Improving Opportunistic Data Dissemination via Known Vector[*]

Jyh-How Huang[1], Ying-Yu Chen[2], Yi-Chao Chen[2], Shivakant Mishra[3], and Ling-Jyh Chen[2]

[1] Department of Electronic Engineering, National Taiwan University
[2] Institute of Information Science, Academia Sinica
[3] Department of Computer Science, University of Colorado Boulder

Abstract

An opportunistic network is a network where each node only has connectiviy with other nodes opportunistically. To disseminate data in this kind of network is a challenging task and had drawn lots of research effort. When two nodes have connectivity with each other, the data bundles stored in their memory will be either replicated or forwarded to the other node in the hope that the data eventually reaches the destination node. The protocols that has only one copy of each data bundle is catagorized as forwarding protocol while the ones with multiple copies called replicating protocols. In opportunistic network, the replicating protocols are preferred over forwarding protocols. Yet, a big overhead is to be solved for replicating routing protocols, i.e. the exchanging of metadata. To avoid sending data bundles the other party already has, replicating protocols usually exchange metadata at the beginning of the connecting period. The naive method, sending the data bundles index summary in the memory, seems reasonable. But over time, this metadata exchange will become a big overhead. This paper propose a new scheme of indexing data bundles in the memory and can also reduces the metadata exchanged by large extent.

1. Introduction

For two nodes to establish a temperarily connected link in an opportunistic network, there are usually three phasese, namely probing phase, metadata exchanging phase and data exchanging phase. Probing phase is for two nodes to discover each other, this is usually done by periodically beaconing and listening. Metadata exchanging phase is for two nodes to exchange the information about the data bundles they have, the direction they are heading to, or how much space they have in the memory. A node applies the metadata it received to the utility function for some specific goal, e.g. maximum delivery rate, minimum average delay, and decides which data bundles it wants the other node to mule for. The third phase of establishing a link is the data exchanging phase. In this phase either one node sends data to the other one or they exchange data bundles in turn. Due to the natural charactor of opportunistic network, i.e. unstable wireless radio links, reliable transmission and how to recover from a temperarily broken link is the main issue in this phase.

In replicating protocols, on receiving metadata, a node can decide which data bundles should be replicated and sent to the other node. That is to say, the time duration and energy spent of data exchanging phase is largely decided by the metadata. With good metadata, only necessary data bundles will be repilcated to the other node and thus the energy used to send/receive redundent data bundles is saved. As important as it is, metadata is rarely discussed. In this paper, we propose a new method to index metadata, called known vector. Through known vector, we can save largely on the overhead of sending naive metadata repeatly. Proposed to solve the problems in the realworld, our protocol uses very little overhead and outperforms the traditional indexing method by a large scale.

In section 2 we will discuss other routing protocols for opportunistic networks. In section 3 we describe in details the known vector protocol. Next section 4 we will show the evaluation of the result through the simulation we did in *The ONE*. In section 5 we conclude the paper.

2. Background

Replication is the most popular design choice for opportunistic routing schemes. For instance, the *Epidemic Routing* scheme [14] sends identical copies of a message simultaneously over multiple paths to mitigate the effects of a single path failure; thus, it increases the possibility of successful message delivery. However, flooding a network with

[*]This paper is based on research supported by the National Science Council of Taiwan under Grant No. NSC 97-2628-E-001-007-MY3 and NSC 97-2631-S-003-003.

duplicate data tends to be very costly in terms of traffic overhead and energy consumption.

To address the problem of excess traffic overhead caused by flooding, Harras et al. proposed a *Controlled Flooding* scheme to reduce the flooding cost while maintaining reliable message delivery [8]. In this scheme, flooding is controlled by three parameters, namely, *willingness probability*, *Time-to-Live*, and *Kill Time*. Additionally, once a message has been delivered to the receiver successfully, a *Passive Cure* is generated to "heal" the nodes in the network that have been "infected" by the message. Therefore, by removing the excess traffic overhead problem, while providing reliable data delivery, controlled flooding can substantially reduce the network overhead.

Node mobility also impacts on the effectiveness of opportunistic routing schemes. Previous studies have shown that if the network mobility departs from the well-known random way-point mobility model (e.g., the Pursue Mobility Model [4] or the Reference Point Group Mobility Model [9]), the overhead carried by epidemic- and/or flooding-based routing schemes can be further reduced by considering node mobility. For instance, the *Probabilistic Routing* scheme [13] calculates the *delivery predictability* from a node to a particular destination node based on the observed contact history, and forwards a message to its neighboring node if and only if that neighboring node has a higher delivery predictability value. The scheme was revised by Leguay *et al.* [10] by taking the *mobility pattern* into account, i.e., a message is forwarded to a neighbor node if and only if that node has a mobility pattern more similar to the destination node. [10, 11] show that the revised *mobility pattern* scheme is more effective than previous schemes.

Another class of opportunistic network routing schemes is based on encoding techniques, which transform a message into a different format prior to transmission. For instance, an integration of *network coding* and epidemic routing techniques has been proposed to reduce the required number of transmissions in a network [16], and [15] proposes combining *erasure coding* and the simple replication-based routing method to improve data delivery for the *worst delay performance* cases in opportunistic networks.

Following the concept of erasure coding-based data forwarding [15], an Estimation based Erasure-Coding routing scheme (EBEC) has been proposed to adapt the delivery of erasure coded blocks using the Average Contact Frequency (ACF) estimate [12]. Moreover, [6] proposes a hybrid scheme, called HEC, that combines the strength of erasure coding and the advantages of *Aggressive Forwarding*. The HEC scheme has been further enhanced by employing the techniques of sequential forwarding (i.e., HEC-SF) [7], probabilistic forwarding (i.e., HEC-PF) [5], fully interleaving (i.e., HEC-FI) [7], and block-based interleaving (i.e., HEC-BI) [7].

3. Meta-Message Exchange Approaches

When two nodes encounter, they usually need to exchange some meta-messages to avoid sending duplicate messages that the other node already has. Many protocols apply a scheme which we call *Summary Vector*. A summary vector is the vector which comprises all identifiers of messages in the buffer of a node. An identifier of a message is unique for each message in the whole network. A replicating protocol usually requires summary vector exchanging to avoid sending/receiving redundant data bundles. But exchanging the summary vector repeatedly can cause a large overhead. Therefore, we propose a scheme, *Known Vector*, to alleviate the overhead introduced by the Summary Vector scheme. The Known Vector scheme can be considered as a pre-processor that can eliminate unnecessary identifiers of messages in a summary vector. Every protocol using the Summary Vector scheme can apply our Known Vector scheme to get better performance. For evaluation, we choose to apply the Known Vector scheme to the Epidemic Routing protocol, and compare it with the original Epidemic Routing protocol with the pure Summary Vector scheme. In section 3.1, we provide an overview of the Epidemic Routing protocol to briefly explain how the Summary Vector scheme works in this protocol. The proposed Known Vector scheme is explained in section 3.2.

3.1. Epidemic Routing

This section provides an overview of the Epidemic Routing protocol. In Epidemic Routing, the buffer of each node consists of messages originated by this node as well as messages relayed on behalf of other nodes. Each node also carries a summary vector which comprises all identifiers of the messages in its buffer. When two nodes encounter, they exchange the summary vector with each other. After receiving the summary vector from the encountering node, a node compares the received summary vector to its own summary vector, and then requests the messages that are not in its buffer. In this paper, we call a request for messages as a request vector. We denote the buffer of node x as BUF_x. A summary vector and a request vector sent from node x to node y are denoted as $SV_{x,y}$ and $RV_{x,y}$ respectively. The following is an example of what two nodes do when they encounter. Suppose that, before encounter, node i has messages M_1, M_2, and M_3 ($BUF_i = \{M_1, M_2, M_3\}$), and node j has messages M_3, M_4, and M_5 ($BUF_j = \{M_3, M_4, M_5\}$). When they encounter, node i generates a summary vector $SV_{i,j} = \{M_1, M_2, M_3\}$ for node j, and node j generates a summary vector $SV_{j,i} = \{M_3, M_4, M_5\}$ for node i. After exchanging summary vectors, node i request messages M_4 and M_5 by computing a request vector $RV_{i,j} = SV_{j,i} - BUF_i$, and node j request messages M_1 and M_2 by computing a request vector $RV_{j,i} = SV_{i,j} - BUF_j$. Suppose they have sufficient time to transmit all requested messages. Both node i and node j have 5 messages (M_1 to M_5) after their encounter.

3.2. The Proposed Approach: Known Vector

While summary vector can be viewed as node based approach, known vector can be viewed as message based approach. In known vector, every message in a node keeps a vector which is a list of node IDs those already have this message this node know of. Note that this vector is not an accurate global network knowledge. The known vector can be regarded as metadata of a message, and is duplicated and sent with the message. The known vectors of replicates in different nodes of one message might not be the same due to different relaying paths. The copy of a message M_k resides in node x is denoted as $M_{k,x}$, and the known vector of this copy is denoted as $KV_{M_{k,x}}$. Initially a message M_k originated by node x doesn't have any record in its known vector $M_{k,x}$. i.e. , $KV_{M_{k,x}} = \{\}$. When two nodes encounter, they will take the following steps: when node i and node j encounter, the state of node i is $BUF_i = \{M_1, M_2\}$, $KV_{M_{1,i}} = \{m, j, n\}$, $KV_{M_{2,i}} = \{a, b\}$, and the state of node j is $BUF_j = \{M_3, M_1\}$, $KV_{M_{3,j}} = \{a\}$, $KV_{M_{1,j}} = \{m\}$.

1. Generate and exchange summary vectors:
   Node x generates $SV_{x,y}$ that contains message IDs of those whose known vector doesn't include y. The algorithm of generating a summary vector is shown in Algorithm 1.For the example, the summary vector $SV_{i,j}$ contains M_2 only, but $SV_{j,i} = \{M_3, M_1\}$.

2. Send a request:
   Node x generates $RV_{x,y} = SV_{y,x} - BUF_x$. e.g., $RV_{i,j} = \{M_3\}$, and $RV_{j,i} = \{M_2\}$.

3. Transmit messages requested:
   For every message M_k transmitted from node x to node y, node x firstly duplicates $M_{k,x}$ as $M'_{k,x}$ including $KV_{M_{k,x}}$. After transmitting $M'_{k,x}$ to node y, node x adds y into $KV_{M_{k,x}}$. After receiving $M'_{k,x}$, node y saves it in the buffer as $M_{k,y}$ with $KV_{M_{k,y}} = KV_{M'_{k,x}} \cup \{x\}$. For the example, after node i transmits $M'_{2,i}$ to node j, $KV_{M_{2,i}}$ becomes $\{a, b, j\}$, and node j saves $M'_{2,i}$ as $M_{2,j}$ with $KV_{M_{2,j}} = \{a, b, i\}$. Also, after node j transmits $M'_{3,j}$, $KV_{M_{3,j}}$ becomes $\{a, i\}$, and node i saves $M'_{3,j}$ as $M_{3,i}$ with $KV_{M_{3,i}} = \{a, j\}$.

4. Update known vectors:
   Node x can infer that node y already has messages in the set $SV_{x,y} - RV_{y,x}$ because node y is supposed to request every message in $SV_{x,y}$ unless it already has that message. Therefore, node x adds y into the known vector of every message in the set $SV_{x,y} - RV_{y,x}$. For example, node j can infer that node i already has M_1, and then $KV_{M_{1,j}}$ becomes $\{m, i\}$.

Algorithm 1 Summary vector generation of node x when it encounters node y in the Known Vector scheme

```
Require: $x \neq y$
Ensure: $SV_{x,y}$
  $SV_{x,y} \leftarrow \{\}$
  for all messages $M_{k,x}$ in the buffer of node $x$ do
    if $y$ not in $KV_{M_{k,x}}$ then
      $SV_{x,y} \leftarrow SV_{x,y} \cup \{y\}$
    end if
  end for
```

Table 2. The properties of the two network scenarios

Trace Name	ZebraNet	iMote
Device	N/A	iMote
Network Type	N/A	Bluetooth
Duration (days)	16	3
Devices participating	34	274
Number of contacts	31,693	28,217
Avg # Contacts/pair/day	3.53086	0.12574

Table 1 shows all situations for a message M_k which resides in node i when node i encounters node j. The second column shows whether the known vector of the copy of message M_k in node i (denoted as $K_{M_{k,i}}$) contains j. The third column shows whether node j also has message M_k (denoted as $M_{k,j}$), and if node j already has $M_{k,j}$, whether the known vector of $M_{k,j}$ contains i is shown in the fourth column. Explanations about situations are shown in the fifth column.

4. Evaluation

We now evaluate the performance, in terms of the delivery ratio and traffic overhead, of the Summary Vector (SV) scheme and the proposed Known Vector (KV) scheme using a Java-based simulator, called *The ONE* [2].

We evaluated two network scenarios based on realistic wireless network traces, namely, the ZebraNet [1] and iMote [1] traces, which are publicly available for research purposes and correspond to the opportunistic wildlife network and people network scenarios. Table 2 outlines the basic properties of the network scenarios.

In each simulation run, the source and the destination pair was randomly selected from all participating peers; and the source peer transmitted messages in the first 10% of the

Table 1. Situations for a message M_k which resides in node i when node i encounters node j

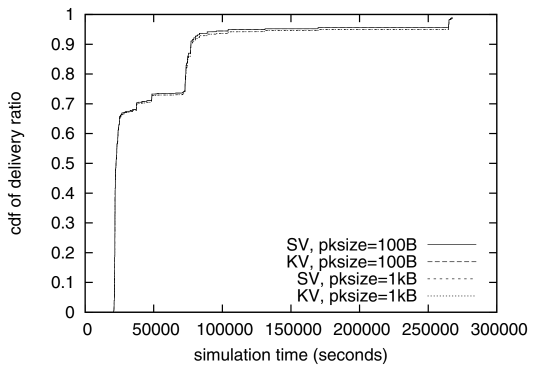
Situation	$KV_{M_{k,i}}$ contains j	Node j has $M_{k,j}$	$KV_{M_{k,j}}$ contains i	Explanation
1	No	No	No	1. M_k is put in $SV_{i,j}$. 2. Node j requests M_k after receiving $SV_{i,j}$. 3. After M_k is transmitted, j is added to $KV_{M_{k,i}}$, and i is added to $KV_{M_{k,j}}$.
2	No	Yes	No	1. M_k is put in both $SV_{i,j}$ and $SV_{j,i}$ (because i does not know j has M_k, and j does not know i has M_k, either). 2. No request for M_k from node i or node j. 3. Node i can infer node j has M_k and adds j to $KV_{M_{k,i}}$; node j can infer node i has M and adds i to $KV_{M_{k,j}}$.
3	No	Yes	Yes	1. M_k is put in $SV_{i,j}$ but not in $SV_{j,i}$ (because node j knows node i has M_k from $KV_{M_{k,j}}$). 2. No request for M_k from node j. 3. Node i can infer node j has M_k and adds j to $KV_{M_{k,i}}$.
4	Yes	No	No	1. $KV_{M_{k,i}}$ shows node j has M_k, but node j doesn't have M_k now. This may happen after the buffer of node j is full and M_k is selected to delete. 2. M_k is not put in $SV_{i,j}$, and node j will not request M_k from node i.
5	Yes	Yes	No	1. M_k is not in $SV_{i,j}$, but is in $SV_{j,i}$ (node j does not know node i has M_k because i is not in $KV_{M_{k,j}}$). 2. No request for M_k from node i. 3. Node j can infer node i has M_k and adds i to $KV_{M_{k,j}}$.
6	Yes	Yes	Yes	1. M_k is not put in $SV_{i,j}$ or $SV_{j,i}$. 2. Do nothing regarding M_k.

simulation time with a Poisson rate of 40 seconds/message in the iMote scenario and 200 seconds/message in the ZebraNet scenario. For simplicity, we assume that data transmission between peers is via ZigBee [3] with a fixed rate of 240 Kbps, and all messages are either 1K bytes or 100 bytes. We varied the buffer size in each set of evaluation, and compared the delivery performance of the two meta-message exchange schemes (i.e., the average percentage of messages received by the destination at the end of the simulation run). All the results presented here are based on the average performance of 200 simulation runs for each network configuration.

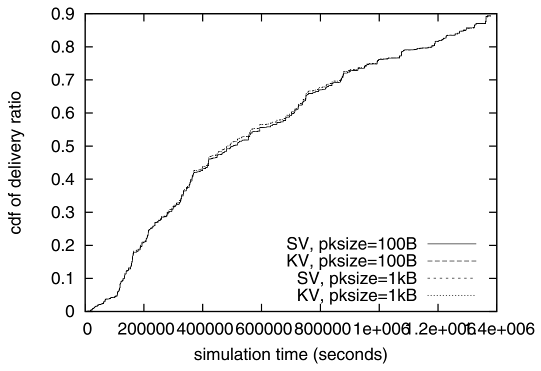
4.1. Evaluation I: The Infinite Buffer Case

In the first set of simulations, we evaluate the two meta-message exchange schemes with infinite buffer in the two network scenarios, when the packet size is 100 bytes and 1K bytes respectively. Figure 1 illustrates the delivery performance in cumulative distribution function (CDF) curves, and Figure 2 illustrates the traffic overhead (i.e., the data bytes of the control messages over that of the data messages) of the two schemes in the two network scenarios.

From the simulation results, we observe that, while the delivery performance of the two meta-message exchange schemes are comparable in the two network scenarios, the SV scheme consumes much more traffic overhead than the KV scheme. More precisely, as shown in Figure 2, the KV scheme is able to reduce about 16% and 31% of the traffic overhead, for both packet size settings, in the iMote and ZebraNet scenarios respectively. The results confirm that the proposed KV scheme is able to reduce traffic overhead

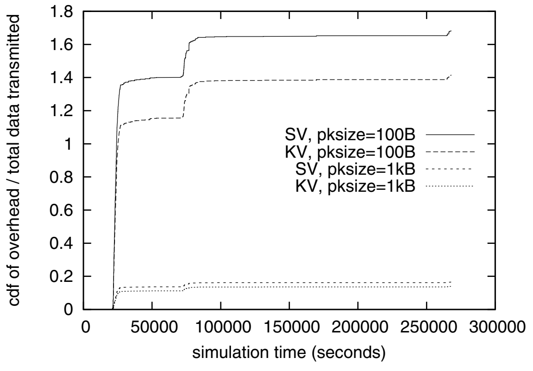


(a) iMote

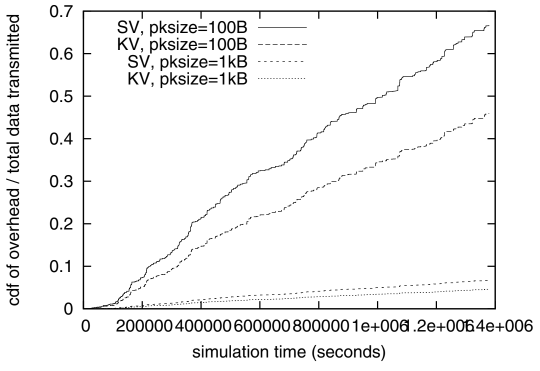


(b) ZebraNet

Figure 1. The delivery performance of the two meta-message exchange schemes with infinite buffer in the two network scenarios.

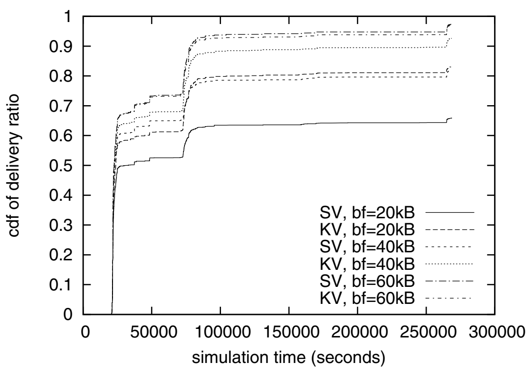


(a) iMote

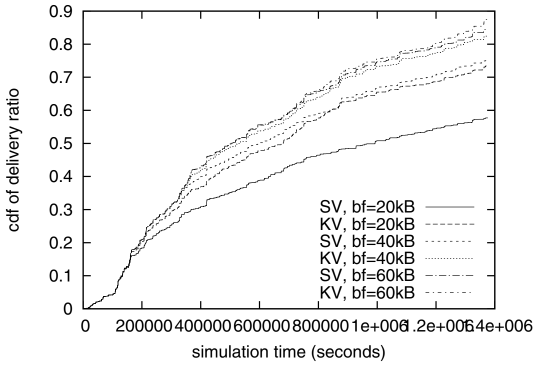


(b) ZebraNet

Figure 2. The traffic overhead of the two meta-message exchange schemes with infinite buffer in the two network scenarios.



(a) iMote



(b) ZebraNet

Figure 3. The delivery performance of the two meta-message exchange schemes with finite buffer in the two network scenarios when the packet size is 100 bytes.

for data dissemination in opportunistic networks, while preserving the delivery performance of the SV scheme.

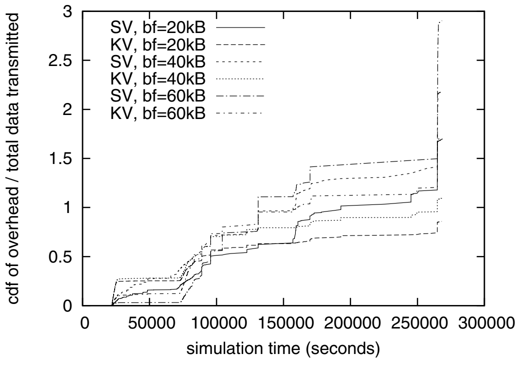
4.2. Evaluation II: The Finite Buffer Case

Next, we evaluate the two meta-message exchange schemes with finite buffer in the two network scenarios. Figure 3 shows the delivery performance of the two schemes with various buffer settings (i.e., 20k/40k/60k bytes) when the packet size is fixed at 100 bytes. It is clear that the KV scheme outperforms the SV schemes in all test cases, and the performance gain increases as the buffer size decreases. Specifically, the performance gain is about 17%, 10%, and 0% in the iMote scenario, and about 16%, 7%, and 3% in the ZebraNet scenario when the buffer size is 20k, 40k, and 60k bytes respectively. The results indicate that the KV scheme is superior to the SV scheme, especially when the end devices are buffer-constrained (e.g., sensors and handhelds).

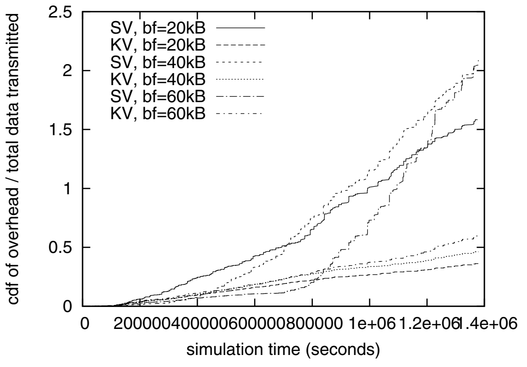
Figure 4 presents the traffic overhead (i.e., the data bytes of the control messages over that of the data messages) of the two schemes with various buffer settings (i.e., 20k/40k/60k bytes), when the packet size is fixed at 100 bytes, in the two network scenarios. We observe that the KV scheme is able to reduce traffic overhead when comparing with the SV scheme. More precisely, the KV scheme reduces about 49%, 49%, and 41% in the iMote scenario, and about 77%, 77%, and 71% in the ZebraNet scenario when the buffer size is 20k, 40k, and 60k bytes respectively. Again, the results show that the KV scheme is superior to the SV scheme when the buffer size is limited for each network participant.

5. Conclusion

In this paper, we study the meta-message exchange in opportunistic network routing protocols. We argue that the traffic overhead may increase substantially as the number of messages buffered on each node increases, and we propose a novel approach, called *Known Vector*, to resolve the problem. Using a comprehensive set of simulations, as well as realistic network mobility traces, we evaluate the



(a) iMote



(b) ZebraNet

Figure 4. The traffic overhead of the two meta-message exchange schemes with finite buffer in the two network scenarios when the packet size is 100 bytes.

proposed scheme with the current meta-message exchange scheme, called *Summary Vector*. The results show that the two schemes are comparable when the network buffer is infinite. When the network buffer is constrained, the proposed scheme is much superior to the Summary Vector, in terms of delivery performance and traffic overhead. More, the proposed scheme is simple and applicable to other opportunistic routing protocols. Work on the analysis of the proposed scheme is ongoing, and we hope to report the results in the near future.

References

[1] CRAWDAD. http://crawdad.cs.dartmouth.edu/.

[2] The opportunistic network environment simulator. http://www.netlab.tkk.fi/tutkimus/dtn/theone/.

[3] ZigBee alliance. http://www.zigbee.org/.

[4] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication and Mobile Computing Journal*, 2(5):483–502, 2002.

[5] L.-J. Chen, C.-L. Tseng, and C.-F. Chou. On using probabilistic forwarding to improve hec-based data forwarding for opportunistic networks. In *IFIP EUC*, 2007.

[6] L.-J. Chen, C.-H. Yu, T. Sun, Y.-C. Chen, and Hao-huaChu. A hybrid routing approach for opportunistic networks. In *ACM SIGCOMM Workshop on Challenged Networks*, 2006.

[7] L.-J. Chen, C.-H. Yu, C.-L. Tseng, H. hua Chu, and C.-F. Chou. A content-centric framework for effective data dissemination in opportunistic networks. *IEEE Journal of Selected Areas in Communications*, 26(5):761–772, June 2008.

[8] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer. Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding in Sparse Mobile Networks. In *IFIP Networking*, 2005.

[9] X. Hong, M. Gerla, R. Bagrodia, and G. Pei. A group mobility model for ad hoc wireless networks. In *ACM International Workshop on Modeling, Analysis and Simulation of Wirelessand Mobile Systems*, 1999.

[10] J. Leguay, T. Friedman, and V. Conan. Dtn routing in a mobility pattern space. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[11] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing for dtns. In *IEEE Infocom*, 2006.

[12] Y. Liao, K. Tan, Z. Zhang, and L. Gao. Estimation based erasure-coding routing in delay tolerant networks. In *IWCMC*, 2006.

[13] A. Lindgren and A. Doria. Probabilistic routing protocol for intermittently connected networks. Technical report, draft-lindgren-dtnrg-prophet-01.txt, IETF Internet draft, July 2005.

[14] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-2000-06, Duke University, 2000.

[15] Y. Wang, S. Jain, M. Martonosi, and K. Fall. Erasure coding based routing for opportunistic networks. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[16] J. Widmer and J.-Y. L. Boudec. Network coding for efficient communication in extreme networks. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.