Hop-by-Hop Reliable, Parallel Message Propagation for Intermittently-Connected Mesh Networks

Hideya Ochiai
The University of Tokyo/NICT
jo2lxq@hongo.wide.ad.jp

Masaya Nakayama
The University of Tokyo/NICT
nakayama@nc.u-tokyo.ac.jp

Hiroshi Esaki
The University of Tokyo/NICT
hiroshi@wide.ad.jp

***Abstract*—Wireless mesh networks suffer from intermittent connectivity, and thus hop-by-hop reliability and parallel message propagation, which DTN researches have explored, can be applied to allow scalable message propagation over such challenged network environments. We implemented those communication schemes onto UTMesh – 50-node scale wireless mesh network testbed, and studied the delivery patterns. On the evaluation result with UTMesh, we have confirmed (1) that hop-by-hop reliability scheme achieves scalable message propagation (e.g., 23 hops), and (2) that message propagation speed increases as the redundancy-level increases. We have also observed that the smallest hop count path does not always achieve the fastest message delivery. This was probably because longer distant links were unstable and message paths over short distant links provided faster propagation.**

***Index Terms*—Delay Tolerant Networks, Mobile Ad Hoc Networks, Intermittent Connectivity, Testbed**

I. INTRODUCTION

Wireless mesh networks frequently suffer from intermittent connectivity even if the network nodes do not move. Intermittent connectivity has been pointed out mostly in the context of mobile cases[13][2][12], and considered as an application area of delay tolerant networking (DTN)[3][5] for such challenged network environments. However, according to the study with real equipments, even static neighbors frequently disappear and become disconnected.

This raises intermittently-connected mesh networks (ICMeN), indicating that this is also an application area of DTN. For this reason, the traditional communication schemes, such as (1) best effort forwarding with end-to-end reliability and (2) message propagation on a single path, sometimes do not work well[25]. Instead, we should take the approaches that DTN researches have explored; i.e., (1) store-and-forward with hop-by-hop reliability and (2) parallel message propagation. We must node that ICMeN is different from intermittently-connected mobile networks (ICMN)[8][10][17][23] because nodes are physically fixed.

In this work, we implemented such DTN communication schemes into UTMesh (Fig. 1) and studied how they enabled scalable message propagation. UTMesh is our real world ICMeN: 50-node scale wireless mesh network. We demonstrate that links among static nodes are highly dynamic and that we have to take those schemes in order to increase the scalability in hop counts and the propagation speed.

The traditional approach [4][19] for wireless mesh networking was mostly based on the Internet design principles: (1) best effort forwarding with end-to-end reliability, and (2) single message path. However, the target environment became quite different, which is characterized by intermittent-connectivity and unintentional packet loss. Packet loss sharply kills the traffic and sometimes causes delivery failure under best-effort communication. Intermittent-connectivity frequently changes the physical network topology and the best path soon becomes obsolete or unavailable even during one message delivery. For these reasons, they do not have scalability in hop counts[25].

Fig. 1. UTMesh overview: 50 wireless nodes (5 lines × 10 rows) gathered at the laboratory, in Eng. Bldg. 2 at the University of Tokyo.

We analyzed the features of wireless links using UTMesh. The result shows that the mesh network topology is certainly highly dynamic – almost always changing (see Section III). This supports the hypothesis that traditional approaches do not work well in such wireless networks.

The researches of DTNs have recently shown that hop-by-hop reliability and parallel delivery (i.e., redundant-delivery over multiple paths) greatly improve the message propagation speed as well as the scalability with regard to the hop count. However, those studies were made mostly in mobile cases[22][18][11]. Hop-by-hop reliable transfer is made by assuring that the message has been certainly transferred to the nexthop node. In this scheme, it repeatedly retries this forwarding process if the receiver could not get the message in the previous transmission. This scheme certainly propagates messages to the nexthop and achieves the scalability in hop counts. By making branch paths at the intermediate nodes

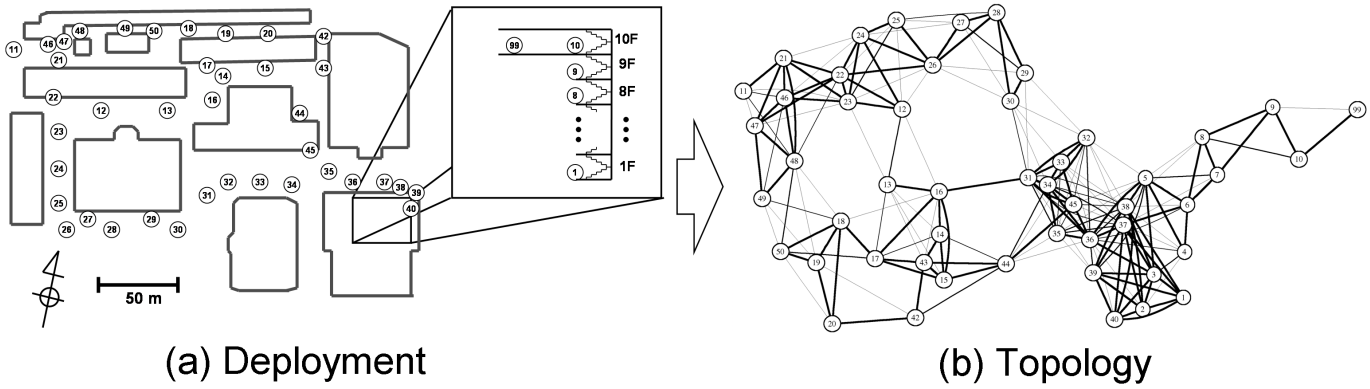

Fig. 2. (a) The experiment setting to analyze intermittent links between wireless nodes deployed in ad hoc manner. (b) Summarized network topology; bolder link indicates higher link availability.

during the delivery, DTN performs message propagation in parallel.

We applied those communication schemes to UTMesh and studied the features of message propagation. We have used potential-based entropy adaptive routing (PEAR)[18] as an implementation of DTNs.

We consider that wireless mesh networking enable rapid and costless deployment of smart meters into green buildings[29]. In such applications, the network must autonomously route messages under the given deployment configurations. This work itself does not focus on the method of node placement[6][20], or high bit rate application[15], or reduction of power consumption[21]. Sensors at power distribution boards, lights and HVAC systems actually do not suffer from power constraints.

The rest of this paper is organized as follows. Section II addresses the related works. In section III, we provide our analysis of wireless links on UTMesh. Section IV describes PEAR focusing on the behavior in static cases. Section V provides our evaluation work. In section VI, we provide the discussions on the results. Finally, we conclude this paper in section VII.

II. Related Work

Link-level measurement on mesh networks was well studied by Daniel et. al.[1] with Roofnet: wireless mesh networking testbed (38 nodes). They concluded "there is no clear distinction between working and non-working links". As for routing on mesh networks, Tschudin et. al.[24][25] discussed the existence of "Ad Hoc Horizon" – "at 2-3 hops and 10-20 nodes where the benefit from multihop ad hoc networking virtually vanishes" from the experiences on APE testbed (37 nodes)[14]. The researches on our UTMesh (51 nodes) have also verified their conclusions.

The benefits of hop-by-hop transfer are well summarized by Heimlicher et. al. [9] (also discussed in many literatures). Related to this, delay or disruption tolerant networking (DTN)[3][5], originally proposed for deep space communication, has been identified as an applicable scheme for intermittently-connected networks. Message routing on DTNs (or intermittently-connected networks) were studied mostly for mobile cases[22][18][13][2] in the last 5 years, and now it is widely acknowledged that multipath delivery improves message propagation speed and delivery probability. We applied the communication scheme even to static cases in this paper.

There has been several studies on multipath or redundant-path packet propagation for mobile ad hoc networks (MANET) for a decade. Stephen et. al.[16] discussed the fault tolerance of multipath routing in MANETs. Tsirigos et. al.[26] provided an analytical work on the benefits obtained from multipath schemes. However, these works seem to assume best-effort forwarding and end-to-end reliability. In our survey, most of the studies are not tested with real implementations.

The system model of PEAR, which we describe in this paper, was defined by our previous literature[18]. However, we described in the context of mobile cases. In this paper, we describe PEAR in static cases and provide our testbed experiments on UTMesh.

III. Intermittently-Connected Mesh Networks

The definition of intermittently-connected mesh networks (ICMeN) is as follows. An ICMeN is composed of stable wireless nodes, however, the links among them are disruptive and unreliable; i.e., they sometimes become connected but also frequently become disconnected. According to our testbed experiments, the typical wireless mesh networks (i.e., composed of 802.11b ad hoc mode) falls into this category.

In this section, we first describe our experiment settings, then show the results. We deployed 50 wireless nodes in our university campus and studied the features of the links. The results strongly indicate that the network topology frequently changes even if nodes are static.

A. Experiment Setting

We deployed 50 nodes in Hongo campus at the University of Tokyo as Fig. 2 (a). Each node was working with Armadillo-220, a Linux[1] embedded computer with an USB

[1]Kernel: linux-2.6.12.3-a9-15

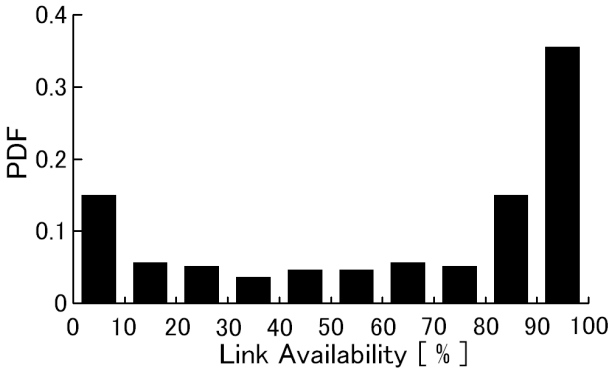

Fig. 3. The distribution of link availability – only 35% was tightly connected.

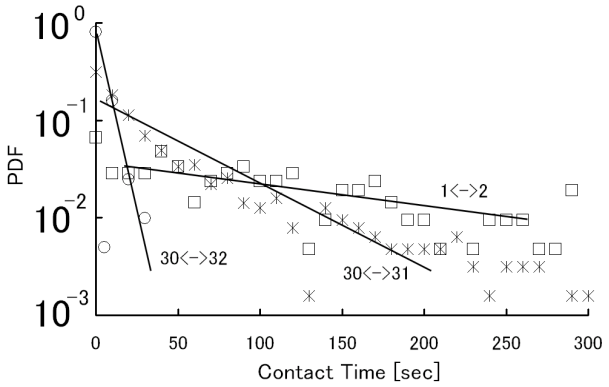

Fig. 4. Distributions of contact time at 1 ↔ 2 (91% availability), 30 ↔ 31(55%) and 30 ↔ 32(3%).

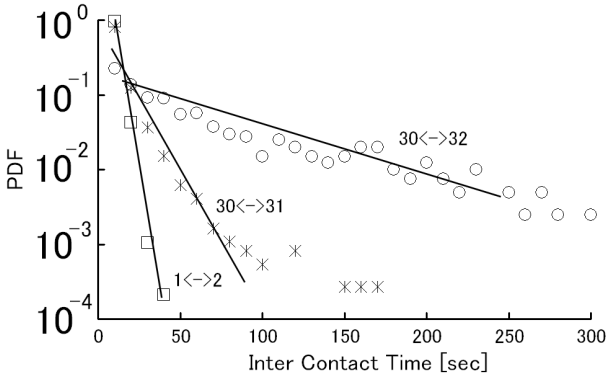

Fig. 5. Distributions of inter contact time at 1 ↔ 2 (91% availability), 30 ↔ 31(55%) and 30 ↔ 32(3%).

IEEE802.11b/g/n module[2]. The IEEE802.11b/g/n module was working in ad hoc mode (of 11b) at channel 1; all the nodes had the same frequency.

The embedded computer was powered by enough batteries (actually, we do not focus on the power usage – it just equipped enough power for the experiment). We packed all of them into a plastic box. Fig. 1 is the overview of the testbed (before deploying into the campus).

In order to analyze the features of wireless links (e.g., availability, contact time, inter contact time), we installed the software that made radio range advertisement in every 10 second. By recording the advertisements received from neighbors at each node, we performed this investigation. The experiment was carried out for 6 hours.

B. Link Availability and Network Topology

Fig. 2 (b) is the summarized network topology. The boldness of links indicates the availability between the nodes, which is specified by,

$$A(a,b) = \frac{1}{2}\left(\frac{R_{a\leftarrow b}}{S_b} + \frac{R_{b\leftarrow a}}{S_a}\right) \quad (1)$$

Here, $A(a,b)$ denotes the availability of links between a and b. $R_{a\leftarrow b}$ is the received advertisement from b at a, S_b is the total advertisement sent by b during the experiment.

From this result, we can see that the availability of links is apparently heterogeneous. Some links seem tightly connected but others are lightly connected. Fig. 3 is the distribution of the availability per link. Only 35% of the links were tightly (more than 90%) connected, and others were disruptive. Actually, 90% availability is not enough if the two nodes need to make session-based communication (e.g., TCP). Furthermore, if the hop count between two nodes increases, the packet loss ratio increases suddenly. This fact indicates that the principle of best-effort and end-to-end reliability cannot be applied to such networks.

[2]GW-USMicroN, Planex Communications Inc.

C. Contact Time and Inter Contact Time

We also looked into the detail of each link, and analyzed the distributions of contact time and inter contact time. Contact time, in this paper, is the duration that a node received advertisements from the other node without losses. If it received five succeeding advertisements but not the sixth advertisement, the contact time is 50 seconds. Inter contact time is the interval between the received advertisements. For example, if it could not receive three succeeding advertisements before receiving the fourth advertisement, the inter contact time is 40 seconds.

Fig. 4 and Fig. 5 shows the distribution of contact time and inter contact time for the links at 1 ↔ 2, 30 ↔ 31, and 30 ↔ 32. They respectively had 91%, 55% and 3% availability.

Links become disconnected at 23%(1↔2), 76%(30↔31) and 100%(30↔32) in 60 seconds. They reconnected again at 100%(1↔2), 98.9%(30↔31) and 59.6%(30↔32) in the next 60 seconds after disconnected.

D. Summary

From these results, we concluded that (1) the connectivity of links changed very frequently even nodes was static, that (2) more than half of them were such links and that (3) such links often have longer distance.

The shortest path, which gives the smallest hop count to the destination, would not be the best path regarding to delivery latency. In order to get the smallest hop count, each hop must reach long distance. However the links of longer distance frequently become disconnected. Instead, shorter links are available. Propagating messages on shorter links sometimes work faster. The results of our evaluation (Section V) clearly have shown this.

This experiment was made with small advertisement traffic at 10[sec] sampling rate. Burst traffic may cause larger packet losses.

IV. Potential-Based Entropy Adaptive Routing

In order to implement hop-by-hop reliable communication and parallel message propagation, we use potential-based entropy adaptive routing (PEAR) in this paper. PEAR inherits the concept of DTNs such as store-carry-and-forward and multipath message propagation. Actually, we have already proposed the definition of PEAR model in [18], but in the context of mobile nodes. Thus, in this paper, we focus on the description of behavior at static cases.

A. Terminology

Let N be a set of network nodes. We denote the neighbor nodes of node $n \in N$ by $nbr(n)$. Here, we define $nbr(n)$ contains n itself: i.e., $n \in nbr(n)$. In the following discussion, we setup a model that links are bi-directional. Thus, $\forall k, n\,(k \in nbr(n) \rightarrow n \in nbr(k))$.

We define potential of $n \in N$ for each destination $d \in N$ at time t by $V^d(n,t)$. $V^d(n,t)$ gives a scalar value, which tells a heuristic proximity to d from n. For example, if $V^d(n_1,t_0)$ is smaller than $V^d(n_2,t_0)$, n_1 is probably closer to the destination d. PEAR develops potential values dynamically according to the node contacts (section IV.C) and uses them for message forwarding decision (section IV.B).

B. Forwarding Scheme

We define two forwarding schemes for PEAR: best candidate selection (BCS) and multiple candidate selection (MCS). BCS chooses the most possible nodes and MCS chooses the better nodes among the contacted neighbors for the nexthop. More formally, we define them as,

Best Candidate Selection (BCS):

$$F^d_{max}(n,t) = \max_{nbr(n)} \{V^d(n,t) - V^d(k,t)\} \quad (2)$$

$$nexthop^d_{BCS}(n,t) = \{k | k \in nbr(n) \wedge F^d_{max}(n,t) = V^d(n,t) - V^d(k,t) > \alpha\} \quad (3)$$

Multiple Candidate Selection (MCS):

$$nexthop^d_{MCS}(n,t) = \{k | k \in nbr(n) \wedge V^d(n,t) - V^d(k,t) > \beta\} \quad (4)$$

Here, α and β are positive constants that give threashold of forwarding.

PEAR implements hop-by-hop reliability scheme, and takes copy-based approach in transferring messages. This makes parallel delivery and achieves faster message propagation.

After the nexthop candidates are determined, the node asks whether the nexthop already has the sending-message or not. If the nexthop has no knowledge about the message, the node forwards the message to it. If the nexthop already has the message, the node does not forward. Actually, the node makes copies of the message, instead of just forwarding. Traditional ad hoc network removed the message after the forwarding process. However, PEAR copies it to the nexthop and does not remove at the forwarding source. This forwarding strategy, which is widely acknowledged in DTN research community[27][22][13], certainly improves delivery performance. This scheme allows to make branch paths from the intermediate nodes and to propagate them in parallel. Even if a certain path became wrong, it finds another path and delivers the messages (see Section IV.E).

The forwarding scheme of PEAR also implements message deletion mechanism for those remained in the network. For more detail, see our previous paper[18].

C. Potential Field Construction

PEAR autonomously develops potential values, and uses them at message forwarding decision. Here, we provide the algorithm for potential-field construction.

PEAR nodes periodically exchange their potential vector (i.e., $\{V^d(n,t) | d \in N\}$) among their neighbors by radio-range multicast. And, each node computes potential values in the following rule.

$$V^d(n,t+1) = V^d(n,t) + D \min_{k \in nbr(n)} \{V^d(k,t) - V^d(n,t)\} + \rho \quad (5)$$

$$V^d(d,t) = 0 \quad (6)$$

$$0 < \rho < D \quad , \quad 0 < D < 1 \quad (7)$$

The potential of destination is always tied to 0 (Eqn. 6). Other potential values dynamically change depending on node-contact patterns. A potential normally grows by ρ at every timeslot, but decreases when the node has encountered a node of smaller potential value (Eqn. 5). D is a diffusion parameter. If it becomes larger, potential values decrease faster when it has encountered lower nodes, and dissemination of low-potential information becomes faster.

At the very early stage of potential-field construction, a node does not know all the nodes in the network over intermittent connectivity. In this phase, the node does not make any potential computation for such unknown destinations. However, the potential values associated to such destinations disseminate over the intermittent connectivity. First, a node opportunistically encounters another node and exchanges its

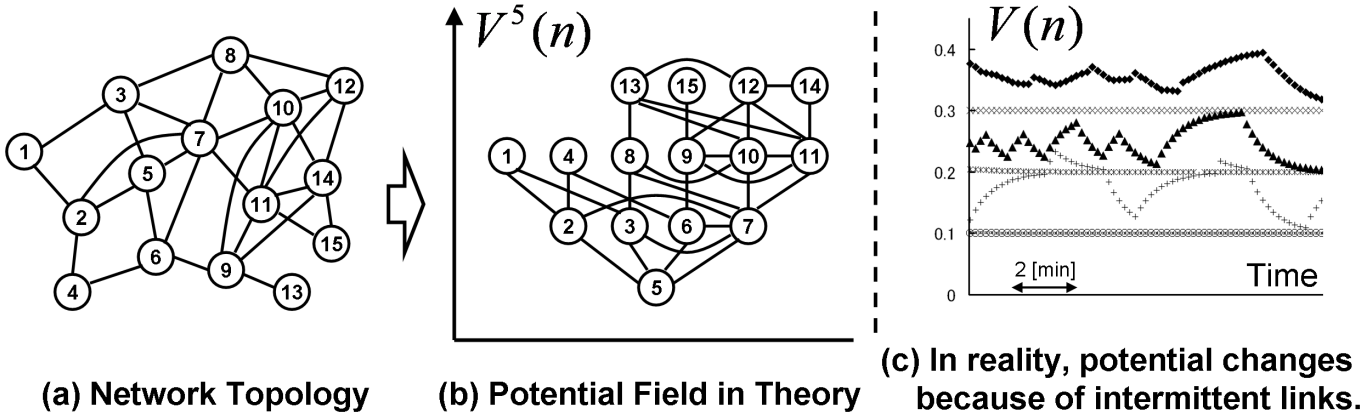

Fig. 6. Potential field construction (in theory and in reality); potential-field converges into (b) if the given network topology (a) does not change. However, links are intermittent and the topology dynamically changes. Potential-field follows this change as (c). (c) is a real trace on UTMesh.

potential vector with each other. Then, it finds that some destinations are not in the local potential vector. So, it adds to the vector and starts computation for the new destinations. The initial potential should be the first potential received. In this way, every node autonomously learns what nodes exists in the network even they are intermittently connected.

D. Potential Field in Stable Scenarios

If the network is stable and connected, a potential-field converges into the same pattern that distance-vector protocols develop; it increases linearly hop-by-hop from the destination. More formally, if the network is stable ($\Leftrightarrow \{nbr(n)|n \in N\}$ are static) and connected, PEAR gets,

$$\forall n \in N, \lim_{t \to +\infty} V^d(n,t) = \frac{\rho}{D} h(d,n) \tag{8}$$

Here, $h(d,n)$ is the minimum hop count from node d to n. Thus, it has an aspect of distance-vector routing.

Fig. 6 (b) shows the developed potential field for destination node n_5 by PEAR. In this example, the potential value increments as the hop count increases for n_5. Theoretically, it converges as Eqn. 8 presents, however in reality, because links are intermittently-connected as we have noted, the potential values change all the time as Fig.6 (c). Here, (c) is the real trace obtained at UTMesh.

E. Parallel Delivery in ICMeN

The same messages propagate in parallel in ICMeN whether it is BCS or MCS. In BCS case, each node chooses only one nexthop at one time. However, as we mentioned, because potential-field changes according to the status of links, the best nexthop candidate also changes in the next time slot. Thus, the node also copies messages to the new nexthop candidate. In this way, it makes multiple paths in message propagation. In MCS case, each node chooses several nexthops at one time, which itself makes multiple path without waiting the change of potential-field.

The major difference of BCS and MCS is the redundancy level in message propagation. MCS apparently creates larger number of delivery paths, indicating that small number of link failure does not cause fatal delay.

Choosing the best single path from the source to the destination is very difficult or impossible in ICMeN, because (1) the best path soon becomes obsolete or unavailable during the delivery of messages and (2) each message has its own best path. PEAR enables to choose the best path for each message by propagating itself in parallel.

V. Evaluation

We carried out experiments for two deployment cases (campus case and building case) and evaluated the features such as delivery latency and buffer usages. We compared the performance of BCS and MCS in both deployment configurations. In this section, we describe the experiment settings, the profile of the experimental networks and the results of delivery latency, message delivery patterns and buffer usages.

A. Experiment Setting

Fig. 7 shows the deployment configuration in Hongo campus and Eng. Bldg. 2 in the University of Tokyo(UT). We used 51 nodes (1, ..., 50, 99) in the experiments. First, all the nodes were gathered at Esaki laboratory at the 10th floor in Eng. Bldg. 2. We powered on between 5 and 10 nodes at the same time and shipped to the specified location. We repeated this until all the nodes were deployed. The wireless interfaces were deployed about 10cm high from the ground (outside) or from the floor (inside). We here configured node 1 to send one 100-byte message to node 99 at every 5 second. In this setting, the ICMeN delivers the messages from 1 to 99 by PEAR routing algorithm.

Before the deployment, we installed two programs into these nodes – the one is for BCS operation and the other is for MCS, and the running mode was automatically changed at specified times. In the campus case, nodes were operated by BCS for the first 3.5 hours, then swapped by MCS and operated for

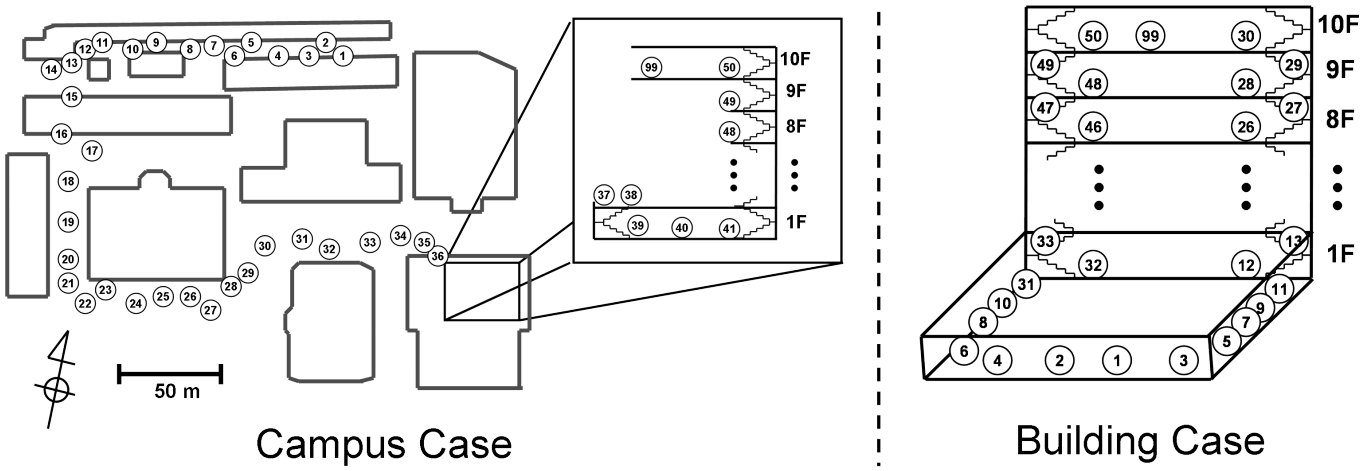

Fig. 7. Deployment configuration; we deployed 51 nodes in Hongo campus (campus case) and Eng. Bldg. 2 (building case) in the University of Tokyo.

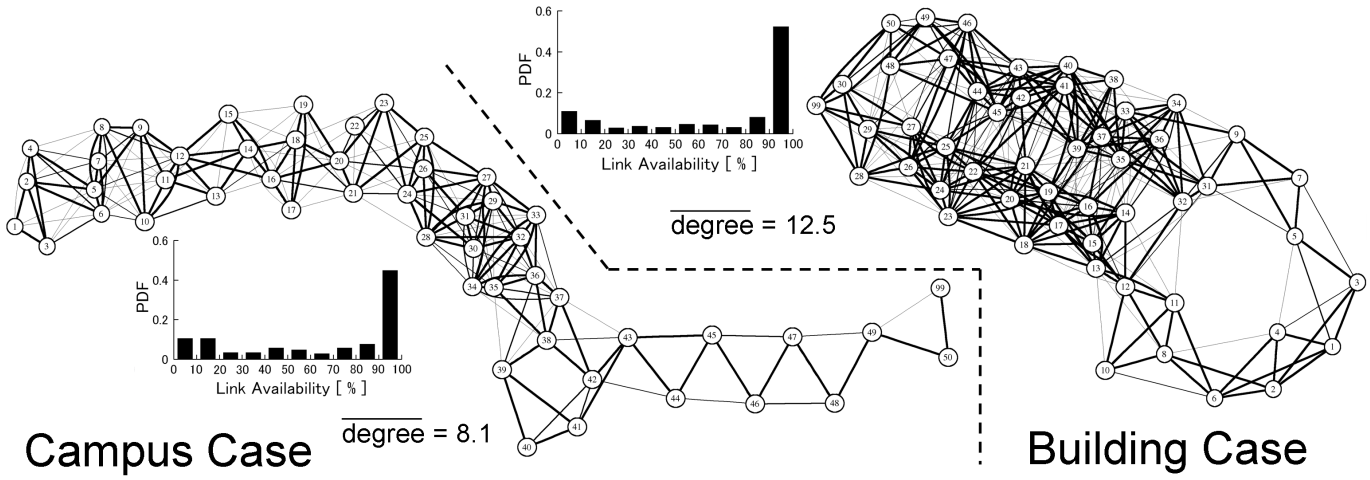

Fig. 8. Topology, link availability and average degree of the deployed networks. Bolder link has larger availability. The distribution of the link availability were almost the same.

the last 3.5 hours. In the building case, the first 6 hours were operated by BCS and the last 6 hours were by MCS. Here, we setup 5 minute break between the two modes.

Finally, we collected the deployed nodes into the laboratory, and retrieved the working logs for analysis.

All the wireless interfaces were operated in ad hoc mode of 802.11b at the same frequency: 2.412GHz (channel 1). The parameters of PEAR were as follows.

- POTENTIAL_TTL: 30[sec]
- MESSAGE_TTL: 3600[sec]
- DISSEMINATION_TTL: 300[sec]
- D: 0.2
- ρ: 0.02
- α, β: 0

We implemented PEAR to achieve small tasks at 10-second step. For example, it exchanged potential values in every 10 second; it generated the next potential-field every 10 second. It also tried re-transmissions for stored messages at every 10 second. Thus, when it failed forwarding messages, the messages must wait there for the next re-transmission, which caused 10 second delay.

B. Features of the Deployed Networks

Fig. 8 is the summarized topology and the distribution of link availability of the deployed networks. The boldness of links are provided by Eqn. 1. As it becomes bolder, the availability increases. The distribution of the availability was almost the same between the two configurations. About one half of the links were tightly (more than 90%) connected. The average degree (the number of links at an average node) was 8.1 (campus case) and 12.5 (building case).

C. Experiment Results

Fig. 9 provides the summary of the evaluation results. The results indicate that hop-by-hop reliability scheme achieves scalable message propagation (e.g., 23 hops), that message propagation speed increases as the redundancy-level increases.

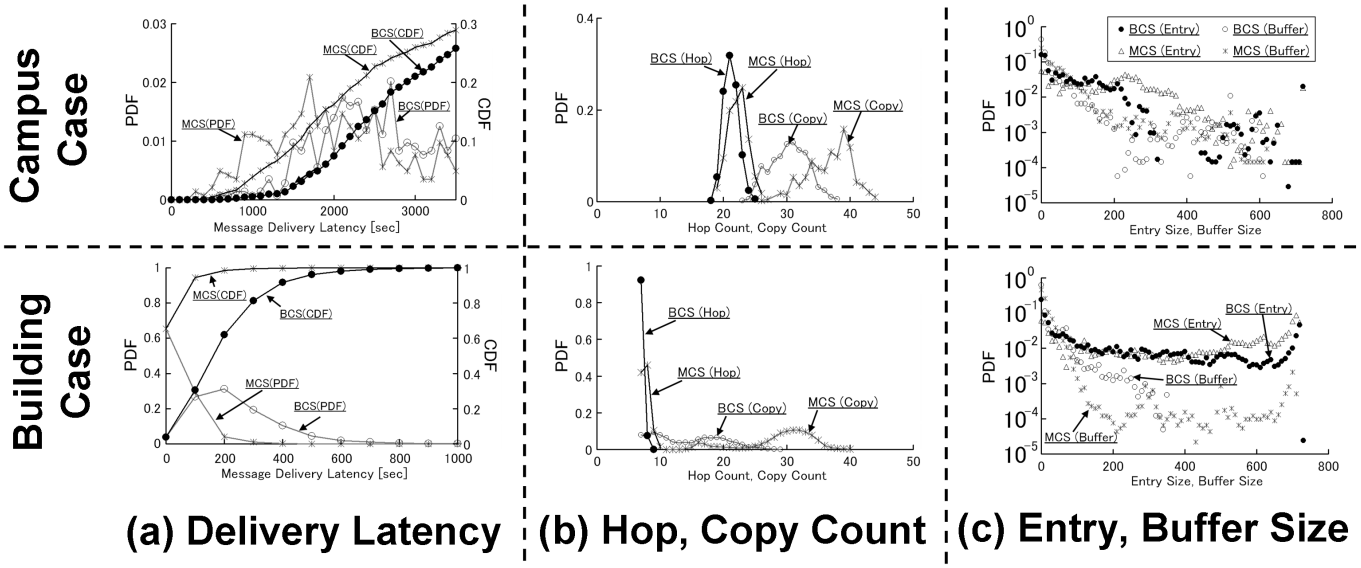

Fig. 9. The distributions of (a) delivery latency, (b) hop and copy count, (c) entry and buffer size for campus case and building case.

- **Campus Case:** PEAR achieved 26% (BCS) and 29% (MCS) delivery in the given message life time: i.e., 3600[sec]. The average hop count from 1 to 99 was 21.2 (BCS) and 22.3 (MCS). Messages were copied 30.4 times (BCS) and 36.9 times (MCS) in average. 96 entries (BCS) and 194 entries (MCS) were used at average nodes for replica management and message deletion control. 36.4 buffers (BCS) and 51.5 buffers (MCS) were occupied by messages at average nodes.
- **Building Case:** PEAR achieved 100% delivery probability for both BCS and MCS. The average delivery latency was 238[sec] (BCS) and 46.6[sec] (MCS). The average hop count from 1 to 99 was 7.08 (BCS) and 7.72 (MCS). Messages were copied 14.5 times (BCS) and 27.7 times (MCS) in average. 197 entries (BCS) and 408 entries (MCS) were used at average nodes. 17.3 buffers (BCS) and 15.4 buffers (MCS) were occupied by messages at average nodes.

Delivery probability is the ratio of the arrived messages during their life time to the sent messages. Average delivery latency is the average time of message travel from the source to the destination. Because the delivery latency is given for each message, we cannot calculate the average if any messages disappear before reaching the destination. Thus, we could present the average delivery latency only for the building case, which has achieved 100% delivery.

Average hop count is the average of message hop counts from the source to the destination. Average copy count is the average of copies for each message made in the network during delivery.

Entry is used to maintain the state of the message. Thus, when a new message arrives at node n, the message consumes one entry at n. The entry remains there until the message expires. Average entry usage shows how many entries are used in the average node, and average buffer usage shows how many message bodies are stored in the average node. Buffer usage is usually smaller than entry usage because PEAR deletion algorithm removes the message body but entry remains there until TTL expires to delete messages.

1) Delivery Latency: Fig. 9 (a) shows the distribution of message delivery latency.

In the campus case, the fastest message was delivered in 600[sec] (BCS) and in 300[sec] (MCS). 10% of the messages were delivered in 2200[sec] (BCS) and in 1600[sec] (MCS). 20% of the messages in 3000[sec] (BCS) and 2400[sec](MCS). They delivered 26% (BCS) and 29% (MCS) of the messages during the given TTL.

In the building case, BCS delivered 4% of the messages in between 0[sec] – 100[sec], whereas MCS achieved 65% for the same latency. 99% of the messages were delivered in 700[sec] (BCS) and 300[sec] (MCS).

Apparently, MCS achieved faster message propagation than BCS did. Especially, in building case, the average performance of MCS was 5.1 times faster (5.1=238[sec]/46.6[sec]).

2) Message Delivery Pattern (Hop and Copy Count): Fig. 10 shows the typical message delivery patterns in each scenario (we picked them up from thousands of delivery pattern samples). In campus case, 21.2 hops and 30 copies were average for BCS, and 22.3 hops and 36 copies were for MCS. In building case, 7.08 hops and 14.5 copies were for BCS, and 7.72 hops and 27.7 copies were for MCS. Fig. 9 (b) shows the distributions of hop count and copy count. Because messages took different delivery paths, each message gave different hop count and copy count.

Hop-by-hop reliability certainly improved the scalability in hop count. (According to [25], MANET could make only several hops for packet propagation).

MCS made larger number of copies especially in building

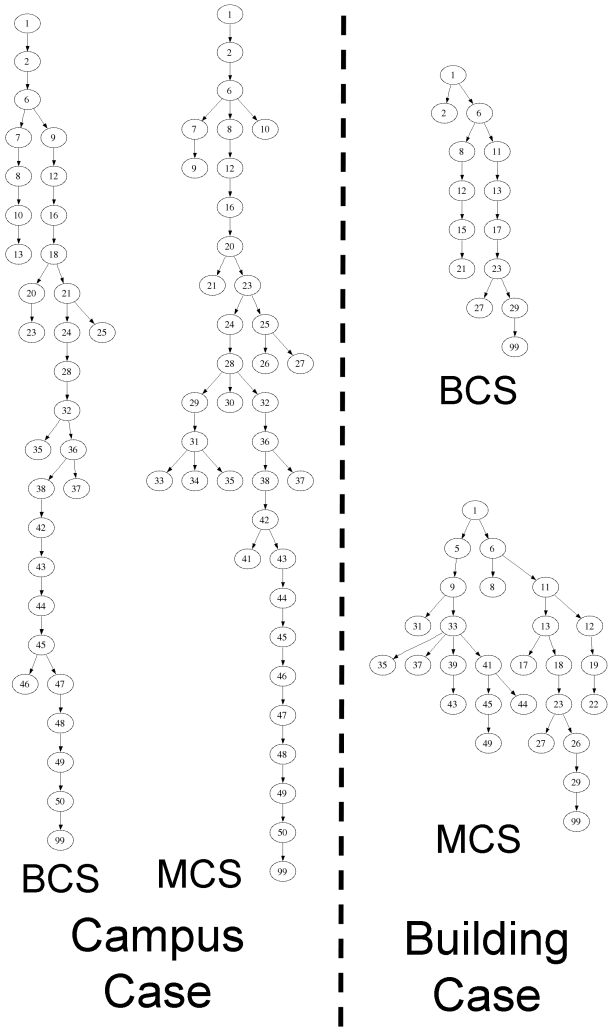

Fig. 10. Message delivery pattern examples from source 1 to destination 99. PEAR achieved scalable message propagation in hop count. MCS made larger number of replicas, found faster delivery path, and took bigger hop count than BCS.

case: i.e., about 1.9 times. Interestingly, MCS also took slightly larger hop counts than BCS did. This indicates that faster delivery path is not always the shortest path. Longer path (i.e., larger hop count) sometimes increases message propagation speed.

3) Entry and Buffer Usage: Fig. 9 (c) shows the distributions of entry and buffer usages. The maximum limit of the usages is 720[count] – given by the message generation traffic (0.2[count/sec]) and its life time (3600[sec]).

In both campus and building cases, most of the nodes had small entry and buffer usages at most of the time. In building case, buffer sizes (BCS and MCS) were much smaller than entry sizes. This indicates most of the message body was removed from the network after the delivery by PEAR deletion mechanism. However, in campus case, although buffer sizes were smaller than entry sizes in total, they were almost the same at large spectrum. This indicates that farer nodes from the destination removed fewer message bodies because they had to remain in the network until they reached the destination.

VI. Discussion

The reason why message delivery took large time (e.g., 100[sec], 1000[sec]) originates in PEAR implementation. PEAR was originally developed for delay tolerant networking, and thus it tried re-transmission in every 10 second. If the message could not be transferred in one transmission (this frequently happens probably because of [1]), it had to wait for the next trial. Because the hop count is large, the total delivery took plenty of time. We would be able to improve the re-transmission scheme for faster propagation, which is our future work.

The experiments have clearly shown that MCS, which gives larger redundancy, delivered messages to the destination especially when the network was densely configured (building case). Here, MCS took larger hop count than BCS did. This indicates that shorter distant links propagate messages faster though it must take larger number of hops.

The experiments were made on IEEE802.11b links (in ad hoc mode). Though the wireless interface itself supports IEEE802.11g and 11n, they do not work in ad hoc mode. The main contribution of this research, we consider, is the implementation report of hop-by-hop reliability and parallel propagation schemes for such unstable networks. Study on other wireless links (11g, 11n and 11s[7][28]) is open research items.

VII. Conclusion

In this paper, we have raised intermittently-connected mesh networks(ICMeN) on the basis that links of typical wireless mesh networks (i.e., composed of 802.11b ad hoc mode) are disruptive and unreliable. The connectivity of links, and the whole network topology, frequently changes, and this makes scalable message propagation in traditional communication schemes difficult.

We proposed to apply hop-by-hop reliable and parallel message propagation, which DTN researches have explored, to such ICMeN. We implemented them to UTMesh – 50 node scale wireless mesh testbed, and studied the delivery patterns.

On the evaluation result with UTMesh, we have confirmed (1) that hop-by-hop reliability achieves scalable message propagation (e.g., 23 hops), and (2) that message propagation speed increases as the redundancy-level increases. We have also observed that the smallest hop count path does not always achieve the fastest message delivery. This was probably because longer distant links were unstable and message paths over short distant links provided faster propagation.

References

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurement from an 802.11b mesh network. In *ACM SIGCOMM*, 2004.

[2] A. Balasubramanian, B. N. Levine, and A. Venkataramani. DTN routing as a resource allocation problem. In *ACM SIGCOMM*, 2007.

[3] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss. Delay-tolerant networking: an approach to interplanetary internet. *IEEE Communications Magazine*, 41(6):128–136, jun 2003.

[4] T. Clausen and P. Jacquet. RFC3626: optimized link state routing protocol (OLSR), oct 2003.

[5] K. Fall. A delay-tolerant network architecture for challenged internets. In *ACM SIGCOMM*, 2003.

[6] A. Franklin and C. Murthy. Node placement algorithm for deployment of two-tier wireless mesh networks. In *IEEE Globecom*, 2007.

[7] R. G. Garroppo, S. Giordano, D. Iacono, and L. Tavanti. On the development of a IEEE 802.11s mesh point prototype. In *ACM TridentCom*, 2008.

[8] J. Ghosh, H. Q. Ngo, and C. Qiao. Mobility profile based routing within intermittently connected mobile ad hoc networks (ICMAN). In *ACM IWCMC*, 2006.

[9] S. Heimlicher and B. Plattner. Reliable transport in multihop wireless mesh networks. In *Guide to wireless mesh networks*, pages 231–254. Springer, 2009.

[10] A. Jindai and K. Psounis. Fundamental mobility properties for realistic performance analysis of intermittently connected mobile networks. In *IEEE Percom*, 2007.

[11] A. Kinalis and S. Nikoletseas. Adaptive redundancy for data propagation exploiting dynamic sensory mobility. In *ACM MSWiM*, 2008.

[12] F. Laurent and G.-C. Felipe. Using delay tolerant networks for car2car communications. In *IEEE Inductrial Electronics*, 2007.

[13] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. *LNCS*, 3126:239–254, sep 2004.

[14] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordstrom, and C. Tschudim. A large-scale testbed for reproducible ad hoc protocol evaluations. In *IEEE WCNC*, 2002.

[15] S. Mao, S. Lin, S. S. Panwar, Y. Wang, and E. Celebi. Video transport over ad hoc networks: multistream coding with multipath transport. *IEEE Journal on Selected Areas in Communications*, 21, dec 2003.

[16] S. Mueller, R. P. Tsang, and D. Ghosal. Multipath routing in mobile ad hoc networks: Issues and challenges. *Performance Tools and Applications to Networked Systems*, pages 209–234, 2004.

[17] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. In *IEEE WoWMoM*, 2005.

[18] H. Ochiai and H. Esaki. Mobility entropy and message routing in community-structured delay tolerant networks. In *ACM AINTEC*, pages 93–102, 2008.

[19] C. Perkins, E. Belding-Royer, and S. Das. RFC3561: ad hoc on-demand distance vector (AODV) routing, jul 2003.

[20] J. Robinson, M. Singh, R. Swaminathan, and E. Knightly. Deploying mesh nodes under non-uniform propagation. In *IEEE INFOCOM*, 2010.

[21] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Communications*, 7, oct 2000.

[22] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Efficient routing in intermittently connected mobile networks: the multiple-copy case. *IEEE/ACM Transactions on Networking*, 16(1):77–90, feb 2008.

[23] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Efficient routing in intermittently connected mobile networks: the single-copy case. *IEEE/ACM Transactions on Networking*, 16(1):63–76, feb 2008.

[24] C. Tschudin, P. Gunningberg, H. Lundgren, and E. Nordstrom. Lessons from experimental manet research. *Elsevier Ad Hoc Networks*, pages 221–233, 2005.

[25] C. Tschudin and E. Osipov. Estimating the ad hoc horizon for TCP over IEEE 802.11 networks. In *Med-Hoc-Net*, 2004.

[26] A. Tsirigos and Z. J. Haas. Multipath routing in the presence of frequent topology changes. *IEEE Communications Magazine*, 39:132–138, nov 2001.

[27] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical report, Duke University, 2000.

[28] T. Vanhatupa, M. Hannikainen, and T. D. Hamalainen. Performance model for IEEE 802.11s wireless mesh network deployment design. *Journal of Parallel and Distributed Computing*, 68:291–305, mar 2008.

[29] K. Yoshida and H. Esaki. Energy saving with ICT: Green university of tokyo project. In *EcoDesign*, 2009.