Communications and Network, **2013, 5, 161-168**
http://dx.doi.org/10.4236/cn.2013.52018 Published Online May 2013 (http://www.scirp.org/journal/cn)

A Cluster Based QoS-Aware Service Discovery Architecture Using Swarm Intelligence

E. Christopher Siddarth, K. Seetharaman
Department of Computer Science and Engineering, Annamalai University, Annamalai Nagar, Chidambaram, India
Email: christophersiddarthphd@gmail.com

Received January 21, 2013; revised February 21, 2013; accepted March 30, 2013

ABSTRACT

The existing mobile service discovery approaches do not completely address the issues of service selection and the robustness faced to mobility. The infrastructure of mobile service must be QoS-aware plus context-aware (*i.e.*) aware of the user's required-QoS and the QoS offered by the other networks in user's context. In this paper, we propose a cluster based QoS-aware service discovery architecture using swarm intelligence. Initially, in this architecture, the client sends a service request together with its required QoS parameters like power, distance, CPU speed etc. to its source cluster head. Swarm intelligence is used to establish the intra and inter cluster shortest path routing. Each cluster head searches the QoS aware server with matching QoS constraints by means of a service table and a server table. The QoS aware server is selected to process the service request and to send the reply back to the client. By simulation results, we show that the proposed architecture can attain a good success rate with reduced delay and energy consumption, since it satisfies the QoS constraints.

Keywords: QoS-Aware; Ant Colony Optimization (ACO); Swarm Intelligence; Mobile Ad Hoc Networks (MANETs)

1. Introduction

1.1. Service Discovery

In mobile ad hoc networks, service can be defined as any facility offered by a device which can be handy for any other device. For instance, a service can be a hardware service like a printer which can be used by other devices to print a file. To make full use of these services, a device should be able to discover them in a network. In the same way, they should also call up the services. For this purpose, the role of service discovery protocols comes into play. The service discovery protocols reduce the interaction among users, devices and services in wired network [1] .The devices using the service discovery protocols automatically discover the services. This makes the administration and configuration of the network simple.

The devices are free to move in wireless mobile ad hoc networks. In particular, the service discovery protocols offer methods to discover a particular service, advertise a service, call up a service, and select a service if there is more than one service of the same type available. It also describes a particular service in order to make an easy search.

In the field of service discovery, there is a wide research taking place. In general, there are three types of networks available as per the research in service discovery is considered. They are the wired networks, single hop wireless networks and the wireless multihop mobile ad hoc networks. The difficulty of service discovery is really troublesome in the wireless multihop mobile ad hoc networks. In wired networks there are many service discovery protocols that have being recommended, few of them have attained the industry standards. For instance, Jini by Sun, Universal Plug and Play (UPnP) by Microsoft, Salutation by IBM and Service Location Protocol (SLP) by IETF. In single hop ad hoc networks; there are other protocols like Bluetooth SDP and DEAPSpace. A lot of research is going on multihop Mobile Ad hoc Networks (MANETs). Yet still it has not reached the Industry standards. This is because of the demanding issues due to the unhampered mobility of devices.

A service discovery protocol (SDP) may employ a routing protocol to invoke and obtain a reply from a specific service inherits on a particular device [1]. There are some SDPs that integrate the functionality of routing and service discovery. Thus service discovery and routing

although are nearly related to each other but specifically have distinct characteristics.

1.2. Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO) can be defined as a paradigm for designing metaheuristic algorithms for combinatorial optimization problems. The important attribute of ACO algorithms is the combination of a priori information about the structure of an encouraging result with a posteriori information about the structure of the results obtained earlier. In 1991, AS, the first algorithm on the lines of this framework was presented, since then; many varied variants of the basic principle have been described in the literature. The key fundamental idea, which is heavily motivated by the behavior of real ants, is that of a parallel search over several constructive computational threads, which is on the basis of local problem data and on a dynamic memory structure which contains information about the quality of earlier acquired result. The collective behavior originating from the interaction of the different search threads has resulted to be fruitful in solving combinatorial optimization (CO) problems [2].

In 1995, Gambardella and Dorigo proposed the Ant-Q algorithm, which is an extension of AS that incorporates some concepts from Q-learning, and in 1996 Ant Colony System (ACS) a shortened version of Ant-Q which performed in the same level, measured by algorithm complexity and by computational results. The first algorithm that used an ACO based algorithm to a more common version of the ATSP problem is Hybrid Ant System for the Sequential Ordering Problem (HAS-SOP) [2]. MAX-MIN Ant System (MMAS) is one of the most thriving ACO variants [3]. M. Dorigo and G. Di Caro invented AntNet, which is a routing protocol for packet switched networks. It is a backup routing algorithm for the recognized OSPF protocol, which is based on Ant Colony Optimization (ACO) [4]. AntHocNet is a hybrid routing algorithm for MANETs. AntHocNet's design is on grounds of a specific self-organizing behavior noticed in ant colonies, the shortest paths discovery, and on the related optimization framework of Ant Colony Optimization (ACO) [5].

Cases of ACO have been employed widely to a diversity of discrete combinatorial optimization problems like the Traveling Salesman Problem (TSP), Sequential Ordering, the Network Communication Routing Problem, Vehicle Routing Problem, the Quadratic Assignment Problem (QAP), Job-Shop scheduling, graph Coloring, time tabling, shape optimization, etc. Continuous Ant Colony Optimization (CACO), the first method, was presented by Bilchev and Parmee, and later it was used by some people. Some of the other methods are Asynchronous Parallel Implementation (API) algorithm by Monmarche, and Continuous Interacting Ant Colony (CIAC) by Dreo and Siarry. For reservoir operation, Jalali *et al.* presented Discrete Ant Colony Optimization (DACO) algorithms [6].

The self-organizing systems in nature, for instance, the insect societies exhibit desirable properties. By employing only a few numbers of relatively simple biological agents (namely, the ants) a range of diverse organized behaviors is created at the system-level from the local communications among the agents and with the environment. The strength and efficiency of such collective behaviors relating to variations of environment conditions are the main features of biological success. These kinds of systems are commonly related to the term Swarm Intelligence [7].

1.3. Problem Identification and Proposed Solution

In mobile ad hoc networks, service discovery suffers serious challenges due to the unavailability of a central intelligence agent. The existing service discovery approaches do not completely address the issues of service selection and the robustness faced to mobility. Hence they are very much unsuitable for wireless ad hoc networks. The usage of virtual backbones or clusters for enhanced efficiency and also the quality in MANET service discovery has been considered by the latest research. The infrastructure of mobile service must be QoS-aware plus context-aware (*i.e.*) aware of the user's required-QoS and the QoS offered by the other networks in user's context. The preference of connectivity and adaptation of application protocol parameters can be wisely made on the basis of the existing information about these offered-QoS.

In this paper, we propose a cluster based QoS-aware service discovery architecture using swarm intelligence. In this architecture, at first, the swarm intelligence is used to establish the intra and inter cluster shortest path routing.

The client sends a service request along with its required QoS parameters like bandwidth, power, distance, CPU speed etc. to its cluster head (CH). Each CH searches the QoS aware server with matching QoS constraints. Then the CH forwards the request to other cluster heads using the swarm's forward and backward agents. Finally the QoS aware server is selected to process the service request and to send the reply to the client. Since our proposed architecture is QoS aware and satisfies the QoS constraints, it can attain a good success rate with reduced bandwidth, delay and energy consumption.

2. Related Work

Zhenguo Gao *et al.* [8] have proposed Forward Node

Minimization enhanced Group-based Service Discovery Protocol (FNMGSDP) to minimize the number of next hop nodes when forwarding request packets by exhaustively utilizing the information in Service Information Cache.

Roshni Neogy *et al.* [9] have proposed a novel service discovery protocol using mobile agents for MANET with reliable performance for a given MANET configuration. In this protocol, the agents select their route dynamically and exchange service information with the nodes in order to speed up the process. Smooth Random Mobility Model (SRMM) is used to estimate node location at a particular time. The experimental results show the robustness of their proposed scheme.

Apostolos Malatras *et al.* [10] have presented here a modular framework for the adaptive reconfiguration of such mechanisms in order to address the diverse and evolving needs of users and applications. This approach is based on ant-inspired, self-organized overlays and exploits context information and policies to manage the monitoring needs of the latter in an autonomic fashion, satisfying continuously changing requirements and leading to optimal operation.

Kaouther Abrougui *et al.* [11] have presented a QoS aware location-based service discovery protocol for vehicular networks to provide load balancing on service providers, and routing paths between service providers and service requesters. It gives the selection of service providers and routing paths between service providers and service requesters that satisfy some performance attributes specified by service requesters.

Jihen Drira Rekik *et al.* [12] have proposed a cross-layer service selection which combines performance metrics measured in the real-time database system to those used by the routing protocol in order to make the best selection decision. It ensures both timeliness and energy efficiency by avoiding low-power and busy service provider node. A multicast packet is used in order to reduce the transmission cost and network load when sending the same packet to multiple service providers.

Cynthia Jayapal and Sumathi Vembu [13] have presented an adaptive service discovery protocol that enhances the performance of service discovery which uses an adaptive core node election mechanism that changes whenever the load increases and is also robust against network failures. They have used a distributed directory based service discovery mechanism that operates in a proactive mode with service advertisements to the core node and selects a provider based both on distance and service capability of the provider.

T. Rajendran *et al.* [14] have proposed a novel technique of using Multi-Agents for Web Services discovery. This novel approach of applying the Multi-Agent System architecture effectively supports Web Services discovery with QoS registration, verification, certification, and confirmation. The Multi-Agents used in this approach are Response Agent, Certification Agent and Query Agent. These agents perform specific functions which provide significant selection of Web Services. There are several unique characteristics of these multi agents that are not supported by existing approaches dealing with QoS for Web Services.

3. Swarm Based Routing

The self-organizing systems in nature, for instance, the insect societies exhibit desirable properties. By employing only a few numbers of relatively simple biological agents (namely, the ants) a range of diverse organized behaviors is created at the system-level from the local communications among the agents and with the environment. The strength and efficiency of such collective behaviors relating to variations of environment conditions are the main features of biological success. These kinds of systems are commonly related to the term Swarm Intelligence [7].

The swarm based routing uses two kinds of agents—Forward and Backward Agents. The forward agents search the network to gather the network traffic details and are routed on normal priority queues. The forward agent is replaced by the backward agent when it reaches the destination and the backward agent takes over the stack enclosed in the forward agent. The backward agent is deterministic and is transmitted on a queue with high priority. The backward agents follow the path of the forward agent and use these details to update the routing tables regularly. These mobile agents are very minute and weightless packets. They contain IP addresses of the source and destination, packet ID, node ID and traveling time.

By employing single hop HELLO message packets, a neighbor list is developed by the node and these packets are forwarded regularly. By allotting a probability value at each node the routing table is initialized at every node. The source generates the forward agents periodically. The forward agents may be unicasted or broadcasted according to the existence of the route. The agent is considered to be unicast if the route is available at the node or otherwise, the agent is broadcast. While progressing towards the destination, the forward agent moves the node ID and the traveling time of every visited intermediate node its stack until it finally arrives at the destination. Then it passes its entire route details to the backward agent and dies off. The backward agents are created by the respective nodes.

The swarm based routing sets the path in a proactive way. The backward agent supports the forward agent in its selection of the discovered path.

4. System Design and Service Discovery Architecture

4.1. Service Discovery Architecture

Setting up a common architecture for service discovery and service selection takes up the thought of numerous aspects. In general, the architecture must be able to take corrective steps by broadcasting service information across the network, matching service discovery requirements with advertisements, offering the application with correct information for the selection of the most exceptional server among those accessible and tracking network topology variations and notifying the application. In the end, the methods for efficient service discovery and selection must be provided by the architecture.

The architecture comprises of two main agents:

1) A Service Discovery Agent (SDA), which is independent of the routing protocol;

2) A Swarm Routing Agent (SRA), which is responsible for routing activities to propagate service discovery messages and trace topology changes.

Service Discovery Agent (SDA): It bestows a regular view of the service discovery mechanism to the client applications and service providers. It saves the information about the services in a table known as SerTable. The table contains the following fields:

- Service description;
- Service location;
- Minimum hop count from the source to the provider.

It also maintains the QoS details of all providers in a server table, which contains the following fields:

- CPU speed of the server;
- Capacity of the server;
- Application specification;
- Available Power of the server;
- Work load of the server.

The service discovery requests of the clients and service advertisements of the servers are controlled by the SDA. To advertise the service discovery requests and transmit the service discovery replies, SDA makes use of the swarm routing agent. SDA invokes an application-specified callback function and notifies the applications, when the service table is changed.

The architecture also consists of a Matching table, which stores the matched entries obtained from the SerTable and server table.

The exact service lookup and matching methodology is based on our previous work [15]. It consists of a middleware framework, which is based on ontology. The middleware framework for context aware service discovery makes use of contextual ontologies in order to permit the serving of semantically enriched contextual requests for services, without restricting the predefined contextual types or values.

4.2. Service Discovery Modes

SDA gives two service discovery modes: Client based discovery and Server based discovery.

In client based discovery, the swarm agent forwards the client request to its source cluster head (SCH). SDA searches its local SerTable and Server table for the respective service description fields within its cluster. If SDA finds a match, it adds the details in the Matching table. Then the swarm agent forwards the same client request to other cluster heads (CHs). Each and every CH searches its respective local SerTable and Server table for the service description fields within its cluster. Now, after finding the matched entries, they are then returned to the SCH and SCH add those entries into the Matching table.

All the entries in the Matching table are then sorted in order to find out the QoS aware server. The sorting is done in ascending order based on the following fields:

- Distance (hopcount);
- Available power of the server (avail_power);
- Work load of the server (wload).

After sorting, it then lookup for the most relevant QoS fields of those servers like CPU.

Speed of the client, Capacity of the client and Application specifications, matching the QoS request of the client.

After the selection of the best server, SDA asks SRA to transmit the service response back to the client.

In Server based discovery, SDA asks SRA to periodically disseminate the service advertisements from the servers for a specific service.

4.3. Format of Packet Headers

We used swarm based routing protocol by adding fields to the standard route request and reply packet headers.

We have added the following fields to the route request packet:

- Service description, which contains a description of the service to be discovered;
- Service discovery flag, which is set when the service description field is nonempty;
- CPU speed of the client;
- Capacity of the client;
- Application specifications;
- Power of the client;
- Work load of the client.

We have added the following fields to the route reply packet:

- Service description, which contains a description of the service as advertised by the service provider;
- Service discovery flag;
- Service location, which contains the location of the service;

- Service metric which contains additional fields as advertised by a service provider.

4.4. Clustering Algorithm

In this paper, the clustering process is done by employing the clustering mechanism of AWCBRP [16] and this can be used in the service discovery mechanism. AWCBRP is an adaptive weighted cluster based routing protocol for mobile ad-hoc networks which adapts quickly to the topological changes and establishes the routing efficiently. In AWCBRP, the cluster head selection is performed by assigning a combined weight value based on the factors Energy Level, Connectivity and Stability.

5. Client and Server Based Service Discovery

5.1. Client Based Service Discovery

Consider a network with clusters C_1, C_2, ⋯ C_n wih cluster heads CH_1, CH_2, ⋯ CH_n. Each server S_i, i = 1, 2 ⋯ M, contains services Sr_j, j = 1, 2 ⋯ K. Each cluster head CHk, contains the service table $SerTable_k$ and server table $ServerTable_k$, k = 1, 2, ⋯ n.

1) When a client C1 from the cluster C_1, wants a service Sr_j, it sends a service request to the cluster head CH_1.

2) The SDA at CH_1 first searches the local service table $SerTable_1$, for the matching description.

3) If it is found, the matching entries of $SerTable_1$ and $ServerTable_1$ will be added in the MatchingTable.

4) The SDA then calls the SRA to forward the service discovery request (SDREQ) packet to the all other cluster heads CH_j, $j \neq 1$. The packet is a modified route request with the service description field populated and the service discovery flag set.

5) On receiving the SDREQ packet, the cluster heads CH_j will check their $SerTable_j$.

6) If their corresponding SDA matches the service description in the packet with the data found in SerTablej, the corresponding server entries from ServerTablej are also matched.

7) SRA generates a service discovery reply SDREP, which is a modified route reply packet, populates all the required matching entries and unicast the packet back to CH_1.

8) CH_1 collects all the matching entries from the SDREP packets send by the cluster heads CH_j.

9) CH_1 then appends these matching entries into the MatchingTable.

10) The MatchingTable is then sorted in the ascending order of the fields hopcount, avail_power and wload.

11) From the top of the sorted table, the SDA then lookup for those servers with most relevant QoS fields matching the QoS request of the client.

12) If the QoS aware server is found, SDA asks SRA to transmit the service response back to the client.

The Service request and reply propagation procedures are illustrated in **Figures 1** and **2** respectively.

5.2. Server Based Service Discovery

If a new service is added or an existing service description changes, the SDA instructs the SA to send a service advertisement (SERADVT) packet to each of the cluster heads. For example, suppose a new service Srp, (p > K) is added for a server Sk, then SRA construct the SERADVT packet, which contains the service description

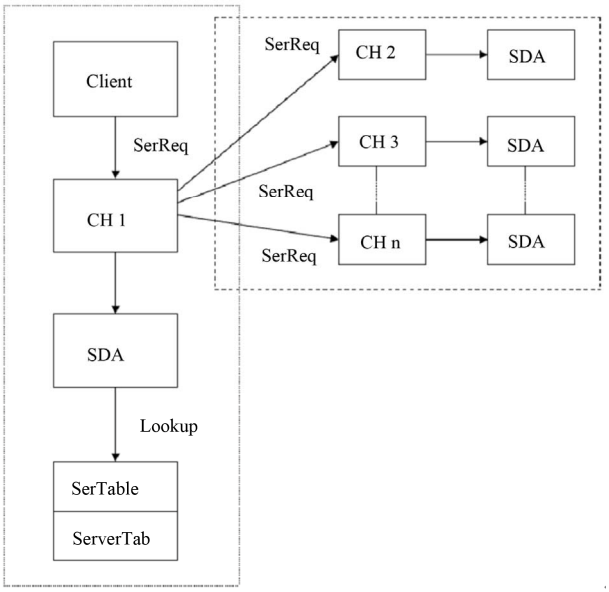

Figure 1. Service request propagation.

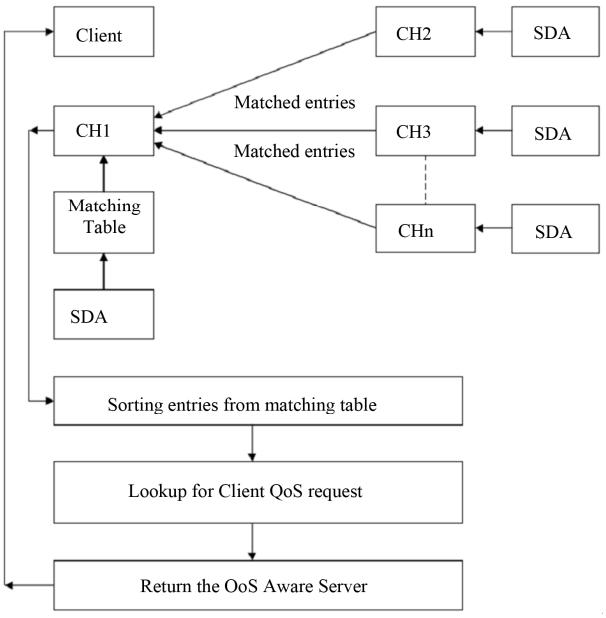

Figure 2. Service response propagation.

and the server id , and broadcast to all the cluster heads CH_i, i = 1, 2 ⋯ n. Then the corresponding SerTable of each cluster head is updated by the SDA.

6. Simulation Results

6.1. Simulation Model and Parameters

We use NS2 [17] to simulate our proposed protocol. In our simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. We use the distributed coordination function (DCF) of IEEE 802.11 for wireless LANs as the MAC layer protocol. It has the functionality to notify the network layer about link breakage. In our simulation, 100 mobile nodes move in a 1000 meter × 1000 meter region for 100 seconds simulation time. Among the total 100 nodes, we treat 20 nodes as servers. The clients that send service requests to the server are varied as 2, 4 ⋯ 10.

Initial locations and movements of the nodes are obtained using the random waypoint (RWP) model of NS2. We assume each node moves independently with the same average speed. All nodes have the same transmission range of 250 meters. In our simulation, the speed of the mobile is varied from 5 m/s to 20 m/s. We have taken the Service Location Protocol (SLP) for service discovery. A SLP service agent is attached to the servers for providing the services and SLP user agent is attached to the clients for requesting the service.

Our simulation settings and parameters are summarized in **Table 1**.

6.2. Performance Metrics

We compare our QoS-aware service infrastructure with the Real-Time Database QoS-aware Service selection (RTDQS) protocol [12] and non-QoS aware service infrastructure. We evaluate mainly the performance according to the following metrics.

Table 1. Simulation parameters.

No. of Nodes	100
Area Size	1000 × 1000
Mac	802.11
Radio Range	250 m
Simulation Time	100 sec
Service Discovery Protocol	SLP
Server Application	SLPsa
Client Application	SLPua
Number of Servers	20
Number of Clients	2, 4, 6, 8 and 10
Speed	5, 10, 15 and 20 m/s
Routing Protocol	SRA

Average Delay: It is measured as the average delay occurred for each client while getting the requested service.

Success Ratio: It is the ratio of the number of services received successfully and the total number of service requests.

Average Power Consumption: The average energy consumed by clients and the average power consumed by servers are calculated.

6.3. Results

6.3.1. Based on Clients

In our first experiment, the number of clients requesting services, is varied as 2, 4, 6 ⋯ 10.

Figure 3 shows the results of average success ratio for varying the clients. Clearly our QoS-Aware scheme outperforms the non QoS scenario and achieves 10% more success ratio than the RTDQS protocol.

Figure 4 shows the results of average end-to-end delay for both the schemes. From the results, we can see that QoS-Aware scheme outperforms the Non QoS-Aware scheme by attaining low delay and 46% less delay than RTDQS protocol.

Figure 5 shows the results of energy consumption for both the schemes. From the results, we can see that QoS-Aware scheme outperforms the Non QoS-Aware scheme by attaining low energy consumption and reduces 4.7% less energy when compared to RTQS protocol.

6.3.2. Based on Speed

To evaluate the impact of mobility on the performance,

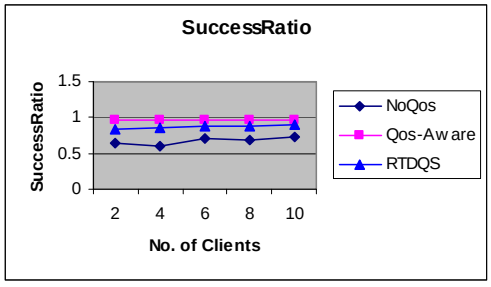

Figure 3. Clients vs success ratio.

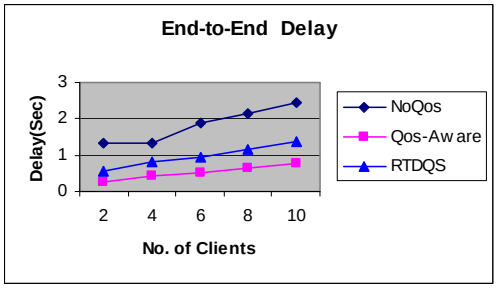

Figure 4. Clients vs delay.

in the next experiment, the speed is varied from 5 m/s to 20 m/s.

Figure 6 shows the results of average success ratio for varying the speed. As we can see from the figure, the success ratio is very low for Non QoS-aware scheme. But QoS-Aware scheme achieves 12% more success ratio than the RTDQS protocol. In both the schemes, the success ratio begins to reduce, as the speed increases.

Figure 7 shows the results of average end-to-end delay for both the schemes, when the speed is varied. For all the schemes, the delay increases, when the speed increases, because in high speeds, route maintenance frequently occurs. But, we can see that QoS-Aware scheme has less 33% less delay when compared to the RTDQS protocol.

Figure 8 presents the results of energy consumption for all the schemes, when the speed is varied. From the results, we can see that QoS-Aware scheme outperforms RTDQS protocol by attaining 2.3% lower energy consumption.

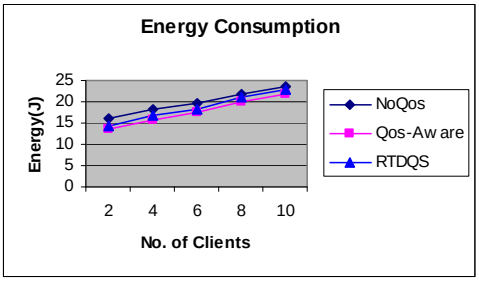

Figure 5. Clients vs energy.

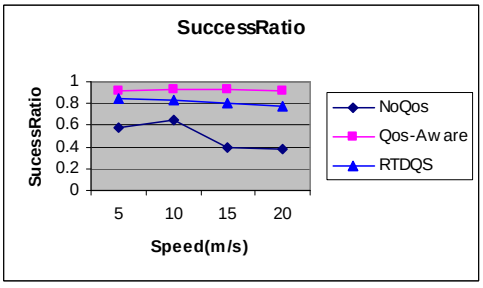

Figure 6. Speed vs success ratio.

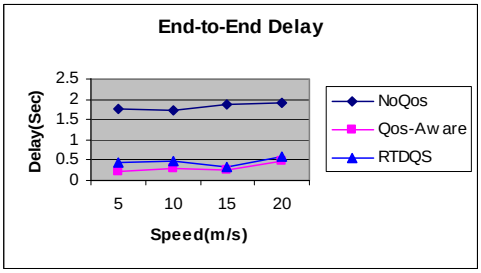

Figure 7. Speed vs delay.

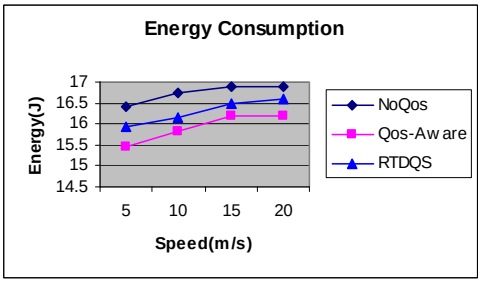

Figure 8. Speed vs energy consumption.

7. Conclusion

In this paper, we have proposed a cluster based QoS-aware service discovery architecture using swarm intelligence. Initially, in this architecture, the client sends a service request together with its required QoS parameters like power, distance, CPU speed etc. to its source cluster head. Swarm intelligence is used to establish the intra and inter cluster shortest path routing. The source cluster head searches the QoS aware server with matching QoS constraints by means of a local service table and server table. Then the source cluster head forwards the request to other cluster heads using the swarm's forward and backward agents and the search for the QoS-aware server is continued. After finding a list of servers, the response is forwarded to a matching table in the source cluster head, where a sorting takes place. Finally the QoS aware server is selected to process the service request and to send the reply back to the client. Simulation results have shown that our proposed QoS aware service discovery architecture satisfies the QoS constraints and achieves a good success rate with reduced delay and energy consumption.

REFERENCES

[1] F. Outay, V. Veque and R. Bouallegue, "Survey of Service Discovery Protocols and Benefits of Combining Service and Route Discovery," *IJCSNS International Journal of Computer Science and Network Security*, Vol. 7, No. 11, 2007.

[2] V. Maniezzo, L. M. Gambardella and F. de Luigi, "Ant Colony Optimization," Optimization Techniques in Engineering, Springer-Verla, Berlin, 2004.

[3] I. Alaya, C. Solnon and K. Ghedira, "Ant Colony Optimization for Multi-objective Optimization Problems," 19*th IEEE International Conference on Tools with Artificial Intelligence*, Vol. 1, 2007, pp. 450-457. doi:10.1109/ICTAI.2007.108

[4] V. Verstraete, M. Strobbe, E. Van Breusegem, J. Coppens, M. Pickavet and P. Demeester, "AntNet: ACO Routing Algorithm in Practice," *Proceedings of the* 8*th INFORMS Telecommunications Conference*, Dallas, 2006.

[5] G. Di Caro, F. Ducatelle and L. M. Gambardella, "Ant-

HocNet: An Adaptive Nature-Inspired Algorithm for Routing in Mobile Ad Hoc Networks," *European Transactions on Telecommunications*, Vol. 16, No. 5, 2005, pp. 443-455. doi:10.1002/ett.1062

[6] M. Hadi Afshar, H. Ketabchi and E. Rasa, "Elitist Continuous Ant Colony Optimization Algorithm: Application to Reservoir Operation Problems," *International Journal of Civil Engineering*, Vol. 4, No. 4, 2006.

[7] B. D. Shirodkar, S. S. Manvi and A. J. Umbarkar, "Multicast Routing for Mobile Ad-Hoc Networks," *International Journal of Recent Trends in Engineering*, Vol. 1, No. 1, 2009.

[8] Z. G. Gao, L. Wang, M. Yang and J. P. Wang, "FNMG-SDP: An Optimized Group-Based Service Discovery Protocol for MANETs," Springer Science Business Media, LLC, 2009.

[9] R. Neogy, C. Chowdhury and S. Neogy, "A Reliable Service Discovery Protocol Using Mobile Agents in MANET," *IEEE Proceedings of Reliability and Maintainability Symposium* (*RAMS*), 23-26 January 2012, pp. 1-7.

[10] A. Malatras, F. Peng and B. Hirsbrunner, "A Self-Management Framework for Efficient Resource Discovery in Pervasive Environments," *ICAC* 8*th International Conference on Automatic Computing*, Karlsruhe, 14-18 June 2011.

[11] K. Abrougui, A. Boukerche and H. Ramadan, "Efficient Load Balancing and QoS-Based Location Aware Service Discovery Protocol for Vehicular Adhoc Networks," *EURASIP Journal on Wireless Communications and Networking*, Vol. 2012, 2012, p. 96. doi:10.1186/1687-1499-2012-96

[12] J. D. Rekik, L. Baccouche and H. Ben Ghezala, "Real-Time Database Qos-Aware Service Selection Protocol for Manet," *International Journal of Database Management Systems* (*IJDMS*), Vol. 3, No. 4, 2011.

[13] C. Jayapal and S. Vembu, "Adaptive Service Discovery Protocol for Mobile Ad Hoc Networks," *European Journal of Scientific Research*, Vol. 49, No. 1, 2011, pp. 6-17.

[14] T. Rajendran and P. Balasubramanie, "An Efficient Multi-Agent-Based Architecture for Web Service Registration and Discovery with QoS," *European Journal of Scientific Research*, Vol. 60, No. 3, 2011, pp. 439-450.

[15] E. Christopher Siddarth, "A Context Aware Ontology Based Middleware Framework for Service Discovery", *IJCSNS International Journal of Computer Science and Network Security*, Vol. 9, No. 3, 2009.

[16] S. Karunakaran and P. Thangaraj, "An Adaptive Weighted Cluster Based Routing (AWCBRP) Protocol for Mobile Adhoc Networks," *WSEAS Transactions on Communications*, Vol. 7, No. 4, 2008.

[17] Network Simulator, http://www.isi.edu/nsnam

