A Socio-ecological Model for Advanced Service Discovery in Machine-to-Machine Communication Networks

LU LIU, University of Derby
NICK ANTONOPOULOS, University of Derby
MINGHUI ZHENG, Hubei University for Nationalities
YONGZHAO ZHAN, Jiangsu University
ZHIJUN DING, Tongji University

The new development of embedded systems has the potential to revolutionize our live and will have a significant impact on future Internet of Thing (IoT) systems if required services can be automatically discovered and accessed at runtime in Machine-to-Machine (M2M) communication networks. It is a crucial task for devices to perform timely service discovery in a dynamic environment of IoTs. In this paper, we propose a Socio-Ecological Service Discovery (SESD) model for advanced service discovery in M2M communication networks. In the SESD network, each device can perform advanced service search to dynamically resolve complex enquires, and autonomously support and co-operate with each other to quickly discover and self-configure any services available in M2M communication networks to deliver a real-time capability. The proposed model has been systematically evaluated and simulated in a dynamic M2M environment. The experiment results show that SESD can self-adapt and self-organize themselves in real time to generate higher flexibility and adaptability and achieve a better performance than the existing methods in terms of the number of discovered service and a better efficiency in terms of the number of discovered services per message.

Categories and Subject Descriptors: **C.2.2 [Computer-Communication Networks]**: Network Protocols

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Machine-to-Machine Communication Networks, Social-ecological model, Service Discovery

1. INTRODUCTION

The Internet of Things integrates a vast number of embedded resources, including sensor networks, medical devices, swarms of robots and vehicles, which offers a great potential of achieving tasks that are far beyond the capabilities of existing systems. The dynamic capability could be delivered by integrating the services via machine-to-machine (M2M) communications for a specific mission objectives.

Sensor networks are the foundation for enabling the Internet of Thing. In a scenario of disaster rescue and relief, sensor networks have the potential to revolutionize the capture, processing and communication of critical data for use of disaster rescue and relief [Lorincz et al. 2004]. The functions of sensors need to be integrated to provide a joint service to meet different search and rescue requirements. For the provision of a dependable rescue capability in a dynamic and unpredictable disaster area, the networked sensors should have ability to autonomously support and co-operate with each other to quickly configure any services available on M2M communication networks to deliver a real-time capability, self-adaptability to modify their behaviors to deliver a sustainable capability according to environmental changes, and ability to share information, generate access and protect information throughout the network. Different types of services across different platforms [Liu et al. 2009] need to be integrated to provide a joint rescue capability. For example, sensors for detecting survivors could employ any of laser, visible specter, heat (infrared), radar or sonar sensors and these heterogeneous sensor services

This work was partially supported by the National Natural Science Foundation of China under Grants No. 61502209 and 91218301, and National Natural the Natural Science Foundation of Jiangsu Province under Grant No.BK20130528.

Author's addresses: L. Liu (corresponding author) and N. Antonopoulos are with the Department of Computing and Mathematics, University of Derby, Derby, UK; M. Zheng is with School of Information Engineering, Hubei University for Nationalities, China; Y. Zhan is School of Computer Science and Telecommunication Engineering, Jiangsu University, China; Z. Ding is with the Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, China.

.

can be deployed on rescue helicopters, UAVs (unmanned aerial vehicles) and search and rescue personnel. The sensors should provide interoperable service interfaces to the integrating applications to deliver dependable capabilities in highly critical situations.

In order to satisfy different search and rescue requirements, different services need to be integrated to deliver a joint search and rescue mission. A dynamic network of services is modelled with a series of sensors, which provides the ongoing data and facilitate timely update [Wu et al. 2015]. In the effort of disaster rescue and relief, a surveillance user can submit real-time requests to the system for information of Points of Interest (POIs) in a specified region. A sequence of services (such as "Get map information", "Get radar reading" and "Display targets on map") can be operated in a workflow in order to provide a regional surveillance service.

However, these capabilities do not come easily. A significant challenge for M2M communications to be successful in reality is to make these devices as autonomous as possible [Villaverde et al. 2014] and minimize the human intervention [Liotta 2013]. Novel methods are needed for just-in-time dynamic discovery and autonomous management of services in M2M communication networks. For provision of capability, the networked nodes should have the ability to: 1) autonomously support and co-operate with each other in a peer-to-peer (P2P) manner to quickly discover and self-configure any services available to deliver a real-time capability; 2) perform advanced service search to resolve complex enquires; 3) self-adapt themselves to environmental changes; 4) self-organize themselves in real time to generate higher flexibility and adaptability.

In response to these needs, in this paper, we are going to propose a novel Socio-Ecological Service Discovery (SESD) model with the supporting algorithms, including the routing algorithm for simple queries, the routing algorithm for multi-function queries and adaptive service discovery algorithm. In SESD, each node in M2M communication networks can self-organize themselves and autonomously support and co-operate with each other by building up a socio-ecological system with associated peers for effective service sharing and discovery. In contrast to the most of distributed search algorithms, the proposed algorithm is not only able to handle simple enquires, but also achieve advanced service discovery with multi-functions and query packs.

The rest of paper is organized as follows: Section 2 outlined the related work. The model design of SESD is introduced in Section 3. The evaluation methodology and simulation results of SESD are shown in Section 4 and Section 5. The work is concluded in Section 6.

2. RELATED WORK

2.1 Theoretical foundation

In 1960s, the social psychologist Milgram conducted a famous small-world experiment [Milgram 1967]. He sent 60 letters to various recruits in Omaha, Nebraska who were asked to forward the letters to a stockbroker living at Massachusetts. The participants were only allowed to pass the letter by hand to friends who they thought might be able to reach the destination, no matter if directly or via a "friend of a friend". The most famous result of his experiment is that the average length of the resulting acquaintance chain is about six, which leads to the well-known phase "six degree of separation".

Working much more recently, Watts and Strogatz [Watts and Strogatz 1998] presented a mathematical model to analyze the small world phenomenon. They explored small world networks by rewiring regular networks to introduce increasing amounts of disorder with probability p. From their analysis, the small world social networks can be highly clustered, like regular lattices, and also have small characteristic path lengths, like random graphs.

In social networks, individual people or aggregate units (such as department, organization or family) are usually analyzed as actors [Haythornthwaite 1996]. A social network is a set of actors

and the relations that hold them together. These actors exchange resources (e.g. information, goods or social services), which then connect them in a social network.

Watts [Watts et al. 2002] presented that individuals in social networks are endowed not only with network ties, but also with identities: sets of characteristics which they attribute to themselves and others by virtue of their associations with social groups. Query messages can be directed efficiently with the combined knowledge of network ties and social identities. Watts also showed that individuals in social networks cluster the world hierarchically into a series of layers, where the highest layer accounts for the entire world and each successively deeper layer represents a cognitive division into a greater number of increasingly specific groups.

Social networks are not only investigated as a whole by using sociocentric approaches [Haythornthwaite 1996; Watts, Dodds and Newman 2002], but also explored with egocentric approaches by analyzing each person and his/her connections to other people [McCarty 2002]. A personal network is a special kind of social network that is centered around a person [McCarty 2002].

Newcomb [Newcomb 1975] indicated that one of the primary differences between the behavior of the newborn infant and that of an adult is that random and diffuse behaviors gradually become more highly organized. Adults' behaviors are mobilized in a selective manner toward specific goals. People tend to manipulate circumstances, so that they benefit in socializing with the people they choose [Newcomb 1975]. The personal network is usually used for personal own benefit. For example, when a person is assigned to a project with a topic, the first thing he/she usually would do is to contact people who know the topic or who can provide background information or give advice on how to solve his/her problems [Waloszek 2002].

The previous study in [Kautz et al. 1997] shows that one of the most effective channels for disseminating of information and expertise as well as finding information within an organization is his/her social network of collaborators, colleagues and friends. Searching for a piece of information in social networks is most likely a matter of searching for an expert on the topic together with a chain of personal referrals from the searcher to the expert [Kautz, Selman and Shah 1997].

2.2 Service Discovery in M2M Communication Networks

The existing service discovery protocols for M2M communication networks can be classified into two broad categories: centralized protocols and distributed protocols. Domain Name System Service Discovery (DNS-SD) [Cheshire and Krochmal 2013] is a service discovery protocol based on standard DNS programming interfaces, the clients (e.g. sensors, applications) could query the DNS-SD service for a record which contains the path to the resource [Jara et al. 2012]. CoAP RD is service discovery protocol based on CoAP proposed by IETF CoRE working group [Shelby et al. 2013]. CoAP is a centralized protocol which allow clients to search a centralized resource directory (RD) for desired services and resources. Similar to CoAP RD, the study [Kwak et al. 2008] proposed a service discovery protocol for mobile sensors which based on a centralized gateway. The centralized gateway can forward service discovery request message to closest directory server for mobile sensors. Both of DNS-SD and CoAP RD can be based on multicast DNS (mDNS). The low complexity of mDNS makes it a suitable candidate for low power networks[Klauck and Kirsche 2012; Ostmark et al. 2012].

In contrast to centralized protocols, distributed protocols, devices discover services by interacting with each other without going through a centralized directory. iMesh is a decentralized service discovery protocol. In iMesh, service providers publish their service by broadcast their location. The node receiving information from multiple service providers only forwards information from the closest provider. Clients conduct a cross-lookup process within their home mesh cells to discover the available services [Li et al. 2009]. NeuroGrid [Joseph May 2002] is a well-known P2P resource discovery method which enables nodes to learn the results of

previous searches to make future searches more focused. NeuroGrid utilizes the historic record of previous searches to help peer nodes make routing decisions. In NeuroGrid, peer nodes support distributed searches through semantic routing by maintaining routing tables at each node. The study [Sethom and Afifi 2005] proposed distributed architecture for service location in WSNs based on Tapestry, a well know P2P search protocol. The architecture inherits load balancing and fault tolerance from the DHT-based structured P2P protocol. The study [Ragusa et al. 2005] introduces an adaptive clustering approach based on on code mobility. This method is self-configuring and autonomous method which does not require any user intervention. The study [Marin-Perianu et al. 2006] presents an energy-efficient cluster-based service discovery method in wireless sensor networks. The protocol forms an underlying clustering structure where cluster heads store a distributed directory for service discovery. During the search process, queries are exchanged among cluster heads to discover the requested services.

Search techniques described above provide useful solutions to the problem of resource discovery in M2M communication networks. However, existing methods either require a high overhead for network maintenance or generate a potentially huge amount of traffic into the network. None of these have explored the possibility of self-organizing peer nodes in a cooperative manner by mimicking different human behaviors in social networks and shown how these human behaviors can bring about benefit in improving the performance of resource discovery in M2M communication networks. Moreover, none of them can handle complex service discovery by resolving query packs. Bearing this in mind, a socio-ecological is presented in the next section, considering whether the small world phenomenon and human strategies in social interactions are useful for resource discovery in M2M communication networks and whether they can avoid the problems of previous methods as discussed in this section.

3. SOCIAL-ECOLOGICAL SERVICE DISCOVERY

In the traditional implementations of Service-Oriented Architecture (SOA), a UDDI (Universal Description Discovery and Integration) registry provides the main responsibility to advertise and discover services, which centralizes all responsibilities for publishing and discovering services. However, in the constrained M2M communication networks, the centralized registry can be considered a single point of failure. Once the service registry failed, all the information about registered services are unavailable. Moreover, data links for access to information on remote repository cannot always be guaranteed in the constrained M2M communication networks, due to low-power and low-bandwidth wireless devices deployed in remote and highly dynamic environments. Therefore, we need new mechanisms to compensate for the centralized registry for effective service discovery and network maintenance. However, any attempts of additional control could be difficult to achieve in this decentralized system due to the lack of a centralized server. In contrast, self-organization could be a good way to solve the control issues in the decentralized machine-to-machine communication networks. Self-organization is a process where the organization of a system spontaneously increases without being managed by an outside source.

Human society is a self-organizing system. Social networks are formed naturally by daily social interactions. A social community is a group of people with common interests, goals or responsibilities which is formed spontaneously. By using social networks, people can find some acquaintances that potentially have knowledge about the resources they are looking for.

Similarly to social networks, where people are connected by their social relationships, two autonomous network nodes in the M2M communication networks can be connected for service discovery and utilization. The similarity between M2M communication networks and social networks, where network nodes can be considered as people and connections can be regarded as relationships, makes us to believe the social theories will be useful in the design of M2M communication networks for improving the performance of service discovery.

3.1 System Design

In this section, we propose a socio-ecological model based on eight different social strategies which are derived from social theories [Haythornthwaite 1996; Kautz, Selman and Shah 1997; McCarty 2002; Milgram 1967; Newcomb 1975; Waloszek 2002; Watts, Dodds and Newman 2002; Watts and Strogatz 1998].

Socio-ecological strategy 1: in social networks, people remember and update potentially useful knowledge from social interactions, and then random and diffuse behaviors gradually become highly organized [Newcomb 1975]. For example, if a man named Bob successfully borrows an Oxford English dictionary from his friend Alice, this successful interaction will be remembered by Bob. When Bob needs this dictionary again, he will preferentially seek help directly from Alice rather than other people. However, if Alice does not have the Oxford English dictionary and cannot provide any useful information to help Bob find this dictionary, Bob would avoid contacting Alice if he needs the dictionary again in the near future.

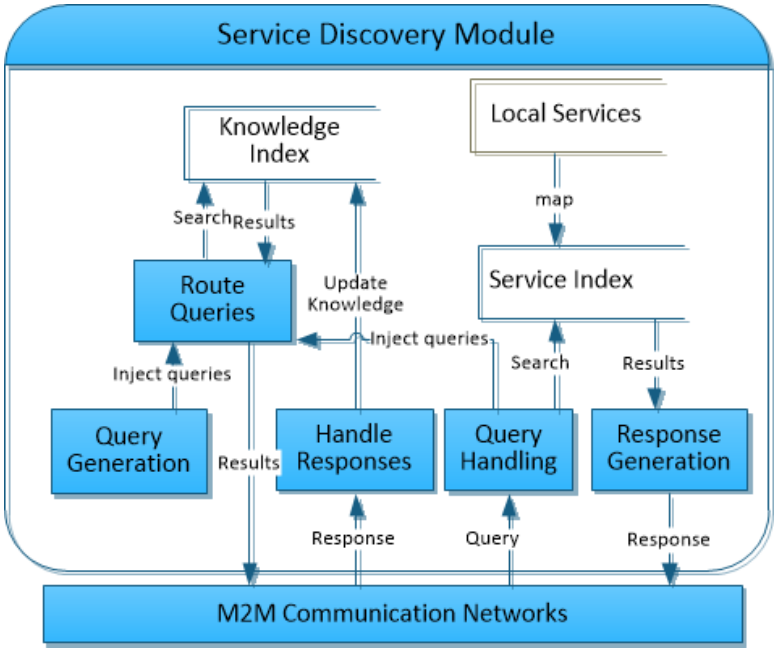

Fig. 1. Structure of Service Discovery Module.

Similarly to people in social networks, each node in the SESD network builds a knowledge index that stores associations between functions and the related nodes by the results of searches. Each knowledge index includes two sub-lists (as shown in Fig. 1): a friend list and a black list. In the friend list, network nodes that have successfully responded to the query are associated with the requested functions. The query originator will put a neighboring node into the black list associated with the query function, if no results on this function are found by the node nor its successor(s).

SESD uses a TTL (Time-to-Live) to prevent infinite propagation. TTL represents the number of times a message can be forwarded before it is discarded. In the SESD network, a newly joined node will send its first query to a set of randomly selected nodes. As shown in Fig. 1, if a search is successful, the query originator updates its friend list to associate the nodes that have responded successfully to the query with the requested function. The new obtained knowledge is stored in the local friend list. In the meantime, the query originator also removes invalid cached knowledge in the local friend list according to the results of searches. The black list is also updated with search results. If the query originator sends a request to a node, but no results about the

requested function are found by queries via this node, this node will be put into the black list of the query originator associated with the requested function.

Therefore, each node can learn from the results of previous searches, which makes future searches more focused. When more searches have been done, more knowledge can be collected from search results. If this process continues, each node can cache a great deal of knowledge that is useful to quickly find the nodes with the required resources in the future.

In SESD, the information saved in the sub-lists of knowledge index (including the friend list and the black list) are maintained in a two-dimensional map. A list can contain a maximum of n functions and each function is associated with a maximum of m nodes, so that a maximum of $n \times m$ pairs of associations between query functions and network nodes can be stored in a list. A Least Recently Used (LRU) policy is used to maintain a list without duplicates, where the most recently used function is at the top and the least recently used at the bottom. When a new association between a query function and network node is added into the list, it will check whether the pair has already existed in the list in order to avoid duplication. The advantages of LRU for removing old index items is that it has constant time and space overhead and can be very efficiently implemented. The least recently used function will be discarded when the list reaches its maximum n. In the same way, the nodes associated with each function are sorted by time last seen, where the most recently seen node at the head and the least recently seen node at the end. The least recently seen node will be dropped when the maximum size m is reached. The nodes stored in the black list are temporarily excluded from the node selection process associated with specific queries. These nodes will be included into the selection process when the associated information is dropped when the maximum size is reached. Therefore, there is no additional overhead to main the knowledge index. The old or out-of-date information will automatically dropped to the bottom of the list and will be updated when the list reaches the maximum.

Socio-ecological strategy 2: in social networks, some events with associated people fade from a person's memory with time [Cowan et al. 1999] and a personal network is adjustable with changing environments[Fisher and Lipson 2004].

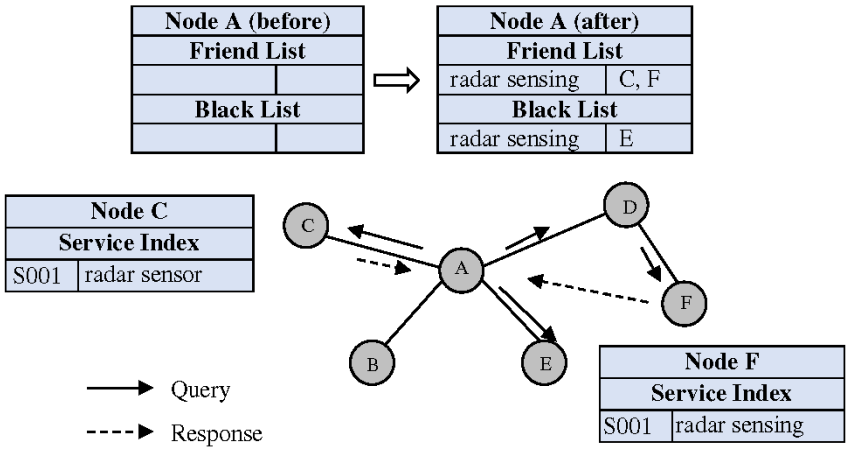

Fig. 2. Example of a knowledge collection.

In the SESD network, network nodes can update their knowledge about other nodes from daily search results. Some old and invalid knowledge is replaced by new obtained knowledge. The invalid knowledge that fails to be updated with daily searches will drop to the bottom of the lists and will be removed by using a LRU policy, when the list reaches a maximum.

Fig. 2 illustrates an example of knowledge collection process of a network node. Node *A* with an empty knowledge index searches for the services on the function "radar sensing". Since no

information is cached, node A will send the query to the randomly selected nodes: node *C*, node *D* and node *E*. Then node *D* further forwards the query to its neighboring node *F*. In this case, the search is successful at node *F*. Thus, node *F* responds to node *A* with the requested function "radar sensing". Node *A* then associates node *F* with the function "radar sensing" into the local friend list. For a following query on "radar sensing", node *A* can find node *F* directly associated with the query from the local friend list and preferentially send the query to node *F* rather than node *D* and node *E*.

As shown in Fig. 2, node *A* updates its black list with search results. Node *E,* a neighbor of node *A*, will be added into the black list associated with the requested function "radar sensing", because no results are found by node *E* nor its successors. But node *D* will not be put into the black list, because node *D*'s successor, node *F*, helps node *D* find the requested services successfully. If node *A* does the same search on "radar sensing" again, it will avoid forwarding the query to node *E*, regarding the information in the black list.

3.2 Routing Algorithm for Simple Queries

3.2.1 Query Processing

When a node generates a query, it will search the local friend list to find some associated nodes. The query originator will prefer to send the query to these nodes found from the friend list and avoid sending the query to associated nodes in the black list (Fig. 3). The query originator also attaches a list of "black nodes" associated with the query function regarding the local black list to help message receivers select nodes.

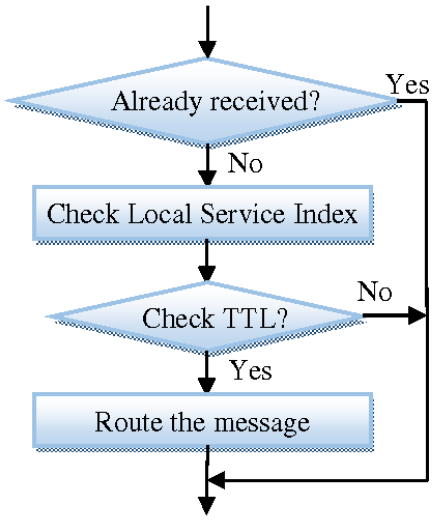

Fig. 3. Flowchart of query handling.

As shown in Fig. 3, when a network node receives a query, it will first check whether the message has been already received. Redundant messages will be discarded without further processing. Then the node will search the local content index to find matched services. If the query needs to be further forwarded ($TTL > 0$), the message receiver will use the local knowledge index to find associated nodes using the SESD routing algorithm and multicast the query to these nodes.

Socio-ecological strategy 3: in a social network, communities are self-organized with regard to the common interests. In the SESD network, because links between nodes are built according to the results of searches, a node has more probability to link to other nodes with the same interests, which have services of interest to him/her, with a high degree of likelihood. Therefore,

network nodes that have the same interests will be highly connected to each other and form a virtual community spontaneously, which is a similar environment to Watts's model [Watts and Strogatz 1998] in social networks.

3.2.2 Adaptive Forwarding

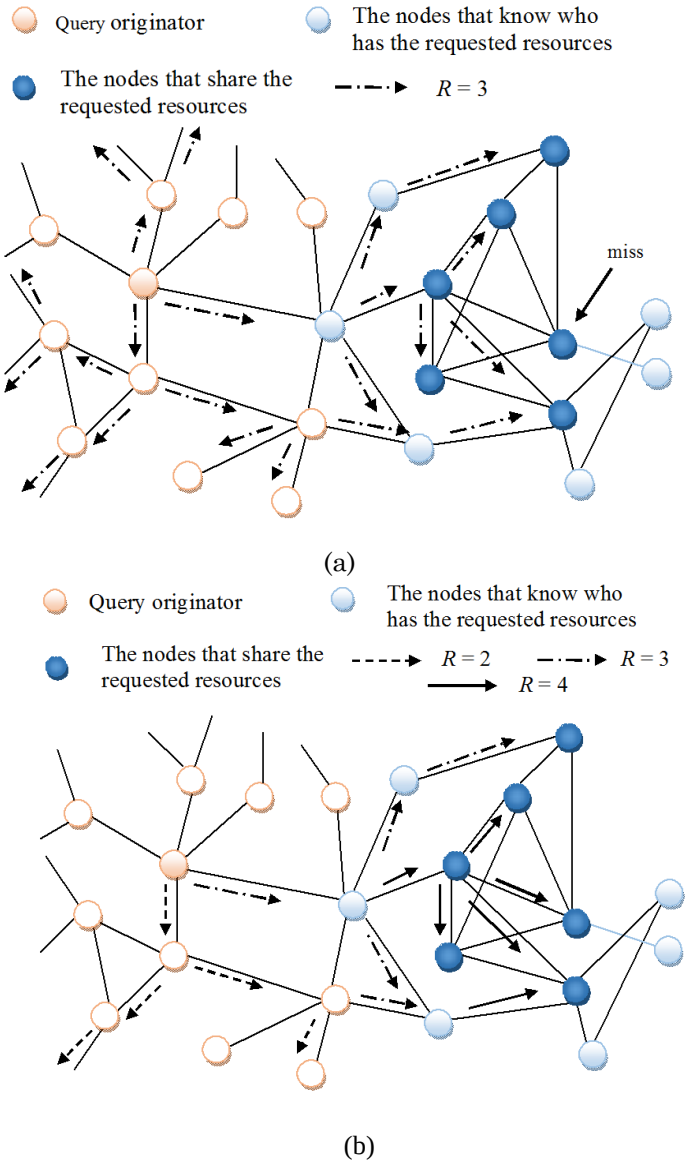

Fig. 4. Routing strategy comparison (a) static routing strategy (b) adaptive routing strategy.

Analogously to social networks, SESD utilizes a semantic approach to route queries to a selected subset of neighboring nodes. In some existing query routing methods (e.g., [Joseph July 2003; Lv et al. June 2002; Maymounkov and Mazi`eres March 2002]), the number of nodes to be forwarded in each hop is set to a static value. A fixed number of nodes are selected in each hop neglecting the probability of finding the requested service in these nodes. In order to conduct a more efficient search in SESD, the number of nodes to be forwarded is adjustable according to the correlation degree of the selected node to the query between a minimal

number $FN_{\min}$ and a maximum number $FN_{\max}$. When the correlation degree of a selected node is high, there is a high possibility that the selected node may share a desired service or has the knowledge about who shares the service. The probability of forwarding the query to this node should also be high by defining a high cut-off of node selection. In contrast, if the correlation degree is low, a low cut-off should be set to limit the scope of querying. Therefore, SESD is especially suitable in the resource-constrained M2M communication networks, as it can autonomously adjust the forwarding degree in message routing and optimize the bandwidth use in M2M commutation networks.

Fig. 4 shows examples of two routing strategies with a static number of receivers per hop (R = 3) and an adaptive number of receivers per hop (R = 2 ~ 4), respectively. In Fig. 4, the forwarding degree is increased to 4 for the nodes that share the requested resources, while the forwarding degree is adapted to 3 for the nodes that are highly correlated with the query and know who has the requested resources. As shown in Fig. 4, the adaptive routing strategy achieves better search performance by finding more target nodes with fewer query messages.

3.2.3 Node Selection

Socio-ecological strategy 4: for resource discovery in social networks, people usually recall information in memory to find the right people to contact. The persons recalled from memory may directly relate to their requests. For example, Bob wants to borrow an Oxford English dictionary and remembers that he once borrowed it from his friend Alice. Therefore, he can directly contact Alice again for the dictionary.

Socio-ecological strategy 5: however, in most circumstances, people cannot find the persons who are directly related to their requests, but people can find some acquaintances that potentially have knowledge about the resources they are looking for, or can provide background information or give advice on how to find the resources [Waloszek 2002]. For example, Bob may never have borrowed or cannot clearly remember whether he has ever borrowed an Oxford English dictionary. But Bob believes his friend Alice, who is a linguist, probably has the dictionary or at least she has more knowledge about who has the dictionary. In this case, the Oxford English dictionary is in the area of linguistics and Bob has found that Alice has abundant knowledge on the interest area of linguistics from previous intercommunications. Alice probably does not have the dictionary, but she will use her own knowledge to help Bob find the dictionary with a high likelihood.

The node selection procedure of SESD is shown in Fig. 5, where n is the number of network nodes that have already been selected in the first and second phase of node selection. Message receivers only trust the information about the "black nodes" provided by the query originator or by their own local black list concerning the security of information. When a node receives a query which needs to be further forwarded, the local routing algorithm will exclude a list of black nodes provided by the query originator from the node selection procedure together with the nodes associated with the requested function in the local black list.

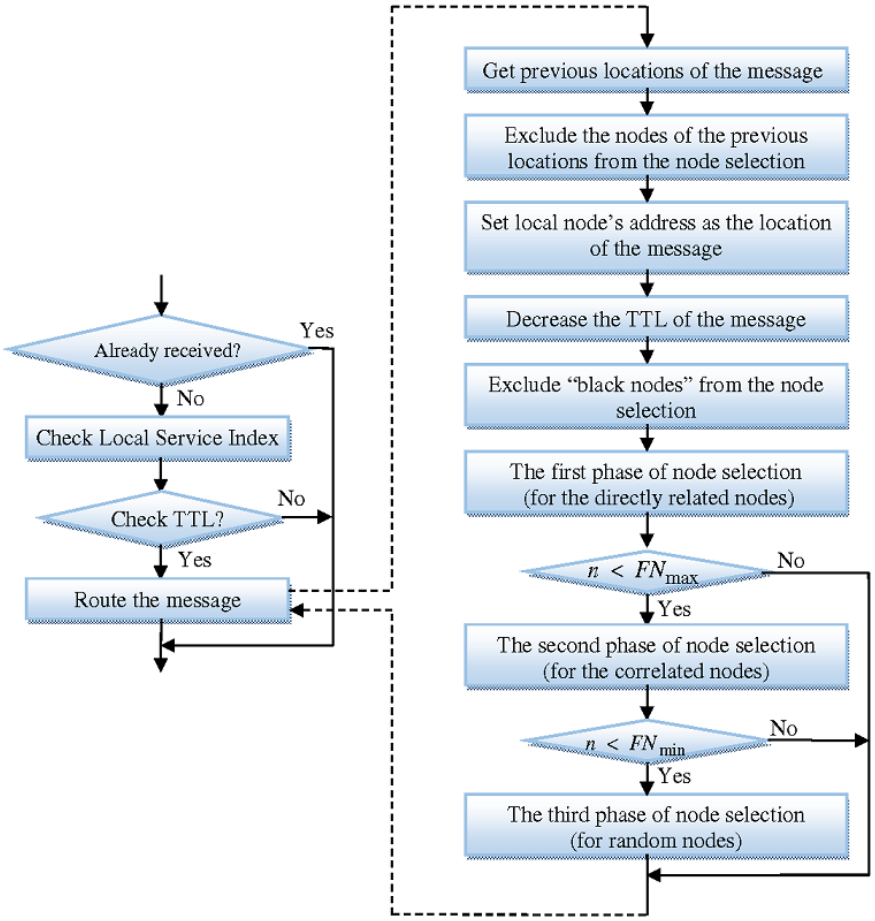

Fig. 5. Flowchart of the node selection procedure.

The routing algorithm then starts the three-phase procedure of node selection: searching for the directly related nodes, searching for the correlated nodes and searching for random nodes, from the local friend list. In the first phase, the routing algorithm searches for the nodes directly associated with the requested function from the local friend list and sorts them with the time of receipt. The node that is inputted or updated most recently will be selected first. These directly associated nodes have the greatest likelihood of finding the requested services. Hence, at most $FN_{\max}$ nodes will be forwarded.

However, the success probability of finding $FN_{\max}$ directly associated nodes in the first phase is very low, especially for new nodes with little knowledge about the M2M communication network. If there are not enough directly associated nodes found in the first phase, the algorithm will move to the second phase that searches for the nodes sharing content associated with the interest area of the requested function in the local friend list. An interest area in SESD is a semantic area with a set of functions. The corresponding interest area of a specific function and the other functions in this interest area can be found from the Open Directory Categories, which is the most widely distributed database with a hierarchical structure. SESD users use the common semantic hierarchy of the Open Directory to formalize a query.

These nodes sharing content associated with the other functions in the same interest area of the requested function will be sorted according to the degree of correlation to the interest

area of the requested function. The routing algorithm prefers to select the nodes with higher degrees of correlation rather than the nodes with lower degrees of correlation. If two or more nodes have the same correlation degree, the node that responded most recently will be put first.

Socio-ecological strategy 6: searching for a piece of information in social networks is most likely a matter of searching the social network for an expert on the function together with a chain of personal referrals from the searcher to the expert [Kautz, Selman and Shah 1997]. One of the most effective channels for disseminating of information and expertise as well as finding information within an organization is his/her social network of collaborators, colleagues and friends [Kautz, Selman and Shah 1997]. The experts who has abundant knowledge on this area will be more likely to help find the required information successfully.

Socio-ecological strategy 7: in social networks, a person does not need to tell everybody he/she is an expert in the areas which has been indicated with his/her social behaviors.

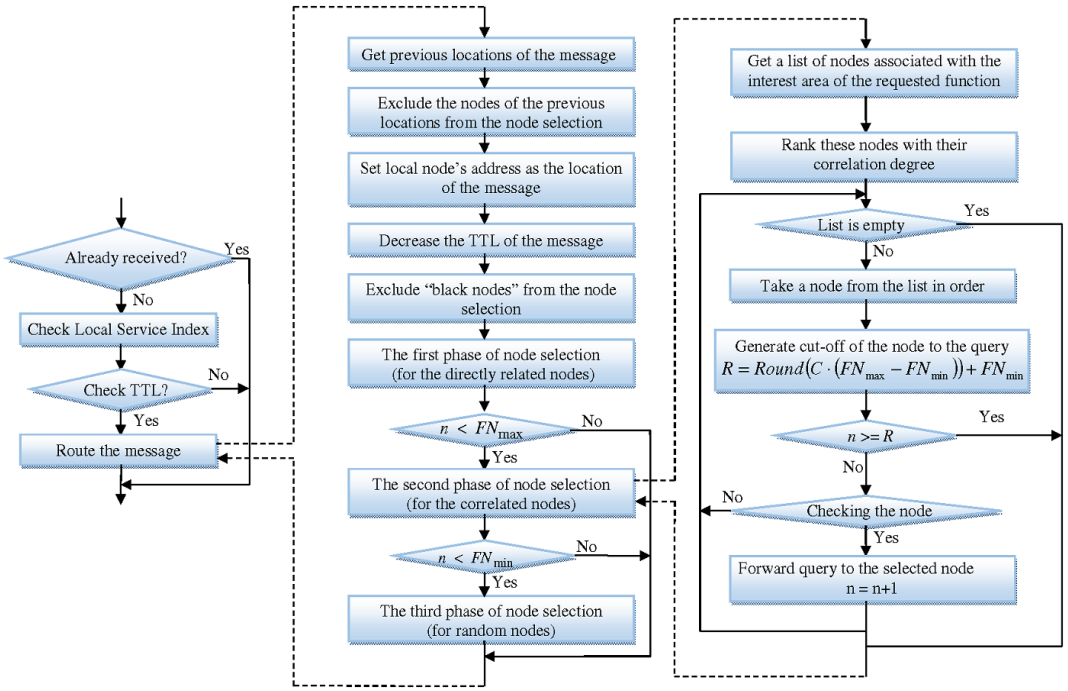

Fig. 6. Flowchart of the second phase of node selection.

In the SESD network, it is not necessary for a node to declare its interest since that has already been implied by its shared services. If a node has a large amount of content in a particular area like an "expert", it is very likely that it will also have other content or knowledge on this area. In our simulations, the correlation degree of a selected node in a particular area is generated by how many functions in the area the node is associated with, which could be calculated according the following formula.

$$C = \frac{n_{matches}}{n_{total}}, \qquad (1)$$

where $n_{matches}$ is the number of functions in this area that the node is associated with and n_{total} is the total number of functions in this area.

Different from any other distributed search algorithm, the cut-off criteria R for the second phase are different for different nodes between the maximum number of receivers $FN_{\max}$ and

minimum number of receivers $FN_{\min}$. The cut-off criterion R of a node with respect to the query is determined by the correlation degree of the node to the interest area of the requested function with the equation:

$$R = Round\left(C \cdot \left(FN_{\max} - FN_{\min}\right)\right) + FN_{\min}, \qquad (2)$$

where the function $Round$(x) returns the closest integer to the given value x. When the correlation degree of a node is very low ($C \approx 0$), there is a low likelihood to find the requested services from the node. Therefore, the probability of querying the node should be low with a low cut-off ($R \approx FN_{\min}$). In contrast, when the selected node is highly correlated with the area of the requested function by matching most functions in this area ($C \approx 1$), the cut-off of the node $R \approx FN_{\max}$.

The flowchart in Fig. 6 shows the second phase of the node selection algorithm. As shown in the flowchart, the nodes associated with the interest area of the requested function are sorted with their correlation degrees. Because nodes are taken from the list in order, the cut-offs of these nodes R will decrease with the reduced correlation degrees C (according to Equation (2)). But n is increased by one for each new node selected. The query will be sent to the selected nodes only when the number of selected nodes n is smaller than its cut-off to the query R ($n < R$).

If $n \geq R$, node selection procedure is completed in the second phase. If all nodes associated with the area of the requested function ($C > 0$) have been taken from the list in the second phase but there are still not enough nodes selected $n < FN_{\min}$, the selection procedure will move to the third phase to randomly pick up nodes from the rest of cached nodes irrelevant to the requested function as well as its interest area ($C = 0$) and forward the query to them, until $FN_{\min}$ nodes are forwarded ($n = FN_{\min}$).

3.2.4 Query Routing Example

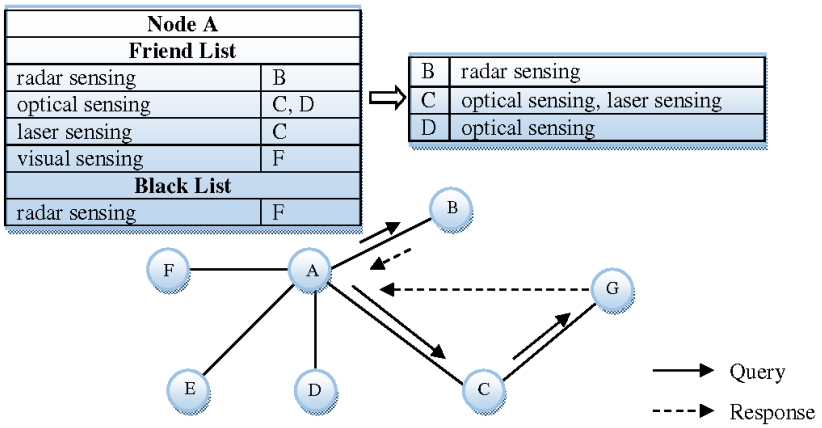

Fig. 7. Query routing example (two nodes selected).

Fig. 7 illustrates a simple example of query routing with SESD where $FN_{\max} = 5$ and $FN_{\min} = 1$. Node A receives a query with the function "radar sensing". Node A will first check the black list to exclude node F which is associated with the requested function "radar sensing" in the black list. In the first round of selection, node B is selected from the local friend list which is directly associated with the requested function "radar sensing". The number of selected nodes n is increased by one: $n = 0 \rightarrow n = 1$.

Due to $n < FN_{\max}$ ($n = 1$, $FN_{\max} = 5$), node A further searches for the nodes associated with the functions "optical sensing", "laser sensing", and "visual sensing" which are from the same interest area of the requested function "radar sensing". In this case, node A gets node C and node D associated with these functions from the friend list. Since node C is associated with two functions "optical sensing" and "laser sensing", but node D is associated with only one function "optical sensing" in the area composed by the four functions, node C ($C_C = \frac{2}{4}$) is more correlated with the area of sensing than node D ($C_D = \frac{1}{4}$) according to the cached knowledge. Hence, node C will be sorted on the top of node D in the list.

The cut-off of node C is $R_C = \frac{2}{4} \cdot (5-1) + 1 = 3$ and the cut-off of node D is $R_D = \frac{1}{4} \cdot (5-1) + 1 = 2$ by using Equation (2). The query will be sent to node C, because the number of selected nodes is smaller than the cut-off of node C: $n < R_C$ ($n = 1$, $R_C = 3$). Then $n = 1 \rightarrow n = 2$. The selection procedure will complete and node D is not selected, because $n \geq R_D$ ($n = 2$, $R_D = 2$). The actual number of queried nodes is *two* in this case.

Node C may not have the requested services, but it will use its own cached knowledge to propagate the query further and try to find the nodes for the query that will have a higher likelihood of success. In this example, node C knows that node G is associated with the requested function according to its local friend list and the requested service is obtained in node G.

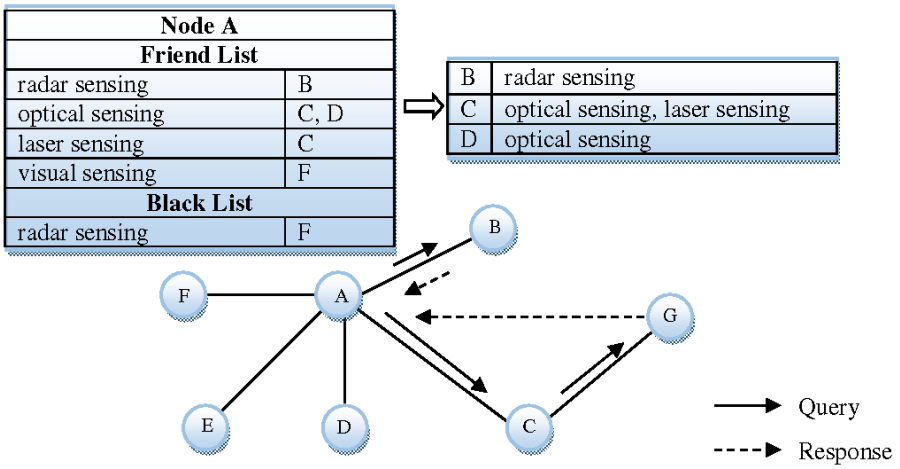

Fig. 8. Query routing example (three nodes selected).

Fig. 8 illustrates a similar example, where node D is associated with one more function "visual sensing" in the same area of the requested function. In this case, node D is also selected, because $n < R_D$ (where $n = 2$ and $R_D = \frac{2}{4} \cdot (5-1) + 1 = 3$)) and the actual number of nodes to be queried is *three* in the second example.

3.3 Routing Algorithm for Multi-Function Queries

3.3.1 Routing Algorithm for Conjunctive Queries

SESD also supports queries with conjunctive functions, such as a query on "radar sensing" and "VHF". For a conjunctive query with multiple functions, a search is successful on a node if a requested service is found by matching all query functions. This node will respond to the query originator with the requested functions. Moreover, the nodes that can find any requested function of the conjunctive query will also inform the query originator with the matched function(s), even though they cannot find the requested service completely matching all query functions. This response information will be cached by the query originator for future queries.

Similarly to the single-function query processing, when a node receives a conjunctive query Q which needs to be forwarded, the "black nodes" that are provided by the query originator and the "black nodes" that are associated with any requested function in the local black list will be excluded from the three-phase selection procedure.

In the first phase, the nodes that can completely satisfy the conjunctive query will be selected from the local friend list with $FN_{\max}$. If $n < FN_{\max}$, the node selection procedure will continue to the second phase to find the nodes that are correlated with the query as shown in Fig. 9. Two kinds of nodes will be considered: the nodes that are associated with any function of the conjunctive query and the nodes that are associated with the interest area of any function of the conjunctive query.

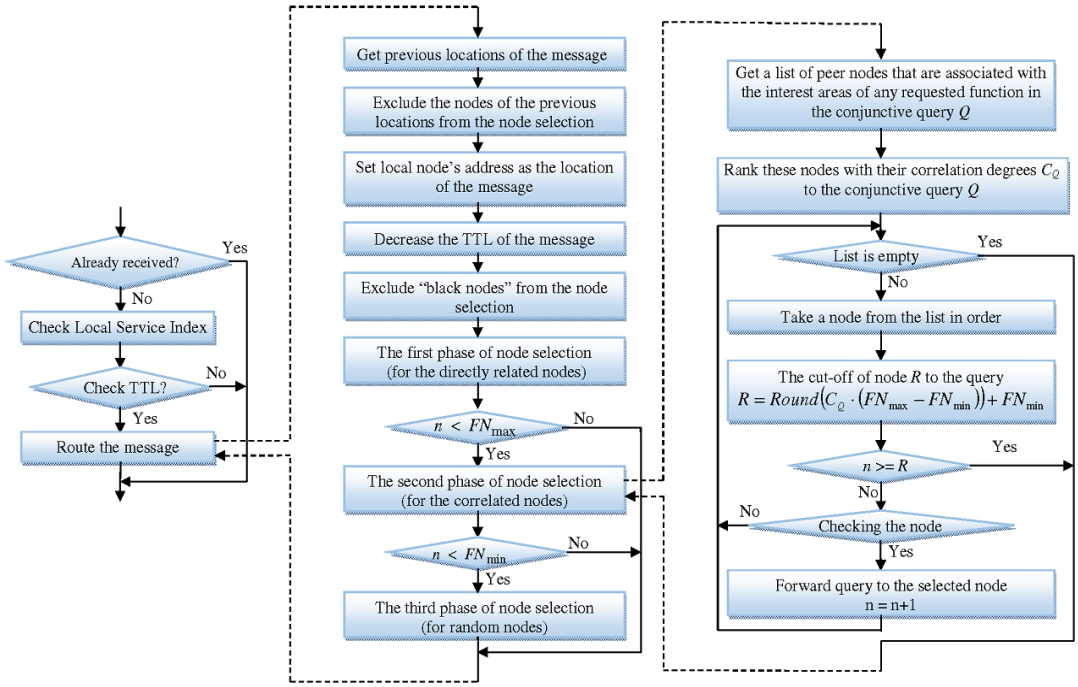

Fig. 9. Flowchart of the second phase of node selection procedure for multi-function queries.

Analogously to the single-function query processing, the correlation degree of a node to the interest area of a requested function T is given by Equation (1):

$$C_T = \frac{n_{matches}}{n_{total}} .$$

For a conjunctive query with N functions $\{t_1 \wedge t_2 \wedge t_3 \ldots . \wedge t_N\}$ (where $\wedge$ represents the logical conjunction function: "and"), a node is associated with $\{t_1, t_2, \ldots., t_m\}$, and is not associated with $\{t_{m+1}, t_{m+2}, \ldots., t_{m+k}\}$ but with corresponding correlation degrees $\{C_{m+1}, C_{m+2}, \ldots, C_{m+k}\}(N = m + k)$.

The correlation degree $C_{indirect}$ of the interest areas that the node is not associated with is given by:

$$C_{indirect} = E(C_{t_i}) = \frac{1}{k} \sum_{i=m+1}^{m+k} C_{t_i} .$$

We prefer to forward the query to the nodes that are directly associated with the most functions of the conjunctive query (with ratio C_{direct}) and simultaneously highly correlated with the interest areas of the requested functions they are not associated to (with ratio $C_{indirect}$). The correlation degree of the node to the conjunctive query Q is given by:

$$C_Q = C_{direct} + (1 - C_{direct}) \cdot C_{indirect}$$

$$= \frac{m}{m+k} + \frac{1}{m+k} \cdot \sum_{i=m+1}^{m+k} C_{t_i} \quad (i = m+1,..., m+k). \qquad (3)$$

The cut-off criterion R of a node with respect to the conjunctive query is determined by the correlation degree of the node to the conjunctive query Q with the Equation (2):

$$R_Q = Round(C_Q \cdot (FN_{\max} - FN_{\min})) + FN_{\min} .$$

If $n < FN_{\min}$, the selection procedure will proceed to the third phase to randomly pick up nodes from the rest of cached nodes irrelevant to the requested functions as well as their interest areas ($C = 0$) with cut-off $FN_{\min}$.

3.3.2 Routing Algorithm for Query Packs

A query pack consists of several conjunctive queries, like ("VHF" and "radar sensing") or ("UHF" and "visual sensing"). To route a query pack, the node selection algorithm first excludes the nodes that are associated with any conjunctive query in the local black list and in the black list provided by the query originator. The node selection algorithm then tries to find the nodes that can satisfy any conjunctive query in the query pack from the local friend list. A maximum of $FN_{\max}$ nodes will be selected.

In the second phase, the routing algorithm produces a list of nodes that are associated with any function of the query pack and the nodes that are associated with the interest area of any function of the query pack. For a given node, the routing algorithm generates a list of correlation degrees $(C_1, C_2, ..., C_M)$ of these nodes to all conjunctive queries $(Q_1, Q_2, ..., Q_M)$ in the query pack $(Q_1 \vee Q_2 \ldots \vee Q_M)$ by using Equation (3) (where $\vee$ represents the logical disjunction function "or"). The query pack is successful if any conjunctive query in the query pack can be satisfied. We use the largest correlation degree of a conjunctive query as the correlation degree of the node to the query pack $C = MAX(C_1, C_2, ..., C_M)$ to generate the cut-off R of the node by using Equation (2).

These nodes are sorted according to their correlation degrees to the query pack. The query pack is forwarded to a given node only in case the cut-off R of the node is larger than the number n of selected nodes, which is the same as the single-function query and the single conjunctive query processing.

3.4 Adaptive Service Discovery

In order to efficiently search the network, two kinds of queries are generated at different stages of searches, known as ordinary queries and active queries. For the ordinary queries, the target nodes sharing the desired services will respond to the information related to the requested function only. For the active queries, the target nodes will not only respond to the requested function but also inform the query originator of other associated functions it shares in the same interest area. For example, when the query originator generates a query with the function "radar sensing", the target node that shares the desired services will answer the query about "radar sensing" as well as the associated functions "optical sensing", "laser sensing" and "visual sensing". The newly obtained information will be put into the local friend list by the query originator for future queries.

With these active queries, the query originator can gather more pieces of knowledge from each successful query, but extra traffic will be generated for shipping such additional knowledge. The extra traffic could be significant if every node generates all queries in this manner, which is difficult to be handled by bandwidth-limited networks. In contrast, if network nodes search the network only with ordinary queries, each new node accumulates knowledge slowly by gathering one piece of knowledge from each successful query, especially for those nodes which are seldom online or request the network.

To address these issues, SESD adopts a trade-off solution for the constrained M2M communication networks. The recently joined nodes will utilize active queries to quickly accumulate a large amount of useful information regarding their interests. After the cached knowledge reaches a certain threshold ratio r of the maximum size of the friend list $n_{\max\_knowledge}$, the nodes will use ordinary queries to discover the required services with low traffic cost. If the ratio of cached knowledge $r_{cached} \le r$, active queries will be used to search the network. Otherwise, ordinary queries will be adopted. Moreover, this process is not only applicable for recently joined nodes, but also enables nodes to quickly recover from unpredictable knowledge loss.

Socio-ecological strategy 8: the query generation strategy of SESD is similar to the human strategy in social networks where newly joined persons normally are more active to adapt to new environments. When a person joins to a new society, he/she will not only passively learn knowledge by remembering useful information from daily occasional events happening in the society, but also actively collect potentially useful knowledge consciously. When he/she seeks out help in a new society, communication with the other people who can be of assistance is not normally limited to a strict exchange of assistance; often, he/she would like to know more about the people who can be of assistance in order to expand his/her knowledge of the new society, which may be useful at a later time.

4. EVALUATION METHODOLOGY

In this section, we evaluate the performance of SESD by simulations in a dynamic M2M environment with a comparison to other relevant methods. The SESD simulator is programmed using the Java Language. The main components are illustrated in Fig. 10. At the beginning of the simulations, 1280 different high level functions were generated and distributed to 1000 services, and each service performed three functions. In our simulations, each peer node was randomly assigned a primary interest area and provided services to the network with a probabilistic method: these services are mostly relevant to the interest area of the node with a probability of 90%, but are occasionally irrelevant to this area. For services relevant to the primary interest area, at least one of the functions of each service should be in the interest area of the hosting node. A total of 32 interest areas are generated and each covers 40 functions.

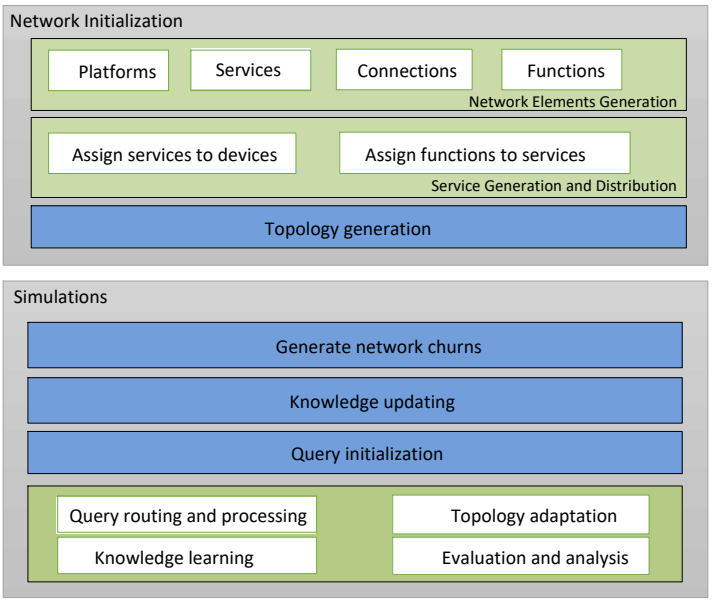

Fig. 10. Structure of the SESD simulator.

In order to better observe the evolution of network topology in the simulations, we set up a growing network started with a small-size random network (with 100 nodes). In the beginning of each simulation, each node randomly connects to four nodes bi-directionally to generate a random topology. Since there has been no interaction between nodes at the beginning of each simulation, each node keeps an empty knowledge index. Each list of the knowledge index can contain a maximum of 60 functions and each function can be associated a maximum of 60 nodes in the index. The threshold ratio r of SESD is defined as 80%. The simulation network starts from a small set of nodes (100 nodes). A number of nodes (30 nodes) join the network each day of the first month in each simulation until the network reaches 1000 nodes. Then the network becomes a mature network with 1000 nodes.

Each query is tagged by a TTL to limit the life time of a message to four hops in the simulations, if no other setting is mentioned. Ongoing changes have been generated by nodes frequently going online and offline due to limited battery life. In each time step of simulations, an online network node was chosen randomly as the originator to start a lookup. The selected node needs to search the network for a specific service on a different network node to deliver a joint capability. These query functions are generated with a probabilistic method: each function is randomly selected from the current primary interest area of the query originator with a probability of 90%, but sometimes from a random area with a probability of 10%.

At each loop of simulation, the performance was measured by the number of discovered services. Each peer node generates an average of two requests each day. Simulations were performed to trace the results of about two months (60 days, 92,100 time steps). Each average result is generated from the experimental results of each hour. Each average result is generated from the experimental results of each day.

5. SIMULATION RESULTS

Due to the complexity of SESD and numerous customized functions offered by the SESD simulator, there are many combinations of parameters to experiment with and lots of scenarios to test which could generate far too many graphs to analyze. In this section, we only present an analysis of simulation results from the most pertinent experiments as we see.

5.1 Performance Comparison to Relevant Methods

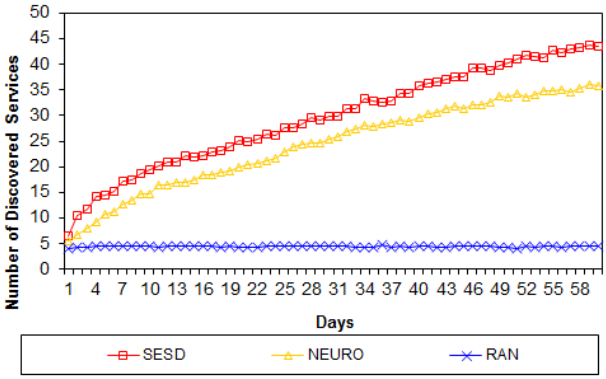

Fig. 11. Number of discovered services.

The initial simulation results are carried out to provide a comparison among the SESD, two relevant methods:

- RAN: a constrained Gnutella routing strategy. Received queries are randomly passed to FN_{min} connected nodes in each hop.
- NEURO: NeuroGrid routing strategy with the adaptive number of receivers. In each hop, received queries are passed to a maximum of FN_{max} nodes that are directly associated with the requested function from the local knowledge index. If not enough matches are found (< FN_{min}), the algorithm randomly forwards a query to FN_{min} nodes from the rest of the connected nodes.

From the results in Fig. 11, SESD achieves better performance and higher efficiency than NEURO and RAN by finding more services. As shown in Fig. 12, SESD-O needs a few more query messages introduced by the second phase of node selection procedure, but the search efficiency of SESD is still better than NEURO by achieving higher the number of discovered services per query message as shown in Fig. 13.

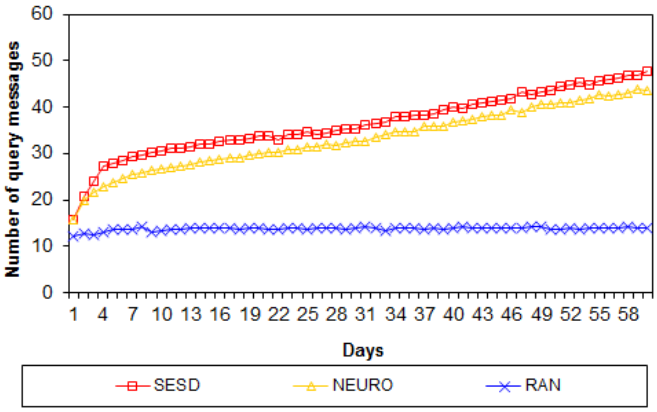

Fig. 12. Number of query messages.

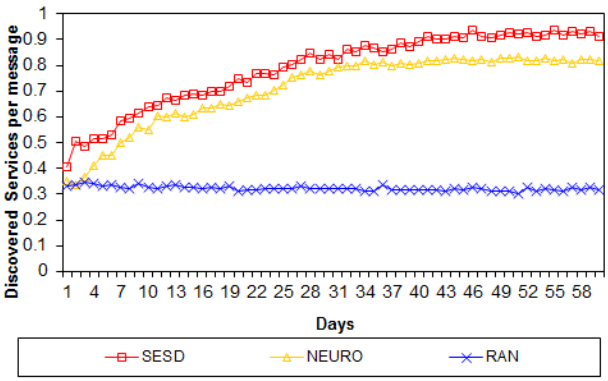

Fig. 13. Number of discovered services per message.

At the early stage of searches, it is very difficult for nodes to find the directly associated nodes with the requested function from the local knowledge index by using either SESD or NEURO method due to the limited knowledge cached, but SESD is capable of retrieving the nodes which share the associated services with the relevant functions more often. These selected nodes which are highly correlated with the interest area of the requested function have more knowledge about the query than randomly selected nodes. Thus, SESD can find the requested services more efficiently with the same knowledge. More successful searches, in turn, help to build the knowledge index more efficiently. Therefore, SESD-O has better search capabilities and better knowledge-collecting capabilities. With these advantages, SESD achieves better performance than NEURO and RAN.

5.2 Effects of Size of Lists

From this section, simulations employing different parameters are carried out to show the effects and contributions of simulation parameters (e.g., the size of lists, request structure, the number of receivers in each hop). The size of list in the knowledge index is an important parameter, which determines the storage overhead required for each node. In this section, we will compare the search performance of SESD with different sizes of lists.

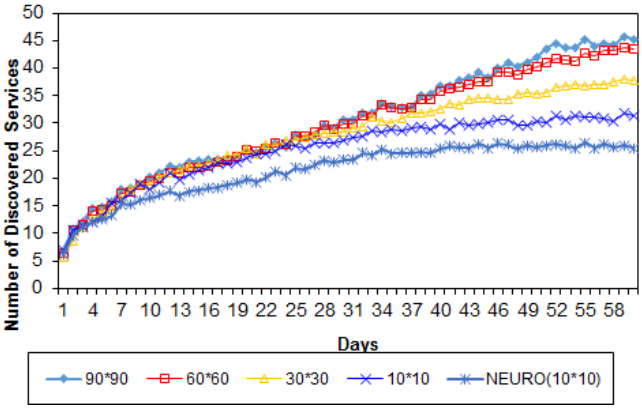

Fig. 14. The number of discovered messages with different sizes of knowledge index.

Fig. 14 shows the number of discovered services by SESD in the network where each node has a knowledge index with lists containing a maximum of 10×10, 30×30, 60×60 and 90×90 entries (pairs) between functions and associated node addresses, respectively. Clearly shown

in Fig. 14, the query originators have more difficulty in finding the requested services from the nodes with less knowledge. However, the number of discovered services only increases a little by changing the size of lists from 60×60 to 90×90 entries. This result suggests that a large knowledge index containing most commonly used functions achieves close performance to a knowledge index with an even larger size. As shown in Fig. 14, in the worst case of 10×10 entries, SESD still achieves obviously higher performance than NEURO due to its better knowledge collection capability and search capability.

5.3 Effects of Request Structure

SESD is simulated with different request structures. By using the SESD simulator, the requested function is selected from the primary interest area of the query originator with a probability p, but is from a random area with a probability $(1-p)$. In the case of $p=90\%$, 90% of the requested functions are randomly selected from the interest area of the query originator. On the contrary, in the case of $p=0\%$, a purely random function is chosen as the requested function which is the worst case since the query originator cannot benefit from the repeated queries in its interest area.

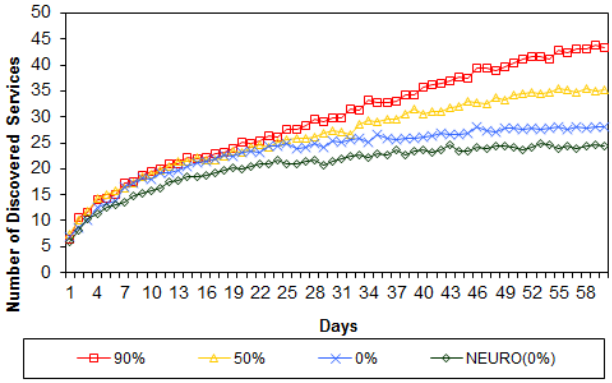

Fig. 15. The number of discovered services with different request structures.

Fig. 15 shows the results of the number of discovered services by SESD on the representative samples of p of 0%, 50% and 90%, respectively. In the simulations, the request scope is enlarged by setting a smaller p. Since the probability of finding directly associated nodes from the knowledge index decreases with smaller p, the number of discovered services decreases along with p, which means the nodes are generally more difficult to target the requested services in the network where users have very wide interests. But the performance of SESD is still better than that of NEURO even in the worst case of $p=0\%$ as shown in Fig. 15, because SESD can still find the nodes that potentially have the knowledge about queries even if the directly associated nodes cannot be found from the local knowledge index.

5.4 Effect of the minimum number of receivers

The minimum number of receivers FN_{min} and the maximum number of receivers FN_{max} are important factors to achieve adaptive query forwarding. In this experiment, SESD is simulated with different FN_{min} and FN_{max} to see the effect that each setting makes. As shown in Fig. 16 and Fig. 17, the number of discovered services achieved at the end of simulation increases by about 67% by changing FN_{min} from 2 to 3, while the number of query messages generated per query increases even more significantly by over 180%. This result shows that network traffic is very sensitive to the change of the parameter FN_{min}, where the number of

query messages jumps from $O\left(\sum_{i=1}^{4} 2^i\right) = O(30)$ to $O\left(\sum_{i=1}^{4} 3^i\right) = O(120)$ ($TTL = 4$). Therefore, flooding would occur in the network by setting a large $FN_{min.}$

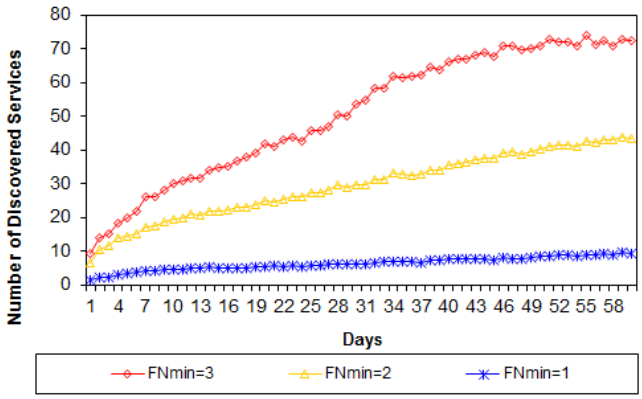

Fig. 16. Effect of the minimum number of receivers on the number of discovered services.

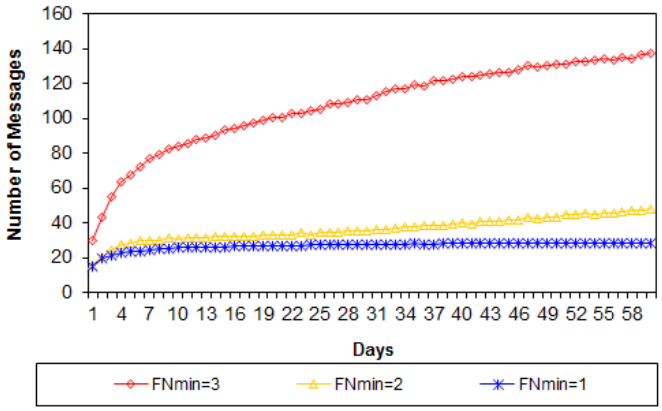

Fig. 17. Effect of the minimum number of receivers on the number of messages.

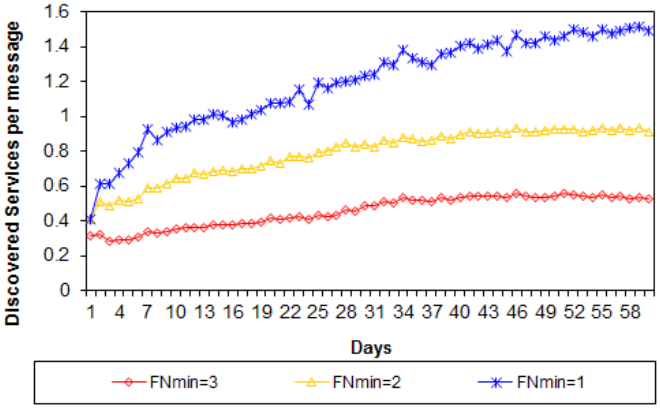

Fig. 18. Effect of the minimum number of receivers on the number of discovered services per message.

From the results in Fig. 18, the number of discovered services per message decreases clearly with increasing FN_{min}. However, if a very small $FN_{\min}$ is defined as $FN_{\min}=1$, a very limited nodes could be accessed for answering a query, which seriously affects the speed of knowledge collection. Therefore, the number of discovered service is very low in the case of $FN_{\min}=1$. From the results discussed above, we suggest that FN_{min} should be carefully defined by application developers who would consider adoption of SESD in their application. In this case, $FN_{\min}=2$ is a good trade-off to achieve both high search performance and efficiency.

5.5 Effect of the maximum number of receivers

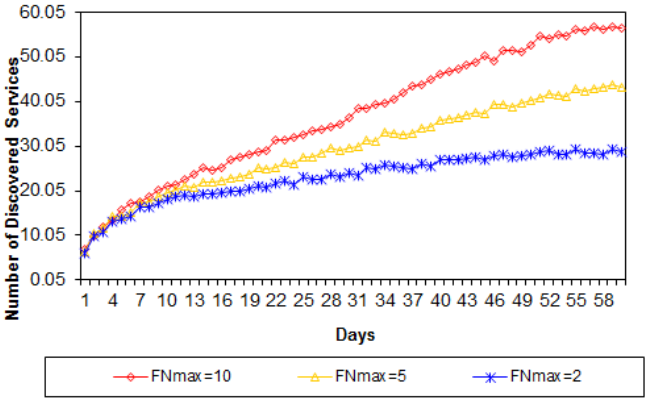

Fig. 19. Effect of the maximum number of receivers on the number of discovered services.

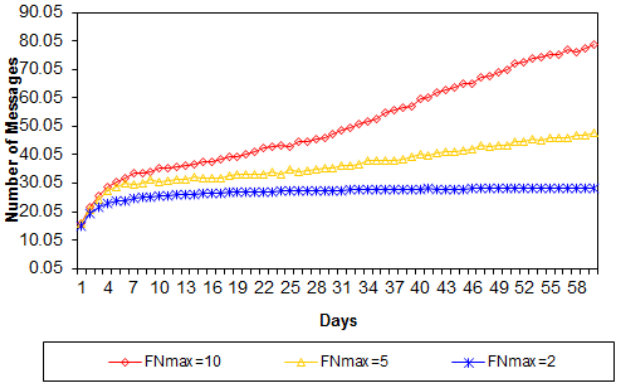

Fig. 20. Effect of the maximum number of receivers on the number of messages

As shown in Figs. 19 and 20 the number of discovered increases by about 30% while the number of messages increases about 65% by changing the maximum number of receivers in each hop FN_{max} from 5 to 10. Compared to the results with changing the minimum number of receivers FN_{min} shown in Fig. 20, the number of messages generated is more sensitive to the alternation of the minimum number of receivers FN_{min} rather than the alternation of the maximum number of receivers FN_{max}.

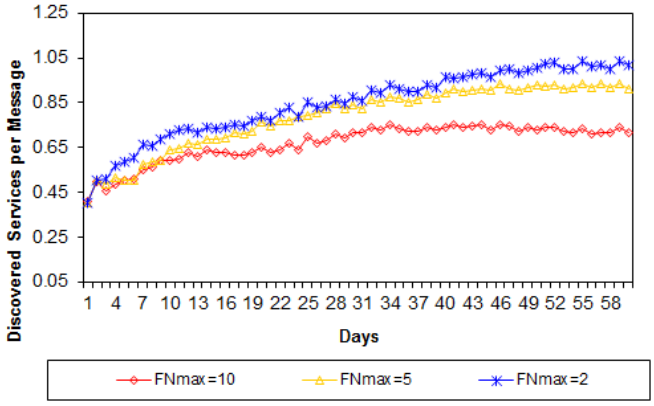

Fig. 21. Effect of the minimum number of receivers on the number of discovered services per message.

The number of receivers in each hop is adjusted according to the correlation of the selected nodes to the query. It is very difficult for a node to reach a very high correlation by matching most of functions in a specific area. The number of receivers in each hop is usually much less than the maximum number of receivers FN_{max}, while the number of receivers in each hop must be larger than the minimum number of receivers FN_{min}. Therefore, the number of messages generated is more correlated to FN_{min} rather than FN_{max}.

The number of discovered services per message is only slightly changed by alternation of FN_{max} as shown in Fig. 21 when compared to the alteration of FN_{min}. It shows that adaptive lookups have been achieved by SESD. Since a high efficiency is performed by a bigger FN_{max}, application developers could consider defining a bigger FN_{max} to discover more services rather than a bigger FN_{min}.

5.6 Query Packs

We continue to evaluate the performance of SESD by simulations with query packs. In this experiment, each query pack consists of two conjunctive queries and each conjunctive query contains two query functions, such as ("VHF" and "radar sensing") or ("UHF" and "visual sensing"). Because the matching probability decreases significantly to match both of the querying functions, in this experiment, the querying functions are chosen under the assumption that there is at least one possible service existing in the network that can satisfy the query pack.

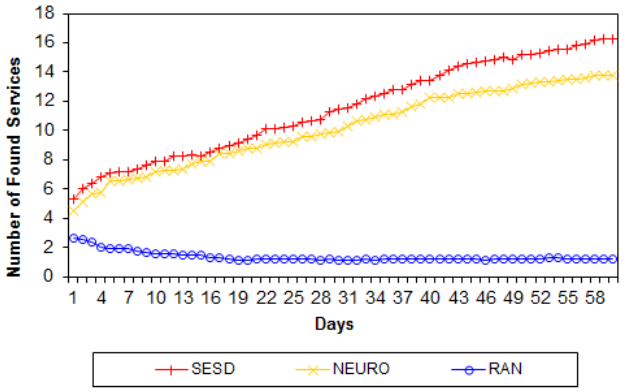

Fig. 22. Number of found services for query packs.

From the results in Fig. 22, SESD achieves better performance than NEURO and RAN, which is similar to that of single function lookups. Compared to single function lookups, the number of discovered services is less than that of single function lookups, because the number of services matching both two functions is less than the number of services matching one function. Different from single function searches, we are surprised to see that the number of found services by RAN decreases at the early stage of searches, while the number of found services by NEURO and SESD keep increasing as shown in Fig. 22. Because the querying functions are chosen under the assumption that there is at least one possible service existing in the network that can satisfy the query pack, the actual matching probability of RAN decreases with new adding services in the early stage, which leads to the decrement of the number of found services.

6. CONCLUSIONS

For resource discovery in social networks, people can directly contact some acquaintances that potentially have knowledge about the resources they are looking for. However, in current machine-to-machine communication networks, each peer node lacks a "social" network, making it difficult to route queries efficiently. In this paper, we have presented a socio-ecological model, SESD, for advanced service discovery in machine-to-machine communication networks by self-organizing autonomous nodes with socio-ecological strategies, which enable peer nodes to meet, get to know each other, and help each other. In the SESD network, each node can learn from the previous search results, both in success and failure, which makes future searches more efficient. No extra communication overhead is required for SESD to obtain additional information from neighboring nodes. Each node maintains a knowledge index (including a friend list and a black list) about resources located in the network. Each node works as an autonomous agent which provides local message processing and routing services. SESD is not able to hand the simple queries but also route the complex queries efficiently. SESD is not only efficient for the previously queried functions but also for the functions that have not been previously queried.

SESD enables nodes not only to forward queries to the "good" nodes that can potentially offer the requested resources or know who has the requested resources, but also to avoid sending queries to the "bad" nodes that cannot help find any requested services. SESD is a self-adaptive algorithm. The number of nodes to be forwarded in each hop is adaptive according to the correlation degree of the node to the query. Moreover, the different types of queries are also utilized in accordance with different search stages, which enables nodes to accumulate knowledge efficiently with low communication cost in the constrained M2M communication networks. SESD is an also self-organizing algorithm, the network is formed and maintained spontaneously, where network nodes are self-organized based on their daily intercommunications. Each node can automatically detect potential interests of other nodes in the network according to their previous behaviors and preferentially links to the nodes that have similar interests. Global behaviors then emerge as the result of all the local behaviors that occur. Finally, the nodes with similar interests will be highly connected to each other and form groups spontaneously in the M2M communication networks. The SESD has been evaluated in dynamic environments. The evaluation results show that the SESD has demonstrate a better performance and efficiency when compared with the existing methods.

In the future work, the real network deployment will be considered in the next step. The SESD tested and evaluated in a machine-to-machine communication network for disaster monitoring and relief. The SESD will be used to dynamically discover and self-configure the disaster relief services in a dynamic M2M communication environment. The SESD will also be implemented in different types of wireless sensor networks to address existing problems of different domains.

REFERENCES

S. Cheshire and M. Krochmal. 2013. DNS-Based Service Discovery. In RFC 6763. ISSN: 2070-1721, Internet Engineering Task Force.
Nelson Cowan, Lara D Nugent, Emily M Elliott, Igor Ponomarev and J. Scott Saults. 1999. The Role of Attention in the Development of Short-Term Memory: Age Differences in the Verbal Span of Apprehension. Child Development, Vol 70, 1082–1097.
K. M. Fisher and Joseph I. Lipson. 2004. Information Processing Interpretation of Errors in College Science Learning. Instructional Science, Vol 14, 49-74.
Caroline Haythornthwaite. 1996. Social Network Analysis: An Approach and Technique for the Study of Information Exchange. Library & Information Science Research, Vol 18, 323-342.
A. J. Jara, P. Martinez-Julia and A. Skarmeta. 2012. Light-Weight Multicast DNS and DNS-SD (lmDNS-SD): IPv6-Based Resource and Service Discovery for the Web of Things. In 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS).
Sam Joseph. July 2003. P2P MetaData Search Layers. In International Workshop on Agents and Peer-to-Peer Computing, Melbourne, Australia.
Sam Joseph. May 2002. NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks. In International Workshop on Peer-to-Peer Computing, Pisa, Italy.
Henry Kautz, Bart Selman and Mehul Shah. 1997. Combining Social Networks and Collaborative Filtering. Communications of ACM, Vol 40, 63-65.
Ronny Klauck and Michael Kirsche. 2012. Bonjour Contiki: A Case Study of a DNS-Based Discovery Service for the Internet of Things. In International Conference on Ad-hoc, Mobile, and Wireless Networks Springer-Verlag, 316-329.
Dongpil Kwak, Joongsoo Lee, Seoungku Kim and Younghee Lee. 2008. A New Address Scheme for Service Discovery supporting Active Mobile Sensor Objects. In 10th International Conference on Advanced Communication Technology, 765 - 768.
Xu Li, N. Santoro and I. Stojmenovic. 2009. Localized Distance-Sensitive Service Discovery in Wireless Sensor and Actor Networks. IEEE Transactions on Computers, Vol 58, 1275-1288.
Antonio Liotta. 2013. The cognitive NET is coming. IEEE Spectrum, Vol 50, 26-31.
Lu Liu, Duncan Russell, David Webster, Zongyang. Luo, Colin Venters, Jie Xu and John Davies. 2009. Delivering Sustainable Capability on Evolutionary Service-oriented Architecture. In IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC 2009), Tokyo, Japan, 12-19.
Konrad Lorincz, David J. Malan, Thaddeus R.F Fulford-Jones, Alan Nawoj, Antony Clavel, Victor Shnayder, Geoffrey Mainland, Matt Welsh and Steve Moulton. 2004. Sensor Networks for Emergency Response: Challenges and Opportunities. IEEE Pervasive Computing, Vol 3, 16-23.
Qin Lv, Pei Cao, Edith Cohen, Kai Li and Scott Shenker. June 2002. Search and Replication in Unstructured Peer-to-Peer Networks. In ACM SIGMETRICS, Marina Del Rey, CA.
R. Marin-Perianu, H. Scholten, P. Havinga and P. Hartel. 2006. Energy-Efficient Cluster-Based Service Discovery in Wireless Sensor Networks. In 31st IEEE Conference on Local Computer Networks.
Petar Maymounkov and David Mazi`eres. March 2002. Kademlia: A Peer to Peer Information System Based on the XOR Metric. In Internation Workshop on Peer-to-Peer Systems, Cambridge, MA.
Christopher McCarty. 2002. Structure in Personal Networks. Journal of Social Structure, Vol 3, 1-19.
Stanley Milgram. 1967. The Small World Problem. Psychology Today, Vol 2, 60-67.
Theodore M. Newcomb. 1975. Social Psychology : the Study of Human Interaction. - 2nd ed. Revised Routledge and Kegan Paul, London.
Åke Ostmark, Jens Eliasson, Per Lindgren, Aart van Halteren and Lianne Meppelink. 2012. An Infrastructure for Service Oriented Sensor Networks. Journal of Computers, Vol 1, 20-29.
Carmelo Ragusa, Antonio Liotta and George Pavlou. 2005. An adaptive clustering approach for the management of dynamic systems. IEEE Journal on Selected Areas in Communications, Vol 23, 2223-2235.
Kaouthar Sethom and Hossam Afifi. 2005. A New Service Discovery Architecture for Sensor Networks. In Wireless Telecommunications Symposium.
Z. Shelby, S. Krco and C. Bormann. 2013. CoRE Resource Directory. IETF Internet-Draft, Standards Track, Vol.
Berta Carballido Villaverde, Rodolfo De Paz Alberola, Antonio J. Jara, Szymon Fedor, Sajal K. Das and Dirk Pesch. 2014. Service Discovery Protocols for Constrained Machine-to-Machine Communications. IEEE Communications and Tutorials, Vol 16.
Gerd Waloszek. 2002. Personal Networks. In SAP AG, Product Design Center.
Duncan J. Watts, Peter Sheridan Dodds and M. E. J. Newman. 2002. Identity and Search in Social Networks. Science, Vol 296, 1302-1205.
Duncan Watts and Steven H. Strogatz. 1998. Collective Dynamics of Small-World Networks. Nature Vol 393, 440-442.
Yan Wu, Chungang Yan, Lu Liu, Zhijun Ding and Changjun Jiang. 2015. An Adaptive Multilevel Indexing Method for Disaster Service Discovery. IEEE Transactions on Computers, Vol 64, 2447-2459.

Received 31 March 2015; revised 6 July 2015; accepted 30 July 2015