Dominating-Set-Index based Searching Algorithm in Mobile P2P Networks

Cheng Chang[1], Ye Wang[2], and Jie Sun[3]
[1]College of Computer Science, Zhejiang University, Hangzhou, China
[2] School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China
[3]Ningbo Institute of Technology, Zhejiang University, Ningbo, China
Email: changcheng316@zju.edu.cn; yewang@zjgsu.edu.cn; jiangbin@zju.edu.cn

Abstract **—**Because of the isomorphism between P2P networks and mobile ad hoc networks (MANET), many research communities are focusing on the work based on the combination of P2P and the backbone strategy, and this produces a field called mobile peer-to-peer (MP2P) networks. An effective algorithm for searching and retrieving information has proven to make a great impact on the performance of MP2P systems. Based on the connected dominating-set (CDS), which overlays the whole Mobile P2P ad hoc network, we propose a dominating-set-index based searching algorithm, named DSI, using dominating set indices to maximize the return of search results. DSI periodically advertises information of shared resources on the backbone of current network topology in a distributed fashion. Similar information will be clustered close to the backbone network. This can accelerate the searching process, as the resource location information is placed close to normal nodes. Both theoretical analysis and experimental results show that DSI outperforms the existing searching algorithms in MANET, OLSR, local index tree (LIT) and pure flooding, in terms of network workload and degree of user satisfaction.

Index Terms**—**Mobile ad hoc network, peer-to-peer networks, virtual backbone, dominating-set-index

I. INTRODUCTION

Mobile ad hoc networks (MANET) [1] have drawn lots of attention in recent years due to potential applications in various areas, such as automated battlefield, search and rescue, and disaster relief. Both peer-to-peer (P2P) network and MANET are self-organized structures. It is reasonable to integrate P2P and the backbone strategy of MANET for resource location and searching in MANET. For any P2P network, resource searching is a basic question [2]. Mobile P2P (MP2P) aims to return effective, correct and sufficient results in a limited time, with limited network bandwidth and involving as many MP2P nodes as possible in one resource search.

Recent research papers based on the P2P MANET resource searching mechanism can be categorized as either reactive, passive searching based on broadcasting or proactive, as follows:

Manuscript received September 18, 2013; revised April 22, 2014.
Corresponding author email: yewang@zjgsu.edu.cn.
doi:10.12720/jcm.9.4.332-342

a. The reactive searching algorithm [3] is a kind of instant traversal algorithm, using a similar method to Gnutella's [11] flooding search in MANET;

b. The passive searching [4] algorithm, based on broadcasting, is a kind of push broadcasting model algorithm. In this kind of algorithm, nodes with resources broadcast their resource fragments to their neighbors periodically but do not check whether any neighbor needs these resources. If the node needs the broadcasted resource, then it is passively accepted. This searching strategy is based on the assumption of space and time validity, which means high usability of resources in a given area.

c. The proactive searching algorithm [5] requires that the nodes have processed their metadata before joining the searching network. This improves the searching efficiency remarkably by organizing the shared metadata properly. Usually the proactive searching strategy uses index technology to form the shared metadata in MANET.

Local Index is a common technique among these algorithms in MANET. In the local index [6] algorithm, each node creates its own index table to provide the shared resource index information in N hops. Based on local index technology, Shi proposed an algorithm called Local Index Tree (LIT) [7]: to get accurate indices, delete the nodes that may disappear at any moment in the index radius and build a tree structure with an index node root by cutting the index table. However, if the network topology changes dramatically, the LIT algorithm will reduce its index radius and become a reactive searching strategy when the index radius is 0. These algorithms neglect to account for the fact that resource location is required to adapt to mobile scenarios in the whole MANET.

To shorten the response time of resource searching in mobile environments, we propose an algorithm, named dominating-set index (DSI). Based on the connected dominating set (CDS) construction algorithm ECARSP (eliminating common adjacency relations with self-pruning) [9], DSI distributes the metadata of shared resources in the virtual backbone MANET. Similar resources are clustered close to the backbone network. The searching process is accelerated by putting resource information close to normal nodes in MANET. In mobile scenarios, DSI provides a novel maintenance algorithm, which involves only 1-hop neighbor nodes for less additional network load. The local dominating set index is re-evaluated when the neighborhood changes. Sufficient simulation is conducted for DSI algorithm performance evaluation. This paper is organized as follows: in Section 2, related work is discussed; Section 3 gives a graph model for ad hoc wireless networks and a CDS construction algorithm; in Section 4, the DSI algorithm is shown in detail; Section 5 focuses on the simulation and result analysis. Finally our conclusion is given in Section 6.

II. Related Work

Napster [10] and Gnutella [11] are two early routing systems which use centralized and decentralized servers respectively. However, they can't be deployed on MP2P since there exists a single point whose failure will lead to a sudden stop of the search operation, jeopardizing the stability of applications. In [12] the authors put forward a locating mechanism in MP2P network, named PDI (Passive Distributed Indexing). The PDI algorithm enables resource-effective searching for files distributed across mobile devices based on simple queries. Building blocks of PDI constitute local broadcast transmission of query and response messages, together with caching of query results at every device participating in PDI. By caching all the replies the peers heard, PDI achieves a considerably better performance. But this approach bothers many participants and the peers have to consume lots of computing power to perform the required tasks.

Klemm [13] proposed the ORION (Optimized Routing Independent Overlay Network) mechanism for searching and file transfer tailored to both the characteristics of MANET and the requirements of P2P file sharing. ORION is completely implemented on the application layer and does not depend on support of a MANET routing protocol. As building blocks, ORION comprises of an algorithm for construction and maintenance of an application-layer overlay network that enables routing of all types of messages required to operate a P2P file sharing system, i.e., queries, responses, and file transmissions.

ORION transfers control and data packets on the best-suited route chosen from a set of redundant routes. Selecting an alternative route provides an efficient mechanism to locally resolve link failures. The ORION transfer protocol enables efficient file transfers on top of the overlay connections established by the search algorithm. However, both PDI and ORION use the flooding mechanism to find files on demand. These solutions are not scalable and curtail throughput as the size of a MANET grows. To address this issue, Ren-Hung Hwang et al. [14] proposed an IPv6-based MANET to support global connectivity and IPv6 mobility. Unique features of this design include mobile hosts form a tree overlay automatically, self-configured logic address of a mobile host is used for IPv6 address configuration and MANET routing, efficient routing without exchanging routing information (on demand or periodically), the tree overlay also helps the development of a P2P file sharing system. However, IPv6-based MANET is not practical for file sharing functionality, since global connectivity is not always available in MANET and power saving is always the main issue for global location. Recently, Huang introduced a P2P architecture called ”WMP2P” in [15], which focuses on how to keep good performance for file retrieval in wireless mobile networks. Because of the movements of mobile nodes, retrieving a file from a fixed resource providing peer is not always a choice in a wireless mobile network. Therefore, WMP2P provided peers a way to discover peers that have better connection quality for file retrieval. However it emphasizes the application layer rather than network routing protocols

In the protocol layer, locality is more important to support file sharing in Mobile P2P environment [2]. The pro-active protocols Optimized Link State Routing (OLSR) [24] needs periodic update with control packet and therefore generates an extra traffic which adds to the actual data traffic. OLSR uses multipoint relay flooding which very significantly reduce the cost of such broadcasts. However, when a node is moving fast and frequently, the link with its neighbors are valid only during a short time internal. If packets are forwarded by an invalid link in OLSR, they are lost. Different from OLSR, the Dominating Pruning (DP) algorithm [16] computes the approximation of MDCS in a distributed manner according to 2-hop neighborhood information. Authors in [17] presented two enhancements of DP aimed at improving the broadcast coverage of an ad-hoc wireless network under conditions of misbehaving nodes. The improvement can be viewed as an enhancement of the network's fault tolerance, as it applies to most scenarios in which a node may fail to carry out its forwarding duties. However, the DP algorithm and its enhancements only concentrate on constructing an efficient routing algorithm by reducing the broadcast redundancy in MANET and do not take into account the MP2P-specific factors. Although Hong [2] provided the domination-set-based peer-to-peer searching algorithm to achieve motivating peers' corporation in the whole overlay of MANET. It focuses on constructing connected-dominating-set based peer-to-peer searching overlay with its economic model. This economic model concentrates on the trading of virtual currency in the message routing process. However, the Hong's design does not use any locality technique and is not suited to mobile scenarios.

On the other hand, the mobility model has become a hot research topic in mobility management recently. In the algorithms mentioned above [12], [14], [16], [17], if many mobile nodes in the network are in movement, the network topology may be greatly affected and thus the complete recalculation of a CDS with a large amount of message exchange is required [18]. Sayaka et al. [23] proposed a self-stabilizing (SS-CDS) distributed approximation algorithm for the minimum CDS in unit disk graphs and the theoretical results prove its good performance in MANET. In this paper, we propose an algorithm named dominating-set-index (DSI), which improves DP to make it more suitable for building an effective search index in MP2P environments and where a single mobile node's movement only affects its neighbors.

III. Network Model

Before details are discussed, definitions are given as follows. The unit disk graph [19] is introduced to represent a MP2P network, where V represents a stable connected dominating set (CDS), G is the set of dominating nodes and C is the set of normal nodes. According to the definition of dominating set,

$$V = D \cup C, D \cap C = \varnothing, \{ f : C \leftrightarrow D \mid dom f = C \} \quad (1)$$

(where $\varnothing$ and f present the empty set and the dominating function respectively. i.e., any node in set C must have at least one corresponding dominator in set D)

And function *Min(S)* yields the node with smallest *id* value in the node set S; function *MaxCov(S)* yields the node in node set S that could cover the most number of ordinary nodes; function $N(p)$ yields the set of all nodes one hop away from node p, function $N^2(p)$ yields the set of all nodes two hops away from node p and function $N(N(p))$ yields the set of all nodes one hop away from at least one node in $(N(N(p)) = \{\forall x \in N(y) \bullet y \in N(p)\})$.

"No-Key dominator": there is only one dominator in all neighborhoods of dominator p, i.e. count($N(p) \cap D$) = 1, or all dominators around p can interconnect without going through p. "Key dominator" is the dominator that does not satisfy the above conditions.

IV. Proposed Algorithm

The DSI algorithm is illustrated in this section. It is a kind of passive index strategy, which means the node will not request to index the metadata of other nodes unless it receives the index responsibility from the sharing nodes. When nodes decide to share some metadata, they will construct the DSI and notify neighbors. All nodes in index set form a CDS and record the ID of the sharing node, shared metadata and the given resource expiration time.

A. Constructing DSI

When a node receives an "ADV" (Advertisement) packet from its neighbor, it will calculate the next dominating nodes according to its 2-hop neighborhood information. When a peer initiates a DSI constructing process, it picks up the dominating peers in its neighborhood to form local dominating set and sends an "ADV" packet to these dominating nodes. The ADV packet contains the peer id, the metadata of its shared resource, the shared resource expiration time, and the local dominating set of that packet. Then the receivers calculate their own local dominating sets and relay the packet. This process continues until the whole network is covered. We assume that u (sender) and v (receiver) are neighbors. The related denotations are (shown in Fig. 1):

- $F(u, v)$: u's local dominating set passed to v;
- $U(u, v) = N(N(v))-N(u)-N(v)$: the exposed node set of node v in $N^2(v)$, which is uncovered by the selected dominating nodes (u is the selector node and v is the candidate forward node);
- $B(u, v) = N(v)-N(u)$
- S_i: the uncovered 1-hop neighbor set of $v_i \in B(u; v)$;
- K: a set including all these uncovered 1-hop neighbor sets;
- Z includes all the covered 1-hop neighbor sets;
- $P(u, v) = U(u, v)-N(N(v) \cap F(u, v))$: the evaluating peers.

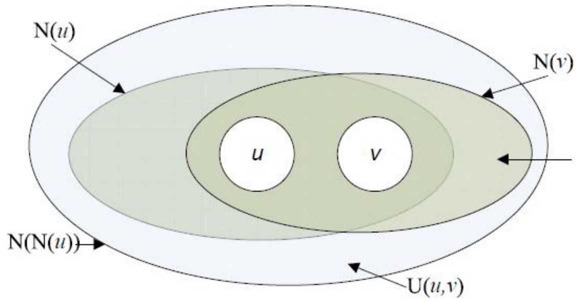


Fig. 1. Some denotations.

The local dominating set of in DSI can be computed in following way:

1) Let $F(u,v) = \varnothing$, $Z = \varnothing$, $K = \cup\{S_i'\}$ where $S_i' = N(v_i) \cap (U(u,v) - N(N(v) \cap F(u,v)))$, $S_i = N(v_i \cap P(u,v))$, for $v_i \in B(u,v)$, $\forall i \in (1, |B(u,v)|)$.
2) If there exists any peer $w \in P(u, v)$ that v can notify only through $v_i \in B(u,v)$, then $F(u,v) = F(u,v) \cup \{v_i\}$, $Z = Z \cup S_i$, $K = K / \{S_i\}$, and $S_j = S_j - S_i$, for all $S_j \in K$, $\forall j \in (1, |K|)$. This step repeats until no peer in P that v can notify only through v_i.
3) Find set S_k with the maximum size in K, max. (If there is a tie, the one with the smallest identification k is selected.)
4) $F(u,v) = F(u,v) \cup \{v_k\}$, $Z = Z \cup S_k$, $K = K / \{S_k\}$ and $S_j = S_j - S_k$ for all $S_j \in K$, $1 \le j \le |K|$.
5) If no new node is added to Z, exit; otherwise, go to step 3.

The constructing process evaluates the number of peers in $U(u, v)$, so called "evaluating peers", which can be covered by v_i in decision of including/excluding v_i in/from $F(u, v)$.

From the constructing process, we can easily figure out that the dominating sets differ from different initiators' view point although their coverage is the same. If different kinds of resources are shared in the network, there will be many initiators which are supported by our algorithm. The details of multiple initiators are explored in Section VI.

The dominating peers for the identical resources maintain cache tables obtained mainly from the ADV header (the header structure is shown in Fig. 2).

<table>
<tr><td>Shared Item</td><td>Initiator</td><td>Contributing Peers</td><td>Expiring Time</td></tr>
<tr><td colspan="2">Previous Dominating Peer</td><td colspan="2">Next Dominating Peers</td></tr>
</table>

Fig. 2. ADV header structure in DSI algorithm.

```
  Node p attempts to share resource R1 and function f(p) for the dominating node of p.
1 Value m denotes whether the resource R1 exists in the network.
  a). if (p is a dominating node) && (p.ADV.contain(R1))
          m←true;
  b). else if (p is a dominated node) && (f(p).ADV.contain(R1))
          m←true;
  c). else
          m←false;
2 if (m)
  a). q←p.ADV.initiator;
  b). q.ADV.constributingPeers.add(p) and broadcast the ADV to other dominating nodes.
3 if (!m)
  p marks itself the initiator of R1 and execute DSI constructing process:
  a). p.ADV.initiator←p;
  b). p.ADV.shareItem←R1;
  c). p.ADV.constributingPeers.add(p);
  d). p.ADV.expireTime←300;
  e). broadcast the ADV to other selected dominating nodes.
```

Fig. 3. Index clustering process.

B. *Index Clustering*

DSI algorithm clusters the metadata of identical resources in the index set, as in Fig. 3.

Here p represents one source peer who shares the identical resource in DSI. When p attempts to advertise its resource, it first performs the searching process mentioned above to see if any participant has already shared the resource. It is the initiator's responsibility to update the dominating index set and include all source peers for the identical resource sharing. Source peers constantly communicate with initiator and inform it of their availability.

When the initiator is invalid, i.e. power-off or leaving the MANET, its neighbor dominating peers will detect it and choose its substitute in the source peer list stored in the DSI. Any static linear attributes (e.g., the peer id or the power left) can be used for the new initiator selection. The selected source peer will be notified and start new DSI constructing process. The dominating indices built by previous initiator are discarded.

C. *Index Maintenance*

In DSI, the initiator peer periodically broadcasts its dominating set to adapt to network changes. The maintenance algorithm only affects the neighborhood of the peer changing status; thus, does not increase the whole system's workload. When network topology changes, the DSI algorithm partially modifies the dominating set. A common CDS network is guaranteed to be formed by the initiator peer whenever any topology change occurs. We summarize topological changes of MP2P networks by categorizing them into three types: Peer Join, Peer Depart, and Peer Move, as follows:

1) Peer join: Suppose node p is joining the system. Then

$$V' = V \cup \{p\}, C' = C \cup \{p\}, T = N(p) \cap D \tag{2}$$

And node *p* becomes an ordinary node in the network (e.g., see (1)). If $T \neq \varnothing$, i.e. there are dominators in *N(p)*, then select dominator *d'* with smallest *id*:

$$d' = Min(T), D' = D \tag{3}$$

If $T = \varnothing$, i.e. there is no dominator in *N(p)*, then select and upgrade the node *d'* with smallest id from *N(p)* to be a dominator, and add it to the dominator set. This selection process is accomplished by *N(p)*'s dominators in $N^2(p)$ (see (4)).

$$T = N(p), d' = Min(T), D' = D \cup \{d'\}, C'' = C' / \{d'\} \tag{4}$$

Finally, renew the dominating relation between dominator *d'* and newly added node *p*.

$$f' = f \cup \{p \mapsto d'\} \tag{5}$$

Especially when many nodes join MANET at the same time, if firstly $T = \varnothing$, the normal node is necessarily promoted to dominating node. Then we need to consider the coverage rate of the new dominating node for the new joining nodes, which means constructing CDS with minimum number of dominating nodes. At that time, *MaxCov(T)* is required to replace *Min(T)*, i.e. *d'* = *MaxCov(T)*.

If node *p* shares resource after join and notifies its dominating node *d'*, then it puts the metadata of shared resources in DSI index and broadcasts in the CDS.

2) Peer depart: Let's suppose the departing peer has no impact on the connectivity of the whole network and only affects the neighbors in the source position. When node *p* exited from the system, the definition can be:

$$V' = V / \{p\} \tag{6}$$

If $p \in C$, then (see (7))

$$C' = C / \{p\}, D' = D, f' = f / \{p \mapsto f(p)\} \tag{7}$$

i.e. if *p* is an ordinary node, then eliminate *p* from the set of ordinary nodes, keep the set of dominators unchanged, and remove the dominating relation with *p*.

Please notice:

If *u* = *f(p)* is a no-key dominator and satisfies $\{\forall v : N(u) \cap C \mid N(v) \cap D \neq \varnothing\}$, i.e. if every ordinary neighborhood around *u* has other dominators, *u* can be degenerated to an ordinary node, then

$$\begin{aligned} & D'' = D' / \{u\}, \\ & T = N(u) \cap C, \\ & C'' = C' / T, \\ & f'' = \{x : C, y : D \mid x \mapsto y \in f' \wedge y \neq u \bullet x \mapsto y\} \end{aligned} \tag{8}$$

For all nodes in T, i.e. nodes originally dominated by u, the operation of entering is performed.

If $p \in D$, i.e. *p* is a dominator, let *u* = *p*, *f'* = *f*, *C'* = *C*, *D'* = *D*, the node is removed after performing (8).

But if p is a key dominator, all nodes in T shall be marked as dominators (after that, leaf dominators shall degenerate to ordinary nodes), i.e.:

$$\begin{aligned} &C''' = C''/T \\ &D''' = D'' \cup T \\ &f''' = \{x : C, y : D \mid x \mapsto y \in f'' \wedge x \notin T \bullet x \mapsto y\} \end{aligned} \tag{9}$$

After being processed by the maintenance algorithm above, CDS shall keep its original attributes. This is because according to supposition, exiting of nodes has no impact on the connectivity of the whole network. When dominator p is removed, network connectivity is kept even if it causes segmentation of the inducement sub-graph in the dominating set.

If node p shared resource and depart, its neighbors detect this change. If node p is a dominating node, the dominating nodes in the neighborhood of p will notify the initiator for the invalid shared resource by node p. Otherwise, p is a dominated node, and its dominating node will notify the initiator node. Finally, the initiator broadcasts the new DSI for the invalid resource from p.

3) Peer move: The departing of non-dominating peers is not taken into account because they do not affect the functionality of dominating set indices. However, the dominating peer's movement can be treated as the composition of peer's join and depart. When the peer is moving, no maintenance is required.

4) Summary: Every dominating peer can start the adapting process as soon as it detects any one of the three topological change patterns mentioned above. It will re-evaluate its evaluating peers and select dominating peers in its neighborhood. This step is iterated in the rest of the network. However, when the dominating peer receives adapting requests, it will select its local dominating set if there is considerable change in its 2-hop neighborhood. Otherwise, the adapting process will cease at this peer. The initiator will broadcast the ADV packet for the DSI update in the maintenance process.

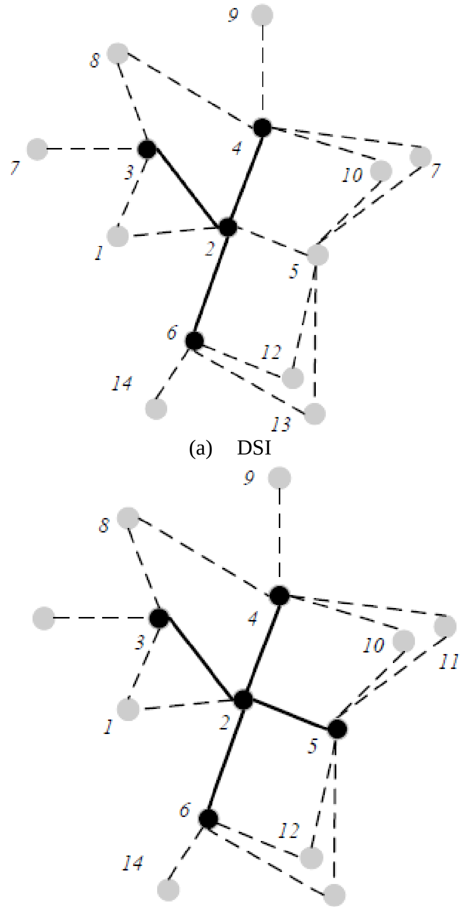
As we can see, the adapting work just involves a small fraction of the whole network. Thus, the cost of DSI index maintenance is low.

D. Case Study

The CDS overlays the entire network so every node can get the results returned in 2-hops. To maximize the returned results, while cutting down the network traffic, we employ a restricted random walk [20] to perform the searching process. Like the algorithm in the constructing process, the querying peer selects key nodes from its neighbors and sends queries to them. Then these key nodes repeat the same process. "Restricted random walk" means queries are forwarded for a limited number of hops.

When the TTL (Time-To-Live) of the query is decreased to 0, the searching process terminates. In this subsection, we will illustrate the DSI mechanism with an example.

1) DSI construction: Fig. 4 shows the construction process of 14 peers by DSI, DP and LIT.



(a) DSI

(b) DP/LIT

Fig. 4. Construction: 14 peers with initiator of peer 1

Assuming that peer 1 initiates a construction process for resource R1, the selected forward list is in Table I. According to the packet received by peer 2 from peer 1, peer 2 knows that peer 3 is also in the forward list. It will be aware that peer 8, that is 2 hops away from it, can be safely excluded from its evaluating set because it knows that peer 3 will tell peer 8 the information of that packet. In DSI, by selecting the key neighbors first, peer 2 will select peer 4 and peer 6 in its forward list, while in the DP (or LIT) algorithm an additional peer of 5 will also be selected because peer 5 is the neighbor of peer 2 with the maximum degree (N(5) = {2, 10, 11, 12, 13}) and will be included in peer 2's forward list at first.

TABLE I: COMPARISON BETWEEN DSI AND DP/LIT

Peer ID	DSI forward list	DP/LIT forward list
1	2, 3	2, 3
2	4, 6	4, 5, 6
Total	4 peers selected	5 peers selected

Using the DSI algorithm, only 4 peers (2, 3, 4 and 6) cover all 14 peers and construct the backbone of current network. The metadata of the resource shared by peer 1 is available on the four peers. If a peer, for example peer 11 here, wants to look up resource R1, it will at first search its cached data. If this step fails, it will continue to consult the key peers which are its neighbors, 4 and 5.

Since peer 4 is in the dominating set indices of resource R1, peer 11 will quickly get the result from it. Because of the mobility of the MP2P network, the backbone of the network will change frequently. Peer 2, 3, 4 and 6 needs to monitor the joining and departing of their neighbors to dynamically adapt to network changes. The expiration time in the advertising packet indicates the time in which the shared resources are still valid. After roaming out of the dominating set, the dominating peers will naturally delete the expired metadata they cached when no more new advertising packets come.

2) Index clustering: If peer 2 and peer 10 require sharing the identical resource R1 in the DSI indices, they both notify the initiator peer 1 for the sharing request. Here peer 10 is dominated by peer 5 which is the contribution peer for the resource sharing of peer 10. Peer 1 broadcast the ADV packet below to peer 2 in the formed CDS (D = {1, 2, 3, 4, 6}), as shown in Fig. 5.

R1	*1*	*2,5*	*300s*
1		*2,3*	

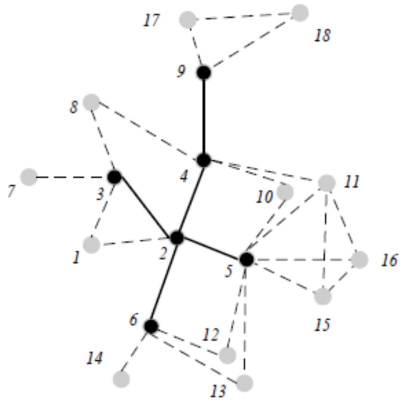
Fig. 5. DSI clustering: ADV for peer 2.

3) Index maintenance: Fig. 6a represents the topological structure and corresponding CDS after adding 4 peers labeled from 15 to 18 according to their entering sequence. It can be seen that with the DSI maintenance algorithm, the CDS network generated still stays relatively succinct. The affiliation relationship between dominated peers and dominators before and after 4 peers are added is listed in Table II.

TABLE II: THE RESULTS OF PEERS JOIN

	Before 4 peers join		After 4 peers join	
	Dominator	Dominated	Dominator	Dominated
	2	1, 5	2	1
	3	7, 8	3	7, 8
	4	9, 10, 11	4	10, 11
	6	12, 13, 14	6	12, 13, 14
			5	15, 16
			9	17, 18
Total	4	10	6	12

Before the addition of new peers, peers 2, 3, 4 and 6 are the dominators which dominate the other nodes. After the 4 peers joined the system, peers 5 and 9 turned out to be the dominators that dominate all the other peers. The addition of peers 15 and 16 upgrades peer 5 to be a dominator while the addition of peer 17 and 18 upgrades peer 9 to be a dominator. After 4 peers join, the DSI is modified as the peer 17 requires to sharing the identical resource R1. The dominating node 9 is the contribution peer for peer 17's sharing, as shown in Fig. 6.b.



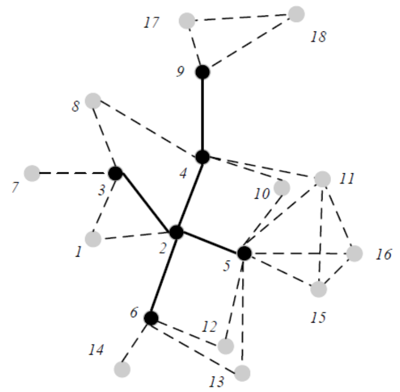
(a) Topology change

R1	*1*	*2,5,9*	*300s*
1		*2,3*	

(b) DSI change for peer 2

Fig. 6. Peers join the network in Fig. 4

Fig. 7a represents the topological structure and corresponding CDS after 7 peers are removed from the 14-peer network in Fig. 4. Because peer 10 and 17 depart, the related contribution peer is removed from the DSI, as shown in Fig. 7b.



(a) Topology change

R1	*1*	*2*	*300s*
1		*2,3*	

(b) DSI change for peer 2

Fig. 7. Peers join the network in Fig. 4.

TABLE III: THE RESULTS OF PEERS DEPART

	Before 7 peers depart		After 7 peers depart	
	Dominator	Dominated	Dominator	Dominated
	2	1, 5	2	1
	3	7, 8	3	7, 8
	4	9, 10, 11	5	12, 13
	6	12, 13, 14		
Total	4	10	3	6

Table III shows the affiliation relationship between dominated peer and dominators, before and after seven peers are removed. Before the exiting of seven peers, peers 2, 3, 4 and 6 are the dominators. After the exiting of peers 4 and 6, peer 5 is upgraded to dominate peers 12 and 13.

E. Algorithm Complexity Analysis

Here n is the number of vertices, *opt* is the size of any optimal Minimum CDS for the given network and ¢is the maximum degree.

1) Optimal approximate factor: Since computing the minimum connected dominating set is an NP-C problem, the approximation factor reflects the approximation degree between the algorithm result and existing optimum solution. Qayyum [25] proved that the size of the CDS produced by the localized heuristic algorithms, such as multipoint relaying in OLSR [24] and dominating pruning (DP) [16], is at mostly log $n * opt$. Based on optimized multipoint relaying, DSI construction algorithm is a heuristic algorithm, so the general approximate factor is log $n * opt$ and the formed CDS size is $O(n)$ in worst case [16]. The size of the solution by SS-CDS [23] is at most $8 * opt + 1$.

Even under worst circumstances, as discussed above, every newly added node introduces one dominator, (the probability for such situation is low). If the number of nodes added is denoted by m, then the approximation factor of the maintenance algorithm is $O(m)$, and this guarantees the efficiency of the algorithm result.

When an ordinary node dominated by dominator *b* is removed, if there are other ordinary nodes that must be covered by node *b*, then the effect of the maintenance algorithm is the same as its construction algorithm; if there is no ordinary node that must be covered by node *b*, and then *b* degrades to an ordinary node. Therefore, the maintenance algorithm keeps the approximation factor of its construction algorithm. The impact on the approximation factor of the exiting of no-key dominator *b* in the dominating set is equivalent to node set $N(b)$ entering into network, and therefore the approximation factor is $O(\Delta)$. The exiting of key dominator *b* in dominating set will also turn $N(b)$ into dominators, so the approximation factor is also $O(\Delta)$.

2) Message complexity: Message complexity represents the number of messages sent when computing CDS in the worst circumstances. Alzoubi *et al*. [26] established the $\Omega(n\log n)$ lower bound on the message complexity for distributed algorithms for leader election, spanning tree and nontrivial CDS in wireless ad hoc networks. In order to implement heuristic DSI algorithm, each two-hop peer has to send out a certain number of HELLO messages to inform its neighbors about itself. After receiving these messages, all one-hop peers have knowledge of their neighbors, and then they also send out HELLO messages to inform their neighbors. The initiator peer will eventually receive all the HELLO messages from its one-hop neighbors and start the DSI construction process. Therefore, the total number of messages sent within two hops in the DSI construction is $O(\Delta+N^2)$.

If the initiator peer is invalid, another contributing peer is selected to be the new initiator. The announcement is required to be sent to all dominating peers. So the message complexity is the CDS size log $n * opt$ and $O(n)$ in worst case.

For any newly added node, if there are dominators nearby, then these dominators will detect newly added nodes and send messages to them, while new nodes will pick only one as its dominator. In the worst circumstances, all neighborhoods of the new node are dominators, and then the message complexity of the maintenance algorithm is $O(\Delta)$. If there is no dominator around the new node, dominators 2-hops away from the new node shall assign a new dominator to cover the new node. As illustrated in Fig. 3, in the worst circumstances all neighborhoods of the new node may be promoted as dominators. Every dominator will send a message to its ordinary nodes, while all neighborhoods of the new node p will also send messages to p. In any case, p will choose one and only one neighborhood as its dominator. In conclusion, the message complexity of the maintenance algorithm is $O(2\Delta)$. The message complexity of the whole maintenance algorithm is $O(2\Delta)$.

For node exiting, if the exiting node is ordinary, in the worst circumstance the exiting of an ordinary node may cause the dominator to change its attributes and broadcast, proclaiming it's degrading to an ordinary node. Ordinary nodes may send an entry request to other dominators, and the message complexity of this algorithm is $O(2\Delta)$. If a dominator is removed, whether the exiting node is the no-key dominator or it is the key dominator, the message complexity of this algorithm is the same as that when neighborhoods are added to the networks, i.e. $O(\Delta\cdot 2\Delta) = O(2\Delta^2)$.

3) Time complexity: In DSI construction phase, the first step can be finished in $O(\Delta)$ time. Step 2 first checks all nodes in B and then traverses all sets K. It needs $O(\Delta^2)$ time for each round. In step 3, DSI traverses K in $O(\Delta)$ time and another $O(\Delta)$ is required for step 4. So the total time for step 3&4 is $O(\Delta^2)$ time for each round in the worst case. Therefore, the overall time complexity of the DSI construction is $O(2\Delta^2)$, referring to the fact that the time complexity of SS-CDS [23] is $O(n^2)$.

For node entering, if there are dominators around the new node, no iterative process is needed for the maintenance algorithm, and the time complexity is O(1). If there is no dominator around, dominators will pick the next hop dominator from neighborhoods to cover the new node, and time consumed is within $O(\Delta)$. Then the time complexity of node entering is $O(\Delta)$.

For node exiting, if the exiting node is ordinary, $O(\Delta^2)$ iterations are needed for its dominator to determine whether there are other dominators near the remaining ordinary's neighborhoods. If a dominator is removed, ordinary nodes surrounding the exiting dominator perform an add operation and, as shown above, the complexity is O(Δ). In conclusion, time complexity of node exiting for maintenance algorithm is $O(\Delta^2)$.

From the analysis above, this algorithm keeps a relatively small approximation factor and time complexity, and thus has satisfying performance.

V. Simulation

A. Environment

In this section, we take simulation tool NS-2 [21] as the experimental platform. The performance of network load and user satisfaction on DSI algorithm will be investigated in the following experiments, comparing with LIT (Local Index Tree) and pure flooding algorithm [22].

In the network setting, every node broadcasts its request every 5 seconds. If the index tree does not change after 4 broadcasts, the index node expands its "index radius". In our simulation, initially 60 nodes are put in the 1000*1000 matrix area. Each node randomly selects 18 files from a file set as share resources. In the simulation process, each node sends a search request every 40-60 seconds. Table IV shows other default simulation parameters.

TABLE IV: Default Simulation Parameter

Parameter	Value
Transmission range	115m
Default number of Peers N-peers	80
Simulation time	1h
Maximum speed S-max	1.5m/sec
Number of shared documents N-docs	18
Simulation area	1000m x 1000m

B. Results & Analysis

1) System response: User response and packet routing delay can be used to represent the performance of the system response. Fig. 8 illustrates the user response time of DSI, OLSR, LIT and pure flooded with different user satisfactions in MANET with 80 nodes. Here user satisfaction is recorded by the quantity of returned results for one single request. We can see that when user satisfactions change, the response time of DSI changes little. On the one hand, DSI uses the clustering method to index the shared resources in one dominating index set and a node only need request finitely many neighbors to get related resources.

On the other hand, as user satisfaction increases, DSI response time increases a little. This is because the maintenance algorithm does not update the index metadata in time and for more returned results, so DSI launches random work to get target resources beyond 1-hop neighbors. Furthermore, the other three algorithms (OLSR, LIT and pure flooded) fail to organize metadata properly. When user satisfaction increases, they need to query most nodes to get more returned results. It is worse for the pure flooded algorithm. The cost time increases more and more to get higher user satisfaction, as more hops are required to forward location request for more returned results.

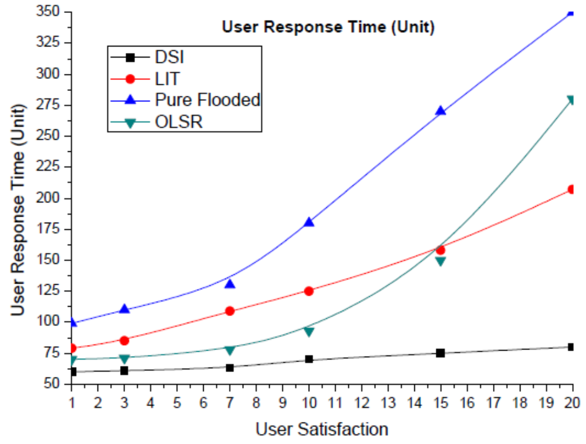


Fig. 8. User response between DSI and other three algorithms in different user satisfaction

Aside from the above statistic network scope simulation, the packet routing delay for different network scope is shown in Fig. 9. Packet routing delay means the time consumed in the transport from source node to destination node. In small networks, the delay difference for the three algorithms is not remarkable. However, as the network expands, the delay increases dramatically for the pure flooded algorithm. Each packet needs more nodes to forward requests in a large network and the path will become longer. DSI uses small CDS to forward requests and this will reduce forwarding redundancy. Thus, DSI can locate the resource more efficiently than OLSR and LIT. When the network expands, the difference between DSI and LIT is more remarkable.

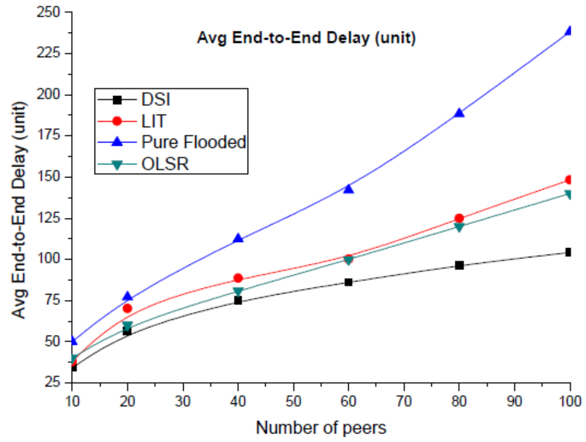


Fig. 9. Packet routing delay between DSI and other three algorithms in different network scope.

2) Network load: Two aspects of network load were simulated: network bandwidth and request routing overhead. Fig. 10 shows the system traffic bandwidth of DSI, OLSR, LIT and pure flooded mechanism with different numbers of participant.

In Fig. 10, we can easily recognize that all of the three methods consume more network bandwidth when increasing peers. The DSI algorithm uses CDS to index data. The cost of bandwidth to forward metadata of shared resources will be reduced; meanwhile, the maintenance algorithm for nodes mobility will shorten the job for the CDS index set to rebuild; and finally DSI only needs to query finitely many neighbors to locate most resources. The above three aspects will make DSI get better network load performance than LIT and pure flooded. LIT requires nodes to request and respond with metadata information frequently and always needs more than 1-hop to forward a search request. It is worse than DSI. Although the pure flooded algorithm will not consume bandwidth when the network id is idle, once a request is launched, all nodes will take part in the job of forwarding the search request. When the network scope expands, the load is obviously increased.

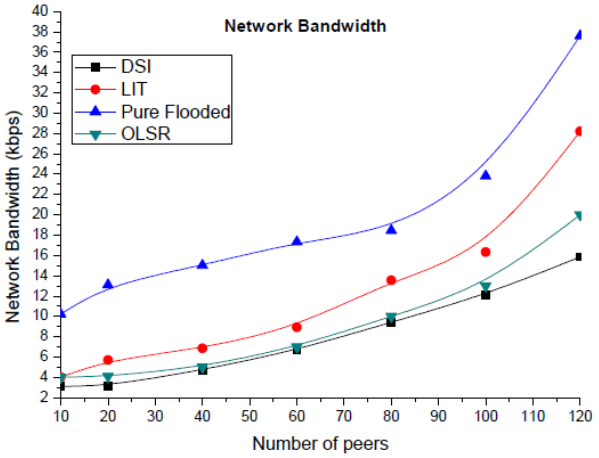


Fig. 10. Network bandwidth between DSI and other three algorithms in different network scope

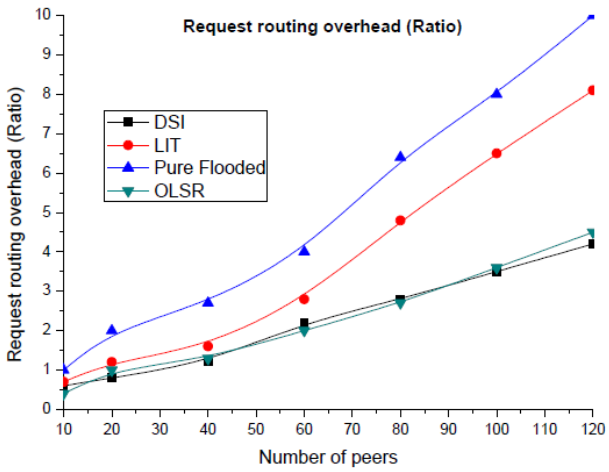


Fig. 11. Request routing overhead (ratio) between DSI and other three algorithms in different network scope.

On the other hand, Fig. 11 illustrates the performance on request routing overhead (all packets overhead/ packets received). The DSI algorithm uses CDS to index data and the relative routing overhead increases in a linear fashion as the network peers grow. The packet overhead of DSI is less than LIT. Although the LIT packet overhead is close to DSI under a small network scope, the ratio of the LIT algorithm increases dramatically when the network expands. The LIT needs more hops to locate the resource than DSI in a large network. The packet overhead of the pure flooded algorithm increases almost exponentially.

Because DSI is based on optimized multipoint relaying technique similar to OLSR, the routing overhead and bandwidth difference between them is small (shown in Fig. 10 and Fig. 11). The advantage of DSI is the bandwidth consumed by search query, as the shared resources' index information is closer than OLSR for all peers.

3) Summary: The metadata of identical resources is clustered in DSI algorithm and the dominating nodes can accelerate the searching process by the DSI data in ADV packet. Our results show that the response time is reduced by DSI algorithm. However, the network load is kept low and no additional cost, by comparison with the classical algorithm OLSR.

VI. Discussion

This paper focuses on the index algorithm for P2P resource sharing in MANET. For our searching algorithm, the following three topics need further discussion:

A. *Multiple Initiators*

The identical resources with the same attribute are shared by one initiator. If two peers share different resources, two initiators will be created. The worst case is that all nodes become initiators. For example, 4 peers in Fig. 12 require sharing files "f1", "f2", "f3" and "f4" (different files) respectively. The 4 peers all become initiators and broadcast the index information. In this case, DSI searching still outperforms pure flooding and OLSR in term of response time, as all peers have all shared resource information. However, the cost of broadcasting messages is huge and its expansibility may not meet the requirement from big networks. This problem exists in other MP2P index algorithm [2], [12], [13] too.

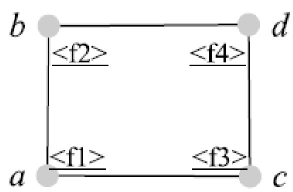


Fig. 12. Sample network with 4 peers.

B. *Multiple Resource Searching*

Multiple resource searching is implemented by different initiators. In our algorithm, the identical shared resources are clustered by one initiator. Resource attribute is used to define the property "identical". The contents of resource attribute depend on the network requirement. For example, file searching requires strict attributes of shared files, such as file name, type, and size and so on. On the other hand, entertainment application only needs loose attribute like classification information of the shared multimedia resource. Currently, DSI support both strict and loose attribute. Strict attribute is used in our simulations for file searching.

Note that different resources may have the same attribute and their location information is store in one initiator's DSI. Therefore, our algorithm does not preclude the use of fuzzy search in dominating peers, such as wildcard- based search.

C. Segmentation of P2P Network with DSI

It is important to prevent from the condition that the P2P network is segmented. In our algorithm, we assume that the connectivity of the whole MP2P network is guaranteed. The only possible segmentation comes from the DSI maintenance, which depends on the dominating peers. A fully segmented P2P network will collapse because not all resources are available and the P2P value is weaken. One case is illustrated in Fig. 13. The 14 peers in Fig. 4 are partitioned into 3 independent sub networks. However, the DSI construction is a multiple relaying broadcasting process and all peers must be covered. The maintenance algorithm guarantees that the dominating peers form a connected dominating set. If the dominating peer 3 loses the DSI information to touch dominating peer 2 and 4, the maintenance algorithm will detect this invalidation according to the neighbors' status, as the connectivity between peers 2, 3 and 4 is guaranteed.

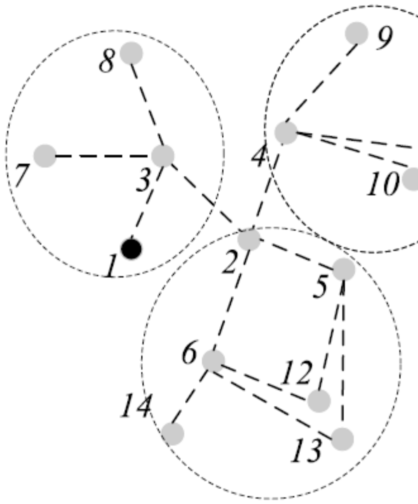


Fig. 13. A possible segmented p2p network with 14 peers in Fig. 4

VII. Conclusion

In integrating the P2P model and the dominating set methodology, this paper proposes a resource searching algorithm DSI, based on the CDS formed by MPR broadcasting. DSI clusters the identical net resource metadata at the dominating peers and makes all available resources as close to every node as possible. It constructs DSI indices and maintains them effectively, so DSI achieves a better network load performance. Simulation results also show that DSI has a big advantage over the three algorithms, both on network load and user satisfaction. As MANET is a typical self-organized structure and its typology changes all the time, it is necessary to dispose of the responsibility of dominating nodes according to typology change and node energy status. This is our task for the future to optimize the DSI algorithm in MANET.

References

[1] IETF. (August 2008). Mobile Ad-hoc Networks. [Online]. Available: http://www.ietf.org/html.charters/manet-charter.html

[2] F. Hong, Y. Feng, M. Li, and Z. Guo, "Constructing incentive oriented overlay on mobile peer-to-peer networks," in *Proc. Int'l Conf. Parallel Processing Workshops*, 2007, pp. 52.

[3] A. K. Leung and Y. Kwok, "Community-based asynchronous wakeup protocol for wireless peer-to-peer file sharing networks," in *Proc. 2nd Annu. Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services*, Boston, 2004, pp. 342-350.

[4] D. Kim, H. Jeong, S. Oh, and J. Cano, "Improving the accuracy of passive duplicate address detection algorithms over MANET on-demand routing protocols," in *Proc. 8th Int'l Symp. Autonomous Decentralized Systems*, Sedona, 2007, pp. 534-542.

[5] S. Hasan, "A peer-to-peer file sharing system for wireless ad-hoc networks," M.S. thesis, Dept. Computer Eng., Bilkent University, Turkey, 2004.

[6] B. Yang, "Resource discovery in peer-to-peer systems," Ph.D. Thesis, Stanford University, USA, 2004.

[7] W. Shi, S. Li, G. Peng, T. Ma, and X. Lin, "Local index tree for mobile peer-to-peer networks," in *Proc. 3rd Int'l Conf. Grid and Cooperative Computing*, Wuhan, 2004, pp. 650-656.

[8] C. Yang, X. Li, and J. Wu, "Dominating-set-based' searching in peer-to-peer networks," *International Journal of High Performance Computing and Networking*, vol. 3, pp. 205-210, 2005.

[9] W. Shi, S. Li, and Z. Yang, "An optimized flooding algorithm based on multipoint relaying for mobile ad hoc networks," *Journal of Computer Research and Development*, vol. 44, no. 6, pp. 924-931, 2007.

[10] Source Forge. (November 2001). Napster Protocol Specification. [Online]. Available: http://opennap.sourceforge.net/

[11] Gnutella. (October 2008). A Protocol for a Revolution. [Online]. Available: http://rfcgnutella.sourceforge.net

[12] C. Lindemann and O. P. Waldhorst, "A distributed search service for peer-to-peer file sharing in mobile applications," in *Proc. Second international Conference on Peer-To-Peer Computing (P2P'02)*, 2002, pp. 73.

[13] A. Klemm, C. Lindemann, and O. P. Waldhorst, "A special-purpose peer-to-peer file sharing system for mobile ad hoc networks," in *Proc. Int'l Workshop Mobile Ad Hoc Networking and Computing*, New York, 2003, pp. 2758-2763.

[14] R. Hwang, C. Li, C. Wang, and Y. Chen, "Mobile ipv6-based ad hoc networks: Its development and application," *IEEE Journal on Selected Area in Communications*, vol. 23, no. 11, pp. 2161-2171, 2005.

[15] C. Huang, T. Hsu, and M. Hsu, "A file discovery control scheme for P2P file sharing applications in wireless mobile environments," in *Proc. 28th Int'l Australasian Conf. Computer Science*, Newcastle, 2005, pp. 39-47.

[16] F. Dai and J. Wu. "Distributed dominant pruning in ad hoc networks," in *Proc. Int'l IEEE Conf. Communications*, New York, 2003, pp. 353-357.

[17] A. Rahman, P. Gburzynski, and B. Kaminska, "Enhanced dominant pruning-based broadcasting in untrusted ad-hoc wireless networks," in *Proc. IEEE Int'l Conf. Communications*, New York, 2007, pp. 3389-3394.

[18] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1, pp. 32-48, 2005.

[19] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Discrete Math*, vol. 86, pp. 165-177, 1990.

[20] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," in *Proc. 23rd Annu. Joint Conf. IEEE Computer and Communications Societies*, 2004, pp. 7-11.

[21] ISI. (September 2008). The Network Simulator--NS2. Available: http://www.isi.edu/nsnam/ns/

[22] H. Lim and C. Kim, "Flooding in wireless ad hoc networks," *Computer Communications Journal*, vol. 24, no. 3-4, pp. 353-363, 2001.

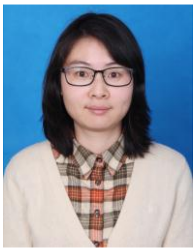
[23] S. Kamei and H. Kakugawa, "A self-stabilizing distributed approximation algorithm for the minimum connected dominating set," in *Proc. Parallel and Distributed Processing Symp.*, Luxor, 2007, pp. 1-8.

[24] T. Clausen and P. Jacquet. (October 2003). Optimized Link State Routing Protocol (OLSR). [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt

[25] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," in *Proc. 35th Annu. Hawaii Int'l Conf. System Sciences*, Hawaii, 2002, pp. 298.

[26] K. M. Alzoubi, P. J. Wan, and O. Frieder, "New distributed algorithm for connected dominating set in wireless ad hoc networks," in *Proc. 35th Annu. Hawaii Int'l Conf. System Sciences*, Hawaii, 2002, pp. 3849-3855.

Cheng Chang received the B.S. degree in Software Engineering from Zhejiang University in 2007. He is a research assistant in the College of Computer Science, Zhejiang University. His research interests include wireless communication and ad-hoc network.



Ye Wang received the Ph.D. degree in Computer Science from Zhejiang University in 2013. She is a lecturer in the School of Computer and Information Engineering, Zhejiang Gongshang University. Her research interests include software engineering and distributed computing.

Jie Sun received the B.S. degree in Software Engineering from Zhejiang University in 2008 and the Ph.D. degree in Computer Science from Zhejiang University in 2014. She is a lecturer in Ningbo Institute of Technology, Zhejiang University. Her research interests include software engineering, requirements engineering and service computing.