TSearch: Target-Oriented Low-Delay Node Searching in DTNs with Social Network Properties

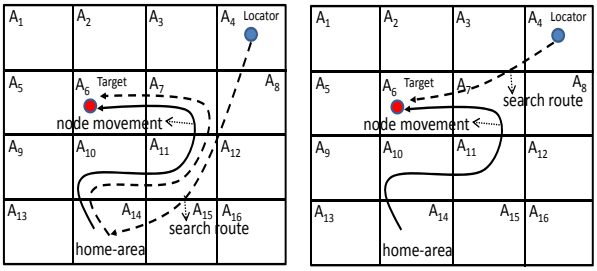
Li Yan, Haiying Shen and Kang Chen
Department of Electrical and Computer Engineering
Clemson University, Clemson, SC 29631
Email: {lyan, shenh, kangc}@clemson.edu

***Abstract*—Node searching in delay tolerant networks (DTNs) is of great importance for different applications, in which a locator node finds a target node in person. In the previous distributed node searching method, a locator traces the target along its movement path from its most frequently visited location. For this purpose, nodes leave traces during their movements and also store their long-term movement patterns in their frequently visited locations (i.e., preferred locations). However, such tracing leads to a long delay and high overhead on the locator by long-distance moving. Our trace data study confirms these problems and provides foundation of our design of a new node searching method, called target-oriented method (TSearch). By leveraging social network properties, TSearch aims to enable a locator to directly move towards the target. Nodes create encounter records (ERs) indicating the locations and times of their encounters and make the ERs easily accessible by locators through message exchanges or a hierarchical structure. In node searching, a locator follows the target's latest ER, the latest ERs of its friends (i.e., frequently meeting nodes), and its preferred locations in order. Extensive trace-driven and real-world experiments show that TSearch achieves significantly higher success rate and lower delay in node searching compared with previous methods.**

I. Introduction

In recent few years, Delay Tolerant Networks (DTNs) attract significant attention from researchers. In such sparsely distributed networks, node searching, in which a *locator* node finds a *target* node in person, is of great value in node management and many applications. For example, in a DTN formed by mobile device holders in a hospital, a campus, a disaster area or a national park, a user needs to find another user in person. In a DTN in battlefield, a node needs to find a node that carries a malfunctioning device in order to fix it. In a DTN formed by vehicles [1], a vehicle may need to find another vehicle to directly communicate with it. The DTN network condition without infrastructures or continuous network connectivity poses a challenge for designing an efficient distributed node searching algorithm.

Some previous object tracking systems [2]–[5] in wireless networks provide high localization accuracy or search efficiency based on the geographical information provided by central base stations or other infrastructures. However, the extra infrastructure requirement is costly and impractical for DTNs (e.g., in battlefields). DTN routing algorithms [6]–[12] can be indirectly used for node searching. In routing, a node forwards the message to the node with a higher probability of meeting the destination. Then, to find a target, a locator can move with the selected message carriers by regarding the target as the message destination. However, since the locator must follow multiple nodes in routing and each node has its own movement path rather than moving directly towards the target, such a node searching method generates a high delay and overhead on the locator by long-distance moving.



(a) Node searching in DSearching. (b) Node searching in TSearch.

Fig. 1: Node searching in DSearching and TSearch.

Recently, a distributed node searching algorithm (called DSearching) has been proposed [13]. It divides the entire DTN area to sub-areas. During a node's movement, it tells several nodes in its current sub-area its next sub-area (called transient visiting record (VR)) before moving out. Each node also deduces its long-term mobility pattern (MP), which indicates the sub-areas it has high probabilities to move to from each of its frequently visited sub-area (i.e., preferred location). It distributes its MP from sub-area A_i to long-staying nodes in A_i. A node's home-area is the sub-area it has the highest staying probability, and this information is stored in all sub-areas. As shown in Figure 1(a), a locator starts from the target's home-area and follows the VRs. When these records are absent in searching, the locator moves to the next sub-area with the highest probability based on the MP. If this information is not available, the locator searches nearby sub-areas for VR and MP.

However, both moving to the target's home-area and tracing along the target's movement path may take a long time. First, as Figures 1(a) and 1(b) show, this tracing process ($A_4 \rightarrow A_{14} \rightarrow A_{10} \rightarrow A_{11} \rightarrow A_7 \rightarrow A_6$) generates high delay. A locator can take a shortcut to directly move towards the target ($A_4 \rightarrow A_7 \rightarrow A_6$). Second, when a locator in a sub-area (say A_{11} in Figure 1(a)) loses trace (i.e., VR), it moves to the predicted next sub-area from A_{11} (i.e., A_7). However, directly moving to the sub-area that the target frequently visits (i.e., $A_{11} \rightarrow A_6$) generates shorter searching delay and

overhead. Also, in this step, the next sub-area with the highest probability may not be the one that the target actually moves to, which leads to high searching delay and even searching failure. Further, storing the target's MP in a very limited number of sub-areas may make it not easily accessible to the locators, which may also increase searching delay.

In this paper, we have conducted trace data [14], [15] study, which confirms the above problems of DSearching and also lays a foundation of our proposed target-oriented method (called TSearch). As DSearching, TSearch is also designed for DTNs with social network properties such as mobility range stability, certain mobility patterns and certain frequently meeting nodes (i.e., friends), and skewed visiting places (i.e., preferred locations) shown in previous works [7], [16], [17]. By leveraging these social network properties, TSearch aims to enable a locator to directly move towards the target.

In TSearch, nodes record and disseminate encounter records (ERs) that indicate the locations and times of their encounters. A locator (N_i) always directly moves to the sub-area in the latest ER of the target (N_j) known by itself (Figure 2). In the absence of N_j's newer ER, N_i relies on the ERs of N_j's friends. In the absence of the friends' ERs, N_i directly moves to the nearest preferred location of N_j, and also requests the nodes sharing the common preferred locations with N_j to search N_j simultaneously. This design is based on our trace study, which shows that this strategy leads to a higher success rate than targeting N_j's most frequently visited preferred location (as in DSearching). To make the information for node searching globally accessible, TSearch adopts the hierarchical structure from [18], [19], in which each sub-area has a long-staying node (called anchor) to collect the information from nodes, and nodes that frequently transit between two sub-areas (called ambassadors) are responsible for the information updates between anchors. TSearch provides an option for nodes to piggyback ERs on the information exchanged between neighbors in order to expedite the information dissemination. In summary, our contributions are threefold:

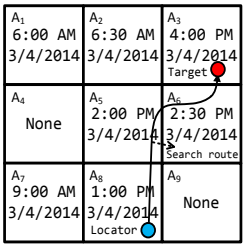


Fig. 2: ER-based node search in TSearch.

(1) Our extensive study on two real traces [14], [15] confirms the drawbacks of DSearching and lays the foundation of the strategy design in TSearch.

(2) We propose TSearch, which is the first work (to our best knowledge) that aims to enable locators directly move towards the targets with easily accessible information to reduce node searching delay by utilizing social network properties.

(3) We have conducted both trace-driven and real-world experiments, which verifies the efficiency and effectiveness of TSearch compared with other previous methods.

The remainder of this paper is organized as follows. Section II presents an overview of related work. Section III presents our trace analysis results. Section IV presents the detailed design of TSearch. Section V presents the experimental results of TSearch. Section VI concludes this paper with remarks on our future work.

II. Related Work

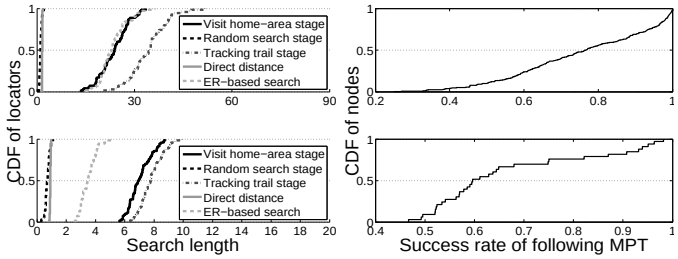
Object searching in disconnected mobile networks has been paid much attention in research. Juang *et al.* [2] proposed a method that sends the positions of animals to the central station through hop-by-hop broadcasting by configuring tracking collars on animals. By utilizing the flock behavior of sheep in wild areas, Thorstensen *et al.* [3] proposed a system that lets the flock leader monitor and report the positions of other sheep to the server through GPRS [20] or satellite communication. Cenwits [4] and SenSearch [5] provide object searching services in wilderness areas. They utilize the opportunistic encounters among nodes to forward location information to infrastructures. However, these methods need extra infrastructures or central servers, which is not practical for DTNs. DSearching [13] was proposed specifically for node searching in DTNs. As indicated previously, it provides insufficiently efficient node search by aiming to enable a locator to trace the target along its movement path from its home-area. Routing algorithms [6]–[12] can be indirectly applied for node searching, but the hop-by-hop routing is not efficient for node searching in DTNs. Unlike these previous methods, TSearch does not need an infrastructure or a central server. It is the first work that enables locators to directly move towards targets to achieve low search delay and overhead.

III. Rationale of TSearch Design

In this section, we present the rationale of the design of TSearch based on trace analysis. We used the DART trace [14] (DART) and the DieselNet AP trace (DNET) [15]. DART is a 119-day record for wireless devices carried by students on Dartmouth College campus. DNET is a 20-day record for WiFi nodes attached to the buses in the downtown area of UMass college town. We filtered out nodes with few occurrences and merged access points (APs) within short ranges to one sub-area. Finally, DART has 320 nodes and 159 sub-areas and DNET has 34 buses and 18 sub-areas.

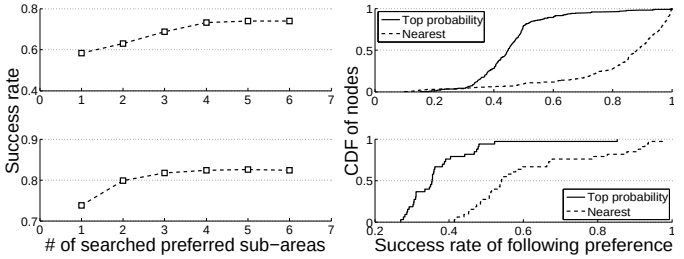
We set the initial period to 30 days for DART and 2.5 days for DNET, during which nodes collect information for node searching. We randomly selected 70 locators and each locator randomly chose a target to search periodically for 90 times and the average experimental result of each locator is reported. The periodical time was set to 1 day in DART and 4 hours in DNET. The search TTL (Time-To-Live) was set to 24 hours in DART and 4 hours in DNET. Node searches using more than TTL are considered as unsuccessful searches. In each of the following figures, the top figure is for DART and the bottom figure is for DNET.

1) Leveraging Encounter Records (ERs): We define *search length* as the number of sub-areas the locator transited in searching. DSearching has three stages: i) a locator moves to the target's home-area, ii) tracks along its moving trail, iii) and may randomly search in neighbor areas. As shown in Figure 1, such searching may generate a long search



(a) The number of searched sub-areas. (b) Choosing the top next sub-area.

Fig. 3: Drawbacks of DSearching.



(a) Searching top preferred locations. (b) Preferred locations to search.

Fig. 4: Node searching based on preferred locations.

length. To confirm this drawback, we measured the cumulative distribution function (CDF) of the search lengths of these three searching stages as shown in Figure 3(a). It also includes the locator-target initial direct distance. We see that the direct distance is very short (within 3 sub-areas in DART and 2 sub-areas in DNET). However, 50% of locators need to travel more than 24 and 6 sub-areas to reach the target's home-area, and also travel more than 35 and 8 sub-areas in the tracking stage in DART and DNET, respectively. These results demonstrate that locators must travel many sub-areas in the first two stages in TSearching. Therefore, we aim to design a method that avoids the unnecessary travel and enables a locator to directly move to current location of the target. For this purpose, we propose the concept of encounter record (ER), which records the location and time of a node. The ERs of nodes are disseminated among nodes for locators to access, so they can move directly to the most recent locations of the targets. We measured the search length of this method as shown in Figure 3(a) when we temporarily let nodes piggyback ERs on the messages exchanged between neighbors for dissemination. The result shows that ERs are effective in enhancing the node searching speed of DSearching.

2) Leveraging Preferred Locations: In DSearching, by referring the target's MP Table (MPT), the locator always moves to the target's next sub-area with the highest probability. To verify the effectiveness of this method, for each node, we used this method to search the node's next sub-area from its previous sub-area in its entire movement path, and calculated the success rate. Figure 3(b) shows the CDF of the success rate. We see that 20% of the nodes have success rate less than 60% and 50% of the nodes have success rate less than 75% in DART, while 25% of the nodes have success rate less than 55% and 75% of the nodes have success rate less than 70% in DNET. Therefore, a locator should not ignore the other preferred locations that a target has a high probability (though not the highest probability) to move to. When a target stays in a sub-area most recently, it may be on the way to a nearby preferred location. Then, searching the nearest preferred location may lead to a higher success rate. To verify these, we draw Figures 4(a) and 4(b). A node's *preferred locations* are defined as the sub-areas the node frequently visits. We ranked each node's visited sub-areas based on the visiting frequency and consider the top sub-areas that constitute 60% of visiting frequency as its preferred locations.

Figure 4(a) shows the average success rate of searching different numbers of preferred locations. We see that searching the top preferred location only leads to 59% and 74% success rates in DART and DNET, respectively. Searching top 4 (in DART) and 3 (in DNET) preferred locations can achieve 73% and 82% success rate, respectively, and then searching additional 1 or 2 preferred locations only generates a very marginal improvement. Figure 4(b) shows the success rates of searching the top and nearest preferred location, respectively. From this figure and Figure 3(b), we can see that selecting the nearest preferred location is more accurate than selecting the top preferred location.

3) Leveraging Frequently Met Nodes (i.e., Friends): We define that node N_i and node N_j are friends if their encounter frequency is higher than a threshold. Since each node has certain frequently meeting nodes (i.e., friends) [7], [16], [17], if a locator moves towards the target's friend, it should have a high probability to meet the target. To verify this conjecture, we draw Figure 5 that shows the CDF of success rate of following the ERs of the target and the target's friends, respectively. We regard a node's friends as the nodes that take up at least a high percentage (60%) of all contacts with the node. We see that following the targets' ERs, about 60% of the locators have success rate higher than 92% in DART and 91% in DNET. Following the ERs of the target's friends, about 60% of the locators have success rate higher than 70% in DART and 80% in DNET. The result confirms that the ERs of the target's friends can be used for node searching as a complementary method.

4) Search Range Constraint: Based on the normal node velocity V and the time and location in the latest ER of a node, the range of the area that the node possibly stays (called coverage area) can be determined. It is a circle with VT as the radius and the ER location as the center, where T is the elapsed time since the time in the ER. For each node, at each of its locations, we checked whether it is within its coverage area based on its previous location. Figure 6 shows the CDF of the *coverage ratio* defined as the ratio of the number of locations in the coverage areas. We see that 80% of nodes have coverage ratio higher than 70% in DART and DNET. The result shows that the coverage area can be used to limit the searching areas of the locators to reduce search delay and overhead.

5) Information Dissemination: Nodes may move locally in only a few sub-areas [7], [16], [17], so a locator may not receive ERs of very distant targets. To make ERs globally accessible, we adopt a hierarchical structure in previous works [18], [19], which verified the existence of long-staying

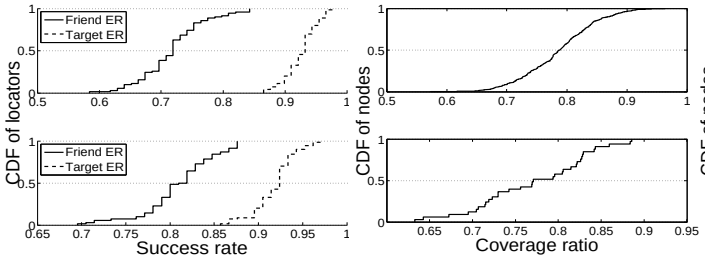
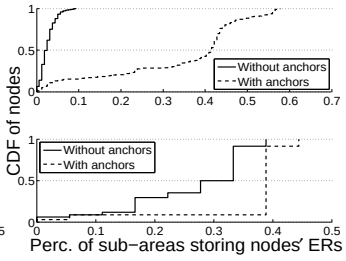
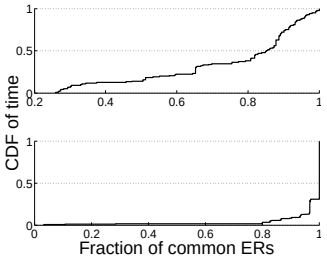


Fig. 5: Following the ERs of the target or its friends.

Fig. 6: Constraining searching range.



(a) Effectiveness of anchors.



(b) Effectiveness of ambassadors.

Fig. 7: Role-based information Dissemination.

nodes (i.e., anchors) in each sub-area and nodes that frequently transit between two sub-areas (i.e., ambassadors).In our method, nodes report information to anchors and ambassadors are responsible for the record updates between anchors.

In order to see the effectiveness of this method, we measured the percent of sub-areas that have the ERs of a certain percent of nodes at the time point of 120 hours and 20 hours in DART and DNET, respectively, We see that 50% of nodes have their ERs disseminated to less than 2% and less than 40% of all the sub-areas without and with the anchors in DART, and to less than 27% and less than 39% of all the sub-areas without and with the anchors in DNET. The result confirms the importance of using anchors for easy information access.

In order to see the degree of consistency relying on the ambassadors, in each hour during the 120 hours and 20 hours in DART and DNET, respectively, we measured the ratio of common ERs among all anchors. The results are shown in Figure 7(b). We see that about 80% of the time, the ratio of common ERs among anchors is higher than 40% in DART, and higher than 90% in DNET. It confirms that the ambassadors can help maintain a high degree of consistency among anchors.

IV. The Design of TSearch

Using the same method in DSearching, we partition the whole network area into several sub-areas (denoted by A_i) to represent node positions (Figure 1). Each sub-area means a popular place [8], [21]–[23] that nodes usually have "gathering" preference. TSearch is designed for DTNs with the social properties [7], [16], [17] (mentioned in Section I) and it leverages these properties for efficient node searching.

- Mobility range stability means that the mobility range of each user is significantly smaller than the whole area, and the change of its mobility range is small over time [16], [18]. Therefore, ERs can be used to search targets. Even though a target is no longer in an ER's location, it is very likely to stay nearby (Figure 3(a)).
- Following the ERs of a target's friends (i.e., frequently meeting nodes) can be used as a complementary approach (Figure 5).
- As each node has preferred locations, moving towards a target's preferred location has a high probability of finding it on the way or at the destination (Figure 4).
- The mobility pattern feature indicates that some nodes are relatively stable while some nodes transit between certain sub-areas frequently. As previous work in [18], [19], we assign different roles (i.e., anchors, ambassadors) to nodes with certain mobility features for information dissemination methods (Figure 7).

Accordingly, TSearch has three types of location information of the target: ERs, friends' ERs and preferred locations. We first introduce the location information in Section IV-A, then present the node searching algorithm in Section IV-B, and finally explain the information dissemination in Section IV-C.

A. Information for Node Searching

A node generates an ER for each of its neighbors (i.e., the nodes in its transmission range) upon encountering. Node N_i generates ER for neighbor N_j in the form of $< N_i, N_j, L_{ij}, T_{ij} >$, where L_{ij} and T_{ij} denote the current sub-area and current time. If N_i already has N_j's ER, it only needs to update the L_{ij} and T_{ij} in the existing ER. Finally, each node maintains its ER table based on its encounters with other nodes as shown in Table I. To constrain the storage overhead for ERs and ensure their validity in guiding node searching, TSearch sets a TTL for ERs. Each node deletes ERs after TTL upon their creation. Due to the mobility range stability, the number of nodes that node N_i encounters is limited [7], [16], [17], which means that N_i's ERs are created in a limited number of nodes. In Section IV-C, we will introduce methods to enable a locator of N_i in the network to access its ERs.

TABLE I: An encounter record (ER) table.

ER creator	Node ID	Sub-area	Time
N_2	N_3	A_1	1:00pm, 3/4/2014
N_1	N_7	A_2	2:00pm, 3/4/2014
...	...	...	...

After a node joins in the system, it accumulates enough records during its movement and calculates its friends and preferred locations as shown in Table II. Because each node has a skewed visiting preference and relatively stable friends, these types of information do not update frequently. Through accessing this table, the locator knows that its target node N_1 has the probability of 0.95 to appear in A_3, probability of 0.8 to appear in A_4 and probability of 0.75 to appear in A_5.

TABLE II: Friends and preferred locations of N_1.

Node	Friends	Meeting prob.	Preferred locations	Visiting prob.
N_1	N_3	0.9	A_3	0.95
	N_4	0.8	A_4	0.8
	N_4	0.7	A_5	0.75

B. Target-Oriented Node Searching

The priority to use the three types of information is ordered by ERs, friends' ERs and preferred locations. The information

is collected and disseminated through the anchors. A DTN can also choose to piggyback ERs on packets exchanged between neighbors to expedite the information dissemination if it can afford this additional transmission overhead. We will introduce the details for the information dissemination in Section IV-C. In TSearch, if a locator does not have the higher-priority information, it uses the lower-priority information. Specifically, when a locator searches a target, if it has the ER of the target, it moves towards the location in the ER. During the movement, if the locator receives a newer ER (with a more recent time), it moves towards the new location in the ER. In the absence of an ER in searching, the locator finds the ER of the target's friend with the highest meeting probability with the target, and then moves towards the location in this ER. If the ERs of the target's friends are not available, the locator moves towards the nearest preferred location of the target. In order not to miss other frequently visited places of the target, we propose an agent-based simultaneous searching scheme, in which the locator requests a certain number of nodes sharing these preferred locations to search the target simultaneously. In the absence of all the information, the locator randomly searches nearby sub-areas. In the following, we present the details of each step of the node searching process.

1) Node Searching Based On ERs: If a locator has or can access the ER of its target, it directly moves to the location in the ER, say A_i. The ER may not provide the current location of the target and the target may move to a sub-area near A_i. Therefore, during searching, if the locator receives a newer ER which represents a more recent appearance place of the target, it moves to this new place.

2) Node Searching Based On Friends' ERs and Preferred Locations: It is possible that in the disconnected DTN with sparsely distributed nodes, the most recent ERs of the target are not transmitted to the locator or its local anchor in time. In the absence of the target's ER initially or when the locator arrives at the moving destination but cannot find the target, the locator queries the target's friends and their ERs, and the preferred locations of the target from its local anchor. The locator finds the ER of the friend that has the highest meeting probability with the target and moves towards the location in this ER. As the friend has a high probability of meeting the target, the locator has a high probability of finding the target.

In the case that no newer ER of the target or no ERs of the target's friends can be found, the locator can use the target's visiting preference for node searching. Based on our observations in Section III, the locator itself moves to the nearest preferred location of the target, and relies on M number of nodes (as agents) to search the target in top M preferred locations of the target. M is an empirical parameter determined by the node mobility, the network size and etc. For example, as shown in Figure 4(a), $M = 4$ for DART and $M = 3$ for DNET. The selected agents must have high probabilities of meeting the target and of moving to the M preferred locations. These agents then should be the nodes that have these common preferred locations with the target. The locator queries the local anchor for such nodes in the current sub-area. For each of the M top preferred location A_k, the locator queries the nodes that have A_k as their preferred locations about the time they will move to A_k, and then chooses the node with the earliest time to search the target. If an agent finds the target, it uses a routing algorithm [6]–[12] to send a notification message with the latest ER to the locator. Then, the locator moves to the new destination in the ER.

3) Node Searching in Coverage Area: Section III finds the coverage area of a target where the target possibly stays currently. In the absence of all types of information for node searching, the locator then searches the target's coverage area rather than randomly searching the nearby sub-areas as in DSearching. If the locator moves around itself in searching, it generates high overhead on the locator. To handle this problem, the locator then uses the agent-based simultaneous searching scheme to search the coverage area.

C. Role-based Information Collection and Dissemination

Based on the hierarchical structure in previous works [18], [19], we design a role-based information collection and dissemination scheme to enable the information for node searching to be globally accessed.

1) Role-based Scheme: The role-based scheme selects a relatively stable node with high storage and computing capacity in each sub-area to be "anchor", and selects a number of nodes frequently transiting between two sub-areas as their "ambassadors". Anchors are responsible for collecting the ERs, friends and preferred locations of nodes in different sub-areas. When a node moves into a sub-area, it reports its stored ERs, friends and preferred locations to the sub-area's anchor. An anchor only stores the latest ER of each node. Therefore, once a locator moves into a sub-area, it can quickly access the information of its target from the sub-area's anchor.

An ambassador for sub-areas A_i and A_j are responsible for maintaining the consistency of stored information in the anchors of A_i and A_j. When the ambassador moves from A_i to A_j, it carries the updated and new information (since the last update) in the anchor of A_i to the anchor of A_j. The anchor of A_j then adds the information not in its own storage, and updates the latest ERs. The same applies when the ambassador moves from A_j to A_i. Once a new encounter event happens in a sub-area, the ambassador will carry the new ER to other sub-areas. Thus, a locator can access the information of nodes in remote sub-areas from the local anchor for node searching. In the absence of the target's ER, the locator can use the preferred locations, and the ERs of the target's friends for node searching.

In a DTN, nodes always need to exchange packets with neighbors to identify their neighbors. For a DTN that can afford the overhead of transmitting a few more packets in the packet exchanges, nodes can piggyback ERs on the exchanged packets to expedite the information dissemination. Then, a locator can quickly receive the ERs of nodes in nearby sub-areas.

2) Role-based Node Selection: We next introduce how to select the anchors and ambassadors. We use the nodes' probability of staying in a certain sub-area to determine

whether they can be the anchors of this sub-area. The staying probability of a node, say N_i, at sub-area A_k is defined as $P_{N_i}(A_k) = T_i/T_u$, where T_i is the total time that N_i has stayed in sub-area A_k during a unit time period T_u. If $P_{N_i}(A_k)$ is larger than a high threshold, N_i can be the anchor for A_k. By exchanging messages, the node with the highest staying probability becomes the anchor of a sub-area, and all other qualified nodes become anchor backups. Before the current anchor moves out of sub-area (A_k), it chooses the anchor backup with the highest $P_{N_i}(A_k)$ as the new anchor, transfers all of its information to the new anchor and notifies the nodes in the sub-area about this new anchor.

The ambassadors for two sub-areas, say A_i and A_j, are the nodes that have high frequency of transiting between A_i and A_j. A node records the number of transits between two sub-areas during time period T_u. If this transit probability is larger than a threshold, this node can be an ambassador between these two sub-areas. Then, it reports to the anchors of the two sub-areas, which choose a number of nodes that have the highest transiting frequencies as the ambassadors.

V. Performance Evaluation

We conducted trace-driven experiments based on the DART [14] and DNET [15] traces as introduced in Section III. Unless otherwise specified, the experiment setting is the same as that in Section III. Search rate is defined as the number of locators generated every 24 hours in DART and every 4 hours in DNET and it was set to 40 by default. Since both traces do not provide map information, we assume that the locator needs 10 minutes to move from one sub-area to another neighbor sub-area on average. The expiration TTL for ERs was set to 4 hours and 2 hours in DART and DNET, respectively. The staying probability threshold for determining anchors and the transit probability threshold for determining ambassadors were set to 0.8.

We evaluated TSearch with and without the ER exchanges between nodes, denoted by *TS** and *TS*, in comparison with two representative algorithms: the *DSearching* distributed node searching method (*DS* in short) [13], and a routing based method (denoted by *Routing*) [24] as explained in Section I. In order to show the effect of ERs in node searching in TSearch, we also evaluated TSearch that only uses ERs without anchors (denoted by *ER*). That is, nodes record the ERs with their encountering nodes and exchange the records. We measured the following metrics in the experiments.

- *Success rate*: The percentage of locators that successfully find their target nodes within searching TTL.
- *Average delay*: The average time (in seconds) used by locators to search for the target nodes. Note that the time spent by unsuccessful locators, which is the searching TTL, is also considered in calculating this metric.
- *Average transmission overhead*: The average number of all packets transmitted among nodes.
- *Average node memory usage*: The average number of memory units used by each node. Each piece of location information (i.e., VR, MPT entry, ER, friend, preferred location) takes one memory unit.

A. Experiments with Different Search Rates and Search TTLs

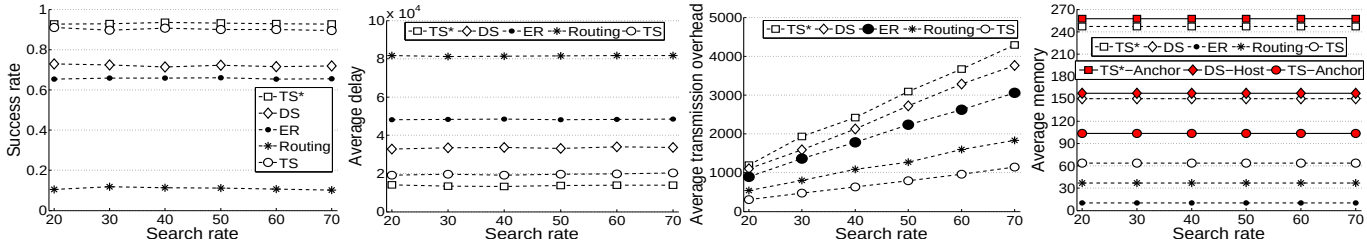
We conducted two experiments. In one experiment, we varied the search rate from 20 to 70 with 10 as the step size. In the other experiment, we varied the search TTL from 18 hours to 24 hours in DART and from 2 hours to 7 hours in DNET.

1) Success Rate: Figure 8(a) and Figure 9(a) show the success rates of the algorithms under different search rates in DART and DNET, respectively. Figure 10(a) and Figure 11(a) show the success rates under different search TTLs in DART and DNET, respectively. In these figures, we find that in DART, the success rates follow: *TS**$\succ$*TS*>*DS*>*ER*$\gg$*Routing*, while in DNET, the success rates follow: *TS**$\succ$*TS*$\succ$*ER*>*DS*$\gg$*Routing*, where $\succ$ means "slightly higher" and $\gg$ means "significantly higher".

In both traces, *Routing* always produces the lowest success rate. This is because the locators do not move proactively to search the targets, but only adhere to nodes that have the highest probabilities of meeting the targets. Since the nodes in the system have independent mobility patterns, many locators fail to find their target nodes within TTL. Compared with *Routing*, *DS* and *TS* have remarkably higher success rate. *TS* has higher success rate than *DS* in both traces. It demonstrates that *TS* can more successfully find the targets within TTL. Because a locator moves directly to the target's latest appearance location in *TS*, while follows the movement path of the target in *DS*. Also, in *DS*, locators must move to the home-areas of the targets first, while in *TS*, the locators directly search from their current locations. Therefore, *TS* enables locators to search more quickly. Also, *TS* relies on multiple agents to search the target in its possible locations, while *DS* only considers the place with the highest visiting probability as the next destination. We also see that *TS** generates a slightly higher success rate than *TS* in both traces, which indicates that ER exchanges can slightly improve the success rate.

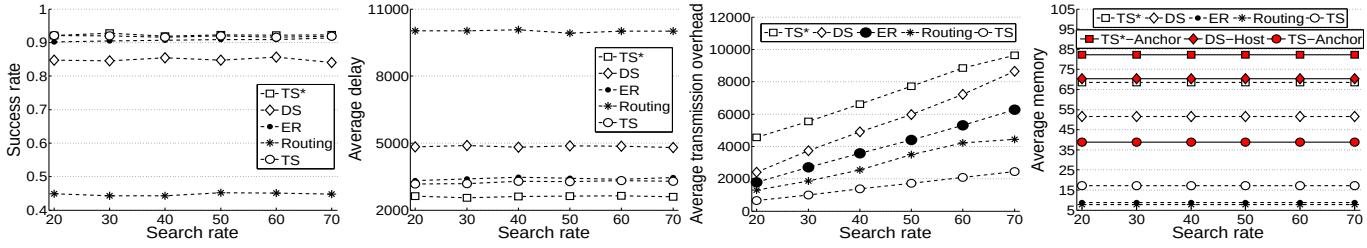
The success rate shows *DS*>*ER* in DART, but shows *ER*>*DS* in DNET. DART has much more nodes and sub-areas than DNET. Then, some locator may fail to receive the ERs of their targets in time, leading to lower success rate. This result verifies the effectiveness of ERs in guiding node searching in a small network. It also implies the necessity of anchors to facilitate the global information dissemination for node searching, especially in a large network area. We see that *TS** always achieves higher success rate than *ER*, which indicates the effectiveness of friend and visiting preference records in guiding node searching. Additionally, we find that except for *TS** and *TS*, the other three algorithms exhibit obvious improvement in success rate as TTL increases. This verifies the performance of *TS** and *TS* under small TTL.

2) Average Delay: Figure 8(b) and Figure 9(b) show the average delay for node searching in the algorithms under different search rates in DART and DNET, respectively. Figure 10(b) and Figure 11(b) show the average delay under different search TTLs in DART and DNET, respectively. We find that the average delays follow: *TS**<*TS*<*DS*<*ER*$\ll$*Routing* in



(a) Success rate. (b) Average delay. (c) Average transmission overhead. (d) Average node memory usage.

Fig. 8: Performance with different search rates using the DART trace.



(a) Success rate. (b) Average delay. (c) Average transmission overhead. (d) Average node memory usage.

Fig. 9: Performance with different search rates using the DNET trace.

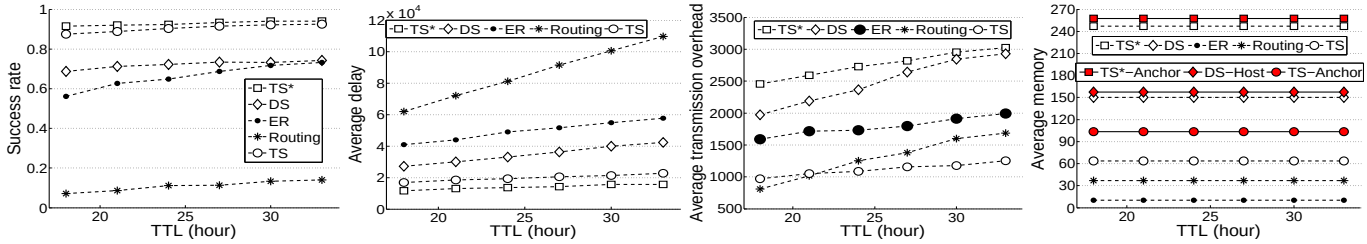
the DART trace, while it follows *TS**<*TS*≺*ER*<*DS*≪*Routing* in the DNET trace, where ≺ means "slightly smaller" and ≪ means "significantly smaller". The results are caused by the same reasons explained previously. Higher success rate means more node searches are successful during TTL. Recall we used the TTL as delay for failed searches. Thus, the methods with higher success rates produce low searching delay while the methods with lower success rates produce higher searching delay. *TS** generates a lower average delay than *TS* because locators may not need to query ERs from anchors. On average, *TS** reduces the average delay of *TS* by 30% and 5900 seconds (1.64 hours) in DART, and by 20% and 651 seconds (11 minutes) in DNET. The results indicate the high efficiency of *TS* and *TS** in terms of searching delay, and the effectiveness of their different components.

3) Average Transmission Overhead: Figure 8(c) and Figure 9(c) show the average transmission overhead of the algorithms under different search rates in DART and DNET, respectively. Figure 10(c) and Figure 11(c) show the average transmission overhead under different search TTLs in DART and DNET, respectively. We find that the result follows: *TS*<*Routing*<*ER*<*DS*<*TS**. *TS* has the least transmission overhead because nodes only need to report ERs to and request ERs from anchors without packet exchange between nodes. Nodes only report to anchors their friends and preferred locations once. Each ambassador carries the information in an anchor only when it moves to another sub-area. *Routing* and *ER* produce higher transmission overheads than *TS* because they require packet exchange between nodes upon entering. In *Routing*, a node keeps its meeting probabilities with other nodes and exchanges this information with its neighbors. In *ER*, a node keeps the ERs of node encounters. Therefore, *Routing* produces lower transmission overhead than *ER*. In *DS*, each node tells neighbors its transient VR before moving. The node also distributes its MP to long-staying nodes in this sub-area. Each node's home-area information is stored in all sub-areas. Therefore, *DS* generates a higher transmission overhead. *TS** has the transmission overheads of both *TS* and *ER*, thus it produces the maximum transmission overhead among the algorithms. We also see that the average transmission overhead of each algorithm increases as the TTL increases because more packet transmissions occur during a longer time period. On average, *TS* reduces the average transmission overhead of *TS** by 74% and 2047 packets in DART, and by 78% and 5603 packets in DNET. ER exchanges enable nodes to receive ERs more quickly for faster node searching. Then, TSearch can activate or inactivate the ER exchange function based on applications. Also, nodes can choose to use ER exchanges based on their individual desires.

4) Average Node Memory Usage: Figure 8(d) and Figure 9(d) show the average node memory usage of the algorithms under different search rates in DART and DNET, respectively. Figure 10(d) and Figure 11(d) show the average node memory usage under different search TTLs in DART and DNET, respectively. The figures also include the results for the anchors in *TS** and *TS* and the hosts in *DS* since they store more information than normal nodes. We see that the average memory usage follows: *ER*<*Routing*<*TS*<*DS*<*TS** in DART and *ER*≈*Routing*<*TS*<*DS*<*TS** in DNET.

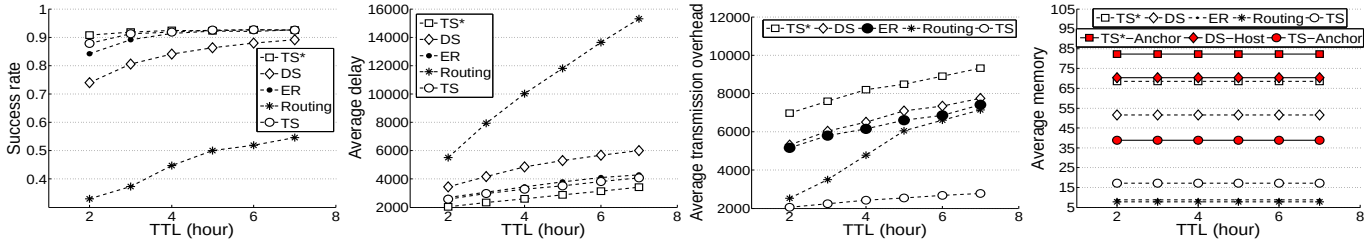
In *ER*, each node stores its received ERs of node pairs and deletes ERs after TTL. In *Routing*, each node stores its meeting probabilities with all other nodes. Therefore, the average memory usage of *ER* is less than *Routing* in DART. DNET has much fewer sub-areas, which enables a node to meet more nodes and also receive the ERs of most of other nodes. Thus, the average memory usage of *ER* is close to *Routing* in DNET.

In *TS* and *TS**, normal nodes maintain their own ERs, and friend and preferred location lists, and anchors store such information from nodes. In *TS**, nodes additionally need to exchange ERs. Thus, their average memory usage is higher than *ER* and *Routing*. In *DS*, each node needs to store its home-



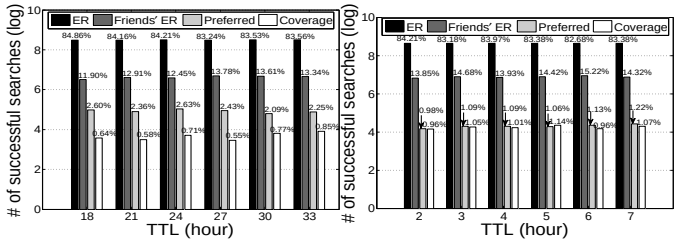
(a) Success rate. (b) Average delay. (c) Average transmission overhead. (d) Average node memory usage.

Fig. 10: Performance with different locator TTLs using the DART trace.



(a) Success rate. (b) Average delay. (c) Average transmission overhead. (d) Average node memory usage.

Fig. 11: Performance with different locator TTLs using the DNET trace.



(a) DART. (b) DNET.

Fig. 12: Breakdown of success rate in different search stages (log).

area and its MPT. The hosts in each sub-area need to store the home-area of each node, the transient VRs and MPT entries of some nodes. Since the MPT and VRs are collected and stored in hosts but ERs are generated upon encountering and shared among nodes in *TS**, the memory usage of *TS** is larger than *DS*. The average memory usage of hosts and anchors follows: *TS**>*DS*>*TS* in both traces. *TS**>*DS* is because that one sub-area has several hosts but one anchor, and the information is distributed to hosts in different sub-areas but each anchor needs to store global information of all nodes. In *TS**, due to ER exchange between nodes, anchors can more quickly receive ERs of far-away nodes. In *TS*, an anchor may not quickly receive the ERs of nodes in other sub-areas, which are carried by ambassadors. Therefore, anchors in *TS** have higher memory than in *TS*.

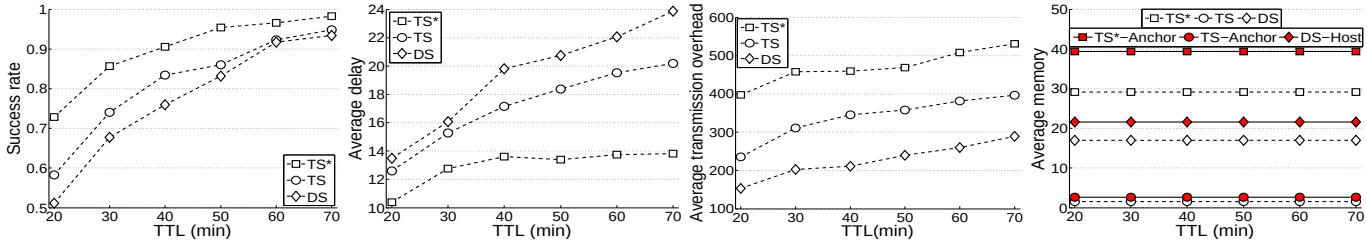
We use an example to illustrate the memory usage of *TS**. We find the peak number of information entries stored on each node is about 210 and 50 in DART and DNET, respectively. Each memory unit takes about 40 bytes. This means the actual average memory usage on each node is only about 8.4KB and 2KB in the two tests. Based on the average node memory usage and average anchor/host memory usage, we can conclude that *TS** is applicable on modern mobile devices.

5) Contribution of Different Stages in TSearch: To better illustrate the respective contribution of different search methods on success rate, we break down the total success rate into 4 stages by using different information of the target in node searching: ERs, friends' ERs, preferred locations and coverage area. Figure 12 shows the number of successful searches in log format. We see that most of the successful searches are achieved by following the target's ERs. The ERs of the target's friends have the second highest contribution on the success rate. The target's preferred location information has the third highest contribution on success rate. Finally, searching the coverage area contributes the least on the success rate, which means that the locators do not need to launch this searching stage in most cases. These results indicate the effectiveness of using the different information in node searching.

B. Experiment in Real Environment

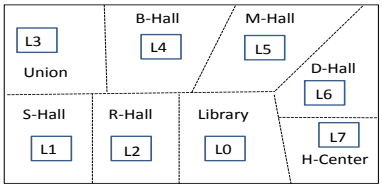
To test *TSearch*'s performance in real environment, we deployed *TSearch* on our campus and collected the mobility information of 9 students from 4 departments in our university. We selected 8 buildings frequently visited by the 9 students as the sub-areas. Based on the GPS on mobile phones, each node can determine its location. Compared with the previous two traces, the distance hence the node movement latency between two sub-areas is more accurate in this real-world test. The distribution of sub-areas and the summary of the data are shown in Figure 14(a) and 14(b). Since different search TTLs influence the results of different performance metrics, we varied the search TTL from 20 minutes to 70 minutes and set the search rate to 40. According to our campus map, a locator usually takes about 5 minutes to move from one sub-area to a neighboring sub-area.

The test results for different metrics for *TS**, *TS* and *DS* are shown in Figure 13. The orders of the results between the three algorithms are the same as those in the previous experiments due to the same reasons. When the search TTL increases, the success rate, average delay and average transmission overhead increase. When the TTL increases, more locators can find their targets after a longer delay, and along with higher transmission



(a) Success rate. (b) Average delay. (c) Average transmission overhead. (d) Average node memory usage.

Fig. 13: Performance with different TTLs using real environment data.



(a) Maps for sub-area division.

Item	Value
# Sub-areas	8
# Nodes	9
# Transits	147
Duration	4 days

(b) Statistical data.

Fig. 14: Configuration in the real environment.

overhead. We also see that when the TTL was set to 70 minutes, a successful locator takes only about 14 minutes to find the target node on average. Further, each node only needs 7 units of memory on average to support node searching. In conclusion, *TSearch* is effective and efficient in searching nodes.

VI. Conclusion

Previous node searching method in DTNs cannot achieve low searching delay by tracing a target along its movement path and also cannot guarantee high success rate by targeting the most frequently visited place (i.e., preferred locations) of the target. Our real trace data analysis confirms these drawbacks and also provides foundations for the design of our proposed TSearch node searching method. Rather than tracing along target's moving trail, TSearch aims to enable a locator to directly move towards the target by leveraging social network properties. It enables a locator to always move to the target's latest appearance place known by itself, the latest appearance place of the target's most frequently meeting node, or the preferred locations of the target. Then, the locator can find the target in its movement or destination. Also, to increase the searching success rate, the locator itself moves to the nearest preferred location of the target and asks a limited number of nodes that share other common preferred locations with the target to assist node searching. Extensive trace-driven and real-world experiments show that TSearch has much higher efficiency and effectiveness in node searching compared with previous methods. In our future work, we plan to further exploit nodes' social network properties to reduce node searching delay and overhead.

Acknowledgements

This research was supported in part by U.S. NSF grants NSF-1404981, IIS-1354123, CNS-1254006, CNS-1249603, Microsoft Research Faculty Fellowship 8300751.

References

[1] Y.-T. Yu, T. Punihaole, M. Gerla, and M. Sanadidi, "Content routing in the vehicle cloud," in *Proc. of MILCOM*, 2012.

[2] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," in *Proc. of ASPLOS-X*, 2002.

[3] B. Thorstensen, T. Syversen, T. Walseth, and T.-A. Bjørnvold, "Electronic shepherd - a low-cost, low-bandwidth, wireless network system." in *Proc. of MobiSys*, 2004.

[4] J. Huang, S. Amjad, and S. Mishra, "Cenwits: a sensor-based loosely coupled search and rescue system using witnesses." in *Proc. of SenSys*, 2005.

[5] J.-H. Huang, L. Jiang, A. Kamthe, J. Ledbetter, S. Mishra, A. Cerpa, and R. Han, "Sensearch: Gps and witness assisted tracking for delay tolerant sensor networks," in *Proc. of Ad Hoc-Now*, 2009.

[6] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem." in *Proc. of SIGCOMM*, 2007.

[7] S. Lu, Y. Liu, Y. Liu, and M. Kumar, "Loop: A location based routing scheme for opportunistic networks." in *Proc. of MASS*, 2012.

[8] L. Xie, P. Chong, and Y. Guan, "Leader based group routing in disconnected mobile ad hoc networks with group mobility," *WIRELESS PERS COMMUN*, vol. 71, no. 3, 2013.

[9] W. Gao and G. Cao, "On exploiting transient contact patterns for data forwarding in delay tolerant networks." in *Proc. of ICNP*, 2010.

[10] X. Zhang and G. Cao, "Transient community detection and its application to data forwarding in delay tolerant networks." in *Proc. of ICNP*, 2013.

[11] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proc. of MobiHoc*, 2007.

[12] K. Chen and H. Shen, "Smart: Lightweight distributed social map based routing in delay tolerant networks." in *Proc. of ICNP*, 2012.

[13] K. Chen and H. Shen, "DSearching: Distributed Searching of Mobile Nodes in DTNs with Floating Mobility Information," in *Proc. of INFOCOM*, 2014.

[14] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in *Proc. of MobiCom*, 2004.

[15] X. Zhang, J. Kurose, B. N. Levine, D. Towsley, and H. Zhang, "Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing," in *Proc. of MobiCom*.

[16] J. Zhao, Y. Zhu, and L. M. Ni, "Correlating mobility with social encounters: Distributed localization in sparse mobile networks," in *Proc. of MASS*, 2012.

[17] K. Chen and H. Shen, "Leveraging social networks for p2p content-based file sharing in disconnected manets." *IEEE TMC*, 2013.

[18] K. Chen, H. Shen, and H. Zhang, "Leveraging social networks for p2p content-based file sharing in disconnected manets." *IEEE TMC*, 2014.

[19] F. Li and J. Wu, "Mops: Providing content-based service in disruption-tolerant networks." in *Proc. of ICDCS*, 2009.

[20] B. Karp and H.-T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proc. of MobiCom*, 2000.

[21] M. Kim, D. Kotz, and S. Kim, "Extracting a mobility model from real user traces." in *Proc. INFOCOM*, 2006.

[22] L. Xie, P. Chong, and Y. Guan, "Routing strategy in disconnected mobile ad hoc networks with group mobility," *EURASIP Journal on WCN*, vol. 2013, no. 1, 2013.

[23] M. Thomas, A. Gupta, and S. Keshav, "Group based routing in disconnected ad hoc networks," *EURASIP Journal on WCN*, vol. 4297, 2006.

[24] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks." *Mobile Computing and Communications Review*, vol. 7, no. 3, 2003.