Topology Control Based on Set Cover for Delay-Tolerant Networks

Xuyan Bao, Xiaojin Zhou, Xiaojuan Wang, Yong Zhang, and Mei Song
Laboratory of Work Safety Intelligent Monitoring, Beijing University of Posts and Telecommunications, Beijing 100876, China
Email: {sdlbxy19890106, zxjxwn}@sina.com; wj2718@163.com; {Yongzhang, songm}@bupt.edu.cn

Abstract —In traditional Delay Tolerant Networks (DTNs), unpredictable mobility pattern, network partitioning and the lack of continuous end-to-end path make it challenging to design network protocols. However, the applicable range of DTNs has been extended to some new types of networks such as Mobile Social Network (MSN), vehicular ad-hoc network (VANET), etc. Based on these DTN traces, we first reveal two major characteristics namely periodicity and path redundancy by quantitative analysis, which substantially provides the feasibility and operability of topology control. In this paper, we study the topology control problem in DTN by modeling such time-evolving network as a space-time graph and connecting it with the classical set cover problem. The aim of topology control is to build a sparse subgraph from the original space-time graph such that 1) any two nodes are connected; 2) the cost of subgraph is minimized. Furthermore, we propose a topology control algorithm that can significantly reduce the total cost of network while guarantee the connectivity. We conduct our experiments on both random DTN networks and realistic DTN traces. Evaluation results demonstrate the efficiency of the proposed algorithm.

Index Terms—Topology control, delay tolerant networks, set cover

I. Introduction

Delay-Tolerant Networks (DTNs) [1] arises from the traditional Ad-Hoc networks to support a variety of applications in challenging environments, such as space communications, battlefields, mobile sensor networks. Some characteristics of DTNs like low node density, high mobility, and the lack of end-to-end path would inevitably yield intermittent connectivity and network partitioning. Thus, some routing schemes [2]-[9] have been proposed to deal with the time-varying topology and improve the performance of data forwarding. However, current methods either exploits the forwarding metric to stochastically control the data replication or use static social metrics to select the best relay nodes. All these methods ignore the time-evolving characteristics of the network, thus may lead to the inferior performance.

The topological form of DTNs generally evolves over time since dynamic changes always happen if the frequent movements and link breakage of some nodes arise. Such time ordering dynamics are often neglected in routing design or simply modeled by a stochastic contact process. Nowadays, the applicable range of DTNs has been extended to some new types of networks such as pocket switched networks [7], mobile social networks [10], vehicular networks [11], where high node contact density exists and the evolution of topology depends on human mobility behavior. Based on traces collected at different DTN scenarios like university campus, conference sites and taxi system, two major characteristics can be observed as follows: 1) *Periodicity*, which indicates that the realistic trace always repeat over time due to the recurrent daily schedule of human activity. 2) *Path Redundancy*, which indicates that there exists more than one hop-by-hop paths between a pair of nodes. In this kind of time-evolving DTNs, traditional communication protocols for DTN may be inefficient. However, it facilitates us to discover the evolving pattern of topology from historical traces and conduct efficient control mechanism.

Topology analysis has become a remarkable issue in Ad-Hoc networks and different control algorithms aim at different objectives (connectivity, energy efficiency, throughput and robustness to mobility). The goal is to construct a desired structure based on a static and connected topology. However, in DTNs, the lack of continuous end-to-end path makes existing control algorithms failed. Therefore, it is vital to maintain an efficient and low-cost topology of DTNs especially with various link costs or high-density of wireless devices. In this paper, we model a time-evolving DTN as a space-time graph which captures information from both space and time dimensions and study the topology control problem in DTNs. Our major contributions are summarized as follows:

- We design four novel metrics *periodic index* (*PI*), (α, β, γ) to extract characteristics from realistic DTN traces. To our best knowledge, this is the first attempt to quantitatively confirm the existence of periodicity and path redundancy in DTNs.
- We first define the set cover problem (SCP) on space time graph and propose a heuristic method to identify the time window that fulfills partial connectivity based on enhanced greedy set cover algorithm.
- We define the fully-covered time window based on SCP and then define the topology control problem

Manuscript received September 30, 2015; revised March 8, 2016.
Corresponding author email: sdlbxy19890106@sina.com.
doi:10.12720/jcm.11.3.272-281

that aims to build a sparse subgraph from the original space-time graph such that 1) any two nodes are connected; 2) the cost of subgraph is minimized.

- We propose a novel topology control algorithm based on the fully-covered time window that can significantly reduce the total cost of network while guarantee the connectivity.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 gives a quantitative analysis of real mobility traces. A topology-control algorithm based on set cover is proposed in Section 4. Section 5 evaluates the performance under different set cover algorithms by synthetic and trace-driven simulations. Section 6 concludes the paper.

II. Related Work

A. Topology Control in Ad-Hoc and Delay-Tolerant Networks

The absence of a fixed infrastructure and the feature of mobility imply that an ad-hoc network does not have a stable or static topology. Indeed, an important task of such a network form is to determine an appropriate topology that can be evaluated based on the following factors: connectivity, energy-efficiency, throughput, and robustness to mobility [12]. To tackle the issue of intergroup communication in Autonomous Mobile Mesh Network (AMMNET), a distributed tracking and adaptation approach is proposed in [13], which is capable of ensuring good connectivity. Wattenhofer *et al*. [14] provide a cone-based topology control algorithm by varying the transmission power, in which the power consumption can be close to the optimal. Utilizing Cognitive Radio (CR) technology, a distributed Prediction-Based Cognitive Topology Control (PCTC) scheme [15] is able to construct an efficient and reliable topology based on link prediction. All of these topology control schemes are tailored for some specific types of ad-hoc networks. Besides, how to control topology in delay-tolerant networks have also been studied in [16], [17], [18] recently.

In [16], Huang *et al*. model the time-evolving network as a directed space-time graph and derive a series of topology control algorithms from the *directed Steiner tree* problem, which minimizes the total cost of topology while maintaining the connectivity. Moreover, the new reliable topology design problem is formulated in [17], as extended research of [16], aiming at ensuring the reliability to be higher than a required threshold. Also, in [18], taking the energy consumption into consideration, Altman *et al*. formulate an optimal control problem based on fluid models to maximize the message delivery probability. However, for one thing, some of these methods are only applicable in a specific type of ad-hoc networks, for another thing, in time-evolving DTNs, current research mostly focus on volatile and temporal topology while neglecting the fact that there exists path redundancy in realistic traces, which yields stable vertex set that can cover all the nodes over time.

B. Time-Evolving Characteristics

The problem of how to model or analyze the time-evolving characteristic has been studied in delay tolerant networks. Community analysis is one of the promising research direction, however most existing community-detection methods are only effective in static networks. To cope with topology evolution, a contact-burst-based clustering method is proposed in [19] to detect transient communities and a dynamic clustering framework [20] can be used to detect group changes over time in a hierarchical approach. In addition, [11] uses evolving graph to capture the characteristics of the vehicular network topology and design the reliable routes. Ferreira *et al*. [21] use evolving graph to evaluate routing protocols in mobile wireless networks based on journey metrics. However, all of these works only focus on routing design by utilizing either dynamic communities or evolving graphs. In this paper, we will investigate topology control based on space-time graph [22] in which the temporal characteristics from both the space and time dimensions can be captured.

C. Set Cover Problem

The Set Cover Problem (SCP) has been extensively used in wireless networks to model the problem of selecting [23] or deploying [24] the nodes that achieve the optimal coverage. In [23], the classic Greedy Set Cover (GSC) algorithm is improved by elimination of fully covered nodes and used to select the optimal forwarding set for broadcasting. Wang *et al*. [24] propose a device placement framework for wireless sensor networks by formulating the minimum set covering problem, aiming to minimize the device cost with lifetime constraints. Indeed, the SCP is one of the well-known NP-hard problems [25]. A thorough comparative analysis of nine approximation algorithms, including greedy variants, fractional relaxations, etc, is presented in [26], which suggests that the GSC is one of the best polynomial approximations to the SCP. In this paper, our proposed algorithm for topology control DTNs is the first attempt to exploit the time window on which the SCP can be solved.

III. Characteristics Extraction

In order to extract the characteristics from realistic DTN traces, we analyze four experimental data sets gathered in CRAWDAD archive, referred to as *Infocom05, Calabria, Cabspotting, Roma*. We pick out these four data sets as research objects since almost all the DTN applicable scenarios are covered. In the traces mentioned above, the contact characteristics introduced by either human-carried mobile devices or vehicular-based terminals are resulting from human activity which reflects periodicity in a certain extent. Also, some people who share the same part of time schedule would meet

together, which creates high-density contacts that yield path redundancy. The network forms behind these four traces are classified as: i)*Mobile Social Networks (MSNs)* in which mobile users move around and communicate with each other via short-distance wireless devices [10]; ii)*Vehicular Ad-hoc Networks* (*VANET*) in which vehicle-to-vehicle communications are enabled by exploiting ad-hoc connectivity [11]. These traces cover various environments, ranging from conference site to university campus, to metropolitan city, with a timescale from several days to half a month.

TABLE I: COMPARISON OF THE FOUR EXPERIMENTAL DATA SETS

Experimental data set	Infocom05	Calabria	Cabspotting	Roma
Device	iMote	Android Phone	GPS tracker	GPS tracker
Technology	Bluetooth	Bluetooth	GPS	GPS
Duration	3 days	1 week	1 month	1 month
Granularity (seconds)	120	180	30	7
Number of Devices	41	35	536	320
Number of Contacts	22,459	12,571	1,060,922	197,297

- *Infocom05*: The contact trace was collected in INFOCOM 2005 conference, where devices named iMote were distributed to 41 attendees for 3 days. Each contact is recorded as a tuple (MAC address, start time, end time). The meeting between iMotes is classified as an 'internal' contact, while the meeting of other types of Bluetooth devices is an 'external' contact.
- *Calabria*: The experiment lasted one week (only working days) and was conducted at the campus of University of Calabria in Rende (CS), Italy. The Bluetooth device proximity data were collected by an ad-hoc Android application called SocialBlueConn. This application logged both the internal contacts between 15 participants and other 20 external mobile nodes.
- *Cabspotting*: This data set contains mobility traces of taxi cabs in San Francisco, USA. It contains GPS coordinates of approximately 500 taxis collected over 30 days in the San Francisco Bay Area. Each taxi is equipped with a GPS receiver and sends a location-update (time-stamp, identifier, geo-coordinates) to a central server.
- *Roma*: This data set, which is very similar to *Cabspotting* except that the urban area is larger, contains mobility traces of approximately 320 taxi cabs in Rome, Italy over 30 days. Each taxi driver has a tablet that periodically (7s) retrieves the GPS position and sends it to a central server.

The four data sets are summarized in Table I, with diverse statistics in terms of trace duration, the number of contacts, test environment, etc. In the following, we conduct numerical analysis on all the data sets for characteristics extraction.

A. Periodicity

One salient characteristic of realistic mobility trace is *reappearance*, which is supported by recurrent daily or weekly schedule of human activities. Next, to quantitatively analyze the *Periodicity* of mobility traces, we design a *periodic index* based on the repetitive rate of encountered node-set over a complete period. To proceed, node contacts are represented by the network *contact graph* $G(V,E)$, where the contact process between a pair of nodes $u,v \in V$ is modeled as an edge $e_{uv} \in E$. Then, for any $u \in V$, the *periodic index* $PI(u)$ is defined as:

$$PI(u) = \frac{\left| N_u^{[t_1,t_2]} \cap N_u^{[t_1+T_P,t_2+T_P]} \right|}{\left| N_u^{[t_1,t_2]} \right|} \tag{1}$$

where $N_u^{[t_1,t_2]}$ denotes the neighbor set of u over the aggregated time window $[t_1,t_2]$. The numerator term indicates the mutual nodes that u has contacted in both $[t_1,t_2]$ and $[t_1+T_p,t_2+T_p]$ time interval. Thus, Eq.(1) reveals such a measurement of *periodic index* as the size of mutual node set over that of previous neighbor set.

We fix $T_p = 24$ hours for all data sets and set the length of time window as 1 hour for *Calabria*, and 4 hours for the rest. The reason why the fine-grained setting is suitable for *Calabria* is that the testing device is only powered-on from 12:00 to 20:00, furthermore, there is no contact after 17:00 in each day. Consequently, we can investigate the aggregated *PI* over the following time windows 12:00-13:00 ,..., 16:00-17:00 for *Calabria* and 0:00-4:00, 4:00-8:00 ,..., 20:00-24:00 for the rest. The specific calculation is a two-step approach: First, for each node, we compute the *PIs* between successive days across the time-span of data sets and take the average. Second, for each time window, the aggregated *PI* is obtained by taking the average of all nodes' *PIs* with equal weights.

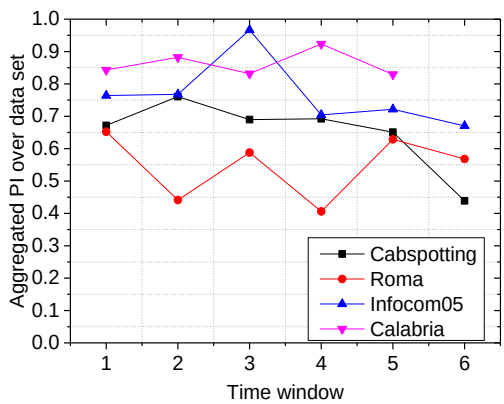


Fig. 1. Aggregated PI on various time window

As shown in Fig. 1, in *Cabspotting* trace, the *PIs* of the first 5 time windows are over 65 percent, which means the neighbor set of a node has 65% similarity between successive days. However, the last *PI* is just 43.9 percent,

the reason is that the time window referring to this point is 0:00-4:00. During such period, a cab is more likely to stay at a fixed location, which decreases the contacts with other cabs. As previously described, *Roma* covers larger urban area than *Cabspotting*, which dilutes the node density and implicitly reduces the contact probability. Therefore, the Median of *Roma's PI* is about 58 percent. *Infocom05* and *Calabria* present superior periodicity since the Medians of *PI* are about 74.3 and 84.3 percent respectively. The most remarkable point is the *PI* of the third time window in *Infocom05*, i.e., 96.67%. This agrees with the fact that people always stay in a fixed conference site everyday based on uniform meeting schedule. To our best knowledge, this is the first approach that quantitatively measures the *periodicity* of realistic mobility traces. Above all, we show that the *periodicity* does exist in *MSNs* and *VANET*.

B. Path Redundancy

Another un-conspicuous characteristic of realistic mobility trace is *path redundancy*, which is uncovered by the numerical analysis of four data sets. Next, we propose three quantitative metrics to fully represent the degree of *path redundancy* over a time window $[t_1, t_2]$ as follows:

$$\alpha(t_1,t_2) = \frac{\sum_{(u,v)\in V_c^{[t_1,t_2]}} \left| P_{u\to v}^{[t_1,t_2]} \right|}{\left| V_c^{[t_1,t_2]} \right|} \tag{2}$$

$$\beta(t_1,t_2) = \frac{2\left| V_c^{[t_1,t_2]} \right|}{|V|(|V|-1)} \tag{3}$$

$$\gamma(t_1,t_2) = \frac{\left| V_{c^+}^{[t_1,t_2]} \right|}{\left| V_c^{[t_1,t_2]} \right|} \tag{4}$$

The variables appeared in Eq. (2)-(4) are defined in Tab II. Observing from Eq. (2), Eq. (3) and Eq. (4), the above indexes α, β, γ represent the average path number of connected node-pairs: accumulating the path number of each connected node-pair $u \to v$ during $[t_1, t_2]$ divided by the number of connected node-pairs; the percent of node-pairs that have at least one path: the number of connected node-pairs divided by the number of all possible node-pairs; the percent of connected node-pairs that have more than one path: the number of node-pairs that have redundant paths divided by the number of connected node-pairs; respectively.

To facilitate our analysis, we generate a special data structure, where the value of *i*th row and *j*th column denotes the neighbor set of node j in *i*th time-slot. Then, all available paths of a node-pair in a given time window can be searched in a *recursive* approach. For all data sets, we investigate α, β, γ under various length of time window, denoted as T_l. To proceed, we fix the value of T_l orderly from *5min* to *40min* with *5min* in between and divide one day into 24 hours, then calculate α, β, γ over 24 time windows with length T_l, which are randomly selected from each and every hour, to take the average. Here the value of T_l is fixed at some discrete values with 5min in between. The reason why we adopt such simplified handling is to decrease recursive depth, which further reduce computational overhead. Suppose we take a fine-grained step unit, i.e., 1min and vary T_l from 1min to 60min. If T_l is taken as 10min, the recursive search for any node-pair has to be conducted in a path tree with 10 hops, which yield tremendous computational complexity. Indeed, the revelation of the rising tendency is enough to uncover the desired characteristic.

TABLE II: ANNOTATIONS IN SECTION III-B

$V_c^{[t_1,t_2]}$	The set of connected node-pairs, $(u,v) \in V_c^{[t_1,t_2]}$ if there exists at least one path connecting (u,v) in $[t_1,t_2]$.
$P_{u\to v}^{[t_1,t_2]}$	The path-set including all the paths which exist in $[t_1,t_2]$ ***from*** u ***to*** v.
$V_{c^+}^{[t_1,t_2]}$	The set of connected node-pairs, $(u,v) \in V_{c^+}[t_1,t_2]$ if there exists more than one path connecting (u,v) in $[t_1,t_2]$.

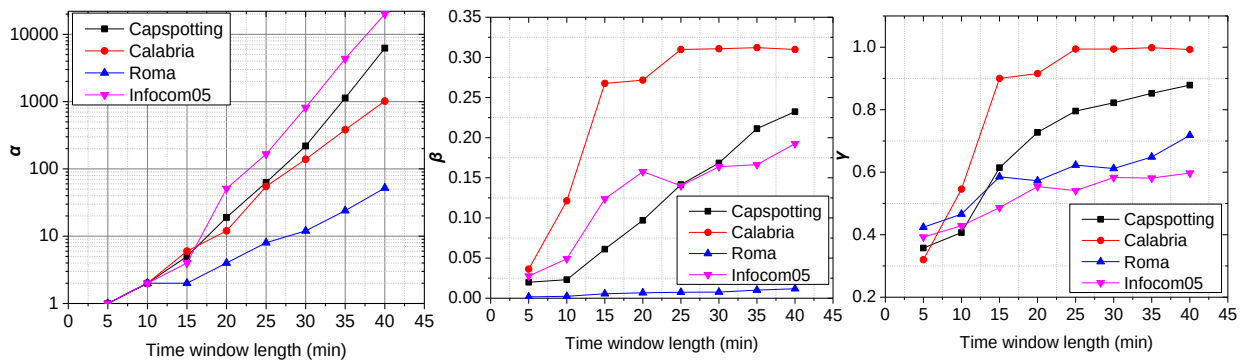
As shown in Fig. 2(a), the average path number of all data sets are sharply increasing as we enlarge the time window, since more opportunistic paths with larger delay for any pair of nodes become available. In addition, we present the value of α in a log-scale to accommodate display. Observing from Fig. 2(b), the β index which reflects the network connectivity is rising up when T_l varies from *5min* to *40min* but remains within 0.31, which reveals a fact that a realistic network can hardly reach fully-connected state. Fortunately, the medians of γ shown in Fig. 2(c) are 0.95, 0.76, 0.60, 0.55, respectively. It can be seen that path redundancy does exist between connected node-pairs laying the foundation for topology control.

IV. TOPOLOGY CONTROL

The existence of *periodicity* and *path redundancy* has been validated in Section III, which substantially provides the feasibility and operability of topology control. In this section, we first model the evolving topology as a space-time graph, then formulate and solve the set cover problem. Last, a topology-control algorithm is proposed based on optimal covering.

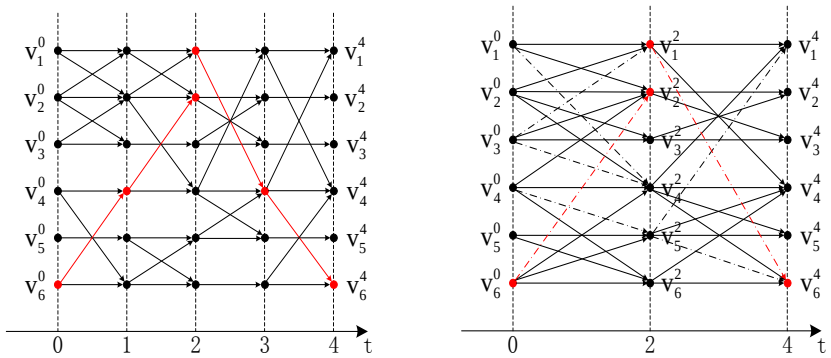
A. Space-Time Graph

Traditional static graph cannot represent the evolution of contact process among nodes. In fact, one static graph is a snapshot of nodes and their contacts appear at certain time point. To capture such dynamic evolutions, we use a sequence of network snapshots to build a space-time graph [22].



(a) α index (b) β index (c) γ index

Fig. 2. Path redundancy index on different T_l



(a) (b)

Fig. 3. Space-Time graph

Assume that the time is divided into equal and discrete time slots $\{1,...,T\}$. Let $V=\{v_1,...,v_n\}$ be the set of network nodes. Let $G^t=(V^t,E^t)$ be a static graph representing the network topology at time slot *t*. Then, the network evolution can be modeled as a union of consecutive static graphs $\{G^t \mid t=1,...,T\}$ where a space-time graph $\mathcal{G}=(\mathcal{V},\mathcal{E})$ is constructed. Such representation includes temporal information about the transient topology and order of contacts. See Fig. 3 (a) for illustration. In $\mathcal{G}$, $T+1$ slices of nodes are defined by the vertex set $\mathcal{V}=\{v_i^t \mid i=1,...,n \text{ and } t=0,...,T\}$ and the node set in each slice is denoted by $V^t=\{v_i^t \mid i=1,...,n\}\subset\mathcal{V}$. The space between consecutive slices is a time slot. A link $\overrightarrow{v_i^{t-1}v_i^t}$ (horizontal links in Fig. 3) connects the same node v_i across $(t-1)$th and *t*th slices, which means v_i carrying the packet in the *t*th time slot. A pair of symmetric links $\overrightarrow{v_i^{t-1}v_j^t}$ and $\overrightarrow{v_j^{t-1}v_i^t}$ are generated if v_j contact v_j in the *t*th time slot, which can be used to forward a packet between them.

Notice that for basic space-time graph in Fig. 3(a), we assume that each time slot is long enough for only one-hop transmission. Multi-hop paths can be obtained by aggregating time slots as shown in Fig. 3(b). The first and last two time slots are merged respectively to generate the aggregated time window. All the one-hop links are reserved while the repetitive links are ignored, meanwhile all the two-hop links are available. For example, In Fig. 3(b), the original links $\overrightarrow{v_1^0v_2^1}$ and $\overrightarrow{v_1^1v_2^2}$ in Fig. 3(a) are identical and helpless to be kept, they become a single link $\overrightarrow{v_1^0v_2^2}$ in new aggregated time window $[0,2]$; two-hop paths like $\overrightarrow{v_6^0v_4^1v_2^2}$ and $\overrightarrow{v_1^2v_4^3v_6^4}$ (marked by red dashed line) are built at time window $[0,2]$ in Fig. 3(b), which consists of $\overrightarrow{v_6^0v_4^1}$, $\overrightarrow{v_4^1v_2^2}$ and $\overrightarrow{v_1^2v_4^3}$, $\overrightarrow{v_4^3v_6^4}$ (marked by red solid line) in Fig. 3(a), respectively. The aggregation of consecutive time slots increases the path redundancy, but also provides more links for selection. In addition, we assume that for each one-hop link $e\in\mathcal{E}$ there is a cost $c(e)$, which corresponds to the energy consumption of delivering a packet upon that link. Moreover, we can define the *minimum cost path* $p(u,v)$ from u to v and the total cost is the sum of costs of all links in $p(u,v)$.

B. Set Cover Based on Space-Time Graph

We now define the *set cover problem* (SCP) on space-time graphs.

Definition 1: Given a connected space-time graph $\mathcal{G}$ *over* $[0,T]$ *and an aggregated time window* $\mathcal{T}_w=[t_s,t_e]\,(0\le t_s<t_e\le T)$, *which is obtained by aggregating* t_e-t_s *time slots from* t_e *to* t_s, *the aim of set cover problem is to determine whether there exists a minimum vertex set* $V_{\min}\subseteq V^{t_s}$ *whose members can reach all nodes in* V^{t_e}.

Note that *connectivity* has a different definition from that of a static graph, which is described in [16].

Definition 2: A space-time graph $\mathcal{G}$ is connected over $[0,T]$ if and only if there exists at least one directed path for each pair of nodes $(v_i^0, v_j^T)(i \text{ and } j \text{ in} [1,n])$.

It is guaranteed that we can always find a trivial solution to the *set cover problem* on space-time graphs, i.e., take $\mathcal{T}_w = [0,T]$, $V_{\min} = V^0$. The basic idea of Definition 1 is to identify a certain time interval $[t_s, t_e]$ whether every node can be reached at t_e by a minimum number of nodes through directed paths or not. It is interesting to see that there exists more than one $\mathcal{T}_w$ on which the SCP can be solved over the period of *T*. As shown in Fig. 3(b), $\mathcal{T}_w = [0,2]$ is a feasible time window since there exists a minimum vertex set $(v_2^0, v_6^0) \subset V^0$ covering all the nodes in V^2. To identify such $\mathcal{T}_w$, we propose an algorithm named *Time-window Identification based on Enhanced Greedy Set Cover (TwI-EGSC)* by heuristically solving the SCP on a given $\mathcal{T}_w$.

The pseudo-code of *TwI-EGSC* is illustrated in Algorithm 1. To proceed, let $N(u)$ denote the neighbor set of $u \in V^{t_s}$ and we say that:

Definition 3: $\forall v \in V^{t_e}$, $v \in N(u)$ iff there exists at least one directed path from u to v.

The output variables: *flag* indicates whether $\mathcal{T}_w = [t_s, t_e]$ is a valid aggregated time window on which the SCP can be solved, and $V_{\min}$ is used to save the minimum vertex cover set. In the first phase (lines 3-9), $C(V^{t_s})$ eventually represents all the neighbor nodes in V^{t_e} that V^{t_s} can cover, then the condition in line 7 is sufficient for judging if a feasible solution can be found. *TwI-EGSC* continues with the enhancement phase (lines 12-18) where the nodes in V^{t_s} that are unique in covering a node in V^{t_e} are selected. Also, V^{t_s}, V^{t_e} are updated by removing all the chosen and covered nodes. In the final phase, the steps defined by *GSC* (lines 19-24) are executed and the node that covers more nodes in V^{t_e} is elected, then the algorithm terminates when V^{t_e} becomes empty. However, *TwI-EGSC* only fulfills the partial connectivity in a given time window $[t_s, t_e]$, we will propose an extended algorithm for topology control by exploiting the minimum covering set in order to realize the fully-connectivity over a space-time graph.

C. Topology Control Based on Fully-Covered Time Window

To illustrate the *topology control problem*, let $\mathcal{W} = \{w_1, ... w_m\}$ be the set of all time windows on which the SCP can be solved over $[0,T]$ and each $w_i \in \mathcal{W}$ is represented as $[s_i, t_i]$, $0 \le s_i < t_i \le T$. Then, let $\mathcal{C} = \{C_1, ..., C_m\}$ be the set of all minimum vertex sets corresponding to $\mathcal{W}$, each $C_i \in \mathcal{C}$ can be obtained by invoking *TwI-EGSC* upon w_i. Last, a fully-covered subset $\mathcal{W}_f = \{w_1^f, w_2^f, ...\} \subseteq \mathcal{W}$ is given by the following definition.

Algorithm 1 Identify $\mathcal{T}_w$ based on Enhanced Greedy Set Cover

Input: $\mathcal{G}, \mathcal{T}_w = [t_s, t_e] (0 \le t_s < t_e \le T)$.

Output: $flag, V_{\min}$.

1: $flag \leftarrow 0$, $V_{\min} \leftarrow \phi$
2: */∗ Identify $\mathcal{T}_w$ if the SCP cannot be solved.*
3: $C(V^{t_s}) \leftarrow \phi$
4: **for all** $u \in V^{t_s}$ **do**
5: $C(V^{t_s}) \leftarrow C(V^{t_s}) \cup N(u)$
6: **end for**
7: **if** $C(V^{t_s}) \ne V^{t_e}$ then
8: **return**
9: **end if**
10: */∗Improved greedy search for the minimum vertex set*
11: $V_{\min} \leftarrow \phi$
12: **for all** $v \in V^{t_e}$ do
13: **if** $\exists u \in V^{t_s} : \left[(v \in N(u)) \wedge (v \notin N(\varepsilon), \forall \varepsilon \ne u \in V^{t_s}) \right]$ **then**
14: $V^{t_e} \leftarrow V^{t_e} - N(u)$
15: $V_{\min} \leftarrow V_{\min} \cup u$
16: $V^{t_s} \leftarrow V^{t_s} - u$
17: **end if**
18: **end for**
19: **while** $V^{t_e} \ne \phi$ **do**
20: $u = \arg\max_{\varepsilon \in V^{t_s}} \left\{ \left| N(\varepsilon) \cap V^{t_e} \right| \right\}$
21: $V^{t_e} \leftarrow V^{t_e} - N(u)$
22: $V_{\min} \leftarrow V_{\min} \cup u$
23: $V^{t_s} \leftarrow V^{t_s} - u$
24: **end while**
25: $flag \leftarrow 1$
26: **return** $flag, V_{\min}$

Definition 4: $\forall w_i \in \mathcal{W}$, we say $w_i \in \mathcal{W}_f$ iff there exists at least one directed path for each pair of nodes (u,v) over $[0, s_i]$, where $u \in V^o, v \in C_i$.

Obviously, for any $w_i^f \in \mathcal{W}_f$, C_i can be regarded as a bridging node set that achieves fully-connectivity over $[0, t_i]$. Now, the *topology control problem* on a space-time graph $\mathcal{G}$ by exploiting $\mathcal{W}_f$ can be defined as:

Definition 5: Given a connected space-time graph $\mathcal{G}$ over $[0,T]$, the aim of topology control problem is to build a sub space-time graph $\mathcal{H}$ of $\mathcal{G}$, based on a fully-covered

time window set $\mathcal{W}_f$ *, such that 1)* $\mathcal{H}$ *is still connected; 2) the cost of* $\mathcal{H}$ *is minimized.*

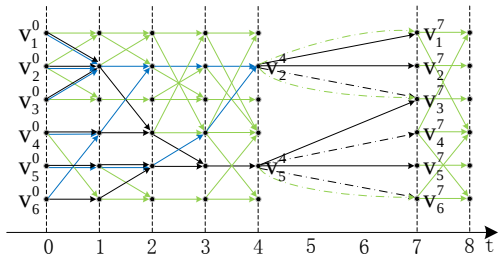


Fig. 4. Topology control on space-time graph

Notice that our objective is to construct a cost-oriented and fully-connected sparse structure. Fig. 4 shows a feasible solution for topology control on a space-time graph over $[0,8]$ using a fully-covered time window $\mathcal{T}_w = [4,7]$. In $\mathcal{T}_w$, we run Dijkstra's path-searching algorithm to precisely determine the neighbor relationships between $v_i^4 \in V^4$ and $v_i^7 \in V^7$, then invoke *TwI-EGSC* to output the minimum vertex set such that every node in V^7 can be reached by the nodes in $V_{\min}$ of $\mathcal{T}_w$, i.e., $\{v_2^4, v_5^4\}$, through one-hop or multi-hop paths. All the minimum cost paths from $V_{\min}$ to V^7 are kept and marked as black lines (solid and dashed). In $[0,4]$, we again run Dijkstra's algorithm and keep all the minimum cost paths from V^0 to v_2^4 (in blue) and v_5^4 (in black). Last, all the links or paths in green are removed to build a sparse $\mathcal{H}$ of $\mathcal{G}$. Apparently, all node-pairs are guaranteed to be connected in $\mathcal{H}$ over $[0,7]$. However, Fig. 4 only presents a possible fully-covered time window that constructs a connected sub space-time graph $\mathcal{H}$. Cond 2) has not been met in Def. 5, thus we propose an algorithm named *Topology Control based on Fully-covered Time-window* ($TC-FcTw$) by traversing all the possible fully-covered time windows and then identify an optimal sub graph on a given space-time graph.

The pseudo-code of $TC-FcTw$ is presented in Algorithm 2. The outer-loop (line 3) is used to traverse all the time windows over the period of T . In the first phase (lines 5-13), a variable named *flag* is used to indicate a fully-covered $\mathcal{T}_w$. Meanwhile, a minimum vertex set $C \subseteq V^{t_s}$ that cover every node in V^{t_e} can be obtained by invoking $TwI-EGSC$ if it exists. Continuing with the second phase (lines 15-23), for each $v \in C$, we check if v can be reached by every node in V^0 by running Dijkstra's algorithm. In addition, all the minimum cost paths connecting C and V^{t_e} or V^0 and C are stored in $P_{\mathcal{T}}$ and P_l respectively (lines12, 22). After this if *flag* still equals to 1, which means the current $\mathcal{T}_w$ is a feasible solution to achieve fully-connectivity, the steps of lines 24-26 would be skipped over. At last, we construct a subgraph $\mathcal{H}$ from the union path set of $P_{\mathcal{T}}$ and P_l (lines 28-33) and update the optimal subgraph $\mathcal{H}_m$ if the total cost of $\mathcal{H}$ is less than that of $\mathcal{H}_m$.

Algorithm 2 Topology Control based on Fully-covered $\mathcal{T}_w$

1: $\mathcal{H} \leftarrow \phi, \mathcal{H}_m \leftarrow \phi, c(\mathcal{H}_m) \leftarrow \infty$
2: */* Traverse all the possible time windows.*
3: **for all** $\mathcal{T}_w = [t_s, t_e], 0 \le t_s < t_e \le T$ **do**
4: */*Identity whether the SCP can be solved on* $[t_s, t_e]$ *or not.*
5: $(flag, C) = TwI-EGSC(\mathcal{G}, [t_s, t_e])$)
6: **if** $flag = 0$ **then**
7: **continue**
8: **end if**
9: $P_{\mathcal{T}} \leftarrow \phi$
10: **for all** $(v,u) \in C \times N(v)$ **do**
11: find the minimum cost path $p(v,u)$ from v to u .
12: $P_{\mathcal{T}} \leftarrow \{p(v,u)\}$
13: **end for**
14: */*Identity whether* $[t_s, t_e]$ *is a fully-covered* $\mathcal{T}_w$ *or not.*
15: $p(u,v) \in V^0 \times C$
16: **for all** $(u,v) \in V^0 \times C$ **do**
17: find the minimum cost path $p(u,v)$ from u to v .
18: **if** there do not exist a path between (u,v) **then**
19: $flag \leftarrow 0$
20: **break**
21: **end if**
22: $P_l \leftarrow \{p(u,v)\}$
23: **end for**
24: **if** $flag = 0$ **then**
25: **continue**
26: **end if**
27: */* Construct* $\mathcal{H}$ *from all the minimum cost paths.*
28: $P = P_l \cup P_{\mathcal{T}}$
29: **for all** $p \in P$ **do**
30: **if** $e \in p$ **then**
31: $\mathcal{H} \leftarrow \mathcal{H} + e$
32: **end if**
33: **end for**
34: */* Update the optimal* $\mathcal{H}_m$ *if necessary.*
35: $c(\mathcal{H}) = \sum_{e \in \mathcal{H}} c(e)$
36: **if** $c(\mathcal{H}) < c(\mathcal{H}_m)$ **then**
37: $\mathcal{H}_m \leftarrow \mathcal{H}$
38: $c(\mathcal{H}_m) \leftarrow c(\mathcal{H})$
39: $\mathcal{H} \leftarrow \phi$
40: **end if**
41: **end for**
42: **return** $\mathcal{H}_m$

V. Performance Evaluation

In this section, we evaluate our proposed topology control algorithm ($TC-FcTw$) renamed as *greedy* by comparing its performance with two variants of *TC-FcTw*, namely, *random* and *LP* in which *TwI-EGSC* is replaced

by other classical set cover algorithms respectively, i.e., *randomized algorithm* and *linear programming*. We implement these algorithms on *Matlab* simulation platform. The time-evolving networks are either generated from a random graph or extracted from *Cabspotting* and *Infocom05* traces. In addition, we take the following metrics for performance evaluation:

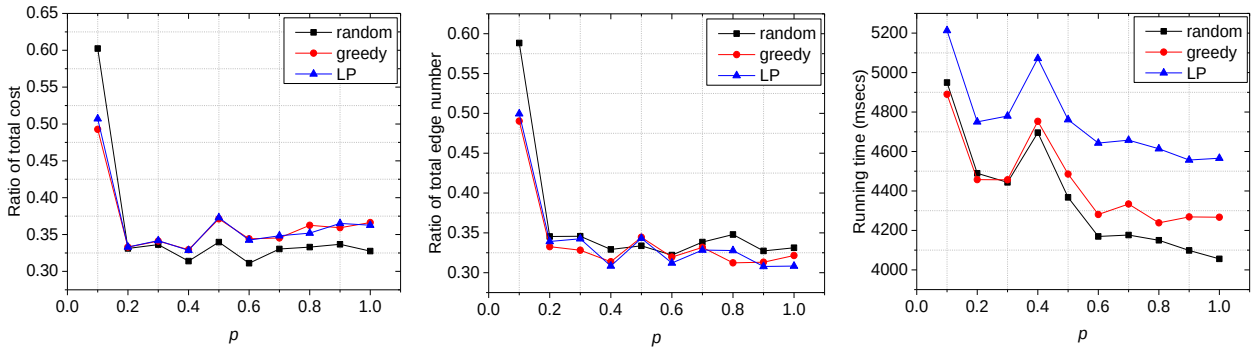
- **Total Cost:** The total cost of the optimal subgraph $\mathcal{H}_m$, i.e., $c(\mathcal{H}_m) = \sum_{e \in \mathcal{H}_m} c(e)$
- **Total Number of Edges:** The total number of edges in $\mathcal{H}_m$, i.e., $|\mathcal{H}_m|.|\mathcal{G}|$ denotes the number of edges of graph $\mathcal{G}$.
- **Running Time:** The total time to generate the optimal topology $\mathcal{H}_m$.

For all the simulations, we repeat the experiment, record the average values of each metric and compare the performances of topology control under different set cover algorithms.

A. Simulations on Random Networks

We first generate a series of adjacent matrices to represent each individual static graphs G^t which forms a time-evolving DTN. Here, we build a network with 10 nodes (n = 10) lasting over 10 time slots ($T = 10$). For any pair of nodes v_i, v_j in each time slot t, we insert an edge $\overrightarrow{v_i v_j}$ with a fixed probability p. The cost of edge is randomly taken from 1 to 5. Then, we convert G^t into a space-time graph $\mathcal{G}$ with $n(T+1)$ nodes. In addition, the links which connect the same nodes between successive time slots always exist and their costs are randomly taken from 0 to 1. Last, *greedy*, *random*, *LP* can act upon the same $\mathcal{G}$ for topology control.

To observe the performance of three algorithms under different network density, we increase p from 0.1 to 1.0. Small value of p result in a sparse DTN, and p = 1.0 yields a sequence of complete graphs in each time slot. Given a value of p, we generate 30 random graphs and output the average performance among them. Figs 5(a) and 5(b) present the ratio between the total cost/edges of the optimal subgraph $\mathcal{H}_m$ and that of the original graph $\mathcal{G}$. This ratio reflects how much cost can be saved by exploiting various topology control algorithms. It is obvious that all the algorithms lead to a sparse $\mathcal{H}_m$ with lower cost level by removing unnecessary links while guarantee the connectivity. Even with the sparsest network (p = 0.1), *random* and other algorithms can save around 40/50 percent cost and 40/50 percent edges respectively. When p varies from 0.2 to 1.0, *random* slightly outperforms other algorithms in terms of total cost, but overall the ratio of cost/edges for all algorithms remains stable and under 37/35 percent, which indicates visible saving can be achieved in random networks. Fig. 5(c) shows the running time of algorithms. It reveals that all algorithms use less time with a denser graph since more links make it easier to locate a fully-covered time window. Compared with *greedy* and *random*, *LP* consumes more time due to its complexity. We also conduct experiments on networks with various node and time scales, which also yields the similar results and conclusions.



(a) $c(\mathcal{H}_m)/c(\mathcal{G})$ (b) $|\mathcal{H}_m|/|\mathcal{G}|$ (c)Running time

Fig. 5. Simulation results of topology control on random networks with different densities

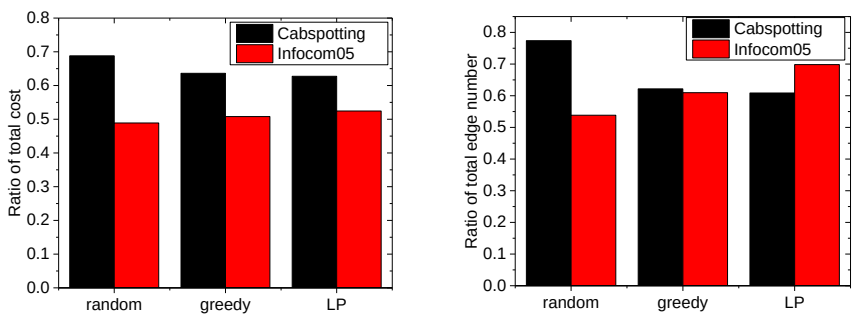


Fig. 6. Simulation results of topology control on realistic traces

B. Simulations on Realistic Traces

Previously, we have investigated and extracted some major characteristics from realistic mobility traces, which facilitates topology control on real-world networks. Taking this advantage, we evaluate our topology control algorithms on *Cabspotting and Infocom05* which represent *VANET* and *MSNs* respectively. The details of these data sets have been illustrated in Section III. In our simulations, for *Cabspotting*, 100 mobile nodes are randomly chosen from 536 taxis and we take one day as a time period and one hour as a time slot, for *Infocom05*, we only consider the 41 internal devices and take 12 hours as a time period and 30 minutes as a time slot. It is obvious that the time period of the tracing data is divided into 24 time slots. For each time slot *t*, if there exists a contact record in this slot, we add a corresponding link in G^t. Due to node scale, we randomly pick out a partial network with 10 nodes for one round, repeat 15 times and output the average results. Costs are generated as previously described. Some conclusions can be drawn from Fig. 6: 1) all algorithms can reduce the cost (around 35/50 percent); 2) Different algorithms may yield various performances upon different network forms, e.g. *random* has the least performance on *Cabspotting* while fits into *Infocom05*. This result is determined by the intrinsic characteristics of data sets. In *Cabspotting*, we take 100 taxis as sample capacity. These taxis are actually highly mobile nodes within a big area (San Francisco Bay Area), which means in each round, we indeed pick up 10 nodes with distinguished contact properties and incur significant randomness to the outcome resulting from *random* algorithm. The *random* has to pick more nodes and edges to maintain a connected topology since 10 nodes may have low contact density compared with the original sample capacity. In *Infocom05*, all the attendees meet in a conference site and maintain stable links between each other, which presents high-density and homogeneous contact property. That is why *random* significantly reduces the cost and performs almost the same compared with *greedy* and *LP*.

VI. Conclusions

We first reveal two major characteristics namely *periodicity* and *path redundancy* by quantitative analysis from realistic DTN traces, which substantially provides the feasibility and operability of topology control. We then study the topology control problem in DTN by modeling such time-evolving network as a space-time graph and connecting it with the classical set cover problem, which aims to build a sparse and least-cost topology that fulfills DTN routing between all pairs of nodes. Then, we propose a topology control algorithm that can significantly reduce the total cost of network while guarantee the connectivity. Evaluation results from random networks and realistic DTN traces demonstrate the efficiency of our algorithm.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (61171097).

References

[1] K. R. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. Sigcomm Proceedings of the Conference on Applications Technologies Architectures and Pro*, vol. 33, no. 4, pp. 27–34, 2003.

[2] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[3] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Efficient routing in intermittently connected mobile networks: The single-copy case," *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, pp. 63–76, Feb 2008.

[4] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication routing in dtns: A resource allocation approach," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 596–609, April 2010.

[5] A. Elwhishi, P. H. Ho, K. Naik, and B. Shihada, "Self-adaptive contention aware routing protocol for intermittently connected mobile networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1422–1435, July 2013.

[6] E. Daly and M. Haahr, "Social network analysis for information flow in disconnected delay-tolerant manets," *IEEE Transactions on Mobile Computing*, vol. 8, no. 5, pp. 606–621, May 2009.

[7] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *IEEE Transactions on Mobile Computing*, vol. 10, no. 11, pp. 1576–1589, Nov 2011.

[8] W. Gao, Q. Li, B. Zhao, and G. Cao, "Social-aware multicast in disruption-tolerant networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1553–1566, Oct. 2012.

[9] K. Wei, D. Zeng, S. Guo, and K. Xu, "On social delay-tolerant networking: Aggregation, tie detection, and routing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1563–1573, 2014.

[10] X. Mingjun, W. Jie, and H. Liusheng, "Community aware opportunistic routing in mobile social networks," *IEEE Transactions on Computers*, vol. 63, no. 7, pp. 1682–1695, 2014.

[11] M. H. Eiza and Q. Ni, "An evolving graph-based reliable routing scheme for vanets," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 4, pp. 1493–1504, 2013.

[12] R. Rajaraman, "Topology control and routing in ad hoc networks: A survey," *Sigact News*, vol. 33, no. 2, pp. 60–73, 2002.

[13] W. L. Shen, C. S. Chen, K. C. J. Lin, and K. Hua, "Autonomous mobile mesh networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 2, pp. 364–376, 2014.

[14] R. Wattenhofer, L. Li, P. Bahl, and Y. M. Wang, "Distributed topology control for power efficient operation in multihop wireless ad hoc networks," in *Proc. INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2001, pp. 1388–1397.

[15] Q. Guan, F. R. Yu, S. Jiang, and G. Wei, "Prediction-based topology control and routing in cognitive radio mobile ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 9, pp. 4443–4452, 2010.

[16] M. Huang, S. Chen, Y. Zhu, and Y. Wang, "Topology control for time evolving and predictable delay-tolerant networks," in *Proc. IEEE 8th International Conference on Mobile Ad hoc and Sensor Systems*, 2011, pp. 82–91.

[17] F. Li, S. Chen, M. Huang, Z. Yin, C. Zhang, and Y. Wang, "Reliable topology design in time-evolving delay-tolerant networks with unreliable links," *IEEE Transactions on Mobile Computing*, vol. 14, p. 1, 2015.

[18] Altman, A. P. Azad, T. Basar, and F. D. Pellegrini, "Combined optimal control of activation and transmission in delay-tolerant networks," *IEEE/ACM Transactions on Networking*, vol. 21, no. 2, pp. 482–494, 2013.

[19] X. Zhang and G. Cao, "Transient community detection and its application to data forwarding in delay tolerant networks," in *Proc. 21st IEEE International Conference on Network Protocols*, 2013, pp. 1–10.

[20] P. Caravelli, Y. Wei, D. Subak, L. Singh, and J. Mann, "Understanding evolving group structures in time-varying networks," in *Proc. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2013, pp. 142–148.

[21] A. Ferreira, A. Goldman, and J. Monteiro, "Performance evaluation of routing protocols for manets with known connectivity patterns using evolving graphs," *Wireless Networks*, vol. 16, no. 3, pp. 627–640, 2010.

[22] S. Merugu, M. H. Ammar, and E. W. Zegura, "Routing in space and time in networks with predictable mobility," *Georgia Institute of Technology*, 2004.

[23] S. Agathos and E. Papapetrou, "On the set cover problem for broadcasting in wireless ad hoc networks," *Communications Letters IEEE*, vol. 17, no. 11, pp. 2192–2195, 2013.

[24] Q. Wang, K. Xu, G. Takahara, and H. Hassanein, "Transactions papers device placement for heterogeneous wireless sensor networks: Minimum cost with lifetime constraints," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2444–2453, July 2007.

[25] T. H. Cormen, *Introduction to Algorithms, Second Edition.*, The MIT Press, 2001.

[26] T. Grossman and A. Wool, "Computational experience with approximation algorithms for the set covering problem," *European Journal of Operational Research*, vol. 101, no. 1, pp. 81-92, 1997.

Xu-Yan Bao received his B.E. degree in Telecommunication Engineering and Management from Beijing University of Posts and Telecommunications, Beijing, China, in 2011. He is now a doctoral researcher at ICN&CAD lab. His research fields include Ad-hoc networks, delay tolerant networks, topics including routing & caching design, buffer management, topology analysis and mobility modeling; and network virtualization in future networks, topics including Software Defined Network, virtualized source management and network integration.

Xiao-Jin Zhou received her Bachelor's degree from Beijing University of Posts and Telecommunications (BUPT), Beijing, China in 2014. Since 2014, she has been pursuing graduate studies at the Beijing University of Posts and Telecommunications. Her research interests include delay tolerant network, mobile communication and device-to-device communication systems.

Xiao-Juan Wang received the B.S. degree in mathematics from Hebei Normal University in 2008 and M.S. degree in computer application from Beijing technology and business University in 2012. She is currently working toward the Ph.D. degree with Beijing University of Posts and Telecommunications (BUPT), China. Her research interests include energy-efficient resource management in OFDM wireless networks, coverage and capacity optimization in heterogeneous networks, and cognitive radio networks.

Yong Zhang has been an associate professor in School of Electronic Engineering at the Beijing University of Posts and Telecommunications, China. He holds a Ph.D. degree in Beijing University of Posts and Telecommunications. His research interests include self-organizing network, mobile communication, and cognitive network.

Mei Song received the B.E. and M.E. degrees from Tianjin University, China. She is currently a Professor in Beijing University of Posts and Telecommunications (BUPT), China, and also the Vice Dean of Electronic Engineering Department. Her current research interests include resource allocation and mobility management in heterogeneous and cognitive network, cooperative communications, and other advanced technology in future communications.