# Design of Service Discovery Scheme based on Connection Degree in Mobile Ad-hoc Network Environments

**Gu Su Kim†**

School of Information and Communication Eng., Dongyang University, South Korea

**Summary**
The proliferation of mobile devices and the pervasiveness of wireless technology have provided a major impetus to replicate the network-based service discovery technologies in wireless and mobile networks. However, existing service discovery protocols mainly support one-hop ad-hoc network. In this paper, we propose the service discovery protocol based on the connection degree information of the nodes in the mobile ad-hoc networks. Our proposed protocol is based on the concept of peer-to-peer caching of service advertisements. Each node manages the basic information (connection degree, service degree, and power degree) of adjacent nodes in its own neighbor connection degree table. For the service advertisement and discovery request messages, the intermediate node delivers these messages to adjacent nodes with connection degree larger than or equal to its own connection degree. Therefore, our proposed protocol allows nodes to discover the service in peer-to-peer fashion.
***Key words:***
*service discovery, ad-hoc* network, p2p fashion, connection degree .

## 1. Introduction

A mobile ad-hoc network (MANET) is a system of wireless mobile nodes that dynamically self-organize in arbitrary temporary network topologies. The mobile hosts of this network can communicate with each other without pre-existing communication infrastructure. These networks are characterized by their lack of required infrastructure and ease of network formation; each participating device has mobility and the networks are formed temporarily. Communications between two network nodes that are not in direct radio range of one another takes place in a multi-hop fashion, with other nodes acting as routers. Ad-hoc networks can be used in military and rescue operation, as well as in meetings where people want to share information quickly [1].

Service is the computing resource used by users, user programs, or other services. Service discovery protocols address the problem of discovering service providers by specifying desired properties of the services. The main task is to find the IP-address of the service provider, which can then be contacted by the client for the service session. Service discovery and management is a main challenge for MANET. MANET applications often times need to utilize resources or services that are present on other mobile nodes in its neighborhood. This leads to greater utilization of resources that are available in our service-rich vicinity. Hence, it is important for applications to be able to seamlessly discover other remote services/resources present on nearby mobile devices and to carry out transactions with other services [2].

In the existing wired network, centralized directory-based system is used and directory information of resources such as meta data and addresses of resource providers is registered at directory servers. To search the directory information of a requested resource, a client contacts its corresponding directory server. For the Internet, this approach has shown to be very efficient for resource discovery [3]. However, in MANET, since there is no fixed topology, maintaining a hierarchical structure of directory or measurement servers is not an easy task. Moreover, statically configured domains do not reflect the dynamic relations of mobile nodes.

In this paper, we propose the service discovery scheme that delivers the service advertisement message of servers and the service discovery request message of clients on the basis of the information about connection degree and service information of its neighbor nodes without the central directory in MANET.

The rest of the paper is organized as follows. In Section 2, we describe related work on the service discovery for MANET. Section 3 describes the system architecture, data structure, and service discovery protocol of our proposed scheme. Section 4 shows the performance evaluation of our scheme. Finally, Section 5 concludes with a summary.

## 2. Related Work

Existing resource discovery protocols can be categorized according to whether a global index of resource is maintained centrally, a distributed index is maintained, or resources are discovered as they are needed. They can also

Manuscript received May 5, 2008
Manuscript revised May 20, 2008

be categorized according to whether they are based on one-hop network or multi-hop network.
The main purpose of a central directory is to support efficient management of a large number of resources as in Jini [4~6]. Sun Microsystems' Jini is based on Java technology. One special feature of Jini is the mobile Java codes, which may be moved among clients, services, and directories. The advantage of Jini is its platform independency, but the disadvantage is that all the clients, services, and directories depend on Java runtime environments directly or indirectly.

Universal Plug and Play (UPnP) [7~9] is from the UPnP Forum whose major player is the Microsoft Corporation. UPnP targets unmanaged networking environments such as home environments. UPnP supports the device oriented service discovery protocol. All the service information and communication is in the XML format, which is a platform and programming language independent language and greatly increases interoperability between devices.

The Salutation Consortium has rolled out the Salutation protocol [10~12] which is an open source royalty-free protocol. One advantage of the protocol is that it implements two interfaces. One interface is for applications. The other interface is designed to be independent of the transport layer, so that it is very flexible to use various underlying transport protocols. Furthermore, a mapping of Salutation over the Bluetooth Service Discovery Protocol has been specified.

Posted by IETF as a standard track protocol, Service Location Protocol (SLP) Version 2 [13, 14] is for enterprise environments. As the protocol name states, SLP only defines a way to locate a service and leaves the interaction between clients and services after service discovery open. URLs are used for service locations.

## 3. The Service Discovery Protocol based on Connection Degree

In this Section, we describe our proposed service discovery protocol based on node's connection degree. Our proposed discovery protocol uses the concepts of peer-to-peer caching and delivery of service advertisements and service discovery requests on the basis of the connection degree of the node.

In this paper, we assume that MANET consists of nodes supporting the same network interface with IP-level connection using any routing protocol among the protocols for MANET.

### 3.1 System Architecture

Each mobile node manages three basic information: $C_d$(Connection Degree), $S_d$(Service Degree), and $P_d$(Power Degree). $C_d$ is the connection degree, which means the number of the node's connection with adjacent nodes. $S_d$ is the service degree, which means the number of services directly supported by the node. $P_d$ means the available service time of the node. Each node exchanges the basic information with its neighbors when the node is connected to or disconnected from the node.

We define that a server is the node to provide other nodes with its services, and a client is the node that requests the service discovery and uses the service of the server. All mobile nodes have 6 components related with service discovery. Figure 1 shows the service discovery component architecture of the mobile node. The 6 components are as follows:

- NCDT(Neighbor Connection Degree Table) : The table that has the basic information about $C_d$, $S_d$, $P_d$ of its neighbors.
- LSD(Local Service Directory): The directory that has the service information provided by the node itself.
- NSC(Neighbor Service Cache): The cache that has the information about services provided by its neighbors.
- SAC(Service Advertisement Cache): The cache that has the service information included in service advertisement messages.
- SD(Service Discoverer): The components that processes service advertisement message, service discovery request message, and service discovery response message.
- MF(Message Forwarder): The component that delivers the service advertisement message and service discovery request message.

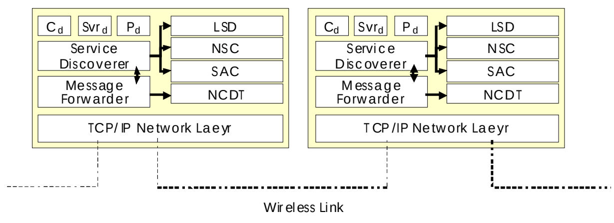


Fig. 1 Service discovery component architecture of mobile node.

### 3.1 Message Format

We define three kinds of messages for our service discovery protocol: *SAM*(Service Advertisement Message), *SQM*(Service discovery reQuest Message), and

*SRM*(Service discovery Response Message). The *SAM* message has the following format:

<*Packet_Type*, *Server_Address*, *Cd*, *AST*, *TTL*, *Service#*, *Service_Category1*, *Service_Name1*, ... , *Service_Category n*, *Service_Name n*>

*Packet_Type* denotes the type of the message, *Server_Address* is the IP address of the server, *Cd* is the connection degree value of the node, *AST* is the available service time, *TTL* is the value of time to live, *Service#* is the number of services included in *SAM* message, *Service_Category* is the category name of the service, and *Service_Name* is the user friendly name of the service. The *SQM* message has the following format:

< *Packet_Type*, *Service_Category*, *Service_Name*, *Source_Address*, *TTL*>

In the *SQM* message, *Source_Address* is the IP address of the node that requests the service discovery by *SQM* message. Other fields have the same meaning as in *SAM* message. The *SRM* message has the following format:

< *Packet_Type*, *Service_Category*, *Service_Name*, *Server_Address*, *AST*, *TTL*>

The roles of the fields in *SRM* message are the same as those described above.

## 3.3 Service Discovery Protocol

In this Section, we describe the details of our proposed service discovery protocol. Our service discovery protocol consists of three steps: service advertisement step, service discovery request step, and service discovery response step.

**(1) Service Advertisement**

Servers in MANET first register their own services into their *LSD* whenever new services are registered at the server. When the server advertises their services, the server gathers the service entries in *LSD* and generates *SAM* message with the packet type as SA(Service Advertisement). The server broadcasts the *SAM* message to its neighbors. Figure 2 shows the algorithm of the service advertisement at the server.

All neighbors of the server receive the *SAM* message. If the packet type of *SAM* message is SA, it means that the node receiving the message is the neighbor node of the server. If the node receiving the message is the neighbor node, the node searches the services included in *SAM* message in its own *NSC*. If the service entry is not found in *NSC*, the node registers the service information into its own *NSC*. The node changes the packet type to AD(Advertisement Delivery) and mediates the TTL value as k, which is the threshold value of the delivery. If the packet type of the *SAM* message is AD, it means that the node is not the neighbor node of the server. So, the node searches the services included in its own *SAC*. If the service entry is not found in *SAC*, the node registers the service information into its own *SAC*. The node decreases the TTL value in *SAM* message and, if the value is 0, the service advertisement delivery algorithm exits. Otherwise, the node delivers the *SAM* message to its neighbor nodes.

The node changes the $C_d$ value of *SAM* message to its own $C_d$ value and tries to deliver the *SAM* message to the next intermediate nodes. The node selects nodes with $C_d$ value larger than or equals to its own $C_d$ value in its own *NCDT* as the next delivery node. Figure 3 shows the algorithm of *SAM* delivery.

```
ServiceAdvertise()
// input: none
// output: none
{
    SAM msg;            // Service Advertisement Message

    msg.packe_type = SA;          // SA : Service Adervtisement
    msg.server_address = address;// assigns the server address
    msg.cd = Cd;     // assigns connection degree
     msg.ast = Pd;    // assigns the power degree to available service time
     msg.ttl = 1;       // one-hop broadcasting
    // adds service information of the node to SAM
     for (i=0; i < LSD.size; i++)
         MakeServiceAdvertisementMessage(msg, LSD[i]);

     OnehopBroadcast(msg);
}
```

Fig. 2 Service advertisement algorithm at the server.

(2) Service Discovery Request

When the client requests a service, the client sequentially searches the services in *LSD*, *NSC*, and *SAC*. If the client finds out the service, the client requests the service to the server specified in the entry. If the service is not found, the client generates the *SQM* message and sends it to its neighbors with $C_d$ value larger than or equal to its own $C_d$ value in its own *NCDT*. Figure 4 shows the algorithm of the service discovery request at a client.

The node receiving *SQM* message searches the service specified in *SQM* message at its own *LSD*, *NSC*, and *SAC*. If the service is found, the node generates *SRM* message with the information in the service entry found, and responses the *SRM* message to the client, elsewhere the node selects next intermediate nodes in its own *NCDT* and delivers the *SRM* message to the next intermediate nodes selected. Figure 5 shows the *SQM* delivery algorithm.

```
SAM_Delivery(SAM msg) {
// input: SAM
// output: none
    If (msg.packettype == SA) {
      // registers services included in SAM message into its own NSC
      RegistService(NSC, msg);
       // changes the packet type to AD(Advertisement Delivery)
       msg.packettype == AD;
       msg.TTL = k;
    }
    else {
        // registers services included in SAM message into its own SAC
         RegistService(SAC, msg);
         msg.TTL--;
         if(msg.TTL ==0) return;
    }

   // delivers the SAM message if its own Cd is larger than or
   // equal to the Cd in SAM message
   If (Cd >= msg.Cd) {
        msg.Cd = Cd;
        for (i=0; i<NCDT.size; i++)
            if (NCDT[i].Cd >= Cd)
                ForwardMessage(NCDT[i].NID, msg);
    }
}
```

Fig. 3 The algorithm of the service advertisement delivery.

(3) Service Discovery Response

In the *SQM* delivery stage, if the service specified in *SQM* message is found in *LSD*, *NSC*, or SAC, the node responses *SRM* message to the client. The client receives *SRM* messages during some time interval, and selects the server with the largest available service time among the *SRM*s received, and requests the service to the server.

```
ServiceDiscoveryRequest(servicecategory, servicename)
// input : service category string , service name string
// output: the reference of service found
{
    Service s;
    s = SearchLSD(servicecategory, servicename);
    if (s!=null)  return s;
    s = SearchNSC(servicecategory, servicename);
    if (s!=null)  return s;
    s = SearchSAC(servicecategory, servicename);
    if (s!=null)  return s;

    SQM msg;
   // assigns the packet type as SQ(Service discovery request)
    msg.packettype = SQ;
    msg.servicecategory = servicecategory;
    msg.servicename = servicename;
    msg.TTL = k;
      for (i = 0; i<NCDT.size; i++) {
         if (NCDT[i].Cd >= Cd) // selects the neighbor to receive the SQM
            ForwardMessage(NCDT[i].NID, msg);
    }
}
```

Fig. 4 The algorithm of service discovery request at a client.

```
SQM_Delivery (SQM msg)
// input : SQM message
// output: the reference of service found
{
    Service s;
    s = SearchLSD(msg.servicecategory, msg.servicename);
    if(s!=null)  return s;
    s = SearchNSC(msg.servicecategory, servicename);
    if(s!=null)  return s;
    s = SearchSAC(msg.servicecategory, servicename);
    if(s!=null)  return s;

    if(msg.ttl == 0) return;
    msg.Cd = Cd;
    for(i = 0; i<NCDT.size; i++) {
        if(NCDT[i].Cd >= Cd)
           ForwardMessage(NCDT[i].Nid, msg);
    }
}
```

Fig. 5 The delivery algorithm of SQM message.

## 3. Performance Evaluation

In this Section, we evaluate the performance of our proposed scheme through simulation. Our proposed scheme is based on the peer-to-peer caching of the *SAM* message so that the basis of *SAM* and *SQM* delivery is the most important factor affecting the performance. In our proposed scheme, there are two bases: the connection degree and the service degree of the node. We simulated our proposed scheme according to the number of nodes. In our simulation, 30% nodes act as the server providing services and each node moves with the speed following exponential distribution with mean 5seconds.

Figure 6 shows the mean delivery count of *SAM* message according to the number of nodes at a service advertisement stage. The mean delivery count of *SAM* delivery based on $C_d$ is lower than that based on $S_d$. This means that the delivery based on $C_d$ produces the amount of messages less than that based of $S_d$. However, the numerical difference between the two is very small.

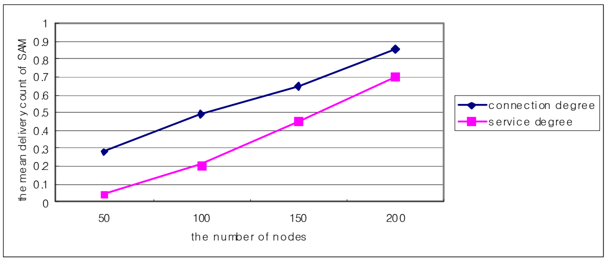


Fig. 6 The mean delivery count of SAM message according to the number of nodes..

Figure 7 shows the mean delivery count of *SQM* message according to the number of nodes per service discovery request message. The mean delivery count of *SQM* message based on $S_d$ is lower than that based on $C_d$. However, considering the message amount of *SAM* messages sin Figure 6, the overall performance of the delivery based on $C_d$ is better than that based on $S_d$.

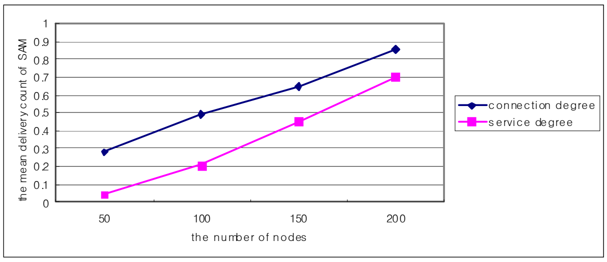


Fig. 7 The mean delivery count of SQM according to the number of nodes.

Table 1 shows the service discovery ratio at each cache(*LSD*, *NSC*, and *SAC*) at the *SQM* delivery stage. In the case that the basis of the delivery is $C_d$, services are discovered at *LSD* with 1% probability, *NSC* with 35%, and *SAC* with 63%. However, in the case that the basis of delivery is $S_d$, the ratio of *NSC* is about 1%, and the ratio of *NSC* is dramatically decreased as the number of nodes increases. On the other hand, the ratio of *SAC* is rapidly increases as the number of nodes increases. This results from that the delivery based on $S_d$ leads to incur more frequent deliveries than that based on $C_d$ regardless of the number of neighbor nodes.

Table 1: The service discovery ratio at each cache

| number of nodes | LSD | | NSC | | SAC | |
|---|---|---|---|---|---|---|
| | $C_d$ | $S_d$ | $C_d$ | $S_d$ | $C_d$ | $S_d$ |
| 50 | 1% | 1% | 39% | 30% | 60% | 69% |
| 100 | 4% | 2% | 35% | 23% | 61% | 75% |
| 150 | 1% | 0% | 34% | 19% | 65% | 81% |
| 200 | 0% | 0% | 35% | 8% | 64% | 92% |

## 5. Conclusion

In this paper, we proposed a service discovery protocol supporting multi-hop networks based on the connection degree. The server broadcasts service advertisement to adjacent nodes, and the intermediate nodes deliver the service advertisement message to their neighbor nodes with $C_d$ value larger than or equal to its own $C_d$ value in *NCDT*. In the delivery stage of the service advertisement message, each node registers the service information in service advertisement message into its own *NSC* or *SAC*. When a client requests service discovery, each node sequentially searches its own *LSD*, *NSC*, and *SAC*. If the node finds the service specified in the request message in the directory or cache, the node responses the service information to the client. Otherwise, the node delivers the service discovery request message to the neighbor nodes with $C_d$ value larger than or equals to its own $C_d$ value in its own *NCDT*. Therefore, our proposed scheme allows each node to discover the service based on the basic information of neighbors.

## References

[1] C. E. Perkins, Ad Hoc Networking, Addsion-Wesley, 2001.
[2] D. Chakraborty and A. Joshi, "GSD: A Novel Group-based Service Discovery Protocol for MANETS," Proc. 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN 2002). Stockholm. Sweden. Sep. 2002.
[3] Decentralized Resource Discovery in Large Peer-based Networks, http://cubicmetercrystal.com/alpine/discovery.html.
[4] Sun Microsystems, "JINI Architectural Overview: Technical White Paper," http://www.sun.com/software/jini/whitepapers/architecture.html.
[5] M. Uhlmann and S. Rupp, "Service Discovery in Bluetooth and JINI," INFOTECH Seminar Advanced Communication Services(ACS), 2004.
[6] Sun Microsystems, Jini Architecture Specification, Jun. 2003.
[7] Microsoft Corporation, "Understanding UPnP: A White Paper," Jun. 2000, http://www.upnp.org/download/UPNP_UnderstandingUPNP.doc.
[8] Microsoft Corporation, UPnP Device Architecture v1.0.1 Draft, 2003.
[9] Microsoft Corporation, Understanding Universal Plug and Play White Paper, Jun. 2000.
[10] Salutation Consortium, Salutation Architecture Specification Version 2.0c Part 1, Jun. 1, 1999, http://www.Salutation.org.
[11] C. Lee and S. Helal, "Protocols for Service Discovery in Dynamic and Mobile Networks," International Journal of Computer Research, Vol. 11, No .1, 2002.
[12] White paper, "Salutation Architecture: overview," http://www.salutation.org/whitepaper/originalwp.pdf, 1998.
[13] E. Guttman, "Service Location Protocol: Automatic Discovery of IP Network Services," IEEE Internet Computing, 1999.
[14] E. Guttman, C. Perkins, J. Veizades, and M. Day, "Service Location Protocol, Version 2(RFC 2608)," http://www.faqs.org/rfcs/rfc2608.html, 1999.

**Gu Su Kim** received his B.S., M.S. and Ph.D. degrees from the School of Electrical and Computer Engineering at Sungkyunkwan University in 1994, 1996, and 2006 respectively. He is a professor at Dongyang University in Korea. He is currently studying embedded systems..