Relationship Networks as a Survivable and Adaptive Mechanism for Grid Resource Location

Lei Gao¹ and Yongsheng Ding^{1,2,*}

¹ College of Information Sciences and Technology

² Engineering Research Center of Digitized Textile & Fashion Technology,
Ministry of Education, Donghua University,
Shanghai 200051, P. R. China
ysding@dhu.edu.cn

Abstract. Grid technologies have emerged to enable large-scale flexible resource sharing among virtual organizations. Resource location of next generation grid systems needs to be fully decentralized, more importantly, be survivable and adaptive to uncertain factors in grid environments. This paper applies the concepts and principles of relationship networks in human society to design a dynamic location approach. The simulation results have proved that our approach can: form relationship clusters to improve the location performance effectively; well adapt to different resource distributions and user request patterns; and survive from the changes of dynamic environments such as partial failure of agents.

1 Introduction

Grid systems have evolved over nearly a decade to enable large-scale flexible resource sharing among dynamic virtual organizations (VOs) [1]. The next generation grid systems [2] will exist in an unstable environment (where a large number of nodes join and leave the grids frequently). When this happens, new services that can survive and adapt to a user's environment have to be developed.

Resource location means that given a description of desired resources, a location mechanism can return a set of contact addresses of resources that match the description in a wide-area grid environment. As a critical activity in grid systems, it inevitably faces the challenges brought by next generation grid systems. It needs to be fully decentralized, more importantly, be survivable and adaptive to uncertain factors (such as different resource distribution and variable user request patterns) in grid environments. Unfortunately, existing grid location technologies such as Globus's MDS have not addressed the survivability and adaptability of dynamic location. So, it is valuable for us to look at the location approach in other fields. In the real life, it is a natural way for us to depend on our relationship networks for finding relevant information: our acquaintances, our acquaintances' acquaintances, and so on. These relationship entities and relationships among them can provide a fully decentralized, naturally adaptive and effective way to gather and disseminate the resource information.

Inspired by this, we study an approach that places the intelligence on the grid entities, enabling the users to locate desirable resources based on relationship networks. The work in this paper is the further results of our previous research [3], which viewed a grid as interacting agents and applied some key mechanisms of natural ecosystems to build a grid middleware named Ecological Network-based Grid Middleware (ENGM). The rest of this paper is organized as follows. Based on ENGM platform, Section 2 presents a basic framework for resource location. Relationship networks-based location mechanism is demonstrated in Section 3, which is followed by a simulation and corresponding analysis in Section 4. Section 5 briefly describes related work in P2P field and concludes our efforts.

2 A Wide-Area Resource Location Framework

2.1 Design of Ecological Network-Based Grid Middleware

From the bottom up, the architecture of ENGM system is presented as three-layers: Heterogeneous and Distributed Resources layer, ENGM layer, and Grid Applications for VOs layer. (1) *Heterogeneous and Distributed Resources* consist of different types of resources available from multiple service providers distributed in grids. Via a java virtual machine, an ENGM platform can run on a heterogeneous distributed system that established in a network node. (2) *ENGM* provides the services support a common set of applications in distributed network environments. It is made up by ENGM functional modules, ENGM core services, grid agent survivable environment, and emergent grid common service. (a) ENGM functional modules deal with the management of networks and systems. (b) ENGM core services layer provides a set of general-purpose runtime services that are frequently used by agents such as niche sensing service. (c) Grid agent survivable environment is runtime environment for deploying and executing agents that are characterized by high demand for computing, storage and network bandwidth requirements. (d) Emergent grid common services are kernel of middleware and responsible for resource allocation, information service, task assignment, and so on. These common services are emerged from the autonomous agents. (3) *Grid Applications for VOs* use developing kits and organize certain agents and common services automatically for special purpose applications.

2.2 Framework for Resource Location

Every grid resource supplier maybe has one or more servers that store and provide access to their resources. We regard these servers as nodes. An ENGM platform-based wide-area framework is set up on these nodes. Furthermore, some nodes can form a *niche* that refers to a logically defined area where agents can learn their surrounding environment. For instance, an agent may sense which agents are in the niche, what services they perform, and which resources it can access. Here, a niche can be regarded as a VO.

Agents in the framework fall into two categories: grid user agents (GUAs) and grid service agents (GSAs). A GUA represents a kind of user tasks. GSAs

are used to comprise the main components of ENGM, such as grid information service agent (GISA). Agents are represented on their unit functions. Every unit function is defined as a metadata structure that enables collaboration among agents. The metadata of an agent consists of *agentID* (a global unique identifier of an agent), *agentAddress* (location of this agent), *serviceType* (service type of this agent), *serviceDescription* (description of service information), and *relationshipDescription* (information about its acquaintances, agents know each other are called acquaintances). Besides basic information of its acquaintances, *relationshipDescription* of an agent still consists *TrustCredit* (indication of reliability to acquaintance) and *collaborationRecord* (collaboration history records).

A GUA makes search instructions from a user into request messages and send them to GISA in local niche. The GISA responds with the matched resource descriptions if it has them locally, otherwise it forwards the requests to another GISA outside the niche until the request hit returns or request time exhausts. Note that there is only one GISA in a niche. Each GISA only knows about a subset of all GISAs, then it can select one to forward, and the selected one delivers the request to one of its acquaintances, and so on. Broadcast-type search strategies will lead to high bandwidth cost, scalability problem and congestion constraints. Instead of broadcasting a request to a large fraction of the network, a request is only passed onto one node at each step in our framework.

3 A Relationship Network-Based Location Approach

A three-phased location approach based on relationship networks is given out as follows: GISA relationship construction, request processing strategy and trust-based reconstruction of relationship. GISA relationship construction is responsible for collecting and updating information about the currently participating GISAs and for the forming relationship networks. Request processing strategy performs the search itself. Trust-based reconstruction of relationship makes the necessary preparations for a more efficient search.

1. GISA relationship construction. A GISA joins the grid by contacting a member GISA. Contact addresses of member GISA can be learned through sensing mechanism integrated in ENGM [3]. Once a nearby GISA is found, a relationship can be established. GISAs can be aware of their acquaintances and update the relationships of changed GISA.

2. Request processing strategy. A GUA receives a search instruction sent by a grid user in local niche, and then it creates a request message. A request message contains the message ID, information on request originator and the forwarder, a set of parameters to specify target resource attributes and a set of weights to describe the importance degree of each request resource attribute. When a GISA receive a request message, it first examines the message ID to check whether it has seen the message in the past. If has, it discards the message. Otherwise, it evaluates the request by its resource matching model and request forwarding

model. Then, the GISA decides to respond the request with its local resource, forward the request to its acquaintances, or reject the request. A GISA can both respond the request and forward the search request. Next, we give out resource matching model and request forwarding model as follows. The parameters used to describe the target resources are defined as the form of vector space. Grid resources provided in a peer are modeled as an attribute vector. The resources in the request message are also modeled as an attribute vector.

(a) Resource matching model: Given a request vector $R = \langle r_1, r_2, \dots, r_n \rangle$, a weight vector $W = \langle w_1, w_2, \dots, w_n \rangle$ (where $\sum_{t=1}^n w_t = 1$) that indicates the importance degree of each request attribute about R , and a grid resource vector $G = \langle g_1, g_2, \dots, g_n \rangle$, the matching strength between and is defined as: $MS_{vec}(R, W, G) = \sum_{t=1}^n MS_{sin}(r_t, g_t) \cdot w_t$. If g_t satisfies r_t , $MS_{sin}(r_t, g_t) = 1$. Otherwise, $MS_{sin}(r_t, g_t) = 0$. For example, a single request attribute value r_t represents *more than 128 MB of available memory* and a single resource attribute value g_t stands for *256 MB of available memory*. So, we have $MS_{sin}(r_t, g_t) = 1$. In addition, the request originator will specify a threshold for resource matching denoted as ω_i ($0 \leq \omega_i \leq 1$). If $MS_{vec}(R, W, G) \geq \omega_i$, there is a matching between R and G .

(b) Request forwarding model: Assume that A_i is an any GISA and A_{i_j} is its one of acquaintances. The forwarding strength that A_i imposes on A_{i_j} is relative to *trustCredit* and *collaborationRecord*. The optimal matching strength on requests in *collaborationRecord* is defined as $MS_{opt}(R, W, R_{i_j}^k) = \max_{k=1}^m MS_{vec}(R, W, R_{i_j}^k) \cdot \delta_{i_j}^k$, where R and W have the same meaning in resource matching model, $R_{i_j}^k = \langle r_{i_j}^{k1}, r_{i_j}^{k2}, \dots, r_{i_j}^{kn} \rangle$ is a request vector A_{i_j} previously answered and/or forwarded and $\delta_{i_j}^k$ is a parameter in $[-\frac{1}{2}, 1]$ that indicates the user evaluation of location with $R_{i_j}^k$. Given a weight η to *trustCredit* and *collaborationRecord*, the forwarding strength that A_i imposes on A_{i_j} that is defined as $FS_{vec}(R, W, A_i, A_{i_j}) = \eta \cdot trust_{i,i_j} + (1 - \eta) \cdot MS_{opt}(R, W, R_{i_j}^k)$, where $trust_{i,i_j}$ is a number between $[0, 1]$ that represents *trustCredit* value that A_i has on A_{i_j} . If optimal forwarding strength $FS_{opt}(R, W, A_i, A_{i_k}) = \max_{k=1}^n FS_{vec}(R, W, A_i, A_{i_k})$ is obtained, a decision can be easily made by A_i regarding which acquaintance is more likely to lead towards desired resources. A_i can specify a threshold for request forwarding θ_i ($0 \leq \theta_i \leq 1$ and usually $\theta_i \leq \omega_i$). If $FS_{opt}(R, W, A_i, A_{i_k}) \geq \theta_i$, A_i will forward the request message. What's more, A_i will discard the message if it neither responses the request with its local resources nor forwards the request.

3. Trust-based reconstruction of relationship. This dynamic mechanism on reconstruction of relationship, can contribute to resource location. It updates and strengthens relationship networks by establishing and changing the *trustCredit* values among the GISAs. Reliability is expressed through a *trustCredit* value with which each GISA labels its acquaintances. On receiving a request hit, the request originator returns a defray message including a collaboration record and a credit that stands for user evaluation of request hit based on service defraying model. A credit could be a reward or a penalty, which indicates the degree of

its preference of the received request hit. This message is propagated through the same path where the location request has been originally forwarded. When an intermediate GISA on the path receives message, it adjusts the *trustCredit* value of the relationship that has been used to forward the original location request. *trustCredit* value is increased for a reward (i.e., high degree of the request originator's preference), and is decreased for a penalty (i.e., low degree of the request originator's preference). Service defraying model is given out as follows. Given a credit γ ($-\frac{1}{2} \leq \gamma \leq 1$) which stands for a reward or penalty contained in a defray message, A_i updates the *trustCredit* value of $A_{i,j}$ using following method: if $\gamma \geq 0$, $trust_{i,i,j}^+ = trust_{i,i,j}^- (1 - \gamma^2) + \gamma^2$; if $\gamma < 0$, $trust_{i,i,j}^+ = trust_{i,i,j}^- (2 - \frac{1}{1+\gamma})$, where $trust_{i,i,j}^-$ is a number in $[0, 1]$ that represents *trustCredit* value which A_i imposes on $A_{i,j}$ before updating. Correspondingly, $trust_{i,i,j}^+$ is the *trustCredit* value after updating. We have chosen the above method with the purpose of remaking ratings rise slowly and fall quickly.

To evaluate the approach, a simulator is developed upon ENGM platform. The simulation setup, results and analysis are described in next section.

4 Simulation Study

4.1 Simulation Setup

1. Initial relationship network topology. Define that a cycle starts from a request message sent by a GISA and ends till all the relevant messages disappear in the system, and 100 cycles is a generation in the simulation. We adopt the generation method of network topology proposed in [4] to establish relationship networks, whose nodes are GISAs and whose edges connect pairs of GISAs those know each other. The generated topology follows power law. It is pointed out emphatically that the statistic results from domain level and router level also follow above power law rule, and GISAs just play their roles on these two levels.

2. Resource distribution. Two involved thresholds, θ_1 and ω_i are predefined as fixed values. We make a set of common resources (contains 20000 different resource vectors) and a set of new-type resources (contains 2000 different resource vectors that are completely different from common resources). We experiment on two strategies. (a) Balanced distribution strategy: initially each GISA provides 3-5 resource vectors, which are randomly picked out from the common resource set. (b) Unbalanced distribution strategy: a few number of GISAs provide most of the resource vectors from common resource set, while the large number of GISAs share the small part of the resources.

3. User requests and user evaluation. Requests are initiated at a fixed percentage of randomly selected GISAs and contain resource vectors. The resource attributes of each request have the same weight. Two user request patterns are studied: (a) unbiased user request scenario (requesting all the vectors randomly) and (b) biased user request scenario (using a great probability to request a small part of and special vectors available in the simulation network). We randomly assign a fixed threshold to each GISA for indicating the satisfaction level of a

request originator. On receiving a request hit, the request originator will examine the match degree of responded resources. If the matched result is no greater than the threshold, the originator returns the rate message with a credit 0.1 to resource responder; otherwise, the credit will be -0.1. The GISAs in the credit-propagated chain will be given the same credit.

4.2 Simulation Results and Analysis

The simulation evaluates the survivability and adaptability from 3 aspects: resource distributions, user request patterns, and the reliability of GISAs. Also, we have a discussion of the performance distinctions affected by different η (0, 0.25, 0.5, 0.75, 1). Considering the randomness, we repeat the experiments multiple times. The results given out are average values of measurements.

A. Effects of Resource Distribution. Fig. 1 shows the performance results with different η in balanced and unbalanced resource environments. At the beginning of simulation, relationships are random, and location performs poorly. Many hops need to be visited to hit the target GISA. As more simulation cycles elapse, GISA gradually obtain many relationships similar to themselves, leading to improved performance in location process. We find that such improvement results from the clustering of vector-matched GISAs. The clusters have not formed in the process of random forwarding simulation, and the location will go aimlessly till it meets the matched GISA. So the random forwarding has the lowest efficiency, though it has lowest cost (no need to store any information in GISAs). The comparison between the results has proved that our approach can form clusters and improve the location performance adaptively. The condition with $\eta = 1$ (locate totally according to *trustCredit* value) takes the minimum overhead costs, but it is with the lowest location efficiency in this scheme. While $\eta = 0$ (locate totally according to *collaborationRecord*), it is with the highest location efficiency, but not greater consumption. In unbalanced environments, the efficiency of the *trustCredit*-based location (with minor value of η) is much better than that in balanced environments. When a few number of GISAs hold most of the resources and so have rather more relationships, it is easier for them to form clusters. Compared to in a balanced environment, in a highly unbalanced environment, a GISA that had already responded some requests is more likely to have answers to other requests as well.

B. Influence of User Request Scenario. Comparing Fig. 1 with Fig. 2, we can see that, as a whole, the performance of our approach is better within biased request scenario than unbiased request scenario. Especially, the performance with $\eta = 1$ has evidently been improved, which means that location fully according to *trustCredit* value can adaptive to the relationship networks very well and can better support the special user request scenario. It shows very good efficiency in the unbalanced environments, response latency in this environment is almost half of that in the balanced environments. Similarly, in the unbalanced environments, the performance has been improved comparing to the balanced environments as η equals 0.75, 0.5 and 0.25. Relative to the unbalanced envi-

Fig. 1. Average number of hops per request as a function of simulation time, within unbiased user request scenario and 5000 GISAs, for two resource distribution (Left: balanced. Right: unbalanced)

ronments, when $\eta = 0$, the performance is better in the balanced environments, which demonstrates that the fair distribution of information can help to find a useful GISA as there is no relevant previous information of the current request.

C. Reliability of GISAs. We adopt the same simulation setup as the right of Fig. 2 (unbalanced and biased) to conduct a comparison when the GISAs are

Fig. 2. Average number of hops per request as a function of simulation time, within biased user request scenario and 5000 GISAs in two environments (Left: balanced. Right: unbalanced)

not always available. At each generation, we chose 1% of GISAs randomly in the simulation and set them *unavailable* for location. The concept of difference $D = \frac{N_{hop}^{unreliable} - N_{hop}^{reliable}}{N_{hop}^{reliable}}$ is defined as a measure of the adaptability to unreliable GISAs, where $N_{hop}^{reliable}$ and $N_{hop}^{unreliable}$ are respectively the average number of hops per request for a certain number of GISAs in a static (all reliable) and dynamic scenario. The performance is compared between these two scenarios to determine how the approach is impacted by the unreliable GISAs. The average differences (equals to $\frac{1}{N} \sum_{i=1}^N D$, where $N = 5$ stands for 5 experiment points on the number of GISAs) with each η ($\eta = 0, 0.25, 0.5, 0.75, 1$) for all the numbers of GISAs are respectively 0.0373, 0.039, 0.0403, 0.0302, and 0.0429. Each average difference is very small, which suggests that the approach could be survivable to unreliable GISAs.

5 Related Work and Conclusions

Peer-to-peer technology are a very popular technology to deal with highly variable distributed environment. A distributed search approach in Gnutella is broadcasting a request to all peers in an unreasonable manner. CAN, Chord, and Pastry build distributed hashing structures that provide good scalability and search performance. However, the routing table in them for each node is fixed and thus the network is not reconfigurable. The location mechanism of Freenet is based on usage patterns, which make the popular files closer to users, while discard eventually the least popular files. Our design also proceed from this user-centered intention. However, it does not view that unpopular data is unimportant data.

This paper applies the concepts of relationship networks to design a dynamic location approach. The performance comparison between the approach and a random forwarding has proved that our approach can form relationship clusters and improve the location performance. The simulation results demonstrate the approach can well adapt and survive to different resource distributions, user request patterns, even the partial failure of GISAs. We also give out the performance distinctions of different η to help users better understand the tradeoffs between overhead costs and performance, and chose the preferable parameter.

Acknowledgments

This work is supported by the National Nature Science Foundation of China (No. 60474037 and 60004006), Program for New Century Excellent Talents in University, and Specialized Research Fund for the Doctoral Program of Higher Education from Educational Committee of China (No. 20030255009).

References

1. Foster, I., Kesselman, C., and Tuecke, S.: The anatomy of the grid: enabling scalable virtual organizations. *Int. J. Supercomputers Applications*. **15** (2001) 205–220
2. Cannataro, M. and Talia, T.: Semantics and knowledge grids: building the next-generation grid. *IEEE Intelligent Systems*. **19** (2004) 56–63
3. Gao, L., Ding, Y.-S., and Ren, L.-H.: A novel ecological network-based computation platform as grid middleware system. *Int. J. Intelligent Systems*. **19** (2004) 859–884
4. Barabasi, A.-L. and Albert, R.: Emergence of scaling in random networks. *Science*. **286** (1999) 509–512