World of Computer Science and Information Technology Journal (WCSIT)
ISSN: 2221-0741
Vol. 2, No. 7, 220-224, 2012

A New Approach For Flexible Matching of Grid Resources

Mehdi Bayat
Computer Engineering Department
Iran University of Science and Technology
Tehran, Iran

Morteza Analoui
Computer Engineering Department
Iran University of Science and Technology
Tehran, Iran

Abstract— Grid environment is an infrastructure in which many heterogeneous resources participate in solving large scale and high computational problems on the internet. Heterogeneity and large number of resources makes resource matching an important issue in the field of grid networks. This paper presents a new approach for grid resource matching. This paper proposes an approach that utilizes ontology to present the semantic relations of the environment and discover the resources that are most relevant to the request. Our method can provide approximate matches if no exact match exists for the given request. The evaluation results show that our method is more effective in resource matching compared with other mechanisms such as UDDI and OWL-S.

Keywords- Grid computing; Resource Discovery; ontology; matchmaking.

I. Introduction

Grid computing (GC) has been defined in a number of different ways; however, there is generally a consensus that GC involves the integration of compute resources that offer performance unattainable by any single machine [1]. In other word, a computational grid can be viewed as sort of “metacomputer”, whose software and hardware resources are distributed over disparate networked machines (*nodes*) [2]. Typical Grid infrastructures should include (at a minimum): knowledge management resources, an integrator such as open grid service architecture (OSGA), data and computing resources, and the appropriate network to accommodate interactions [1]. Computational grids may span domains of different dimensions, starting from local grids, where the nodes belong to a single organization via a LAN connection, to global grids, where the nodes are owned by different organizations and linked via Internet. In both cases a special software (the so called “*grid middleware*”, GM) allows to access the dispersed resources as if they were local [2].

Grid technologies enable sharing, exchange, discovery, selection, and aggregation of geographically distributed or Internet-wide heterogeneous resources (e.g. sensors, computers, databases [3], workstations, clusters, and mainframes) with various individual properties (e.g. main memory, CPU speed, bandwidth, virtual memory, hard disk, operating system, CPU vender, number of CPU elements, and etc) where they can independently join and leave the grid environment [4]. In large scale Grid environments, resource discovery is a challenging task due to a potentially large number of resources, users and considerable heterogeneity in resource types and user requests [5]. One of the first approaches to service discovery problem is UDDI. UDDI is an industrial initiative whose goal is to create an internet wide registry of web services. UDDI supports the registration of physical attributes of services (such as name, address and service they provide) via a construct called Tmodel. A Tmodel is a form of meta data that provides a reference system for information about services. Since search in UDDI is restricted to keyword matching, no form of inference or flexible match between keywords can be performed [6]. UDDI has been utilized by [7] and [8] for grid service discovery. As expected these methods don’t show good effectiveness in service discovery because UDDI provides poor search facilities.

Recently with the popularity of semantic web technologies, there has been an increased interest in the use of ontology for service descriptions and the application of reasoning mechanisms to support discovery and matching [9]. Paolucci et al. [6] has proposed a semantic matching approach by combining DAML-S and UDDI. DAML-S/UDDI matchmaker supports flexible semantic matching between advertisements and requests on the basis of the ontology available to the services and the matching engine. This work has been utilized in many other researches to provide semantic based methods for web service discovery and grid service discovery. Bandara et al. [9] has used ontology to describe the services and provide a semantic matching solution. This paper divides all possible properties occurring in the individual requirements of a resource description, into three groups and explains how to determine similarity within each of these property types during the matching process. It has utilized fuzzy logic to calculate the similarity degree between two properties of numeric type.

Pukkasenung et al. [10] has introduced a hybrid matching based on semantic distance and cosine law for web service discovery. It utilizes ontology to calculate semantic distance of two concepts. This matching method also inspires the degrees of match presented in [6], which defines five levels of matching for the request and the service advertisement. It considers the request and the service advertisement as two vectors and uses cosine law to calculate the angle between them. It uses the obtained cosine value to classify a match into one of the five defined levels of matching.

Another school of thought that more recently has been utilized in grid resource discovery is Rough set theory. Rough set theory, proposed by Pawlak in 1982 [11], is a mathematical method to deal with uncertainty and vagueness in data and knowledge discovery. Uncertainty of service properties exists when matching services. "An uncertain property is defined as a service property that is explicitly used by one advertised service but does not appear in another service advertisement that belongs to the same service category" [12]. To face the problem of uncertainty of service properties, Yu et al. [13] has proposed an algorithm named RSSM. RSSM uses rough set theory to reduce dependent properties of advertised services and in this way remove some uncertain properties that exist among the reduced properties. Li et al. [12] presents a rough set-based search engine for grid service discovery called ROSSE. ROSSE is an improved version of RSSM, the difference lies in the formula of match degree between request property and advertised service property. Ataollahi et al. has introduced three rough set-based grid resource discovery methods namely CRSRD [4], DRSRD [14] and FDRSRD [15]. CRSRD has the same approach as RSSM, the difference is in the implementation details and DRSRD is the next version of CRSRD. DRSRD uses dynamic rough set presented in [16] and selects an optimum set of resources which are most likely to satisfy the requested service and then applies CRSRD on this selected optimum set of resources. FDRSRD is the same as DRSRD except that it utilizes fuzzy logic to calculate match degree and rank the resources.

The remainder of the paper is organized as follows: Section II is a description of our proposed method. In Section III we compare our method with UDDI and OWL-S mechanisms in terms of effectiveness in resource matching. Finally, we conclude this analysis in section IV.

II. RESOURCE DISCOVERY

In this section we present our matchmaking algorithm. Suppose a request R is received and a repository of advertised resources is available. We match the request R with each advertised resource and assign a score to each resource in the repository. Then sort the advertised resources based on their score in decreasing order. The rank of each available resource shows how good it can satisfy the request. Each property of the advertised resource compares with the corresponding property in the request to determine the score that the advertised resource gets for satisfying the request. The score given to each advertised resource is a quantitative measure so we have to express the result of matching each advertised resource property with the corresponding request property in quantity. The result of matching two numeric-property types can be shown in quantity using their value and mathematical operations. But the result of matching two non-numeric property types has to be expressed as a quantitative value. Therefore we distinguish between two types of properties occurring in the individual requirements of a resource description for the purpose of matching. The way we determine similarity within each of these types during the matching process, are discussed below:

A. Numeric property type

As discussed this type of properties can be matched and the result can easily be expressed in quantity using mathematical operations. Khanli et al. [17] has proposed a grid architecture called Grid-JQA and a matching method has been presented for Grid-JQA. We utilize that matching method for numeric property type's matchmaking in our algorithm. Let

P_R be a property used in a service query and

P_A be a property used in a service advertisement

Satisfaction factor of P_A for P_R is defined as:

$$SF = \frac{P_A}{P_R} \tag{1}$$

B. Non-numeric property type

The calculation of the score which an advertised resource gets for meeting the request, requires quantifying the result of matching two non-numeric properties. The simplest method for matching this type of properties is keyword matching. The keyword matching method only determines two degree of match: exact match and fail. To utilize a more flexible matching method that recognizes an approximate match between properties and present a softer definition of matching degrees, the semantic based methods could be an alternative to keyword matching.

Bandara et al. [9] has proposed a semantic based mechanism that using ontology and taxonomic relations of different concepts of the occurring properties of resources, defines five levels of matching between an advertised property and the corresponding property of the request. Assuming P_R is a property of the request and P_A is the corresponding property of an advertised resource; the possible taxonomic relations and the satisfaction factor assigned in each case are as follows:

For the two cases when P_A is a super concept of P_R and when P_R and P_A intersect; the similarity between the two concepts will be a value between 0 and 1. For the two cases when P_R is a super concept of P_A or when P_R and P_A are an exact match; the similarity between the two concepts will be 1 and if it's none of the above mentioned cases the satisfaction factor will be 0. Actually this approach is based on the probability of satisfying the given requirement. i.e. given that what is available is P_A , we have to judge the likelihood that it is also a P_R. Hence the satisfaction factor for two concepts P_R and P_A can be determined as:

$$SF(P_R, P_A) = \begin{cases} 1 & \text{if } P_R \text{ and } P_A \text{ are an exact match} \\ \dfrac{|A(P_A)|}{|A(P_R)|} & \text{if } P_A \text{ is a super concept of } P_R \\ 1 & \text{if } P_R \text{ is a super concept of } P_A \\ \dfrac{|A(P_A) \cap A(P_R)|}{|A(P_R)|} & \text{if } P_A \text{ and } P_R \text{ intersect} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Where $A(P)$ denotes the set of super class of a class P .

C. *Calculation of the appropriateness score of a resource for a request*

To calculate the overall score an advertised resource gets for satisfying the request, we use the satisfaction factor that each of its properties gets during the matching process. We also involve the priority of property in our calculations by giving a service requester the option of placing priorities/weights on the specified attributes of the service request. Let

R be the request for resource

A be the advertised resource we are matching the request R

P_{A_i} be the property number i in the advertised resource (A) properties set

P_{R_i} be the property number i in the request (R) properties set

W_i be the weight of property number i with respect to other properties

N be number of the request properties

C be the cost of utilizing the resource

SS be the overall satisfaction score the resource gets for meeting the request

Then the $SS(R, A)$ can be computed using the following:

$$SS(R, A) = \frac{\sum_{i=1}^{N} W_i \times SF_i}{C} \quad (3)$$

Where the weights of properties are scaled between 0 and 1:

$$\sum_{i=1}^{N} W_i = 1 \quad (4)$$

If the score that a resource gets is equal to 1 this means the resource matches the request on average based on results of matching their properties, but it doesn't mean each resource property meets the needs of the corresponding property of the request necessarily. If the score that a resource gets is greater than 1 this means the resource capabilities are higher than the request needs on average based on results of matching their properties, but it still doesn't mean each resource property meets the needs of the corresponding property of the request necessarily. If the score that a resource gets is less than 1 this means the resource capabilities are lower than the request needs on average based on results of matching their properties, but we know that definitely at least one of the resource properties doesn't meet the needs of the corresponding property of the request.

It is worth mentioning that our method assumes that resource advertisements and resource requests use consistent properties to describe relevant resources.

D. *Experimental Results*

In this section we compare our approach with UDDI keyword matching and the OWL-S matching [6], from the aspect of accuracy in resource matching. Because our method can not deal with the uncertainty of properties, we don't compare our approach with rough set based resource matching methods. In our simulations we assume a resource repository of six resources as shown in Table I. Each resource has four properties namely the number of CPU cores, the CPU clock, RAM size, the Disk capacity and operating system respectively. The weights of five properties are equal. We also assume that the range of each property is as bellow:

Property 1 :{ 1, 2, 4}

Property 2 :{ 0.133, 0.166, 0.233, 0.3, 0.333, 0.45, 0.533, 0.733, 1, 1.33, 1.4, 1.7, 1.8, 2, 2.2, 2.4, 2.53, 2.66, 2.83, 3, 3.2, 3.4, 3.6}

Property 3 :{ 32, 64, 128, 256, 384, 512, 768, 1024, 1536, 2048, 2560, 3072, 3584, 4096}

Property 4 :{ 5, 10, 20, 30, 40, 60, 80, 120, 160, 200, 250, 320, 400, 500, 750, 1000}

Property 5 :{ Unix, Windows, Linux, BSD, Win95, Win98, Win2000, WinXP-32, WinXP-64, Win-Vista-32, Win-Vista-64, Win7-32, Win7-64}

TABLE I. RESOURCE REPOSITORY

Resource No.	CPU Core	CPU Clock	RAM Size	Disk Capacity	OS
1	4	2.66	4096	1000	Win7-64
2	4	2.83	3584	750	Win Vista-32
3	2	3.2	3072	750	Linux
4	2	2.4	2048	500	Win XP-32
5	1	1.8	1024	160	Unix
6	1	0.45	128	20	Win98-32

And the following ontology holds for all possible applications in Grid Environment:

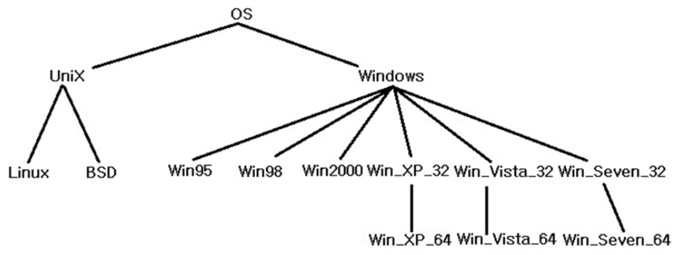

Figure 1. The ontology for the OS property

Suppose a request of {4, 2.4, 3584, 500, Win7-32} has been sent. Intuitively, when there is no cost to utilize each of the advertised resources, the order in which we suggest the advertised resources to fulfill the request is as Table I. As you can see, resources #1 and #2 are more powerful than other resources and they can satisfy the request to a better degree than other resources. So we expect a higher score and better rank for these two resources. Resources #5 and #6 are two weak resources and they are not a good match for the requests which needs a powerful resource. Resource #3 and #4 are comparatively two average resources. UDDI keyword matching, the OWL-S matching and our method, rank the six resources to meet the request as shown in Table II.

TABLE II. RESOURCE MATCHING RESULTS

Our Method		UDDI keyword matching		OWL-S matching	
Res No.	*Score*	*Res No.*	*Score*	*Res No.*	*Score*
1	1.25	2	0.4	4	0.7
2	1	4	0.4	1	0.6
3	0.837	1	0.2	2	0.6
4	0.68	3	0	5	0.6
5	0.321	5	0	6	0.6
6	0.168	6	0	3	0.5

We observe that our method ranks advertised resources in a reasonable way. UDDI keyword matching only supports an exact match, so three resources get no score if UDDI keyword matching has been utilized because they do not have any exact matches when matching the request. Resources #2 and #4 are top two suggestions of UDDI keyword matching because two of their properties have an exact match. Obviously it is neither flexible nor the best method, when there is no exact match for the request and that we are looking for appropriate approximate matches.

OWL-S matching supports approximate matching by utilizing ontology that presents conceptual relations of the environment and a scoring system based on semantic matching. In this method the potential matches are classified into Exact (E), plug-in (P), Subsume (S) and Fail (F). We assume that the match degree of exact, plug-in, subsume and fail are assigned 1, 0.75, 0.5 and 0 respectively. Matching the five properties of resource #1 with the properties of the request, results in one exact and four subsume matches (1E4S). The matching result of resources #2 to #6 are 2E2S1F, 2S2P1F, 2E2P1F, 4P1F and 4P1F respectively. Considering the above mentioned match degrees, the score of each resource for satisfying the request, are calculated according to table II. As you can see the resource ranking offered by OWL-S matching is not proper for the request and the scores are not reasonable. That is because first, the plug-in match (advertisement being more general than the request) is considered a better level of matching than the subsume match (advertisement being more specific than the request) in OWL-S. It’s not true most of time because an advertisement that is more general than the request, can satisfy the common features of its descendant but it can not necessarily fulfill the specific features of the request. On the other hand, when the request is more general than the advertisement, it only needs some general features that could be found in its descendant. Secondly, it doesn’t take the advantage of comparableness of numerical properties and utilizes ontology and semantic matching for all kinds of properties.

The scores show that OWL-S can not differentiate well between the advertised resources with different performance levels. For example resource #1 is a powerful resource and resource #5 is a weak resource, but they get the same score for satisfying the request. The first reason of the closeness of their score is that plug-in match is considered more preferable than subsume match in OWL-S matching as mentioned before. The second reason is that the matching of the advertised resource property and the corresponding request property (especially for numerical property types) results in many different matching scenarios but OWL-S matching classifies this entire matching situation into just four different classes. It is true that we assign a value to each matching level and if other values are assigned the score will yet change, but we tried to assign an average value to plug-in and subsume match levels and considering the definition of these two match levels, other reasonable values should be close to our chosen values, so assigning other value to plug-in and subsume match levels would not result in a dramatic change in scores.

The examination of the rankings proposed by our approach, shows that it ranks the advertised resources in a more reasonable way according to the request. The request needs a powerful resource and our method first suggests resources #1 and #2, then the two average resources (#3 and #4) and finally the two weak resources (#5 and #6). In addition, the score assigned to each resource, expresses the differences in the capabilities of the advertised resources in a much more clear and meaningful way. Resources #1 and #2 that are two powerful resources which have high capabilities to meet the request’s needs get a score close to 1 and resource #6 which has the lowest capabilities according to the request, gets a score near 0.

III. CONCLUSION

In this paper we have presented a new approach for grid resource matching. Our method divides the resource properties into two classes: numeric properties and non-numerical properties. It utilizes ontology to compare the non numeric properties of the request and the advertised resources. It also uses some simple mathematical operations to match the numeric properties of the request and the advertised resources.

Our experimental results show that the ranking process in our proposed method is more effective in resource matching than UDDI matching and OWL-S matching methods. We also clarify the reasons which justify the correctness of the experimental results.

REFERENCES

[1] J. S. Hurley: Overview of Grid Computing: http://net.*educause*.edu/ir/library/pdf/DEC0306.pdf, 2003.

[2] L. Tarricone and A. Espositio, "Grid Computing," in Advances in Information Technologies for Electromagnetics: Springer, 2006, ch. 5.

[3] Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 259-278, 1999.

[4] I. Ataollahi and M. Analoui, “Resource discovery using rough set in Grid environment,” in 14th International CSI conference (CSICC2009), Tehran, 2009.

[5] K. R. Bindu and S. M. Bhanu, “Group Aware Semantic Matchmaker for Grid Resource Discovery,” in International Conference on Advances in Recent Technologies in Communication and Computing, 2009.

[6] M. Paolucci, T. Kawamura, T. R. Payne and K. Sycara, "Semantic Matching of Web Services Capabilities," Proceedings of first International Semantic Web Conference. (ISWC2002), Berlin, 2002.

[7] S. Banerjee, S. Basu, S. Garg, S.J. Lee, P. Mullan, and P. Sharma, "Scalable Grid Service Discovery Based on UDDI," Proc. Third Int'l Workshop Middleware for Grid Computing (MGC 2005), pp. 1-6, Dec 2005.

[8] B. Sinclair, A. Goscinski, and R. Dew, "Enhancing UDDI for Grid Service Discovery by Using Dynamic Parameters," Proc. Int'l Conf. Computational Science and Its Applications (ICCSA 2005), pp. 49-59, May 2005.

[9] A. Bandara, T. Payne, D.Roure, N. Gibbins, and T. Lewis, "Semantic Resource Matching for Pervasive Environments: The Approach and its Evaluation," Technical Report School of Electronics & Computer Science, University of Southampton, 2008.

[10] P. Pukkasenung, P. Sophatsathit, and C. Lursinsap, "An Efficient Semantic Web Service Discovery Using Hybrid Matching," Proceedings of the 2nd International Conference on Knowledge and Smart Technologies (KST2010), 2010.

[11] Z. Pawlak,"Rough Sets," International Journal of Computer and Information Science, Vol. 11, no. 5, pp. 341-356, 1982.

[12] M. Li, B. Yu, O. Rana, and Z. Wang, “Grid Service Discovery with Rough Sets,” IEEE transactions on knowledge and data engineering, vol. 20, June 2008.

[13] B. Yu, M. Li, "RSSM:A Rough sets Based service Matchmaking Algorithm," Proc. of UK e-Science All Hands Meeting (AHM2006), Sep. 2006.

[14] I. Ataollahi and M. Analoui, “Resource Matchmaking Algorithm using Dynamic Rough Set in Grid Environment,” (IJCSIS) International Journal of Computer Science and Information Security., vol. 4, 2009.

[15] I. Ataollahi, M. Bakhshi, and M. Analoui, "Fuzzy-Based Dynamic Rough Set Resource Discovery According to User Preferences in Grid Environment," (IJCSIS) International Journal of Computer Science and Information Security, Vol. 8, No. 5, August 2010.

[16] Dong Ya Li , Bao Qing Hu, “A Kind of Dynamic Rough Sets,” in Proc. 5th Int. Conf. on Fuzzy Systems and Knowledge Discovery, Vol.3, p.79-85, August, 2007.

[17] L. Khanli, M. Analoui, "QOSs Guarantee in Grid Computing Using Active Database ," Ph.D Thesis ,Iran University of Science and Technology, Tehran, 2007.

[18] Xiaobing Pei, YuanZhen Wang, “an Approximate Approach to Attribute Reduction”, in Int. Jour. on Information Technology, VOL. 12, NO.4, 2006.