Confidence-based grid service discovery

Youcef Derbal

Ted Rogers School of Information Technology Management
Ryerson University
350 Victoria Street, Toronto
Ontario, M5B 2K3, Canada
E-mail: yderbal@ryerson.ca

Abstract: Grids are geographically distributed aggregates of resource nodes that support the provision of computing services, including computing cycle, simulation services, data mining and data processing services. Grids span multiple management domains with different service provisioning strategies. In this paper, we present a decentralised strategy of service discovery that utilises a selective service capacity state dissemination and an experience-based confidence model. The model provides a measure of the likelihood that a discovery request forwarded to a peer node would lead to a match between the requested capacity and the node's available service capacity. The simulation results show that the proposed algorithm outperforms both the Flooding as well as the random forwarding discovery schemes.

Keywords: service discovery; service capacity state dissemination; confidence model; neighbourhood topology; computational grids.

Reference to this paper should be made as follows: Derbal, Y. (2008) 'Confidence-based grid service discovery', *Int. J. Web and Grid Services*, Vol. 4, No. 2, pp.189–210.

Biographical notes: Youcef Derbal (Ing'86, MSc'89, PhD'94) is an Assistant Professor at the Ted Rogers School of Information Technology Management, Ryerson University. His research interests include programming models for high performance computing systems as well as computational grids. His current research activities are focused on new parallel computing models for multicore computing platforms to support scientific computing and, in particular, molecular dynamics simulations.

1 Introduction

Grids are large-scale distributed networks of computing resource nodes bound by a management framework to enable the provision of computing services. Often, grids span multiple management domains associated with different service providers. The issues pertinent to the grid computing framework include service discovery, task scheduling, flow management and the provision of services to users with a desirable quality of service (Foster and Kesselman, 2004). This paper addresses one mechanism of this evolving framework, namely service discovery. In its basic form, service discovery refers to the search for a service provider that has a sufficient capacity to handle some user-submitted service requests. This is akin to the general resource discovery problem associated with networked environments (Harchol-Balter *et al.*, 1999; Abraham and Dolev, 2003) and distributed computing systems (Maheswaran, 2001; Dasgupta, 2003; Dimakopoulos and Pitoura, 2003; Wu *et al.*, 2004; Zhu and Zhang, 2004). For distributed computing systems, the traditional resource discovery problem refers to the question of finding hosts with the necessary capacity and attributes to execute computing jobs. The hosts are identified by names or IP addresses and are characterised by their dynamic states, which may include the number of currently available slots. This formulation of resource discovery has been addressed in works related to grid computing (Cao *et al.*, 2001; Maheswaran, 2001; Dimakopoulos and Pitoura, 2003; Iyengar *et al.*, 2004). Maheswaran (2001) developed a taxonomy of resource discovery schemes and a comparison between various resource disseminations approaches. Iyengar *et al.* (2004) propose an approach where resource state information is selectively disseminated among neighbours. In this scheme, the propagation of resource state is performed more frequently and with higher resolution to closer repository. The resulting reduction in network congestion, compared to full dissemination, makes it more scalable.

Service discovery shares many of the issues associated with the traditional problem of resource discovery. Indeed, however formulated, the discovery mechanism in grids faces the overall challenges of a geographically distributed environment spanning distinct management domains with distinct exploitation policies, and subject to dynamic load, subsystem failures, intermittent participation of service providers, and the unavoidable latency. These grid environmental conditions may require some form of dynamic dissemination of service availability to be developed with tradeoffs between three extreme alternatives (Cao *et al.*, 2001), namely:

1 grid wide dissemination of the state of service capacity

2 no dissemination of the state of service capacity

3 centralised grid wide service registry.

The first alternative is clearly not scalable with respect to the network traffic, bandwidth consumption, network latency, and the communication and storage overheads. The absence of any service capacity state feedback is counter intuitive and would require a complex service discovery where a potentially large number of grid nodes would have to be traversed before a suitable service provider is found. The third alternative is equally problematic since a centralised registry would ultimately result in a bottleneck and limits the scalability of the grid. In addition to the necessity for the above tradeoffs to achieve fast convergence of the discovery schemes, the need for scalability is a theme that is recurrent in the research works on traditional resource discovery in network environments, including grids (Foster and Iamnitchi, 2003; Bukhari and Abbas, 2004).

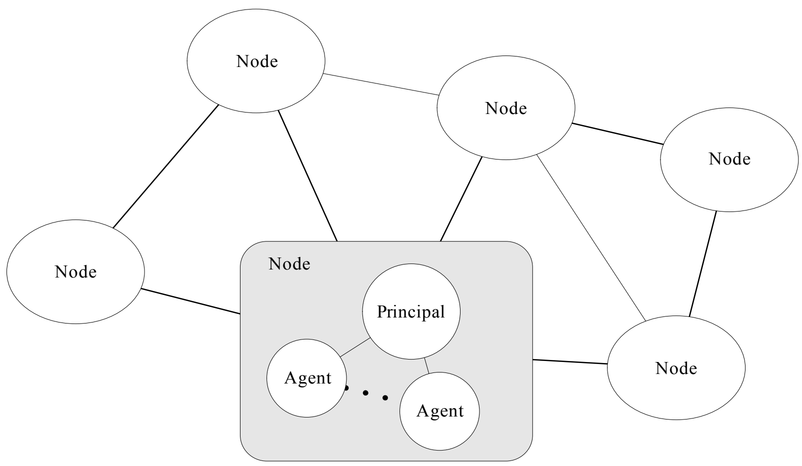
In this work, the approach to service discovery is formulated in the context of the emerging Open Grid Service Architecture (OGSA) (GGF, 2005). The contributions include a decentralised strategy of service discovery that utilises a selective dissemination of the state of service capacity and an experience-based confidence model. The model provides a measure of the likelihood that a discovery request forwarded to a peer node would lead to a match between the requested capacity and the target node's available service capacity. According to the simulation results, the proposed algorithm outperforms the widely used strategies of random forwarding and Flooding respectively.

The rest of the paper is organised as follows: Section 2 includes a description of the considered grid architectural framework. Section 3 provides a detailed exposition of the proposed model and associated strategy of service discovery. Simulation results and their analysis are given in Section 4, while related works are discussed in Section 5, followed by conclusions and future works given in Section 6.

2 Grid architectural framework

The assumed architectural framework views the grid as a dynamic federation of service providing nodes spanning distinct management domains (see Figure 1). Service requests may be submitted from any grid node, which may delegate their handling to another node based on the availability of service capacity. There are no restrictions on the IP connectivity between any pair of grid nodes since such connectivity is essential to the implementation of grid-wide scheduling strategies and delegation of service request handling. In order to support the development of a scalable mechanism of service discovery, the state of hosted service capacity is exclusively communicated among neighbouring nodes only. Hence, the pathways of state information exchange, shown by the continuous lines in Figure 1, define a network topology that we refer to as the Grid Neighbourhood (GN) topology. This is an alternative to the establishment of message pathways between all peer nodes which would clearly be detrimental to the scalability of the grid (Iyengar *et al.*, 2004). While the internal node's organisation is transparent to the discovery process, we will assume without loss of generality that it is made up of a set of service providing agents attached to computing hosts. In each node one agent, called the Principal, is responsible for the mediation of inter-node interactions. Grid nodes may join or leave the grid by informing at least a single neighbouring node, which subsequently informs its neighbours about the recent change of the neighbourhood configuration. For a newly joining node, the identity information of the contact node may be acquired through an out of bound process. The identity information may include the IP address and port numbers as well as any other credential or configuration parameters necessary for the establishment of communication links.

Figure 1 The grid as a federation of independent service providing resource nodes



Aside from the restriction on the exchange of the state of service capacity, the considered architectural framework is similar to the organisation of many large experimental grids in operations; including the Earth System Grid (ESG) (Bernholdt, 2005) and the *GriPhyN* grid (Avery and Foster, 2001).

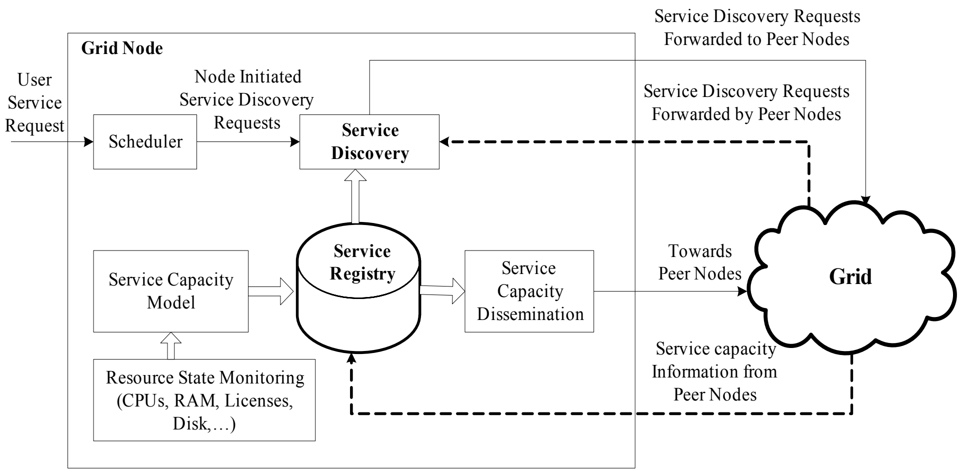
3 Model of service discovery

With the adoption of service oriented grid architectures recommended by the OGSA specifications (GGF, 2005) and the WSRF standards (OASIS, 2005), the traditional problem of resource discovery problem needs to be reformulated as one of service discovery. In such formulation, grid resources which include processors, documents, storage services, business applications, and data processing services are uniformly exposed for exploitation through grid services in compliance with the WSRF standards. The reliance on a service oriented view of grid resources introduces a significant simplification of the grid management mechanisms; including scheduling and discovery (Graupner *et al.*, 2003; Derbal, 2005). However, in order to evolve the traditional resource-attribute-based search approach to discovery and resource selection towards a service discovery strategy, the service capacity of a node needs to be quantified. In general, service discovery entails the consideration of the following elements:

- the model of service capacity
- the dissemination strategy of the service capacity information
- the discovery strategy.

Figure 2 provides an illustration of the relationship between service discovery, and service capacity state modelling and dissemination. The details of these components are articulated in the sections that follow below.

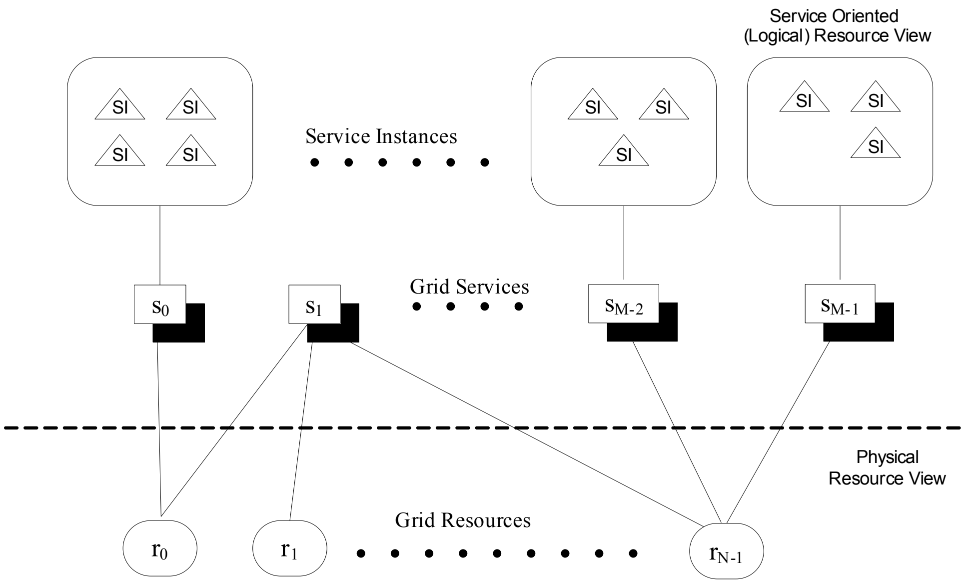
Figure 2 Context for the problem of service discovery



3.1 Service capacity modelling

Let $S = \{S_0, S_1, \ldots, S_{M-1}\}$ be the set of grid services deployed on a grid node $h \in G$, where G denotes the set of nodes that make up the grid system under consideration. Let $R = \{r_0, r_1, \ldots, r_{N-1}\}$ denotes the set of node resources which may include CPUs, storage space, software licenses, database servers, deployed application services, and special devises. The instantiation of a service, the invocation of its methods and the maintenance of its state require the availability, in sufficient level or numbers, of one or more of the elements of R. The dependence of services on a common subset of resources creates an inter-service operational coupling as these services compete for the use of the limited computing resources of their hosting node (see Figure 3). As a result, the ability of a grid node to handle a service request relies in the end on the extent to which the collective aggregate availability behaviour of its resources can satisfy the requirements of the service in question. The challenge then is, to develop a measure of service capacity that quantifies the aggregate residual potential of the node to handle inbound requests. In this work, the node service capacity is defined as the number of service requests that can be concurrently handled. Such measure is naturally dynamic and varies as a function of the varying load and the availability of node resources. The details of the model underlying this measure and its experimental validation are provided in Derbal (2007).

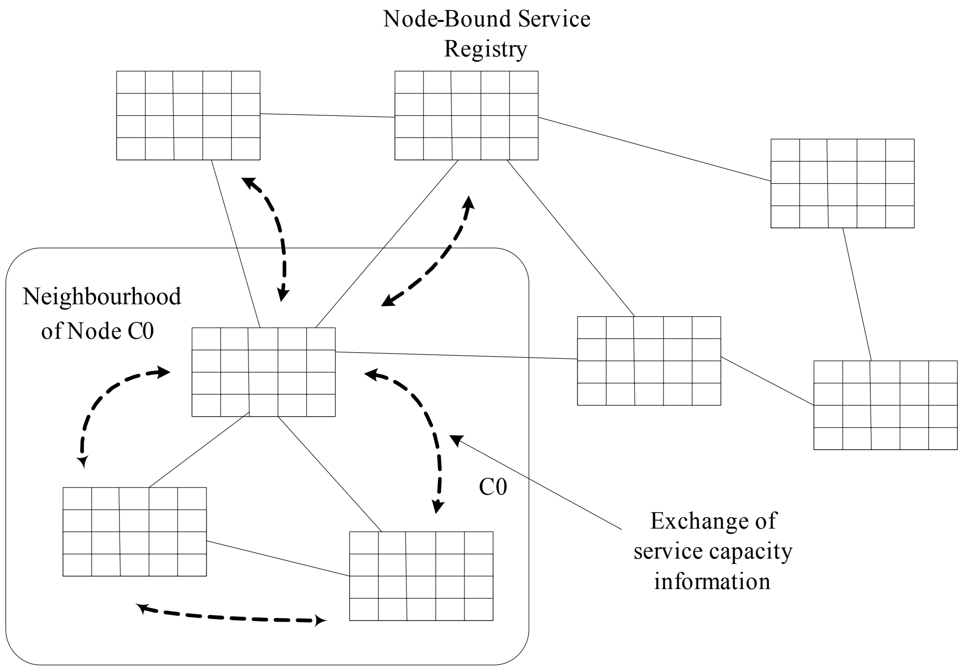
Figure 3 Mapping the grid resources to the hosted grid services



3.2 Service capacity state dissemination

As mentioned above, a scalable strategy of service discovery has to avoid the excessive use of network bandwidth and introduce as little network congestion as possible. In this respect, grid nodes need to be aware of the nature of provisioned services and the associated available capacity. Due to the unavoidable latency, a universal dissemination of the nodes' service capacity would not necessarily yield the desired goal of timely awareness about the capabilities of peer nodes. Similarly, maintaining a centralised registry for the nodes' service capacity is detrimental to the scalability of the grid management mechanisms (Zhang and Schopf, 2004). Hence, a selective dissemination model is devised to strike a balance between awareness and scalability, whereby the capacity information is disseminated among neighbours only. This encourages the consumption of services hosted by closer peers before seeking the services of distant ones, reducing hence the resulting latency and network congestion. The dissemination model is supported by a distributed set of service registries maintained by the grid nodes (see Figure 4).

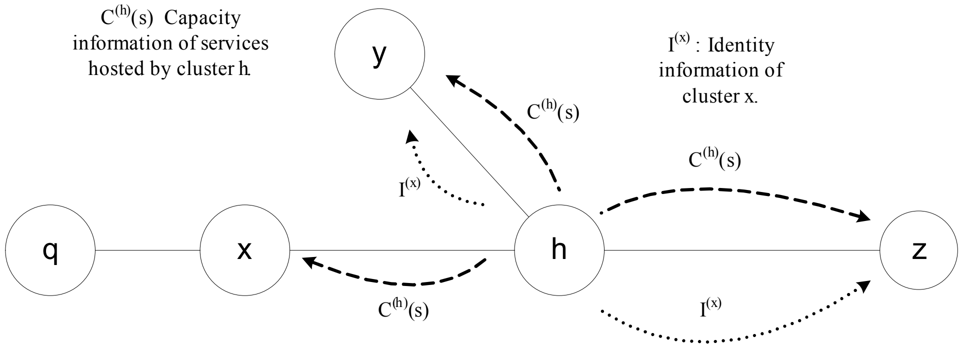
Figure 4 Dissemination of the state of service capacity



Each node-bound registry includes the capacity information of grid services hosted by the home node in addition to the disseminated information about the services hosted by neighbouring nodes. The entries of the node registry include the name and description of the service, the interface definition of the service, the identifier of the provider and the currently available capacity. The introduced notion of grid neighbourhood is defined as any arbitrary grouping of nodes that regularly share service capacity information in addition to being aware of each others identities. As a result any two nodes that belong to a common neighbourhood are said to be neighbours. Furthermore, two non-neighbouring nodes that have a common neighbour are said to be secondary neighbours. For the assumed grid architectural framework, a node may be a member of more than one neighbourhood and may join or leave a neighbourhood grouping at will. The service capacity and identity information are selectively disseminated among neighbours as follows (see Figure 5):

- Each node *h* disseminates to its neighbours *x*, *y*, and *z* the capacity state $C^{(h)}(s)$ of hosted services such as *s* on start-up of the node, and following any changes of the service capacity.
- Each node *h* disseminates to its neighbours the identity information $I^{(x)}$ of one of its secondary neighbours which is in this case *x*. The secondary neighbour is randomly selected. If nodes *x* and *y* were neighbours, *h* would disseminate to *y* the identity information of a node other than *x*. No identity information is exchanged if a secondary neighbour is non-existent.

Figure 5 Dissemination of service capacity and identity information



The identity information $I^{(x)}$ of node *x* includes the IP address, port numbers, and any other configuration parameters or security credentials required for the establishment of a communication link to node *x*. The exchange of identity information of one secondary neighbour among any pair of neighbouring nodes allows, as will be explicated in the next section, the discovery mechanism to hop beyond the neighbourhood where the discovery process is initiated. The set of registries associated with neighbouring nodes contain redundant information about the capacity of services hosted by the neighbourhood (set of immediate neighbours). As a result the failure of one node-bound registry, although disruptive, may be addressed through a recovery process that includes the delegation of incoming discovery requests to another node in the neighbourhood until the normal operational state is restored. The proposed approach of capacity state dissemination results in a network of non-disjoint registries that enables the discovery strategy to explore the solution space at a faster pace. However, the advantages of robustness, scalability and fast search convergence associated with the proposed distributed registries of service capacity are associated with the need for extra storage capacity at the node level. The size g_{Size} of a node registry may be bound as follows:

$$g_{Size} \leq (ns_{\max} + ns_{\max} ne_{\max})w \tag{1}$$

where $ne_{\max}$ is the maximum number of neighbours, and $ns_{\max}$ is maximum number of deployed services per node. *w* is the required storage space associated with a single service record which may be no more than 38 bytes (32 bytes for the service name, 4 bytes for the node ID, and 2 bytes for the service capacity). For a given number of neighbours, the bound is clearly a linear function of the number of hosted services. The implementation of the node service registry may be realised using a relational database, an XML database or an LDAP directory.

3.3 Second order discovery algorithm

Assuming the grid architectural framework described above, a new model of service discovery called the Second Order Discovery Algorithm (SODA) is proposed. It leverages the selective dissemination of service capacity among neighbouring nodes, in addition to a confidence model that estimates the likelihood that peer nodes would have the capacity to fulfil the needs of a given service request. Let a User Service Request (USR) be defined as follows:

$$USR = (\Omega, \Phi, Q). \tag{2}$$

$\Omega = \{(s_0\ c_0),\ldots,(s_{m-1}, c_{m-1})\}$ is the set of pairs of required grid services and their capacities respectively. Q is a user defined set of Quality of Service (QoS) parameters such as the maximum wait time before scheduling. The USR handling flow Φ defines the execution sequence of the various tasks associated with the handling of the USR. The consideration of the flow in the service discovery will be limited to the prioritisation of the USR discovery sequence.

The basic question of service discovery is to find the providers that host the necessary grid services with sufficient capacity levels to handle a submitted USR as defined by Equation (2). In other words, given a grid made up of a set G of nodes, the service discovery problem reduces to finding a subset $K \subset G$ of nodes that host the required gird services with a sufficient residual capacity to handle the submitted USR. Once the subset K is determined, the scheduling strategy chooses the smallest subset of K that leads to an optimal handling of the USR in some defined sense. The proposed SODA algorithm is described by the UML activity diagram given in Figure 6. The discovery process starts at the submission node where the service registry is queried about the required services and their current available capacities. The node records a successful discovery for the subset of services for which a match was made, and forwards the appropriate discovery requests for the remaining subset of required grid services to a secondary neighbour. This secondary neighbour attempts a similar matching process and notifies the submission node about the subset of discovered services and forwards the discovery requests to one of its secondary neighbours for the non discovered services (see Figure 7). This discovery cycle is applied recursively until all the required grid services are discovered or the process is timed out when a defined Time-to-Schedule (TTS) limit is reached. Each one of the individual discovery requests is restarted by the submission node whenever a defined Time-to-Live (TTL) limit is exceeded. The discovery notifications originated from the potential providers are recorded by the submission node which ultimately asserts the discovery of the entire set of grid services required by the submitted USR.

Figure 6 Summary of the proposed confidence-based service discovery algorithm

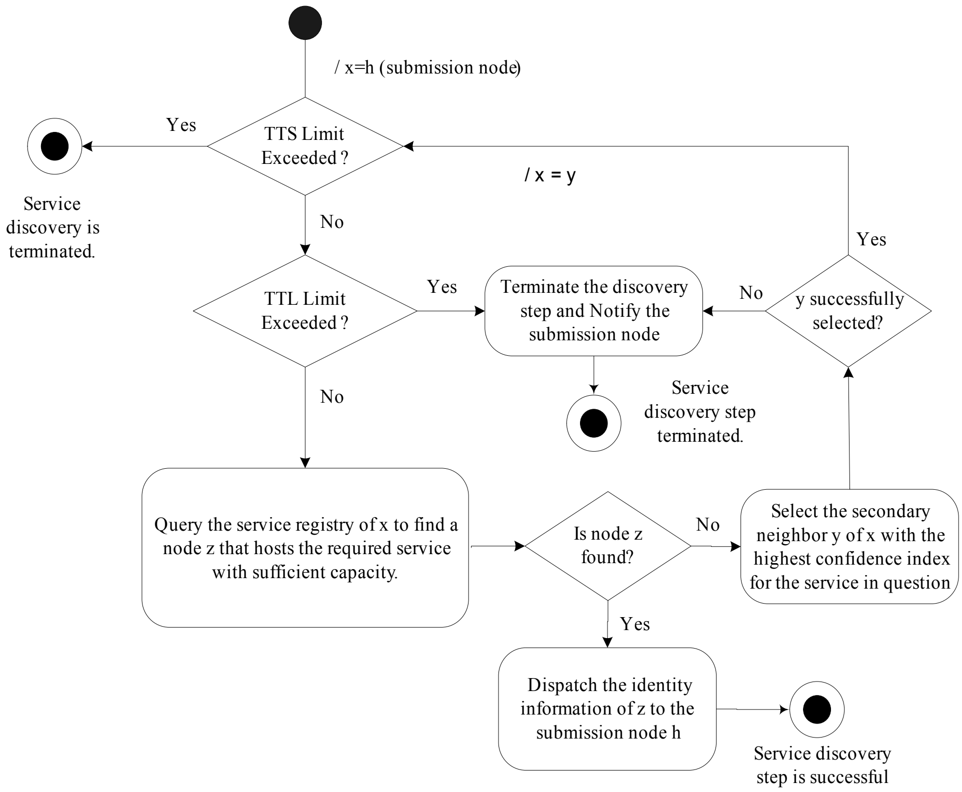
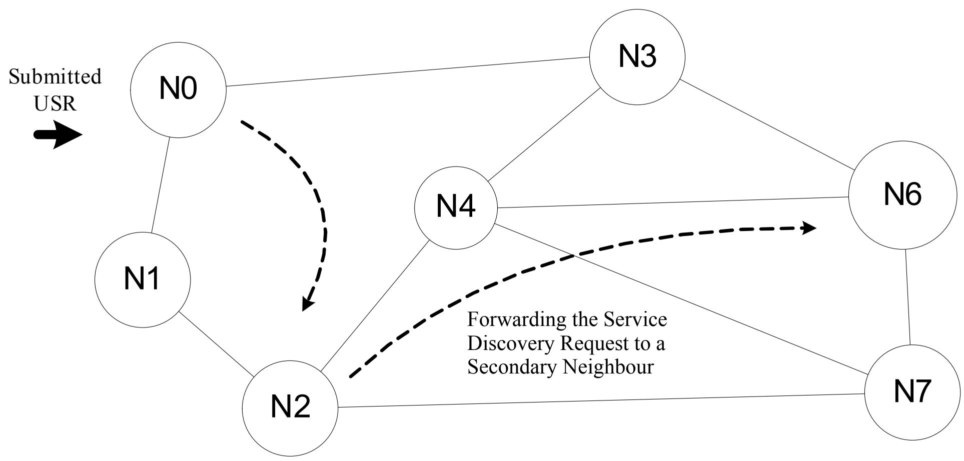


Figure 7 Forwarding a service discovery request to a secondary neighbour



The choice of the secondary neighbour is made using a confidence model that estimates the likelihood that a service discovery request would result in a successful outcome if forwarded to a peer node. The confidence model consists of a set of confidence indices maintained by the grid nodes about their secondary neighbours. The secondary neighbour associated with the highest confidence index is chosen as the target node to receive the service discovery request. In the case where the indices are equal, a random selection of the secondary neighbour is made. For a pair of grid nodes x and y, the proposed confidence index to be maintained by x *vis-à-vis* y is computed using the history of past quality of service experienced by node x in response to grid service requests it made to node y. Let $n_{x,y}^{(s)}(t)$ be the total number of requests to service s that have been delegated to node y by node x up to time t. Let $m_{x,y}^{(s)}(t)$ be the size of the subset of successfully handled requests in compliance with some specified QoS. The confidence index is then defined as follows:

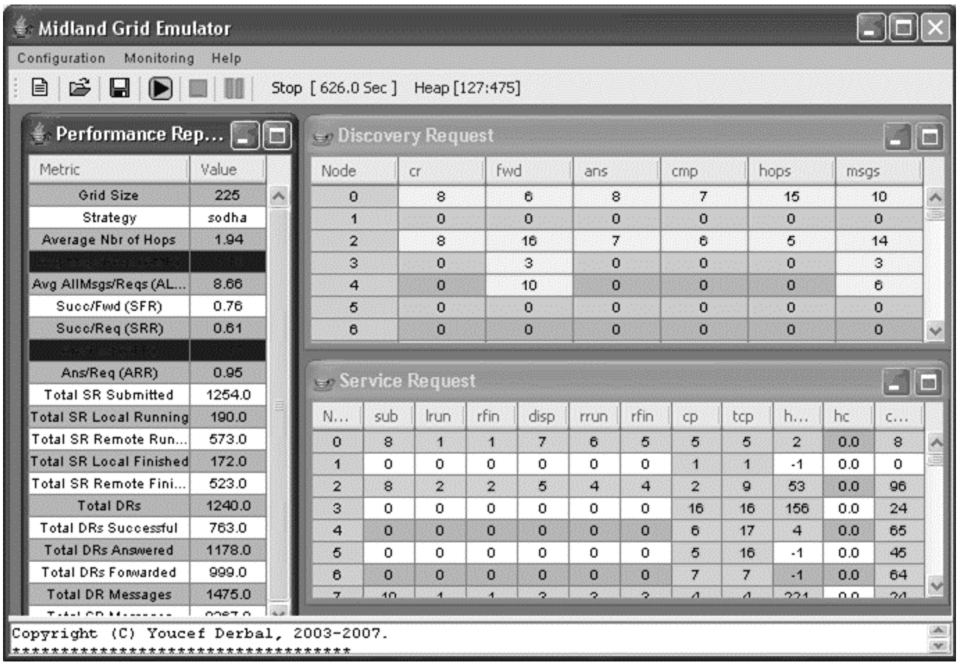
$$P_{xy}^{(s)}(t) = \frac{m_{x,y}^{(s)}(t)}{n_{x,y}^{(s)}(t)}. \tag{3}$$

In the absence of any prior dealing with a peer node, the confidence index is set to the median value of 0.5. This choice is motivated by the interpretation of the index $p_{xy}^{(s)}$ as the probability of successful handling of a request delegated by x to y, and as such the value of 0.5 represents the maximum uncertainty regarding the expected outcome. An alternate definition of the confidence index may use the number of time a forwarded discovery request is satisfied by some given node. This would provide a feedback to the discovery strategy that implicitly takes into account the dynamic nature of grid state as opposed to the open loop nature of many algorithms encountered in the literature. However, since the purpose of service discovery is to support the service exploitation strategies, the feedback should rather be dependent on the requests' handling performance. Indeed, due to latency, once a service provider is discovered, there are no assurances that subsequently dispatched requests to the discovered provider's node would find a sufficient capacity for their handlings, unless a scheme of capacity reservation on discovery is implemented. Hence, closing the loop at the exploitation level, as defined above, is more congruent with the ultimate goal of effective service exploitation than if the loop was closed at the discovery level. The proposed approach utilises an easy to implement confidence model to support the proposed feedback-based discovery strategy. However, with the interpretation of the confidence index as a probability measure, any model of performance that enables the estimation of such probability may be used in conjunction with the proposed neighbourhood-based capacity dissemination and caching that underlies the SODA algorithm.

4 Simulation results

The reported simulations were conducted using the Midland Grid Emulator developed by the Author (see Figure 8). The Emulator can be configured for an arbitrary number of nodes where neighbours are randomly assigned in order to avoid any favourable configuration to the discovery process. Different grid-wide distributions of service hosting and associated capacity may be configured. In this respect, let $P_x^{(s,c)}$ be the probability that node *x* hosts service *s* with capacity *c* at configuration time, then the grid capacity distribution is said to be balanced for service *s* if for any two arbitrarily considered nodes *x* and *y* we have $P_x^{(s,c)} \approx P_y^{(s,c)}$. On the other hand, a grid capacity distribution is said to be unbalance if $P_x^{(s,c)} \neq P_y^{(s,c)}$ for any two distinct nodes *x* and *y*.

Figure 8 Midland grid emulator



For the simulation reported in this work, balanced distributions are realised by configuring each node to offer a randomly selected subset of the services associated with the grid. For each hosted service the initial (maximum) capacity is selected using a Uniform distribution over the interval [1 17]. For a non-balanced configuration, the initial capacity is assigned using a Normal distribution with mean 10 and a standard deviation of 4. The service requests are submitted to the grid nodes with an inter-arrival time simulated using a Poisson process with a rate of 0.0075. In order to account for the bursty nature of requests' submission, on each arrival a group of requests are created and queued at the submission node. The number of requests queued is generated using a Pareto process with a range of 7 and a shape of 3. The service requests are assigned a Gaussian distributed makespan with a mean of 60 s and a standard deviation of 15 s. The required capacity for the service requests are selected using a Uniform distribution over the interval [1 7]. The sleep time for the threads running the service request submission, discovery, scheduling, and request handing was set to 10 ms for all nodes. In order to simulate the confidence index, a simple scheduling/delegation strategy is used. In a first step, a service discovery request is initiated. Based on the discovery results, the request is either queued locally for execution or dispatched to a remote provider (peer node) suggested by the discovery algorithm. Every dispatching instance of service request from node *x* to node *y* is counted as a delegation event associated with the handling of the service request in question. This translates into the incrementing of the variable $n_{x,y}^{(s)}(t)$ by 1. The service request handling starts upon arrival at node *y*, provided that the actual availability of the required grid service capacity is found to be sufficient. If such capacity is sufficient then the delegation is deemed successful and $m_{x,y}^{(s)}(t)$ is incremented by 1. The handling of the service request is emulated using a timer programmed to run for the duration of the handling makespan. Once the timer expires, the used capacity is reclaimed by the node, and the service request handling is deemed completed.

In order to provide a comparative analysis of SODA's performance, two discovery strategies which have been in wide use under different forms, are considered; namely:

1 Random Neighbour Forwarding (RNF)

2 Flooding.

In RNF a service discovery request is forwarded to a randomly selected neighbour and recursively relayed from there to a random neighbour until the service is discovered or the TTL expires. For the Flooding algorithm, the discovery request is relayed to every neighbour and recursively relayed from there to every neighbour until the service is discovered or the TTL expires. The performance of discovery strategies in distributed systems including grids, is essentially characterised by the convergence speed, the density of message exchange necessary for the discovery process, and the ratio of answered to forwarded discovery requests within a TTL period. These performance properties may be quantified using the following metrics:

$$\overline{n}_{hops} = \frac{1}{N_a} \sum_{i=1}^{N_a} hops(r_i) \tag{4}$$

$$\overline{m}_{msg} = \frac{1}{N_r} \sum_{i=1}^{N_r} msg(r_i) \tag{5}$$

$$q_{ar} = \frac{N_a}{N_r} \tag{6}$$

where $\overline{n}_{hops}$ is the average number of hops for the discovery requests that were successfully answered within the TTL. $\overline{m}_{msg}$ is the average number of exchanged messages for the forwarded requests. q_{ar} is the discovery success rate defined as the ratio of the grid-wide total number of answered discovery requests over the grid-wide total number of forwarded service requests. $hops(r_i)$ and $msg(r_i)$ are the number of hops and messages associated with the discovery request r_i. Given the grids' large scale distribution, the scalability of service discovery is critical to the practical feasibility of any grid exploitation strategy. In this respect, the grid size, and the maximum number of neighbours per node are used as parameters for the comparative analysis of the proposed approach.

The first set of simulation results is illustrated in Figures 9–14, where the considered performance metrics are recorded as the grid size is increased from 100 to 1200 nodes for balanced and unbalanced service capacity distributions respectively. The TTL was set to 150 ms, which is sufficiently high for the simulation time scale, so as to minimise its effect for this simulation run.

Figure 9 Average number of hops for a balanced service capacity distribution

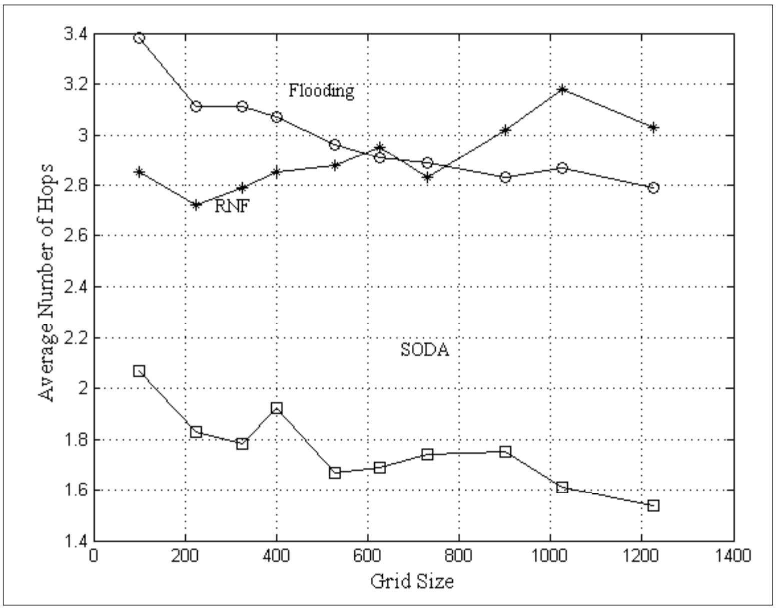


Figure 10 Discovery success rate for a balanced service capacity distribution

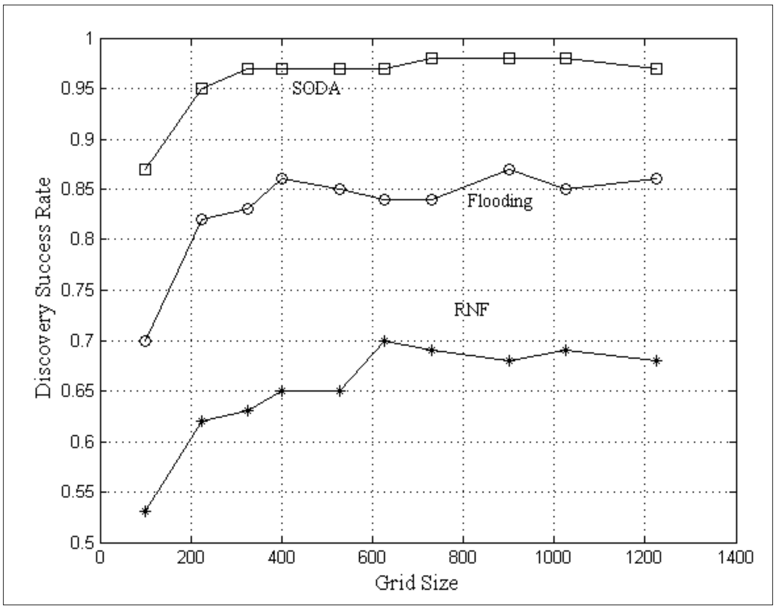


Figure 11 Average number of exchanged messages for a balanced service capacity distribution

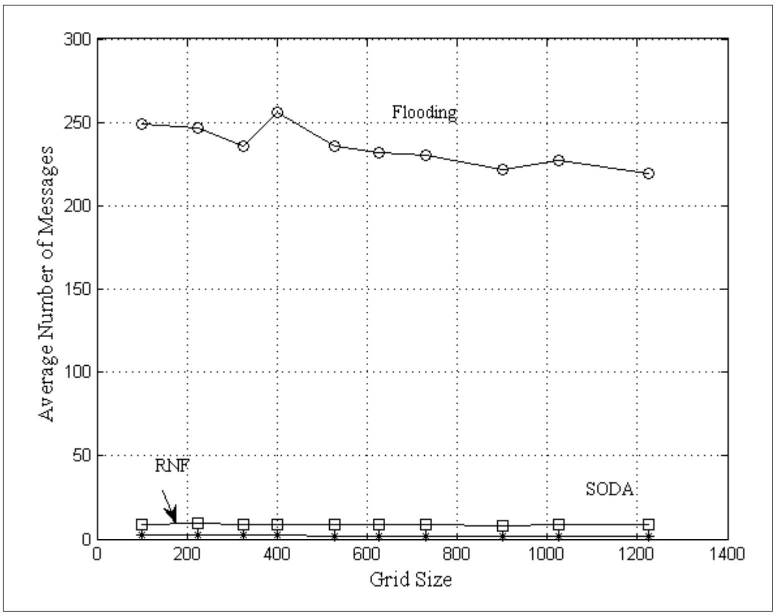


Figure 12 Average number of hops for an unbalanced service capacity distribution

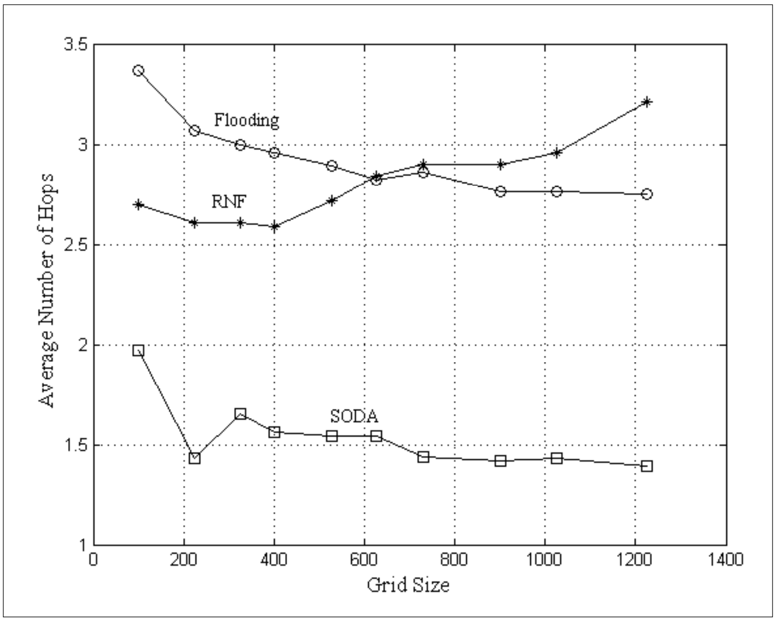


Figure 13 Discovery success rate for an unbalanced service capacity distribution

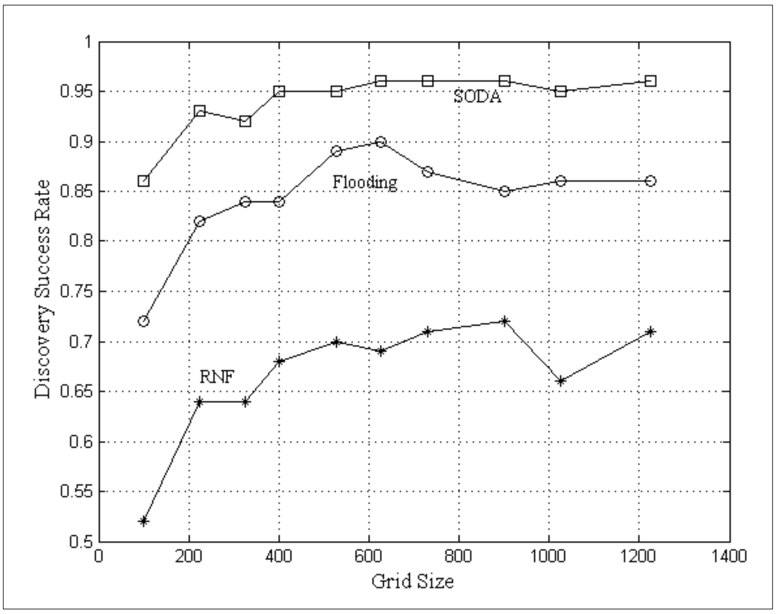
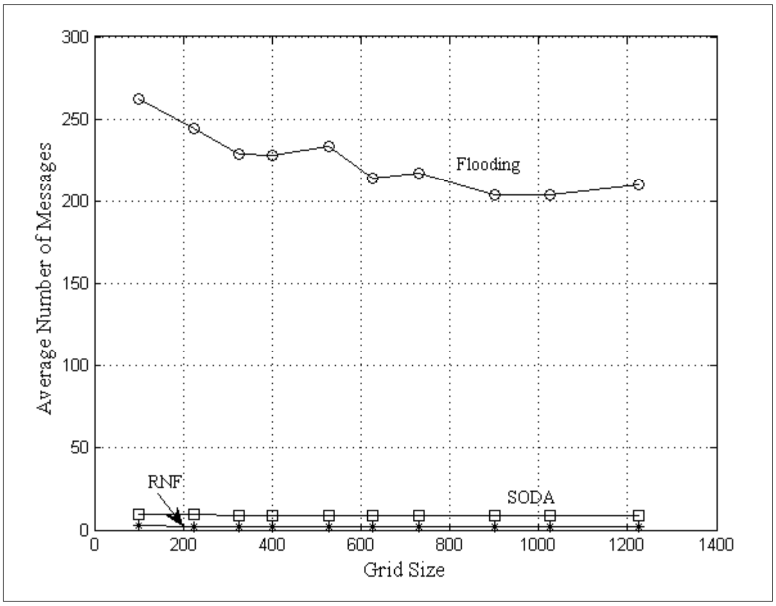


Figure 14 Average number of exchanged messages for an unbalanced service capacity distribution



The above simulation results suggest that for both balanced and unbalanced distributions, SODA outperforms both the RNF and Flooding strategies with respect to the average number of hops and the success rate q_{ar}. The Flooding strategy exhibits a relatively high average number of messages compared to a moderately low and stable value of $\bar{m}_{msg}$ for SODA. Note that for the proposed discovery strategy, the capacity's dissemination messages are included, which explains the slightly higher number compared to the RNF strategy. SODA delivers a much higher rate of discovery success compared to both Flooding and RNF respectively. While a differential in the performance of the three considered schemes is noticeable for the two considered distributions of service capacity, such deviation is not significant. This is expected since the distributions of requested and provisioned service capacities are chosen to be random so as to avoid any significant supply-demand resonance pattern that might bias the performance of the discovery strategy. The performance of the considered algorithms shows robustness against the grid size. Indeed, while SODA steadily outperforms the other schemes, these last maintained an approximately constant discovery success rate and response time (average number of hops) as the grid size is increased. The reason for this behaviour lies in the adopted neighbourhood topology of the grid whereby all three strategies benefit from the reliance on a progressive widening of the search radius starting from the node where the requests are initiated. As a result, 'hot spots' of high density of inbound requests originating from various consumer nodes are less likely to develop as the grid size increases. This would mean less competition among incoming discovery requests over the residual service capacity of the target provider, leading to a stable performance irrespective of the grid size as illustrated by Figures 9–14.

The second set of simulation runs deals with the effect of varying the connectivity level among peers as defined by the maximum number of neighbours (Tables 1–3). As the number of neighbours increases, the convergence speed of SODA improves due to the widening domain of awareness about the available capacity of peer nodes. However, this is accompanied with an increase in the number of exchanged messages. For RNF, all three performance metrics under consideration did not significantly change with the increase of the number of neighbours. While the small and stable number of exchanged messages is an advantage of the RNF strategy, the speed of convergence and the discovery success rate stayed unfavourably lagging by as much as 40% behind SODA. Flooding does perform relatively well with respect to the discovery success rate and the speed of convergence. However, the exponential increase in the number of exchanged messages is a serious disadvantage that limits the applicability of the strategy.

Table 1 Average number of Hops as a function of the maximum number of neighbours

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
Maximum number of neighs	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
3	1.77	2.37	3.15	1.62	2.64	3.06
4	1.41	2.73	3.30	1.34	2.64	3.23
5	1.23	2.74	3.41	1.29	2.67	3.33
6	1.25	2.43	3.45	1.14	2.65	3.36

Table 2 Discovery success rate as a function of the maximum number of neighbours

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
Maximum number of neighs	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
3	0.95	0.66	0.78	0.94	0.64	0.83
4	0.96	0.60	0.80	0.96	0.63	0.9
5	0.98	0.64	0.82	0.96	0.69	0.9
6	0.97	0.62	0.87	0.97	0.67	0.87

Table 3 Average number of messages as a function of the maximum number of neighbours

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
Maximum number of neighs	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
3	9.24	2.37	248	9.7	2.29	244
4	12.4	2.39	608	12.4	2.33	546
5	23.15	2.38	1109	15.6	2.28	1096
6	25.17	2.43	1904	18.4	2.28	1901

The last set of simulations addresses the effect of TTL on the discovery's performance as illustrated by Tables 4–6. An increase in TTL improves the discovery success rate since the requests are given more time to be satisfied by a potential provider. However, this is associated with an increase in the average number of hops and the average number of exchanged messages for all considered schemes. Overall the SODA algorithm outperforms both the RNF and the Flooding algorithms respectively. The increase in TTL from 30 to 75 ms resulted in a 90% increase of the discovery success rate for the Flooding scheme. However, the exponential increase of the average number of exchanged messages casts an unfavourable shadow on the overall usability of the strategy.

Table 4 Average number of hops as a function of TTL

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
TTL (milliseconds)	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
30	1.29	1.95	2.04	1.11	1.9	1.95
45	2.03	2.29	2.57	1.58	2.28	2.51
60	2.17	2.85	3.34	1.86	2.69	3.38
75	2.48	3	4.27	2.01	3.04	4.18

Table 5 Discovery success rate as a function of TTL

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
TTL (milliseconds)	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
30	0.72	0.37	0.38	0.72	0.38	0.4
45	0.79	0.45	0.6	0.77	0.54	0.57
60	0.83	0.55	0.66	0.9	0.54	0.7
75	0.89	0.54	0.76	0.8	0.59	0.75

Table 6 Average number of exchanged messages as a function of TTL

	Balanced service capacity distribution			*Unbalanced service capacity distribution*		
TTL (milliseconds)	*SODA*	*RNF*	*Flooding*	*SODA*	*RNF*	*Flooding*
30	7.72	1.07	5.55	7.84	1.05	5.6
45	8.42	1.85	38	8.43	1.8	38.3
60	9.23	2.48	244	9.67	2.43	260
75	9.51	3.07	1696	8.9	2.96	1733

The overall picture that emerges from the simulation results suggests that SODA delivers better performance than Flooding with a significantly lower communication overhead which was shown to be stable with respect to the grid size, the level of grid connectivity, and the TTL parameter. While RNF incurs a relatively small communication overhead, its discovery success rate and convergence speed perform much lower than SODA. In summary, it may be concluded from the simulation results that the selective awareness about the state of the neighbours' service capacity, and the confidence feedback are leveraged by SODA to deliver a more scalable performance compared to the brute force search of the Flooding strategy, and the random search of RNF.

5 Related works

As mentioned above, there is a large body of works that addresses the problem of resource discovery in distributed systems (Harchol-Balter *et al.*, 1999; Cao *et al.*, 2001; Kutten *et al.*, 2001; Kutten and Peleg, 2002; Abraham and Dolev, 2003; Dimakopoulos and Pitoura, 2003; Bukhari and Abbas, 2004). However, most of these works have framed the problem at hand as a name-based or ID-based discovery of resources with a static state or a binary state of availability at best. Other works that are more relevant to the grid environment have relied on a centralised registry of resource information (Litzkow *et al.*, 1988; Zang *et al.*, 2004; Bradley *et al.*, 2006). This approach of centralised indexing of resource information was shown to exhibit poor performance with an increase of user load (Zhang and Schopf, 2004). On the other hand, there is a collection of works that are closely related to our proposed model either in their approach to the dissemination of resource state information or the consideration of a distributed network of registries as a means to avoid the potential bottleneck of a centralised registry (Iamnitchi and Foster, 2001; Dimakopoulos and Pitoura, 2003; Chunlin and Layuan, 2004; Mastroianni *et al.*, 2005). Iamnitchi and Foster (2001) views the grid as a geographically distributed network of resource nodes. The discovery strategy relies on recursive forwarding of discovery requests to peer nodes chosen based on various criteria; including random selection, and past performance experience. This discovery request forwarding in a peer-to-peer framework is similar to our model of neighbourhood-based discovery. However, in contrast to our approach there is no dissemination or caching of peers' resource state information. This was shown to be detrimental to the discovery success rate in the reported simulation results for the case of the RNF strategy which is comparable to the algorithms proposed by Iamnitchi and Foster (2001). In Chunlin and Layuan (2004), the grid is partitioned into a collection of communicating units of agents. Grid services provided by the unit members are published in a hierarchically organised set of service repositories bound to their home units. The discovery process follows a hierarchical search through the repositories starting from the home repository where the search is initiated. In the absence of scalability analysis it is hard to conjecture on the potential performance of such discovery scheme for a real grid system. The super-peer model developed in Mastroianni *et al.* (2005) is similar in its architecture to the proposed neighbourhood topology, where the grid is organised as a federation of peer resource nodes independently managed within distinct administrative domains. The discovery process relies on a scheme of delegation to peer nodes based on statistical metrics of past interactions. The recursive delegation of service discovery request is terminated when a time-to-live period expires. The absence of any dissemination of resource state among super peers is clearly non-intuitive. Given the varying dynamics of resource availability, the statistical metrics are not a substitute for a regular observation of the actual resource state by potential consumer peers. In this respect, the model of discovery proposed in this reviewed work is less likely to be feasible for a production grid. In Dimakopoulos and Pitoura (2003), an open system of cooperating agents is considered where each agent maintains the contact information of a set of resources offered by a single peer agent. The agent-maintained information constitutes a distributed network of cache meant to yield a well performing discovery process without reliance on a centralised repository of resource information. Various algorithms where proposed for the search of a resource through the navigation of the distributed cache. The underlying idea of distributed caching of resource information is similar to our proposed model of service discovery. However, the implicit assumption of static resource state may make the adjunct search algorithms ill-equipped for the time varying resource state of a grid environment.

The dynamic nature of the grid state resulting from the intermittent resource participation, the time varying load, and the inevitable failures requires the discovery mechanism to rely on a feedback system that provides the updated state of service capacity. This consideration is not adequately addressed in most of the above works. In contrast, the proposed model of service discovery provides a comprehensive approach that avoids the drawbacks of both the hierarchical and the centralised organisations of service repositories in favour of a scalable distributed network of service registries.

The neighbourhood-based dissemination of the state of service capacity enabled the development of an alternate strategy to the central registry-based schemes of many related works, while maintaining a relatively low density of information exchange making it hence practically feasible. The emphasis on the minimisation of peer's operational coupling through the avoidance of a centralised registry and the adoption of a neighbourhood-based peer-to-peer grid topology gives the proposed approach a significant advantage over related works. Such advantage facilitates the integration of the proposed discovery model within the emerging SLA-based frameworks of grid resource exploitation (Czajkowski *et al.*, 2005).

6 Conclusions and future works

In recognition of the challenges inherent to the grid environment, we approached the problem of service discovery along three axes, namely:

1. the grid topology
2. the service capacity state dissemination
3. the service discovery scheme.

The resulting discovery strategy is adaptive, decentralised and uses a feedback in the form of a confidence model based on the performance of the service exploitation strategies. Compared to the RNF and Flooding strategies, the simulation results suggest that the proposed approach outperforms both schemes while incurring a significantly lower communication overhead than Flooding. This communication overhead is shown to be slightly higher than that of random forwarding, but is stable with respect to the grid size, grid connectivity level, and the TTL parameter. For future works further attention may need to be given to the effect of the rate of capacity state dissemination between neighbours, and the filtering of inter-node propagated capacity information. Furthermore, to ease its integration within an SLA based service exploitation framework, the service discovery may need to be regulated using a QoS driven adaptation mechanism.

References

Abraham, I. and Dolev, D. (2003) 'Asynchronous resource discovery', *Proceedings of the Annual ACM Symposium on Principles of Distributed Computing*, pp.143–150.

Avery, P. and Foster, I. (2001) 'The GriPhyN Project: toward Petascale virtual data grids', Technical Report GriPhyN-2001-15, http://vdt.cs.wisc.edu.

Bernholdt, D., *et al.* (2005) 'The earth system grid: supporting the next generation of climate modeling research', *Proceedings of the IEEE*, Vol. 93, No. 3, pp.485–495.

Bradley, A., Curran, K. and Parr, G. (2006) 'Discovering resources in computational GRID environments', *Journal of Supercomputing*, Vol. 35, No. 1, pp.27–49.

Bukhari, U. and Abbas, F. (2004) 'A comparative study of naming, resolution and discovery schemes for networked environments', *Proceedings – Second Annual Conference on Communication Networks and Services Research*, pp.265–272.

Cao, J., Kerbyson, D.J. and Nudd, G.R. (2001) 'Performance evaluation of an agent-based resource management infrastructure for grid computing', *Proceedings of the First IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp.311–318.

Chunlin, L. and Layuan, L. (2004) 'Agent framework to support the computational grid', *Journal of Systems and Software*, Vol. 70, Nos. 1–2, pp.177–187.

Czajkowski, K., Foster, I. and Kesselman, C. (2005) 'Agreement-based resource management', *Proceedings of the IEEE*, Vol. 93, No. 3, pp.631–643.

Dasgupta, P.R. (2003) 'Incentive driven node discovery in a P2P network using mobile intelligent agents', *Proceedings of the International Conference on Artificial Intelligence IC-AI 2003*, pp.750–756.

Derbal, Y. (2005) 'Service oriented grid resource modeling and management', *Proceedings of the 1st International Conference on Web Information Systems and Technologies (WEBIST 2005)*, pp.146–153.

Derbal, Y. (2007) 'A model of grid service capacity', *Journal of Computer Science and Technology*, Vol. 22, No. 4, pp.505–514.

Dimakopoulos, V.V. and Pitoura, E. (2003) 'Performance analysis of distributed search in open agent systems', *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS03)*, p.20b.

Foster, I. and Iamnitchi, A. (2003) *A Peer-to-Peer Approach to Resource Location in Grid Environments*, Kluwer Publishing.

Foster, I. and Kesselman, C. (2004) *The Grid: Blueprint for a New Computing Infrastructure*, Elsevier Science.

GGF (2005) 'Open Grid Services Architecture (OGSA)', http://www.ggf.org/documents/GFD.53.pdf.

Graupner, S., *et al.* (2003) 'Service-centric globally distributed computing', *IEEE Internet Computing*, Vol. 7, No. 4, pp.36–43.

Harchol-Balter, M., Leighton, T. and Lewin, D. (1999) 'Resource discovery in distributed networks', *Proceedings of the Annual ACM Symposium on Principles of Distributed Computing*, pp.229–237.

Iamnitchi, A. and Foster, I.T. (2001) 'On fully decentralized resource discovery in grid environments', *Proceedings of the Second International Workshop on Grid Computing*, pp.51–62.

Iyengar, V., *et al.* (2004) 'Non-uniform information dissemination for dynamic grid resource discovery', *3rd IEEE International Symposium on Network Computing and Applications (NCA04)*, pp.97–106.

Kutten, S. and Peleg, D. (2002) 'Asynchronous resource discovery in peer to peer networks', *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, pp.224–231.

Kutten, S., Peleg, D. and Vishkin, U. (2001) 'Deterministic resource discovery in distributed networks', *Annual ACM Symposium on Parallel Algorithms and Architectures*, pp.77–83.

Litzkow, M.J., Livny, M. and Mutka, M.W. (1988) 'Condor – a hunter of idle workstations', *Proceedings of the International Conference on Distributed Computing Systems*, pp.104–111.

Maheswaran, M. (2001) 'Data dissemination approaches for performance discovery in grid computing systems', *15th International Parallel and Distributed Processing Symposium (IPDPS'01)*, pp.910–923.

Mastroianni, C., Talia, D. and Verta, O. (2005) 'A super-peer model for resource discovery services in large-scale Grids', *Future Generation Computer Systems*, Vol. 21, No. 8, pp.1235–1248.

OASIS (2005) 'Web Services Resource Framework (WSRF)', http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-01.pdf.

Wu, X-C., Li, H. and Ju, J-B. (2004) 'A prototype of dynamically disseminating and discovering resource information for resource managements in computational grid', *3rd International Conference on Machine Learning and Cybernetics*, pp.2893–2898.

Zang, T., *et al.* (2004) 'The design and implementation of an OGSA-based grid information service', *Proceedings of the IEEE International Conference on Web Services (ICWS 2004)*, pp.566–575.

Zhang, X. and Schopf, J.M. (2004) 'Performance analysis of the Globus toolkit monitoring and discovery service, MDS2', *IEEE International Performance, Computing and Communications Conference*, pp.843–849.

Zhu, Y. and Zhang, J-L. (2004) 'Distributed storage based on intelligent agent', *3rd International Conference on Machine Learning and Cybernetics*, pp.297–301.