# Resource Efficient Urban Delay/disruptive Tolerant Networks

**Xiangfa Guo**

**NATIONAL UNIVERSITY OF SINGAPORE**

**2014**

# Resource Efficient Urban Delay/disruptive Tolerant Networks

**Xiangfa Guo**

(B. Engineering, M. Computing )

**A THESIS SUBMITTED**

**FOR THE DEGREE OF DOCTOR OF PHILOSOPHY**

**DEPARTMENT OF COMPUTER SCIENCE**

**SCHOOL OF COMPUTING**

**NATIONAL UNIVERSITY OF SINGAPORE**

**2014**

# DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

---

Xiangfa Guo

April 29, 2014

## ACKNOWLEDGEMENTS

I first and foremost express my deep and sincere gratitude to my supervisor Prof. Chan Mun Choon. Without his inspiration and guidance, it would not be possible for me to finish my Ph.D journey. In the past five and half years, Prof. Chan taught me invaluable methodologies and critical thinking. His great enthusiasm for researching and teaching deeply impressed me. In many days of paper deadlines, we worked together overnight. I could not remember how many times he enlightened me when I got stuck in research. Prof. Chan's continuous encouragement and support energized and inspirited me when I was depressed. I am feeling lucky to learn to research under Prof. Chan. I am most grateful to his nutrition for both my research skills and personality.

I would like to express my sincere thanks to my dissertation committee members, Prof. Wei Tsang Ooi, and Prof. Seth Gilbert. They have spent valuable time reviewing my graduate research report, thesis proposal and dissertation, and giving me insightful comments and suggestions.

I appreciate the culture of Communication and Internet Research Lab (CIRL) built by Prof. Chan and Prof. Ananda, where everyone is motivated, and all lab mates are selfless cooperative. Throughout my stay in CIRL, I have made many wonderful friends. Their friendship, encouragement, and interesting discussions have really help with my research and life. Specially, I would like to thank the following friends in CIRL: Bhojan Anand, Binbin, Chaodong, Chengwen, Faicheong, Girisha, Hande, Hwee-xian, Liu Xiao, Kartik, Manjunath, Mingze, Mobashir, Mostafa, Nabha, Nimantha, Prashanth, Shao Tao, Xiaofeng, Xiuchao, Yuda, and Ziling. Especially to Binbin, he helped me on neighbor discovery work with insightful discussions and strong encouragement. I also give thanks to my teachers and friends in NUS, Chang Ee-Chien, Dong Xinshu, EeAkkihebbal L. Ananda, He Dongfang, He Jinghui, Hsu Wynne, Hugh Anderson, Li-Shiuan Peh, Li Pei, Li XiaoLei, Li Yu, Liu Xuan, Long Le, Manoranjan Mohanty, Liu Xiaomin, Qi Dawei, Roland Yap, Thuy Ngoc Le, Shi Lei, Wang SuYun, Wang Chun-dong, Wang Wei, Wang Qiang, Wei Chuangzhen, Wu Weijie, Zhang Zhengchen, Zhao Zhenwei, Zhu Minhui. I thank all of my teachers and friends. They give me good time during my Ph.D journey.

I thank my parents and parents-in-laws. They give me and my wife more understanding and support than we could ever expect. I thank my wife. She gives me and our family endless love and patience. *This dissertation is dedicated to my wife.*

# Contents

# Summary

This thesis presents systematic research on Delay/disruptive Tolerant Network (DTN), especially for urban DTNs, e.g., opportunistic networks consisted of smart phones, and Vehicular Ad hoc NETworks (VANET) of vehicles. These DTNs are the emerging class of new communication paradigms for mobile nodes. With an emphasis on resource efficiency, we present three systematic solutions, from 'how to discover communication chances', to 'how to route messages from a source node to a destination node', to 'how to aggregate data via DTN communication'. The three questions share the common features with layers in the Internet stacks, namely, link layer, network and transport layer, and application layer.

First, we discuss the solution for discovering nearby communicable neighbors within DTN transmission range, which is an essential building block for all applications relying on short range wireless communication. We firstly analytically review existing solutions and investigate the requirements on energy consumption and neighbor discovery latency. Then, we discuss our solution $R2$, a synchronous neighbor discovery algorithm based on Wi-Fi access points. It is energy efficient and has short discovery latency. Evaluations show that R2 can discover 90% of neighbors within 50 seconds at 1% duty cycle, with the number of neighbors from 10 to 60.

Next, we discuss message routing algorithms, which is an indispensable part of DTN communication. Our solution *Plankton* was motivated by an observation that most existing DTN connection prediction algorithms were not reliable. We propose a connection prediction algorithm based on short-term bursty contacts association and long-term contacts association. This significantly improves the reliance of connection prediction. By Plankton, a node controls message replica quota by the connection predictions, which serves well the goal of the resource efficiency. Evaluations show we can save 14% to 88% of message replicas compared to existing DTN routing solutions.

Finally, we present a solution for data aggregation over DTN communication, with the application of ubiquitous sensor data aggregation. We propose a novel concept on how propagated information may have changed, namely *change awareness*. Based on the concept of change awareness, we propose an algorithm for computing a minimum set of nodes, via which a server can collect a snapshot of the all nodes information. Extensive evaluations show that a server can obtain a global snapshot on nodes' updates by collecting information from only 15% to 25% nodes.

In a summary, we systematically investigate DTNs with the focus on resource efficiency. With the growing penetration of smart mobile commutative devices, the solutions in this thesis can contribute to the growing DTN applications.

**Keywords:** Delay/disruptive Tolerant Network; resource efficiency; DTN routing; Information Collection; Neighbor discovery of mobile nodes.

# List of Publications

[1] Xiang Fa Guo and Mun Choon Chan. Plankton: An efficient DTN routing algorithm. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*, pages 550–558. IEEE, 2013.

[2] Xiang Fa Guo and Mun Choon Chan. Change awareness in opportunistic networks. In *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*, pages 365–373. IEEE, 2013.

[3] Xiang Fa Guo and Mun Choon Chan. On the utility of overhearing in DTN. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2014 9th International Conference on*. IEEE, 2014.

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 The background of Delay/disruptive Tolerant Networks

### 1.1.1 Brief history of DTNs and the evolution

The idea of Delay/disruptive Tolerant Network (DTN) can be traced back to 1970s, when researchers began developing technology for routing between computers having non-fixed locations. Intensive research on DTNs started from the beginning of 2000s. In 2002 the Delay-Tolerant Networking Research Group (DTNRG)[1] was formed with the sponsor of Internet Research Task Force. The concept of delay tolerance feature was introduced by the inter-planet communication because it has extremely high delays compared to normal terrestrial communication. Long delay is one main challenge that deep space communication protocols need to handle.

DTNs were later widely referred to the terrestrial networks that consist of intermittently commutable mobile devices. In 2003, Kevin Fall proposed 'architecture for delay tolerant network' [1]. Following this proposal, the research scope and interest on DTNs was significantly expanded. The growth of DTN research is inspired by two factors. First, a great potential is seen in the urban environment DTNs, as more and more mobile devices have been equipped with wireless chipsets, e.g., vehicles and smart phones. Smart phones are becoming lighter and smaller, yet providing more computation and

[1]http:www.dtnrg.org, checked on 30 Aug 2014

communication capability, and more sensors, which leads to the prosperity of mobile applications. The penetration of smart phones makes the pervasive computation and communication potentially feasible. Second, the challenging features of DTNs attract more research interest. The challenge has multiple folds, which include the power constraints, the neighbor discovery for mobile devices, and fleeting and unplanned connections. For example, how to quickly and efficiently detect mobile phones within wireless transmission range? How to efficiently route messages and collect information via intermittent connections?

More recently, we see new wireless communication techniques implementation in smart phones. 'Wi-Fi Direct' [2] and 'Bluetooth low energy' [3] are probably the most important two techniques. 'Wi-Fi Direct' makes phone-to-phone short range wireless communication easy to use. A user only needs to press buttons on a phone and starts direct Wi-Fi communication with another phone without involving wireless access points. 'Bluetooth low energy (BLE)' saves power for phones. BLE targets at power efficiency by fast switching on/off wireless chipsets, packet length restriction and low peak power consumption. With these ready-to-use techniques, we see increasing applications in mobile phones that exploit locality and short range communication, as discussed in section 1.1.2. Based on the contexts, commonly seen DTNs are listed in Figure 1.1. The Internet provides reliable information access to most users at most time. A DTN provides a communication channel when the Internet is not available (e.g., in extreme networks) or is available but treated as a second choice (e.g., in Pocket Switch Networks[4]) due to the concern of energy or financial cost. In order to be easily accessed, data from DTNs usually needs to be presented in the Internet, which connects to different types of DTNs. In Figure 1.1, five types of DTNs were illustrated. (i) Deep space network provides communication for interplanetary spacecraft and earth-orbiting missions. (ii) Land challenging/extreme network provides information communication for environment where the Internet or cellular network is unavailable. For example, the mine communication and the wildlife tracking. (iii) Pocket switch network is the network communication among smart phones, wearable devices, and other communicable de-

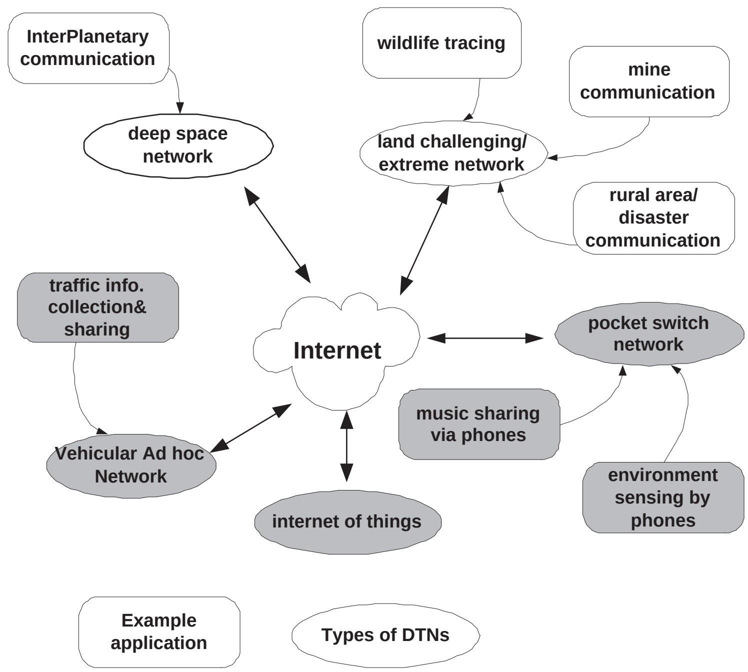



Figure 1.1: Different types/applications of DTNs. The bottom marked ones are urban DTNs, which are the focus of this thesis.

vices. Many potential applications operating on smart phones can use this type of networks to communicate information. (iv) Communication for the Internet of Things can exchange information between two things via mobile nodes. (v) Vehicular ad hoc network helps exchange messages between moving vehicles, which has many applications in intelligent traffic systems.

This thesis presents the solutions for DTNs, especially for the DTNs in city areas, e.g., vehicular ad hoc network and pocket switch network of smart phones, as the ones at the bottom of Figure 1.1. We call these types of DTNs as urban DTNs, as these types of DTNs are more likely to appear in cities comparing to other types of DTNs. Urban DTNs will be further exploited as more and more smart phones and vehicles gain short range wireless communication capability.

| Data Size | LTE ($\mu J/bit$) | 3G ($\mu J/bit$) | Wi-Fi($\mu J/bit$) |
|---|---|---|---|
| 10KB | 170 | 100 | 6 |
| 10MB | 0.4 | 4 | 0.1 |

Table 1.1: Energy consumption of wireless communication.

### 1.1.2 DTN characteristics and potential applications

**DTN characteristics: disadvantages and advantages**

In DTNs, a node moves and communicates with intermittently encountered[2] nodes. This naturally generates much longer delays compared with the message transmission via continuous end-to-end connections. In the Inter-Planetary Network, the delay can be computed in advance, as the satellite movement and transmission chances are scheduled. In many other types of DTNs, e.g., PSNs and VANETs, it is not easy to reliably estimate delays, as the communication chances are not known in advance.

For example, a node ‘*S*’ needs to transmit a message for a node ‘*D*’. ‘*S*’ forwards the message to nodes it encounters, which we call relay nodes. Then, the relay nodes carry the message and forward it to nodes they encounter. The message delivery delay depends on the earliest time when a relay node transmits the message to the destination ‘*D*’. The delivery delay is usually long.

Even given a long delay, the message delivery in DTNs is often not guaranteed. In the above example, if no relay node encounters the destination ‘*D*’, then the message cannot be delivered to the destination. In the reality, a message is useful to the destination only when it can be delivered within a valid time. For example, it is useless to deliver an advertisement message for an exhibition after the exhibition finishes. Thus, each message has a valid delivery time, i.e., time to live, which depends on the context of applications. In short, DTNs cannot guarantee message delivery.

Though DTNs have unwanted long delays and cannot guarantee message delivery, they have desirous advantages as follows.

[2]By saying that two nodes encounter, contact or connect, we mean that the two nodes are close that they can directly exchange messages with each other.

**Energy efficient** Mobile nodes usually have two general means in wireless communication. The first one is the long distance communication (e.g., cellular networks), which is close to real-time with negligible delay. The second one is short distance wireless transmission (e.g., Wi-Fi, Bluetooth), which is used by DTNs. Regarding to energy efficiency per bit, the short distance wireless transmission, has large advantages over long distance communication. The energy efficiency for download links for Wi-Fi, 3G, and LTE is compared in Table 1.1 [5]. It shows that, compared with LTE, Wi-Fi can save up to 95% energy when downloading small size data and can save up to 55% energy when downloading large size data. More experiments on power savings of download and upload by Wi-Fi, Bluetooth, and cellular network can be found in [6, 7].

Short range wireless communication clearly has advantages in term of energy consumption per bit. However, in practice, good designs are still required to reserve the advantages, e.g., low duty cycle, fast neighbor discovery, the control of transmission hops.

**Infrastructureless and Robust** DTN communication occurs between mobile nodes within communication range and thus is de-centralized and independent from infrastructures. This makes it more practical to carry applications that require mostly local data communication. Nowadays, a large size of data are generated and to be shared by mobile devices. This generates large data communication load and degrade infrastructure communication performance. DTN communication is a good candidate approach to offload data traffic from infrastructure to distributed and infrastructureless way. For example, mobile nodes can first locally aggregate data through short range wireless communication before infrastructure communication, and this can significantly reduce infrastructure communication. In urban DTNs, infrastructure communication, e.g., 3G and LTE, is performed when a requirement is beyond the capability of DTNs. The independence from infrastructure makes DTN robust. The fact that multiple copies and diverse message paths for one source-destination pair communication also makes DTN robust.

DTNs are robust but it is not free from attacks. Choo et al. conclude that the robustness of DTNs message routing become weak as the number of hops increases [8].

**Financial Saving** Cellular network data communication is usually expensive. The financial cost is higher as more data communication is conducted. Many application data, e.g., pictures, video, continuous sense results, are large, thus, the expense on cellular communication rises sharply if all data go through cellular communication. Comparatively, DTN communication can be free when the motivation scheme is properly designed [9]. Smart phone users usually have a cap in cellular data communication per month. When delay tolerant data can be offloaded to DTN communication, users can use more cellular communication allowance to communicate the data requiring shorter delays. This ultimately saves users financial cost.

**Potential DTN applications: to exploit the advantages**

DTNs naturally have the long message delivery delay and low message delivery ratio, but they are robust, suitable for ubiquitous communication and computation, and resource efficient. These characteristics fit some types of applications. Different applications have different tolerance to the communication reliability and delays, as shown in Figure 1.2. The applications on the right part potentially fit DTNs. Considering the availability of cellular network infrastructures, we get Figure 1.3, which illustrates the space for DTN applications.

In the literature, we see DTN applications from the environment having no communication infrastructure to the environment having communication infrastructure. Some applications have been implemented and tested. The *ZebraNet*[10] by Princeton University exploits energy efficient hardware, lightweight operating system and communica-

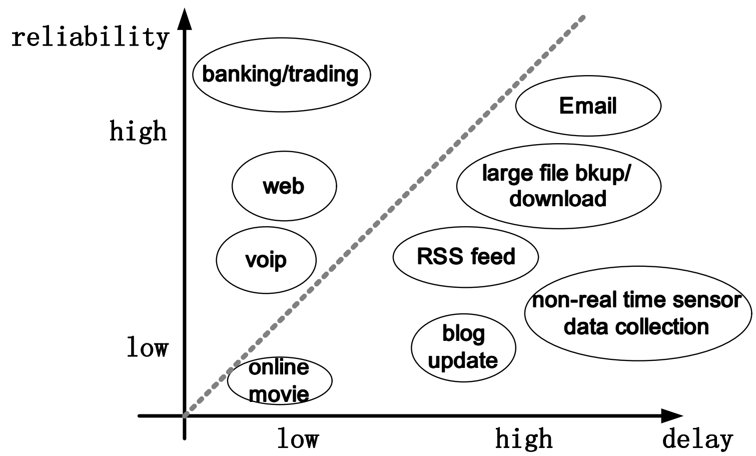



Figure 1.2: The reliability and delay requirement of different applications

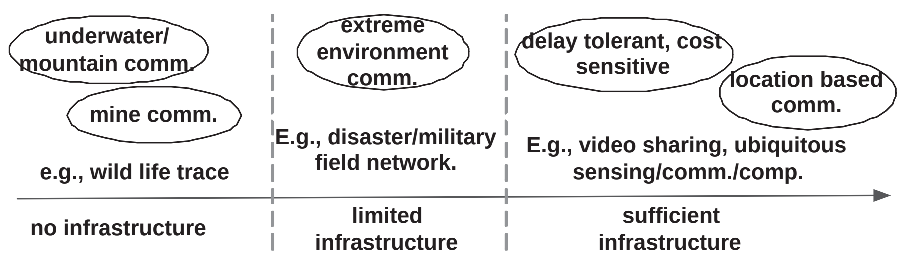



Figure 1.3: The space of DTN applications

tion protocols. The devices mounted on wild animals use intermittent communication chances to exchange messages, and relay information to deployed base station. The *CarTel* [11] is a mobile sensor computing system designed to collect, process, deliver, and visualize data from sensors located on mobile units such as automobiles. It has been used to analyze commute information and metropolitan Wi-Fi deployments. The *Haggle* [3] project proposed seamless networking for mobile applications [12] and PSN [4]. The applications in extreme environment include 'underground tunnel communication' [13], 'networks in military or disaster fields' [14, 15]. Some DTN applications adapt the Internet applications onto DTNs, e.g., 'PeopleNet' [16], 'HTTP over DTNs' [17], 'Email over DTNs' [18], 'Disruption tolerant Shell' [19], new distribution over DTN in urban area [20]. Some vehicular communication based DTN applications, e.g., MobTorrent [21] allows vehicles to get high speed transmission to the Internet, and

[3]http://www.haggleproject.org/, checked on 30 Aug 2014

| Application | Description | Installs | Rate |
|---|---|---|---|
| Badoo [30] | Chatting, dating with people nearby | $1 \times 10^7 \sim 5 \times 10^7$ | 4.5 |
| Skout [26] | Discovering, meeting people around | $1 \times 10^7 \sim 5 \times 10^7$ | 4.2 |
| Circles [31] | Finding people nearby | $5 \times 10^6 \sim 1 \times 10^7$ | 4.3 |
| Sonar [32] | Connect with like-minded people nearby | $1 \times 10^6 \sim 5 \times 10^6$ | 4.0 |
| Waze[4] [33] | Community-driven traffic information | $1 \times 10^7 \sim 5 \times 10^7$ | 4.6 |
| Groupon [34] | Finding local deals and discounts | $1 \times 10^7 \sim 5 \times 10^7$ | 4.5 |
| Foursquare [35] | Find interesting places nearby | $1 \times 10^7 \sim 5 \times 10^7$ | 4.2 |

Table 1.2: Examples of Social-Proximity Applications - Android Apps (As of 1 May 2014)

'web search from a bus' [22].

More recently, DTN applications on smart phones arouse more research interest. Hui et al. propose to share media files between passengers on urban transportation via the carried mobile phones [4]. The BikeNet [23] is a mobile sensing system to map the cyclist experience. They exploit Wi-Fi access point for delay-tolerant data transmission, and use cellular networks to transmit real-time data. Han et al. in [24] propose to offload mobile phone data from cellular network load to opportunistic communication. Many emerging new businesses exploit local communication and locality, e.g., Phewtick [25], Skout [26], Bump [27], and Momo [28]. Though most of them still rely on the cellular networks to transmit data, DTN techniques definitely prompt them by reducing or bypassing cellular data communication, which is definitely more attractive to users. Table 1.1.2 summarizes some social-proximity applications. More DTN applications can be found in the book [29].

The literature of DTN applications shows a clear evolution path from extreme environments (e.g., ZebraNet, disaster field communication) to the urbane area communication (e.g., mobile proximity applications) for phones and vehicles. Though cellular networks evolve fast from 2G, 3G to LTE these years, the data transmission requirement by mobile devices also expand in a fast speed. This requires the offload of data communication from cellular networks to distributive local wireless communication.

The journey of seeking DTN applications is not smooth. We are seeing small scale DTN experiments and more potential delay/disruptive tolerant network applications,

[4]Waze has been acquired by Google on June 11th 2013 for 1.1 billion dollars.

but the ‘killer applications’ are still to be found [36][37].

## 1.2 The contribution of this thesis

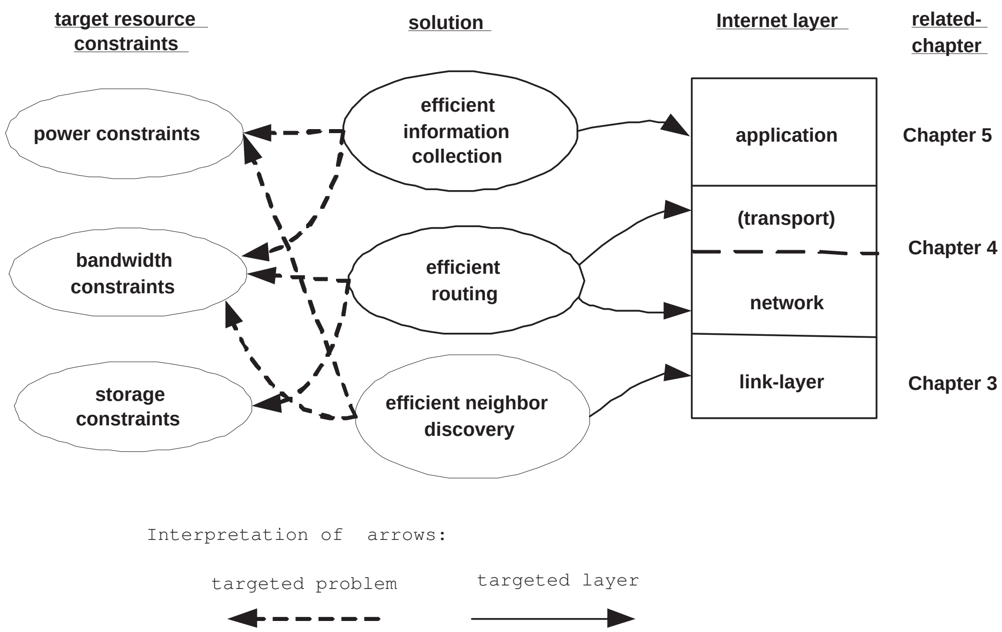



Figure 1.4: Overview of the thesis: the target problems, solutions, the stack layers, and relative chapters

DTN research shares a similar stack as the Internet. One difference is that the transport layer has less content (due to lack of end-2-end connection) and is well mixed with network layer. A DTN routing solution considers both the message path selection and message congestion control. Thus, a dash line is used to separate transport layer and network layer in the stack in Figure 1.4. Figure 1.4 describes our contributions, and the targeted problems and related layers. In details, this thesis contributes in three aspects.

- **Link-layer: Access Point based neighbor discovery protocol (*R2*).** Link layer is the physical and logical network component used to locally interconnect nodes in a network. Link lay protocols consist of a suite of methods and standards that operate only between one-hop neighboring nodes of a local network segment, e.g., media access control. We are interested in mobile nodes neighbor discovery protocols, which is one of the most important topic in the link layer. We identify the potential design space as well as the limitation of existing neighbor discovery protocols. Specifically, while a locally synchronized protocol can potentially operate at a low duty cycle while achieving short discovery delays, its synchronous nature makes it inefficient in a dense neighborhood. We propose a scheme R2, a locally synchronized protocol based on Wi-Fi AP beacons. The scalable design makes it work in dense mobile phone environment, which is often seen in urban environments. We also investigate the synchronization issue when a large number of APs are present. R2 saves power and exploits more communication chances for phones. This sets up a solid corner stone for all applications that rely on the communication between mobile nodes.

- **Network layer and transport layer: Resource efficient routing algorithm (*Plankton*).** The network layer mainly does packet forwarding including routing through intermediate routers in the Internet. Transport layer is responsible for data stream support, reliability, and flow control . In DTNs, the two layers are often mixed. DTN routing protocols provide message routing, reliability, and flow control. We propose a novel DTN routing algorithm named Plankton. Plankton utilizes contact probability estimation and replica control to reduce overhead to reduce resource usage while enhance routing performance. Plankton has two major contributions to DTN routing methodology. (i) It proposes a framework for connection prediction and predicts connections via both short-term bursty contacts and long-term, association based contact pattern. (ii) It controls message replica quotas based on estimated contact probabilities and delivery probabilities. The two techniques can shed light for future research on the measurement of connection prediction reliance and its utility in message replica control.

- **Application layer: Resource efficient data aggregation scheme (*change awareness based data aggregation*).** Application layer uses underlying network layers to provide services to end users. We are interested in aggregating data via DTN connections to reduce infrastructure communication. By intermittent short range connections, nodes can exchange information. As nodes' connections are heterogeneous, some nodes can grasp more information, while others only can grasp less information. Based on this observation, we propose a novel concept of *change awareness*. By change awareness, informative nodes can be computed given dynamic connections. A node is more informative when it can know more other nodes' data updates, e.g., the sense data. Based on the computation, we propose an algorithm that computes a minimum set of nodes whose information union can form a global snapshot. A server can collect global data by communicating with only these informative nodes.

## 1.3 The organization of this thesis

We discuss mobile nodes neighbor discovery solution in Chapter 3. Following this, our solutions on resource efficient DTN routing are presented in Chapter 4. In Chapter 5, we describe change awareness and its application on resource efficient data aggregation via DTNs. The conclusion and future work are presented in Chapter 6.

# Chapter 2

# Overview of DTN Research

## 2.1 Resource constraints in DTNs

Researches in DTNs share many common interesting topics with the Internet research community. Figure2.1 illustrates DTN layers and relevant research topics. DTNs have attracted a great research efforts, and plentiful impressive proposals have been raised. However, practical DTN solutions to mobile node neighbor discovery, efficient message routing, and information collection still face substantial challenges. Main challenges come from resource constraints as bellows.

- **Limited power capability.** Limited battery capability is a known major constraint of mobile devices. Research in [38] investigates the power constraints in smart phones. By their measurement, continuous Wi-Fi transmission can drain one normal mobile phone battery in six hours. The battery can last 27 hours in a regular use mode; the battery can last 21 hours in the business mode where more cellular communication exists; If more power consumption by today mobile applications is considered, the battery life likely becomes shorter.

- **Limited transmission capability.** DTN transmission capability is limited by two factors. First, nodes are mobile, and therefore their wireless bandwidth is often not as good as static nodes. Second, nodes' connection durations are usually

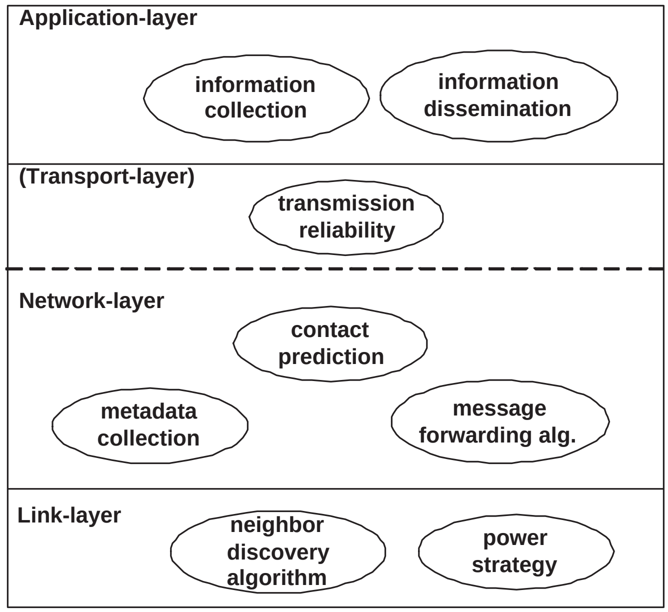



Figure 2.1: DTN stacks

short. For example, the average connection length is 45s in MIT Reality trace [39], 22s for roller net trace [40], and 50s in SF taxi trace (with 100 meters wireless transmission range). Many contacts have short contact duration as shown in Figure 3.2 (a). If we exclude the neighbor discovery delay, the usable contact length is even shorter.

- **Limited buffer capability.** Mobile devices are buffer constrained, especially when multimedia data are stored. The use of external storage can abate the constraints, but it also introduces extra writing/reading delays and power consumption.

The three constraints are not equally applicable to all nodes in DTNs. For example, nodes mounted on vehicles are more limited by transmission capability, and are less confined in buffer and power capabilities. DTNs consisted of smart phones have all the three types of constraints.

## 2.2 Challenges in solving resource constraints

We illustrate the challenges by an example. A node '*S*' has a message to be sent to '*D*' via DTN connections. By the feature of DTNs, no end-to-end communication path exists most of the time, and the connections are not scheduled. We are interested in the following three challenging questions.

An immediate question is how to discover neighbor nodes within wireless (e.g., Wi-Fi, Bluetooth) transmission range. In practice, mobile nodes' clocks are usually asynchronous. Continuous wireless scan can promptly discover neighbors, but it drains too much power and hence is not applicable to devices with limited power supply, e.g., phones. To save power, a node needs to sleep its wireless interface as often as possible. Under this setting, we like to capture most connections such as the connections longer than some time period, say 50s. Under these requirements, energy efficient neighbor discovery is a challenge.

Next, after having discovered neighbors, how to route messages via intermittent connections? This equals to the follow question. When a node buffers messages for different destinations, how can the node select messages to forward to different neighbors? The trivial solution of message flooding does not generate good results especially when message traffic is high. To wisely select relays for messages, a node needs to predict future connections, which is non-trivial given that mobile nodes have non-scheduled movements. Another challenging part is to control the number of message replicas. A message to be transmitted needs to generate multiple replicas to reduce delivery latency and improve delivery chances. Messages for some source-destination pairs need more replicas, while messages for other pairs need fewer replicas. However, it is not easy to dynamically control the number of replicas as connections are not scheduled and the communication between nodes are intermittent and distributive. Hence, it is not easy to design a resource efficient routing algorithm, i.e., achieve good performance with fewer replicas.

Finally, how to collect or estimate up-to-date information from nodes in DTNs?

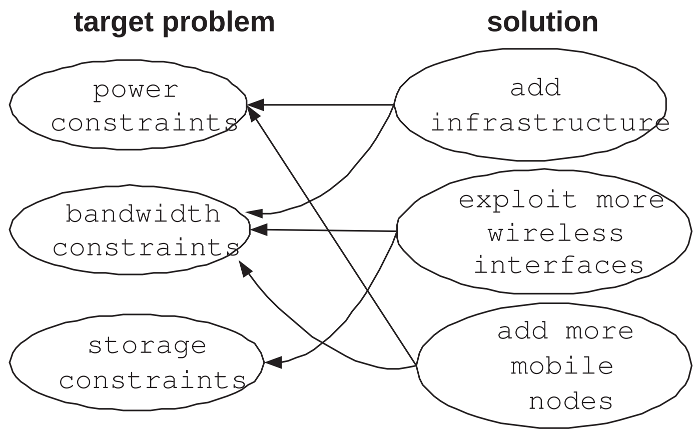



Figure 2.2: The architecture design for DTN resource constraints

Up-to-date information from nodes can facilitate applications, e.g., ubiquitous sensor date collection. When an application needs up-to-date data, it is unavoidable to utilize infrastructure (e.g., cellular networks) communication, since DTN in-band communication (i.e., short range communication between mobile nodes) is severely confined by nodes' movement. However, DTNs can help offload traffic from infrastructure communication by aggregating data via short distance communication. Thus, the target problem is how to exploit DTN in-band communication to minimize the use of cellular networks.

## 2.3 Researches for DTN resource constraints

The published solutions for resource constraints can be classified into two categories: (i) the solutions of architecture level and (ii) the solutions to problems of different DTN stack layers. Within the second category, we borrow Internet stacks to classify the solutions, as DTN research shares many common problems as the Internet research. The second set of work, i.e., solutions to DTN stack layers, is more relevant to this thesis. Here we shortly discuss related work. More lengthy discussion on related work can be found in later chapters.

### 2.3.1 Architecture level design

The architecture level solutions handle DTN resource constraints from three aspects, as shown in Figure 2.2. In MORA [41], Burns et al. propose to add in autonomous agents as mobile elements to improve DTN performance. Similar work in [42, 43] also exploits extra nodes to handle resource constraints and improve performances. Some other work add in static infrastructures to handle resource constraints. Authors in [44, 45, 20] add in infrastructure, e.g. throw-boxes or info-stations, to leverage message transmission under resource constraints. Multiple wireless interfaces are also proposed to lessen wireless transmission constraints. June et al. propose to exploit both short range and long range wireless transmission [46]. Harras et al. propose ParaNets[47] that uses satellite networks, cellular networks et al. to improve the performance of DTNs. Liberator et al. propose to add in second Bluetooth to DTN nodes [48]. Banerjee et al. propose to use Xtend Ratio in throw-box to detect the movement of mobile nodes and alert short range wireless transmission when necessary [49].

### 2.3.2 Link layer: mobile nodes neighbor discovery

Neighbor discovery in the scenario of DTNs is challenging, as the nodes' clocks are usually asynchronous and the durations when nodes are within transmission distance are often momentary. Continuous scan can quickly drain the power. To save power, energy constraint devices need to set wireless interface off as much as possible, e.g., 99% of time, and wake up in the rest time slices, e.g., 1% of time. The goal of neighbor discovery algorithms are to quickly discover neighbors with small power consumption. Existing solutions for neighbor discovery can be categorized into three groups, asynchronous probabilistic, asynchronous deterministic, and synchronous.

The asynchronous probabilistic algorithm e.g., 'birthday protocol'[50] requires nodes to wake up, transmit packets at a slot with a probability by power budget, e.g., 0.01. Each node independently switches on/off wireless interface. When two nearby nodes switch on wireless interfaces at the same time, then they can discover each other. This can provide good average performance due to birthday paradox[51]. However, the protocol does not guarantee the worst discovery latency.

The asynchronous deterministic protocols [52, 53, 54, 55, 56] aim to guarantee worst case performance by utilizing wake patterns. For example, the quorum based protocols [52, 53] put time slots in a $m \times m$ square, a node wakes up by one column and one row. SearchLight [56] extends the idea by probing row slots, which reduced both average delay and worse case delay. Another type of solutions exploit Chinese remainder theorem to schedule wake slots, such as Disco[54], and U-conn[55].

Synchronous neighbor discovery protocols [57, 58, 59] exploit available infrastructure to synchronize the wake-up clock. The infrastructure can be the Wi-Fi Access Points (AP) or the NTP server. Hao et al. utilize Wi-Fi beacons to synchronize wireless sensor networks [60]. Wi-Fi AP beacons are used to locally schedule mobile phone wake-up in the paper [57, 59]. Li et al. [58] propose globally synchronizing phones using Network Time Protocol (NTP) to achieve fast neighbor discovery.

Synchronization based neighbor discovery algorithms can significantly improve the chance of wake-up overlap, but they incur two types of overhead. The first one is the overhead of synchronization and the second one is the wireless collision when mobile neighbors are dense. Wireless collision only applies to synchronous neighbor discovery, not asynchronous neighbor discovery, as the asynchronous neighbor discovery naturally distributes phones' wake-up time into different time slots.

### 2.3.3 Network and transport layer: message routing and control

DTN routing schemes have been extensively investigated for more than 10 years, beginning when DTN was proposed by Kevin Fall [1]. Existing proposals can be classified into three categories by how much the contact prediction is exploited, as articulated by the survey papers [61, 62, 63]. The first type of proposal routes message based the prediction on future contacts. Some assume strong contact patterns, e.g., periodical contacts in RCM [64] and 'scalable routing'[65], and exponential distribution RAPID[66], Max-Contribution[67]. Others predict contacts by recent contacts, e.g., LER[68], 'FRESH'[69], 'LER under RWP'[70], 'spray and focus'[71], MaxProp[72], PRoPHET[73], E-PRoPHET[74]. As seen in more recent research, social network based routing uses social network analysis techniques to facilitate message forwarding decisions 'simBet'[75], 'simBetAge'[76], 'map contacts to social network'[77], 'label routing'[78], 'community detection'[79], 'Bubble Rap' [80], SMART[81], 'Friendship based routing'[82], 'social-feature based routing'[83].

The second type of proposal controls the message replica quota without using contact predictions, e.g., 'Simple Counting Protocol'[84], 'two-hop routing'[85, 86], 'spray and wait'[87], 'gossip based routing'[88].

The third type of proposals use both techniques for routing algorithms, most of which were proposed more recently. EBR in [89] extends 'spray and wait'[87] by splitting replica quota between nodes based on contact profiles. 'multi-period spraying' [90] uses different phases to control message replicas. A new phase is needed only when the previous phases cannot fulfil the routing goals.

All these works handle two fundamental questions: (1) how do nodes control the number of replicas? (2) how to select relays for message replicas? The control on the number of replicas and the placement of replicas have major impact on DTN routing performance.

We define the resource efficiency in neighbor discovery by the control of the number of message replicas. We ask the question: when two nodes have a sufficiently long connection, does one node duplicate all messages it has to the other one? If the answer is no, the algorithm is energy efficient. By this definition, the existing work in the second and the third category [84, 85, 86, 87, 88, 89, 90] are energy efficient routing algorithms.

Existing proposals on DTN routing have two major limits. Firstly, contact prediction accuracy is not clear. So far, we have seen little effort in investigating contact prediction accuracies. We tested typical prediction algorithms on real world traces, and the results are not robust or good enough (see Chapter 4.4 for details). Secondly, existing algorithms fail to efficiently integrate connection predictions and replica control schemes. Some proposals use both techniques, but they fail to efficiently adjust replica quotas by contact predictions. Under the two limitations, proposed algorithms failed to route messages in a resource efficient way. More literature review can be found in Section 4.2.

### 2.3.4 Application layer: data aggregation in DTNs

Lets first have a look at an example application. A server needs to collect or aggregate up-to-date sensor data from all mobile nodes in DTNs. These mobile nodes are intermittently connected, which constrains the information access between nodes within DTNs. In applications requiring up-to-date and global information, the process of aggregation inevitably uses infrastructure communication. We examine the problem: how to minimize the number of nodes using infrastructure by exploiting DTN communication? Practical solutions to data aggregation via DTNs has wide applications. First, it facilitates DTN metadata collection. For example it can provide practical solutions to collect ubiquitous sensor data. Smart phones nowadays have a rich set of sensors. They can participate in sensing and provide data to a server located in the Internet clouds, by which more users can access the sense data. As the sense data collection is long-term and frequent, a practical solution needs to be economical in phone power consumption and data transmission finance expense.

Most of the existing data aggregation and collection solutions are designed for wireless sensor networks, as discussed in the surveys [91, 92]. They either construct tree-based [93, 94] or cluster-based [95, 96] data diffusion protocols. These protocols are suitable to static networks having stable connection topology, not the dynamic ones. Some solutions were proposed for data dissemination and aggregation for VANET [97, 98]. The research on information flow in temporal graph and dynamic networks [99, 100] can be exploited for data aggregation in DTNs. Theoretical results on the minimization of infrastructure communication via mobile node to mobile node communication are articulated in [101]. The solution presented in Chapter 5 works along this direction.

### 2.3.5 Summary

Mobile nodes are usually limited in resource, namely, battery capability, transmission bandwidth, and storage sizes. DTN research community has been focusing on the resource efficiency in DTN solutions. These solutions provide valuable experience and insight into the features of DTN solutions and applications. However, we still see the gap in resource efficient DTN solutions. Inspired by this, in Chapter 3,4, and 5, systematic DTN solutions are proposed to accomplish resource efficient DTN solutions.

# Chapter 3

# Wi-Fi Access Point based neighbor discovery algorithm

Before any type of short range wireless communication between mobile nodes, a node must first detect mobile neighbors within communication range. This process is called neighbor discovery. Neighbor discovery forms a basis for all local node-to-node communication. In this chapter, we first provide an analytical review of existing solutions and then we investigate the gaps between existing solutions and the neighbor discovery requirements. Based on these analysis results, we present our Wi-Fi AP based solution, which provides short discovery delay and is scalable.

## 3.1 Background

Neighbor discovery is an indispensable building block for DTN applications. Also, the mobile nodes neighbor discovery process itself has many immediate applications. Examples of applications that rely on detecting nodes presence include wildlife tracking [102], firefighting [103], search and rescue operations [104], and social networking [105]. In fact, there are already a large number of mobile apps that are specially designed for enhancing interactions among mobile users close by, for example, Badoo [30], Skout [26] and Circles [31]. Each of these apps has more than 5 million downloads as of April 2014[1]. With all new features that are made possible through proximity, it will not be surprising if online social networks like Facebook [106] and LinkedIn [107] would extend their features to include local interactions.

A key technique to support such a service in an energy efficient manner is duty cycling, in which devices sleep as much as possible and wake up infrequently to discover other devices. While duty cycling saves energy, it also increases the discovery latency and decreases discovery ratio. The challenge is to balance the trade-off between energy consumption and discovery latency/ratio.

There have already been many existing discovery protocols [50, 52, 108, 54, 55, 56, 58, 59]. While all of them are designed for achieving low duty cycle and low discovery latency, there has not yet been a systematic investigation to examine their application in mobile phone settings. Somewhat surprisingly, the mobile phone neighbor discovery problem can be as challenging as (if not more challenging than) the well investigated sensor neighbor discovery problem [109].

First, for mobile phones, neighbor discovery is by itself not a standalone application but an enabling function that is required to support proximity-based applications. Hence, a practical solution for phone neighbor discovery should consume as little energy as possible since neighbor discovery needs to be 'always-on', independent of whether there is any neighbor. Existing protocols often evaluate their performance in the range of $1\%$ to $10\%$ power consumption. Instead, a close-up examination of the smart phone power profile reveals that for Wi-Fi based neighbor discovery, $1\%$ should be an upper limit instead of a lower limit. A duty cycle of less than $1\%$ is highly desirable. As an essential building block, if the neighbor discovery protocol consumes too much power, the lifetime of mobile nodes will be dramatically degraded, which further makes DTN applications less attractive.

Second, discovery latencies should be made as short as tens of seconds. This is because a large number of contacts are short (less than 30s) and the ability to utilize such short contacts significantly increases the number of reachable unique devices. A

[1] The number of downloads is checked on the web pages for the respective applications.

study of three different mobility traces (Section 3.2) shows that between $15\%$ to $75\%$ of the contacts are less than $30s$ and from $40\%$ to $80\%$ more unique devices can be reached if these short contacts are utilized. A short discovery latencies can help with bandwidth constraints with more communication time and message route choices by discovering connections with more nodes.

Motivated by the concrete settings of mobile phones, we conduct a review and compare the design space (asynchronous vs. synchronous, and for the latter, globally synchronized vs. locally synchronized) of neighbor discovery protocols. The key to find the best protocol turns out to be at the (constant) values of a few parameters. In general, different settings favor different protocols. Among all protocols analyzed, locally synchronized protocols are the most promising category for simultaneously achieving negligible energy footprint ($< 3\%$ of battery capacity) and low discovery delay ($< 30$ seconds). Intuitively, the advantage of locally synchronized protocols is that they use local time references (e.g., the timestamps in the management frames of nearby Wi-Fi APs) to synchronize neighbor phones, hence they incur significantly smaller ($< 1\%$) synchronization overhead than globally synchronized protocols.

One immediate concern for locally synchronized protocols is whether two neighbor phones can effectively select the same time reference, especially in urban areas where each phone can often be simultaneously covered by dozens of different APs. We show that this problem can be solved by a simple minimum-$k$ strategy with $k$ as small as 3. Here each phone simply uses a global hashing function to hash the MAC address of each AP it can hear. Then it syncs with the $k$ APs with the minimum hash values. We use analysis and simulation results to show that this simple strategy solves the 'too many APs' issue nicely.

We find that the more fundamental challenge for synchronous protocols is to deal with 'too many neighbors'. In urban settings, phones often cluster together (e.g., in classroom, bus terminal, or shops). With an increasing number of neighbors, the synchronous nature of the locally synchronized protocols causes excessive contention, which in turn increases the (re)discovery delay linearly or even super linearly. We provide measurement and evaluation results to illustrate the potential impact of this scalability issue. In contrast, asynchronous protocols scale well with the neighborhood size since they naturally spread the neighbor phones across different slots.

Guided by the above observations, we design R2, a mobile phone neighbor discovery protocol that delivers the promise by providing a discovery delay bound[2] smaller than $30s$ while consuming merely $2.5mW$ of power on average. Consider a typical $6Wh$ phone battery that needs to power a phone for $48hour$ (with the user alternating between active and standby modes), R2's total energy footprint through the $48hours$ translates to only $2\%$ of the battery capacity. Further, R2's performance is largely independent of the neighborhood size. Hence, it continues to provide low discovery delay even under very high network density.

R2 uses multiple unique designs to achieve its supreme performance: First, R2 uses the Wi-Fi radio interface to scan for nearby APs as local time references, but uses the Bluetooth 4.0 radio interface to discover neighbor phones. Second, R2 uses a Rendezvous-Reconnaissance mechanism to decouple the discovery delay from the neighborhood size to solve the scalability problem. Third, R2 lets a node adapt its operation based on the information it gathers, including its mode (discovery vs. maintenance, mobility vs. static) and the perceived neighborhood size.

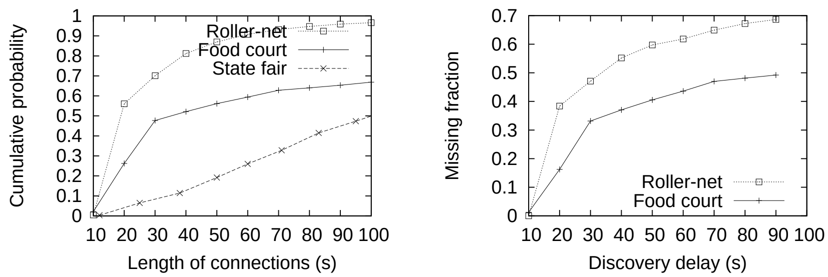
## 3.2 Motivation

Over the past decade or so, researchers have developed an impressive collection of neighbor discovery protocols. To examine different protocols, we first investigate the concrete settings of mobile phone neighbor discovery problem.

### 3.2.1 Discovery delay: How short is short enough?

A direct consequence of operating at a lower duty cycle is that it increases the discovery latency at least inversely linearly. In fact, for the many collection of asynchronous pro-

[2] Under certain protocol-specific assumptions that hold with high probability.




(a) (b)

Figure 3.1: (a) The distribution of contact lengths, and (b) the fraction of unique devices that would be missed under increasing discovery delay.

| Name | Scenario |
|---|---|
| Roller-net | 62 internal and 1039 external nodes, in a sports event |
| Food court | 2000 nodes, foodcourt, area of $1500m^2$, 3 hours |
| State fair | 19 nodes, North Carolina state fair, 3 hours |

Table 3.1: The traces used for neighbor discovery evaluation

tocols [52, 108, 54, 55, 56], as will be discussed in Section 3.3, their worst-case delay $D$ is at least as large as $\frac{T}{f^2}$ (see Table 3.3 for the interpretation of $T$ and $f$), where $T$ is the minimum duration for a phone to wake, transmit or receive 1 or 2 packets and go back to sleep. Even if $T = 1ms$ (which is arguably the limit for Wi-Fi standard), for $f = 0.3\%$, The delay will already be $D = \frac{1\times10^{-3}}{(3\times10^{-3})^2} > 100s$. Since the average discovery latency for such protocols is roughly half of the worst-case latency, the discovery ratio for contacts shorter than $50s$ will be far from the satisfactory (i.e., $< 30\%$).

To illustrate the practical delay requirement for mobile phone neighbor discovery, consider the scenario where two pedestrians walk in opposite directions towards each other at the walking speed of $1m/s$. Suppose their phone-to-phone transmission range is $30m$. The contact duration for their phones is hence only about $30s$.

We analyze three different mobility traces as shown in Table 3.1. The food court trace records information on customers' visit to Newton Hawker center in Singapore. The state fair trace[3] records customer's location update when they move in an open market place. The RollerNet trace [4] contains the connection/disconnection information between motes brought by people during a blading competition. In the analysis, we

[3]http://crawdad.cs.dartmouth.edu/ncsu/mobilitymodels
[4]http://crawdad.cs.dartmouth.edu/upmc/rollernet

| | Active power | Duty cycle for power budget | | | |
|---|---|---|---|---|---|
| | | $2.5mW$ 2% | $5mW$ 4% | $10mW$ 8% | $20mW$ 16% |
| BLE | $0.25W$ | 1% | 2% | 4% | 8% |
| WiFi | $1W$ | 0.25% | 0.5% | 1% | 2% |

Table 3.2: Comparison between Bluetooth Low Energy (BLE) and WiFi. Assume a battery capacity of $6Wh$ and a recharge cycle of $48h$. Hence 2% of energy footprint in the battery translates to $2\% \times 6Wh/48h = 2.5mW$ of power budget. This further translates to 1% of duty cycle for BLE and 0.25% of duty cycle for WiFi.

do not consider contacts that are below 10s. Fig. 3.2(a) shows the CDF of the contact duration for different mobility traces. The result clearly illustrates that a significant proportion of the contacts are short. In particular, about 70% of the contacts for the roller-net trace is between 10s and 30s. For the two traces (roller-net and food court) with a large number of nodes, Fig. 3.2(b) shows the missing ratio of unique discoverable devices under varying discovery delay. This further illustrates the importance of these short contacts: if the discovery delay is increased to $30s$, around 30% to 45% of unique devices would be missed. This translates to 40% to 80% more discoverable unique devices if phones can discover short contacts ($10s$ - $30s$) in addition to the longer ones ($>30s$).

### 3.2.2 Duty cycle: How low is low enough?

Existing efforts [50, 52, 108, 54, 55, 56, 58, 59] often evaluate their protocols with duty cycles in the range between 1% to 10%. While these values may appear low, it is important to note that a neighbor discovery protocol has to be run in the background at *all* time, regardless of whether the user is actively using the phone. Hence, even a seemingly low duty cycle can translate to substantial energy consumption. As an illustrating example, consider a phone with a battery[5] of $6Wh$ and a recharging cycle of 48 hours. A phone normally consumes around $1W$ when using its Wi-Fi interface for transmission. With the neighbor discovery service over Wi-Fi being operated at

[5] As a reference, the battery capacities of iPhone 5 and Samsung Galaxy S3 are $5.45Wh$ and $8Wh$ respectively.

a duty cycle of $5\%$, this service consumes $40\%$ (i.e., $48h \times 1W \times 5\%/6Wh$) of the total battery capacity, which is far from acceptable. In fact, even when operating at the much lower duty cycle of $1\%$, this service's energy footprint is still $8\%$. Given that neighbor discovery is an underlying service that does not provide direct values to the smart phone users by itself, such a level of power consumption (i.e., nearly $10\%$) can already be visible to the end users and may make running such a service unattractive. We argue that for WiFi based neighbor discovery, a duty cycle of $1\%$ should be treated more like an upper limit rather than a lower limit.

Now consider the use of the Bluetooth interface — specifically, Bluetooth Low Energy (BLE) — for neighbor discovery. BLE provides considerably reduced power consumption while maintaining a similar communication range ($\approx\ 30m$) as earlier Bluetooth. We measure that phones consume around $250mW$ (with screen off but CPU running to process packets) when using its Bluetooth interface. Since Bluetooth is about $4\times$ more power efficient than WiFi, its duty cycle upper limit can be set at $4\%$.

Table 3.2 compares the Wi-Fi and Bluetooth interface and how the same power budget maps to different duty cycle values for them. If the discovery protocols are to have negligible energy footprint, a duty cycle of 1% or less is clearly desirable, even for BLE.

Besides the standard duty cycle needed for the search of neighbor phones, neighbor discovery protocols may incur other overheads. For example, a protocol that relies on time synchronization (e.g., the recent RBTP [58]) needs the phone to consume some extra energy for that. To provide a fair comparison between protocols of different designs, we generalize the definition of duty cycle to include all these necessary overheads, and we use $f$ to denote this *generalized duty cycle*. Note that $f$ is defined as regard to a specific radio interface: $1\%$ of duty cycle for Bluetooth consumes only $1/4$ of energy compared to $1\%$ of duty cycle for WiFi.

Finally, the use of either WiFi or BLE implies different possible ranges of slot length, $T$, usable. For WiFi, $T$ would be $100ms$ or larger, , but as energy efficient IoT node appears, the trend shows shorter T is possible as wakeup overhead becomes

| | Definition | Example values |
|---|---|---|
| $P$ | Active power for a selected radio interface | $P_{WiFi} \approx 1W$<br>$P_{BLE} \approx 0.25W$ |
| $f$ | Duty cycle, as extended to capture all overheads (e.g., synchronization cost) and normalized to chosen interface | Desirably:<br>$D_{WiFi} < 1\%$<br>$D_{BLE} < 4\%$ |
| $D$ | Bound for discovery delay (under protocol-specific assumptions) | Desirably $< 30s$ |
| $T$ | Time slot duration needed for an interface to wake up, transmit or receive a few packets, and sleep | $T_{WiFi} = 100ms$ (can be higher, e.g., $2s$ without driver support)<br>$T_{BLE} = 10ms$ |
| $N$ | Neighborhood size | 1 $\rightarrow$ several dozens |
| Key parameters for synchronous protocol | | |
| $\Gamma$ | Energy consumed for one time-synchronization operation | NTP over Cellular:<br>$\Gamma_g = 40J$<br>Local WiFi AP:<br>$\Gamma_l = 0.1J$ |
| $Y$ | Bound for initial synchronization error | NTP over WAN:<br>$Y_g = 20ms$<br>Local WiFi AP:<br>$Y_l = 2ms$ |
| $x$ | Bound for clock drift | $20PPM$ |
| $\kappa$ | Number of reference clocks for locally synced protocols | $\geq 1$<br>$\kappa = 3$ often suffices |

Table 3.3: Notations used for neighbor discovery analysis.

smaller, e.g., GainSpan [6]. For BLE, $T = 10ms$ is possible.

For easy reference, Table 3.3 summarizes the key notations that we will use in the following discussions.

## 3.3 Literature review

This section reviews the major neighbor discovery protocols in the literature (see a summary in Table 3.4). Early neighbor discovery protocols were most designed to serve static wireless sensor networks, which is not suitable for DTNs. In this review, we focus on the works that target mobile networks. We categorize them into two major categories: asynchronous protocols which are designed without time synchronization

[6] http://www.gainspan.com/docs2/GS2011M-PB.pdf, checked on 30 Aug 2014

| Category | Representative protocols | Delay bound $D$ |
|---|---|---|
| Asynchronous probabilistic | Birthday protocols [50] | None |
| Asynchronous deterministic | Quorum-based [52, 108] Disco [54] U-Connect [55] Searchlight [56] | $\theta(\frac{T}{f^2})$ |
| Globally synchronized | RBTP [58] | $\theta(\frac{x\Gamma_g}{Pf^2})$ |
| Locally synchronized | $E^2D$ [59] | $\theta(\frac{\kappa^2 x\Gamma_l}{Pf^2})$ |

Table 3.4: The spectrum of neighbor discovery approaches and how their performance depends on their key parameters.

among the neighbors, and synchronous protocols which make use of certain degree of synchronization.

### 3.3.1 Asynchronous protocols

For asynchronous protocols, there are two subcategories: the early birthday protocols [50] provide simple and robust ways for achieving probabilistic neighbor discovery, while most of the follow-up efforts focus on leveraging certain deterministic designs to provide worst-case delay guarantee.

**Birthday Protocols.** McGlynn and Borbash [50] proposed a family of protocols for neighbor discovery in static ad hoc wireless networks. For each slot, this family of protocols let each node independently decide whether to wake up (switch on wireless interface) to send or receive messages, or sleep (switch off wireless interface). The decision is made in a probabilistic way. The authors discussed how to set the probabilities under different settings (e.g., varying neighborhood size and energy budget) so as to improve the discovery performance. The probabilistic nature of the protocols simplifies the design and also provides smooth performance degradation when the actual deployment differs from the planned case. However, the probabilistic nature of these protocols does not provide any worst case discovery delay guarantee, i.e., $D = \infty$.

Partly motivated by this, a steady stream of efforts has been extended to study *asynchronous protocols that leverage deterministic designs to provide worst-case discovery delay guarantee*. The following protocols are representatives of these efforts:

**Quorum-based protocols.** Tseng et al. [52] proposed a protocol based on a grid quorum system. A quorum system is a collection of sets where any two sets have a non-empty intersection. In this quorum-based protocol, time is divided into a sequence of periods, where each period consists of $n^2$ time slots. Each node uses the same $n$ and interprets the slots in each period as an $n \times n$ array. It then selects a row and a column and wakes up in every slots there. The duty cycle of this protocol is $f = (2n-1)/n^2 \approx \frac{2}{n}$. This quorum-based design ensures that any two asynchronous nodes have at least 2 common wake-up slots every period, and the worst-case discovery delay of $D = (n^2-1)T \approx \frac{4T}{f^2}$. Jiang et al. [108] proved that no quorum-based protocols can achieve $D < \frac{T}{f^2}$. In the same work, Jiang et al. also proposed a number of variant protocols based on different quorum systems that provides constant factor improvement.

**Disco.** Dutta and Culler [54] proposed the Disco protocol, which employs the idea from Chinese Remainder Theorem to ensure guaranteed worst-case discovery delay. In the Disco protocol, each node selects two different primes ($p_1$ and $p_2$) and wakes up when its local time is a multiple of either $p_1$ or $p_2$. The duty cycle of Disco is hence $f = \frac{p_1+p_2}{p_1 p_2}$. If all nodes use the same pair of primes $p_1$ and $p_2$, the worst-case delay is $D = p_1 p_2$. Since $f = \frac{p_1+p_2}{p_1 p_2} > \frac{2}{\sqrt{p_1 p_2}}$, we have $D > \frac{4T}{f^2}$. If $p_1$ and $p_2$ are within a constant ratio, we still have $D = \theta(\frac{T}{f^2})$. One desirable property of Disco is that each phone can base on its own duty cycle budget to choose its own pair of primes.

**U-Connect.** Kandhalu et al. [55] proposed the U-Connect protocol, which adapts the previous quorum-based protocol [52] by incorporating the co-prime property. Recall that each period in the quorum-based protocol [52] contains an $n \times n$ array of slots and each node wakes up at one row plus one column in each period. U-Connect allows different nodes to use different period, as long as its size equals to $p \times p$ for some prime

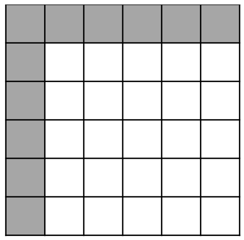
$p$. Hence, a node can choose the value of $p$ based on its own duty cycle budget. Another technique used in U-Connect is to cut the wake-up slots from one column plus one row down to approximately one column plus half of a row. Specifically, U-Connect reduces the duty cycle from $f = \frac{2p-1}{p^2} \approx \frac{2}{p}$ to $f = \frac{p+\lceil\frac{p+1}{2}\rceil}{p^2} \approx \frac{1.5}{p}$ (with a saving of around $25\%$), with negligible impact on the delay $D$.

While only slightly increasing the worst-case delay from $D = (p^2 - 1)T$ to $D = p^2T$ between nodes with the same parameter $p$. For two nodes with different duty cycle of $f_1$ and $f_2$, U-Connect's design ensures that $D = \theta(\frac{T}{f_1 f_2})$. In comparison, the original quorum-based protocol does not enforce the co-prime property and can result in $D = \theta(\frac{T}{(\min(f_1,f_2))^2})$ for two nodes with duty cycle values of $f_1$ and $f_2$ respectively.

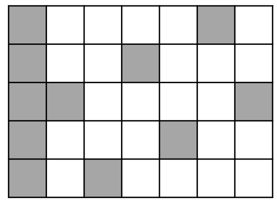
**Searchlight.** Bakht et al. [56] proposed the Searchlight protocol, which further improves U-Connect by introducing a number of techniques. First, Searchlight organized probing in a systematic way. Each period of Searchlight is still an $n \times n$ array of slots and a node still wakes up in one row. Besides that, for the remaining $n \times (n-1)$ array of slots, instead of waking up only in slots of one single column (as in quorum-based protocol and U-Connect), Searchlight lets a node to choose one 'probe' slot in each column. To do 'systematic probing', Searchlight changes the relative position of the probe slot across columns. This basic version of Searchlight has a duty cycle value of $f = \frac{2n}{n^2} = \frac{2}{n}$, and its worst-case discovery delay is $D = n\lfloor\frac{n}{2}\rfloor T \approx \frac{2T}{f^2}$.

To further improve performance, Searchlight proposes to make the wake-up time of a phone 'overflows' from the slot boundary by a small fraction of $\delta$, so that the systematic probing can skip all odd positions of a column. Such a striping design roughly halves the worst-case discovery delay to $D = n\lceil\frac{\lfloor\frac{n}{2}\rfloor}{2}\rceil T \approx \frac{T}{f^2}$ as compared to the non-striped version. Finally, Searchlight supports the use of a random probing pattern. While this does not change the worst-case discovery delay, it improves the average discovery delay.

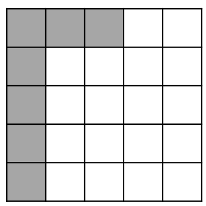
Figure 3.2 illustrates the wake-up slot schedule for the four asynchronous deterministic protocols as we reviewed above. Note that the worst-case discovery delay guarantees for all of them are in the order of $\theta(\frac{T}{f^2})$. With all its optimization tech-



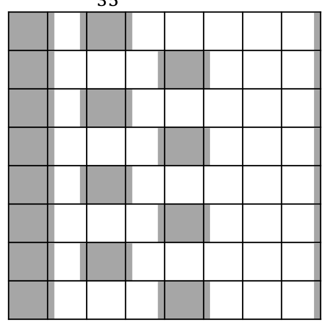
(a) Grid-quorum-based: $n = 6, f = \frac{11}{36} \approx 0.3$, $D = 35T$



(b) Disco: $p_1 = 5, p_2 = 7$, $f = \frac{11}{35} \approx 0.3, D = 35T$



(c) U-Connect: $p = 5$, $f = \frac{7}{25} \approx 0.3$, $D = 25T$



(d) Searchlight: $n = 8, \delta = 0.2$ $f = \frac{16}{64} \times 1.2 = 0.3, D = 16T$

Figure 3.2: Wake-up schedules of major asynchronous deterministic protocols.

niques, the latest Searchlight protocol [56] can reduce the worst-case delay by up to $4\times$ as compared to earlier protocols [52, 54].

### 3.3.2 Synchronous protocols

A few very recent efforts begin to leverage certain degree of time synchronization for better neighbor discovery performance. Based on the scale of synchronization, we divide the synchronous protocols into to categories: the global synchronization protocol and the local synchronization protocol. The prior one requires all nodes to synchronize to a global clock, while the later one requires close-by nodes synchronize to common clock and nodes of different parts can synchronize to different clocks.

**RBTP — A globally synchronized protocol.** Li and Sinha [58] proposed the Recursive Binary Time Partitioning (RBTP) protocol, which leverages global time synchronization sources like NTP servers to reduce discovery delay. They conducted a controlled experiments where $8$ phones, are used. For each phone, its clock skew does not vary with time hence RBTP can use linear fitting to reduce the clock drift rate. In the experiment, the maximum synchronization error can be kept within $100ms$ by letting each phone synchronize with a NTP server every 6 hours. RBTP hence assumes that global synchronization can be done with *negligible overhead*. We will revisit this claim in Section 3.4. Assume overhead can be ignored, for phones with the same duty cycle of $f = \frac{1}{n}$, a trivial design that lets each phone wake up once in every of the $n^{th}$ slot provides the optimal worst case delay $D = nT' = \frac{T'}{f}$, where the slot length $T'$ depends on both $T$ and the synchronization error. Hence, $T'$ can be greater than $T$. The key design of RBTP is to deal with the case when phones can have different duty cycle budgets, say $f_1$ and $f_2$. By letting a node wake up at slots that recursively partition a time frame in a binary fashion, RBTP can achieve a worst case delay of $D < 2\frac{T'}{\min(f_1,f_2)}$, which is near optimal.

Note that RBTP assumes all phones have Internet access for synchronization with the NTP server, which may not be true for some phones (e.g., those without cellular data plan and/or no Wi-Fi access) or at least not desirable (e.g., for roaming phones). Its performance also depends on both the synchronization overhead (i.e., whether it is really negligible) and the synchronization quality (which leads to, e.g., how $T'$ compares with $T$).

**E$^2$D — A Locally Synchronized Protocol.** Camps-Mur and Loureiro [59] proposed the Energy Efficient Discovery ($E^2D$) Wi-Fi mechanism, which uses in-situ Wi-Fi access points (APs) infrastructure to help synchronize mobile nodes. Since the synchronization can be done based on the timestamps in the AP beacons or probe responses, there is no need for the mobile nodes to have permission to associate with these APs or gain Internet access through them. There is also no need to make any changes to the APs. $E^2D$ is designed to discover clusters (instead of individual nodes). Specifically, when a node gets the timestamps from an AP, it uses that as a reference to calculate when it should wake up to meet the cluster of nodes (if there is any) that are synchronized to the same AP.

The functionality provided by $E^2D$ is different from most other neighbor discovery protocols. Specifically, $E^2D$ is designed for phones to quickly discover nearby clusters, instead of discovering and maintaining individual neighbor phones. Section 3.4 will analyze the performance of a locally synchronized protocol (like $E^2D$) that is designed for the traditional goal of pairwise discovery.

Since different APs may not be synchronized, and different clusters may be based on different APs. $E^2D$ lets a node to carry out such a scanning process for multiple reference APs (up to a certain number).

## 3.4 Which protocol to use?

Based on the literature review, we will next discuss the design space of the various neighbor discovery algorithms. As asynchronous algorithms have been well analyzed, we will simply quote the results from the relevant work. For synchronous algorithm, as the analysis results provided in RBTP and $E^2D$ do not explicitly include the synchronization overhead and error, we perform the required analysis to quantify their performance, we will qualitatively analyze their performance.

We analyze how different protocols perform when operating with different radio interfaces (i.e., Bluetooth vs. Wi-Fi) and under varying power budget (as measured by the generalized duty cycle $f$ over the chosen radio interface). The choice of different interface results in different time slot duration $T$ and power consumption $P$, which can affect different protocols in different ways. For synchronous protocols, our analysis further includes $\Gamma$ (synchronization overhead), $Y$ (maximum initial synchronization error), $x$ (maximum clock drift) and $\kappa$ (number of local time references utilized).

### 3.4.1 Asynchronous protocols

One advantage of asynchronous protocols is that there is no need for infrastructure support. They do not incur synchronization overhead and are not sensitive to clock drift. Also, by waking up in an asynchronous manner, these protocols require the minimum coordination among nodes and are simple to implement.

| | $f = 0.1\%$ | $f = 1\%$ | $f = 10\%$ |
|---|---|---|---|
| $T$=10ms | 10,000s | 100s | 1s |
| $T$=100ms | 100,000s | 1,000s | 10s |
| $T$=1s | 1,000,000s | 10,000s | 100s |

Table 3.5: Values of $D$ for different pairs of $T$ and $f$ for asynchronous protocols.

As reviewed in Section 3.3, for all existing asynchronous deterministic protocols, their delay performance scales in the same form with respect to $T$ and $f$ as $D = \theta(\frac{T}{f^2})$. As proved in Jiang et al. [108], this is a tight lower bound. Since Searchlight [56] provides the smallest constant one, we will discuss this category of protocols generically based on Searchlight's delay performance, i.e.:

$$D = \frac{T}{f^2} \tag{3.1}$$

The value of $T$ can vary by two orders of magnitudes: from more than one second (controlling Wi-Fi from the application layer [56]), to 100ms (controlling Wi-Fi from low level Wi-Fi device driver) and $10ms$ or less (using low-power network technologies like ZigBee and BLE). For illustration, consider values of $T$ of $1s$, $100ms$ and $10ms$ and the values of $f$ of $10\%$, $1\%$ and $0.1\%$.

Table 3.5 shows the values of $D$ for different combination of duty cycles and time slot sizes. If the objective is to have $D$ shorter than or equal to $30s$, the only admissible configurations above are for $f = 10\%$, for $T = 100ms$ or $T = 10ms$. On the other hand, if the objective is to keep $f$ small say at $1\%$, then D has to be at least $100s$ even for $T = 10ms$. For lower values of $f < 1\%$, $D$ is larger than $100s$, even for small $T$. Only when larger duty cycles are used that D can be reduced. As shown in the table, asynchronous protocols is attractive if $T$ can be made small, say $\leq 10ms$ and $f$ can be set to a high value of around $10\%$. This is a fundamental limitation for asynchronous protocols.

### 3.4.2 Synchronous protocols

Synchronous protocols incur additional overheads that are not explicitly considered in earlier work, which include clock drift, synchronization overhead, and synchronization accuracy. We consider how these overheads impact the performance in the rest of this section.

**Clock drift.** Clock drift of quartz oscillators was widely observed. And the clock drift depends on many environmental factors, such as temperature, humid, as discussed in [110]. In order to test the clock drift of quartz oscillators on smart phones, we experimented on Samsung Galaxy SII and Galaxy Nexus phones. We first put phones outside, where the air temperature is higher than 35 degree. Then, we placed the phones inside lab, where the air temperature is about 20 to 25 degrees under air-condition control. The testing phones keep checking the time difference between itself and the same AP (under the two different environments) it can hear.

The experiment result shows two points, as showed in Figure 3.3. First, the clock drift rate dramatically varies under different temperatures and the drift rate can be up to $20ppm$. Second, the relative clock drift rate (between two mobile phones) can be much faster than absolute clock drift (between phone and standard time). The clock drift direction (with the reference to the AP in the test) are opposite under the different environments we tested, i.e., clock slower under some environment while clock faster under some other environment. For example, when two phones are synchronized, and then placed into two environments with different temperatures for one hour, the time difference between them can be 90ms.

Based on this observation, we can see that phones have different clock drift rates under the same environment, and the same phone has different clock drift rates under different environment. Thus, a good clock compensation solution for smart phones should consider both nature of quartz oscillators and the environmental factors. This is not easy.

Figure 3.5.1 shows that clock drift changes dramatically with temperatures. The

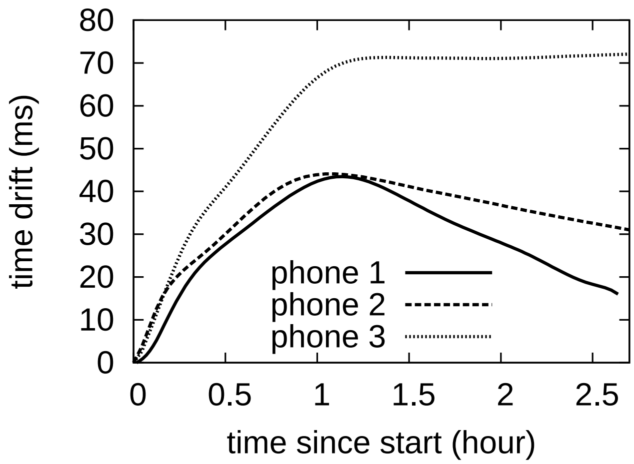



Figure 3.3: Mobile phone clock drift experiments in Singapore. In the first one hour, the three phones were placed outdoor, and in the next hour the phones were put in a research lab with air condition. The temperature difference between lab and outdoor is about 20 degree.

results show that under different temperatures, a phone's clock drift can make the clock faster and can also make the clock slower. It is dependent to temperatures.

**Synchronization overhead and accuracy.** We use $\Gamma$ to denote the energy overhead required for achieving certain degree of initial synchronization accuracy. We use $Y$ to denote the maximum initial synchronization error, which is the maximum clock difference between two phones immediately after they synchronize to the same time reference.

The synchronization parameters (overhead $\Gamma$ and initial error $Y$) for globally synchronized protocol and locally synchronized protocols can differ by multiple orders of magnitude. We denote them by $\Gamma_g$, $Y_g$ , and $\Gamma_l$, $Y_l$, where the subscript $g$ and $l$ stand for 'global' and 'local' respectively.

For a globally synchronized protocol like RBTP, one round of synchronization can take on average $18s$ [58]. This high overhead is because RBTP uses NTP, which requires the transmission of multiple (e.g., $10$) packets over WAN (e.g., cellular network). Considering the fact that cellular network can consume more than $2W$ of power during transmission, and it incurs a long tail ($\approx 10s$) after the transmission completes. The energy consumption of one time synchronization operation can be in the order of

$\Gamma_g \approx 40J$. Due to the noisy round trip time over WAN, the synchronization accuracy can be in the order of dozens of $ms$, e.g., $Y_g = 20ms$ as reported [58].

For a locally synchronized protocol like $E^2D$, the synchronization overhead can be much lower. Our measurements, as well as those done by $E^2D$ indicates that that $\Gamma_l = 0.1J$ is possible since it only requires a phone to use its Wi-Fi interface for $100ms$ to transmit a probe request packet and receive the probe reply packets from nearby APs. Also, as the synchronization happens locally and is based on packet reception time, the initial synchronization error is also lower. Both $E^2D$ and our experiment show that $Y_l = 3ms$ is possible.

**Performance of globally synchronized protocols.** We now qualitatively analyze how the clock drift, synchronization overhead and accuracy, as well as the selected radio interface, affect the performance of a globally synchronized protocol like RBTP.

In the ideal synchronization algorithm, where perfect synchronization applies without synchronization overhead, the worst discovery delay ($D$) is $T/f$. In this case, once two close-by nodes wake up, they discover each other.

Next we take the time drift and synchronization overhead into account. One key factor in this case is how often a node needs to synchronize. If a node synchronize too often, it generates massive synchronization overhead. On the other hand, if a node conducts too sparse synchronization, then the node has to have longer wake-up time (i.e., a larger $T$ ) to compensate the time drift.

Without loss of generality, let us consider the case when each phone performs time synchronization periodically, where the synchronization cycle is $\frac{T}{f}h$. It means every $h$ cycle a node do one time synchronization. In this setting, the worst time drift can be $\frac{T}{f}hx$, i.e., a node needs to wake up $\frac{T}{f}hx$ time to compensate the time drift. Considering this extra cost, the new cycle is adapted to $\frac{Thx/f+T}{T}$.

Next, we take into the cost for synchronization overhead. Each time synchronization overhead is $\Gamma/P$. In the computation, this overhead is regarded as average distributed to each cycle, which cause overhead of $\frac{\Gamma}{Ph}$ per cycle. This extra cost again causes enlarge the cycle by a factor of $\frac{\Gamma/Ph+T}{T}$. The follow equation is the worst case delay considering all overheads.

$$D = \frac{T}{f}\Big(\frac{Thx/f + T}{T}\Big)\Big(\frac{\Gamma/Ph + T}{T}\Big) \tag{3.2}$$

By optimizing $h$, a minimum value of $D$ is as follows:

$$D_{mini} = \frac{T}{f}\Big(\frac{\Gamma x}{fPT} + \sqrt{\frac{2\Gamma x}{PTf}} + 1\Big) \tag{3.3}$$

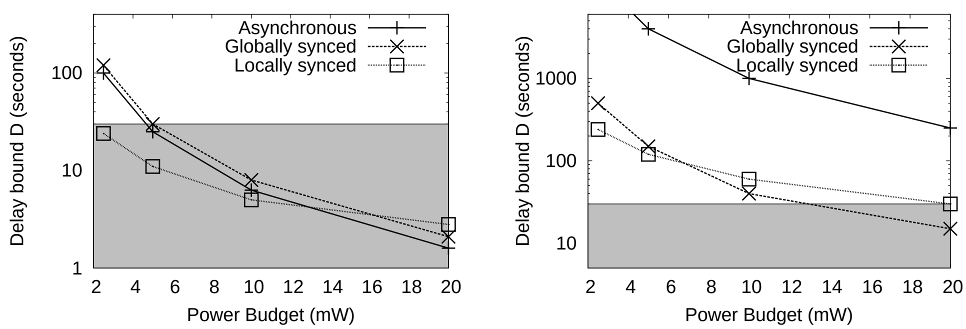
By the equation 3.3, we can see that the synchronization approaches actually have the same asymptomatic results $\theta(\frac{x\Gamma}{Pf^2} + \sqrt{\frac{2\Gamma x}{PTf}})$ as the asynchronous protocols.

**Performance in realistic settings.** It is easy to see that the delay performance of neighbor discovery is dependent to the time drift speed ( $x$), the time length per wake-up ($T$), and the synchronization overhead ($\Gamma$). It can be observed that the ratio of $\frac{x\Gamma}{P}$ plays an important role in determining how large $D$ is as it is the factor for the $\frac{1}{f^2}$ term. Intuitively, if the product of the clock drift and synchronization energy overhead is large, the first term can dominate the second term with decreasing value of $f$.

We illustrate this using the settings of RBTP and assume that BLE is used. We consider $f = 1\%$, $x = 20PPM$, and recall that $\Gamma_g = 40J$ and $P_{BLE} = 0.25W$, hence, $\frac{x\Gamma}{P} = 3.2 \times 10^-3s = 3.2$. Consider the different time drift direction, the constant will be doubled to 6.4. Note that the above equation does not include the synchronization error. Suppose, the synchronization error is 20ms, and then each wake-up length should be dragged from 10ms to 30ms, and the cycle will three time longer in order to keep the constant cost. Thus, the constant will become 19.2, which is larger than $T$ as used in asynchronous protocols.

Recall that for asynchronous protocols, the constant factor for $\frac{1}{f^2}$ is only $T = 10ms$ for Bluetooth setting. Hence a globally synchronized protocol can perform worse than asynchronous protocols when the time slot $T$ is small and when the time synchronization overhead is relatively high as gauged by the radio's power consumption. Combining this with the fact that the asynchronous protocols do not rely on any infrastructure, asynchronous protocols is the clear winner in such a setting.

On the other hand, if Wi-Fi interface is used, since $P_{Wi-Fi} = 1W$, RBTP's constant becomes $3.2ms$. Meanwhile, the constant for asynchronous increases from $10ms$ to $100ms$. Hence RBTP becomes the clear winner.




(a) Bluetooth settings (b) Wi-Fi settings

Figure 3.4: The neighbor discovery power consumption under different interface settings.

**Performance of locally synchronized protocols.**

Instead of syncing to a single reference for the global time, a protocol like $E^2D$ uses local reference(s). On the positive side, the syncing overhead and the time synchronization error can be both reduced. For example, to find the timestamps of multiple nearby APs, a phone needs only $100ms$ to send the probe request message and receives the probe replies. Hence $\Gamma_l$ can be in the order of $0.1J$, which is significantly smaller than the overhead of global synchronization $\Gamma_g = 40J$. On the down side, as all these references are local, a phone may need to use multiple reference APs in order to improve the opportunity that it shares a common reference AP with its neighbor. Let the number of reference APs needed be $\kappa$, to achieve a delay bound of $D_0$ (under the assumption that there is already one common AP identified), for each syncing cycle $C_{sync}$, a phone needs to wake up $\frac{\kappa C_{sync}}{D_0}$ times.

Another important concern for locally synchronized protocols is that as a phone moves, it needs to refresh its reference APs as well. Hence, even when time synchronization is not an issue, the detection of movement (e.g., via accelerometer values) should trigger AP scanning. As an illustrating example, half of neighbor discovery wake-up energy is lent to movement synchronization wake-up is enough. Therefore, the duty cycle increases 0.5%, where a node can wake up 20ms every 4s. Within 20 ms, a node can receive more than 90% of probe replies from near-by APs. Within 4s, a mobile node carried by a pedestrian usually can move shorter than 10 meters. It is a safe distance that the node can promptly discover potential new APs without much delay. By following the same analysis step for global synchronization approaches, we can get the results as indicated by equation 3.4. It counts in the number of AP required and the synchronization cost triggered by nodes movement.

$$D_{mini} = \frac{4\kappa^2 T}{f}\Big(\frac{\Gamma x}{PTf} + \sqrt{\frac{2\Gamma x}{PTf}} + 1\Big) \tag{3.4}$$

Note that asymptotically, the locally synchronized protocol still scales by $\frac{1}{f^2}$ with decreasing duty cycle. This is the same as all the other protocols. However, since $\Gamma_l$ is significantly (two to three orders of magnitudes) smaller than $\Gamma_g$, as long as $\kappa$ does not need to be very large, locally synchronized protocols can potentially outperform both asynchronous protocols and globally synchronized protocols, especially with decreasing value of $f$.

**Summary.** Figure 3.4 compares the delay performance of different categories of protocols, where the locally synchronized protocols use a default $\kappa = 3$ value (which is sufficient for most cases as we will show in Section 3.5). As illustrated in the figure, depending on the selected radio interface and duty cycle requirements, different protocols' performances will be better:

- When the power budget is relatively high (e.g., 8% to 16% of the battery capacity):
  - *Asynchronous protocols* perform best when wake-up slots are short – e.g.,

10ms for Bluetooth Low Energy (BLE).

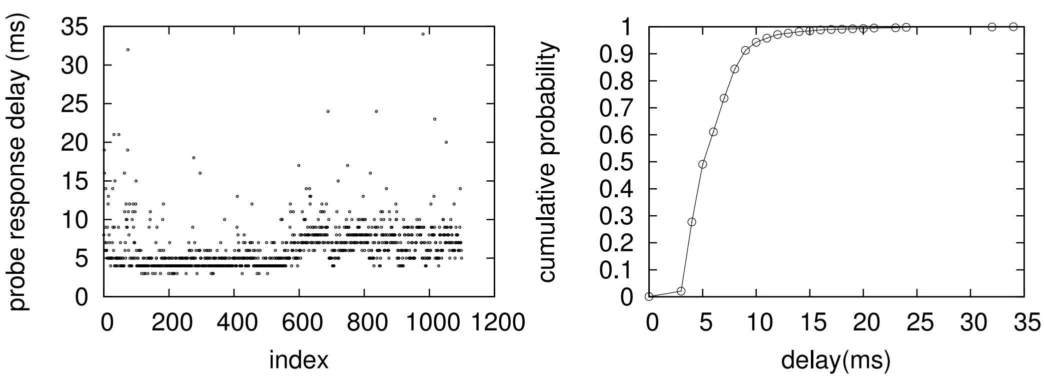
– *Globally synchronized protocols* perform best when wake-up slots are longer – e.g., 100ms for Wi-Fi.

- When the power budget is lower (e.g. 2% to 4% of the battery capacity), the *locally synchronized protocols* used with BLE performs best.

Among all protocols analyzed, locally synchronized protocols are the most promising category for simultaneously achieving negligible energy footprint ($< 3\%$ of battery capacity) and low discovery delay ($< 30$ seconds).

## 3.5 Locally synchronized protocols: the potential and the gap

### 3.5.1 Local sync by APs




(a) (b)

Figure 3.5: (a) The delay between the time sending a probe request packet to the time receiving the probe reply packets. (b) Cumulative probability of the delays in Figure (a).

In this part, we evaluate the sync accuracy by AP beacons. AP transmits two types of packets containing timestamps. The first type is the AP beacon, which an AP broadcasts every about 104ms. The second type is the probe reply, which is transmitted in the reply to probe request. In both of the two types of packets, the receiving phone can parse the packets and get the timestamp filled by the sender AP. A phone can get the time information by overhearing beacons or by explicitly sending probe request packets to trigger probe reply packets. If a phone is performing neighbor discovery on Bluetooth interface, it can still get time information through its Wi-Fi interface. Here we mainly discuss the second method, i.e., the probe request/reply, as it can be used in more general cases, and have short delay in retrieving timestamps.

For the case of using probe request/reply packets, we need to answer two questions. Firstly, how fast can a phone get probe reply from surrounding APs? The answer to this question determines the length of wake-up for obtaining AP timestamps for sync, i.e., $\Gamma_l$. Secondly, how accurate is this form of sync? This decides the overhead wake-up to compensate the sync inaccuracy, i.e., $Y_l$.

For the first question, we wrote a program to send probe request periodically and mark the time delay of receiving respective probe replies from different APs. We run the programme on different phones and at the locations with more than 10 APs. We list the results on the delays in Figure 3.5. From the results, we can see that more than 98% of probe replies can be received within 20ms after a phone broadcasts probe request packets. This implies that a phone only needs to wake up about 20ms for one time sync.

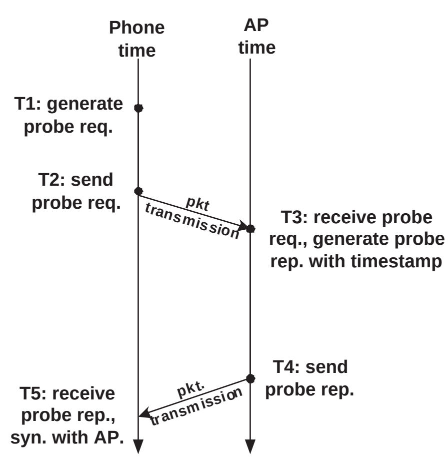



Figure 3.6: The Wi-Fi probe request/reply packet process.

For the second question, the inaccuracy is caused by wireless transmission delay.

Figure 3.6 illustrates the process of probe request/reply beacon generation and transmissions. The sync accuracy relies on the delay between the time point when the AP generates the probe reply and the time point when the phone receives the probe reply from the AP. The packet transmission time is usually short, as the Wi-Fi transmission range is extremely short for wireless signal transmission speed. Thus, we can safely omit the delay of T2-T3 and T4-T5. Thus, the main delay is the wireless traffic queue delay, i.e., the time difference between T3 and T4. If this delay is constant, the sync between mobile phones will be accurate. Unfortunately, this delay depends on Wi-Fi transmission traffic load. Therefore it usually varies drastically.

We measure the variation of the time period between T1 and T5, as shown in Figure 3.6. The delay consists of two main segments of time delays, the T1 to T2 and T3 to T4, both of which are caused by wireless channel access. The experimental phone sends out one probe request every one second, and measures the time difference ($D_t$ in millisecond) of receiving adjacent probe replies from the same AP. If the delay is constant, the time difference of receiving the probe replies should be one second. We measure the delay variation by $D_t - 1000(ms)$, as shown in the upper part of Figure 3.7 for a duration of 60s. We can see that at most time the variation is within 3ms. Considering that some replies are slower, and some are faster than the average delay, then the maximum in sync inaccuracy between two phones (with the same AP as clock reference) can be 6ms. In order to reduce the inaccuracy, we check the delay variation by taking the delay by averaging the delays of last three probe replies. As we have expected, the average delay is much smoother than the individual delay. The variation is reduced to 1ms, as shown in the lines in the lower part of Figure 3.7.

### 3.5.2 A small number of reference APs suffice

In the context of multiple APs, one major concern for locally synchronized protocols is whether two neighbor phones can effectively select the same time reference, especially in urban areas where each phone can often be simultaneously covered by dozens of different APs. Figure 3.8 (a) plots the distribution of the number of APs a phone can

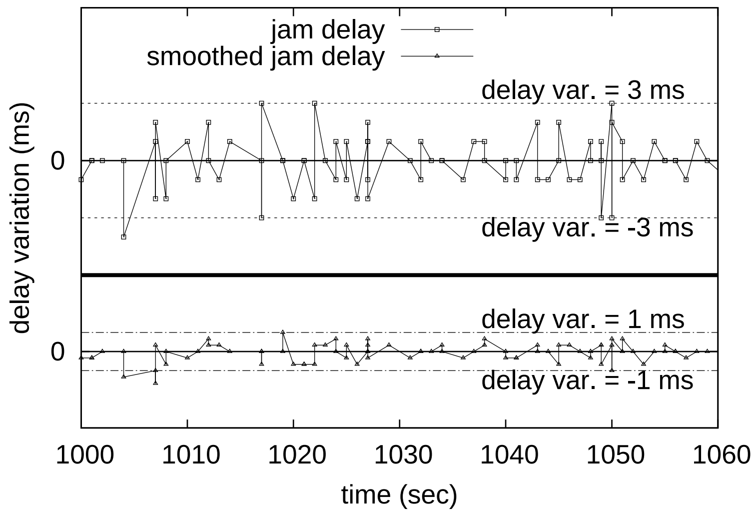



Figure 3.7: The variation of traffic jam delays.

overhear over all channels (within a 10 seconds interval) when the phone user roams at walking speed in the town areas of two different cities over a period of 9 hours, covering an area of $15km^2$. As shown in Figure 3.8, for both cities, a phone is covered by more than 20 APs for more than half time, and covered by more than 50 APs at around $20\%$ of time.

If a phone needs to synchronize with all APs that it can overhear, $\kappa$ can be a large value. Recall that the locally synchronized protocol has a factor of $\frac{\kappa x \Gamma_l}{P}$ for the $\frac{1}{f^2}$ term, while the globally synchronized protocol has a factor of $\frac{x\Gamma_g}{P}$. A large $\kappa$ can offset a significant fraction of a locally synchronized protocol's advantage over the globally-synchronized protocols.

We use both analysis and trace-driven simulation to show that this problem can be solved by a simple *minimum-$k$* strategy with $k$ as small as 3. Under such a minimum-$k$ strategy, each phone simply uses a global hashing function to hash the MAC address of each AP it can hear. Then it synchronizes with the $k$ APs with the minimum hash values.

For the analysis, we assume both APs and phones are distributed uniformly and independently in space. We assume a unit disk model, where a phone can communicate

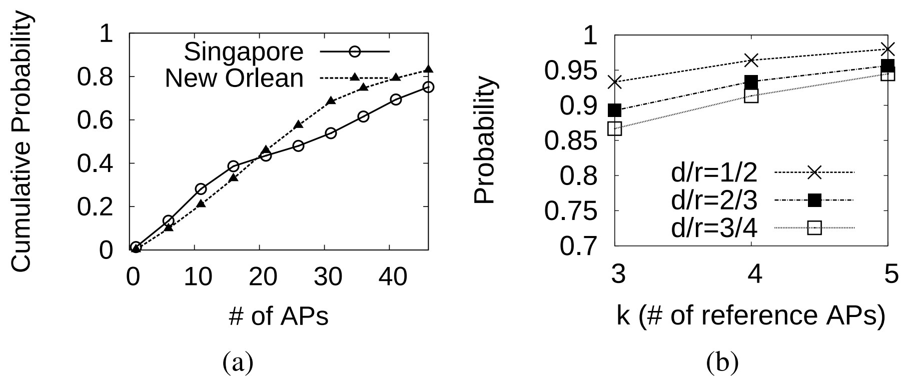



Figure 3.8: (a) The distribution of the number of APs a phone can overhear in urban settings. The Singapore trace is collected when walking through university campus. The New Orlean trace is collected when walking through CBD area. (b) The probability for two phones to select at least one common reference AP.

with an AP if their distance is less than $r$, and two phones can communicate with each other if their distance is less than $d$. Since APs normally have higher antenna gain than phones, $d$ is usually smaller than $r$. If the phones use Bluetooth to discover neighbors, $d/r$ can often be smaller than $1/2$. We also assume a dense AP environment, where each phone can hear at least 5 APs (as shown in Figure 3.8, this holds with high probability). Under this model, we prove that the probability that there is no common AP between two neighbor phones following the minimum-$\kappa$ strategy is less than:

$$\frac{4}{\kappa+2} \times (\frac{2d}{3r})^{\kappa} - \frac{1}{\kappa+1} \times (\frac{2d}{3r})^{2\kappa} \tag{3.5}$$

For example, if $\kappa = 3$, for $\frac{d}{r} = 3/4$, $2/3$, and $1/2$, the probability that two phones have at least one common reference AP by following the minimum-$\kappa$ approach is $90.4\%$, $93.2\%$, and $97.1\%$ respectively. See the appendix (section A.1) for details proofs. Hence, when one can tolerate a small probability of contact missing rate, $\kappa = 3$ suffices. To further validate our analysis results, we conduct a simulation based on real-world AP locations[7]. As shown in Figure 3.8 (b), the results match our analysis well and $\kappa = 3$ suffices for achieving $90\%$ of probability for two neighbor phones to share a common AP.

[7]http://www.crawdad.org/dartmouth/wardriving/, checked on 30 Aug 2014

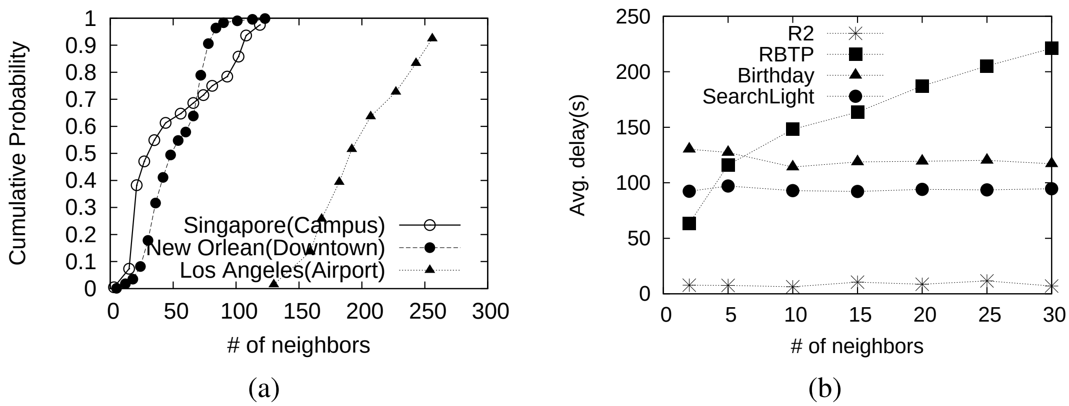



Figure 3.9: (a) The distribution of the number of neighbors in an urban setting, and (b) The performance of different protocols under increasing neighborhood size.

### 3.5.3 The scalability problem

Instead of worrying about 'too many APs', we find that the more fundamental challenge for synchronous protocols is to deal with 'too many neighbors'. In urban settings, phones often cluster together (e.g., in classroom, bus terminal, or shops). As shown in Figure 3.9 (a), in two town areas of the two cities, there can be more than $50$ neighbors for $20\%$ to $50\%$ fraction of time. The trace collected in a crowded waiting area of the airport C shows an even higher density, with the number of neighbors ranging from $100$ to nearly $300$.

With an increasing number of neighbors, the synchronous nature of these protocols can cause excessive contention, because by default all neighbors wake up at around the same time and all compete to announce their presence to their neighbors. For Bluetooth where the wake-up slot is only $10ms$, nearly all packets collide unless the contention window is optimized according to the neighborhood size. Even assuming low collision probability, only one or two packets can get through in each $10ms$ wake-up slot. Hence when a new node comes, since its broadcast message needs to compete with all the $N$ existing nodes, it needs to wait for $\theta(N)$ slots before its turn to send. This dramatically increases the discovery delay. While this can be addressed by giving a new node higher priority to transmit, there is still a need for existing neighbors to periodically rediscover each other. This is summarized as the need for 'rendezvous' by Dutta and

Culler [54], which allows phones to 'deliver messages to previously discovered neighbors with predictable and controllable latencies'. Even when all the transmissions are well coordinated and the collision is minimized, the discovery / rediscovery delay for synchronous protocols increases linearly with the number of neighbors.

Note that asynchronous protocols are less affected by the neighborhood size since their asynchronous nature automatically spreads the neighbor phones uniformly across different slots.

We evaluate the potential impact of this scalability issue on different protocols by simulating a setting with increasing neighborhood size. More details about the simulation is explained in Section 3.7. As shown in Figure 3.9 (b), asynchronous protocols are less affected by the neighborhood size (but their delay is large overall). The discovery delay for RBTP increases dramatically with the neighborhood size, as expected. In fact, for $N = 60$, their performance already becomes worse than that of the asynchronous protocols.

Motivated by the scalability problem, this thesis proposes a better protocol, R2, to decouple a protocol's (re)discovery delay from the neighborhood size. As shown in the figure, R2 scales well with $N$ and can provide less than $10s$ discovery delay for all values of $N$. The detail design of R2 will be discussed in the following part.

## 3.6 R2: a low-duty-cycle and low-delay protocol

Guided by the above observation and analysis, we design a new discovery protocol, called R2, to fill the gap for simultaneously achieving low duty cycle ($f$) and low delay ($D$) for cases with high mobile phones density.

**Basic design.** Similar to $E^2D$, R2 is a locally-synchronized protocol. R2's name comes from its unique Rendezvous-Reconnaissance design that decouples the (re-)discovery delay from the neighborhood size. The basic operations of a phone running R2 are as follow:

1) When movement has been detected (e.g., via the change of reference APs, or its

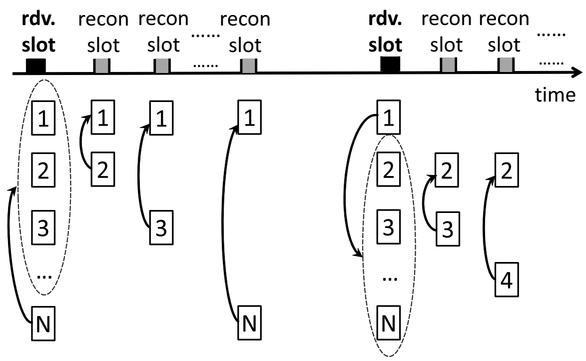



Figure 3.10: The Rendezvous-Recon design that decouples the rendezvous delay from the neighborhood size.

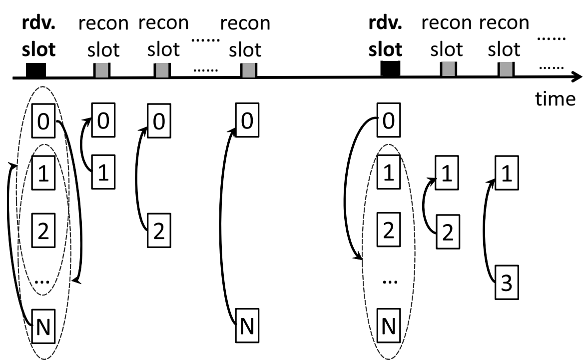



Figure 3.11: The Rendezvous-Recon design for a newcomer to quickly discover nodes in an existing cluster.

accelerometer), a phone begins to periodically send probe requests to obtain the list of nearby Wi-Fi APs and their local clocks. It then uses the minimum-$\kappa$ method to select $\kappa$ APs as local time references. Note that getting the clock of APs do not require the phone to have authenticated access to APs. The time reference can be obtained from standard 802.11 management frames, hence require no change to APs.

2) R2 then uses the local time references of Wi-Fi APs to schedule the wake-up slots for BLE interface.

**Maintenance:** Consider a simplified case where there is a single time reference and all phones have already discovered all of their neighbors. All phones are hence in the *maintenance phase*, where a node wants to know whether its neighbors are still in vicinity, and when is the next slot that it can reach a neighbor if needed. The goal now is for each phone to minimize the maintenance (or so called rediscover) delay, which measures how long a phone can get an update about the state of a neighbor phone. R2 considers all phones that are synced to the same AP as a single cluster.

For a cluster with fewer than $m$ nodes ($m$ is the average number of possible transmissions in a time slot $T$), each node simply wakes up in rendezvous slots and broadcasts to all neighbors its knowledge of the cluster's membership. When the cluster size grows beyond $m$ nodes, R2 introduces extra slot(s), called the *recon slot(s)* to address the scalability problem.

If a node is unable to broadcast during the rendezvous slot, it wakes up in a randomly chosen recon slot to broadcast the information it has collected. The number of recon slots is dynamically adjusted according to the number of nodes $N$ that a node thinks is in the neighborhood and is set to be $N/m$. The slots are spaced $\frac{m}{f}$ slots apart and the first $N/m$ slots are chosen. $N$ is taken to be the maximum of all $N$ broadcast by nodes sending in the rendezvous slot.

As illustrated in Figure 3.10, each period of time now consists of one rendezvous slot and $\frac{N}{m}$ recon slots. We call this a *Rendezvous-Recon round*, or a round for short. For each round, one node is selected to serve as the *scout*. In the first rendezvous slot depicted in the figure, node N, the scout of the previous round, broadcasts the information it has gathered and appoints node one to be its successor. Hence, node 1, being the new scout, wakes up in all the following recon slots to listen, while each of the other N-1 non-scout nodes uniformly in random selects a single recon slot to wake up and updates the scout (i.e., node 1) about their status. Node one broadcasts the information it collects back to the cluster at the rendezvous slot of the following round. In the ideal case (no collision or packet loss), every node gets an update about every neighbor within a round.

**Discovery:** Now consider the case when a new node arrives. In the example of Figure 3.11, node one to N already form a cluster while node 0 is a newcomer. In the first rendezvous slot, node N who is the scout for the previous round broadcasts, from where the newcomer node 0 learns about the existence of the cluster and its size. Node

0 will broadcast, so the other nodes know about the presence of the newcomer. Since only the scout and new nodes broadcast in the rendezvous slot, the new node has high probability to announce its presence to the group. It is natural for the newcomer to serve as the scout so it can immediately meet all neighbors in the following recon slots. Once a new node (wrt. a specific reference AP) succeeds its broadcast in the rendezvous slot, it switches to the maintenance mode.

**Analysis of R2's performance.** Consider a single cluster of $N$ nodes. Each node has a probability of $\frac{1}{N}$ to serve as the scout, for which it wakes up for $\frac{N}{m}+1$ slots (1 for the rendezvous slot, and $\frac{N}{m}$ for listening to other nodes in the recon slots) in each round. With probability of $1-\frac{1}{N}$, a node only wakes up in 2 slots (1 rendezvous slot plus one recon slot) in each round. Hence, for each node, the expected wake-up slot in a round is:

$$\frac{1}{N} \times (\frac{N}{m} + 1) + (1 - \frac{1}{N}) \times 2 < 3 \tag{3.6}$$

Hence, for increasing number of neighbors, R2's round length needs to be at most tripled when compared to the case when cluster size $N < m$. To keep the same duty cycle, the above reasoning trivially generalizes to the case of $\kappa > 1$, by simply considering each of the $\kappa$ instances as operating their own runs independently.

## 3.7 Evaluation

### 3.7.1 System Evaluation

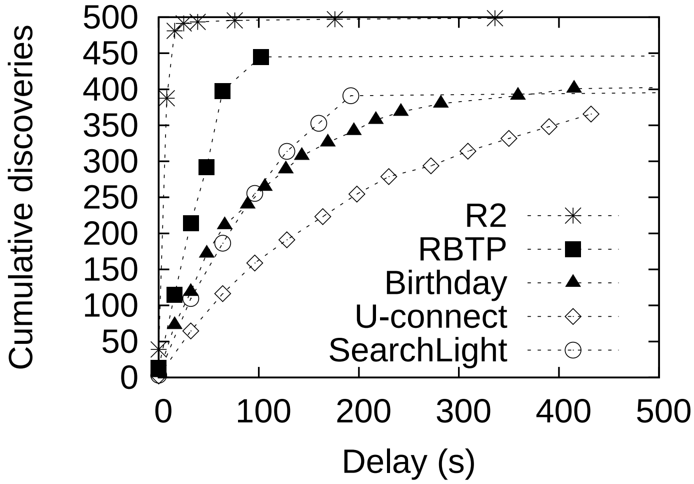
In this Section, we demonstrate the feasibility of a locally synchronized protocol using Wi-Fi beacons.

First, we show that publicly deployed Wi-Fi APs can provide timely beacons for synchronization with high probability. It is known that in response to a probe request from a Wi-Fi device, a Wi-Fi AP will response with probe reply. This is true even if the mobile device has no access permission. We run an experiment where a smartphone sends probe requests periodically and records the delay of receiving respective probe replies from surrounding APs. We run the programme at different locations with more than 10 APs and different time with varied background Wi-Fi data traffics. The results are shown in Figure 3.5(b). From the results, we can see that more than 95% of probe replies can be received within 10ms after a phone broadcasts a probe request indicating timely feedback is generally available.

Next, we measure the ability of smartphones to continuously synchronize with each other. The experiment uses two smartphones. Initially, one of the smartphones sends a probe request and the AP responses with a probe reply that can be heard by both phones. The first phone then periodically sends a single hello message (message transmission time is less than 1ms over Wi-Fi) every one second after the probe reply is received. The second smartphone listens for the hello message and computes the difference between the expected time of reception and actual reception time. The first smartphone sends a probe request every 5 minutes to mitigate the effect of clock drift. Results show that with an allowed time offset of 10ms, 93% of the total hello messages can be received over a six hour period. A short synchronization timeslot of more than 10ms is thus sufficient for discovery purpose if the local synchronization is performed every few minutes.

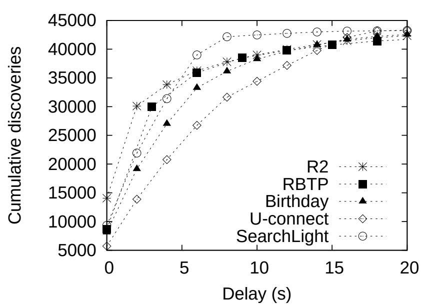
Finally, we performed a small experiment whereby we distribute Galaxy Nexus phones to six volunteers who are asked to move around a single floor of a building. Each phone runs in monitor mode, and broadcasts a hello message every second. In this way, we collected the ground truth phone-to-phone connection information and AP beacons with timestamps over a six hour period. We evaluate the performance of various protocols using the trace collected and the results with 1% duty cycle are shown in Figure 3.12. With only six phones, the trace (and number of contacts) is rather small. However, the results do show that R2 works well in a realistic environment with actual phones and AP beacons.

We now evaluate R2 using traces with more users.

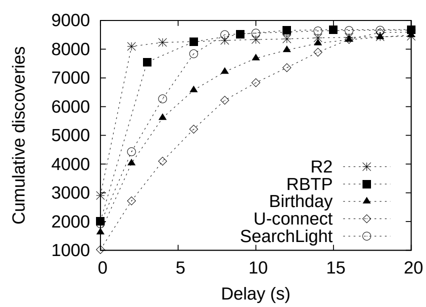



(a) Discovery delay

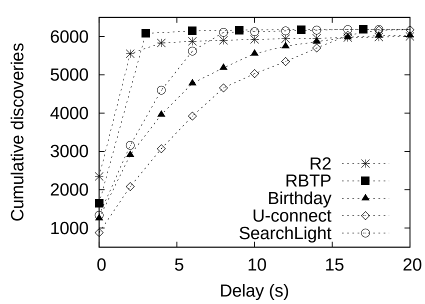
Figure 3.12: Neighbor discoveries on smartphones implementation. Cumulative number of neighbor discoveries in our six-phone experiments.




(a) Static (Synthetic)

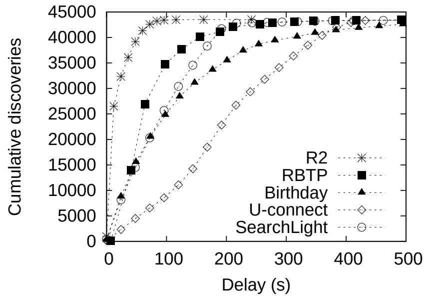



(b) Newton Food Court (Singapore)

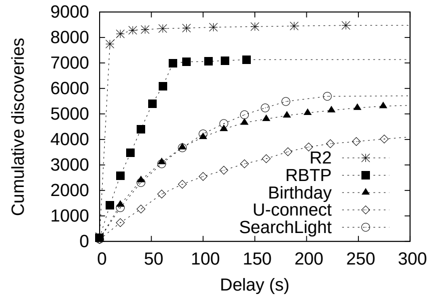



(c) State Fair (N.C. American)

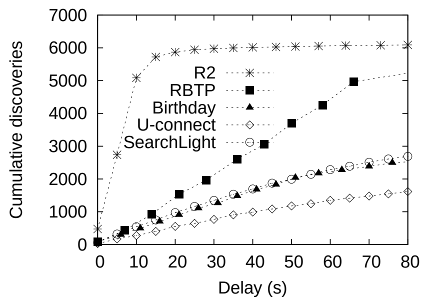
Figure 3.13: Cumulative number of neighbor discoveries (5% duty cycle)




(a) Static (Synthetic)

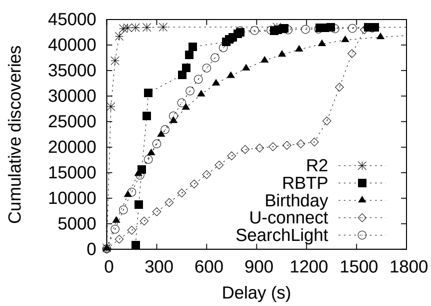



(b) Newton Food Court (Singapore)

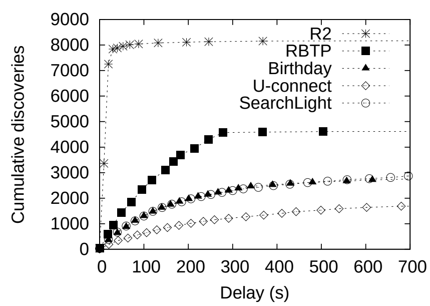



(c) State Fair (N.C. American)

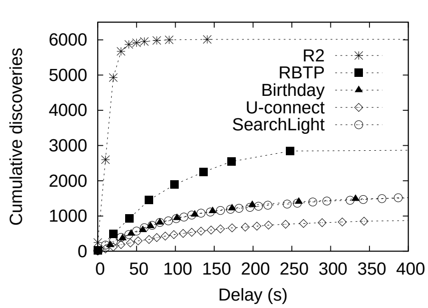
Figure 3.14: Cumulative number of neighbor discoveries (1% duty cycle)




(a) Static (Synthetic)




(b) Newton Food Court (Singapore)




(c) State Fair (N.C. American)

Figure 3.15: Cumulative number of neighbor discoveries (0.5% duty cycle)

### 3.7.2 Performance Evaluation

We compare R2's performance to the following protocols: (1) Birthday protocol, (2) Searchlight, (3) RBTP, and (4) U-connect. We evaluate R2 based on the three traces in Table 3.1, except for roller-net, which does not have location information for nodes. Instead, we generate a static trace, where there are 30 static nodes and their connections stay for sufficiently long time to allow all protocols to discover all neighbors.

For the state fair trace, we randomly placed APs in the area so as to include local synchronization source for R2. The phone-to-phone transmission range ($d$) is 50m and phone-to-AP range ($r$) is 100m.

We consider directional discovery delay. For two nodes 'A' and 'B', the delay for 'A' to discover 'B' is the time gap between their contact start time and the first time when 'A' directly receives a beacon from 'B'. In our evaluation, we use the default values listed in Table 3.3.

For RBTP, we set half of the energy consumption is used for neighbor discovery and the other half for synchronization. The overall synchronization time overhead for 3G is set to 40s of Wi-Fi power consumption and the synchronization period is set according to the duty cycle used. Clock drift varies randomly between $\pm$20ppm.

The results are shown in Figures 3.13 to 3.15 with duty cycles of 5%, 1% and 0.5%. Overall, all protocols perform better in absolute terms with larger duty cycle and performance degrades as duty cycle decreases. This is true for all traces and protocols. The key takeaway is thus not the absolute performance metrics, but the changes in relatively performance.

With a higher duty cycle of 5%, it can be observed that the asynchronous protocols, Searchlight and U-connect, perform relatively well and R2 does not perform significantly better than the other protocols. In fact, one can argue that Searchlight performs the best in the static scenario.

For the lower duty cycles like 1% and 0.5%, the results clearly show that synchronous protocols outperform asynchronous protocols. For all three traces, the two protocols with the best performance are always R2 and RBTP. As expected, R2 outperforms RBTP due to its significantly smaller synchronization overhead. The performance gap increases with decreasing duty cycle. For duty cycle of 0.5% and for the two traces with user mobility, RBTP can discover at most 60% of what R2 can discover and with a much longer delay. The relative performance of different protocols under varying duty cycles matches well with our analysis (see Table 3.4).

## 3.8 Summary

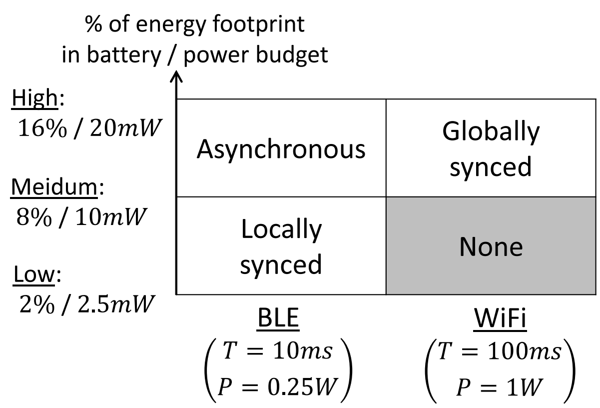



Figure 3.16: Neighbor discovery performance setting and performance.

This chapter discusses an essential building block for practical urban DTNs. It provides analytical models in which the performances of different categories of neighbor discovery protocols can be compared and evaluated against their key parameters, as shown in Figure 3.16. Figure 3.16 indicates the advisable cases for different types of neighbor discovery protocols. The one marked with 'none' is open to new designs. Our solution belongs to 'local sync' brunch. We have identified the performance potential as well as the limitations of existing synchronous protocols. Specifically, while a locally synchronized protocol can potentially operate at a low duty cycle while achieving low discovery delay, its synchronous nature can make it less efficient in a dense neighborhood. The proposed scheme R2, a locally synchronized protocol, can discover 90% of connections within 50s at the cost of 1% duty cycle. Its good scalability makes it work in dense mobile phone environment, which is often seen in urban setting. R2 saves power and captures more communication chances for mobile nodes. R2 builds a solid base for DTN routing and related applications, as we will discussed in later chapters.

# Chapter 4

# Resource efficient DTN routing

An efficient mobile node neighbor discovery solution, as discussed in Chapter 3, provides a solid base for DTNs by discovering more communication chances. In this chapter, we discuss how to route messages after discovering communicative neighbor nodes. The message routing algorithm plays a key role in many DTN solutions. When a routing algorithm can fast deliver messages from a source node to a destination node, it benefits all applications relying on message communication. On the other hand, slow message delivery can degrade the application performance and cause DTN applications less attractive. A practical DTN routing algorithm needs to conserve resources and targets more message deliveries with shorter latency.

In this chapter, we first give a literature review on DTN routing solutions. Then, we evaluate the accuracy of existing contact predictors and propose our contact prediction algorithm. After that, we discuss the solution on how to use contact prediction to control the number of replicas. Finally, the chapter ends with DTN routing performance evaluation.

## 4.1 Background

In the last decade since the work by Vahdat and Becker [111], the DTN research community has been focusing on designing routing algorithms which can deliver more messages in shorter latencies with constrained resources, such as limited transmission capability and message buffer. Many of these DTN routing algorithms can be divided into two broad approaches based either on (1) the limit placed on message replicas or (2) connection predictions.

Routing algorithms using the first approach limit the number of replicas for a message without considering future connections' characteristics and how easy message transmission between source node and destination node. Examples include 'single copy routing' [112], 'Spray and Wait' (S&W) [87], 'two hop' [85], gossip [88] and 'simple counting protocol' [84].

Routing algorithms using the second approach perform routing by selecting relays based on predictions on future connections. Examples algorithms include FRESH [69], 'Last Encounter' [68], PRoPHET [113], PER [114], RCM [64], SimBet [75] 'Bubble Rap' [115], MaxProp [72], 'delegation forwarding' [116], RAPID [66], Max-Contribution [67], and 'social feature space routing' [83]. Some algorithms are not included by the two approaches, e.g., Epidemic [111] and 'Spray and Focus' (S&F) [71] and EBR [89]. Epidemic duplicates messages without replica control nor connection predictions, while S&F and EBR incorporate the both approaches.

In the two types of approaches, there is a fundamental trade-off between the performance and the overhead. Algorithms that placed limits on replication tend to incur less overhead while algorithms that utilize contact information and duplicate messages without limit concentrate on performance. It is thus natural to design an algorithm that combines both the limit on message replicas and connection prediction. Intuitively, if a node carrying a replica of a message is very likely to meet the destination, the number of extra replicas needed should be small. Conversely, if no relay has a high 'confidence' in delivering the message, then the message needs to generate more replicas to increase the delivery chance. It is roughly like a method that an initial number of replica quota for a message is set and as the quota is used, i.e., some replicas of the message are spread to some relays, the quota is adjusted by the delivery probabilities of these relays.

The design of such an algorithm requires answers to the three questions: (1) How large should the initial replicas quota be? (2) How can future contact probability be predicted accurately? (3) How should the replica quota be adjusted according to contact probability to minimize overhead?

While existing works demonstrate the importance of exploiting replication control and contact probability prediction, they either fail to utilize both techniques simultaneously or fail to combine these two techniques effectively. For example, in MaxProp, RAPID and Max-Contribution, while messages with higher 'utility' are given higher transmission priority, the number of their replicas keeps increasing as long as there is available buffer space and transmission capacity. 'two hop routing', S&W, and 'simple counting protocol' control message replicas, but they do not predict future contacts. S&F uses replicas control in 'spray phase' and uses contact prediction in 'focus phase', but it keeps replica quota constant, independent from contact chances between message relays with destinations. As far as we know, no existing DTN algorithm integrates the two techniques for DTN routing. In addition, it also came as a surprise to us that many existing, widely used contact probability estimation algorithms (e.g., those used by PRoPHET, MaxProp and RAPID) are fairly inaccurate and in some cases, almost useless.

In this chapter, we discuss Plankton, a DTN routing algorithm that utilizes both message replication control and contact probability estimates. The two main components of Plankton are (1) a contact prediction algorithm that is based on both short-term bursty contacts and longer term, association based contacts, and (2) an algorithm to dynamically adjust replica quotas based on estimated contact probabilities and delivery probabilities. We show in our evaluation using seven contact traces that our contact prediction algorithm is much more accurate than existing algorithms. Further, we show that while Plankton incurs much lower communication overhead compared to S&W, MaxProp and RAPID with savings of 14% to 88%, it can also achieve similar, if not better, delivery ratios and latencies.

## 4.2 Literature review

The design of DTN routing solutions faces three main challenges: power constraints, intermittent connections, and incomplete view in a distributive scheme. Since we focus on resource efficiency, the work on DTN routing is reviewed in two categories: energy-oblivious routing algorithms and energy-aware routing algorithms. Energy-oblivious algorithms focus on delivering more messages within shorter latencies, while energy-aware routing algorithms aim to deliver messages with less energy consumption.

### 4.2.1 Energy-oblivious routing algorithm

As a seminal work in DTN routing research, routing algorithms under the assumption of contacts oracles access were articulated by Jain et al. [117]. The work adapted Dijkstra's shortest path algorithm by utilizing different cost functions. This type of algorithm includes Earliest Delivery(ED), Earliest Delivery with Local Queue(EDLQ), and Earliest Delivery with All Queue(EDAQ), which are respectively used for different levels of contact oracles. They provided comprehensive studies and a breakthrough on DTN routing, while they failed to consider energy efficiency and their assumption on contact oracle likely different from realistic settings.

The assumption on the access to contacts oracle was released by most following research. As the first work based on contacts prediction, PRoPHET [113, 73] is a milestone in DTN routing research. Nodes in PRoPHET collect past contacts, predict future contacts, and prioritize relays for messages by contact probabilities. In PRoPHET prediction algorithm, when a node 'A' encounters a node 'B', then the delivery probability from 'A' to be increased by a constant value. The more 'A' has encountered 'B', the high value the delivery probability between 'A' and 'B'. The delivery probability exponential decay over time since the last contact. The contact probabilities consider both one-hop delivery and multiple-hop delivery. Its simulation results indicate that predictions based relay selection can achieve shorter delivery latency and higher delivery ratio than random relay selection. PRoPHET only counts events of contacts, but it does not count contact durations. In E-PRoPHET [74], Li et al. adds in contact durations into prediction. PRoPHET and E-PRoPHET are energy oblivious, as nodes greedily use their bandwidths and buffers.

Similar to PRoPHET, in MaxProp[72], John et al. proposed a new approach for contact prediction and relay selection. MaxProp prioritizes recently encountered nodes as relays. MaxProp computes delivery likelihood (DL) (i.e., delivery probability) between two nodes as follows. At the beginning when ‘A’ only encounters ‘X’, the DL for ‘A’ to ‘X’ is 1. Later, when ‘A’ encounters ‘Y’, then DL ‘A’ to ‘Y’ is also 1. After normalization, the DLs for (A,Y) and (A,X) both are 0.5. After this, ‘A’ encounters ‘X’ again, then the DL for (A,X) is increased to 1.0 (e.g., $0.5 + 1$). After normalization, the DL for (A,Y) and (A,X) respectively are 0.75 and 0.25. The DL values only prioritize the message transmission order, but it does not control the total replicas of a message. Instead, a node greedily uses resources, it stops transmitting messages to the encountered node only when it has no messages to transmit or the connection tears down.

Following MaxProp, Balasubramanian et al. proposed a far-reaching solution RAPID (2007) [118, 66]. RAPID predicts contact by assuming inter-contact-length follows exponential distribution. Different from previous solutions, RAPID counts in both the delivery probabilities and the number of existing message replicas when it prioritize message transmission. DTN routing problem is tackled as resource optimization problem. Similar to the aforementioned work, RAPID exhaustively uses resources without considering power consumption. However, RAPID has two significant contributions. First, the work viewed DTN routing algorithms as an optimization problem under resource constraints and provided metric-based solutions. Second, RAPID was tested in a real DTN system with forty nodes, which is the first solution that was implemented on devices mounted on 40 buses.

Besides these aforementioned solutions, many other works also proposed energy oblivious routing solutions. These works mainly include Drop-Least-Encountered algorithm(DLE) [119], SOLAR(Sociological Orbit aware Location Approximation and

Routing) [120], Routing in Cyclic Mobility (2008) [64], SMART [81], and max-Contribution [67].

Another new trend in energy oblivious routing algorithm is based on social network. They utilize social network analysis techniques to find proper relays for messages. This set of routing algorithms contains 'Label routing' [78], SimBet [75], SimBetAge [76], Bubble Rap [115] and 'Social Feature Routing' [83].

### 4.2.2 Resource-aware routing algorithm

Different from energy-oblivious routing algorithms, energy-aware routing algorithms target good performance with efficient use of energy.

Two-hop routing (2002) [85] is probably the earliest energy-aware routing algorithm for an ad hoc network, which is also applicable to DTNs. In two-hop routing, a source node duplicates its messages to the encountered nodes, while a relay node forwards a message only to the message's destination. Nodes save energy because they transmit fewer replicas. It is questionable as its performance are limited in the contexts where a node can reach most other nodes within two hops.

Following two-hop routing, Spyropoulos et al. proposed resource aware routing algorithm 'Spray and Wait' [87] and its extension 'Spray and Focus' [71]. In 'Spray and Wait', when a source node transmits a message to a destination, it sets a replica quota ($Q$) for the message. The quota defines the maximum relays (i.e, the maximum transmission) of the message. When $Q$ quota are used, then no relay will replicate the message to other nodes except the destination node. The name 'spray and wait' comes from the two process stages: spray $Q$ quota and wait for the destination. By this way, the source node can control the maximum bandwidth resource, energy resource and buffer resources the message can use. The late proposed solution 'Spray and Focus' [71], modifies the 'wait' stage. A relay node in the 'wait' stage can forward (note replicate) the replica in hands to a new node that is more likely to encounter the destination node.

Along the same line of controlling the number of message replicas, Walker et al. proposed 'simple counting protocol (SCP)'[84]. The core idea of SCP is to avoid over generate replicas by balancing the number of replicas of different messages. In SCP, a node removes the most often seen message replicas from its buffer, and duplicates the message replicas that are least frequently seen. This scheme balances the number of replicas of different messages, and saves power by avoiding duplicating too many replicas. However, SCP fails to exploit future contact predictions and select proper relays for messages.

Besides the above algorithms, many other energy-aware routing algorithms were also published. These works comprise "random single copy routing" [112] and "gossip-based routing" [88].

## 4.3 Gap and the contribution

### 4.3.1 The gap in resource efficient DTN routing solution

Though extensive research have achieved impressive progress in DTN routings, we find inadequacy in resource efficiency routing solutions. For example, although contact prediction has been widely used to improve routing performance, it is not well exploited to improve resource efficiency. Prediction accuracy determine the utility of the prediction in messages replicas control. A node can save resources by avoiding generating redundant messages if it can know reliable contact prediction. Prediction reliance measurement, however, arouse little research interest. To our best knowledge, no previous work has evaluated prediction accuracy.

The detailed gaps can be categorized into the following points:

**Algorithms for flexible control of replica are required.** Current routing algorithms set the maximum copies of a message as a fixed quota. By using fixed quota, messages for different source-destination pairs equally share bandwidth and buffer resources. Messages for different source-destination pairs, however, should be treated with different resources, as the difficulty of delivering them are different. Messages for some node pairs require fewer replicas if their sources are 'near' to the destinations, which means that they can be easily delivered; some other messages require more replicas if the source nodes are 'far' from the destinations, i.e., they are more difficult to delivery. The rigid control of replica quota can cause too many or too few replicas. Contact prediction naturally plays a crucial role in replica control, but it has not aroused much interest to current DTN research.

**The measurement on contact prediction reliance is not well investigated.** Message routing decision based on reliable contact prediction can help speed up message delivery, while message routing decision based on unreliable one might make wrong message replication, which can be worse than routing without contact prediction. To save resources without spoiling message delivery performance, the reliable contact prediction is especially important. Thus, the measurement on contact prediction reliance is critical.

**No robust yet energy efficient routing algorithms are proposed.** Proposed algorithms on energy efficient routing are often tested on a small number of synthetic traces, which often hides the limitations occur in real traces tests.

### 4.3.2 Contribution

This chapter discusses energy efficient routing algorithms that can perform well in a wide range of contacts under different environments. The main contributions include the follows.

**Control message replicas with reliable contact predictions.** The messages with different source-destination pairs should be assigned different mounts of resources. This can avoid both over-duplicating messages and under-duplicating messages.

**Propose a novel algorithm for measuring contact prediction accuracy.** Prediction reliance is measured by its utility in routing algorithms. The proposed measurement algorithm is expected to work well on a wide range of contacts and prediction algorithms.

| Trace | No. of nodes | Transmission range | Scenario |
|---|---|---|---|
| RollerNet[40] | 62 | Bluetooth | outdoor roller blading |
| Haggle IC06[121] | 98 | Bluetooth | conference |
| Reality[39] | 100 | Bluetooth | campus |
| SF taxi[122] | 500 | 50m | city taxi |
| Seattle Bus[123] | 1200 | 50m | city shuttle bus |
| RPGM[124] | 100 | 10m | reference point group mobility |
| Random Way Point (RWP) | 100 | 10m | random and free node movement |

Table 4.1: This table contains the traces often used by DTN research community. The RollerNet trace is sport trace collected when a group of people having outdoor blading competition, which has dense contacts. The Haggle IC06 trace are collected at the conference of InfoComm06, which has the feature of contact of clusters. The Reality trace is collected on MIT campus, which has sparse contacts. SF taxi trace contains taxi location updates, which was collected in SF bay area. Seattle bus contains locations of city buses, which has a large number of buses which sparse contacts. RPGM and Random Way Point traces are the synthetic ones.

**Propose a resource efficient routing scheme with robust performance.** We propose components required by a robust and resource efficient algorithm and integrate them into a routing algorithm. These components include (i) replica control scheme, (ii) a novel and accurate contact prediction scheme, which is measured by the prediction reliance measurement algorithm, and (iii) a resource efficient distributive scheme for collecting replica information.

## 4.4 Evaluation of current contact prediction algorithms

### 4.4.1 Contact prediction of PRoPHET, MaxProp, and RAPID

First, we evaluate three representative contact prediction algorithms in PRoPHET, MaxProp, and RAPID. They can be briefly described as follows. Let a node $v$'s probability of encountering a node $u$ be $P_{v,u}$.

- In *PRoPHET*, $P_{v,u}$ additively increases when $v$ has a new contact with $u$, $P_{v,u} = P_{v,u\,(old)} + (1 - P_{v,u\,(old)}) \times p_{init}$, and multiplicative decreases by $k$, the time since the last contact, $P_{v,u} = P_{v,u\,(old)} \times \gamma^k$. We set $p_{init}$ to 0.75 and $\gamma$ to 0.98 as used by [113].

- In *MaxProp*, $P_{v,u}$'s initial value is zero. $P_{v,u}$ on $v$ halves when $v$ encounters a node other than $u$, and increases to $(1 + P_{v,u})/2$ when $v$ encounters $u$.
- *RAPID* uses a prediction algorithm based on exponential inter-contact distribution. The probability is $Pr(encounter(v, u) < t) = 1 - e^{-\lambda t}$, where $1/\lambda$ is the average inter contact length (ICL) between $v$ and $u$.

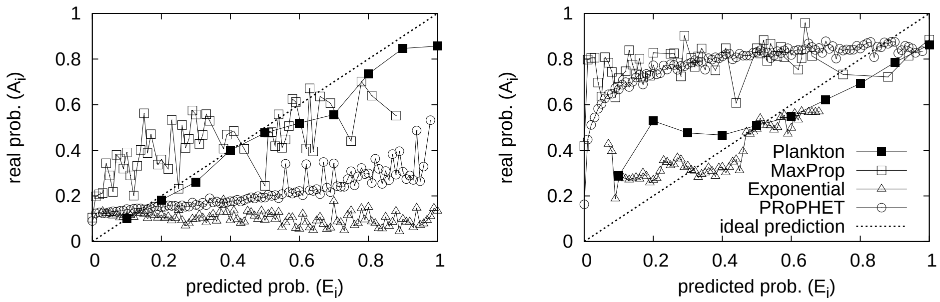
When two nodes meet, after meta-data exchange, each node predicts the probabilities of encountering other nodes in the future. Such computations are performed for each contact. The total number of estimates can be up to the product of the number of contracts and the square of the number of nodes in the network.

Let it be $e_i$. For each $e_i$ estimated at time $t$, we can consult contact oracles whether the predicted contact actually occurs or not before the time $t + d$, where $d$ is the time period of interest. Based on this answer, we form a tuple of $\langle e_i, a_i \rangle$, where $a_i$ equals zero if the contact does not occur or equals one if it does.

These tuples are then placed into $b$ bins, and each bin is of size $1/b$. The tuples with $e_i$ between $\frac{j}{b}$ and $\frac{j+1}{b}$ are placed into bin $j$, $0 \leq j < b$. For each bin $j$, we compute the accuracy $A_j$ as $\frac{\sum_{e_i \text{ in bin } j} a_i}{\text{No. of elements in bin } j}$. We exclude estimates collected during warm-up and cool-down periods. Also, if the number of estimates in a bin is too low, say below 30, we do not consider the result of the bin.

Finally, we use the midpoint of the bin as the value of the estimate. Let this value be $E_j$. Hence, after the processing, we get a series of data points, $\langle E_0, A_0 \rangle \cdots \langle E_{b-1}, A_{b-1} \rangle$. If the estimate can provide useful information to routing algorithms, we should expect the sequence $\{E_0 \cdots E_{b-1}\}$ and $\{A_0 \cdots A_{b-1}\}$ have strong correlation. This evaluation can apply to the scheme where the predicted values are utilities instead of contact probabilities, e.g., MaxProp, PRoPHET.

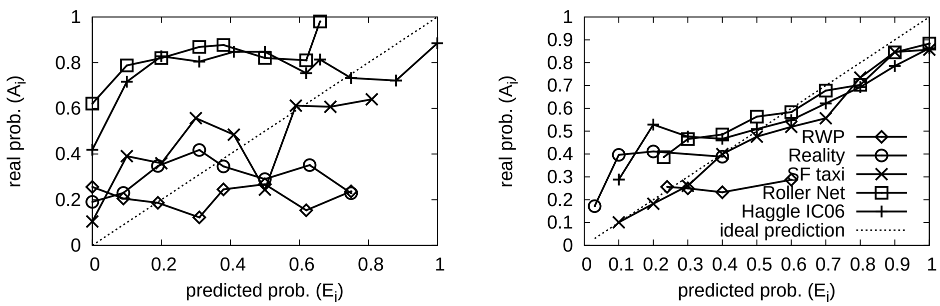
In the approach described, we only evaluate one-hop contact probability. Since multi-hop delivery probabilities are based on one-hop contact probabilities, one-hop




(a) Prediction on the SF taxi trace.

(b) Prediction on the Haggle InfoCom2006 trace.

Figure 4.1: Contact prediction accuracy of different contact prediction algorithms. The two figures share the same label listed in Figure 4.1(b).




(a) MaxProp predictions on different traces.

(b) Plankton prediction on different traces.

Figure 4.2: Prediction accuracy on different traces of MaxProp and Plankton. The two figures share the same label listed in Figure 4.2(b).

prediction evaluation is sufficient.

Connection predictions were evaluated on seven traces in Table 4.1 with Different settings of $d$ ($1800s$, $3600s$, and $7200s$) and $b$ (10, 50, 100, and 200). As the results from different settings are similar, we will only show the results where $d$ equals $3600s$ and $b$ equals 100 and 10.

First, we present the results of different algorithm on a particular trace, namely the SF taxi trace. Figure 4.1(a) shows the plot of $E_i$ versus $A_i$. The case for ideal prediction, where $E_i$ equals $A_i$, is included for reference. The results show that all three algorithms are fairly inaccurate. The Exponential algorithm is the most inaccurate. MaxProp and PRoPHET generate more accurate results than Exponential-based prediction does, but their predictions are still noisy.

| Trace | Exponential | PRoPHET | MaxProp | Plankton |
|---|---|---|---|---|
| RollerNet | -0.58 | 0.92 | 0.60 | 0.99 |
| Haggle IC06 | 0.84 | 0.75 | 0.37 | 0.95 |
| Reality | -0.43 | -0.35 | 0.16 | 0.60 |
| SF taxi | -0.26 | 0.86 | 0.68 | 0.99 |
| Seattle Bus | -0.22 | 0.65 | 0.57 | 0.94 |
| RPGM | -0.44 | 0.98 | 0.52 | 0.87 |
| RWP | 0.08 | 0.04 | 0.13 | 0.64 |

Table 4.2: Correlations between tested predictions and ideal predictions.

To view the evaluation on different traces, we plot MaxProp's prediction on different traces in Figure 4.2(a). It shows that $E_i$ is often far away from the respective $A_i$ on most traces.

We quantitatively measure prediction reliability by computing the Pearson correlation between the estimate values and the ideal prediction. A strong positive correlation indicates a reliable prediction. Table 4.2 lists the correlations from the three algorithms on all traces. We can make the following observations.

First, while correlations vary among different traces, these prediction algorithms are often unreliable. In many cases, the Exponential algorithm has negative correlations. The reliability of PRoPHET varies significantly from -0.35 (Reality) to 0.98 (RPGM). The results from MaxProp are better, but the correlations still vary from 0.16 (Reality) to 0.68 (SF taxi).

Second, the results indicate that while routing algorithms using these contact prediction schemes can achieve high message delivery ratios and/or low latencies, the good performance sounds unlikely due to accurate contact probability estimations.

### 4.4.2 Contact prediction of Bubble Rap

The investigation on prediction accuracy of *Bubble Rap* is done separately. Instead of pair-wise contact probability, Bubble Rap utilizes nodes' centrality for selecting relays. In Bubble Rap, when $u$ with message $m$ encounters $v$ without $m$, $u$ either duplicates $m$ to $v$ ($v$ is a relay) or does not duplicate $m$ to $v$, depending on the result of centrality

| Algorithm | Roller | IC06 | Reality | SFtaxi | SeattleBus | RPGM | RWP |
|---|---|---|---|---|---|---|---|
| BubbleRap | 0.24 | 0.19 | 0.18 | 0.12 | 0.02 | 0.20 | 0.25 |
| Plankton | 0.29 | 0.31 | 0.55 | 0.30 | 0.58 | 0.25 | 0.25 |

Table 4.3: Prediction comparison between BubbleRap and Plankton.

comparison and community information. For Bubble Rap, we record the duplication decision, and offline check if the relay and destination actually meet within a given delay $d$. A good estimator should make a larger proportion decisions where the relays do meet the destinations.

Table 4.3 shows the utility of the decision in terms of how often the relays meet the destination. 2% to 25% replication decision by 'Bubble Rap' results in one-hop delivery, with an average of 17% over all seven traces.

## 4.5 Plankton contact prediction

Plankton's contact prediction is based on two main ideas. First, links (i.e., contacts or connections) can be classified into either *strong* or *weak* links depending on how likely a message sent on this link can be delivered to the destination. Second, strong links can be identified through association relationships that can be observed at different time scales.

In the rest of this section, we first discuss how weak and strong links are defined, followed by how strong links are discovered and finally how these estimations are combined into a single contact prediction.

### 4.5.1 Weak links ($\rho$)

The comparison of weak link with strong link is relatively. As the name suggests, a message sent on a weak link is less likely to be delivered to the destination than one sent on a strong link. Hence, the first task is to define what a weak link is. The approach taken by Plankton is to initially classify all links as weak links. A link is reclassified as strong later using the association relationships to be defined below.

| Symbol | Interpretation |
|---|---|
| $\rho$ | pairwise average delivery probability |
| $P_{v,u}$ | the contact probability between $v$ and $u$ |
| $P^a_{v,u}$ | the contact probability between $v$ and $u$ based on associated contacts |
| $P^b_{v,u}$ | the contact probability between $v$ and $u$ based on bursty contacts |
| $P_l(m)$ | $m$'s local delivery probability |
| $P_g(m)$ | $m$'s known maximum local delivery probability |
| $d_m$ | $m$'s destination node |
| $p_{tie}$ | the threshold value for dividing strong ties and weak ties |
| $p_{ach}$ | the threshold value for achievable delivery probabilities |

Table 4.4: Summary of symbols.

Next, since all links are initially classified as weak, we approximate the probability that delivery on a weak link can result in successful message delivery as the average contact probability, $\rho$, between any two nodes within a time window ($win$). A good value for $win$ is the remaining lifetime of the message. With the assumption that contacts are i.i.d., $\rho$ can be approximated as follows:

$$\rho = \frac{\text{avg. no. of encountered nodes within } win \text{ by a node}}{\text{total no. of nodes} - 1}. \tag{4.1}$$

$\rho$ provides a baseline for contact probabilities. The value of $\rho$ varies with mobility. For example, a node in the Roller-net trace can encounter more than 20% of all nodes within 900s, while a node can only encounter fewer than 2% of all nodes within 900s for the Seattle bus trace. The value of $\rho$ can be either estimated online or computed off-line through historical information.

Next, we will describe how strong links can be identified through association relationships

### 4.5.2 Prediction by recent contacts ($P^b_{v,u}$)

Bursty and self-similar events have been well investigated and exploited in many fields, such as web caching and Internet traffic modeling. In real-life mobility pattern, once two nodes are in the same proximity, there is good chance they are still in proximity in the near term. These could be because these nodes are two persons in the same seminar room attending a talk, two passengers on the same subway train or two vehicles traveling along a road in the same direction. In DTN routing, such bursty contacts have also been exploited in last contact based routing [68, 71]. In these algorithm, only the most recent contact is taken into account.

In Plankton, for computing contact probability based on recent contacts, we study the number of most recent intervals that a node have appeared. Let $P^b_{v,u}$ denote the contact probability between $v$ and $u$ based on recent contacts. To compute $P^b_{v,u}$, $v$ checks the most recent $n$ time intervals, where each interval has a duration of average pairwise inter-contact length (ICL). If $v$ encounters $u$ in $n_u$ out of the $n$ intervals, $P^b_{v,u}$ equals $n_u/n$.

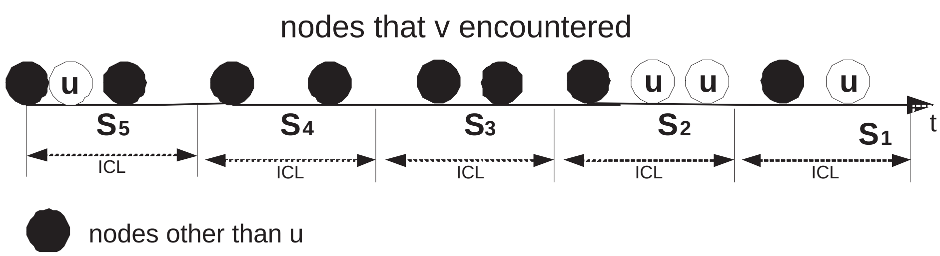



Figure 4.3: Prediction by bursty contacts.

Figure 4.3 illustrates an example. $v$ computes $P^b_{v,u}$ at $t$ and $n$ is set to be five. $P^b_{v,u}$ equals $3/5$, as $v$ encounters $u$ in three out of the five intervals.

$P^b_{v,u}$ provides an estimate based on short-term contact features, and its implementation is straightforward. We use average pairwise ICL, instead of an absolute time length because ICL is adaptable to different traces. ICL is easy to compute by a node with local contact information. Another parameter Plankton uses is $n$. The settings of $n$ has little effect on performance as long as it is not too big (larger than 20) or too small (less than five). We set it to 10 in our evaluation.

### 4.5.3 Prediction by indirect associations ($P^a_{v,u}$)

The intuition behind prediction by indirect association is as follows. If whenever $v$ encounters $w$, it often encounters $u$, then it is likely that once $v$ meets $w$, the probability of meeting $u$ is high. Such association captures indirect relationship among nodes that occurs over longer time scale. It exploits community relationships among nodes without the need to perform community clustering or search for nodes with high centrality as is done in BubbleRap.

We write the contact probability between $v$ and $u$ through the indirect association with $w$ as $P^a_{v,w,u}$, which can be computed by the following steps. $v$ first locates the recent $n$ non-overlapping intervals with the length of ICL that start with an encounter with $w$. Next, $v$ counts the intervals that also contain at least one encounters with $u$ and let this number be $n_u$. $P^a_{v,w,u}$ is computed as $n_u/n$. Since the association can happen through different nodes, $P^a_{v,u}$ is taken as the maximum value over all indirect associations through the nodes that $v$ recent encounters. $w$ is $v$'s recent encounter if the time since the last contact between $w$ and $v$ is less than ICL, which uses the same length as the interval for checking indirect association. Indirect association probability can be computed as follows:

$$P^a_{v,u} = \max_i\{P^a_{v,i,u}\} \text{ } (i \text{ is } v\text{'s recent encounter.})$$

Figure 4.4 illustrates an example, where $n$ equals five. In the past five non-overlapping intervals starting with an encounter with $w$, $v$ encounters $u$ in four intervals. Thus, $P^a_{v,w,u}$ equals $4/5$.

By taking the maximum value over all possible associations, we follow the approach taken in popular universal prediction algorithm for data compression [125] and mobile nodes location prediction [126], which looks for the longest subsequence.

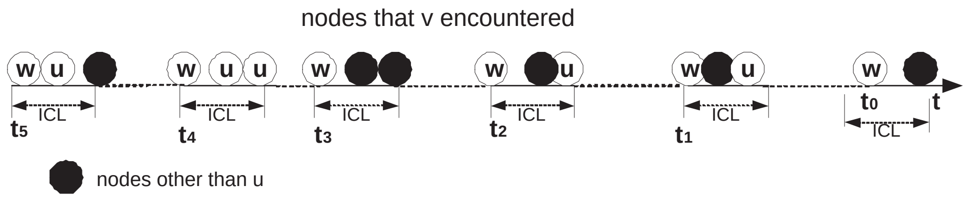



Figure 4.4: Prediction by associated contacts.

### 4.5.4 Combination of different predictions

By the above discussion, $v$ has three estimates for the contact probability between $v$ and $u$, based on the three predictions used, namely $\rho$, $P^b_{v,u}$ and $P^a_{v,u}$. Given that $P^b_{v,u}$ and $P^a_{v,u}$ are meant to discover strong links, they are only useful or meaningful when their values are larger than $\rho$.

Next, while it is possible to combine the values of $P^b_{v,u}$ and $P^a_{v,u}$ by assuming that they are independent, we choose not to take this assumption for the following reasons. First, there is no strong reason to believe that these estimates are independent. Second, as we will explain later, due to our approach to dynamic quota adjustment based on contact probability, we prefer conservative estimates. The quota adjustment can tolerate false negative to some extent, but we need low false positive for high contact probability. We thus compute the final contact probability between $v$ and $u$ $P_{v,u}$ as:

$$P_{v,u} = \max\{\rho, P^b_{v,u}, P^a_{v,u}\}. \tag{4.2}$$

### 4.5.5 More discussion on contact prediction estimators

These three estimators are complementary. The first estimator, $\rho$, explores the contacts that are difficult to predict, e.g., the contacts having few past contacts of respective node pair to exploit. While these contacts are 'unreliable', they dominate more than 80% of all contacts in the mobility traces we investigated. Many existing algorithms either simply ignore such contacts by setting the delivery probability to zero or incorrectly estimate the probability because of the lack of past contact information. These contacts however can significantly contribute to messages dissemination since there are so many of them [127]. By exploiting average contact chances by $\rho$, Plankton can efficiently utilize these contacts without much historical contact information on a specific nodes pair.

The other two estimators capture different features of contact patterns when past contacts can provide sufficient contacts information. Recent contacts based prediction captures burstiness of contacts by investigating short term contacts, while indirect association based prediction captures contact association by investigating long term contacts. These associations cover different set of nodes and exploit different association behaviors.

### 4.5.6 Evaluation

We evaluate the accuracy of Plankton prediction and compare its performance to other predictors presented in Chapter 4.4 and the results are shown in Figure 4.2. Results in Figure 4.1(a) and Figure4.1(b) show that Plankton can achieve much more accurate predictions than the other three estimators on SF taxi trace and Haggle InfoCom2006 trace. The comparison between Figure 4.2(b) and Figure 4.2(a) shows that Plankton achieves much more accurate prediction than MaxProp on all tested traces. From Table 4.2, we can see that Plankton outperforms the other three estimators on all traces. It achieves correlation equal to or larger than 0.87 on all traces except Reality (0.60) and RWP (0.64). Even for these two traces, Plankton achieves much higher correlations than the other estimators.

As Bubble Rap does not make contact prediction but instead make binary decision on whether a message should be forwarded to another node, we compare the utility of the decision of Bubble Rap and Plankton based on the ratio of message delivery for every message forwarded. Table 4.3 compares the utilities of the forwarding decision in Bubble Rap and Plankton. It shows that the utility of the Plankton is always better or equal to Bubble Rap. Except for RWP where there is no difference, the improvement varies from 20% for RollerNet to 2800% for the Seattle Bus trace.

## 4.6 Routing algorithm: Plankton

### 4.6.1 Overview

To deliver a message $m$, the source node first assigns $m$ a replica quota by $\rho$. As $m$ is duplicated, the quota is divided among the replicas. A novel feature of Plankton is that it decreases the replica quota when $m$ is duplicated to a node that has high contact probability with the destination.

### 4.6.2 Initial quota setting

The replica quota $\mathcal{Q}$ defines a message's maximum delivery probability. When $\mathcal{Q}$ is fully utilized, i.e., $\mathcal{Q}$ replicas are transferred onto $\mathcal{Q}$ relays, the message gains maximum delivery probability. In order to accurately set $\mathcal{Q}$, the source needs to know who will be the relays and their contact probabilities to the destination, which is generally infeasible in DTNs. Plankton estimates $\mathcal{Q}$ using the delivery probability of weak link $\rho$.

Suppose that $\mathcal{X}$ replicas of $m$ are duplicated onto $\mathcal{X}$ relays $(x_1, x_2, \cdots, x_{\mathcal{X}})$, whose delivery probabilities ($P_{x_i,d_m}$) may be lower or higher than $\rho$. It is reasonable to assume that $\sum_{i=1}^{\mathcal{X}} P_{x_i,d_m} \approx \mathcal{X} \times \rho$ for sufficient large $\mathcal{X}$.

When a source node generates $m$ and estimates $\mathcal{X}$, it does not know which node will carry $m$ as a relay in future. With a simple assumption on the independence of relays' delivery chances, the following approximation can be made:

$$P = 1 - \prod_{i=1}^{\mathcal{X}} \left(1 - P_{x_i,d_m}\right) \quad \geq 1 - (1-\rho)^{\mathcal{X}}$$

By above inequality, we can get $\mathcal{X} \leq \log(1-P)/\log(1-\rho)$. Since the actual quota $\mathcal{Q}$ cannot be larger than $\mathcal{N}$, the number of total nodes, we can get:

$$\mathcal{Q} = \min\{\frac{\log(1-P)}{\log(1-\rho)}, \mathcal{N}\}. \tag{4.3}$$

Note that $\mathcal{Q}$ is a conservative approximation without the information of online contact probability when replicas are generated. Replica quotas are accordingly reduced once a node duplicates $m$ to a relay that has high delivery probability, which we discuss as follows.

**Input**: a contact event between $u$ and $v$
**Output**: a set of triggered actions
$u$ and $v$ exchange the meta-data, including:
(I) message acknowledgements
(II) contact probabilities
(III) known $P_g(m)$
(IV) the message list in buffer.
On receiving (I):
remove delivered messages and update $p_{ach}$
On receiving (II):
**foreach** $m$ *in* $u$*'s buffer* **do**
  update $P_l(m)$ by formula 4.5
  $P_g(m) \leftarrow \max\big(P_g(m), P_l(m)\big)$
  **if** $P_g(m) > p_{ach}$ **then**
    set the quota for $m$ to *zero*
  **end**
**end**
On receiving (III):
**foreach** *received* $P_g(m)$ **do**
  $P_g(m) \leftarrow \max\big(P_g(m)_{rcvd}, P_g(m)\big)$
**end**
On receiving (IV):
generate duplication list by Alg. 4.2

**Algorithm 4.1**: Metadata exchanges and routing actions.

### 4.6.3 Dynamic quota adjustment based on contact probability

Plankton defines two types of links, strong links and weak links. A node pair $(u, v)$ has a *strong link* if the contact prediction for $u$ and $v$, $P_{u,v}$, is larger than $\rho$. Otherwise, it is a *weak link*. The typical scenario of checking strong links is as follows. Node $u$ has a message for the destination $v$. Node $u$ encounters $w$. Node $u$ checks whether $w$ has a strong link with $v$ or not.

The judgement behind the division of strong links and weak links is the reliability of contact prediction. Labeling two nodes with weak links does not mean their contacts are useless in message forwarding, but it indicates that it is difficult to attain reliable prediction on their contacts. It also indicates that two weak links may have minor difference in contact chances, but it is difficult and likely unworthy to differentiate them. Strong links have opposite properties. Strong links can be reliably predicted and two strong links can have dramatically different contact chances, which is knowable and worthy to know. Thus, weak links and strong links have different contributions to routing performance. Message duplications over weak link helps to seek strong links. Large number of replicas via weak links themselves certainly can generate considerable large chances of message delivery. Strong links mainly contribute to direct deliver messages. Therefore, Plankton wields weak links and strong links in different policies as follows.

Suppose node $v$ buffering message $m$ encounters $u$ that do not buffer $m$. If $u$ and $d_m$ have a weak link, $v$ performs duplication procedure same as binary split S&W. If $u$ and $d_m$ have a strong link, $v$ always duplicates $m$ to $u$ when $v$'s quota for $m$ is larger than zero. $v$ adjusts the quota by computing the number of weak links that equal the strong link between $u$ and $d_m$, measured by the chance of delivering $m$. Let $h$ be the number of weak links. We can get the contact probability between $u$ and $d_m$, $P_{u,d_m}$, equals to $1-(1-\rho)^h$, and $h = \log(1-P_{u,d_m})/\log(1-\rho)$. Let $q$ denote $m$'s quota on $v$. $v$ sets its new quota to one and $u$ sets its new quota to $q'$ as computed by the following equation:

$$q' = \max(1, q-h). \tag{4.4}$$

The replica quotas adjustment and send-list generation is listed in algorithm 4.2. We would like to highlight the follows. First, the above quota adjustment scheme requires low false positive in contact prediction. This explains our preference for conservative contact probability estimates by Equation 4.2. Second, the number of strong links do not have to be large. A small number of strong links is sufficient.

### 4.6.4 Dynamic quota adjustment based on delivery probability

Besides reducing replica quotas through strong links, Plankton stops replication when it detects that replicas generated are sufficient to meet the delivery probability achievable by available DTN's resources. To perform this task, Plankton estimates the delivery probability by all generated replicas, as well as set a *realistic* target for delivery ratio.

Delivery probability of $m$ is computed based on the contact probability on all relays of $m$. Plankton does not utilize multiple-hop delivery probability for three reasons. (1) DTNs are small world network [128, 80], which means messages can be delivered in a small number of hops. (2) Our measurements and evaluations have indicated that multiple-hop delivery probabilities are often highly inaccurate. (3) Multiple-hop delivery probability estimation requires expensive exchange pairwise contact probabilities. Thus, we compute delivery probability from multiple relays by direct delivery. This may underestimate the true delivery probability by ignoring multiple-hop delivery chances, but we can have a conservative estimate from which we can safely reduce quota. Also, note that while the multi-hop delivery probability is not utilized, multiple-hop transmissions happen in Plankton.

The local delivery probability of a message $m$ computed by a node $u$ is:

$$P_l(m) = 1 - \prod_{v \in V} \left(1 - P_{v,d_m}\right),$$

where $V$ is the set of nodes that $u$ has encountered and known to have buffered $m$. Nodes exchange their local estimates $P_l(m)$ and actual estimated delivery, $P_g(m)$, is simply the largest among all known local estimates.

Sometimes, the target delivery ratio cannot be met. This can be due to node specific behavior (e.g., the destination node is isolated) or network congestion which would make the increase in replicas a bad option. Plankton tries to approximate the achievable delivery ratio by computing achievable delivery probability by averaging $P_g(m)$ of delivered messages. Duplication for $m$ stops if $P_g(m)$ is larger than the achievable delivery ratio measured.

**Input**: $v$'s buffer list
**Output**: sorted send-list for $v$: $list$
**foreach** $m$ *in* $u$*'sbuffer, not in* $v$*'s buffer* **do**
  **if** $v = d_m$ **then**
    $gain(m) \leftarrow 1$, add $m$ to $list$
  **else if** $m$ *'s quota* $> 0$ **then**
    **if** $P_{(v,d_m)} \leq \rho$ *and* $m$*'s quota* $> 1$ **then**
      $gain(m) \leftarrow (1 - P_g(m)) \times P_{(v,d_m)}$
      add $m$ to $list$
      evenly split $m$'s quota
    **end**
    **if** $P_{(v,d_m)} > \rho$ **then**
      $gain(m) \leftarrow (1 - P_g(m)) \times P_{v,d_m}$
      add $m$ to $list$
      compute quota by Equation 4.4
    **end**
  **end**
**end**
decreasingly sort $list$ by $gain(m)$

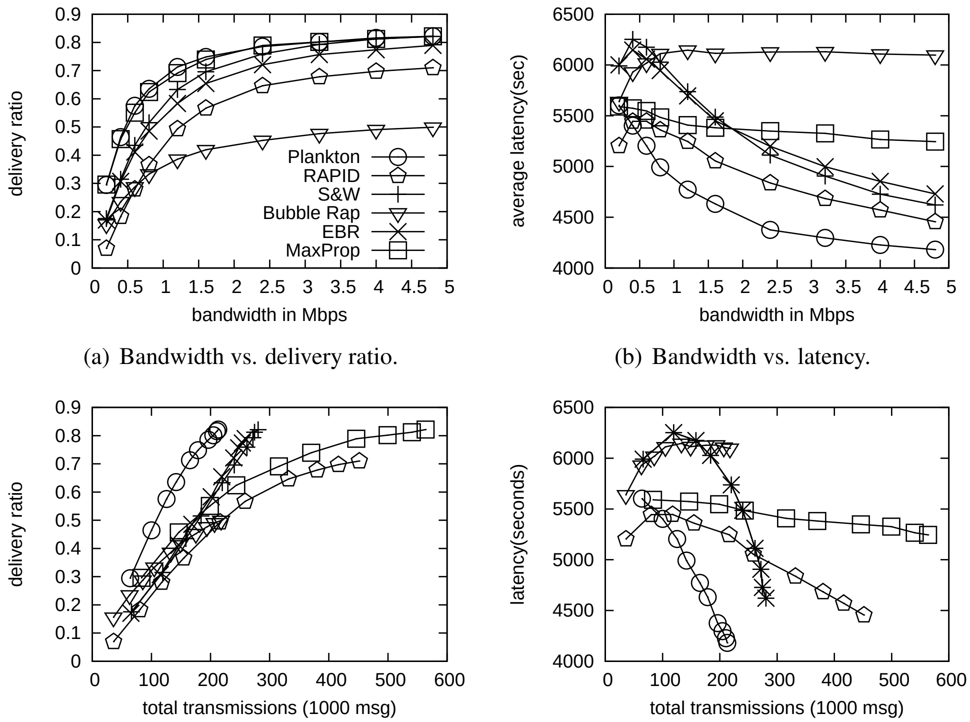
**Algorithm 4.2**: Node $u$ Generates duplication list for $v$.

## 4.7 Performance evaluation

We evaluate Plankton with an event-based simulator adapted from ONE [129]. We simulate a wide range of buffer and bandwidth settings on extensive traces to ensure that the simulation settings are more realistic [130]. Message lifetime is used to describe the validity period of a message since it is created. The message life is set to 1,000s for the Roller net trace, and 20,000s for the Reality trace, and 7,200s for all other traces. Message sizes are uniformly distributed between 1KB and 100KB. Warm-up and cool-down periods are set and the metadata communication overhead are simulated.

We compare Plankton with MaxProp, RAPID, S&W, EBR and Bubble Rap. For fair comparison, acknowledgement packets are flooded for all algorithms since MaxProp and RAPID use acknowledgements. The quota for S&W and EBR is set to be the same as the initial quota used by Plankton.

DTN routing performance is often measured by delivery ratio (DR) and average latency. DR refers to the ratio of delivered messages to unique messages to be delivered.




(a) Bandwidth vs. delivery ratio.

(b) Bandwidth vs. latency.

(c) Transmissions vs. delivery ratio.

(d) Transmissions vs. latency.

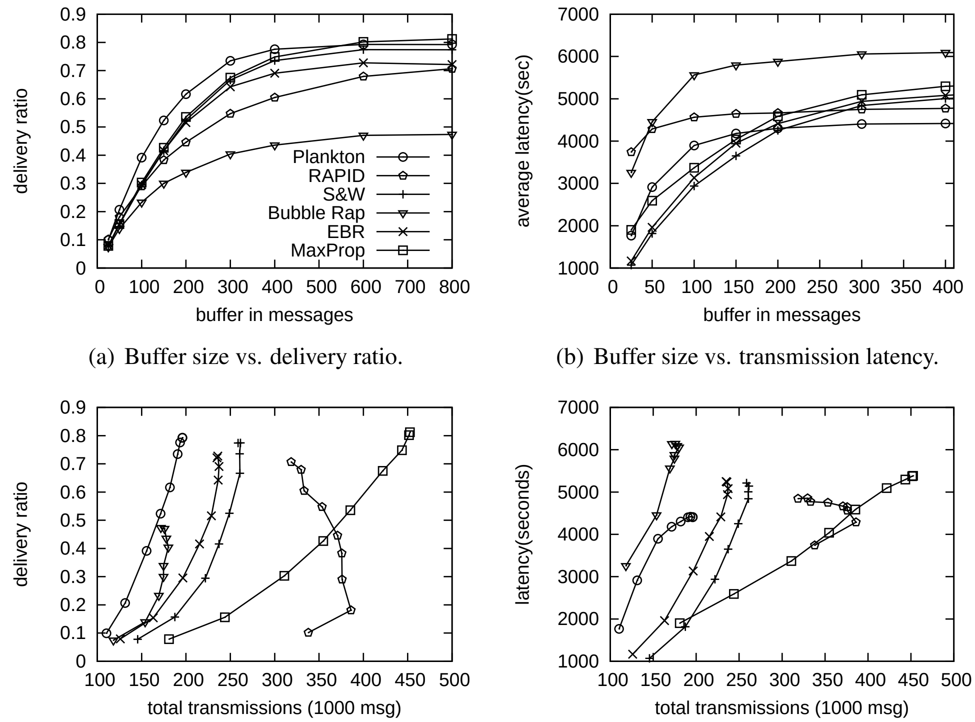
Figure 4.5: Performance when the bandwidth changes. The four figures share the same labels as indicated in Figure 4.5(a).

Average latency is the average latencies of delivered messages. We measure delivery overhead by dividing the total number of replicas of all messages by the number of delivered messages.

### 4.7.1 Evaluation on the San Francisco taxi trace

#### Performance of Varying Bandwidth and Buffer

Figure 4.5 shows the results when bandwidth is varied. Figures 4.5(a) and 4.5(b) show that Plankton achieves much higher DR (up to 80% higher) and lower latencies (up to 70% lower). Plankton achieves similar DR as MaxProp but with much lower latencies and less overhead (saving up to 60%). Figure 4.6 shows the results when buffer size is varied. Plankton achieves greater performance with similar or less overhead.




(a) Buffer size vs. delivery ratio.

(b) Buffer size vs. transmission latency.

(c) Transmissions vs. delivery ratio.

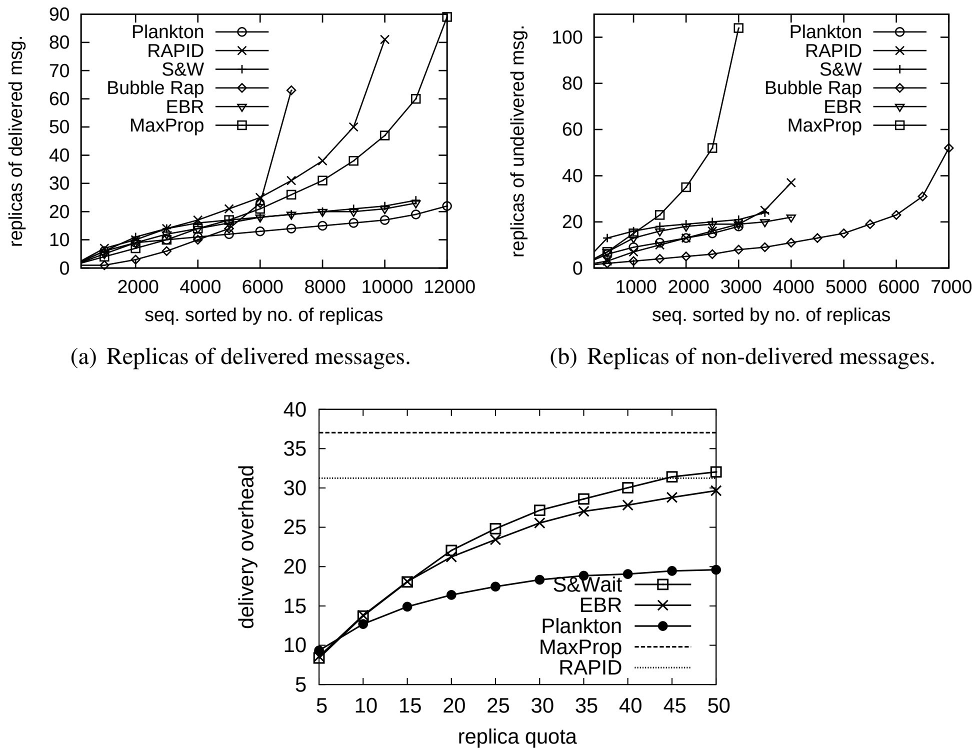
(d) Transmissions vs. latency.

Figure 4.6: Performance when the buffer sizes change. The four figures share the same labels as listed in 4.6(a).

### Overhead comparison

Figures 4.5(c) and 4.6(c) compare the delivery overhead when buffer or bandwidth is varied. For the same delivery ratio, Plankton incur the lowest delivery overhead. Plankton can save up to 60% replicas compared to performance-oriented algorithm like MaxProp and Rapid which do not limit replicas. On the other hand, Plankton can also achieve savings (up to 40%) unlike efficiency-oriented algorithms such as EBR and S&W that limits replicas.

We also examine the distributions of the number of replica made for both delivered and undelivered messages. Figure 4.7(a) shows that MaxProp and RAPID generate many more replicas than Plankton for delivered messages. These are overhead Plankton attempts to save. Similarly, Figure 4.7(b) shows that when the message cannot be delivered, many more replicas are generated by MaxProp. However, since these desti-




(a) Replicas of delivered messages.

(b) Replicas of non-delivered messages.

(c) Quota vs. delivery overhead.

Figure 4.7: Routing overhead with the bandwidth of 2.4MBps and the buffer size of 500 messages.

nations are unreachable, the limit on replication is needed to avoid generating excessive replicas.

Plankton, S&W, and EBR use quotas to control message replicas. Their efficiency varies with changes in the quota used. We examine their energy efficiencies when quota is varied. Results in Figure 4.7(c) shows that Plankton incurs significantly less overhead for all quota settings which are evaluated. We can see that Plankton's delivery overhead generates the smallest increase in overhead among the three algorithms when quota size increases. This demonstrates the effectiveness of the dynamical quota adjustment in Plankton.

The saving on replica can be explained by the existence of strong links that are encountered during message delivery. Figure 4.8 shows that the number of strong links encountered by a message over different time periods using the SF taxi trace. Over a

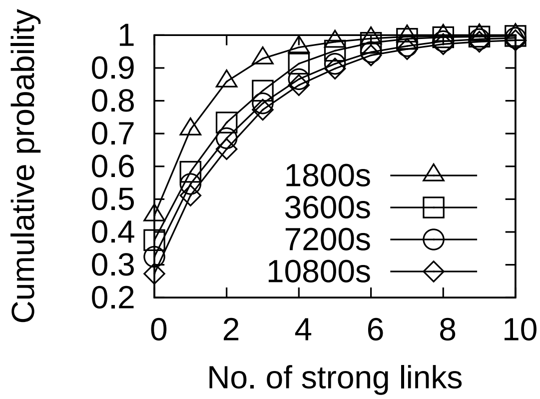



Figure 4.8: Number of encountered strong links.

1800s duration, 55% of the messages encounter at least one strong link. For a longer duration of 7200s, about 70% of the messages encounter at least one strong link. Encounters with strong links allows Plankton to significantly reduce the needed replicas without downgrading performance.

### 4.7.2 Evaluation on other traces

Next we present the results on seven traces in Table 5.3. $\rho$ is computed using Equation 4.1, and $\mathcal{Q}$ is computed via Equation 4.3 with $\mathcal{P}$ set to $0.9$. Bandwidth is set to 2.4 Mbps and buffer size is 500 messages. We compare Plankton to five other algorithms. For each mobility trace, we rank the algorithms in ascending order of delivery overhead, with the algorithm that incurs the least overhead listed first.

Plankton incurs the lowest communication overhead per delivered message in four of the seven traces, namely: SF taxi, Seattle Bus, RPGM, and RWP with saving from 14% to 88%. For the other three traces, Plankton occurs larger overhead than only Bubble Rap, but has significantly higher DRs and lower latencies.

In all mobility traces, Plankton has the highest or close to the highest delivery ratio among all algorithms evaluated. The lowest relative delivery ratio is for the Haggle trace (73% for Plankton and 76% for RAPID). The 3% loss in delivery ratio is compensated by 57% gains in communication overhead saving. The result is similar for latency. Plankton has the lowest latency in 4 out of 7 traces. It is able to perform better even with fewer replicas due to its more accurate contact probability prediction. We have two interesting observations. (1) MaxProp transmits more replicas and the performance is better than RAPID. One reason is that MaxProp generates much less metadata overhead than RAPID does. (2) Performance of Bubble Rap is not as good as other algorithms. Our observation suggests two reasons. (1) Bubble Rap renders too few replicas as it is very selective in choosing relays, which lowers delivery overhead but loses in the DR and the latency; (2) Bubble Rap selects relays by 'centrality' that reflects nodes' dissemination capability. Such a scheme can cause congestions in the central nodes.

## 4.8 Summary

The primary objective of this study was to investigate the accuracy of the connection prediction and exploit the prediction results for solving DTN routing problem. DTN connection prediction has aroused great interests because reliable connection prediction can elevate the DTN routing performance and energy efficiencies. One main goal of this study was to investigate how to leverage DTN connection properties to predict connection and thereby enhance both DTN routing energy efficiency and the performances, i.e., to deliver more messages for DTNs with less resource consumption. Another goal of this study was to further achieve a robust solution, i.e., the target solution can suit different types of traces.

The above two goals of this study are quite different from the existing ones. For the first goal on performances and resource efficiencies, most existing work either focused on saving resources or improving performances that were usually examined by delivery ratios and average latencies. This study targets both parts in one solution. For the second goal on robustness, previous work usually were evaluated in synthetic traces or few empirical traces, which is probably defective when being applied in diversified traces. This study however gives exhaustive evaluations on different types of traces.

This work addressed the gap by proposing and utilizing a novel connection prediction algorithm to control message replicas and evaluating the algorithm over popular traces of varied properties. As the first step, for measuring the accuracy of connection prediction, we proposed a connection prediction measurement algorithm. By this algorithm, we tested contact prediction reliance over a diversified range of real world traces. The testing results unfortunately indicated that most existing connection prediction algorithms fail to generate reliable and accurate results required for controlling the number of replicas. A reliable yet simple prediction algorithm based on bursty connections and associated connections thereby was proposed by this study. It was validated by real world trace based simulation that it can generate reliable prediction required by the replica control. Built on the connection prediction algorithm and the replica control scheme estimated from it, we constructed a resource efficient DTN routing algorithm. The algorithm was extensively evaluated on various traces with different settings of resource constraints, which secures its robust performances and efficiency in practices.

Besides the promising results as discussed above, the methodologies in this study have significant contributions to future researches. The most inspiring one is the utilization of contact prediction accuracy measurement. In most, if not all, previous solutions, various connection prediction algorithms were proposed and then mixedly utilized together with other techniques for DTN routings, and finally good outputs were achieved. This type of solution however fails to articulate how much its connection prediction algorithm contribute to the final results. It means that the final outputs are probably attributed to connection predictions or to other techniques that are simultaneously explored. This ambiguity can be addressed by measuring the connection predictions. Our study has provided a framework for the connection prediction assessment together with concrete measurement algorithm. New metrics or algorithms therefore can be raised and fit into the framework. As another significance, hybrid of replica control and connection predictions in this study can inspire future research on energy efficient DTN routings. In future, either the progresses in replica control or the progresses in connection predictions can simultaneously enhance DTN routing performances and DTN resource efficiencies.

This study can boost two types of future work as follows.

First, more theoretical work can be fulfilled. This work stressed on the robustness and practical feasibility through extensive evaluations. Theory rooted investigation however can be accomplished from theory perspective. Pure theoretical understanding is claimed to be notoriously hard as the heterogeneous properties of DTNs. We believe that sound evaluations of intuitive algorithm can provide clues for theoretical investigation.

Second, future research can delve more reliable and simple connection prediction algorithms. This work bestows one such algorithms and more algorithm of this type we believe can be rendered. Exploitation of such algorithm can directly enhance both DTN routing performances and DTN routing efficiencies.

| Trace/ $\rho$ /$\mathcal{Q}$ | Algorithm | DR | Delivery overhead | Latency (sec) |
|---|---|---|---|---|
| RollerNet/ 0.23/ 9 | *Bubble Rap* | 0.66 | 9.82 | 516 |
| | **Plankton** | **0.90** | **10.34** | **291** |
| | *EBR* | 0.83 | 12.33 | 432 |
| | *S&W* | 0.86 | 12.70 | 387 |
| | *MaxProp* | 0.88 | 13.93 | 322 |
| | *RAPID* | 0.90 | 15.19 | 314 |
| Haggle IC06/ 0.1/ 22 | *Bubble Rap* | 0.49 | 13.61 | 3890 |
| | **Plankton** | **0.73** | **17.25** | **1639** |
| | *S&W* | 0.73 | 22.58 | 1800 |
| | *EBR* | 0.74 | 17.85 | 1840 |
| | *MaxProp* | 0.76 | 37.53 | 1491 |
| | *RAPID* | 0.76 | 40.64 | 1571 |
| Reality/ 0.03/ 75 | *Bubble Rap* | 0.34 | 7.07 | 9052 |
| | **Plankton** | **0.56** | **14.01** | **7815** |
| | *S&W* | 0.52 | 16.11 | 8538 |
| | *EBR* | 0.52 | 18.57 | 8621 |
| | *MaxProp* | 0.55 | 23.95 | 8793 |
| | *RAPID* | 0.54 | 28.15 | 8569 |
| SF taxi/ 0.1/ 20 | **Plankton** | **0.79** | **16.00** | **3939** |
| | *S&W* | 0.76 | 21.19 | 4625 |
| | *EBR* | 0.71 | 24.77 | 5473 |
| | *Bubble Rap* | 0.45 | 24.80 | 6126 |
| | *RAPID* | 0.65 | 31.88 | 4451 |
| | *MaxProp* | 0.79 | 36.22 | 4818 |
| SeattleBus/ 0.04 / 55 | **Plankton** | **0.90** | **39.74** | **2955** |
| | *EBR* | 0.89 | 51.61 | 3043 |
| | *S&W* | 0.90 | 53.44 | 2919 |
| | *Bubble* | 0.76 | 100.16 | 3898 |
| | *MaxProp* | 0.94 | 331.15 | 2759 |
| | *RAPID* | 0.90 | 341.50 | 2611 |
| RPGM/ 0.22/ 9 | **Plankton** | **0.81** | **14.09** | **1709** |
| | *EBR* | 0.72 | 17.86 | 2413 |
| | *Bubble Rap* | 0.22 | 19.02 | 2383 |
| | *S&W* | 0.73 | 19.68 | 2218 |
| | *MaxProp* | 0.82 | 48.72 | 2122 |
| | *RAPID* | 0.85 | 53.92 | 2035 |
| RWP/ 0.24/ 8 | **Plankton** | **0.85** | **8.56** | **2155** |
| | *EBR* | 0.69 | 9.91 | 2845 |
| | *Bubble Rap* | 0.42 | 10.64 | 2944 |
| | *S&W* | 0.80 | 12.61 | 2788 |
| | *RAPID* | 0.82 | 14.18 | 2034 |
| | *MaxProp* | 0.80 | 19.24 | 2608 |

Table 4.5: Performance comparison on different traces.

# Chapter 5

# Resource efficient data aggregation in urban DTNs

In this chapter, we investigate a DTN application: how to exploit DTN communication to reduce infrastructure communication in global snapshot data collection. We investigate this problem in the context of ubiquitous sensing. The phones get ambient sense data and upload the data to the Internet to cater to more users. We first discuss a basic observation that some nodes know more information than other nodes in a DTN system. Based on this, we propose a concept *change awareness* and how to find the mobile nodes that know more information. After this, we discuss our algorithm that computes the smallest node set that can provide the global up-to-date information.

## 5.1 Background

Opportunistic networks[1] formed by mobile devices with short range wireless communication, e.g., Pocket Switch Network (PSN) [131] and Vehicle Ad hoc Networks (VANET) [132], have emerged and gained popularity in the past decade. The most distinctive feature of opportunistic networks is the intermittency of connections. Inter-

[1] In this chapter, we use 'opportunistic network' as a synonym name for DTN. We emphasize the connection opportunities to aggregate data by mobile nodes.

mittent connections enable the opportunistic communication, while they also complicate many problems that have much simpler solutions in static networks with persistent connections, e.g., the computation of the distance in connection topology between two nodes, the up-to-date node states in a DTN.

The behavior of opportunistic networks are often measured by information latency [133, 99] and reachability [100]. We propose a novel measure with a focus on the *freshness* of the information available. In particular, we measure information freshness by capturing how much the latest information received by a receiver differs from the most up-to-date information being propagated by the sender. Such differences are due to the changes caused by recent connections. This *information freshness* measure is independent of the transmission latency. It instead depends on how much global information may have changed from the local (the receiver's) view. We illustrate the usefulness of information freshness by the following two examples.

(1) In a dynamic network, many applications require nodes to collect connection topology data. However, it is often impossible to collect real-time topology information due to the intermittent connectivity. Knowledge on how much the topology has changed provides a way to view this information update and can be used to design efficient routing algorithms and data collection schemes.

(2) Mobile phones or vehicles can be used to sense an environment. An efficient data aggregation for opportunistic networks is challenging to design. In static wireless sensor networks, data aggregation algorithms have been extensively investigated. Most of them focus on selecting cluster headers or good information flow paths based on a static topology. However, we do not see general solutions for data aggregation in opportunistic networks.

We present the following main contributions in this chapter.

1. We propose a measure to quantify information freshness, called **change awareness**, in opportunistic networks and how it can be computed.

2. We propose the concept of change awareness coverage and show how a small

number of informative nodes can be used to construct a fresh (or real-time) snapshot of network states. Our trace based evaluation shows that depending on the application scenarios, the information collected from 0.3% to 50% of the nodes can form a fresh snapshot on topology states.

3. Finally, we evaluate the applications of change awareness in data aggregation. Extensive evaluation on real world contact trace shows that change awareness based solution with only the single last connection of each node can achieve similar performance as the solution with the connection oracle.

## 5.2 Literature review

### 5.2.1 Measurement of information path of opportunistic networks

As a natural model for dynamic networks, Time-Varying Graphs (TVG) [99] have attracted a plethora of research interest. In a TVG, edges, i.e., communication chances, appear and disappear and hence, a node's information on other nodes is usually delayed and incomplete. Research on TVGs has evolved from the study on the common feature with static graphs to the focus on temporal features. Initially, the research on TVGs or dynamic networks investigates the connectivity of TVGs [134], k-failure invulnerability problem [135], computation of computing shortest paths [133] and broadcast properties [136]. Message dissemination in edge-Markovia dynamic graphs are also investigated [137]. More recent work focuses on 'temporal features', e.g., information latencies and reachability in TVG. Casteigts et al. formalized temporal related features that are common for dynamic networks and proposed techniques for investigating the evolution of network properties [99]. The work in [100] formalizes reachability graphs and presents a theoretical framework for reachability graphs computation.

Along with these theoretical analysis, many research studies dynamic networks by investigating traces or modeling real world networks. They measure information latencies, information reachability and nodes influence in information propagation.

Most trace-based solutions and evaluations were done on OSN traces, as OSNs provide convenient avenues of collecting scalable traces. For example, twitter and Digg data are investigated in [138] to understand how network structures affect information dissemination in dynamic networks. Message reachability was also investigated on twitter data in [139].

Information latencies in social networks and the essential edges in the shortest paths were explored in [140]. They propose solutions to compute network backbones that appear to contribute most to shorten message latencies. Extended from [140], Casteigts checks message latencies in DTNs with arbitrary long contacts [141]. Temporal centrality and closeness are proposed in [142] to analyze information flows. Models for predicting influential users for OSNs were proposed in [143], which are evaluated by data from Digg.

These existing works investigate information transmission by examining information latencies. On the other hand, our work examines changes observable in the network and how these changes affect application data collection. In fact the concept introduced in this chapter is similar in spirit of the seminal work [144] in that we attempt to find a meaningful snapshot of the network state that can be useful for information aggregation.

### 5.2.2 Data aggregation of opportunistic networks

We briefly discuss the work on data aggregation relative to our evaluations. For a detailed survey, readers can refer to [91]. Our solution in data aggregation is close to the cluster based solutions [145] and the network-flow based solutions [146]. Existing solutions are often designed for static wireless networks and are unsuitable for opportunistic networks due to connectivity dynamics. Most solutions for opportunistic networks exploit the location information of vehicles and road side infrastructures [97]. Our approach does not rely on location information.

Cornejo et al. investigates the possibility of collecting data via short range connections to a subset of nodes to reduce the use of long range connections [101]. They proved theoretical bounds and show that no aggregation algorithm can achieve better

| Symbol | Interpretation |
|---|---|
| $\mathcal{G}(V, E, \mathcal{T}, \rho)$ or $\mathcal{G}$ | a Time-Varying Graph |
| $\varepsilon(u, v, t)$ | a connection between $u$ and $v$ ending at $t$. |
| $Time(\varepsilon(u, v, t))$ | the end time of connection :$\varepsilon$, i.e., $t$. |
| $I_p(u, v, t_s, t_e)$, $I_l(u, v, t_s, t_e)$, $I_f(u, v, t_s, t_e)$ | information path, the latest information path, fresh information path from $u$ to $v$ starting at $t_s$ and ending at $t_e$ |
| $\mathbb{I}_p(u, v, t)$, $\mathbb{I}_l(u, v, t)$, $\mathbb{I}_f(u, v, t)$ | the set of information paths (the latest information paths, fresh information paths) from $u$ and reachable to $v$ no later than $t$ |
| $F(\mathcal{G}, u, v, t)$ | pairwise change awareness: the change awareness measured by $v$ regarding to $u$ at $t$. |
| $F(\mathcal{G}, v, t)$ | vertex change awareness: the centrality of $v$ on receiving changes from all other vertices at $t$. |
| $F(\mathcal{G}, t)$ | graph change awareness: the average change awareness of vertices in $\mathcal{G}$ at $t$ |
| $\mathrm{c}_m$ | the smallest number of out-of-band connections required to achieve fresh snapshot |
| $\mathcal{P}$ | the set of smallest panorama informative nodes |

Table 5.1: Table for key symbols.

performance than the optimal solution with connection oracles. Our solution in this chapter shares the same goal of reducing the use of long range communications. We investigate how much can change awareness help minimize the number of uploaders, i.e., cluster headers.

### 5.2.3 Temporal graph models and information paths

#### Temporal graphs and connections

We use Time-Varying Graphs (TVGs) to model dynamic networks, as TVGs can naturally demonstrate dynamic properties. Our notations for TVGs are based on those used in [100, 99].

**Definition 1** (Temporal graph)**.** *Let $V$ be the set of vertices*[2]*, and $E \subseteq V \times V$ be the set of edges. $\mathcal{T}$ denotes* time span*, which starts at $\mathcal{T}_s$ and ends at $\mathcal{T}_e$, where $\mathcal{T}_s, \mathcal{T}_e \in R_+$*

[2] 'vertex' and 'node' are interchangeably used in this chapter.

*and $\mathcal{T}_s \leq \mathcal{T}_e$. $\rho(e,t)$: $E, R \rightarrow \{0,1\}$ is edge presence function, indicating whether an edge exists at a given time t or not. $\rho(e,t_-)$ indicates the presence of e before or at t; $\rho(e,t_+)$ indicates the presence of e after t. We call $\mathcal{G}(V,E,\mathcal{T},\rho)$ as a TVG or $\mathcal{G}$ for short.*

One occurrence of an edge corresponds to a connection in dynamic networks, which represents a communication opportunity. Two nodes can have non-negative number of connections during $\mathcal{T}$. The edge $\varepsilon(u,v,t)$ denotes a connection between $u$ and $v$ ending at $t$, and $\mathcal{E}_{\mathcal{G}}$ denotes the set of universal connections in $\mathcal{G}$. We use $\mathcal{E}_{\mathcal{G}}(v,t_-)$ and $\mathcal{E}_{\mathcal{G}}(v,t_+)$ to denote $v$'s connections before and after $t$ respectively. The brimming time point $t$ can be viewed as being included. The set $\mathcal{E}_{\mathcal{G}}(t)$ contains all connections in $\mathcal{G}$ at time $t$, i.e., the set of occurred edges (connections) in the snapshot of $\mathcal{G}$ at $t$.

Each node can be an information source. In a single connection, we assume that two nodes can exchange all requested information. When we need to refer to the time instance of a connection, we consider its connection end time.

### Information paths

An information path in TVGs consists of a list of connections that can relay information from one node to another.

**Definition 2** (Information path)**.** *$\forall u,v \in V$, $I_p(u,v,t_s,t_e)$ designates an information path from u to v. $I_p(u,v,t_s,t_e)$ has the format of $(u,v_1,t_1) \rightsquigarrow (v_1,v_2,t_2) \rightsquigarrow \cdots \rightsquigarrow (v_{k-1},v,t_k)$, where $(t_s = t_1 \leq t_2 \leq \cdots \leq t_k = t_e)$. Also, $v_0 = u, v_k = v$, $\rho(e_i,t_i) = 1$, where $e_i : \langle v_i, v_{i+1}\rangle$ and $i = 0,\cdots,k-1$. $I_p^i(u,v,t_s,t_e)$ indicates the $i^{th}$ connection in $I_p(u,v,t_s,t_e)$, i.e., $\varepsilon_i = \langle v_{i-1}, v_i, t_i\rangle$; $I_p^{last}(u,v,t_s,t_e)$ denotes the last connection in $I_p(u,v,t_s,t_e)$. Let $Time(I_p^i(u,v,t_s,t_e))$ present the time of the $i^{th}$ connection. $\mathbb{I}_p(u,v,t)$ symbolizes the set of information paths $I_p(u,v,t_s,t_e)$, where $t_e \leq t$.*

**Definition 3** (Latest information path)**.** *An information path $I_l(u,v,t_s,t_e)$ is the latest information path in $\mathbb{I}_p(u,v,t)$ iff $t_s$ is the latest (largest) among all information paths*

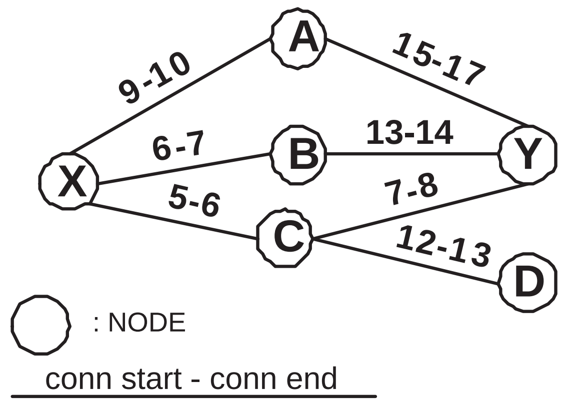



Figure 5.1: An example of temporal graph.

*in* $\mathbb{I}_p(u, v, t)$ *Hence,* $I_l(u, v, t_s, t_e)$ *carries the latest information from* $u$ *to* $v$ *available at time* $t$*.* $\mathbb{I}_l(u, v, t)$ *symbolizes the set of latest information paths from* $u$ *to* $v$ *no later than time* $t$*.*

**Definition 4** (Fresh information path)**.** *An information path* $I_p(u, v, t_s, t_e)$ *is a fresh information path at time* $t$ *or* $I_f(u, v, t)$ *iff* $\rho(\langle u, x \rangle, t_{s+}) = 0$ *for* $\forall x \in V - \{u\}$*. In other words,* $u$ *has no connection with any other nodes in the interval* $t_s$ *to* $t$*, and the information carried by this path is absolute* fresh*, as the node* $u$ *has no connection to update its information.* $\mathbb{I}_f(u, v, t)$ *denotes the set of all fresh information paths from* $u$ *to* $v$ *at time* $t$*.*

The idea of information path and these terms introduced can be illustrated by the dynamic network in Figure 5.1. At time 8, $Y$ receives update from $C$ about $X$'s communication with $C$. $Y$ does not yet know about the connection that $X$ has with $B$ (at time 7) through $C$ because it occurs after the disconnection with $C$ at time 7.

At time 14, $Y$ finally knows the connection between $X$ and $B$ but is unaware of the connection between $X$ and $A$. At time 17, it finally knows all connections occurred to $X$.

$\mathbb{I}_p(X, Y, 17)$ contains three information path: $X \to A \to Y$, $X \to B \to Y$ and $X \to C \to Y$. The shortest path from $X$ to $Y$ in terms of delivery latency is $X \to C \to Y$.

At the time 14, the latest information path set $\mathbb{I}_l(X, Y, 14)$ contains $X \rightarrow B \rightarrow Y$ or $I_p(X, Y, 7, 14)$. There is no fresh path at time 14 since X has another contact with $A$ after the contact with $B$. Next, at time 17, the latest and fresh information path are the same and is $X \rightarrow A \rightarrow Y$ or $I_p(X, Y, 10, 17)$.

The information paths defined have the following properties. First, $\mathbb{I}_f(u, v, t) \subseteq \mathbb{I}_l(u, v, t) \subseteq \mathbb{I}_p(u, v, t)$. Second, $\mathbb{I}_l(u, v, t) = \emptyset$ if and only if $\mathbb{I}_p(u, v, t) = \emptyset$. Third, based on local observation only, a node can know which path is the latest information path but it cannot judge whether a path is fresh or not.

## 5.3 Change awareness

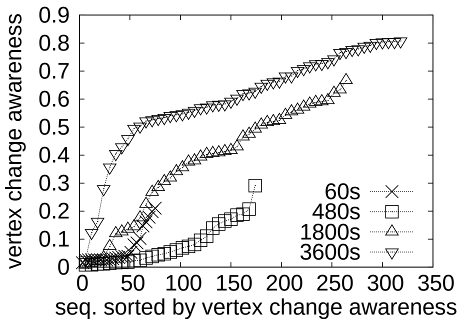
Based on the information paths introduced, we now present a concept on how *changes* in the network can be quantified. In particular, we are interested in the changes that occur as a result of communication between two nodes. Such changes can affect both the metadata (e.g. connection status or buffer content), as well as the application data.

Change awareness is measured by information paths involved. Three closely related metrics for information awareness are thereby proposed: the pairwise change awareness, the vertex change awareness, and the graph change awareness. These three measurements have peer metrics that have stirred a plethora of research interest: the geodesic vertices distance, the vertex centrality, and the mean geodesic distance.

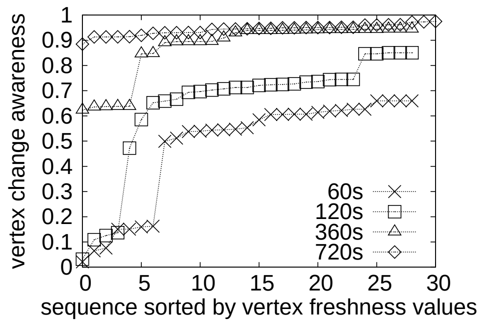
### 5.3.1 Definition on change awareness

**Definition 5** (Pairwise change awareness)**.** *The pairwise change awareness $F(\mathcal{G}, u, v, t)$ is the ratio of $u$'s connections that can be known by $v$ at $t$. $F(\mathcal{G}, u, v, t) = |\mathcal{E}_\mathcal{G}(u, \tau_-)|/|\mathcal{E}_\mathcal{G}(u, t_-)|$, where $\tau$ is the latest connection time with $u$ in $\mathbb{I}_l(u, v, t)$. $\tau$ is set to $\mathcal{T}_s$ if $\mathbb{I}_l(u, v, t)$ is $\emptyset$.*

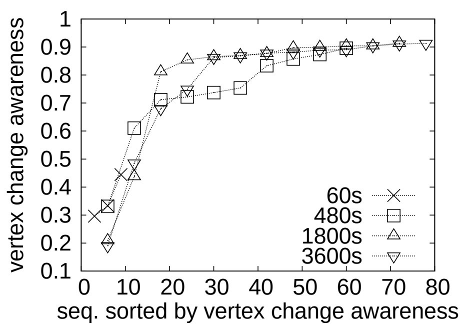
$F(\mathcal{G}, u, v, t)$ is computed based on the set of latest information paths and measures the portion of $u$'s connections no later than $\tau$ and $u$'s connections no later than $t$. The largest freshness value, one, occurs when a fresh information path exists. The smallest freshness value, zero, occurs when no information path exists from $u$ to $v$. As an ex-




(a) SF taxi trace (50m).




(b) Roller Net trace.




(c) Haggle InfoCom06 trace.

Figure 5.2: Vertex change awareness with different time span lengths.

ample, the pairwise change awareness of the network at 17s in Figure 5.1 is presented in the following table, where a row stands for a source ($u$) and a column indicates a destination ($v$).

| | $A$ | $B$ | $C$ | $D$ |
|---|---|---|---|---|
| $A$ | $2/2$ | $0/2$ | $0/2$ | $0/2$ |
| $B$ | $2/2$ | $2/2$ | $0/2$ | $0/2$ |
| $C$ | $2/3$ | $2/3$ | $3/3$ | $3/3$ |
| $D$ | $0/1$ | $0/1$ | $1/1$ | $1/1$ |

**Definition 6** (Vertex change awareness)**.** *The vertex change awareness of a node $v$: $F(\mathcal{G}, v, t)$ measures how much $v$ can be aware of the changes of other nodes. It can be computed as $\sum_{x \in V, x \neq v} F(\mathcal{G}, x, v, t)/(|V| - 1)$.*

**Definition 7** (Graph change awareness)**.** *The graph change awareness of $\mathcal{G}$: $F(\mathcal{G}, t)$ designates the average awareness over all vertices in $\mathcal{G}$. It can be computed through the pairwise change awareness: $\sum_{x,y \in V} F(\mathcal{G}, x, y, t)/P(|V|, 2)$, or the vertex change*

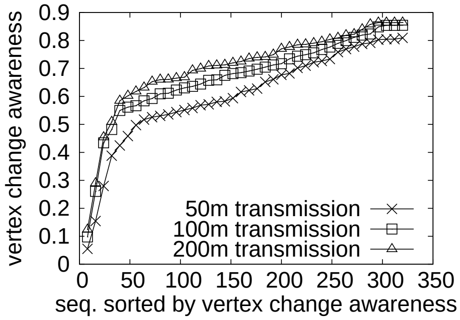
**Data**: $\mathcal{G}(V, E, \mathcal{T}, \rho, \xi)$
**Output**: $F(\mathcal{G}, v, t)$
init.: $last(x) \leftarrow \mathcal{T}_s, \forall x \in V, known(\varepsilon) \leftarrow false, \forall \varepsilon \in \mathcal{E}, W \leftarrow \{v\}$,
$last(v) \leftarrow t$
**while** *($W$ is not empty)* **do**
  select $u$ having the maximum $last(\cdot)$
  **for** *each* $\varepsilon : \varepsilon(u, y, \tau)$ *or* $\varepsilon : \varepsilon(y, u, \tau)$ **do**
    **if** $known(\varepsilon) = false$ **then**
      **if** $\tau \leq last(u)$ **then**
        $last(y) \leftarrow \max(last(y), \tau)$
        **if** $y \notin W$ **then**
          add $y$ to $W$
        **end**
        $known(\varepsilon) \leftarrow true$
      **end**
    **end**
  **end**
  remove $u$ from $W$
**end**
**for** $x \in V$ **do**
  $F(\mathcal{G}, x, v, t) \leftarrow \frac{\text{No. of known conn. involving } x}{\text{No. of conn. involving } x}$
**end**
$F(\mathcal{G}, v, t) \leftarrow \sum_{x \in V, x \neq v} F(\mathcal{G}, x, v, t)/(|V| - 1)$

**Algorithm 5.1**: Vertex change awareness computation.

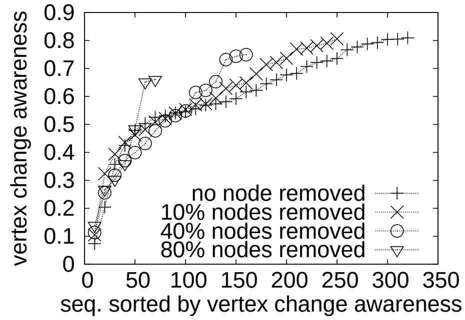
*awareness:* $\sum_{v \in V} F(\mathcal{G}, v, t)/(|V|)$.

### 5.3.2 Computation of change awareness

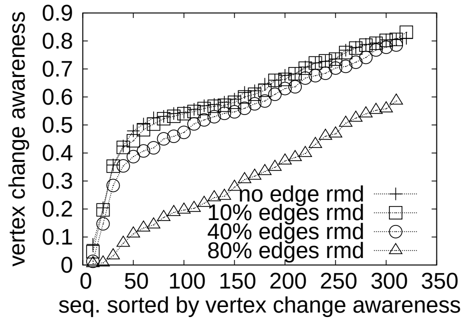
Pairwise change awareness can be computed through 'last departure time': $Time(I_l^1(u, v, t))$ [128]. The vertex and graph change awareness can be computed via repetitively running 'last departure time' algorithm for each node, but this can be expensive. They however can be computed by a more efficient algorithm that adapts the dijkstra algorithm for computing the shortest paths from one vertex to all other vertices. This is shown in Algorithm 5.1 with the time complexity of $\Theta(|\mathcal{E}_{\mathcal{G}}| + |V| \log |V|)$ [147].




(a) Transmission range change.




(b) Nodes removal.




(c) Probability gossip

Figure 5.3: Vertex change awareness in different situations.

### 5.3.3 Change awareness of traces

We investigate the three types of change awareness on traces in Table 4.1. In this chapter, we only present the results of vertex change awareness on representative traces SF taxi, Haggle InfoCom06, whose contexts respectively represent vehicle traffic, outdoor human mobility, and indoor human mobility. We investigate the change awareness with respect to four factors: (1) the length of time span, (2) the connection density (by varying transmission range), (3) the node density (by removing random nodes), and (4) the edge density (by removing random edges to simulate duty cycling) [56].

For measuring the effects of the length of time spans ($\mathcal{D}$), we vary the time span length from 60s to 3600s. Results in Figure 5.2 reveal two points. First, some nodes are much more aware of changes in the network than others. For example, in Figure 5.2(a), in each case, some nodes have much higher vertex awareness while others have much smaller vertex change awareness. Second, the pairwise change awareness usually increases as the time span becomes longer. For example, in the SF taxi trace, vertex change awareness varies from 0 to 0.8 when the time span is 3600s. For a short time span of 60s, the awareness varies from 0 to 0.38.

For testing the effects of contacts density, we set the transmission ranges as 50m, 100m and 200m, which respectively have the density of 16.8, 28.1, 48.7 contacts per hour per node for SF taxi trace. Figure 5.3(a) shows the results for the time span of 3600s. The increase in node density has two opposite effects. One effect is to make information transmitted by $v$ become stale faster, which decreases change awareness. Another effect is to speed up message transmission, which increases change awareness. The results indicate that the positive effects slightly outperform the negative effects as the transmission range increases. In an extreme case where the contact density is so large that end-to-end paths always exist, all change awareness values approach one.

We test the effects on change awareness by removing random connections. This has similar effects of letting nodes duty cycle their wireless interfaces. The results in Fig 5.3(c) show that the vertex change awareness decreases slowly with edge removals. With up to 60%, the decrease is insignificant. Striking decreases in change awareness are observed when 80% of nodes are removed. The results indicate that nodes can have duty cycling to save energy without significant loss in change awareness.

We also test the effects on change awareness by removing nodes from a network. The results in Figure 5.3(b) show no significant effects on vertex change awareness till more than 40% of the nodes are removed. The reason could be that while the removal of nodes reduces the chance of updates, it slows the information dissemination. In the extreme case where there are only two nodes, change awareness is always one.

We also investigate vertex freshness change with time, as shown in Figure 5.4. At beginning, the node known nothing about connections of the network. As the node gets more connections, it knows more about the connections. At the same time, the connections of the whole network gradually grows. When the node has no connection, its vertex change awareness decreases gradually; when the node get a connection, it can get information from the connected node, its vertex change awareness jumps up. How

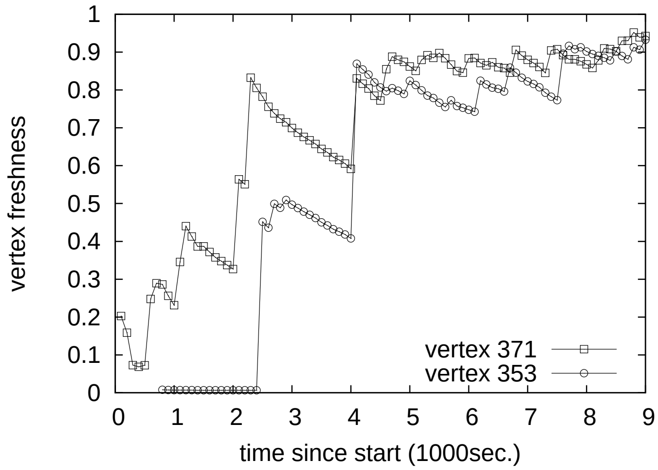



Figure 5.4: Vertex freshness change with time (SF taxi trace). Vertex change awareness is computed every 50 seconds since the first connection of the selected nodes.

much can its vertex change awareness jumps depends on how much new information it can get from the connected node.

## 5.4 Informative node

Next, we will investigate how a node's change awareness can be exploited. In particular, we focus on nodes that have the most information about changes in an opportunistic network. We call such nodes *informative nodes*. An informative node is the last node in a fresh information path, i.e., $v$ in $I_f(u, v, t_s, t_e)$, or $v$ in $F(\mathcal{G}, u, v, t)$ when $F(\mathcal{G}, u, v, t)$ equals one.

We will show that a set of informative nodes can be combined to provide a *fresh* or 'real-time' snapshot of a dynamic network and we aim to find the minimum number of informative nodes required to obtain a *fresh snapshot*.

### 5.4.1 Node fresh coverage

In an ideal case, a node $v$ has fresh information paths from all vertices, which guarantees that $v$ can access global fresh (up-to-date) information. This ideal case, however, only occurs in some special topologies, e.g., complete bipartite graph. To measure how much absolute fresh information a node can collect, we introduce the concept of change awareness coverage.

For two vertices, $u$ and $v$, we say $u$ is covered by $v$ if $u$ has a fresh information path to $v$. The coverage of $v$ is defined as the set of nodes that have at least one fresh information path to $v$. The node $v$ is thus an informative node. An informative node can cover multiple vertices and a vertex can be covered by multiple informative nodes. A set of informative nodes is called the *panorama informative node set* if the union of their coverage equals the set of whole nodes. The set of all vertices obviously form a panorama informative node set, as each node can cover itself. We are interested in computing the smallest panorama informative node set. The smallest panorama informative set has potential applications in opportunistic networks, such as the one we will discuss in Section 5.5.

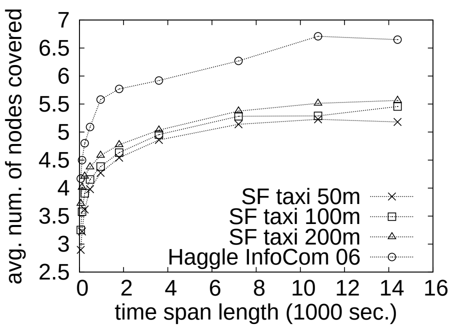
```
Input: 𝒢, where connections in ℰ_𝒢 are sorted by connection time.
Output: ⟨v, C(v)⟩, v ∈ V.
initialize coverage association sets, M ← {⟨v, C(v)⟩}, v ∈ V, C(v) ← {v}
while ℰ_𝒢 is not empty do
    remove the earliest connection ε : ⟨u, v, t⟩ in ℰ
    if only one node (e.g., u) in ε is isolated ³ then
        add C(u) to C(v)
        remove ⟨u, C(u)⟩ from M
    end
    if both v and u are isolated nodes then
        randomly select a node (e.g., u)
        add C(u) to C(v)
        remove ⟨u, C(u)⟩ from M
    end
end
return M
```

**Algorithm 5.2**: The computation for smallest panorama informative node set.

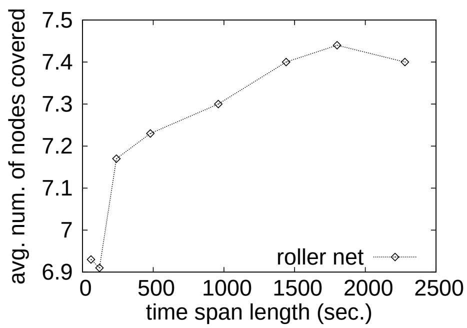
The smallest panorama informative node set can be computed by Algorithm 4 with the time complexity of $O(|\mathcal{E}|)$. For example, Algorithm 4 outputs the coverage association $\{\langle C(or\ D), \{C, D\}\rangle, \langle A(or\ Y), \{A, B, X, Y\}\rangle\}$ for Figure 5.1. We write such an association in the format of $\langle v, C(v)\rangle$, where $v$ is an informative node and $C(v)$ is the coverage set of $v$. We name the set of informative nodes computed by Algorithm 4 as

[3] A node is isolated if it has no connection to any other node.

$\mathcal{P}$. Next, we would like to state the following lemmas and theorem. The proofs can be found in Appendix B.




(a) SF taxi and Haggle InfoCom06 traces.




(b) Roller net trace.

Figure 5.5: The average number of nodes covered by one informative node in different traces. The inverse of this average provides an rough estimate of the number of uploaders required to gain a global snapshot.

**Lemma 1.** *The coverage sets of nodes in $\mathcal{P}$ forms a partition of $V$. That is, for any two informative nodes $p_i, p_j \in \mathcal{P}$, $C(p_i) \bigcap C(p_j) = \emptyset$ if $p_i \neq p_j$, and $\bigcup_{p \in \mathcal{P}} C(p) = V$.*

**Lemma 2.** *$F(\mathcal{G}, u, p, t)$ equals to 1, for $\forall u \in C(p)$, $p \in \mathcal{P}$, i.e., a fresh information path exist from $u$ to $p$ at $t$.*

**Lemma 3.** *Algorithm 4 generates the smallest panorama informative node set.*

Based on these three lemmas, the following theorem holds.

**Theorem 4.** *A fresh snapshot of a dynamic network at time $t$ can be obtained by assembling information from all informative nodes in $\mathcal{P}$, and the size of $\mathcal{P}$ is minimum.*

The theorem can be reached by the above three lemmas. Note that while $\mathcal{P}$ is not unique, but the size of $\mathcal{P}$ is the same.

### 5.4.2 Coverage of informative nodes in traces

We study how $\mathcal{P}$ varies with the changes of time spans and transmission ranges. We divide one day of the SF taxi trace into segments of different lengths ($t_s, t_s + D$) with $t_s = 0, 1, \cdots, 86400 - D$. Instead of measuring the size of $\mathcal{P}$ directly, we measure the average number of nodes that can be covered by an informative node. The result is shown in Figure 5.5.

The result in Figure 5.5(a) shows that the average coverage increases slowly when the transmission range grows from 50m to 200m. For a fixed transmission range, as time span increases, average coverage increases quickly but stabilizes beyond 3600s.

An important indication by the results is that it provides a lower bounder for the number of required nodes to get a snapshot of the network. This has an interesting application in data aggregation, as discussed in the following section.

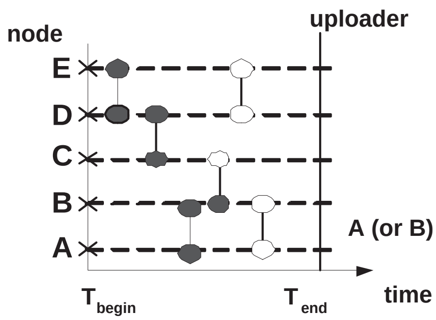
## 5.5 Change awareness applications in data aggregation

In this section, we apply change awareness to the data aggregation problem. In our evaluation, each node in opportunistic networks has short range wireless communication, e.g., Wi-Fi, and long range wireless communication, e.g., 3G and LTE. A server regularly collects sensor readings from all nodes. The objective is to minimize the use of long range communication connections, i.e., uploaders, for collecting application data. The reduction comes in two forms. We would like to reduce both the total amount of data uploaded, as well as the number of uploading nodes.

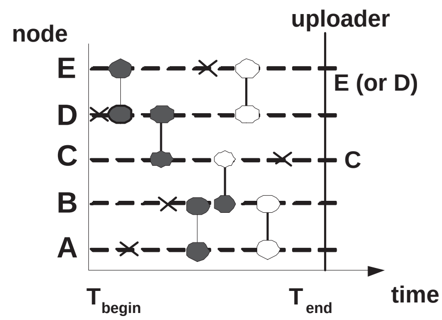
### 5.5.1 Types of sensor data in opportunistic networks

We discuss data aggregation for the following three types of sensor data in opportunistic networks, as shown in Figure 5.6. Most, if not all, sensor data generation belong to one of the three categories.

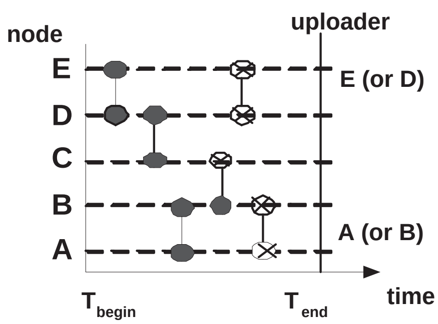
1 **Synchronous Sensor Reading (SSR).** It is shown in Figure 5.6(a). In SSR, sensors values on all nodes are read at a synchronized time. One example application is to build pollution or temperature maps for a city. Suppose that all taxis in a city have synchronized time and Global Positioning System (GPS). They hourly sense the air temperatures or pollution. The server can collect all the sensor readings together with GPS and form a temperature/pollution map. In SSR, if the data is




(a) Synchronous Sensor Reading (SSR).




(b) Asynchronous Sensor Reading (ASR).




(c) Conn.-triggered Sensor Reading (CSR).

Figure 5.6: The illustration of different types of ubiquitous environmental sensing applications. A line between two nodes indicates a connection. An empty circle is the last connection of a relative node. A solid circle indicates a connection that is not the last one of the relative node. A crossing symbolizes the point when the sensor data generate.

required to be collected at the sensing time, then no data aggregation algorithm based on short range communication can help reduce the number of uploaders. Thus we consider the case when delays are allowed. For example, the temperature/pollution maps for 10:00AM are published at 11:00AM.

2 **Asynchronous Sensor Reading (ASR).** It is shown in Figure 5.6(b). In this type of applications, each sensor logs its sensor readings once it gets a new reading. The sensor readings are done in an asynchronous manner. One example application is the monitor of noise pollution levels. The pollution level changes every 20dB. The pollution is at level zero if the noise decibel is within 0 - 20dB; it is level one if the noise decibel falls into 20 - 40dB, and so on. A sensor in a taxi records the sensed noise pollution level only when the new level is different from

the previous one. Another example is the collection of taxi occupation states, e.g., whether a taxi has passengers on board. This type of information are updated in an asynchronous manner. The sensed data is aggregated and uploaded to the server at the end of an aggregation period. The server receives real-time data.

3 **Connection-triggered Sensor Reading (CSR).** It is shown in Figure 5.6(c). In this type of applications, the data changes only when the hosting node has a connection with other nodes. For example in a system where news or advertisements are disseminated over a VANET, the business operators are interested to know the distribution information of advertisement copies in order to charge customers and to steer advertisement dissemination. In this case, the advertisements distribution information changes only when a node is connected with another node.

Figure 5.6 illustrates three types of applications. The server periodically collects data on all nodes through a subset of nodes called the uploaders. These uploaders can aggregate data via short range connections. For example, in Figure 5.6(a), the node E can transfer its data to D. Next, E and D's data can be collected by C via the connections between C and D, and so on. Finally, either only node A or B needs to upload data. However, in Figure 5.6(b), D's data can reach C, but E's data cannot reach C. In this case, either E or D has to upload data. In Figure 5.6(c), due to the connection patterns, two nodes E (or D) and A (or B) have to upload. The lower bound for the number of uploaders can be estimated in the follow way.

Assuming that the server has the access to all connection states, it can compute the minimum number of uploaders needed by formulating the problem as a minimum Set Covering Problem (SCP), which is NP hard. A greed algorithm with proved approximation ratio is used in our benchmark algorithm (ORCALE Greedy) for comparison.

We list the minimum number of uploaders of the three types of applications in Table 5.2. Note that only CSR tightly fits the definition of the change awareness. However, we also show how change awareness based solutions help the data aggregation for SSR, ASR.

| App | Uploader no. | Note |
|---|---|---|
| SSR | $[0, \|\mathcal{P}\|]$ | dependent on the reachability of the data |
| ASR | $[0, \|\mathcal{N}\|]$ | the node whose data are generated after its last connection need to upload the data itself. |
| CSR | $= \|\mathcal{P}\|$ | as shown by Theorem 4. |

Table 5.2: The lower bound of the number of uploaders. ( $\mathcal{N}$ is the total number of nodes)

### 5.5.2 The change awareness based data aggregation algorithm

In change awareness based solution, for each data aggregation period, the server needs to collect each nodes' last connection to compute informative nodes using Algorithm 4. Because the last connection information is of constant and small data size (several bytes), it can be transmitted via the control or low bit rate channel of the cellular network, e.g., Short Message Service (SMS), which is power efficient. Once the informative nodes are identified, they upload collected sensor data via the cellular data channel.

We tried two different change awareness based algorithms: CA-Base and CA-Plus. In CA-Base, each node uploads its last connection information to the server, and then the server can compute a set of coverage associations $\langle v, C(v) \rangle$ by Algorithm 4. The computed informative nodes upload the sensor readings of nodes under its coverage. By this method, the server can collect all data that are generated before the hosting nodes' last connection. In ASR, the nodes, whose sensor readings are generated after their last connection, need to upload their own sensor reading by themselves. Actually, no algorithm can help aggregate these data as there is no short range communication after the data is generated.

The second approach (CA-Plus) enhances CA-Base by controlling different uploaders. Each node also needs to upload its last connection to the server (data size is only a few bytes). When a node generates data after its last connection (as in ASR), the data is only known to itself. Such nodes upload their own sensor data and the collected data to the server. Then, the server computes the informative nodes and their coverage, and greedily selects the next uploader from informative nodes that has the most uncollected sensor data. This step is repeated until the server has collected data from all nodes.

### 5.5.3 Data aggregation algorithms for comparison

#### The benchmark algorithm (ORACLE-Greedy)

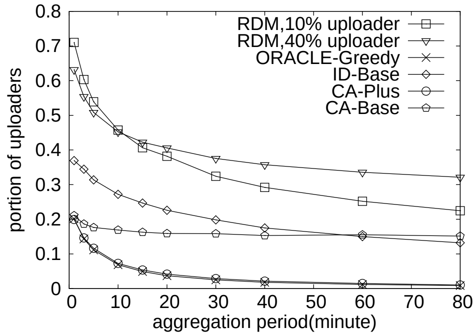
In the benchmark algorithm, a server collects all short range connection information and thus can compute the complete information reachability. ORACLE-Greedy algorithm uses the greedy algorithm [148] that selects the next uploader as the one having the most information to upload, e.g., the node having the most information that are unknown to the server. ORACLE-Greedy can compute the minimum number of uploaders by constrained approximation ratios, but it causes high overhead as all nodes need to upload all its connection information.

#### Random uploader algorithm and ID based algorithm

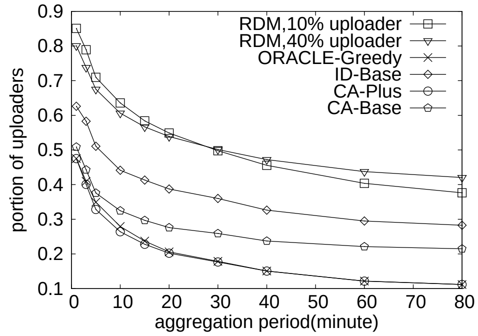
Besides the benchmark algorithm, we also compare our solution with two other algorithms: RDM and ID-Base. (1) RDM is adapted from LEACH [145]. RDM randomly selects $x\%$ of nodes as fixed uploaders. A non-uploader forwards its data to an encountered uploader. At the aggregation time point, both the fixed uploaders and non-uploaders that have never encountered any uploader upload data to the server. We evaluate RDM by varying the ratio of uploaders. (2) ID-Base algorithm chooses uploaders by nodes' IDs. When two nodes encounter, the node with the smaller identifier transfers its sensor data to the node with the larger identifier. At the aggregation time, all nodes with sensor data upload the collected data.

### 5.5.4 Evaluation settings and results

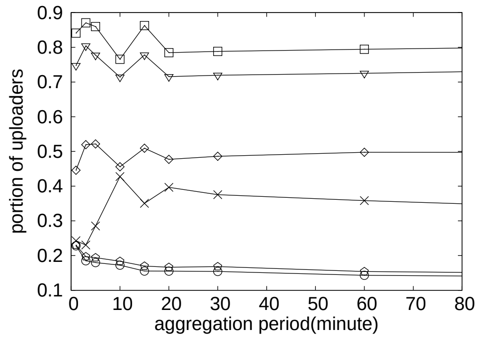
We present evaluation results on two vehicle traces, Shanghai taxi trace and SF taxi trace, where the Shanghai Taxi trace has 10 times more taxies than the SF taxi trace. The results are shown in Figure 5.7 and 5.8. The ratio of uploaders ranges from 0.003




(a) Synchronous Sensor Reading.




(b) Asynchronous Sensor Reading.



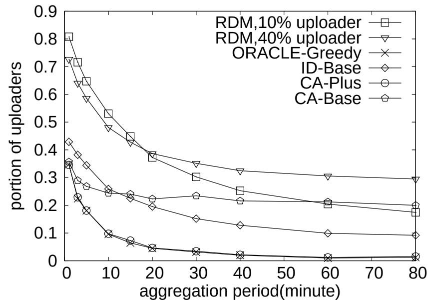

(c) Connection-triggered Sensor Reading.

Figure 5.7: Performance of different algorithms with varying aggregation period (ShangHai taxi trace).

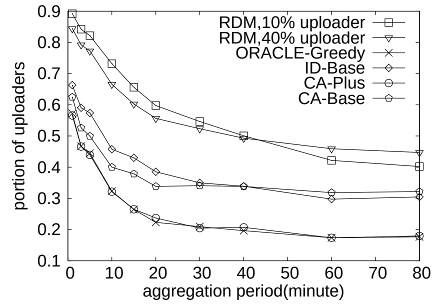
(for SSR, ORACLE-Greedy and CA-Plus) to 0.9 (for CSR, RDM). For CSR, CA-Base can achieve similar performance as CA-Plus and ORACLE-Greedy. CA-Plus and ORACLE-Greedy achieve similar results for SSR and ASR. In most settings, the three algorithms require significantly fewer uploaders than RDM and ID-Base algorithms.

These results further demonstrate the following points. First, for CSR, CA-Base provides a lower bound for the number of uploaders required for a fresh snapshot, as shown by Theorem 4. Second, for many cases in SSR and ASR, not all informative nodes need to upload data. The number of nodes required to upload all data depends on the data reachability constrained by intermittent connections. This can be computed based on connections, as is done in ORACLE-Greedy.

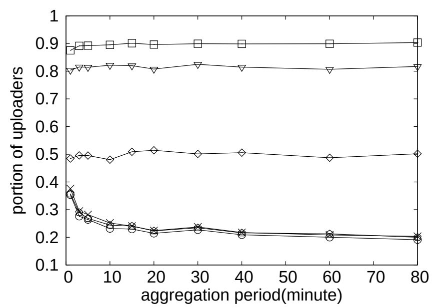
The results on more traces are listed in Table 5.3. These results show that CA-Plus can gain similar performance as ORACLE-Greedy, which is expected to perform well since it has the access to all connection information. However, CA-Plus can save much




(a) Synchronous Sensor Reading.




(b) Asynchronous Sensor Reading.

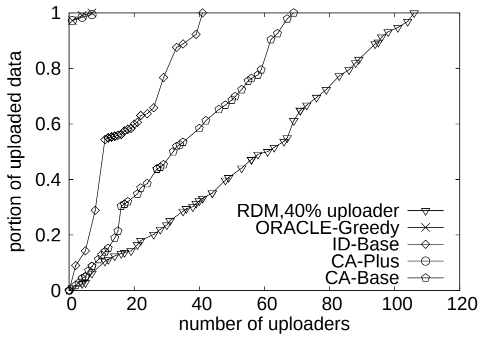



(c) Connection-triggered Sensor Reading.

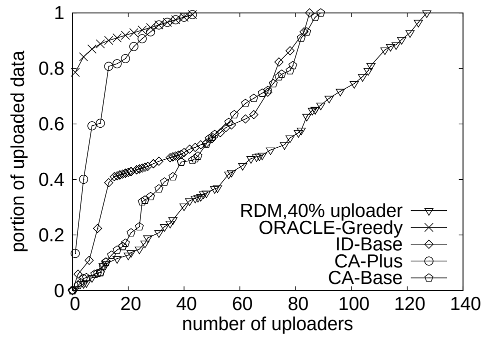
Figure 5.8: Performance of different algorithms with varying aggregation period (San Francisco taxi trace).

communication overhead ranging from six times to 15 times.

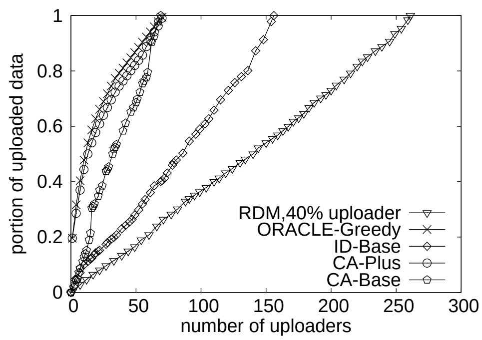
We also investigate intermediate states of the data collection process, i.e., how much information is collected versus how many uploaders are utilized. This is meaningful to applications where only a portion of data needs to be collected. The result in Figure 5.9 shows that for a fixed number of uploaders, CA-Plus and ORACLE-Greedy can collect much more data than other algorithms. The CA-Plus and ORACLE-Greedy collect similar size of data for SSR and CSR. For ASR, the ORACLE-Greedy outperforms CA-Plus for the first 22 uploaders. For example, with the first 10 uploaders, the ORACLE-Greedy can collect 85% of data, while CA-Plus only can collect 60% of data. When the number of uploaders are larger than 22, the two algorithms perform similarly. Given fewer number of uploaders, CA-Plus performs not as good as ORACLE-Greedy. The reason is that without complete connection information, CA-Plus is unable to always select the best uploader under the asynchronous scenario.




(a) Synchronous Sensor Reading.




(b) Asynchronous Sensor Reading.




(c) Connection-triggered sensor Reading.

Figure 5.9: Dynamics of data aggregation on SF taxi trace.

## 5.6 Summary

As pointed by the paper [37, 36], DTN research community is still seeking applications. As smart phones have more and more sensors, they can cooperate to sense environment and communicate sensor data. They require scalable, energy and finance cost efficient communication solutions. DTNs can play an important role. In this chapter, we discuss the solution to exploit DTN local communication scheme to minimize infrastructure communication in sensor data collection. The concept change awareness is proposed to compute the 'informative nodes' that can know more information of other nodes in the system. We discuss an algorithm to compute the minimum nodes to upload data to the server via infrastructure communication. The evaluation on human traces and vehicle traces shows only about 1/5 of total nodes are needed to upload data.

| Trace | Algorithms | SSR | ASR | CSR | Overhead |
|---|---|---|---|---|---|
| SF taxi (50m) | RDM | 0.531 | 0.677 | 0.849 | 0 |
| | CA-Base | 0.222 | 0.360 | 0.224 | 269 |
| | ID-Base | 0.256 | 0.430 | 0.492 | 0 |
| | CA-Plus | 0.080 | 0.272 | 0.211 | 269 |
| | ORACLE-Greedy | 0.075 | 0.267 | 0.229 | 5156 |
| ShangHai Taxi | RDM | 0.484 | 0.615 | 0.823 | 0 |
| | ID-Base | 0.268 | 0.428 | 0.538 | 0 |
| | CA-Base | 0.160 | 0.282 | 0.160 | 4037 |
| | CA-Plus | 0.053 | 0.216 | 0.147 | 4037 |
| | ORACLE-Greedy | 0.048 | 0.224 | 0.230 | 266550 |
| Haggle IC06 | RDM | 0.659 | 0.788 | 0.867 | 0 |
| | ID-Base | 0.277 | 0.558 | 0.529 | 0 |
| | CA-Base | 0.138 | 0.476 | 0.138 | 60 |
| | CA-Plus | 0.109 | 0.422 | 0.122 | 60 |
| | ORACLE-Greedy | 0.116 | 0.430 | 0.190 | 650 |
| Seattle Bus | RDM | 0.682 | 0.797 | 0.807 | 0 |
| | ID-Base | 0.269 | 0.559 | 0.506 | 0 |
| | CA-Base | 0.241 | 0.454 | 0.241 | 921 |
| | CA-Plus | 0.139 | 0.383 | 0.234 | 921 |
| | ORACLE-Greedy | 0.133 | 0.389 | 0.280 | 5870 |
| Roller Net | RDM | 0.356 | 0.437 | 0.813 | 0 |
| | ID-Base | 0.187 | 0.200 | 0.471 | 0 |
| | CA-Base | 0.179 | 0.212 | 0.179 | 61 |
| | CA-Plus | 0.027 | 0.080 | 0.172 | 61 |
| | ORACLE-Greedy | 0.023 | 0.093 | 0.249 | 5870 |
| RWP | RDM | 0.515 | 0.684 | 0.817 | 0 |
| | ID-Base | 0.23 | 0.442 | 0.478 | 0 |
| | CA-Base | 0.248 | 0.376 | 0.248 | 100 |
| | CA-Plus | 0.03 | 0.256 | 0.246 | 100 |
| | ORACLE-Greedy | 0.022 | 0.268 | 0.26 | 1560 |
| RPGM | RDM | 0.471 | 0.624 | 0.827 | 0 |
| | ID-Base | 0.216 | 0.365 | 0.506 | 0 |
| | CA-Base | 0.220 | 0.316 | 0.220 | 100 |
| | CA-Plus | 0.094 | 0.228 | 0.202 | 100 |
| | ORACLE-Greedy | 0.085 | 0.232 | 0.257 | 2860 |

Table 5.3: The uploader ratio ((no. of the uploaders)/(no. of the total nodes)). We set the aggregation period as 1800s. For RDM, we set 40% of the nodes as prefixed uploaders. The overhead is computed as the number of contacts required to be uploaded for computing uploaders.

# Chapter 6

# Conclusion and future work

## 6.1 Conclusion

The work in this thesis is done under the background that mobile communicative devices sharply increase. The number of smart phones grows strongly, and the capability in sensors, computation, and short range wireless communication is frequently escalated. More and more vehicles carry wireless transmission chips and smart sensors. Under this trend, the research on the intermittent communication between mobile nodes, i.e., DTN, provides a strong base for potential applications on mobile devices.

The thesis targets resource efficient DTN solutions. The scope covers three parts: (i) neighbor discovery for nearby communicable mobile devices, (2) message routing from a source mobile node to a destination mobile node, and (3) the data aggregation via DTN communications for mobile nodes.

The first part on neighbor discovery investigates how to exploit WiFi AP to synchronize mobile nodes to speed up neighbor discovery with resource conservation. We take a deep insight into existing solutions and analyze the proper protocols for different requirements and hardware settings. Our solutions feed the greedy power saving requirement, e.g., less than 1% power consumption and capturing more than 90% of neighbors with 50s or more connection period. R2 can perform well in neighborhood with dense phones. In the urban settings, it becomes more often that a large number of mobile devices cluster in some locations, e.g., train stations, lecture halls, shopping malls. To serve this requirement, our solution $R2$ is scalable in term of wireless collision control.

The second part on DTN routing studies the messages routing from a source mobile node to a destination mobile node. This work has two main contributions. First we propose and evaluate a reliable and accurate contact predictor. The contact predictor is different from other prediction algorithms in that it integrates different prediction perspectives. We also proposed two concrete contact prediction algorithms by social network contact pattern, (1) the bursty contacts and (2) contact associate between different node pairs. Second, we propose a novel method for replica quota control by contact probabilities and delivery probabilities. Evaluation on seven traces shows that our routing algorithms can achieve robust performance with surprisingly low resource consumption.

The third part proposes the concept of change awareness in DTN information flow and evaluates its application in sensor data collection. The change awareness is a matric for measuring how much a node can be aware of the other nodes' update via intermittent connections. In data aggregation application, we would like to exploit the nodes having more information to aggregate data via DTN short distance communication and upload data to a server. We propose the data aggregation solution for mobile nodes global snapshot with the minimum use of infrastructure communication. Evaluation on real world traces shows that a sever can get real-time global snapshot on some types of sensor data with only 1/5 nodes' data upload.

## 6.2 The contribution to DTN industrialization

The three work in this thesis focus on DTN resources efficiency. The resource efficiency is one of the critical factors in the feasibility of practical DTNs. When users tolerate longer delays, they naturally expect less power consumption and finance cost. This thesis contributes to DTNs industrialization.

First, neighbor discovery can help decentralize friends discovery in mobile social network. We see many proximity based mobile applications, as in Table 1.1.2 in Section 1.1.2. Most of the applications still rely on centralized message exchange to discover close friends, which often use cellular data communication. Fast distributed neighbor discovery without relying on central server allows users without cellular data plan to discover and communicate with near-by friends and help cellular data plan users reduce cost. As a building block for mobile to mobile communication, a good neighbor discovery solution assuredly helps a lot of DTN applications. This can accelerate the development of DTN industrialization.

Second, the DTN routing and data aggregation help offload data from cellular networks to DTNs. In urban environment, more phones and vehicles create data to share and consume data to obtain information. This sharply increases the data communication requirements. It is desirous to offload cellular data communication to DTNs without spoiling application. In the second part of this thesis, Plankton was evaluated to be able to shorten data delivery latency while use less resource. The results apparently motivate to offload data to DTNs. In the third part of this thesis, an algorithm based on change awareness was proposed to use DTNs to aggregate data before using cellular network to communicate. The solution sharply reduces the number of nodes using cellular network for data communication. The work in these two parts definitely prompts practical ubiquitous computation and communication on smart phones and other types of smart devices.

## 6.3 Future work

It is a clear trend that mobile phones will replace personal computers as the first popular platform for information access, data generation, and computation. By market statistic, the number of sold phones are more than the number of sold PCs [1]. Thus, we can reasonably expect more mobile device based applications in the future. This also demands

[1] www.businessinsider.com

more researches on the types of DTNs disclosed in this thesis. The future research work on DTNs along the line of this thesis includes three main parts.

### 6.3.1 DTN killer application

Good applications drive both academic research and industrial implementation. More computation and communication power is available on mobile nodes, and applications on the Internet is migrating to mobile phones. In this thesis, we propose an application of data aggregation for ubiquitous environmental sensing in chapter 5. However, we are still seeking killer applications based on intermittent connections, i.e., DTN communication [37]. Many potential applications have been discussed in [37, 149, 150, 151, 36]. One probably promising direction is on the DTN consists of heterogeneous mobile nodes. What's the potential applications when we consider heterogeneous mobile nodes. Smart phones are the most popular nodes of DTNs. Smart phones can help carry and forward messages within a small community. Another types of mobile nodes, the vehicles, such as the cars and buses can help carry the messages in a larger area within a city. Another type of even fast and far-reaching nodes, the aircraft, can help relay messages inter-cities or inter-countries [152]. This will lead to large scale DTNs. We have seen proposals [153, 154] on large scale DTNs. More applications will be incubated with the more understanding on large scale and heterogeneous DTNs.

### 6.3.2 DTN connection prediction

The intermittent and non-scheduled connections are the root reason of most DTN challenges. If connections are well predicted, the neighbor discovery would be trivial, the routing algorithm would be deterministic (if the traffic load is within capability.) and the information aggregation would become drastically easier. However, the connection prediction is really challenging. In this thesis, as our contribution in chapter 4, we give a framework for integrating prediction results from different prediction perspectives, and a concrete social network based connection prediction algorithm. A big progress has been made, but we still see wide space to explore more accurate prediction algorithms.

From our experience on DTN connection prediction, we propose to predict the highly predictable connections, instead of predicting all connections as done by most DTN routing solutions. We have two reasons for this proposal. (i) It is extremely hard, if possible, to correctly predict all connections with high probability. A heuristic reason is that a node's movement, i.e., a man's movement, is often randomly. If we try to predict all, then a lot of noise results will be generated. If they are used for routing or applications, it would downgrade the performance. (ii) A small amount of correct connection prediction sufficiently helps DTN routing and applications. A node usually encounters hundreds or thousands of connections. For example, in Plankton, the utilization of reliable predictions of 1% to 2% of total connections has already significantly boost the performance.

Another important topic in DTN connection prediction is to discover hidden contact patterns. In most of existing solutions, recent contacts and exponential distribution based connection models are used to prediction connections. We do not see much data mining on mobile phone connections. Connections between pairs of nodes can have hidden patterns which cannot be easily revealed by existing techniques. Thus, algorithm design to reveal potential hidden patterns is an important topic in DTN connection prediction.

### 6.3.3 The distributed DTN data aggregation

In chapter 5, we present a solution for data aggregation on DTNs. The solution utilizes the control channel of cellular networks to collect nodes' last connection information. With these meta-data on last connections, a server computes nodes to upload information. The infrastructure communication e.g., cellular networks, is necessary given the server needs 100% snapshot of the whole system. However, if the requirement is downgraded, e.g., only a portion of data are required, a distributed solution, i.e., without centralized information collection of last connections, is more practical and economical. A distributed data aggregation solution is challenging. First, a node needs to distributively decide whether itself should aggregate and upload data to the server. Second, if a node is an uploader, it needs to decide the nodes it need to aggregate and upload to the server. To make proper decisions, a node needs to know other nodes' connections. However, a node's information on other nodes' connection is constrained with intermitted connections. To solve this challenge, a good connection prediction algorithm is required and a wise algorithm for aggregating data is required to reduce errors and data redundancy.

# Bibliography

[1] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34, New York, NY, USA, 2003. ACM.

[2] WiFi Alliance. Wi-fi peer-to-peer (p2p) technical specification v1.4. https://www.wi-fi.org/. updated on 30 Aug 2014.

[3] Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753, 2012.

[4] Pan Hui, Augustin Chaintreau, Richard Gass, James Scott, Jon Crowcroft, and Christophe Diot. Pocket switched networking: Challenges, feasibility and implementation issues. In *Autonomic Communication*, pages 1–12. Springer, 2006.

[5] Junxian Huang, Feng Qian, Alexandre Gerber, Z Morley Mao, Subhabrata Sen, and Oliver Spatscheck. A close examination of performance and power characteristics of 4g lte networks. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 225–238. ACM, 2012.

[6] Aaditeshwar Seth, Matei Zaharia, Srinivasan Keshav, and S Bhattacharyya. A policy-oriented architecture for opportunistic communication on multiple wireless networks. *Unpublished Manuscript. Available at http://blizzard. cs. uwaterloo. ca/keshav/home/Papers/data/06/ocmp. pdf*, 2006.

[7] Ganesh Ananthanarayanan and Ion Stoica. Blue-fi: enhancing wi-fi performance using bluetooth signals. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, MobiSys '09, pages 249–262, New York, NY, USA, 2009. ACM.

[8] Fai Cheong Choo, Mun Choon Chan, and E-C Chang. Robustness of DTN against routing attacks. In *Communication Systems and Networks (COMSNETS), 2010 Second International Conference on*, pages 1–10. IEEE, 2010.

[9] Chan Moon chun Chen Bin bin. Mobicent: a credit-based incentive system for disruption tolerant network. Technical report.

[10] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiuan Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet. In *ASPLOS-X: Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*, pages 96–107, New York, NY, USA, 2002. ACM.

[11] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen K. Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. CarTel: A Distributed Mobile Sensor Computing System. In *4th ACM SenSys*, Boulder, CO, November 2006.

[12] Nirmala Shenoy. A framework for seamless roaming across heterogeneous next generation wireless networks. *Wirel. Netw.*, 11(6):757–774, 2005.

[13] Philip Ginzboorg, Teemu Kärkkäinen, Aki Ruotsalainen, Mikael Andersson, and Jörg Ott. DTN communication in a mine. *ExtremeCom 2010*, 2010.

[14] Robert A Nichols and A Roger Hammons. Performance of DTN-based free-space optical networks with mobility. In *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pages 1–6. IEEE, 2007.

[15] Sujoy Saha, Anirudh Sheldekar, Rijo Joseph C, Amartya Mukherjee, and Subrata Nandi. Post disaster management using delay tolerant network. *Recent Trends in Wireless and Mobile Networks*, pages 170–184, 2011.

[16] Mehul Motani, Vikram Srinivasan, and Pavan S. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 243–257, New York, NY, USA, 2005. ACM.

[17] Jörg Ott and Dirk Kutscher. Bundling the Web: HTTP over DTN. *Proceedings of WNEPT*, 2006.

[18] Tuomo Hyyryläinen, Teemu Kärkkäinen, Cheng Luo, Valdas Jaspertas, Jouni Karvo, and Jörg Ott. Opportunistic email distribution and access in challenged heterogeneous environments. In *Proceedings of the second ACM workshop on Challenged networks*, pages 97–100. ACM, 2007.

[19] Martin Lukac, Lewis Girod, and Deborah Estrin. Disruption tolerant shell. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pages 189–196. ACM, 2006.

[20] Tuan-Minh Pham and Serge Fdida. DTN Support for News Dissemination in an Urban Area. *Comput. Netw.*, 56(9):2276–2291, June 2012.

[21] Bin Bin Chen and Mun Choon Chan. Mobtorrent: A framework for mobile internet access from vehicles. In *Proc. INFOCOM 2009. The 28th Conference on Computer Communications. IEEE*, pages 1404–1412, April 19–25, 2009.

[22] Aruna Balasubramanian, Yun Zhou, W Bruce Croft, Brian Neil Levine, and Aruna Venkataramani. Web search from a bus. In *Proceedings of the second ACM workshop on Challenged networks*, pages 59–66. ACM, 2007.

[23] Shane B. Eisenman, Emiliano Miluzzo, Nicholas D. Lane, Ronald A. Peterson, Gahng-Seop Ahn, and Andrew T. Campbell. Bikenet: A mobile sensing system

for cyclist experience mapping. *ACM Trans. Sen. Netw.*, 6(1):6:1–6:39, January 2010.

[24] Bo Han, Pan Hui, VS Anil Kumar, Madhav V Marathe, Jianhua Shao, and Aravind Srinivasan. Mobile data offloading through opportunistic communications and social participation. *Mobile Computing, IEEE Transactions on*, 11(5):821–834, 2012.

[25] Phewtick. http://www.phewtick.com/. checked on 30 Aug 2014.

[26] Skout. https://play.google.com/store/apps/details?id=com.skout.android. checked on 30 Aug 2014.

[27] Bump. http://bu.mp/company/. checked on 30 Aug 2014.

[28] Momo. http://www.immomo.com/. checked on 30 Aug 2014.

[29] TEEMU KARKKAINEN, Mikko Pitkanen, and IOERG OTT. Applications in delay-tolerant and opportunistic networks. *Mobile Ad Hoc Networking: The Cutting Edge Directions*, 35:317, 2013.

[30] Badoo. https://play.google.com/store/apps/details?id=com.badoo.mobile. checked on 30 Aug 2014.

[31] Circles. https://play.google.com/store/apps/details?id=com.discovercircle10. checked on 30 Aug 2014.

[32] Sonar. https://play.google.com/store/apps/details?id=me.sonar.android. checked on 30 Aug 2014.

[33] Waze. https://play.google.com/store/apps/details?id=com.waze. checked on 30 Aug 2014.

[34] Groupon. https://play.google.com/store/apps/details?id=com.groupon. checked on 30 Aug 2014.

[35] Foursquare. https://play.google.com/store/apps/details?id=com.joelapenna.foursquared. checked on 30 Aug 2014.

[36] Jon Crowcroft, Eiko Yoneki, Pan Hui, and Tristan Henderson. Promoting tolerance for delay tolerant network research. *SIGCOMM Comput. Commun. Rev.*, 38(5):63–68, 2008.

[37] Anders Lindgren and Pan Hui. The quest for a killer app for opportunistic and delay tolerant networks: (invited paper). In *CHANTS '09: Proceedings of the 4th ACM workshop on Challenged networks*, pages 59–66, New York, NY, USA, 2009. ACM.

[38] Aaron Carroll and Gernot Heiser. An analysis of power consumption in a smartphone. In *Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, pages 21–21. USENIX Association, 2010.

[39] Nathan Eagle and Alex (Sandy) Pentland. CRAWDAD trace mit/reality/blueaware/devicespan (v. 2005-07-01). Downloaded from http://crawdad.cs.dartmouth.edu/mit/reality/blueaware/devicespan, July 2005.

[40] Pierre Ugo Tournoux, Jérémie Leguay, Farid Benbadis, Vania Conan, Marcelo Dias de Amorim, and John Whitbeck. The accordion phenomenon: Analysis, Characterization, and Impact on DTN Routing. In *Proc. IEEE INFOCOM*, 2009.

[41] Brendan Burns, Oliver Brock, and Brian Neil Levine. Mora routing and capacity building in disruption-tolerant networks. *Ad hoc networks*, 6(4):600–620, 2008.

[42] Rahul C Shah, Sumit Roy, Sushant Jain, and Waylon Brunette. Data MULEs: modeling a three-tier architecture for sparse sensor networks. In *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pages 30 – 41, 11 2003.

[43] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198, New York, NY, USA, 2004. ACM.

[44] Wenrui Zhao, Yang Chen, Mostafa Ammar, Mark Corner, Brian Levine, and Ellen Zegura. Capacity enhancement using throwboxes in DTNs. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 31–40. IEEE, 2006.

[45] Nilanjan Banerjee, Mark D. Corner, Don Towsley, and Brian N. Levine. Relays, base stations, and meshes: enhancing mobile networks with infrastructure. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, pages 81–91, New York, NY, USA, 2008. ACM.

[46] Hyewon Jun, Mostafa H. Ammar, Mark D. Corner, and Ellen W. Zegura. Hierarchical power management in disruption tolerant networks with traffic-aware optimization. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS '06, pages 245–252, New York, NY, USA, 2006. ACM.

[47] Khaled A Harras, Mike P Wittie, Kevin C Almeroth, and Elizabeth M Belding. Paranets: A parallel network architecture for challenged networks. In *Mobile Computing Systems and Applications, 2007. HotMobile 2007. Eighth IEEE Workshop on*, pages 73–78. IEEE, 2007.

[48] Marc Liberatore, Brian Neil Levine, and Chadi Barakat. Maximizing transfer opportunities in bluetooth DTNs. In *Proceedings of the 2006 ACM CoNEXT conference*, page 3. ACM, 2006.

[49] Nilanjan Banerjee, Mark D. Corner, and Brian Neil Levine. throwboxesign and field experimentation of an energy-efficient architecture for DTN throwboxes. *IEEE/ACM Trans. Netw.*, 18:554–567, April 2010.

[50] Michael J McGlynn and Steven A Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 137–145. ACM, 2001.

[51] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA, 2005.

[52] Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Yueng Hsieh. Power-saving protocols for ieee 802.11-based multi-hop ad hoc networks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 200 – 209 vol.1, 2002.

[53] Rong Zheng, Jennifer C. Hou, and Lui Sha. Asynchronous wakeup for ad hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '03, pages 35–45, New York, NY, USA, 2003. ACM.

[54] Prabal Dutta and David Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 71–84, New York, NY, USA, 2008. ACM.

[55] Arvind Kandhalu, Karthik Lakshmanan, and Ragunathan Raj Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 350–361. ACM, 2010.

[56] Mehedi Bakht, Matt Trower, and Robin Hilary Kravets. Searchlight: won't you be my neighbor? In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 185–196. ACM, 2012.

[57] Daniel Camps-Mur. Using deployed wi-fi access points to enhance asynchronous wake up protocols.

[58] Dong Li and Prasun Sinha. Rbtp: Low-power mobile discovery protocol through recursive binary time partitioning. *Mobile Computing, IEEE Transactions on*, 13(2):263–273, Feb 2014.

[59] D. Camps-Mur and P. Loureiro. E2d wi-fi: A mechanism to achieve energy efficient discovery in wi-fi. *Mobile Computing, IEEE Transactions on*, PP(99):1–1, 2013.

[60] Tian Hao, Ruogu Zhou, Guoliang Xing, and Matt Mutka. Wizsync: Exploiting wi-fi infrastructure for clock synchronization in wireless sensor networks. In *Real-Time Systems Symposium (RTSS), 2011 IEEE 32nd*, pages 149–158. IEEE, 2011.

[61] Zhensheng Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Communications Surveys Tutorials*, 8(1):24–37, 2006.

[62] Thrasyvoulos Spyropoulos, Rao Naveed Rais, Thierry Turletti, Katia Obraczka, and Athanasios Vasilakos. Routing for disruption tolerant networks: taxonomy and design. *Wirel. Netw.*, 16(8):2349–2370, November 2010.

[63] Ioannis Psaras, Lloyd Wood, and Rahim Tafazolli. Delay-/disruption-tolerant networking: State of the art and future challenges. *www.ee. ucl. ac. uk/ũceeips/dtn-srv-ipsaras.pdf*, 2010.

[64] Cong Liu and Jie Wu. Routing in a cyclic mobispace. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 351–360, New York, NY, USA, 2008. ACM.

[65] Cong Liu and Jie Wu. Scalable routing in delay tolerant networks. In *MobiHoc*

*'07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 51–60, New York, NY, USA, 2007. ACM.

[66] Aruna Balasubramanian, Brian Neil Levine, and Arun Venkataramani. Replication Routing in DTNs: A Resource Allocation Approach. *Networking, IEEE/ACM Transactions on*, 18(2):596 –609, april 2010.

[67] Kyunghan Lee, Yung Yi, Jaeseong Jeong, Hyungsuk Won, Injong Rhee, and Song Chong. Max-contribution: On optimal resource allocation in delay tolerant networks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 2010.

[68] Matthias Grossglauser and Martin Vetterli. Locating nodes with EASE: last encounter routing in ad hoc networks through mobility diffusion. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1954 – 1964 vol.3, 30 2003.

[69] Henri Dubois-Ferriere, Matthias Grossglauser, and Martin Vetterli. Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 257–266, New York, NY, USA, 2003. ACM.

[70] Natasa Sarafijanovic-Djukic, Natasa Sara Janovic-djukic, and Matthias Grossglauser. Last encounter routing under random waypoint mobility. In *In Proc. IFIP-TC6 NETWORKING*, pages 974–988, 2004.

[71] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference on*, pages 79 –85, 19-23 2007.

[72] John Burgess, Brian Gallagher, David Jensen, and Brian Neil Levine. Max-

prop: Routing for vehicle-based disruption-tolerant networks. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1 –11, April 2006.

[73] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *Lecture Notes in Computer Science*, 3126:239–254, January 2004.

[74] Yun Li, Xin Li, Qilie Liu, and Zhanjun Liu. E-prophet: a novel routing protocol for intermittently connected wireless networks. In *IWCMC '09: Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing*, pages 452–456, New York, NY, USA, 2009. ACM.

[75] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 32–40, New York, NY, USA, 2007. ACM.

[76] Jó Ágila Bitsch Link, Nicolai Viol, André Goliath, and Klaus Wehrle. Simbetage: utilizing temporal changes in social networks for pocket switched networks. In *U-NET '09: Proceedings of the 1st ACM workshop on User-provided networking: challenges and opportunities*, pages 13–18, New York, NY, USA, 2009. ACM.

[77] Theus Hossmann, Thrasyvoulos Spyropoulos, and Franck Legendre. Know Thy Neighbor: Towards Optimal Mapping of Contacts to Social Graphs for DTN Routing. In *INFOCOM, 2010 Proceedings IEEE*, pages 1 –9, 14-19 2010.

[78] Pan Hui and Jon Crowcroft. How small labels create big improvements. *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference on*, pages 65 –70, march 2007.

[79] Pan Hui, Eiko Yoneki, Shu Yan Chan, and Jon Crowcroft. Distributed commu-

nity detection in delay tolerant networks. In *MobiArch '07: Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, pages 1–8, New York, NY, USA, 2007. ACM.

[80] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: Social-based forwarding in delay-tolerant networks. *Mobile Computing, IEEE Transactions on*, 10(11):1576–1589, 2011.

[81] Lei Tang, Qunwei Zheng, Jun Liu, and Xiaoyan Hong. Selective message forwarding in delay tolerant networks. *Mob. Netw. Appl.*, 14(4):387–400, 2009.

[82] E. Bulut and B.K. Szymanski. Friendship based routing in delay tolerant mobile social networks. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–5. IEEE, 2010.

[83] Jie Wu and Yunsheng Wang. Social feature-based multi-path routing in delay tolerant networks. In *2012 Proceedings IEEE INFOCOM*, pages 1368–1376. IEEE, March 2012.

[84] Brenton D Walker, Joel K Glenn, and T Charles Clancy. Analysis of simple counting protocols for delay-tolerant networks. In *CHANTS '07: Proceedings of the second ACM workshop on Challenged networks*, pages 19–26, New York, NY, USA, 2007. ACM.

[85] Matthias Grossglauser and David N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.*, 10(4):477–486, 2002.

[86] Venkatesh Ramaiyan, Eitan Altman, and Anurag Kumar. Delay optimal scheduling in a two-hop vehicular relay network. *Mob. Netw. Appl.*, 15(1):97–111, 2010.

[87] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, 2005. ACM.

[88] Zygmunt J. Haas, Joseph Y. Halpern, and Li Li. Gossip-based ad hoc routing. *IEEE/ACM Trans. Netw.*, 14(3):479–491, 2006.

[89] S.C. Nelson, M. Bakht, and R. Kravets. Encounter-based routing in DTNs. In *INFOCOM 2009, IEEE*, pages 846–854. IEEE, 2009.

[90] Eyuphan Bulut, Zijian Wang, and Boleslaw Karol Szymanski. Cost-effective multiperiod spraying for routing in delay-tolerant networks. *IEEE/ACM Transactions on Networking (TON)*, 18(5):1530–1543, 2010.

[91] R. Rajagopalan and P.K. Varshney. Data-aggregation techniques in sensor networks: a survey. *Communications Surveys Tutorials, IEEE*, 8(4):48–63, 2006.

[92] Elena Fasolo, Michele Rossi, Jorg Widmer, and Michele Zorzi. In-network aggregation techniques for wireless sensor networks: a survey. *Wireless Communications, IEEE*, 14(2):70–87, April 2007.

[93] Jie Gao, Leonidas Guibas, N. Milosavljevic, and John Hershberger. Sparse data aggregation in sensor networks. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 430–439, April 2007.

[94] Bhaskar Krishnamachari, Deborah Estrin, and Stephen Wicker. The impact of data aggregation in wireless sensor networks. In *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, pages 575–578, 2002.

[95] M. Haase and D. Timmermann. energy adaptive clustering hierarchy with deterministic cluster-head selection. In *In IEEE Conference on Mobile and Wireless Communications Networks (MWCN) (2002*, pages 368–372.

[96] Ossama Younis and Sonia Fahmy. HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *Mobile Computing, IEEE Transactions on*, 3(4):366–379, Oct 2004.

[97] Tamer Nadeem, Sasan Dashtinezhad, Chunyuan Liao, and Liviu Iftode. Trafficview: traffic data dissemination using car-to-car communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 8(3):6–19, 2004.

[98] Ismail Salhi, Mohamed Oussama Cherif, and Sidi Mohammed Senouci. A new architecture for data collection in vehicular networks. In *Proceedings of the 2009 IEEE international conference on Communications*, ICC'09, pages 2705–2710, Piscataway, NJ, USA, 2009. IEEE Press.

[99] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *Ad-hoc, Mobile, and Wireless Networks*, pages 346–359, 2011.

[100] John Whitbeck, Marcelo Dias de Amorim, Vania Conan, and Jean-Loup Guillaume. Temporal reachability graphs. pages 377–388, 2012.

[101] Alejandro Cornejo, Seth Gilbert, and Calvin Newport. Aggregation in dynamic networks. In *Proceedings of the 2012 ACM symposium on Principles of distributed computing*, pages 195–204. ACM, 2012.

[102] Ting Liu, Christopher M Sadler, Pei Zhang, and Margaret Martonosi. Implementing software on resource-constrained mobile sensors: Experiences with impala and zebranet. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 256–269. ACM, 2004.

[103] Hengchang Liu, Jingyuan Li, Zhiheng Xie, Shan Lin, Kamin Whitehouse, John A. Stankovic, and David Siu. Automatic and robust breadcrumb system deployment for indoor firefighter applications. In *MobiSys '10: Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 21–34, New York, NY, USA, 2010. ACM.

[104] Jyh-How Huang, Saqib Amjad, and Shivakant Mishra. Cenwits: a sensor-based loosely coupled search and rescue system using witnesses. In *Proceedings of*

*the 3rd international conference on Embedded networked sensor systems*, pages 180–191. ACM, 2005.

[105] Stuart M. Allen, Gualtiero Colombo, and Roger M. Whitaker. Uttering: social micro-blogging without the internet. In *Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, MobiOpp '10, pages 58–64, New York, NY, USA, 2010. ACM.

[106] https://www.facebook.com/. checked on 30 Aug 2014.

[107] https://www.linkedin.com/. checked on 30 Aug 2014.

[108] Jehn-Ruey Jiang, Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Hwang Lai. Quorum-based asynchronous power-saving protocols for ieee 802.11 ad hoc networks. *Mobile Networks and Applications*, 10(1-2):169–181, 2005.

[109] Steven A Borbash, Anthony Ephremides, and Michael J McGlynn. An asynchronous neighbor discovery algorithm for wireless sensor networks. *Ad Hoc Networks*, 5(7):998–1016, 2007.

[110] Fred L Walls and Jean-Jacques Gagnepain. Environmental sensitivities of quartz oscillators. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on*, 39(2):241–249, 1992.

[111] Amin Vahdat, David Becker, et al. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, 2000.

[112] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S Raghavendra. Single-copy routing in intermittently connected mobile networks. In *Proc. First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks IEEE SECON 2004*, pages 235–244, October 4–7, 2004.

[113] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[114] Quan Yuan, Ionut Cardei, and Jie Wu. Predict and relay: an efficient routing in disruption-tolerant networks. In *MobiHoc '09: Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*, pages 95–104, New York, NY, USA, 2009. ACM.

[115] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 241–250, New York, NY, USA, 2008. ACM.

[116] Vijay Erramilli, Mark Crovella, Augustin Chaintreau, and Christophe Diot. Delegation forwarding. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 251–260, New York, NY, USA, 2008. ACM.

[117] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 145–158, New York, NY, USA, 2004. ACM.

[118] Aruna Balasubramanian, Brian Levine, and Arun Venkataramani. Dtn routing as a resource allocation problem. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 373–384, New York, NY, USA, 2007. ACM.

[119] James A Davis, Andrew H Fagg, and Brian Neil Levine. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks. In *ISWC '01: Proceedings of the 5th IEEE International Symposium on Wearable Computers*, page 141, Washington, DC, USA, 2001. IEEE Computer Society.

[120] Joy Ghosh, Hung Q. Ngo, and Chunming Qiao. Mobility profile based routing within intermittently connected mobile ad hoc networks (icman). In *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, pages 551–556, New York, NY, USA, 2006. ACM.

[121] James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD trace cambridge/haggle/imote/infocom2006 (v. 2009-05-29). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle /imote/infocom2006, May 2009.

[122] Michal Piorkowski, Natasa Sarafijanovoc-Djukic, and Matthias Grossglauser. A Parsimonious Model of Mobile Partitioned Networks with Clustering. In *The First International Conference on COMmunication Systems and NETworkS (COMSNETS)*, January 2009.

[123] Jorjeta G. Jetcheva, Yih-Chun Hu, Santashil PalChaudhuri, Amit Kumar Saha, and David B. Johnson. CRAWDAD data set rice/"ad hoc city" (v. 2003-09-11). Downloaded from http://crawdad.cs.dartmouth.edu/rice/ad_hoc_city, September 2003.

[124] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching-Chuan Chiang. A group mobility model for ad hoc wireless networks. In *MSWiM '99: Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 53–60, New York, NY, USA, 1999. ACM.

[125] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *Information Theory, IEEE Transactions on*, 23(3):337–343, 1977.

[126] Sajal K Das, Diane J Cook, Amiya Battacharya, Edwin O Heierman III, and Tze-Yun Lin. The role of prediction algorithms in the mavhome smart home architecture. *Wireless Communications, IEEE*, 9(6):77–84, 2002.

[127] Peter Csermely. *Weak Links: The universal key to the stability of networks and complex systems*. Springer Verlag, 2009.

[128] Augustin Chaintreau, Abderrahmen Mtibaa, Laurent Massoulie, and Christophe Diot. The diameter of opportunistic mobile networks. In *Proceedings of the 2007 ACM CoNEXT conference*, CoNEXT '07, pages 12:1–12:12, New York, NY, USA, 2007. ACM.

[129] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The ONE simulator for DTN protocol evaluation. In *Simutools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[130] Samo Grasic and Anders Lindgren. An analysis of evaluation practices for DTN routing protocols. In *Proceedings of the seventh ACM international workshop on Challenged networks*, CHANTS '12, pages 57–64, New York, NY, USA, 2012. ACM.

[131] Pan Hui, Augustin Chaintreau, James Scott, Richard Gass, Jon Crowcroft, and Christophe Diot. Pocket switched networks and human mobility in conference environments. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 244–251, New York, NY, USA, 2005. ACM.

[132] Yasser Toor, Paul Muhlethaler, and Anis Laouiti. Vehicle ad hoc networks: Applications and related technical issues. *Communications Surveys & Tutorials, IEEE*, 10(3):74–88, 2008.

[133] B Bui Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. 2002.

[134] David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference prob-

lems for temporal networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 504–513. ACM, 2000.

[135] Kenneth A. Berman. Vulnerability of scheduled networks and a generalization of menger's theorem. *Networks*, 28(3):125–134, 1996.

[136] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. Deterministic computations in time-varying graphs: Broadcasting under unstructured mobility. *Theoretical Computer Science*, pages 111–124, 2010.

[137] Andrea EF Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time of edge-markovian evolving graphs. *SIAM journal on discrete mathematics*, 24(4):1694–1712, 2010.

[138] Kristina Lerman and Rumi Ghosh. Information contagion: An empirical study of the spread of news on Digg and Twitter social networks. In *Proceedings of 4th International Conference on Weblogs and Social Media (ICWSM)*, 2010.

[139] Xufei Wang, Huan Liu, Peng Zhang, and Baoxin Li. Identifying information spreaders in twitter follower networks. Technical Report TR-12-001, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85287, USA, 2012.

[140] Gueorgi Kossinets, Jon Kleinberg, and Duncan Watts. The structure of information pathways in a social communication network. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 435–443. ACM, 2008.

[141] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. Measuring temporal lags in delay-tolerant networks. In *Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 209–218. IEEE, 2011.

[142] John Tang, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Vincenzo Nicosia. Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, SNS '10, pages 3:1–3:6, New York, NY, USA, 2010. ACM.

[143] Rumi Ghosh and Kristina Lerman. Predicting influential users in online social networks. *arXiv preprint arXiv:1005.4882*, 2010.

[144] K. Mani Chandy and Leslie Lamport. Distributed snapshots: determining global states of distributed systems. *ACM Trans. Comput. Syst.*, 3(1):63–75, February 1985.

[145] Wendi Beth Heinzelman. *Application-specific protocol architectures for wireless networks*. PhD thesis, Massachusetts Institute of Technology, 2000.

[146] Razvan Cristescu, Baltasar Beferull-Lozano, and Martin Vetterli. On network correlated data gathering. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 4, pages 2571–2582. IEEE, 2004.

[147] Michael L Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[148] Vasek Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235, 1979.

[149] Wei Suna, Congmin Liu, and Dan Wang. On delay-tolerant networking and its application. *International Proceedings of Computer Science & Information Technology*, 51, 2012.

[150] Kartik Sankaran. Middleware for Accelerated DTN Application Development. http://www.comp.nus.edu.sg/ kartiks/nusdtn/Downloads/KartikSankaran_GRP.pdf.

[151] Thomas Jonson, Jonah Pezeshki, Victor Chao, Kristofer Smith, and James Fazio. Application of delay tolerant networking (DTN) in Airborne Networks. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1–7, Nov 2008.

[152] Ruben Martinez-Vidal, Ramon Marti, and Joan Borrell. Characterization of a transoceanic aircraft delay tolerant network. In *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pages 565–572, Oct 2013.

[153] Yong Li, Depeng Jin, Pan Hui, Li Su, and Lieguang Zeng. Revealing contact interval patterns in large scale urban vehicular ad hoc networks. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 299–300, New York, NY, USA, 2012. ACM.

[154] Pan Hui, Richard Mortier, Michal Piórkowski, Tristan Henderson, and Jon Crowcroft. Planet-scale human mobility measurement. In *Proceedings of the 2nd ACM International Workshop on Hot Topics in Planet-scale Measurement*, HotPlanet '10, pages 1:1–1:5, New York, NY, USA, 2010. ACM.

# Appendix A

# Appendix for neighbor discovery (Chapter 3)

## A.1 The number of required reference APs

For locally synchronous protocols, a phone uses the timestamp in the AP beacon as a time reference. In urban environment, one phone can usually overhear multiple APs, and different APs are usually not synchronized with each other. Hence, if two neighbor phones synchronize to two different APs, there are high chance that they won't be able to wake up at the same time.

One naive way to solve this problem is to let a phone use all APs that it can overhear as references. This guarantees that two neighbor phones will sync to the same reference if they have at least one common AP. However, the large number of APs in an environment also makes this approach expensive.

We find that by choosing a very small number of APs (e.g., as low as $\kappa = 3$) with a simple strategy, two neighbor phones can already have very high chance ($> 99\%$) to select at least one shared reference AP. We show this by both analysis and simulation.

In the analysis, we assume both APs and phones are distributed uniformly and independently in the space. We assume a unit disk model, where a phone can communicate with an AP if their distance is less than $r$, and two phones can communicate with each

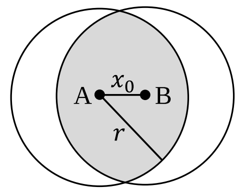



Figure A.1: The area where the reference APs for two neighbor phones can appear.

other if their distance is fewer than $d$. Since APs normally have higher antenna gain than phones, $d$ is usually smaller than $r$. We also assume a dense AP environment, where each phone can hear at least 3 APs.

We let a phone to choose $\kappa$ APs in the following way: it uses a global hashing function to hash each AP's BSSID (i.e. its MAC address) to a number. Denote this hashed number for the $i$th AP by $h_i$. Out of all APs that a phone can overhear, it synchronizes with the $\kappa$ APs of the minimum $h_i$s. We call this the minimum-$\kappa$ approach.

Consider two neighbor phones. Since they can hear each other, their distance $x$ is in the range of $[0, d]$. Since they are uniformly distributed in the space, $Prob[x < x_0] = (\frac{x_0}{d})^2$, for $x_0 \in [0, d]$.

Consider two phones $A$ and $B$ that are $x_0$ distance apart. As illustrated in Figure A.1, an AP in the union of the two disks (each with a radius of $r$) can be heard by at least one phone, and an AP in the overlapped section of the two disks (i.e., the gray section) can be heard by both phones. The area of the gray section is:

$$A = 2r^2 cos^{-1}(\frac{x_0}{2r}) - \frac{x_0}{2}\sqrt{4r^2 - x_0^2} \geq \pi r^2 - \frac{2}{3}\pi r x_0$$

For phone $A$, since it chooses the $\kappa$ APs using the minimum-$\kappa$ approach and since APs are uniformly randomly distributed, the probability for the event $\Phi$ that none of these $\kappa$ APs fall in the grey area is

$$Pr[\Phi] = (1 - \frac{A}{\pi r^2})^\kappa \leq (\frac{2x_0}{3r})^\kappa$$

If both phone $A$ and phone $B$ choose at least one AP in the gray section, they must

| | $\kappa = 3$ | $\kappa = 4$ | $\kappa = 5$ |
|---|---|---|---|
| $\frac{d}{r} = 1/2$ | 97.1% | 99.2% | 99.8% |
| $\frac{d}{r} = 2/3$ | 93.2% | 97.4% | 99.0% |
| $\frac{d}{r} = 3/4$ | 90.4% | 96.0% | 98.2% |

Table A.1: The probability that two neighbor phones choose at least one common reference AP.

choose at least one common AP, i.e., the one with the smallest hashed value out of all APs in the gray section. Hence, the probability that they don't choose a common AP is equal to the probability that at least one of them do not select any AP in the gray section. This probability is:

$$2 \times Pr[\Phi] - Pr[\Phi]^2 \leq 2 \times (\frac{2x_0}{3r})^{\kappa} - (\frac{2x_0}{3r})^{2\kappa}$$

Now integrate this probability over all possible $x_0 \in [0, d]$ according to the probability that $Pr[x < x_0] = (\frac{x_0}{d})^2$, the probability for this happen between two randomly selected neighbor phones becomes no greater than:

$$\frac{4}{\kappa+2} \times (\frac{2d}{3r})^{\kappa} - \frac{1}{\kappa+1} \times (\frac{2d}{3r})^{2\kappa}$$

We evaluate this probability for different $\frac{d}{r}$ ratios and for different values of $\kappa$, and the results are summarized in Table A.1.

When we can tolerate a 3% of missing rate, for $\frac{d}{r} = \frac{1}{2}$, $\kappa = 3$ suffices, and even for $\frac{d}{r} = \frac{3}{4}$, $\kappa = 5$ suffices.

# Appendix B

# Appendix for change awareness based data aggregation (Chapter 5)

**Proof of Lemma 1:** First, we prove $C(p_i) \bigcap C(p_j) = \phi, p_i! = p_j$. $\forall v \in V$, if $v$ is isolated in $\mathcal{G}$, then $v$ only occurs in $C(v)$. The lemma holds. If $v$ is connected, two cases can occur. (i) Other nodes' coverage joins $C(v)$, but $C(v)$ never joins other nodes' coverage. It is thereby obviously that $v$ only occurs in $C(v)$. (ii) $C(v)$ joins other node's coverage when $v$'s last connection is processed. In this case, nodes in $C(v)$ join other node, say $u_1$'s coverage. $C(v)$ is removed; $v$ only occurs in $C(u_1)$. It might happen that $v$ moves again as $u_1$ joins other node's coverage, but $v$ only occurs in one node's coverage by induction. Therefore, $C(p_i) \bigcap C(p_j) = \phi$, when $p_i! = p_j$.

Second, we prove $\bigcup_{v \in \mathcal{P}} C(v) = V$. It is obviously that in the initialization step each node occurs in its own fresh coverage. Later, before a fresh coverage is removed, vertices in the coverage is moved to another nodes' coverage; no vertex is not covered. Therefore, $\bigcup_{v \in V} C(v) = V$. $\square$

**Proof of Lemma 2** An informative node, $p$, is either isolated or connected in $\mathcal{G}$. When $p$ is isolated, this lemma obviously holds since $C(p)$ equals $\{p\}$. When $p$ is connected, for $u \in C(p), u \neq p$, if $u$'s last connection is between $u$ and $p$, the last connection is the fresh information path from $u$ to $p$. If $u$'s last connection is between $u$

and another node, say $u_1$, then $u \in C(u_1)$ when $u$'s last connection is processed. $u$ has information path to $u_1$. Because $u$ is in $C(p)$, $C(u_1)$ must sweep a sequence of nodes called as $u_2, u_3, \cdots, u_k$, till it joins $c(p)$. Because connections are processed in time increasing order, the adjacent nodes in $u_1, u_2, \cdots, u_k, p$ can form an information path from $u_1$ to $p$, and the start time of the information path is no earlier than the time of $u$'s last connection. By the definition of fresh information path, $u$ has a fresh information path to $p$.$\square$

**Proof of Lemma 3:** By Lemma 1, algorithm 4 generates a partition (by panorama informative node set coverage) of $V$. Let the partition be $M : \{C(p_i), p_i \in S, i = 1, 2, \cdots, l_1\}$ with poster set $S$.

We prove it by confliction. Let suppose there exist $S'$, and $|S'| < |S|$. The coverage of nodes in $S'$ generates another partition of $V$, as $M'$: $\{C(p'_j), p'_j \in V, j = 1, 2, \cdots, l_2\}$, with poster set $S'$, $l_2 < l_1$. Both $M$ and $M'$ satisfy Lemma 1 and Lemma 2. (By the concept, a panorama informative node set ($\mathcal{P}$) coverage should follow Lemma 1 and 2.)

Firstly, we adjust $S$. for $p \in S$, and $p$'s last connection is $< p, x, t_p >$, if $x \in S'$, we substitute $x$ for $p$. This adjustment still makes $S$ is a valid output from algorithm 4 without changing the size of $S$.

As $|S'| < |S|$, therefore, $\exists p \in S$ while $\exists p \notin S'$. As $M'$ satisfies Lemma 1, then $\exists p' \in S'$ where $p \in C(p')$. As $M's$ satisfies Lemma 2, therefore $F(\mathcal{G}, p, p', t) = 1$, i.e., there is path between $(p, u_1, t_1) \rightsquigarrow (u_1, u_2, t_2) \rightsquigarrow \cdots \rightsquigarrow (u_{k-1}, u_k, t_k)$, where $u_k = p'$, and $t_1 < t_2 < \cdots < t_k$. Please note that $p' \neq u_1$ is not possible as we have done aforementioned adjustment. $(p, u_1, t_1)$ is the p's last connection before $t$. This conflicts with the fact that both $p$ and $u_1$ are isolated after removing $(p, u_1, t_1)$ because $p$ is a poster in output from Algorithm 4. Thus, there exist no such $p$. that is no such $S'$ that $|S|$ is smaller than $|S'|$.$\square$