PRoPHET+: An Adaptive PRoPHET-Based Routing Protocol for Opportunistic Network

Ting-Kai Huang
University of California, Riverside
Email: huangt@cs.ucr.edu

Chia-Keng Lee
University of California, Riverside
Email: clee071@ucr.edu

Ling-Jyh Chen
Institute of Information Science, Academia Sinica
Email: cclljj@iis.sinica.edu.tw

***Abstract*—We propose PRoPHET+, a routing scheme for opportunistic networks designed to maximize successful data delivery rate and minimize transmission delay. PRoPHET+ computes a deliverability value to determine the routing path for packets. Deliverability is calculated using a weighted function consisting of evaluations of nodes' buffer size, power, location, popularity, and the predictability value from PRoPHET. Even though the proposed PRoPHET+'s weights are chosen based on qualitative considerations, it is possible for PRoPHET+ to perform even more efficiently in various environments by shifting the weights accordingly. Our simulation illustrates that PRoPHET+ can perform better or equal to the routing protocol PRoPHET if logical choices for weights are used.**

I. Introduction

Wireless network infrastructures have been expanding at a rapid pace throughout the world. However, wireless networks may still not be available in areas such as poor regions, underwater sensors, or military operations. In order to provide networking support for situations where there are no directly connectivity paths, opportunistic network can be applied. Opportunistic network is a type of delay tolerant, intermittently connected network using an ad-hoc like structure. When a node wants to deliver data to another node but there does not exist a direct connection between them, packets can be forwarded to intermediate participating nodes which aid in delivering the packet from the source to the destination. Unlike a typical ad-hoc structure, however, opportunistic network assumes there is almost never a fully connected path between source to destination and the intermediate nodes may not encounter other nodes frequently or consistently. In some cases, intermediate nodes may have to buffer the packets received for a long time.

Due to the uncertainty of packet delivery success in opportunistic networks, numerous routing protocols were proposed to maximize packet delivery rate. One of the most well known routing protocol for opportunistic networks is a protocol called PRoPHET [1]. Since the chance of having a directly connected path from a source node to the destination node is rare or non-existent, identifying potential intermediate carriers for the packets to be transferred is essential. Forwarding data to intermediate carriers that rarely encounter the destination node will, in the worst case, fail to deliver the data. PRoPHET uses a predictability value, which is calculated using the history of encounters between nodes to evaluate the packet forwarding preference. While PRoPHET has shown decent results, there is still room for improvements. Due to the FIFO queuing nature of PRoPHET, packets may be dropped consistently when packets are forwarded to a few concentrated nodes. Packets may also be lost due to node failures or incomplete transmissions. Aside from packet loss, reducing transmission delay in networking is commonly desired. Even though opportunistic network is a type of delay-tolerant network, it could still benefit from shorter transmission delays. Transmission may be delayed when nodes are offline or when nodes have a significant amount of packets to deliver. In order to reduce packet loss and transmission delay during the described situations, we take buffer, power, bandwidth, location, and popularity into consideration.

We begin with related work followed by the presentation of our protocol. After the protocol details, the simulation environment is described, with the evaluation in the same section. Finally, we discuss certain issues, future work, and end with conclusion.

II. Related Work

Most routing protocols for intermittently connected networks can be classified into either Epidemic, "Message Ferrying", or Coding Based routing protocol.

In Epidemic routing [17], the sender node replicates packets and forwards them to all nodes that come in contact with it, therefore essentially flooding the network with identical data. PREP [12] is an example of epidemic routing protocol that prioritizes flooding based on buffer. In Harras [4], various methods were evaluated to control the flooding caused by epidemic routing. While Epidemic routing effectively minimizes the chance of data loss due to node failure situations, it also produces unnecessary buffer and power overhead.

In order to minimize overhead, various protocols were proposed using parameter measurements to forward packets. In"Message Ferrying" or utility-based routing, packets do not get replicated as nodes come in contact with one another. Instead, the sender node evaluates all contacting nodes with the chance of successful packet delivery using some defined function and "ferries" the packets to the node with the highest success rate. Since there will be at most one instance of every unique packet, overhead in terms of storage is reduced tremendously. Various different methods were proposed to evaluate the packet forwarding route. The most commonly used parameter is the history of encounters [1], [9], [11] and

physical location or mobility behavior of nodes [14], [7], [6]. There are also protocols that combine the two parameters, such as MV [10]. Spyropoulos [16] uses randomized forwarding to seek nearby nodes, and utilizes utility-based forwarding when focused on a set of nodes with high delivery success value.

Another broad category of routing protocol for intermittently connected networks is the use of Coding Based [18], [5], [19], [8]. In Coding Based routing protocol, packets are transformed into n blocks in such a way so that the packet can be recovered from a subset of the blocks. Therefore, as long as the destination receives enough subsets of the blocks, it can recover the original packet.

Our proposed protocol is a type of Message Ferrying routing protocol. Similar to our work, PR_CD [9] is also based on PRoPHET. Instead of using the frequency of contacts, however, PR_CD measures the contact duration of each contact in order to provide nodes with the maximum data transfer rate, therefore reducing transmission latency and overhead. Another difference is that PR_CD modifies the algorithm that PRoPHET uses, while our protocol is an extension of PRoPHET. MaxProp [2], another protocol, assumes that the nodes are aware of the schedule of meetings. As a result, MaxProp address the buffer issue by "prioritizing both the schedule of packets transmitted to other peers and the schedule of packets to be dropped" [2].

III. Weighted Function Routing

Before we begin describing the protocol, certain assumptions were made to make the problem more tractable. First, all nodes participating in the packet forwarding behaves well. We do not consider security issues, resource cheating issues, and integrity issues. Second, we assume that all packets transmitted are of the same size. Third, all devices are capable of performing basic processing capabilities. Fourth, we assume that in a network, nodes in the network are either all rechargeable or are all not rechargeable. Fifth, we assume that storage used for storing parameter values and idle power consumption are negligible. Lastly, all devices are able to determine their own buffer size, power consumption, remaining power.

In order to differentiate between different nodes, we define terms for nodes that perform different actions. Source node is the node that originate the packets, and destination node is the final destination node that the packets are meant to be delivered to. When describing a packet forward from one node to another, the terms sender node and receiver node are used respectively. Candidate nodes are nodes that are in contact with the sender node when packet forwarding is desired.

When a sender node wants to send data to a receiver node, it queries all candidate nodes with their deliverability value. The source node will send the packet to a candidate node if and only if the candidate node's deliverability value is above a certain threshold, which we define in this paper as a value better than the sender node's deliverability value. If there are multiple candidate nodes that meets the threshold restriction, the node with the highest deliverability value is picked as the receiver node.

The parameters in the weight function that determines deliverability consists of prophet's predictability value in addition to buffer, power, bandwidth, location, and popularity. To obtain a deliverability value, values of each parameter are normalized to a value between 0 and 1. The deliverability value is then a value from 0 to 1, which is obtained by summing each normalized parameter value multiplied by the weight factor defined for the parameter. The following sections describe in detail the reasonings behind selecting each property and methods to measure each property.

Buffer Parameter

In a FIFO queuing structure, when a packet is forwarded to a device with a full buffer, the first packet in the device is removed. When this occurs, the packet that is removed is lost. Data will also be lost if the device that had received packets from other sources also generated its own data and stored the data in the same buffer. In order to minimize packet loss due to buffer size, a function used to take buffer size into account is described.

Taking into consideration the data generated by nodes in the network, all nodes define a threshold B_{thresh}. B_{thresh} is computed as B_{total} - B_{self}, where B_{total} is the total buffer size of the node, and B_{self} is the amount of storage the node needs to contain self-generated data. Since the amount of data generated by a node may not be static, the node logs its data size for a certain amount of time, and determines an average data size. We specify an arbitrary amount of time x and an evaluation period of y. After every x amount of time passes, the node logs the capacity used for the data generated within that time frame. At the end of y periods of x, the node sums up the capacity log and divides it by y to get an average capacity used.

When a sender node queries the candidate node for probability value parameters, the candidate node sends the value of current remaining buffer size and B_{thresh} to sender node. The sender node then calculates V_B, which is the value of the buffer parameter, by dividing the remaining buffer size minus the packet size of the sender node over B_{thresh}. All negative values are set to 0. The calculation of V_B is done at sender node instead of candidate node to provide flexibility. While we assume all packet sizes from all devices are the same, by calculating V_B at sender side, it could potentially adapt to multiple size packets.

Power Parameter

Typically, there are two cases where nodes fail, either through malfunctioning of the device hardware or power outage. When nodes fail, the data that is stored in the buffer may be lost forever either due to irrecoverable storage parts or power recharge restrictions. While the malfunctioning parts may be unavoidable, data loss due to power outage may be reduced. While data loss may be reduced for devices that cannot be recharged, power consumption factor may also benefit for rechargeable devices. When rechargeable devices run out of power, this causes downtime for chances of delivering packets

to other devices. Therefore, by keeping a device up for as long as possible, the transmission delay may be shortened. As a result, two different functions are used to evaluate power for the two different cases.

In a network where all nodes are not rechargeable, we like to ensure not only if the node has enough power to receive a packet, but also if it will have enough remaining power to forward the packet. Furthermore, in order to increase the uptime of a node, the less power the candidate node has, the less favorable it is to send to the node. To calculate V_P, the power parameter's value of a given node, for non-rechargeable nodes, the function $\frac{(P_{remain}-P_{receive}-P_{send}-P_{threshold})}{P_{total}}$ is used, where P_{remain} is the remaining power, $P_{receive}$ is the amount of power required to receive a packet, P_{send} is the amount of power required to send a packet, and P_{total} is the total power of the device. $P_{threshold}$ is determined using the number of packets in the buffer multiplied by P_{send}. By setting a threshold where power is retained for packets in the buffer, the power required to forward most, if not all of the packets in the buffer before the node fails is ensured. This diminishes the loss of data due to power outage.

In the cases where recharge is possible, the use of weighting the value of power is to increase uptime of the node. While we do not consider rechargeable devices during our weighing determination and evaluation due to certain limitations, we include a possible simple function to calculate power values for rechargeable devices. The V_P for rechargeable devices can be calculated simply by $\frac{P_{remain}}{P_{total}}$. As a node decreases in power, the likeliness of the node suffering downtime increases.

Bandwidth Parameter

Depending on the networking scheme, nodes transferring data may or may not know if a packet is forwarded successfully. In cases where transmission completion status is unknown, bandwidth may become an issue in terms of data loss. In order to reduce the chance of unsuccessful packets due to short contact periods between nodes, using a faster bandwidth is desired. This is especially common in high mobility networks. The result of sending packets unsuccessfully affects power usage since the power used to perform the transfer is wasted. Bandwidth also can potentially delay packet delivery. This can be a major factor in networks where the encounter of nodes are rare. For example, assume that there's only one intermediate node that encounters the destination node frequently. If the sender node encounters the intermediate node, then it would want to forward all the packets that is needed to be delivered to the destination node to this candidate node. If the bandwidth is limited and contact time is short, only partial data can be transferred. The rest of the data will have to be delivered in the next encounter of the intermediate node, which delays the deliver time. In the case of networks with rare contact between nodes, the next encounter may be days or weeks.

In order to determine V_A, the bandwidth parameter's value of a given node, the sender node sends at its max transmission rate to all candidate nodes. Candidate nodes then calculate V_A using their own max transmission rate divided by sender's max transmission rate. If the output value is greater than one, then it is normalized to one since it can only support the max transmission rate of the sender.

Popularity Parameter

A possible cause for data loss is single point of failures. If numerous packets are forwarded to only a single node with a large buffer size due to the node being a node with good carrier properties, then when this single node fails and becomes unrecoverable, all data forwarded to this node is lost. Single point of failure is partially mitigated with the buffer and power parameters. By dispersing the data packets based on the amount of buffer size available, parts of the data may still be delivered when a receiver node fails since not all packets are located on a single node. When buffer size is not taken into account, if the receiver node fails, the entire data will be lost. The same concept applies to power. However, the issue of single point of failure could not be determined when there are nodes with large amount of buffer and power. For example, if one of the nodes in the network is a laptop containing 500GB of disk space and the typical size of data transferred are 1KB, such as a text message, then it will take a long time before this node becomes a less likely candidate. Furthermore, since V_B is a value from 0 to 1, the sender node does not know the amount of data the candidate node has. The more data there are in a buffer, the longer it is to transfer the data when the node becomes a sender node. When contact time is short, data forwarding may take multiple attempts, thus delaying the delivery of data. Therefore, a popularity parameter is considered to take into account single point of failure and delay due to significant amount of data in a queue.

Popularity parameter value, defined by V_O, is defined by $1-\frac{N_t}{M_t}$. N_t is the number of transmissions the node has performed in a certain amount of time. M_t is the maximum number of transmissions the node can perform in that same amount of time. A transmission is defined as a successful send or receive of a packet.

IV. Identifying Weight Values

In order to generate the deliverability value of candidate nodes, a weight for each parameter value needs to be determined. V_D, the deliverability value, is represented by the function of:

$$V_D = W_B(V_B) + W_P(V_P) + W_A(V_A) + W_O(V_O) + W_R(V_R)$$

V_R represents the predictability value from PRoPHET, and W_B, W_P, W_A, W_O, and W_R are weights for buffer, power, bandwidth, popularity, and predictability value respectively. We derived a general weighting factor that can perform relatively well under most general situations. While the weighting factors may not be optimal in specific cases, users who implement this protocol may shift the weights to accommodate specific environments.

In our speculation, we believe that most packets are lost due to buffer or power related issues. While opportunity is

TABLE I
THE PROPERTIES OF THE NETWORK SCENARIOS IN THIS STUDY

Trace Name	Haggle
Device	iMote
Network Type	Bluetooth
Duration(days)	4
Devices participating	274
Number of Contacts	28,250
Avg # Contacts/pair/day	0.27

an issue, as long as the node containing the packet remains functional, the packet should be delivered eventually. However, since in most cases, the chance of encountering nodes with high probability of encounter destination node may be rare in intermittently connected networks, carriers must be chosen carefully. As a result, we place most emphasis on the predictability value. The second most valuable parameters are buffer and power, since data loss and node load increases once buffer and power decreases. While popularity could potentially pose as an important feature, we consider it to be less of a factor in our weighted function since we consider general cases where popularity affects delay issues more than data loss. We speculate that in most cases, the network either contains all nodes with small amount of buffer size, such as wireless sensors, or nodes with large amount of buffer size that are unlikely to fail, such as laptops. In cases where the above presumptions are not true, popularity weight should be reevaluated. Lastly, we do not weigh bandwidth heavily since most state of the art protocol transmit some type of notification if an unsuccessful delivery occurs. Furthermore, we believe that most modern bandwidth connections should be capable of transmitting at a high rate. In cases where various different bandwidth rates exist, the bandwidth factor should be increased. Due to the nature of the paper, delay of delivery for a packet is less of a concern than packet loss, which is the main contribution of bandwidth. Following these guidelines, we picked initially the the following function:

$$V_D = 0.25(V_B) + 0.25(V_P) + 0.1(V_A) + 0.1(V_O) + 0.3(V_R)$$

We show in our simulation that this initial guess for weights already show promising improvements compared to solely using the predictability value from PRoPHET.

V. EVALUATION

In order to simulate real world situations [15], we used a trace called iMote [13]. IMote trace is a human mobility trace logged at the 2005 IEEE INFOCOM conference. The properties of the iMote trace is listed in table I. Each iMote device logs other Bluetooth devices that it encounters, including the time and duration of each encounter. To create the simulation, we used DTNSIM [3], a Java based, discrete event simulator for delay tolerant networks. 20 nodes were chosen randomly to generate data for the experiment. Data generation is simulated using a Poisson rate of one file per 900 seconds in the beginning 40% of the simulation period. Each file size is 10 MB in size. We picked 10 MB because it is about the size of a typical mp3 file encoded in 320kb/s. To reduce complexity, all packets of the same file are to be forwarded to the same node. A delivery is considered success if and only if the entire file is transmitted from the source node to the destination node.

In order to evaluate if the parameters are beneficial, we first test the parameter with PRoPHET. Since each parameter independently performs much worse than PRoPHET, we decided to run the tests in terms of parameter plus PRoPHET compared to PRoPHET. We picked a weight of 0.5 per parameter and 0.5 predictability value since we wanted a weight that does not bias toward either factor. Unless otherwise specified, delivery ratio is the number of files delivered to destination divided by the number of files total to be delivered to destinations. To identify data loss and delay respectively, we first evaluate each parameter versus the delivery ratio and the delays, and then an initial weighted function as a test to assess our hypothesis of PRoPHET+'s performance. Lastly, we ran our proposed weighted function. In both the weighted functions, we also look at the delay differences between PRoPHET+ and PRoPHET.

A. Delivery Ratio

In this section we observe the delivery ratio for each parameter in comparison with PRoPHET. Buffer, Power, Bandwidth, and Popularity were evaluated respectively.

1) Buffer Parameter Delivery Ratio: We evaluated buffer under two circumstances, one where all nodes contain a fixed buffer size, and another where the buffer sizes are distributed using a uniform distribution. The reason we looked at fixed and random cases is because there may be situations where all devices in a network have identical specs, and other cases where devices have different specs.

As we observe in Figure 1(a), PRoPHET + Buffer performs slightly better than PRoPHET under small buffer size, improves as buffer size increases, peaks between 1024MB and 2048MB, and performs similar to PRoPHET after 4096MB. This trend occurs possibly because that at small buffer sizes, there is not enough capacity to prevent packet loss due to the large number of packets versus the amount of storage available. As the buffer size approaches 2048MB, the nodes begin to have enough capacity to withhold the transmitted files. We see an increase in performance with PRoPHET + Buffer because data is spread throughout nodes, therefore reducing the packets dropped from FIFO queue. In PRoPHET, since all the data is sent to one best candidate node, data is continuously being removed when the buffer of the best candidate node becomes full. Lastly, after the buffer size becomes large enough for few nodes to accept all files without data loss, PRoPHET + Buffer loses its effectiveness since no packets are removed from queue due to buffer issue.

For randomized buffer size allocated to devices in the network, each value in x-axis represents the minimum buffer for a device in a uniform distribution, with the max buffer size being 10 times the value. We observe in Figure 1(b), PRoPHET + Buffer performs better than PRoPHET under small buffer size, but quickly loses its effectiveness near 256

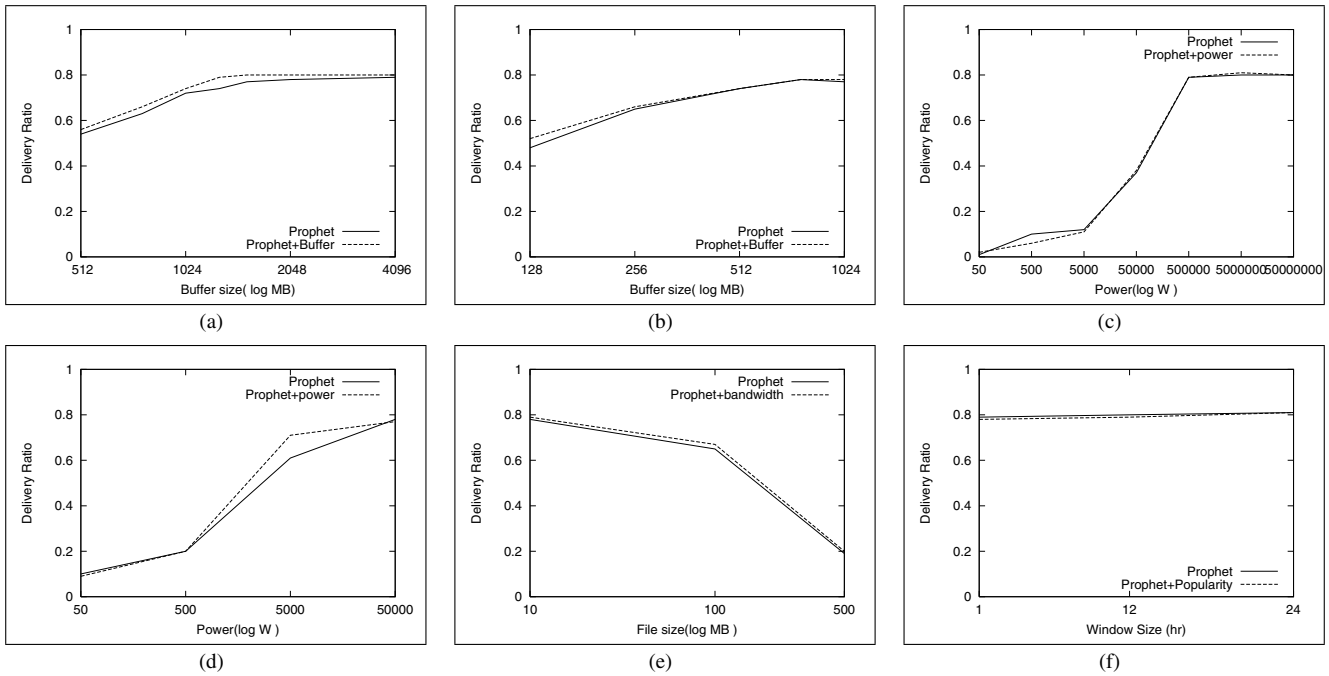

Fig. 1. (a) Buffer Delivery: Fixed Buffer Size from 512MB to 4096MB versus Delivery Ratio. (b) Buffer Delivery: Randomized Buffer Size from 512MB to 1024MB versus Delivery Ratio. (c) Power Delivery: Fixed Power Size from 50 watt to 5,000,000 watt versus Delivery Ratio. (d) Power Delivery: Randomized Power Size from 50W to 50000W versus Delivery Ratio. (e) Bandwidth Delivery: File size to be transmitted versus Delivery Ratio. (f) Popularity Delivery: Number of send/receives per window size versus Delivery Ratio

MB buffer size. We speculate this result is generated due to uniform distribution. At 128MB mark, the maximum buffer size of a device is 1280MB. Since half of the devices will have a buffer size greater than 640MB, there is enough total storage available to allow PRoPHET + Buffer to be effective. After 256MB mark, however, most devices have a storage capacity capable of supporting most of the data transmission, therefore resulting in PRoPHET + Buffer to lose its value.

2) Power Parameter Delivery Ratio: Similar to buffer, we evaluated power under two circumstances. The amount of power is measured in watts. Every send and transfer removes a certain amount of power. We also included a small idle time power consumption to simulate real world situations.

We first evaluate the effect of fixed power for all nodes. As we observe in Figure 1(c), PRoPHET + Power performs slightly worse than PRoPHET under small power, but performs similarly well to PRoPHET as power increases. We speculate that since we reserve an amount of power to ensure forwarding of the existing packets in buffers, in cases where there's a small amount power available throughout the nodes, nodes will reject incoming transmissions fairly quickly. Once all candidate nodes refrain from accepting files, the remaining packets from the sender node will not be forwarded. When encountering of potential candidate nodes or destination node are rare, some sender nodes may run out of power before forwarding all packets in their buffers to another node or destination node due to idle power consumption. PRoPHET performs slightly better because PRoPHET does not need to reserve power for delivery when receiving packets, therefore it is able to receive more files and therefore mitigates the negative effect of idle time power consumption.

For randomized power size, each value in x-axis represents the minimum buffer for a device in a uniform distribution, with the max buffer size being 100 times the value. We observe in Figure 1(d) that PRoPHET + Power performs better than PRoPHET as power increases, and flattens out as power reaches near infinity. This is because at low power most nodes will have a similar effect as in low power during fixed power. It performs slightly better than fixed power in low power situation mainly due to the existence of some nodes that have ample amount of power to forward and receive packets without the idle power consumption being too significant in the reduction of remaining power. Once power capacity approaches 5000 watts, many nodes are able to support a reasonable amount of transmissions, PRoPHET + Power performs better than PRoPHET since it will transmit to nodes with more power available, therefore keeping the nodes alive longer and increasing the chances of packets forwarding. Since PRoPHET does not take power into assumption, some packets will be forwarded to nodes that have low power, therefore resulting in packet loss and faster shorter lifespan for low power capacity nodes. Furthermore, PRoPHET +Power attempts to keep packets alive, since if the node containing the packets is about to run out of power, PRoPHET + Power will attempt to deliver packets to a node with more power. PRoPHET in this case will not keep data alive since nodes do not take into account their own power status.

3) Bandwidth Parameter Delivery Ratio: Bandwidth is measured using the file size per transmission. The standard transmission rates of 802.11b is used, with the different transmission rates randomly distributed to each device in the network. We use file size as the measurement to observe the effect of forwarding data that requires long transmission time versus short transmission time. As we observe in Figure 1(e), PRoPHET + Bandwidth performs similar to PRoPHET. There is a small gain near 100MB file size, but otherwise remains consistent with PRoPHET. We believe that since the trace is obtained from a conference, the contact time between nodes is relatively long. Since most of the file sizes are either too small to impact transmission rate within the contact time, or too large to successfully deliver even if a fast connection is used, the effect of bandwidth was not significant. However, we believe that PRoPHET + Bandwidth should perform better in an environment where contact periods are short.

4) Popularity Parameter Delivery Ratio: As we observe in Figure 1(f), PRoPHET + Popularity performs nearly identical to PRoPHET. We analyzed the data of iMote, and noticed that in a conference, there is no single node that is especially popular. In a conference, nodes encounter other nodes relatively often. Therefore, while PRoPHET + Popularity does not show any improvement in this scenario, we believe that similar to bandwidth, it should show some promising results in other cases.

B. Delay

After delivery ratio, we observe delay in each of the different parameters. In all simulations, we compare the amount of data delivered after a certain amount of time. Given a point in time, the higher the data delivery ratio, the less delay there is for the protocol.

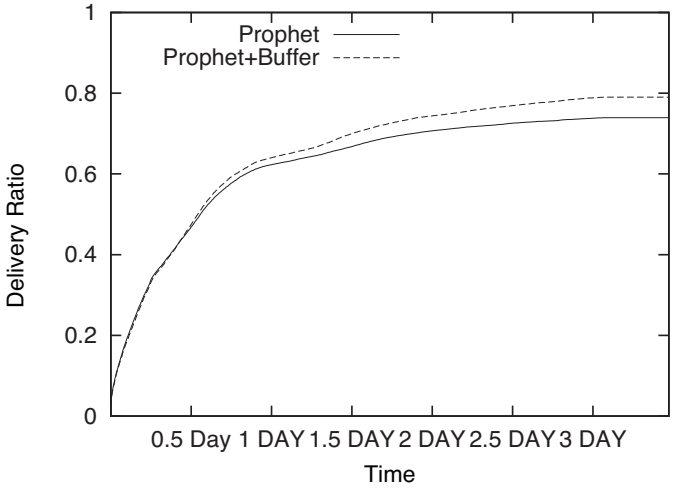

Fig. 2. Buffer Delay: Number of days versus Delivery Ratio

1) Buffer Parameter Delay Evaluation: As we observe in Figure 2, PRoPHET + Buffer does not cause delay of transmission. In fact, it slightly decreased the delay due to the dispersion of data to multiple nodes, therefore effectively allowing multiple delivery to occur simultaneously. Furthermore, we can observe from the graph that PRoPHET + Buffer does deliver more data than PRoPHET eventually. This is caused in short by the mitigation of data loss caused by FIFO queues in PRoPHET + Buffer.

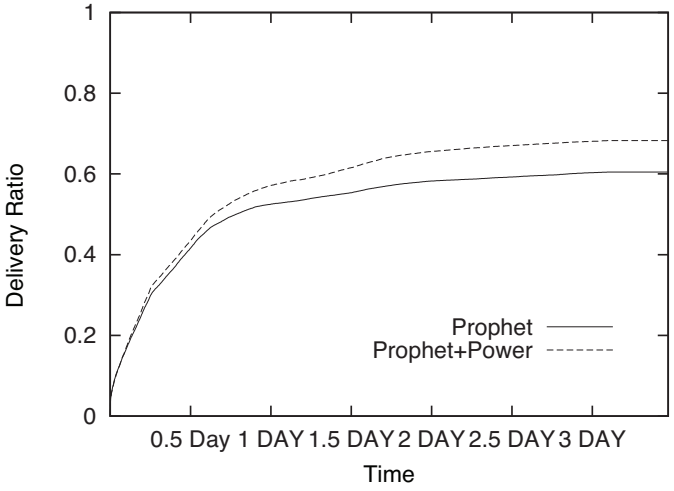

Fig. 3. Power Delay: Number of days versus Delivery Ratio

2) Power Parameter Delay Evaluation: As we observe in Figure 3, PRoPHET + Power also does not cause delays in data delivery. Performance of PRoPHET + Power can be explained using the same methodology as PRoPHET + Buffer.

3) Bandwidth and Popularity Parameter Delay Evaluation: While we did run experiments for bandwidth and popularity to test delay, there were no significant improvements. We analyzed the trace, and noticed that due to the characteristics of the trace, contact time is typically long and popularity is spread evenly throughout most nodes.

C. Initial Weighted Function

As we observe in Figure 4, PRoPHET+ has a higher delivery ratio than PRoPHET. Since this is an initial test case without application of any well-formed weighting values, predictability value weight was set to 0.5 to conform to the individual parameter testing setup. The rest of the 0.5 was distributed arbitrarily into the other four parameters. The purpose of this simulation is to confirm our hypothesis that a weighted function should perform better than PRoPHET in terms of delivery ratio. The figure also indicated an improvement in terms of delay. We observe from the graph that PRoPHET+ is able to deliver the total amount of data PRoPHET faster than PRoPHET. PRoPHET took approximately one day to deliver up to PRoPHET's upper bound delivery ratio, where PRoPHET+ took a little more than half a day to deliver up to that delivery ratio. We believe that this is caused mainly because of the dispersion of data to various nodes due to buffer and power. The dispersion creates an environment where data can be forwarded by more nodes at one time. Next, we present our proposed weighted function using careful considerations.

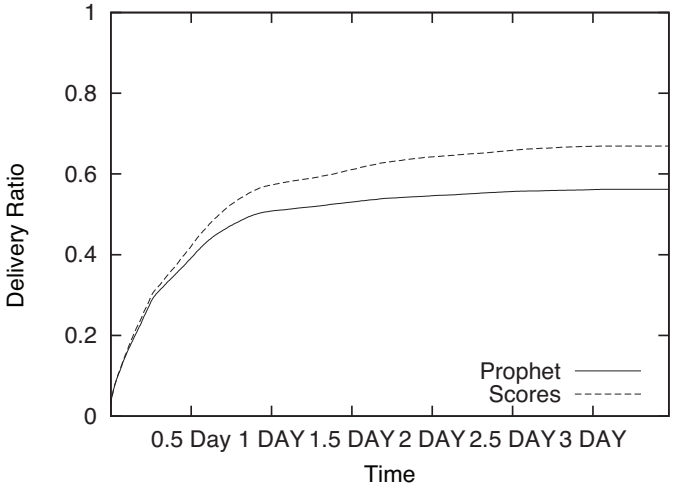

Fig. 4. Number of days that has passed versus Delivery Ratio using V_D = $0.2(V_B)$ + $0.2(V_P)$ + $0.05(V_A)$ + $0.05(V_O)$ + $0.5(V_R)$.

D. Proposed Weighted Function

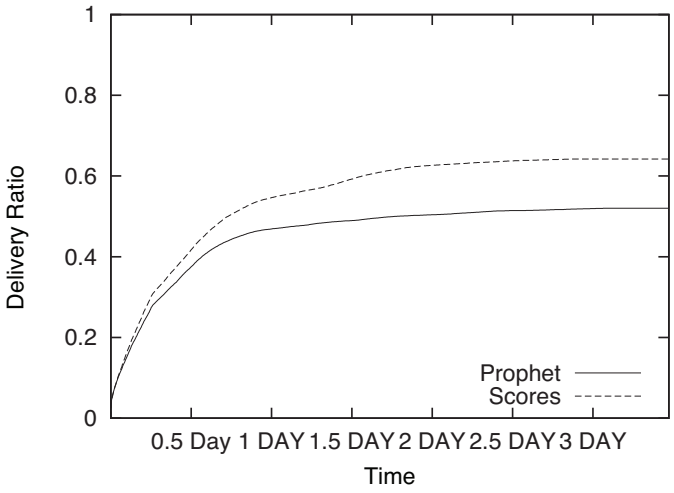

Fig. 5. Number of days that has passed versus Delivery Ratio using V_D = $0.25(V_B)$ + $0.25(V_P)$ + $0.1(V_A)$ + $0.1(V_O)$ + $0.3(V_R)$.

As we observe in Figure 5, PRoPHET+ delivered more data than PRoPHET and had less delay. While the overall graph looks similar, we can see that improvement in PRoPHET+ began earlier than the initial test case. Furthermore, the difference of delivery ratio between PRoPHET+ and PRoPHET was greater than the previous weighted function. This graph supported our hypothesis that not only could PRoPHET+ perform better or equal to PRoPHET, but by shifting the weights, PRoPHET+ could also increase its performance.

VI. Future Work

We believe location is another parameter that should be considered. If the node that contains the packets to be forwarded are in a location that the destination node visits frequently, then the packets to be forwarded have a greater chance to be delivered to the destination node than nodes containing the data that is in locations where the destination node does not visit often. There is also room for improvement for evaluation of PRoPHET+ in different environments. We only simulated PRoPHET+ in one scenario, therefore we cannot guarantee that the function we used will be suitable for other situations. Furthermore, our trace data is obtained in a conference, which means that the contact time tends to be long, and there is less difference in popularity. Moreover, if some type of coding techniques is implemented, PRoPHET+ could potentially have an increase in delivery success rate. Finally, our weights were generated using a logical hypothesis and a trial test. We believe that if optimum weights are identified for specified environments, PRoPHET+ should yield an even better result.

VII. Conclusion

We proposed PRoPHET+, a routing protocol for opportunistic networks that reduces chances of data loss and delivery delay while maintaining the high probability of delivery successfulness between a source and destination node by providing communication opportunities. Buffer, power, bandwidth, popularity, and predictability value from PRoPHET were used to create a weighted function. In theory, the weights could be shifted to adapt to various environments. Even though our weights were determined based on our hypothesis and a few trials, simulations performed have shown that in a closed community situation using our weights, PRoPHET+ performs well in both delivery success rate and delay.

References

[1] Probabilistic routing in intermittently connected networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(3):19–20, 2003.

[2] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proc. IEEE INFOCOM*, pages 1–11, 2006.

[3] Frans Ekman, Ari Keränen, Jouni Karvo, and Jörg Ott. Working day movement model. In *MobilityModels '08: Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models*, pages 33–40, New York, NY, USA, 2008. ACM.

[4] Khaled A. Harras, Kevin C. Almeroth, and Elizabeth M. Belding-royer. Delay tolerant mobile networks (dtmns): Controlled flooding schemes in sparse mobile networks. In *IFIP Networking*, 2005.

[5] Sushant Jain, Michael Demmer, Rabin Patra, and Kevin Fall. Using redundancy to cope with failures in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 35(4):109–120, October 2005.

[6] J. LeBrun, Chen-Nee Chuah, D. Ghosal, and M. Zhang. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, volume 4, pages 2289–2293 Vol. 4, May-1 June 2005.

[7] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing for dtns. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–10, Barcelona, Spain, April 2006.

[8] Yong Liao, Kun Tan, Zhensheng Zhang, and Lixin Gao. Estimation based erasure-coding routing in delay tolerant networks. In *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, pages 557–562, New York, NY, USA, July 2006. ACM.

[9] Chien-Shiu Lin, Wei-Shyh Chang, Ling-Jyh Chen, and Cheng-Fu Chou. Performance study of routing schemes in delay tolerant networks. In *AINAW '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications - Workshops*, pages 1702–1707, Washington, DC, USA, 2008. IEEE Computer Society.

[10] Daddy Marasigan and Papa Rommel. Mv routing and capacity building in disruption tolerant networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 1, pages 398–408 vol. 1, March 2005.

[11] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. In *World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, pages 183–189, June 2005.

[12] Ram Ramanathan, Richard Hansen, Prithwish Basu, Regina Rosales-Hain, and Rajesh Krishnan. Prioritized epidemic routing for opportunistic networks. In *MobiOpp '07: Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 62–66, New York, NY, USA, 2007. ACM.

[13] James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD data set cambridge/haggle (v. 2006-09-15). Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, September 2006.

[14] Chien-Chung Shen, G. Borkar, S. Rajagopalan, and C. Jaikaeo. Interrogation-based relay routing for ad hoc satellite networks. *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, 3:2920–2924 vol.3, Nov. 2002.

[15] Libo Song and David F. Kotz. Evaluating opportunistic routing protocols with large realistic contact traces. In *CHANTS '07: Proceedings of the second ACM workshop on Challenged networks*, pages 35–42, New York, NY, USA, 2007. ACM.

[16] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 235–244, 2004.

[17] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Department of Computer Science, Duke University, Durham, NC, 2000.

[18] Yong Wang, Sushant Jain, Margaret Martonosi, and Kevin Fall. Erasure-coding based routing for opportunistic networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 229–236, New York, NY, USA, August 2005. ACM.

[19] Jörg Widmer and Jean-Yves Le Boudec. Network coding for efficient communication in extreme networks. In *WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 284–291, New York, NY, USA, 2005. ACM.