An Integrated Approach for Efficient Routing and Service Discovery in Mobile Ad Hoc Networks

Chang-Seok Oh
Graduate School of Information &
Communication, Ajou University
Suwon, Korea
Email: csoh@ajou.ac.kr

Young-Bae Ko
College of Information &
Communication, Ajou University
Suwon, Korea
Email: youngko@ajou.ac.kr

Yong-Sung Roh
Samsung Advanced
Institute of Technology
Kiheung, Korea
Email: yongsung.roh@samsung.com

Abstract—**Service discovery is an important problem in a self-configurable ad hoc network, where each mobile node is required to automatically discover and provide the available network services among themselves. Several protocols have been presented, with a goal of achieving efficient service discovery in mobile ad hoc netwoks. Basically, we have the same objective of developping an efficient protocol, but with an integrated approach. Our work is motivated by the observation that the current protocols mostly assume the existence of some routing protocols underneath, separated or loosely coupled from the service discovery protocol. We argue that, however, such an assumption clearly causes the problem of ”inefficiency” because both protocols are based on a network-wide flooding mechanism, resulting in unnecessarily large number of redundant packet floods. In this paper, we propose a novel approach of integrating these two different but similar discovery protocols for services and routes. Our simulation results show that the proposed integrated scheme, named as HAID (Hybrid Adaptive protocol for Integrated Discovery), can reduce network overhead as well as end-to-end delay in mobile ad hoc networking environments.**

I. Introduction

Recent research in a mobile ad hoc network (MANET) has mainly focused on providing some basic networking support such as medium access control, routing and TCP protocols. There is no doubt that these problems are critical and have to be solved to meet several unique requirements of ad hoc networking environments (i.e., a shared wireless medium, multi-hop routes and their dynamic changes, limited computing and networking resources, and etc.). However, there are also other important research issues that have not yet received much attention and are hence still in their infancy stages. These include among others energy management, network security, and middleware services. Especially, research on middleware for ad hoc networks is important for making the development and deployment of MANET applications easier [8].

Among various middleware services, service discovery plays a critical role in MANETs. Upon joining self-configurable and stand-alone ad hoc networks, mobile nodes should be able to automatically and efficiently discover their available network services (e.g., DNS, Web server, and printer). A variety of protocols have been proposed for service discovery in the context of wired networks [2], but unfortunately these existing protocols may not work well in the real ad hoc environment due to its dynamic nature and the scarce resources. Those service discovery protocols, like Jini, UPnP and SLP, mostly rely on their assumption of the use of centralized directory services, and hence are not suited to ad hoc networking scenarios.

In recent years, some protocols have been presented to support service discovery specifically targeted at MANET environments [2]–[5], [9], [10]. These ad hoc service discovery protocols can be divided into two categories: *centralized directory-based protocols* and *distributed directory-less protocols*. In [4] and [10], the centralized directory-based scheme derived from legacy Internet service discovery protocols has been presented. In these works, some directory agents (or service brokers) are involved as a logical entity in a service discovery process for a communication of clients and service providers. On the other hand, in [2], [3], [5], [9], the distributed directory-less approach with no special directory server has been proposed. Here, every node is required to provide some form of directory services and to discover resources in a peer-to-peer manner. Distributed directory-less protocols are deployed with two different working models: ”Push model” and ”Pull model”. With the push model, service advertisements are actively and periodically flooded by service providers so clients passively learn about the services simply by receiving those advertisements. Whereas, with the pull model, clients are rather active, flooding service queries with the hope that the corresponding servers will eventually reply back.

Although the above proposed algorithms for MANET applications differ in the approach used for searching and providing desired services, they are all middleware oriented solutions and have the common assumption that some routing protocols may exist underneath (separately from the middleware layer) for supporting request/service delivery from clients to service providers or vise versa. This can make them inefficient use in wireless ad hoc networks due to redundant control packet flooding problem, possibly causing extremely high network overhead. It is important to observe that both protocols for service discovery and route discovery are based on a network-wide flooding mechanism with different goals of discovery. Motivated by this interesting observation, we focus on the

[0]This research was supported by Samsung Advanced Institute of Technology, University IT Research Center Project and grant of the 21C Frontier R&D Program from Ministry of Science & Technology in South Korea.

issue of combining these two different (but quite similar in some sense) discovery problems such that service discovery can be supported along with route search for those services.

The rest of the paper is organized as follows. Related work is covered in Section 2. We then describe our proposed scheme in Section 3, followed by simulation results in Section 4. Finally, conclusions and future works are discussed in Section 5.

II. Related Work

There has been recent attention on service discovery in ad hoc networks. First, in [4], service discovery is performed based on the concepts of a virtual backbone with distributed directory servers. Their solution consists of two phases: one for forming and managing the virtual backbone, and the other for performing the actual service discovery function. A subset of nodes (called, *Service Broker Nodes*) needs to be elected to constitute a dominating set, form the virtual backbone, and provide directory service. MARE [10] is also an approach based on the centralized directory service. This protocol exploits a combination of mobile agents and tuple spaces to address resource configuration in ad hoc environments. Note that a tuple space can be thought as a centralized/shared pool for passing configuration data (containing service descriptions, agents and messages) within the system.

A peer-to-peer, policy-driven caching scheme for service discovery is presented in [9], with a goal of better access to services in the vicinity. The use of multicasting is assumed to forward request messages among group members belonging to the same multicast group only - such a locally scoped group is defined based on the concept of *alliance*. Konark [2] is another middleware solution designed for discovery and delivery of services in mobile ad hoc networks. It is also based on a peer-to-peer model and the multicast support by underlying ad hoc routing protocols. Similar to [9], it may come at the cost of excessive messaging overhead as the two unnecessarily redundant floodings are required for each service discovery and delivery: one at the middleware layer for service discovery and the other at the routing layer for request message delivery. [5] also investigates a directory-less resource discovery with QoS awareness, but it only considers the middleware approach.

Overall, the current service discovery protocols for ad hoc networks have been designed as a pure middleware solution. In this paper, we take a different approach, i.e., an integrated approach between service discovery and route discovery. By doing so, network overhead can be significantly reduced. The closest work with ours is the one presented by Koodli and Perkins [3]. They also tried to extend ad hoc routing protocols with service discovery operation so that the wanted information can be obtained at once. Their work is somewhat similar to ours, in the sense that the service discovery operation is not separate from the one for route discovery. Their scheme however, mainly relies on reactive routing protocols such as AODV [7] and therefore it may suffer from increased latency of service acquisition as network size and node mobility becomes larger, due to on-demand route discovery and maintenance properties. To rectify such a problem, we propose a hybrid adaptive routing approach that combines proactive and reactive routing strategies to bring together the advantages of them.

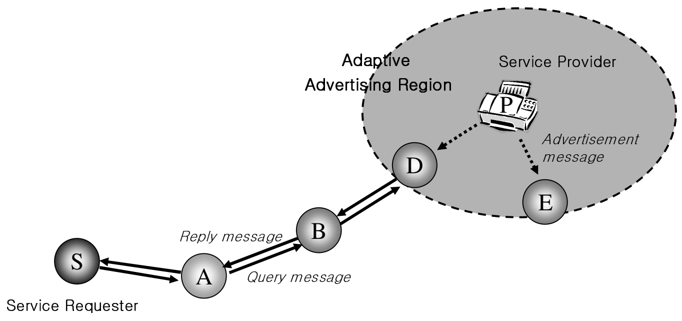


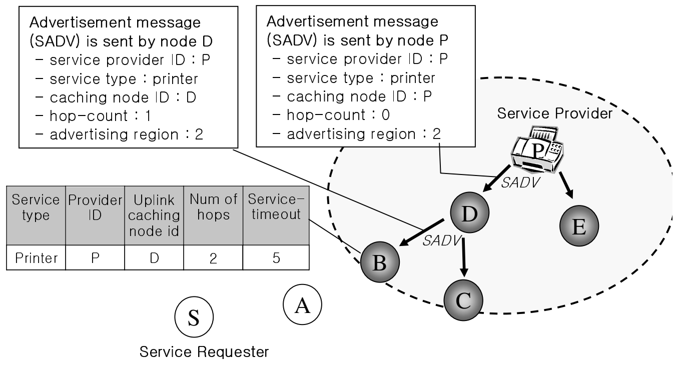
Fig. 1. Overview of the proposed HAID

III. A Hybrid Adaptive Protocol for Integrated Discovery

This section describes our proposed scheme, HAID (Hybrid Adaptive protocol for Integrated Discovery), that efficiently integrates the service and route discovery components. The proposed HAID adapts between a push-based service/routing advertisement and a pull-based service/route query, by dynamically varying the region for advertisements, named *advertising region* around a service provider (See Figure 1). These two main characteristics are further elaborated below.

■ ***Integrated discovery of service and routing information***: As mentioned earlier, most of service discovery schemes proposed for ad hoc networks exploit an application level flooding to obtain or to provide appropriate service information. However, we argue that such a flooding causes too much overhead in resource scared ad hoc environments and hence it needs to be integrated with a routing level flooding. Our argument is based on the simple observation that service advertisement or query messages generated at the upper layer will pass down through a routing layer to be flooded for the whole network. Note that suitable ad hoc routing protocols can be extended to provide support for service query or advertisement along with routes to those services. In our HAID, service information is being piggybacked on the packets for route information. Thus, any corresponding service information such as service type and service provider's ID is embedded within routing control packets.

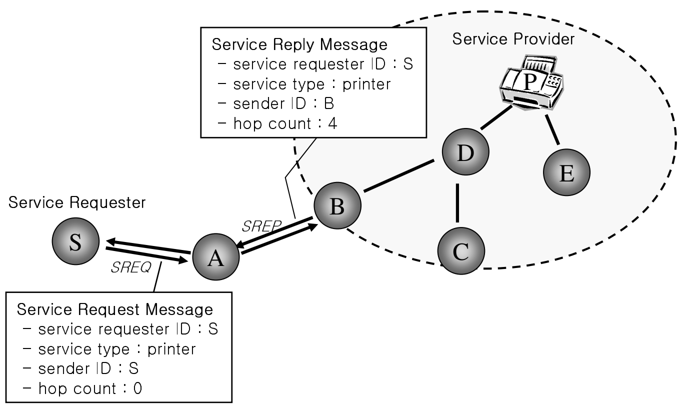
■ ***Hybrid adaptive approach***: The proposed HAID is designed as a hybrid scheme that utilizes a push model for service advertisements and a pull model for service queries. In general, there is a trade-off between push versus pull model. While the push model used by service providers for advertising their service information is more reliable with relatively low latency, it is more expensive in terms of control packet overheads. Whereas, the pull model used by each service requester for querying any desired services can minimize


Advertisement message (SADV) is sent by node D
- service provider ID : P
- service type : printer
- caching node ID : D
- hop-count : 1
- advertising region : 2

Advertisement message (SADV) is sent by node P
- service provider ID : P
- service type : printer
- caching node ID : P
- hop-count : 0
- advertising region : 2

Service type	Provider ID	Uplink caching node id	Num of hops	Service-timeout
Printer	P	D	2	5

Fig. 2. Service Advertisement Phase of HAID


Service Reply Message
- service requester ID : S
- service type : printer
- sender ID : B
- hop count : 4

Service Request Message
- service requester ID : S
- service type : printer
- sender ID : S
- hop count : 0

Fig. 3. Service Discovery Phase of HAID

control overhead and energy consumption, but may suffer from increased latency in service acquisition time. Especially, this delay problem might be intolerable for some MANET applications with some real-time requirement. Given this fundamental tradeoff, HAID utilizes both models to find a good balance between service/route information advertized proactively within some limited vicinity of the service provider (i.e., Advertizing region, depicted as dark shadow in Figure 1) and service/route query messages issued by the service requestor for on demand discovery. The advertizing region initially includes just one-hop neighbors of each service provider, but it can vary adaptively based on how often the service provider has been requested for supporting services. Thus, the size of the region is dynamically determined based on the usage frequency of the particular service. All the nodes within the advertizing region are required to cache service information when they receive any service/routing advertisement messages. Note that any nodes currently caching service/routing information are allowed to answer for an appropriate query messages.

We are now ready to explain more details about the HAID, consisting of three main phases as described below.

A. Service Advertisement Phase

In the HAID, service advertisement phase is initiated by any node hosting one or more services. Any service provider periodically generates an advertising message that describes the service characteristics, and sends it to all its one-hop neighbors. The period of advertising message propagation is determined by the node mobility, link stability, and other network characteristics. The advertisement message, called SADV, contains the following fields:

<service type, service provider ID, caching node ID, hop-count, advertising region>

Here, the *service type* indicates information about the provided service. The *service provider ID* represents the original node ID supporting services, whereas the *caching node ID* is the peer node caching and relaying the advertisement message. While the advertising message's caching node ID is initially equal to the service provider ID, it becomes replaced by an intermediate node ID that relays the message. *Advertising region* can be thought as a time-to-live of the advertisement message, and is initially set to 1. Again, this value is not fixed, possibly varying based on the service's usage frequency in some time interval. See Figure 2, where the value of advertising region is now set to 2. When a node receives an advertising message, it checks its local cache table (we call it "Service Table") to see whether that service entry is already in the table. If yes, it simply updates the corresponding service information in the service table; otherwise, it inserts this new entry into the table. Each entry in the service table contains the following fields:

<service type, service provider ID, uplink caching node ID, num-of-hops, service-time-out>

Above, the *uplink caching node ID* represents a node from which the current node has received a service advertising message, and the *num-of-hops* field represents a hop distance between itself and the service provider. For instance in Figure 2, upon receiving SADV message from node D, node B sets its uplink caching node ID field as D. This field is then considered as a reverse path towards the original service provider P. The num-of-hops field at node B is set to 2 (i.e., the hop-count value of SADV plus 1). Any node receiving SADV is required to compare the value of advertising region (in the receiving SADV) to that of the num-of-hops field (in its local service table), in order to determine whether to forward the SADV or not. If the SADV's advertising region value is greater than the service table's num-of-hops value, it still needs to be forwarded. Last, the *service-time-out* represents the valid time duration of service entry, and its default value is set to 5 seconds in our simulation study.

B. Service Discovery Phase

Service discovery phase can be initialized by any node that desires to use services, generating a service query message (called SREQ) with the following fields:

<service requester ID, service type, sender ID, hop-count>

When node Y receives SREQ from node X, node Y increases the hop-count value of SREQ by one and constructs a reverse path to node X. Node Y then examines whether it has cached the service information desired by the original service requester. If the answer is negative, the receiving SREQ will be re-broadcasted. Only when the desired information is found in its local cache, node Y creates a reply message (called SREP) with the same format of SREQ and sends it back to node X – of course, eventually, to the service requester via the reverse path. The hop-count in the SREP is calculated by adding the hop-count value of SREQ to the num-of-hops value of the service table at node Y. As an example in Figure 3, the hop-count value of SREP generated by node B is set to 4, indicating the first two hops from S to B and the next two hops from B to the service provider.

C. Service Invocation Phase

The last step in HAID is the service invocation phase. In this phase, a service requester asks some particular service(s) to the corresponding service provider by sending an actual invocation message. Service provider counts the service usage frequency each time it receives an invocation message. Remind that this frequency information is used to determine the size of the advertising region for the next service advertisement phase. Service provider keeps track of the service usage frequency for a determined interval, called *Advertisement period.* If the current period's service usage frequency is larger than the result of the following equation of (*HIGH_THRESH* $*$ *the last period's usage frequency*), it decides to increase the next advertising region size by one hop larger. In contrast, if the service usage frequency is smaller than the result of the equation of (*LOW_THRESH* $*$ *the last period's usage frequency*), it narrows down its range by one hop smaller. Note that we use some arbitrary threshold values here (*HIGH_THRESH=1.5, LOW_THRESH=1.0*) and more study on their optimal values and effects are one of our future work items.

IV. Performance Evaluation

In this section we evaluate the performance of our HAID by comparing to the fully push-based advertisement method, the fully pull-based query method, and another integrated AODV-based scheme proposed by Koodli and Perkins [3]. We have modified the CMU wireless extended version of ns-2 [1] for implementing all these schemes.

A. Simulation Model

To evaluate the performance of HAID, we have used the following metrics for measurement: *total number of control packets* transmitted during the service discovery and *average end-to-end delay* on each discovery. Initially nodes are randomly distributed in a confined space of $650 * 650$. For simplification of our simulation, the number of service providers is limited to one. We have varied the number of nodes, the period of query message generation, and a pause time. By varying the number of nodes (i.e. 15, 30, 50, and 75), we try to observe the effect of nodes' density on each metrics. Also, by varying the query message generation period (i.e. 0.05, 0.1, 0.25, 0.5, and 1 time/seconds), we can see the effect of various traffic environments. We have varied pause time as well (from 0 to 1500 sec by increasing 500 sec) in order to reflect node mobility. Each simulation has the duration of 1500 seconds, and run 10 times in total.

B. Simulation Results

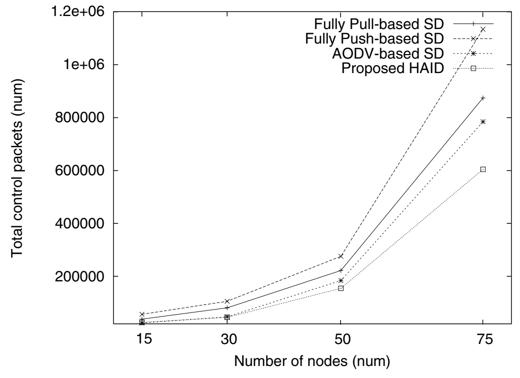
The effect of varying the number of nodes is shown in Figure 4. The other two simulation parameters are fixed (i.e., 0.1/sec of query generation period, and 1500 seconds of pause time). As the network density grows, both the total number of control packets and end-to-end delay are increased. However, our integrated protocol HAID requires much less cost than the other three schemes. The reason is obvious because HAID can reduce the unnecessary redundant packet flooding and prevent collision from occurring. In Figure 4(a), as the number of nodes grows, the control packet overhead rapidly increases. This is because the occurrence of collision is incurred by the broadcast storm problem [6]. In Figure 4(b), the end-to-end delay is longer than the other case when the number of nodes is 15, possibly due to several disjunctions by network division.

Figure 5 displays the effect of varying the period of query message generation while the other two simulation parameters are fixed (i.e., 30 nodes and 1500 seconds of pause time). This graph also shows that our proposed scheme has better performance than the other scheme in terms of total control packet overhead and end-to-end delay. In Figure 5(a), fully pull-based service discovery produces the worse performance than the fully push-based service discovery. This is because the period of query message generation is shorter than the advertisement period. On the other hand, as the event inter-arrival time gets shorter, the end-to-end delay is decreased. It means that service discovery may not happen because the service users cache the required service information previously. However, in fully query-based discovery method and Koodli's integrated discovery method, when the value of the period of query message generation is 1, the end-to-end delay is a bit increased. It is caused by the collision.

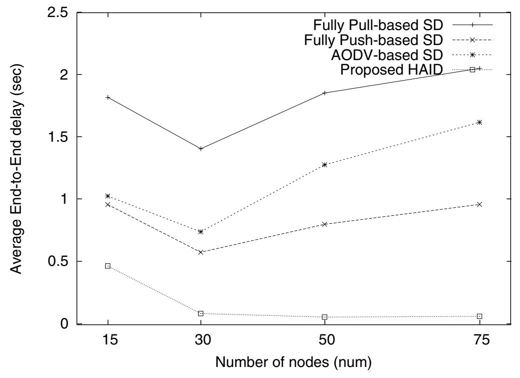
Finally, Figure 6 shows the comparison results by varying the pause time to reflect node's mobility. It is performed with the fixed number of nodes (i.e. 30) and event generation period (i.e. 0.1/sec). Again, our HAID shows better performance than the other service discovery schemes as the pause time has been decreased, in terms of both total control packet overhead and end-to-end delay. In addition, decreasing the pause time causes less degradation of performance in our proposed scheme.

V. Conclusion

We have presented a novel scheme, HAID (Hybrid Adaptive protocol for Integrated Discovery), for the integrated discovery of service and route information. It is a hybrid approach that combines push-based advertising and pull-based querying methods: it can act as a fully push-based integrated discovery protocol for services that are in wide demand, or as a fully pull-based protocol for ones that are in less demand. HAID can automatically find the balancing point between these two extreme approaches by dynamically adjusting the service advertising region size, in which all nodes are required to have knowledge about available services in their local caches. Our simulation results show that the proposed HAID offers much better performance, in terms of network overhead and end-to-end delay.

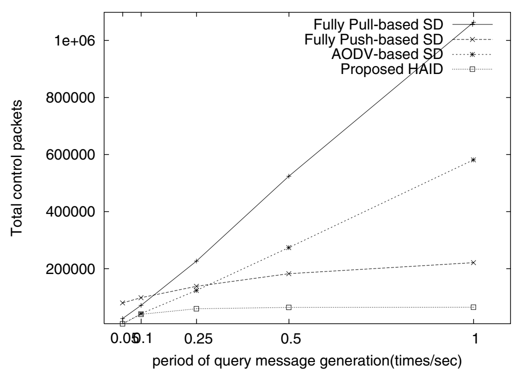


(a) Total control packet overhead

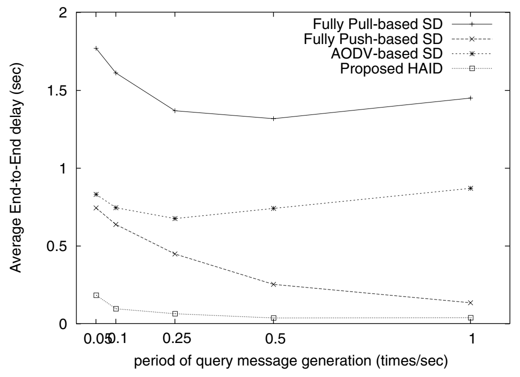


(b) Average end-to-end delay

Fig. 4. Performance of the HAID by varying the nubmer of nodes.

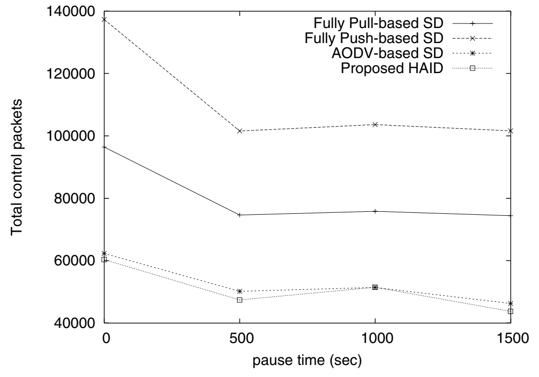


(a) Total control packet overhead

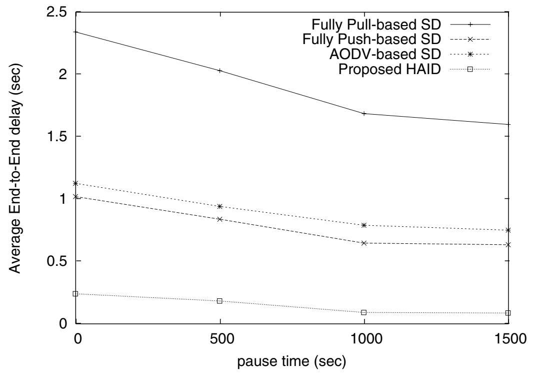


(b) Average end-to-end delay

Fig. 5. Performance of the HAID by varying the period of query message generation



(a) Total control packet overhead



(b) Average end-to-end delay

Fig. 6. Performance of the HAID by varying pause time

In this paper, we have used only one parameter of the service usage frequency to adjust a size of advertising region. We have not investigated the effects of some additional service characteristics, e.g., a bounded loss rate or a delay jitter of services provided, on the change of advertising region and on the performance of the proposed scheme. This will be our immediate future research interest. As a future work, we will consider more intelligent ways of disseminating service advertisement messages. More complicated scenarios such as multiple service providers with different service types will also be considered for a performance evaluation in our future work.

REFERENCES

[1] The monarch project's wireless and mobility extension to ns. [Online]. Available: http://wwww.monarch.cs.rice.edu/cmu-ns.html

[2] S. Helal, "Konark - a service discovery and delivery protocol for ad-hoc networks," in *Proc. of IEEE Conference on Wireless Communication and Networks (WCNC'03)*, Mar. 2003.

[3] R. Koodli and C. E. Perkins, "Service discovery in on-demand ad hoc networks," in *Internet draft, MANET Working Group, draft-koodli-manet-servicediscovery-00.txt*, Sept. 2002.

[4] U. C. Kozat and L. Tassiulas, "Network layer support for service discovery in mobile ad hoc networks," in *The 22th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '03)*, Apr. 2003.

[5] J. Liu, "Resource discovery in mobile ad hoc networks," in *The Hand book of Ad Hoc Wireless Networks*. CRC Press, New York, 2003, ch. 26.

[6] S.-Y. Ni, Y.-C, Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc networks," in *Proc. Of the 5th ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '99)*, Aug. 1999.

[7] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Internet draft, MANET Working Group, draft-ietfmanet-aodv-05.txt*, Mar. 2003.

[8] T. Plagemann, "Towards middleware services for mobile ad-hoc network applications," in *Proc. of the IEEE Workshop on Future Trends of Distributed Computing Systems (FTDC'03)*, 2003.

[9] O. Ratsimor, "Allia: Alliance-based service discovery for ad-hoc environments," in *Proc. of the ACM Mobile Commerce Workshop*, 2002.

[10] M. Storey, G. Blair, and A. Friday, "Mare: Resource discovery and configuration in ad hoc networks," in *Mobile Networks and Applications Journal, No. 7*, 2002, pp. 377–387.