Darpa Contract # W15P7T-05-C-P413
Deliverable #3(b)

Performance of Point-to-multipoint (p2mp) Bundle Delivery in DTN
Lehigh University
Oct 31st, 2005

1. Proposed Ideas

In a typical battlefield or emergency scenario, important information needs to be disseminated to different members within a group or even to members from different groups. Thus, an efficient point-to-multipoint (p2mp) bundle delivery protocol needs to be designed. A conventional IP-multicast approach does not work in a DTN environment since the multicast delivery tree may not exist all the time.

In the proposal of our EDIFY project, we have proposed to design a point-to-multipoint (p2mp) bundle delivery protocol with two techniques: implicit tree forwarding and explicit tree formation. The former approach handles the case in which bundles need to be delivered to all nodes within a group. The latter approach deals with both cases of sending group messages in bundles to all or subset of nodes.

We have focused on studying the second approach because of the following two reasons: (i) The first approach is just a bundle flooding across the DTN nodes in a group, which can be fulfilled by having every DTN node in the group relay any received bundles in a unicast fashion to all its DTN neighbors that have been discovered; and (ii) When the multicast receivers in the second approach include all the nodes in a group, then the second approach achieves the same results as the first approach.

2. Challenges

In DTN networks, data communication challenges exist in network scenarios where an instantaneous end-to-end path between a source and destination may not exist because links between nodes may be opportunistic and predictably/periodically connected. Traditional multicast methods proposed for the Internet (e.g., MOSPF [1] and DVMRP [2]) or mobile ad hoc networks (e.g., AMRoute [3] and ODMRP [4]) are not suitable for DTNs, due to the challenge of frequent network partitions. Firstly, it is difficult to maintain the connectivity of a source-rooted multicast tree (or mesh) during the lifetime of a multicast session. Secondly, data transmissions suffer from large end-to-end delays along the tree because of the repeated disruptions caused by periodically broken branches. Thirdly, the traditional approaches may fail to deliver a message when the possibility of link unavailability becomes high (e.g. ~80%).

In this project, we have proposed an on-demand situation-aware multicast (OS-multicast) approach, which is a dynamic tree-based method that integrates DTN multicasting with the situation discovery by the underlying network layer. We have compared its performance with other algorithms, such as unicast-based, static tree-based, and DTBR (another dynamic tree-based) multicasting strategies, in terms of bundle delivery ratio and bundle delivery latency. We have observed that the OS-multicast approach achieves higher message delivery ratio when the probability of link unavailability is high and the duration of link downtime is large.

3. Assumptions

We view a DTN as an overlay built upon underlying networks, such as wireless ad hoc networks. Only those hosts or nodes that implement DTN functionalities to send and receive bundles are DTN nodes, while the others are denoted as normal nodes. A DTN link may consist of several underlying links. The

underlying network provides unicast routing capability to forward a bundle from one DTN node to another. The multicast service discussed here is only implemented in the DTN overlay.

We assume that before the DTN overlay starts, the underlying network has already been operating for a relatively long time. Also we assume that the routing protocol of the underlying networks, such as dynamic source routing (DSR [5]), can collect network condition information. Thus in this project, we provide situational awareness for the DTN p2mp or multicast bundle delivery by making the multicast implementation collaborate with the routing method (e.g. DSR) in the underlying networks. Periodically, a DTN routing agent sends *situation_req* message to trigger its underlying routing agent to collect the current network conditions, such as the presently available outgoing links and the discovered paths from the current DTN node to the destinations. The underlying routing agent then answers *situation_resp* message with all the detected information back to the DTN routing agent. Both *situation_req* and *situation_resp* messages are system messages transmitted inside a node. Thus, a knowledge base of the link state and network topology is constructed in each DTN node.

4. Multicast Approaches

First, we introduce the OS-multicast approach for p2mp bundle or message delivery in DTN, along with unicast-based multicast and static tree-based multicast approaches.

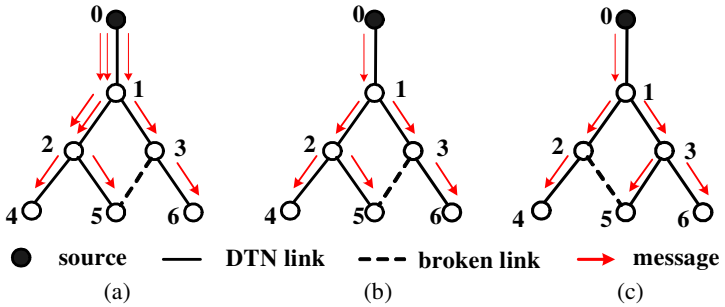


Figure 1. Multicast approaches in DTN. (a) U-multicast; (b) ST-multicast; (c) OS-multicast: when link 2→5 is unavailable and link 3→5 becomes available, node 3 will take advantage of the current available link immediately.

- Unicast-based multicast (U-multicast). This approach implements multicast service by using multiple source-to-destination unicast data transfer. In U-multicast, the source always tries to set up an end-to-end path to each destination and sends a copy of the bundle along the currently available shortest path. If there is no such path for a destination due to the network partition, the source will hold the bundle in its buffer and retransmit it once the destination is connected. For example in Fig.1 (a), the source (node0) sends three identical copies of bundles to the receivers (node4, node5, node6), along the currently available paths 0→1→2→4, 0→1→2→5, and 0→1→3→6. Obviously it's very inefficient to transmit the same bundle twice over link 1→2 using U-multicast.

- Static Tree-based multicast (ST-multicast). In ST-multicast, a source-rooted multicast tree is constructed at initialization by querying the shortest paths to reach each destination from its knowledge base. Then the source keeps sending bundles along this static tree and requires the intermediate DTN nodes with multiple downstream neighbors to duplicate the messages. In this scheme, the bundle will only be duplicated at the branching point so fewer transmissions will be required to deliver the bundle to all receivers. For instance, assume that a multicast tree is constructed (shown as solid lines in Fig.1 (b)). Then each bundle will be forwarded by the intermediate DTN nodes, along this discovered tree. If a branch, e.g. link 2→5, is broken but a non-tree link 3→5 becomes available, ST-multicast approach won't take this opportunity to ask node3 to forward the bundle to node5 since link 3→5 is not in the static tree.

- On-demand Situation-aware multicast (OS-multicast). Unlike the above two methods, OS-

multicast is a dynamic tree-based multicast approach. A unique multicast tree is built for each bundle and the tree varies according to the changes of the network at each intermediate DTN node. First, a source-rooted tree is constructed in a similar manner as the ST-multicast approach. When a DTN node receives a bundle, it will dynamically rebuild the tree rooted at itself to all the destinations based on its knowledge of the current network conditions. If there is a newly available path to a destination, which is not discovered by the upstream DTN nodes, this node will immediately take advantage of that fresh information and send the bundle out along the newly available path.

DTBR [6] is another dynamic tree-based multicasting algorithm designed for DTNs. Similar to OS-multicast, DTBR also requires the intermediate DTN nodes to re-build a multicast tree. However, in each step of the bundle transmission, the upstream node will assign a partial receiver list to its downstream neighbors based on its local view of the network conditions. The downstream nodes are required to forward bundles only to the receivers in the list, even if a new path to another receiver (not in the list) is discovered. Fig. 2 depicts this issue.

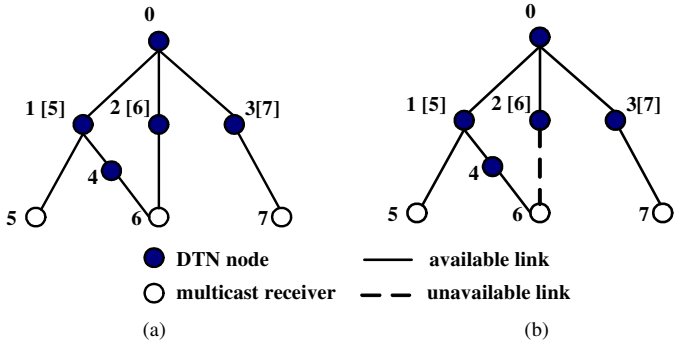


Figure 2. (a) At time t_0, the source (node0) computes the multicast tree and decides that the receiver list for node 1 is {5}, for node 2 is {6}, and for node 3 is {7}. (b) At time $t_1 > t_0$, link 2→6 is down. Node2 then can not forward bundle to node6. However, there is another available path 1→4→6 exists in the networks but node1 cannot take advantage of it just because node6 is not in its receiver list. Before node0 detects this new path, it will keep sending bundles to node2 and ignore the other better opportunity.

The issue illustrated by Fig. 2 is solved in the OS-multicast scheme because in this scheme, each intermediate node has an equal chance to decide the receiver list. If a link towards a receiver becomes available, the DTN nodes which detect that will immediately take advantage of this new opportunity. We call this greedy nature of OS-multicast as the first availability property. This property guarantees that the bundle will be delivered to the receivers as soon as possible and the opportunistic links will be utilized. The price with this approach is the destination may receive more than one copy of the same bundle.

The authors in [6] assumes that each node may have the complete knowledge or the summary of the link states in the networks when they simulate DTBR. However, this is hard to be satisfied in most practical applications. In our design of OS-multicast scheme, the situation-aware knowledge is collected with the help of underlying routing methods. The situation-aware knowledge includes the information of network conditions, such as the current available outgoing links and the possible paths to destinations.

5. Details of OS-multicast Algorithm

a) Membership management

When a DTN node intends to join a multicast group, it registers with its membership period by explicitly sending a *GROUP_JOIN* message. It also informs its DTN routing agent the G_{id} that it registers for. For example, node i wants to join the multicast service during the period $[t_{si}, t_{ei}]$, with the start-time t_{si} and the end-time t_{ei} of its membership. When the multicasting source is informed by the *GROUP_JOIN* message, it puts the membership information into a *membership_list*, denoted as L_M in each bundle. For every bundle received, generated or retransmitted at time t, a DTN node will check the validation of each receiver in L_M. If the membership of a receiver is expired ($t > t_{ei}$) or not activated ($t < t_{si}$), then it will not forward the

bundle to that receiver. A receiver is called a valid receiver only if it owns valid membership for that bundle. This membership management method conforms to the TM semantic model proposed in [6] with explicit receiver list known at the source.

b) Bundle storage

Each DTN node has a local storage with finite size. A received bundle will be saved in the buffer until being eliminated due to the buffer overflow or successfully forwarded toward all the valid receivers based on the bundle acceptance policy, e.g. when the buffer is full, the bundle at the head of the buffer will be dumped first with the arrival of a new bundle.

c) Forwarding state maintenance

A forwarding state is associates with each bundle. There is an *upstream_list* (called L_U) maintained in the bundle and a *pending_list* (called L_P) maintained by the DTN node. When a bundle arrives, a DTN node creates L_P for that bundle by copying L_M information from the bundle. L_P is periodically checked to remove information corresponding to those receivers whose memberships have expired. When node A wants to forward the bundle to a downstream node B to reach a valid receiver C, it first checks if $B \in L_U$ to avoid the redundancy. Then it removes C from L_P. Once L_P is empty, the bundle is then removed from the local buffer of node A.

d) Message forwarding

When a bundle is generated, the source queries its knowledge base to retrieve all the discovered paths to the receivers. Then the source combines those paths to be a multicast mesh that covers all the valid receivers. We denote it as a *static-mesh*. The source then filters this mesh by deleting those currently unavailable outgoing links reported by the underlying routing agent. We denote the result as a *dynamic-mesh*. Based on the *dynamic-mesh*, a source-rooted shortest-path multicast tree is built and the source forwards the bundle to all its available downstream DTN neighbors, carrying the *static-mesh* information in the bundle.

Once receiving a bundle, a DTN node X will first query its knowledge base to find all the possible paths to the receivers. Then it combines the *static-mesh* in the bundle with the query results to be a new static-mesh. A *dynamic-mesh* is then constructed in the same way as what the source has done. Then the DTN node X re-computes the shortest-path multicast tree by taking itself as the root to all the receivers. The new *static-mesh* is put into the bundle again and forwarded further to downstream neighbors. With the knowledge propagation and the dynamic decision made by each intermediate DTN node, eventually the bundle will arrive at those receivers in L_M. Note that the same bundle will not be forwarded twice at each node.

e) Bundle retransmission

A DTN node periodically checks its local storage to see if there is any opportunity to forward the buffered bundles further. It uses the current *dynamic_mesh* to find out if there is a chance to forward the bundle to a receiver in L_P. If so, the bundle is forwarded with its current *static_mesh* and the receiver is removed from L_P. To reduce the overhead of the OS-multicast, there is an upper-bound R_{upper} to limit the maximum retransmission times for each receiver. If the retransmission to receiver D fails more than R_{upper} times, D would be removed from L_P. Once L_P is empty, the bundle will be deleted from the local buffer.

6. Performance Evaluation

To evaluate the performance of different multicast algorithms in this study, we implement U-multicast, ST-multicast, OS-multicast and DTBR in *ns* simulator. Our performance metrics include: *i*) *message delivery ratio*, which is defined as the number of unique multicast bundles successfully arrived at all the receivers over the total number of bundles which are expected to be received; *ii*) *efficiency*, which is the ratio between the unique bundles received by the receivers and the total traffic generated in the networks; and *iii*) *average message delay*, which is the average of end-to-end bundle transmission delays for each algorithm.

6.1. Preliminary Results

In our preliminary test, all simulations have 25 nodes deployed in a 1000×1000 area. The topology is a grid network. 15 DTN nodes are randomly selected to construct the DTN overlay and others are normal nodes. Multicasting algorithms are only implemented in those DTN nodes. DSR is used to be the routing approach for the underlying ad hoc networks. And situation awareness is achieved by the communication between DTN multicasting agent and the DSR routing agent. The MAC layer is IEEE 802.11 with radio transmission range of 250 meters. And all simulations last for 7200 seconds.

We study one multicast session in the DTN overlay. Node 0 is fixed to be the source and 5 DTN nodes are randomly chosen to be the receivers. The message sending rate is 1 bundle per 2 seconds with the bundle size of 512 bytes. Each DTN node can maximally keep 100 bundles in its local buffer. At every 5 seconds, each DTN node will query the underlying routing agent to detect if there is any available opportunistic link to forward the buffered bundle to the destinations. If so a copy of the bundle will be forwarded. We study the performance of different multicasting algorithms by varying the percentage of link unavailability of each link from 10% to 90%, which is defined as the ratio of the total downtime of a link over the total simulation time.

Fig. 3 shows the result of the message delivery ratios of U-multicast, ST-multicast, OS-multicast and DTBR. We observe that *i*) the delivery ratio decreases for all the algorithms when the downtime of links becomes large, i.e., the network is more easily to be partitioned; *ii*) OS-multicast can always achieve the best performance among all the algorithms; and *iii*) U-multicast performs better than ST-multicast when the downtime is small because it basically tries to forward bundles in a multicast mesh than a tree. For each bundle transmitted by each intermediate DTN node, OS-multicast tries to utilize multiple paths to the receivers and take advantage of the currently available opportunistic links to push the data closer to the destinations.

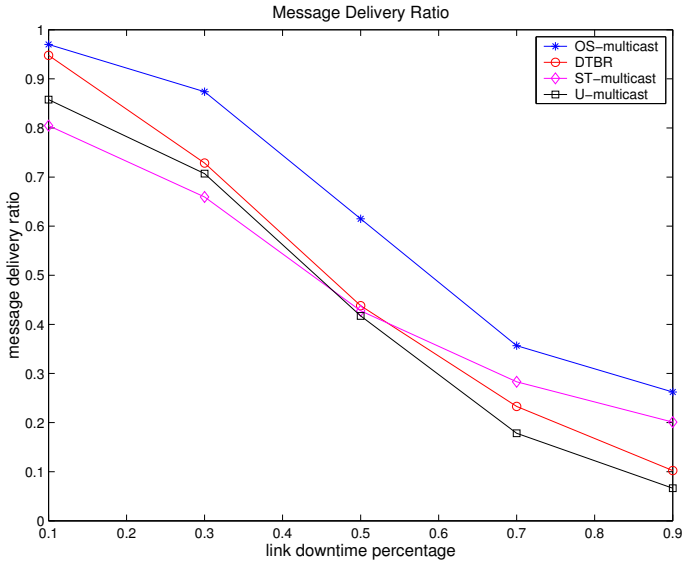


Figure 3. Message delivery ratios of different multicasting algorithms.

The above simulation is conducted with uniformly dividing the downtime of each link into 10 small periods. Fig.4 shows the meaning of 10 small periods of a link with link downtime percentage be equal to 0.7.

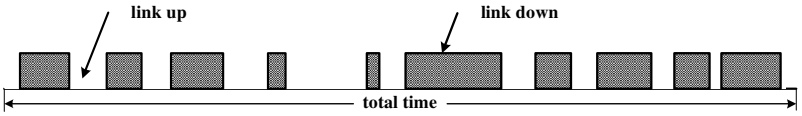


Figure 4. 10 periods of a link up/down pattern with total link downtime being 70% of the total simulation time.

To better show the performance of OS-multicast, we also test the message delivery ratio performance by dividing the total downtime into 30 periods, i.e., making the link states vary more often and increasing the

occurrences of opportunistic links. The result is illustrated in Fig. 5. It is obvious that *i*) all the methods can deliver more bundles and *ii*) OS-multicast outperforms the other approaches more than the result shown in Fig. 3. This proves that OS-multicast is able to utilize the opportunistic links more effectively.

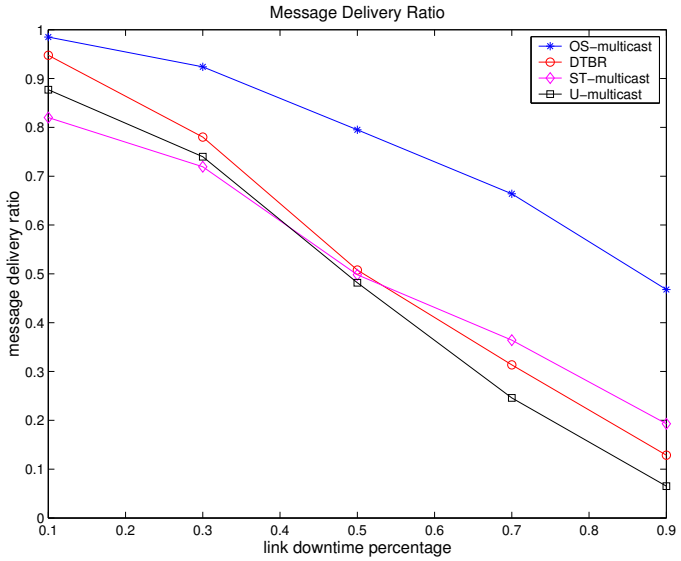


Figure 5. Message delivery ratios of different multicasting algorithms with more occurances of opportunistic links.

The following simulations are all carried out with dividing the downtime of each link into 10 periods. Fig. 6 depicts the median message delay of two dynamic-tree based multicasting approaches: OS-multicast and DTBR. OS-multicast has smaller delays than DTBR due to the issues illustrated in Fig. 2. In DTBR, each node only forwards bundles to the downstream nodes to reach the receivers in its receiver list. However, this receiver list is decided by the upstream node based on its snapshot of the network conditions. In this way, some opportunistic links to the receivers that are not in the list are missed. On the contrary OS-multicast method always uses all the chances to forward bundles to the destinations. Note that in [6] results show that DTBR achieves slightly better delay performance than ST-multicast.

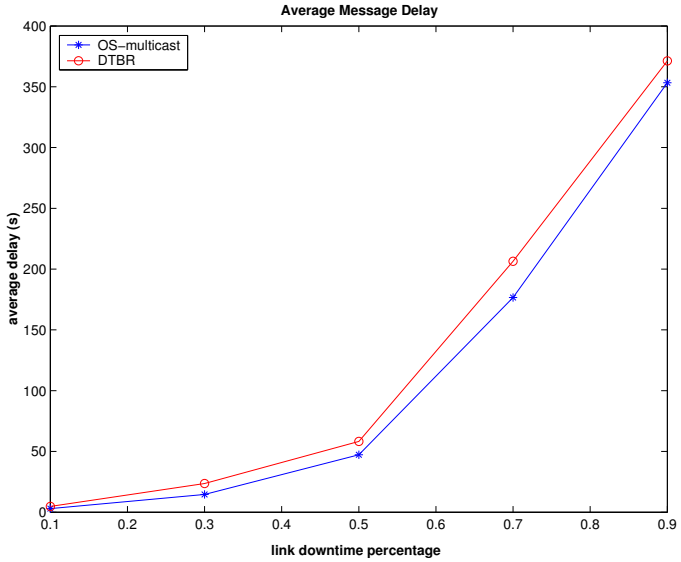


Figure 6. Average message delays between OS-multicast and DTBR.

We also evaluate the efficiency of various multicasting approaches and illustrate the result in Fig. 7. It shows that although OS-multicast has the highest delivery ratio it has the worst efficiency when the link downtime is small. The reason is that a lot of redundant traffic has been introduced by OS-multicast due to

its nature of utilizing multiple currently available links. However, when the link downtime percentage is larger than 80% of the total simulation time, its efficiency increases because the chance of redundancy decreases and it is still able to achieve the high delivery ratio. The efficiencies of the other three algorithms decrease while the network connectivity becomes worse.

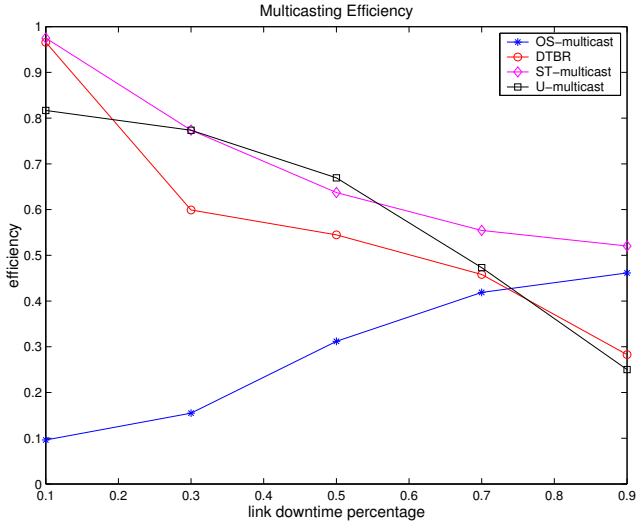


Figure 7. Multicast efficiency of different algorithms.

6.2. Sensitivity Studies of the OS-multicast Approach

The sensitivity study of different multicasting algorithms is conducted by varying the configurations of DTN scenarios, including the percentage of DTN nodes in the whole networks, the local buffer size of each DTN node, and different buffer management policies. We focus on studying how the performance of four P2MP approaches (U-multicast, ST-multicast, DTBR, and OS-multicast) is affected by varying network environments.

All the tests in our sensitivity study still have 25 nodes deployed in a 1000×1000 area to be a grid network and DSR is the routing algorithm for the underlying ad hoc network. One multicasting session is studied with node 0 to be the source and four randomly selected DTN nodes to be the receivers (unlike 5 receivers in our preliminary study). All simulations last for 1800 seconds. The source sends out a DTN bundle every 2 seconds. And each DTN node updates its routing knowledge base and retransmits the buffered bundles to the destinations at every 5 seconds.

a) Varying the DTN node percentage

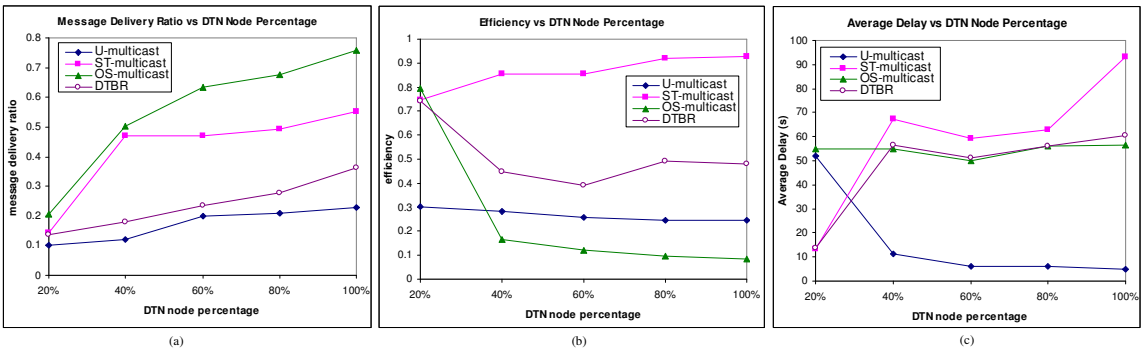


Figure 8. Performances of different multicasting algorithms with varying DTN node percentage

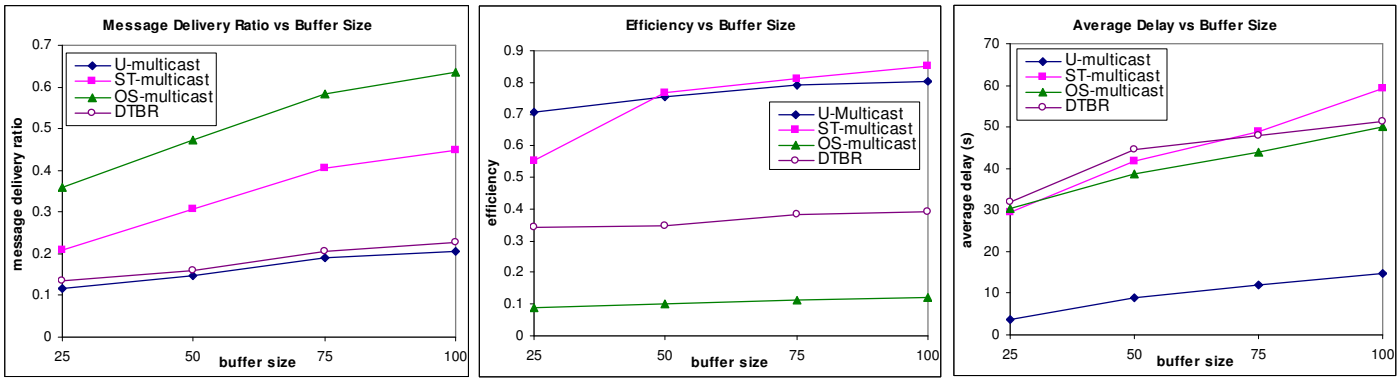
We first change the percentage of DTN nodes in the overall network from 20% to 100%, i.e., from 5 to

25 nodes. In all the tests, each link may randomly be down for 70% of the total simulation time. The results are plotted in Fig. 8.

We observe that *i*) For all multicasting algorithms, the more DTN nodes in the networks, the higher the message delivery ratio they can achieve. They all take advantage of the store-and-forward data forwarding scheme supported by the DTN layer. *ii*) OS-multicast always has the best performance in terms of successful bundle delivery ratio to multiple receivers. The performance gain over other approaches increases when more nodes implement DTN functionalities. *iii*) OS-multicast has acceptable average end-to-end transmission delays comparing to ST-multicast and DTBR. i*v*) However, the efficiency of OS-multicast becomes worse when the number of DTN nodes increases because more redundant traffics will be present in the networks.

b) Varying the local buffer size

We have studied the impact of the node buffer size on the performances of different algorithms, under the condition that 60% of the nodes are DTN nodes. Each DTN node can hold from 25 to 100 bundles in its local storage. The FIFD buffer management policy (see the next section) is used in all the tests. The results are shown in Fig. 9.



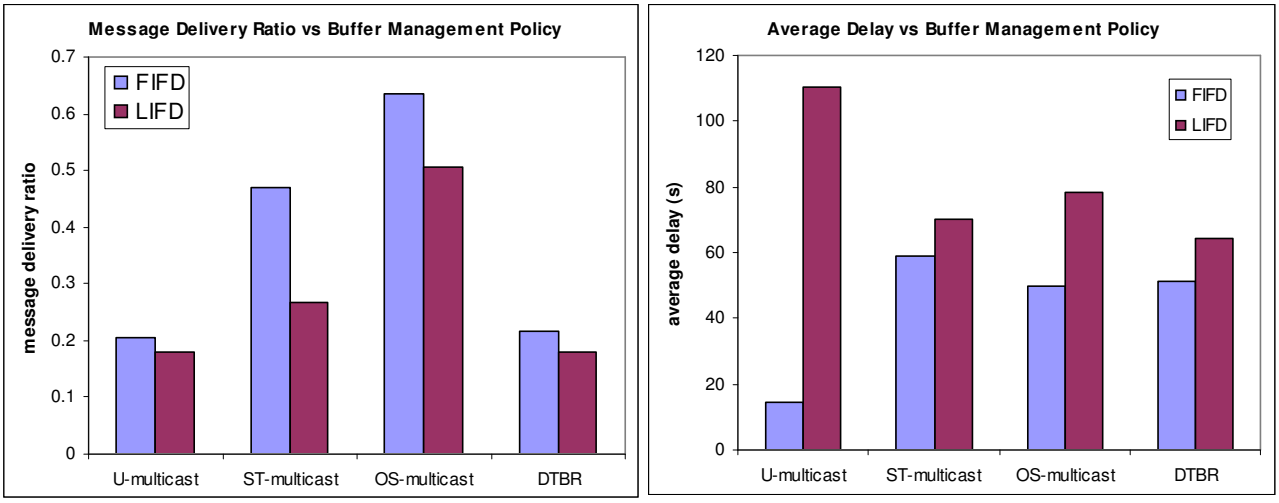
(a) (b) (c)

Figure 9. Performances of different multicasting algorithms with varying the local buffer size

According to Fig. 9, *i*) the larger the buffer size, the higher the achieved message delivery ratio is for all the multicasting algorithms. The reason is because more bundles can be kept by the intermediate DTN nodes. *ii*) OS-Multicast can deliver more bundles to the destinations than other approaches for all the tested buffer size cases. *iii*) OS-Multicast has acceptable average transmission delays but the worst efficiency due to its large redundancy. *iv*) The efficiency of all algorithms becomes slightly better with larger buffer sizes, because the benefit of increasing delivery ratio outperforms the cost of increasing bundle retransmissions.

c) Varying the buffer management policy

Two simple buffer management policies are studied by simulations: FIFD (First In First Drop) and LIFD (Latest Incoming First Drop). When FIFD applied, DTN nodes always drop the first item from its local buffer when the buffer is full and some fresh incoming bundles needed to be saved, i.e., it always discard the oldest bundle kept in the buffer. When LIFD applied, DTN nodes will keep the current content inside its buffer when the buffer is full and always discards the incoming new bundles. FIFD policy implies that the destinations prefer the fresher information from the source and LIFD implies that the history information is more favorable. In all tests, each DTN node has buffer size of 100 and there are 60% DTN nodes in the networks. The results are illustrated in Fig. 10.



(a) (b)

Figure 10. Performances of different multicasting algorithms with varying the local buffer size

We observe that *i*) All four algorithms achieve better message delivery ratio and smaller average delays with FIFD than LIFD. This is because LIFD always sends out historical bundles from the buffer first before the latest fresh incoming bundles get a chance to be stored. *ii*) With LIFD, U-multicast has the worst average transmission delays because it relies on the availability of the end-to-end paths so that the historical bundles saved in the source's buffer can be sent out. When a DTN network suffers from worse connectivity (links are down in 80% of the time), U-multicast has to make buffered bundles wait longer till the end-to-end path from the source to the destinations becomes available.

d) Varying the inter-contact duration pattern

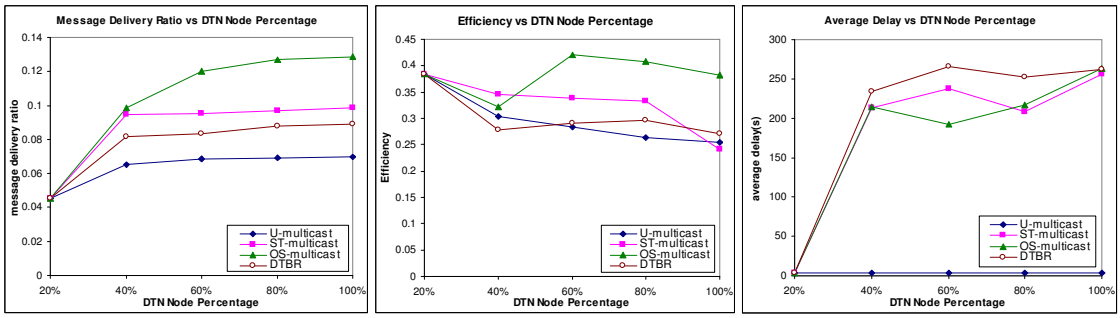


Figure 11. Performances of different multicasting algorithms with power-law distributed inter-contact durations

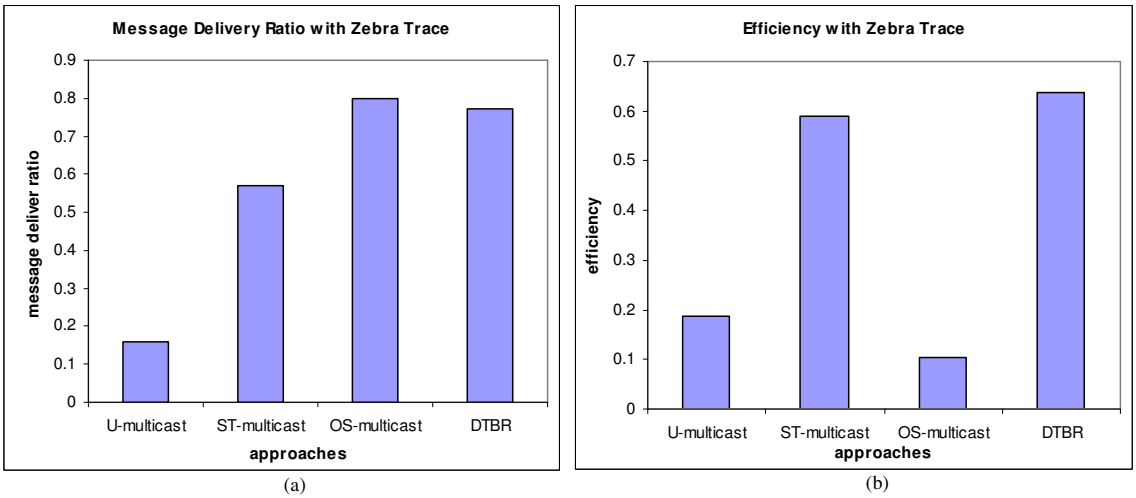
In the sensitivity studies discussed in the earlier section, the inter-contact duration follows a uniform distribution. However, it was reported in [7] that the inter-contact durations observed from the PSN trace can be approximated as a power-law distribution. It implies that the probability of long DTN link unavailability is larger than that can be observed in a uniform distribution . Similar power law pattern of inter-contact durations is also observed in other experiments, e.g. Dartmouth trace [8]. By using the power-law distribution obtained from [7] as the distribution for the link unavailability duration, we compare the performances of all multicasting algorithms under the following conditions: *i*) the total link downtime is 70% of the total simulation time; *ii*) the buffer size is 100; *iii*) The FIFD buffer management policy is used; and *iv*) The DTN node percentage varies from 20% to 100% in the overall networks. The results are shown in Fig. 11.

We observe that *i*) When the power-law distributed inter-contact duration pattern is applied, the message delivery ratio of all algorithms becomes much worse compared with the results shown in Fig. 8. This means that the more long link-downtime periods occurred in DTN, the more difficult to transmit bundles to the destinations. *ii*) OS-multicast still achieve the best message delivery ratio due to its nature of utilizing the

current available contact to push bundles further to the receivers. *iii*) The efficiency of OS-multicast becomes better because it is able to deliver more data than the other algorithms in bad network connectivity conditions. And *iv*) OS-multicast has acceptable average delays compared to DTBR and ST-multicast.

e) Applying Zebranet trace

Using the information derived from the Zebranet trace [6], we derive various mobility and inter-contact duration patterns which we apply to our P2MP simulations. The results depicted in Fig. 12 show that OS-Multicast still provides the best performance in terms of message delivery ratio and the worst performance in terms of efficiency.



(a) (b)

Figure 12. Performances of different multicasting algorithms with Zebranet trace

7. Conclusions and Future Work

In the first nine month of the project, we have developed an on-demand situation-aware multicast (OS-multicast) algorithm, which is able to dynamically adjust multicast routing decisions based on the current network conditions. We compare its performance with those achieved by DTBR, ST-multicast, and U-multicast. Our simulation results show that OS-multicast can achieve better message delivery ratio than existing approaches with similar delay performance. When network connectivity becomes worse due to the high link unavailability (80%), OS-multicast also has better efficiency.

The current OS-multicast scheme that we implemented produces much redundant traffic. Thus, we have also designed an improved p2mp bundle delivery algorithm based on OS-multicast scheme that is expected to achieve higher efficiency at low/medium link availability. In this enhanced version, we prune some unnecessary transmissions. We are currently implementing and evaluating this improved algorithm. In the remaining months of the project, we will also design and evaluate inter-group p2mp bundle delivery approach. We also intend to study the impacts of different group sizes on the performance as well as the impact of having different traffic types for different multicast groups.

Reference

[1] J. Moy, “Multicast extensions to OSPF”, IETF RFC 1584, 1994.

[2] D. Waitzman, C. Partridge and S. Deering, “Distance vector multicast routing protocol (DVMRP)”, *IETF RFC* 1075, 1988.

[3] J. Xie, R.R. Talpade, A. Mcauley, and M.Y. Liu, “AMRoute: ad hoc multicast routing protocol,” *Mobile Networks and Applications*, Vol. 7, Issue 6, pp. 429-439, 2002.

[4] S. H. Bae, S.-J. Lee, W. Su, and M. Gerla, “The design, implementation, and performance evaluation of the on-demand multicast routing protocol in multihop wireless networks”, *IEEE Network*, pp.70–77, January 2000

[5] D.B. Johnson and D.A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, edited by T. Imielinski and H. Korth, chapter 5, pp.153-181, Kluwer Academic Publishers, 1996.

[6] W. Zhao, M. Ammar, and E. Zegura, “Multicasting in delay tolerant networks: semantic models and routing algorithms,” in the *Proceeding of Sigcomm Workshop in DTN*, August 2005.

[7] P. Hui, A. Chaintreau, J. Scott, and et al., “Pocket switched networks and human mobility in conference environments,” in the *Proceeding of Sigcomm Workshop in DTN*, August 2005.

[8] David Kotz and Kobby Essien, “Analysis of a Campus-wide Wireless Network”, Dartmouth TR2002-432.