaking in distributed computing. In *In proceeding of GRID Workshop*, 2005.
16. K. Sycara and M. Klusch. Brokering and matchmaking for coordination of agent societies: A survey. *Coordination of Internet Agents: Models, Technologies and Applications*, pages 197–224, 2001.
17. Katia Sycara, Matthias Klusch, Seth Wido, and Jianguo Lu. Dynamic service matchmaking among agents in open information environments. *SIGMOD Record*, 28:47–53, 1999.
18. Jeffrey Travers and Stanley Milgram. An experimental study of the small world problem. *Sociometry*, 32, 1969.
19. Yamini Upadrashta, Julita Vassileva, and Winfried Grassmann. Social networks in peer-to-peer systems. paper presented at the. In *38th Hawaii International Conference on System Sciences*, pages 3–6, 2005.
20. Alexei Vázquez. Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations. *Physical Review E*, 67(5), May 2003.
21. Watts, Dodds, and Newman. Identity and search in social networks, 2002.
22. Hui Zhang, Ashish Goel, and Ramesh Govindan. Using the small-world model to improve freenet performance, 2002.