A New Approach to Service Discovery in Wireless Mobile Ad Hoc Networks

Yu Yang, Hossam Hassanein, and Afzal Mawji

Telecommunications Research Lab, School of Computing, Queen's University

Kingston, Ontario, K7L 3N6, Canada

{yangy, hossam, mawji}@cs.queensu.ca*

***Abstract*—Service discovery, essential for many wireless applications, is more difficult to achieve in Mobile Ad hoc NETworks (MANETS) than in both wired and traditional wireless networks due to the lack of central control. In addition, the heterogeneity, mobility, and limited energy of the devices precludes the use of traditional service discovery protocols. This paper presents HESED, a fundamentally different service discovery protocol based on multicast query and multicast reply. Clients multicast their service query and matching servers multicast their response to all nodes. This service information is cached by all, reducing the number of queries. HESED uses only symmetric links, providing reliability for its forwarding algorithms. Simulation results show that HESED significantly outperforms a traditional on–demand service discovery algorithm.**

I. Introduction

A Mobile Ad hoc NETwork (MANET) does not use any existing infrastructure or central administration to organize the wireless devices within it. The nodes communicate in multi–hop peer–to–peer mode.

As wireless devices become commonplace, new services are made available for their use. These services include computing resources and services, and also network resources, such as Internet gateways, name servers, database servers, printers, directory agents, etc.

Since many applications make use of services, the performance of the service discovery scheme can greatly affect the performance of the application. However, the lack of central control, device mobility and heterogenity, and limited device energy make service discovery very difficult in MANETs.

This paper presents HESED, a High Efficiency SErvice Discovery protocol that approaches service discovery in a fundamentally different manner than previously. In HESED, both service discovery queries *and* service replies are multicast. Clients join service multicast groups and send all multicast messages to these groups. All nodes in the group cache received service information for future use. When a client requires a service, it first examines its local cache, and if found, calculates the likelihood that the route to the service is still valid. This can significantly reduce the number of service discovery queries, which in turn reduces service delay, and energy consumption. HESED employs a cross–layer design, using application layer information to help make routing decisions.

HESED is a significant step forward in the development of service discovery protocols for MANETs. In addition to the cross–layer design, HESED employs a cache system, and has low delay and a message complexity of $O(N)$ for N-node MANETs, as opposed to $O(N^2)$ for traditional service discovery protocols [1]. Simulation results confirm that HESED requires significantly fewer packets and has a much shorter delay than a traditional on–demand service discovery protocol. HESED is presented in detail in the next section. Section 3 presents simulation results and discusses some scenarios executed with a Java–based implementation of HESED. Section 4 compares some service discovery protocols with HESED. Section 5 presents our conclusions.

*This research has been supported by a grant from Communications and Information Technology Ontario under the Champions of Innovation Program

II. HESED Architecture

Service discovery is initiated in HESED by having clients first search their cache for service information. If the service cannot be found, or if the cached information is no longer valid, the client multicasts a query. Matching servers multicast their information in response, which intermediate nodes cache. Multicast packets are relayed via Edge Node Forwarding, which uses only symmetric links based on information provided by Long Beacon Neighbour Detection. The client then contacts a server via Backward Learning Routing. These algorithms are the components of HESED and are discussed further in this section.

A. Multicast Query and Multicast Reply

HESED is fundamentally different from other post–query strategies [2] because it uses the ideas of query–all *and* reply–all. Clients send multicast queries to all nodes in the service multicast group and service providers then multicast their response to the group. All nodes receiving the reply cache the service information locally.

Clients may receive more than one reply if multiple servers are available. The client chooses the most desirable server and sends a unicast message to it, confirming that server's selection. The server then sends the client an acknowledgment (ACK), which includes detailed information about the service and how to access it. Figure 1 illustrates the exchanges between client and server. All packets use a pre–defined XML template.

Nodes between the client and server(s) do not send a reply to the client if they have relevant information in their cache. There are several reasons for this. Firstly, since there may be several possible candidate servers, it is difficult for an intermediate node to select the best one on behalf of the client. If the intermediate node replies to the query instead of forwarding it, clients may lose some service information. Secondly, many intermediate nodes will send service information to client, resulting in many duplicate responses. Thirdly, each intermediate node would first need to check whether the services in its

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

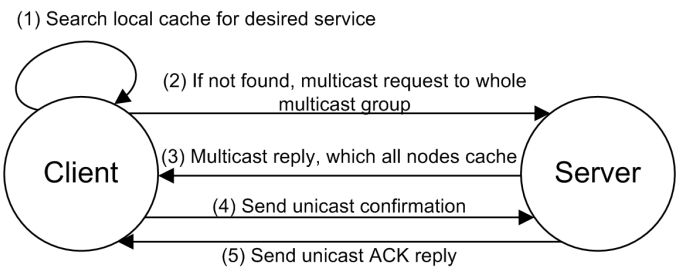

Fig. 1. The Multicast Query and Multicast Reply algorithm

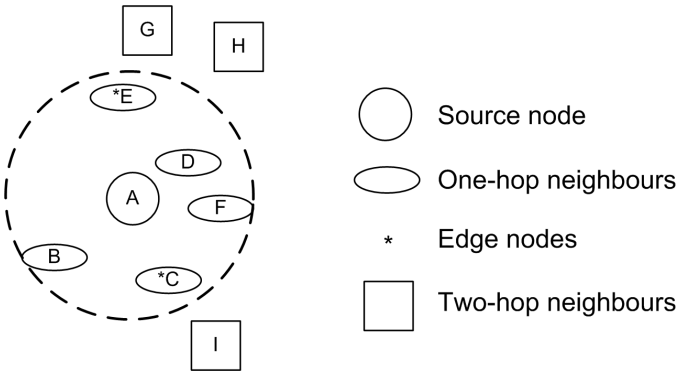

Fig. 2. Edge node set

cache could be matched with the client's query, resulting in additional processing delay and extra energy consumption at each relay node. Fourthly, some cached information may be outdated, leading to not only misinforming, but also to providing conflicting information about services to clients.

B. Long Beacon Neighbour Detection

Long Beacon Neighbour Detection attempts to organize the ad hoc network in an efficient manner to aid the Edge Node Forwarding algorithm. Each node periodically sends a beacon message to all neighbours in order to maintain local topology information. The beacon message includes information about the node's neighbours within one hop, so each node in the network knows about all nodes within two hops of it.

Communication ranges differ among the nodes of the network due to the heterogeneous nature of ad hoc networks. This results in asymmetric links, a major source of unreliability in wireless connections. By avoiding asymmetric links, reverse route discovery is made easier, resulting in better performance due to the shorter delay.

By considering only those nodes connected via symmetric links to be neighbours of one another, asymmetric links are avoided. For example, if node A receives a beacon message from node B, node A then lists node B as a uni–directional neighbour and puts this information in its beacon message. After node B receives node A's beacon, node B finds that it is listed as a uni–directional neighbour. This tells node B that there is a symmetric link between node A and itself. Node B then lists node A as one of its bi–directional neighbours in future beacon messages. Eventually, node A will also realize that node B is a bi–directional neighbour.

C. Edge Node Forwarding

The Edge Node Forwarding algorithm forwards multicast messages created by the Multicast Query and Multicast Reply Algorithm. From the Long Beacon Neighbour Detection algorithm, every node knows of all nearby nodes within a two–hop distance. The one–hop neighbours are separated into two sets: the edge node set, which is the minimal set of nodes that can cover the entire two-hop set, and the remaining nodes, which are called the internal node set. In Edge Node Forwarding, a node asks only its edge nodes to forward the multicast packets, while internal nodes do not forward any packets.

Figure 2 illustrates the operation of the Edge Node Forwarding algorithm. The dashed–line circle is node A's communication range. A has five one–hop neighbours (B–F) and 3 two–hop neighbours (G–I). Nodes C and E can cover all of the two–hop neighbours, so C and E are the edge nodes and B, D, and F are the internal nodes. If A want to send a multicast packet, it asks only C and E to relay it.

Edge nodes are selected by a greedy algorithm. The sending node sorts all one–hop neighbours by the number of one–hop neighbours that that node has, and the sender selects edge nodes from the beginning of the list until the edge node set covers all two–hop set nodes. The edge node selection criteria could be changed, for example, to select those nodes with the minimum response time, or those with maximum remaining energy.

D. Backward Learning Routing

HESED integrates routing together with the service discovery process. Multicast packets contain the source node's IP address as well as the previous node's IP address. These two addresses are stored in the routing table and updated whenever a multicast packet is forwarded. This is similar to the backward learning technique in [3]. Because only symmetric links used, the route between source and destination is guaranteed to be bi–directional.

E. Caching

From the Long Beacon Neighbour Detection Algorithm, each node has some knowledge about the mobility of its neighbours, so it can calculate T_α, the average time taken for half of a node's neighbours to change. This is used when evaluating the validity of a cached multihop route to a server.

Let r_m be the ratio of the number of a node's remaining neighbours to the number of all of its neighbours after time T_m. Therefore $r_m = \frac{|S_1 \cap S_2|}{|S_1|}$ where S_1 is the neighbour node set at time T_0, and S_2 is the neighbour node set at time T_m. Since T_α is the time for half of the neighbours to change, r_m = 0.5 at time T_α. If we assume that neighbouring nodes leave at a fixed rate, then the average leaving rate r_l per time unit is $r_l = \frac{\frac{1}{2}}{T_\alpha} = \frac{1}{2T_\alpha}$.

Let the number of remaining neighbours be r_n, so that $1 - r_n$ nodes have left the neighbourhood. The fraction of nodes at time t is then $t = \frac{1-r_n}{r_l} = 2T_\alpha(1 - r_n)$. The proportion of remaining neighbours after time t, defined as the Connection Validation Probability (CVP), is given by CVP = $r_n = 1 - \frac{t}{2T_\alpha}$.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

Let the Route Validation Probability (RVP) be the probability that a complete route from source to destination is still valid after time t. The RVP depends on the CVPs of all of its intermediate nodes. Therefore, for two hops, $\text{RVP} = (1 - \frac{t}{2T_\alpha^{(1)}}) \times (1 - \frac{t}{2T_\alpha^{(2)}}) \approx (1 - \frac{t}{2T'_\alpha})$, where $T'_\alpha = \frac{T_\alpha^{(1)} T_\alpha^{(2)}}{T_\alpha^{(1)} + T_\alpha^{(2)}}$.

T'_α is called the accumulated half neighbor changing time and is maintained at each intermediate node so that T_α can be obtained for the entire route.

When a client receives a service reply, only basic service information and the route to the server are cached, so a large cache is not necessary. If the cache is full, the RVP for all services in the cache is calculated. The cached service with the minimum RVP is replaced with the new information.

III. Simulation and Implementation

This section evaluates the performance of HESED by comparing it to traditional on–demand service discovery (TOSD) in a simulation. In HESED, client nodes first check whether there is cached information available when they need a service. In the event of a cache hit, a route validation check is started to determine if the route can be reused. A new service discovery process is started if the route is invalid. The cache rate is not fixed at a pre–specified value. Instead, we attempt to model a real MANET and determine the natural cache hit rate. In the TOSD scheme, clients use flooding mode to broadcast the service request and the service providers send a unicast reply after they receive the query. The performance of HESED is tested under three patterns: nodal density, field size, and mobility. We also test varying network traffic for the three scenarios.

A. Simulation Parameters and Metrics

The simulated MANETs have areas of 1000–1600 m in length and 300–480 m in width, and contain 15–63 nodes. All nodes are evenly distributed in the simulation area and service discovery queries are generated at each node according to a Poisson distribution with an arrival rate of 0.1–0.19 queries per second. This query rate was set according to that seen in wired networks, where, when a user is surfing the Internet, many web pages with several objects are requested. Due to the mobility of MANETs, each webpage object should be considered as a new service request, even if they come from the same client.

Transmission rates of 56 Kbps, 11 Mbps, and 54 Mbps with probabilities 20%, 40% and 40% respectively were assumed. The communication range for each node is a Gaussian random variable whose mean value is 200 m and variance is 50 m. The packet processing time for each node is evenly distributed from 1–100 ms. Each node sends the neighbour detection beacon at 3 s intervals. The random waypoint model [4], [5] is used for mobility with nodal velocities distributed according to a Gaussian distribution with mean 5 m/s and variance 1 m/s. The pause time is two seconds and nodal movement time is randomly selected from 1000–5000 ms.

We adopt a simplified multiple access control (MAC) scheme because the IEEE 802.11 MAC protocol does not define the MAC mode for broadcasting, and more than 50% of the packets in HESED are multicast packets. In our simulation, we assume there is a perfect MAC protocol, meaning a node can send a packet as soon as the medium is available and there is no communication overhead, such as RTS/CTS, etc. Therefore, the multicast query and multicast reply algorithm must contend for channel access, as with any other traffic. All clients in the simulation are part of the same multicast group.

Three simulation patterns are used. In pattern A (nodal density pattern), the average velocity and the simulation area are fixed, while the nodal density is increased from 0.00005–0.00014 nodes/m^2 and the service discovery query rate from 0.1–0.19 queries per second per node. This pattern determines performance with high network traffic because not only is the query rate being increased, but the number of nodes within a fixed area is also being increased, thereby resulting in significantly more traffic.

In pattern B (field size pattern), the nodal density is fixed, while the simulation field is changed from 1000 m $\times$ 300 m to 1600 m $\times$ 480 m, and the service discovery rate is varied from 0.1–0.19 queries per second per node. Since the nodal density does not change, the number of nodes increases when the simulation field becomes larger, therefore this pattern determines the performance when one server must provide services to a large area.

In pattern C (mobility pattern), the average velocity is changed from 0–6 m/s and the service discovery query rate is also changed from 0.1–0.19 queries per second per node, so this pattern evaluates the performance under different nodal mobilities.

For each pattern, three performance metrics are determined: average delay for successful queries, average number of packets for successful queries, and cache hit rate. In HESED, the delay includes the time for both a multicast service request and for a multicast service reply, while in TOSD it includes one multicast service request and a unicast service reply. The number of packets in HESED includes two multicast communications and the cost of beacon messages, while in TOSD it includes one multicast communication and one unicast communication. The cache hit rate shows how often the cached service information can be used by a client node in place of a service query, and only concerns HESED.

B. Simulation Analysis

Delay is a critical performance metric for any service discovery scheme because it captures the system's responsiveness to the user. Figure 3 demonstrates that the average delay for HESED is much lower than the TOSD scheme in all cases. Furthermore the delay for HESED does not change significantly in any of the test patterns. In the TOSD scheme, the delay tends to increase with the number of nodes and the nodal velocity.

The number of packets is directly related to energy consumption, so lowering this is very important in MANETs since energy is limited. Figure 4 is a comparison of the number of packets per query for the HESED and TOSD schemes. Though both HESED and TOSD need nearly the same number of packets for a 20 node MANET, as the number of nodes increases, TOSD requires significantly more packets. HESED uses Edge Node Forwarding, which only requests selected nodes to send packets, so the average number of packets hardly changes as the number of nodes increases. In the TOSD scheme the average number of

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

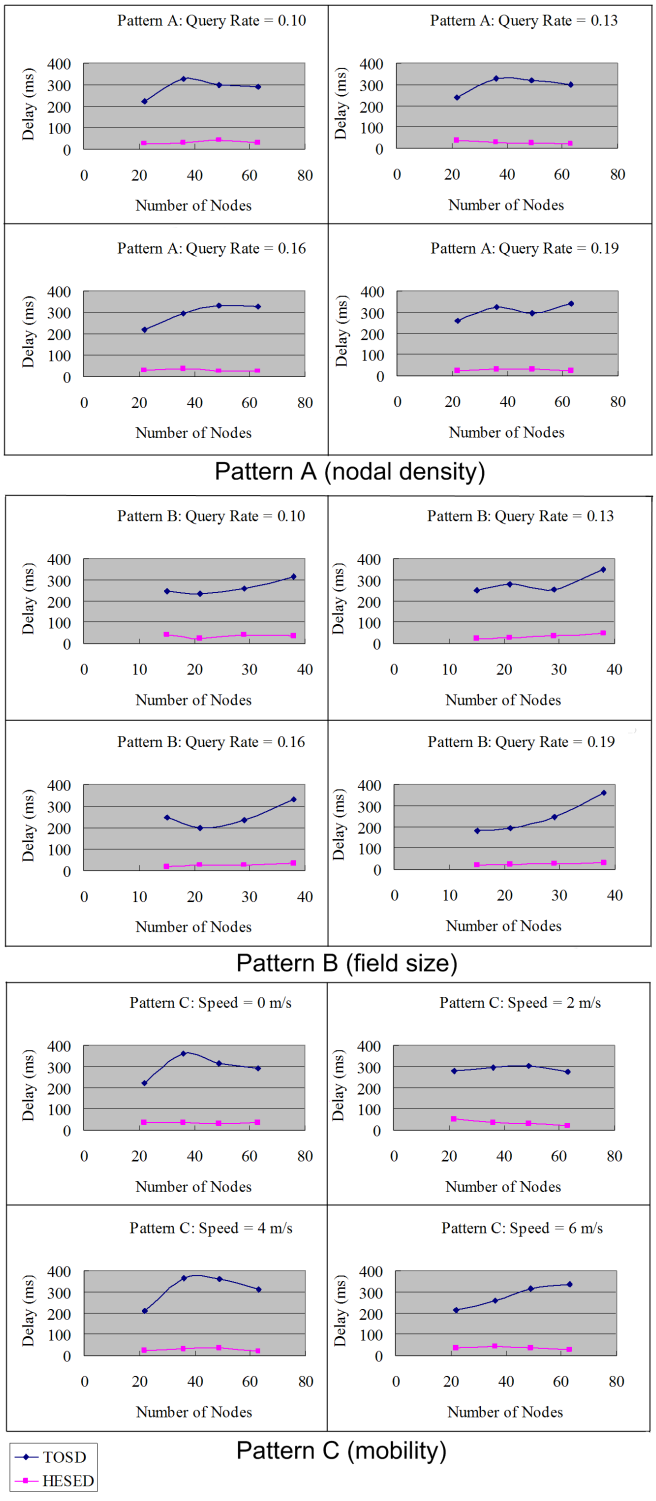

Fig. 3. Comparison of delay

packets always increases with the number of nodes.

The cache hit rate is the ratio between the number of queries using the cache and the total number of queries and gives an indication of how many service discovery queries caching prevents from being made. Since the server node always multicasts its information to all clients, those nodes which are closest to the server receive the latest information and are most up–to–date.

There are two events which cause queries. One is that a node's cached information expires before it needs service, and the other occurs when a node enters the server node's connectivity area and has not received any service information yet. The first event usually occurs when the node is far from the server and the lifetime for the cached information is short. This kind of event increases in frequency when the query rate is low. In such cases, the interval between two service responses is large and there is a greater probability that a node's cached information expires. This can be seen in Figure 5 for patterns A and B, in which the cache hit rate increases with the number of nodes. An increasing query rate also has a positive effect on the cache hit rate.

When nodal mobility is high, the likelihood of the second event increases since more nodes would enter and leave the server's connectivity area during the simulation time period. Thus the cache hit rate decreases when the average velocity increases as seen in pattern C. The cache hit rate remains, however, at respectable levels.

C. HESED Implementation

HESED has been implemented as a Java application and several scenarios were designed to test its abilities. Since it is application layer software, packets go through the operating system and so the performance depends greatly on the OS. Hence, no specific performance data was measured and instead the focus was on proving the feasibility and correctness of the algorithm.

The first scenario tested the asymmetric link detection function of the Long Beacon Neighbour Detection algorithm. The next scenarios tested data transport reliability over both short and long distances. These tests were all successful, but in all of them, there was only one route for communication, so the next scenario tested whether the routing algorithm would choose a proper route from among several alternatives. The test results showed that where possible, a better path was selected, resulting in greater performance than the previous tests. In the final scenario, several concurrent FTP transfers were forced through a single bottleneck node. All transfers were successfully completed with no packet loss.

IV. Existing Service Discovery Protocols

This section examines some agent–based service discovery protocols, including SLP [6], Jini [7], UPnP [8], and some others designed for MANETs. In general, the agent–based algorithms create management nodes which control other nodes and require that query and response information flow through the management nodes. The post–query strategies [2] and Konark [9], which are non–agent–based, are also examined.

The Service Location Protocol (SLP) [6] was developed by the IETF as a vendor–independent standard for TCP/IP networks. Clients send service requests to their User Agent (UA),

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

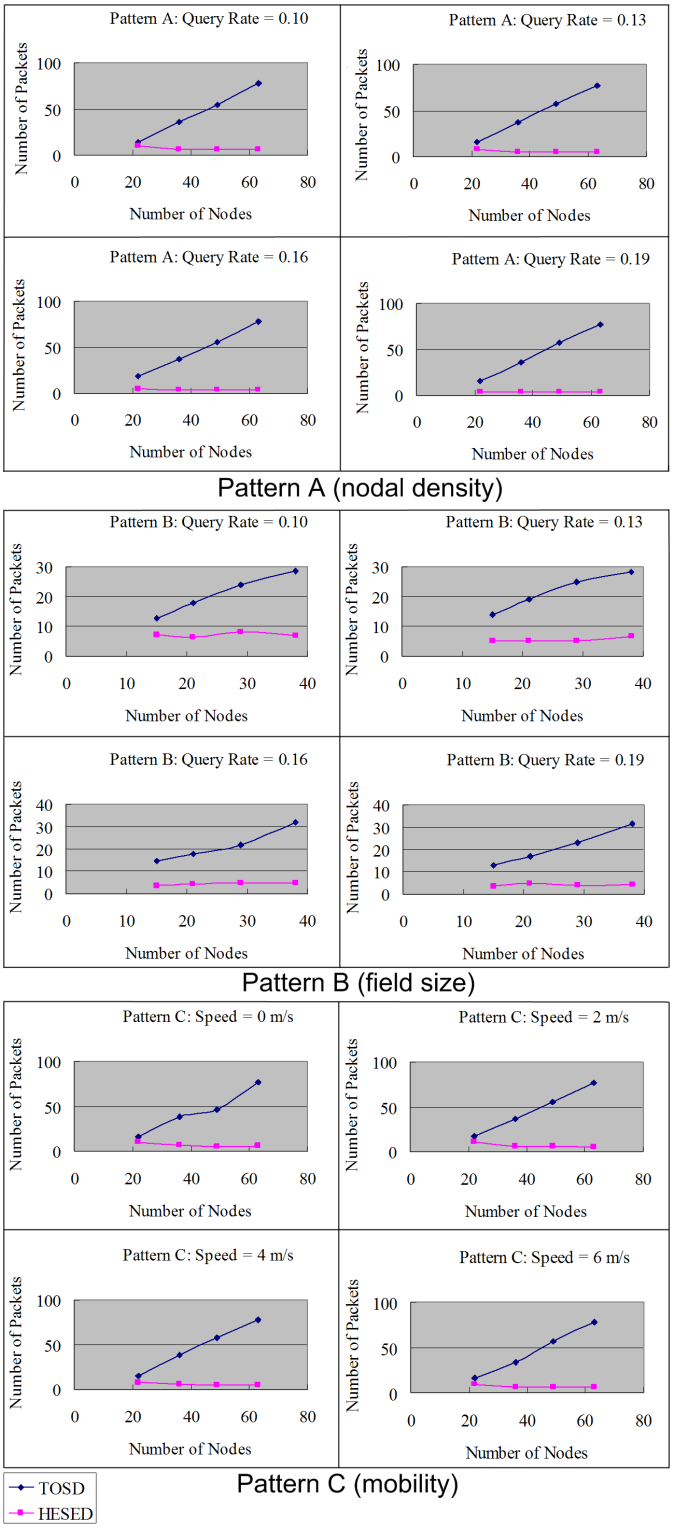

Fig. 4. Average number of packets per query

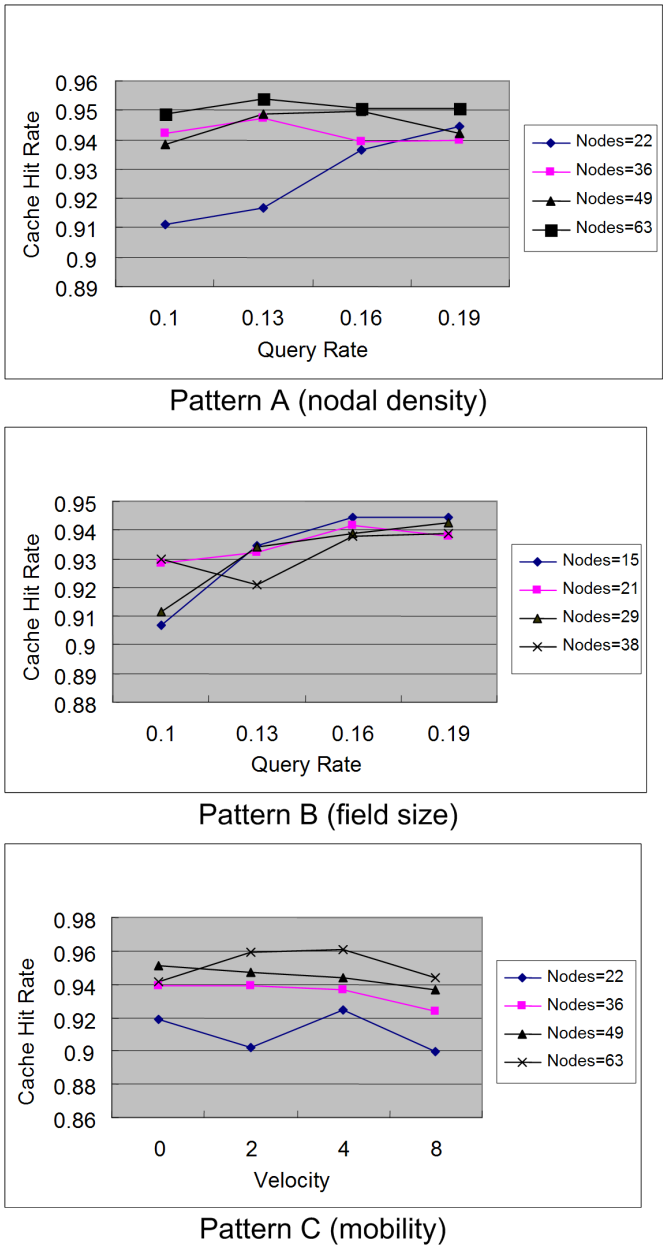

Fig. 5. Comparison of cache hit rates

which pass them on to a Directory Agent (DA). The DA maintains a list of registered services and if it finds a match, sends the information to the UA, which passes it back to the client. Service providers send registration requests to their Service Agent (SA), which passes the information on to the DA. The DA stores the service information and sends an ACK back to the SA, which passes it to the server, indicating that registration has completed. Servers can also send cancellation messages to their SAs when they leave the network.

In Jini [7], based on Java, service providers register with a lookup service. Clients contact the lookup service when they desire a service, which tells them how to contact the service provider directly. Universal Plug and Play (UPnP) [8] is similar to Jini in that servers register with a control point, which acts in

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

similar fashion to the lookup service of Jini.

There are also many agent–based service discovery schemes for MANETs. Kozat and Tassiulas propose a scheme which creates a virtual backbone in the MANET [10]. The nodes with the fewest neighbour changes are selected to form a backbone, and all nodes find a backbone node to act as their Virtual Access Point (VAP). Servers register with their VAP, which then multicasts the information to other VAP nodes, and clients search for services from their VAP.

Other schemes suggest various other infrastructures, such as Service Rings [11], overlays [12], Multi–Layer Clusters [13], a hash indexing self–organized directory agent [14], and a Bluetooth centralized structure [15].

Agent–based protocols are difficult to implement in MANETs because all nodes are unreliable due to their mobility and limited energy. If suitable agents nodes could be found, clients would still need to discover a route to the server. Furthermore, agent nodes would likely have their energy depleted more quickly. For these reasons, HESED does not make use of agent nodes.

The Post–Query protocol [2] is a time–dependent protocol that is executed in rounds modeling request–reply interactions between clients and servers. In each round the server posts service information to some nodes P, and the client requests service information from some nodes Q. Clients will find the service if and only if $P \cap Q$ is not empty in the current round.

Four different strategies exist in this protocol: Post–to–all, Query–to–all, Uniform memory–less, and Incremental post–query. In Post–to–all, the server advertises to all clients, which cache the information. Readvertisements are made before the cached information expires. Clients can then use the cached service information instead of querying the network. In Query–to–all, servers do not advertise at all and instead clients query all nodes. In Uniform memory–less, each server advertises to l nodes, and each client queries l' nodes, where l/l' is fixed. Incremental post–query operates in a sequence. The posting or querying node set begins with a small value and is increased slowly in successive rounds until the client finds the server.

The different strategies of the Post–Query protocol are useful in different situations. For example, Query–to–all is good for high mobility MANETS because although it is expensive, it is also fast. The Post–Query protocol is comprehensive and very general, but it does not consider the routing cost, which is often higher than the service discovery cost in MANETs. HESED makes use of the new strategy of query–all and reply–all, with nodes caching service information. This makes HESED very different from the other post–query strategies. There are no rounds, and the delay and communication costs are minimized. Furthermore, the routing protocol is integrated within HESED.

Konark [9] is a service discovery protocol designed for MANETs in which each server maintains a service tree. From top to bottom, the services become more specific until eventually the service name is found at the leaf node. When a node receives a service request, it searches the service tree from root to leaf.

Konark supports two modes for discovering services. In active pull, a client sends a query to all of the nodes in the MANET to find the service location. In passive push, a server advertises the service information to the entire network periodically. Clients send a discovery message to a fixed multicast group. Each receiver then searches its local service tree to check if it matches the service request. If so, the node sends a service response. Unfortunately, Konark does not consider energy consumption or delay, two critical elements that were kept in mind during the design of HESED.

V. Conclusions

This paper introduced HESED, a High Efficiency SErvice Discovery scheme for MANETs. HESED is fundamentally different from earlier post-query strategies in that it uses query–all and reply–all modes. All nodes cache the service reply information locally but intermediate nodes do not reply to the client. HESED uses Edge Node Forwarding for efficient multicasting, Long Beacon Neighbour Detection to organize the MANET and avoid asymmetric links, and Backward Learning routing for unicast messages.

Simulation results confirm that HESED achieves its design goals of low communication cost and short delay. In comparison to traditional on–demand service discovery schemes, HESED has a lower delay and sends fewer packets, meaning that it also uses less energy. In addition, the caching scheme shows promising results.

References

[1] Yu Yang, "High–efficiency service discovery in wireless mobile ad hoc networks," M.S. thesis, Queen's University, 2005.

[2] M. Barbeau and E. Kranakis, "Modeling and performance analysis of service discovery strategies in ad hoc networks," in *International Conference on Wireless Networks (ICWN)*, 2003.

[3] Y. Zhao, L. Xu, and M. Shi, "On-demand multicast routing protocol with multipoint relay (ODMRP-MPR) in mobile ad-hoc networks," in *International Conference on Communication Technology (ICCT)*, 2003, vol. 2, pp. 1295–1300.

[4] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 3, pp. 257–269, 2003.

[5] G. Lin, G. Noubir, and R. Rajaraman, "Mobility models for ad hoc network simulation," in *IEEE INFOCOM*, 2004.

[6] Internet Engineering Task Force (IETF), "RFC 2608: Service Location Protocol, version 2," http://www.ietf.org/rfc/rfc2608.txt, 1999.

[7] Sun Microsystems, "Jini network technology," http://www.sun.com/software/jini/, 1999.

[8] UPnP Forum, "UPnP documents," http://www.upnp.org/, 1999.

[9] C. Lee, A. Helal, N. Desai, V. Verma, and B. Arslan, "Konark: A system and protocols for device independent, peer-to-peer discovery and delivery of mobile services," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 33, no. 6, pp. 682–696, November 2003.

[10] U. Kozat and L. Tassiulas, "Network layer support for service discovery in mobile ad hoc networks," in *IEEE INFOCOM*, 2003.

[11] M. Klein, B. König-Ries, and P. Obreiter, "Service Rings — a semantic overlay for service discovery in ad hoc networks," in *14th International Workshop on Database and Expert Systems Applications*, 2003.

[12] M. Klein, B. König-Ries, and P. Obreiter, "Lanes – a lightweight overlay for service discovery in mobile ad hoc networks," in *3rd Workshop on Applications and Services in Wireless Networks (ASWN)*, 2003.

[13] M. Klein and B. König-Ries, "Multi-layer clusters in ad-hoc networks an approach to service discovery," in *International Workshop on Peer-to-Peer Computing, co-located with Networking Conference*, 2002.

[14] J. Liu, K. Sohraby, and Q. Zhang, "Resource discovery in mobile ad hoc networks," in *The Electrical Engineering Handbook Series: The handbook of ad hoc wireless networks*, 2003, pp. 431–441.

[15] M. Haase, I. Sedov, S. Preuss, C. Cap, and D. Timmermann, "Time and energy efficient service discovery in Bluetooth," in *57th IEEE Semiannual Vehicular Technology Conference*, 2003, pp. 736–742.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.