High Throughput Routing in Hybrid Cellular and Ad-Hoc Networks

Ioannis Ioannidis, Bogdan Carbunar, Cristina Nita-Rotaru
Purdue University
West Lafayette, IN, 47907
{ioannis,carbunar,crisn}@cs.purdue.edu

Abstract

In this paper we present DST, a routing protocol for hybrid networks that is scalable with the network size and achieves high throughput by taking advantage of multiple channels. DST maintains a close to optimal spanning tree of the network by using distributed topology trees. DST is fully dynamic and generates only $O(\log n)$ messages per update operation. We show experimentally that DST scales well with the network size, making it ideal for the metropolitan environment hybrid networks are expected to operate in.

1 Introduction

The past decade has witnessed rapid developments in wireless communications, from wireless cellular telephony to wireless LANs and PANs. Wireless network cards have become affordable and wireless connections have become fast enough to attract users of traditional wired communication. Current 3G implementations, e.g. of W-CDMA, provide downlink rates of up to 380Kbps, promising in the near future 2.0Mbps (2.4Mbps for cdma2000 1xEV-DO). However, the achievable rate drops significantly as the client moves away from the base station, due to path loss via distance attenuation. Furthermore, the transmission rate can be extremely erratic, making the network unreliable. While Wi-Fi hotspots are already being used to complement the coverage of cellular networks, an architecture consisting of dual, cellular and Wi-Fi equipped devices, simultaneously operating in cellular and ad-hoc mode, has been proposed in [15] to improve the downlink rates of cellular clients. The model replaces direct cellular connections with freshly established paths of relayers whose cellular rates improve upon the rates of the cellular clients. Since wireless LANs offer high throughput (IEEE 802.11b [2] offers up to 11Mbps), albeit in a range of less than 200m, using a web of multihop paths can considerably increase the throughput from the base station to the devices in its cell without requiring modifications in the infrastructure. In addition, Wi-Fi standards offer multiple, non-overlapping channels.

A simple solution to the problem of multihop path discovery in a hybrid network is UCAN [15]. An initiator discovers a path to the base station with a breadth-first search of the network. The protocols presented in [15] have several disadvantages: 1) flooding the network every time a path is needed can cause severe congestion; 2) when multiple hosts try to find a path to the base station, hosts that have a good downlink rate will be congested as they will be on many paths; 3) the effects of interference are ignored.

In this paper we propose a dynamic spanning tree based routing algorithm, DST, that addresses the above problems. DST is based on a spanning tree that lazily converges to a maximum spanning tree [1]. Structuring the routing information as a spanning tree allows DST to generate only $\mathtt{O}(\log \mathtt{n})$ traffic for each routing request, instead of $\mathtt{O}(\mathtt{n})$ when flooding is used ($\mathtt{n}$ is the number of nodes in the network). This bound is achieved by using *topology trees* [7], an example of link-cut trees [20]. DST has two layers: one operates on the spanning tree, by issuing queries and update requests, the other implements these operations on the topology tree. Finally, DST exploits multiple channels to avoid interference and obtain increased throughput. Our simulations show that the achieved throughput is consistently over 80% of the optimal.

Roadmap: Section 2 presents our network assumptions. Section 3 describes our protocol, while Section 4 compares its performance with the optimal achievable throughput. Section 5 overviews related work, and Section 6 presents our conclusions.

2 Network Model

We consider a hybrid wireless network consisting of mobile hosts situated inside the coverage area of a cellular base station. Each host is equipped with a dual cellular and Wi-Fi network card. The base station can support simultaneous

[1] A maximum spanning tree provides the optimal routing for the next flow from a host to the base station.

transmissions with all the hosts in its coverage area and a host can support simultaneous cellular and ad-hoc communications. The mobile hosts can communicate with each other via multiple channels using the Wi-Fi card. All communication links are bidirectional. The network is modeled as a graph, where the mobile hosts and the base station are nodes and links denote Wi-Fi or cellular connectivity. Each link e has a constant weight w(e) equal to its capacity. We assume that this capacity can be estimated (for example, hosts periodically ping their neighborhood and measure the response time). The interference generated by a link transmission is modeled as the set of hosts situated in the transmission range of the endpoints of the link.

3 DST Algorithm

3.1 Interference and Aggregate Throughput

Consider a hybrid network where no host needs a multihop path to the base station (see Figure 1(a)). All links are available to their full capacity. If a host A needs the best available path to the base station BS and there is an optimal path discovery protocol, A can find this path and establish a flow to BS. Let this path be A, B, C, D and the capacity of the links be 11Mbps for (B, A), 5.5Mbps for (C, B), 11Mbps for (D, C) and 2.1Mbps for (BS, D). As the transmission between BS and D does not interfere with the ad-hoc transmissions, the aggregate throughput of the path is the minimum between the capacity of (BS, D) and the aggregate throughput of the ad-hoc path between D and A. Moreover, since each host is equipped with a single transceiver, the transmissions on (B, A) and (C, B) and also the transmissions on (C, B) and (D, C) cannot proceed simultaneously. Note also that the transmission between D and C interferes at C with the transmission between B and A. Thus, the aggregate throughput of the ad-hoc path between D and A is only a fraction of the capacity of the bottleneck link, (C, B).

In the example in Figure 1 (b), the capacity of the path is 1.8Mbps. This means that A can receive data on this path at a rate decided by the capacity of the (C, B) link. This leaves links (BS, D), (D, C) and (B, A) with a residual capacity. We take a conservative approach and block any transmissions on these links for the duration of the flow introduced by A. In addition, transmissions on links of hosts adjacent to the flow path also interfere with the flow. For example, a transmission on link (G, F) interferes at F with the transmission on link (B, A) and a transmission on (F, E) interferes at B with a transmission on (C, B). We conservatively model the interference introduced on links of hosts adjacent to the flow path by blocking transmissions on them for the duration of the flow. As a result, after each time a flow is added or removed, we can deduce the state of the residual network from the physical state of the links and the sequence of flow additions and deletions.

3.2 Multiple Channels

Our approach to model the residual capacity of links is conservative and may reduce the network throughput. To address this problem, we take advantage of the multichannel capabilities of the 802.11b [2] and 802.11a [1] standards. That is, interfering transmissions can be scheduled to occur simultaneously as long as they do not use overlapping frequencies. While 802.11b offers 3 non-overlapping channels, 802.11a provides 12 orthogonal channels.

Selecting the transmission channel for a new flow can be done by a simple traversal of the path. For each traversed link, DST reserves the first locally available channel. Then, it contacts all the hosts whose potential transmissions interfere with the link (hosts adjacent to the link's endpoints) and reserve the chosen channel. If a link of an interfering host is left without any available channels, its residual capacity becomes 0. This process can be performed using a dedicated control channel on which all idle hosts listen. Moreover, interfering hosts are notified of the reserved channel also *on* the reserved channel, in order to discover existing, potentially interfering, communications taking place on the same channel. Figure 1(c) shows an example of this approach, where the ad-hoc links supporting the newly added flow of A reserve channel 1 and subsequently, channel 1 becomes unavailable for transmissions on links interfering with the flow. However, the available capacities of the interfering links are left unaltered.

3.3 Spanning Trees

A routing protocol that maintains the optimal path for each host in the residual network can discover such paths by only keeping the parent for each host. In our example, the parent of A is B, the parent of B is C, the parent of C is D and the parent of D is the base station. After A adds a flow, the parent information might have to change to reflect the decrease in the capacity of the path links. At all times, the routing information constitutes a spanning tree rooted at the base station. Using the cycle property [21] of maximum spanning trees, it can be proved that the maximum[2] spanning tree of a residual network provides the optimal routing information. The path from each host to the root in the spanning tree has the maximum minimum link possible. Since the aggregate throughput of a path is a fraction of the capacity of the bottleneck link of the path, this is guaranteed to maximize the capacity of the entire path.

Given a maximum spanning tree, a host can schedule the next flow by sending a request to its parent, which in turn will forward the request to its parent, until the base station is reached. After adding (or deleting) a flow, the entire network may need to be contacted to derive the new maximum

[2]We use maximum spanning trees because we want to maximize the capacity of a path. In case the link weights denote cost, minimum spanning trees should be used.

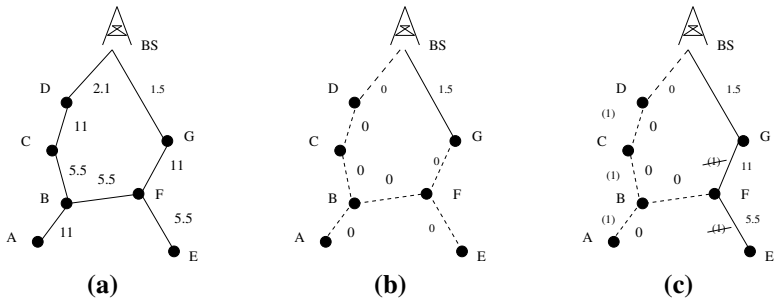

Figure 1. (a) Example of a hybrid network, where labels on the right-hand side of links represent link residual capacities. (b) Residual network of (a) after A adds a flow on links (BS, D), (D, C), (C, B) and (B, A). Due to interference, not only links adjacent to this path but also links of hosts adjacent to this path are blocked. (c) Same scenario as in (b), using the multi-channel capability of Wi-Fi standards. Labels on the left-hand side of links represent channel assignments. Links of hosts adjacent to the flow path, i.e. (G, F) retain their capacity, but cannot use the channel chosen by A's flow due to interference.

spanning tree, which is asymptotically not better than flooding each time a flow needs to be scheduled. To address this problem DST does not change the routing information to reflect the maximum spanning tree each time a flow is added or removed. Instead, each time a host requests a path, its parent pointer is set to the neighbor that has the optimal path to the base station, according to the existing information. The changes are confined in the neighborhood of a host and while queries about the state of the paths of the neighbors have to contact hosts outside the neighborhood, they can be completed much faster. If the network becomes static, the routing information will eventually converge to the optimal, even with this localized updating policy.

This approach requires only $\mathtt{O}(\log \mathtt{n})$ time and messages for each operation to maintain a dynamic tree. In the worst case, DST can be arbitrarily far from the optimal, but our experiments indicate that, on average, the throughput achieved is close (above 80%) to the optimal.

3.4 Maintaining a Dynamic Tree

The DST routing protocol can be split into two layers communicating through a well-defined interface. The top layer is responsible for the maintenance of the distributed spanning tree and issues the following string of operations on the spanning tree:

- $\mathtt{Link}(\mathtt{v}, \mathtt{u}, \mathtt{w})$: Merge the tree rooted at node v with the tree of node u by making u the parent of v. The weight of the new link is w.
- $\mathtt{Cut}(\mathtt{v})$: Split a tree into two by removing the link of node v to its parent.
- $\mathtt{Mincost}(\mathtt{v})$: Return the minimum weight cost edge on the path from node v to the root of the tree.
- $\mathtt{Update}(\mathtt{v}, \mathtt{w})$: Add w to all edges on the path from v to the root of its tree.

The second layer is responsible for efficiently completing the above set of operations. We implement this layer using link-cut trees since they can complete these operations in $\mathtt{O}(\log \mathtt{n})$ time, where n is the number of hosts. This scalability property is important as it translates in $\mathtt{O}(\log \mathtt{n})$ messages when implemented distributively (see [10]). Furthermore, when a new host enters the network and has to query its neighbors on the capacity of their paths, the parallel time complexity is $\mathtt{O}(\log \mathtt{n})$. For implementation purposes we have chosen to use topology trees for this layer. A topology tree is a relatively simple link-cut tree and it has a natural distributed implementation (see [10]). We note that any dynamic tree can be used for this layer, as long as it does not modify the structure of the spanning tree. The root of the tree is fixed to be the base station and the links are oriented towards it.

3.5 Refresh Rate

In dynamic networks, routes become stale quickly. A parent pointer indicating the best available path should be reevaluated at constant intervals to adapt to topology changes. The exact refresh rate depends on how dynamic the hybrid network is and how much traffic per operation is allowed. There are two possible strategies on reassessing the parent pointer. The first strategy is more aggressive, but generates more traffic: a node can cut its parent every k seconds and probe all its neighbors. The traffic generated is $\mathtt{O}(\mathtt{d} \log \mathtt{n})$, where d is the number of neighbors, but the parent pointer is as close to the optimal as possible with the available information. The second strategy is less expensive: every node queries the parent every k seconds and cuts it only if the rate falls below a threshold.

To keep the number of messages per time unit at a scalable level, k must be adapted to the size of the network. As the network becomes more dense, the refresh rate should

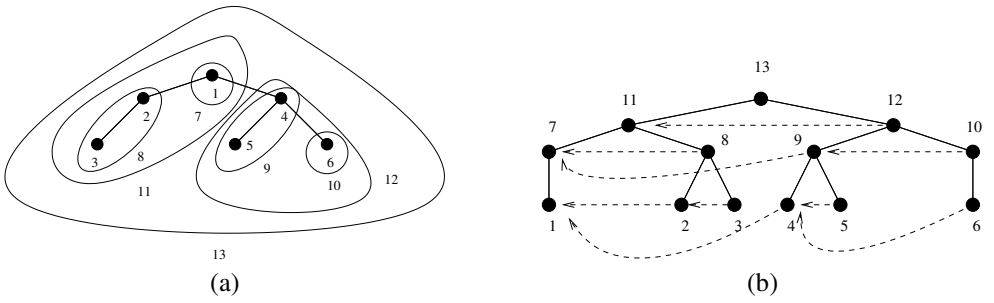

Figure 2. (a) Example of a restricted partition and (b) the resulting topology tree.

drop. If $\mathtt{k} = \Theta(\mathtt{d})$, the traffic generated every k seconds is $\mathtt{O}(\mathtt{k} \log \mathtt{n})$. The exact constants depend on the specifics of the network, but the conclusion is that for dense networks, refreshing should be done more sparingly. This appears to be contrary to the scalability of DST, as the spanning tree maintained should be farther from the optimal. However, dense hybrid networks are more robust, and even if a link disappears, there is a high probability that an alternate equivalent link will be present. A parent pointer can be used with relative confidence for the short time until the next reassessment. In Section 4, we use an alternate approach, scaling $\mathtt{k}$ with $\log \mathtt{n}$, which generates $\mathtt{O}(\mathtt{d})$ messages per time unit for each host. Even at this rate, the scalability of the network is not affected, as probing one's neighbors with a heartbeat broadcast, an operation performed by almost all wireless interfaces, generates $\mathtt{d}$ replies. Experiments confirm that dropping the refresh rate according to $\log \mathtt{n}$ does not affect the performance of DST.

3.6 Topology Trees

In this section we provide a brief overview of topology trees. For a detailed presentation, including topology trees supporting a minimum spanning tree algorithm see [8].

Topology and, in general, link-cut trees can maintain dynamic trees of n nodes with $\mathtt{O}(\log \mathtt{n})$ operations per tree update. In addition, operations that require an aggregate of all the nodes on a path from the root to a leaf, like `Mincost` or `Update`, can be completed in $\mathtt{O}(\log \mathtt{n})$ time. In fact, all operations of the interface to the second layer have $\mathtt{O}(\log \mathtt{n})$ time and message complexity.

Topology trees are derived from a restricted partition of a tree. For an example of a restricted partition see Figure 2(a). To avoid confusion, we refer to nodes of the topology tree as clusters. The leaves of the topology tree are clusters of single spanning tree nodes. A cluster of a higher level is made up of one or two clusters of a lower level. The rules by which clusters are paired are described in [7]. The intuition is that for every pair of clusters that combines for a cluster of a higher level, another cluster is made up of a single lower level cluster to act as a buffer when there is an update in the structure of the topology tree. These buffers are clusters that have two children. In Figure 2(a), cluster 7 consists of only cluster 1 for this reason. It can be shown that the height of a topology tree is $\mathtt{O}(\log \mathtt{n})$ [7]. The resulting topology tree is shown in Figure 2(b). The solid edges indicate the relationship between clusters of consecutive levels. The dashed arrows represent the structure of the tree formed by clusters of the same level. Each such tree is called the induced tree of the specific level. The lowest level induced tree is the actual spanning tree.

Besides the adjacency information, each cluster stores three more fields, `Δcost`, `nodemin` and `minvert`. To calculate the weight of an edge from node `v` to its parent, the topology tree must be traversed from the leaf cluster corresponding to `v` up to the root of the topology tree and sum the `Δcost` fields of the accessed clusters. The `nodemin` and `minvert` fields of a cluster `c` store information about the minimum cost edge in the spanning subtree induced by the leaves of the subtree rooted at `c`. The rules by which these fields are calculated and how the adjacency information should change after an update of the spanning tree can be found in [7] and in more detail in [8]. We note that each of the $\mathtt{O}(\log \mathtt{n})$ steps is a local operation that only needs information from the parent and sibling cluster to complete. This is important not only because it leads to $\mathtt{O}(\log \mathtt{n})$ total time for each of the operations of the topology tree, but also because it facilitates the distributed implementation of topology trees (see [10]).

4 Simulation Results

In this section we present an experimental analysis of the throughput performance of DST with regard to the optimum throughput achievable when a centralized knowledge Bellman-Ford algorithm is executed. We model the ad-hoc network using the unit disk graph model, using the Agere Short Antenna PC Card Extended specification. We use the ARF [12] mechanism to establish the transmission rate of the communication channel between two mobile hosts.

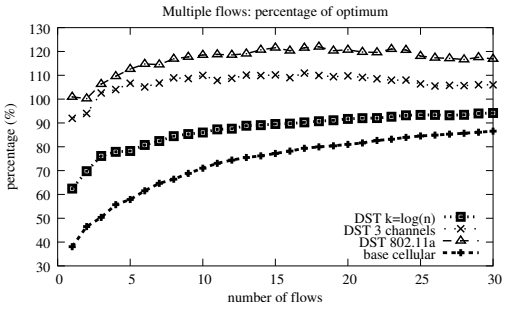

Figure 3. The throughput of DST and the basic cellular rate as percentage of the throughput achieved by Bellman-Ford when the number of flows grows from 1 to 30 for a network of 300 hosts.

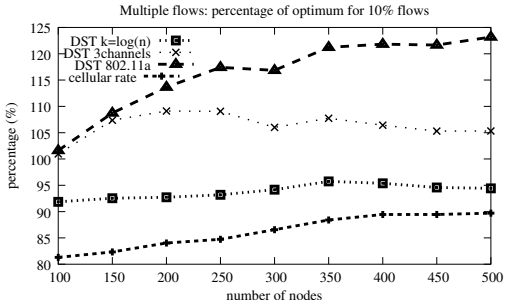

Figure 4. The throughput of DST and the basic cellular rate as percentage of the throughput achieved by Bellman-Ford for networks of 50 to 500 hosts, when 10% of the hosts concurrently hold a flow.

We use only the top two transmission rates, of 11Mbps for distances under 160m and of 5.5Mbps for distances under 270m. The hosts are deployed randomly in a square of area $2830 \times 2830\mathrm{m}^2$ and we use a modified random waypoint model [11] to simulate their movements. We have overestimated the effects of ad-hoc link interference, by blocking any transmission involving hosts adjacent to the link.

We use the UCAN [15] approach to model the dependency between the cellular link rates of hosts and their distance from the cellular base station. The base station is positioned at the center of the $2830 \times 2830\mathrm{m}^2$ deployment square and its cellular transmission range is 1920m. According to this model, each host inside the square is covered by the cellular transmission range of the base station.

We model the optimal throughput to be the one achieved by the Bellman-Ford algorithm. Instead of computing the shortest path, we compute the maximum throughput path between the base station and all the mobile hosts it covers. In the case of multiple concurrent flows the optimal for $\mathtt{n}$ flows is computed by running Bellman-Ford on the residual network obtained after removing the bandwidth consumed and the interference introduced by the first $\mathtt{n}-1$ flows.

We perform each experiment by choosing 5 different initial network configurations. For each such configuration the experiment is run for 100 seconds. Thus, each point on the plots is an average over 500 measurements.

In addition to the performance of DST when a single transmission channel is used, we also experiment with multi-channel transmissions. First, we use the 3 non-overlapping channels of 802.11b. Secondly, we switch to the 802.11a specification, providing 12 orthogonal channels and transmission rates of up to 54Mbps.

In the first experiment we randomly deploy 300 hosts that continuously move with a maximum speed of 9m/s. We increase the number of simultaneously supported flows from 1 to 30. Figure 3 shows the performance of DST relative to the optimal total throughput, achieved when all the client hosts run the distributed Bellman-Ford algorithm to find the best downlink path. The performance of DST increases to achieve more than 90% of Bellman-Ford. In addition, Figure 3 also shows the performance of DST when using the multi-channel capabilities of 802.11b and 802.11a, compared to the flat cellular rate of the client hosts. Using the non-overlapping channels of 802.11b brings an increase of around 10% over the optimum achievable in the case of a single channel, while 802.11a has a 20% increase. Note that even when 30 out of the 300 hosts concurrently support a flow, by using the 3 channels of 802.11b, DST achieves a per-flow increase of 200kbps over the basic cellular rate.

The second simulation experiments with increasing concentrations of mobile hosts and of concurrent flows. In the same square area of $2830 \times 2830\mathrm{m}^2$, we place between 100 and 500 hosts, while also increasing the number of hosts concurrently supporting flows to be 10% of the total number of hosts. Figure 4 shows that DST performs very close to the Bellman-Ford, always higher than 90%. Using the 3 channels of 802.11b brings a 10% increase over the single channel variant, whereas using 802.11a achieves a throughput increase of up to 25% over the optimum Bellman-Ford. While the basic cellular rate remains constant, as the network becomes congested, the throughput achieved by DST per flow gracefully decreases when using the single channel or the multi-channel capabilities of 802.11b. However, when using DST in conjunction with 802.11a, the throughput per-flow saturates at 1050Kbps. This is because the usage of multiple non-overlapping channels alleviates the effects of the congestion generated at the hosts situated in the vicinity of the base station, by allowing concurrent transmissions on their adjacent hosts. Using DST with the 3-channel variant of 802.11b, brings an increase of between 150 and 200kbps over the cellular throughput. When using DST in conjunction with 802.11a, the throughput increase

is more substantial, between 200 and 300kbps.

5 Related Work

Several architectures for ad hoc wireless networks were proposed in [5], [17], [11]. Most of the efforts to integrate infrastructure-based network models with ad hoc components, assume single-interface devices. In [3], GSM terminals are used to relay information to other terminals to improve coverage. In Opportunity Driven Multiple Access [19], transmission power is conserved by relaying traffic from a CDMA host to the base station through multiple, short hops. In [22], some channels are reserved for forwarding when the fixed channels become congested. In [4], a generic wireless network is considered, where hosts contact a mobile base station for access outside their cell, using only one interface. In [9], a hybrid network using the IEEE 802.11 architecture with both DCF and PCF modes is examined, using only one wireless interface. In [14], multihop paths are used to decrease the number of base stations by increasing their coverage. The overall capacity increases only when two communicating hosts are in the same cell.

Double-interface architectures increase the overall capacity by using short-range, high-bandwidth, ephemeral channels to relay traffic and a long-range, low-bandwidth, permanent channel to complete operations like routing and data integrity confirmation. This architecture has been examined in [15]. In [6], traffic is diverted to neighboring cells to increase throughput. The use of dedicated, stationary relays increases the cost of their solution and limits its utility. In [18], wireless nodes in mesh networks are equipped with two Wi-Fi network interface cards. Centralized channel assignment algorithms and a routing protocol, designed to increase the aggregate throughput in the presence of interference, are presented. A study of local area hybrid networks is presented in [13], while a comprehensive presentation of a hybrid network can be found in [16].

6 Concluding Remarks

In this paper we present DST, a scalable routing protocol for hybrid networks. DST achieves its scalability by maintaining a spanning tree of the network that eventually converges to the optimal. By using topology trees to maintain the dynamic spanning tree, DST ensures that each operation can be completed in $O(\log n)$ time, generating $O(\log n)$ messages. In addition, DST takes advantage of multiple channels to avoid interference and achieve high throughput. The experiments presented show that the achieved throughput is consistently over 80% of the optimal.

References

[1] *IEEE Std 802.11a-1999*. 1999. http://standards.ieee.org/.

[2] *IEEE Std 802.11b-1999*. 1999. http://standards.ieee.org/.

[3] G. Aggelou and R. Tafazolli. On the relaying capacity of next-generation gsm cellulat networks. February 2001.

[4] I. Akyildiz, W. Yen, and B.Yener. A new hierarchical routing protocol for dynamic multihop wireless networks. In *Proceedings of IEEE INFOCOM*, 1997.

[5] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th MOBICOM*, pages 85–97. ACM Press, 1998.

[6] S. De, O. Tonguz, H. Wu, and C. Qiao. Integrated cellular and ad hoc relay (icar) systems: Pushing the performance limits of conventional wireless networks. In *Proceedings of HICSS-37*, 2002.

[7] G. N. Frederickson. A data structure for dynamically maintaining rooted trees. In *SODA*, 1993.

[8] G. N. Frederickson. Ambivalent data structures for dynamic 2-edge-connectivity and k smallest spanning trees. *SIAM J. of Comp.*, 26(2):484–538, 1997.

[9] H.-Y. Hsieh and R. Sivakumar. On using the ad-hoc network model in wireless packet data networks. In *Proceedings of ACM MOBIHOC*, 2002.

[10] I. Ioannidis, B. Carbunar, and C. Nita-Rotaru. High throughput routing in hybrid cellular and ad-hoc networks. Technical report, Purdue University, 2005.

[11] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing, volume 353, pages 153-181. Kluwer Academic Publishers*, 1996.

[12] A. Kamerman and L. Monteban. Wavelan-ii: A high performance wireless lan for the unlicensed band. *Bell Labs Technical Journal*, 1997.

[13] S. Lee, S. Banerjee, and B. Bhattacharjee. The case for a multi-hop wireless local area network. In *Proceedings of IEEE INFOCOM*, 2004.

[14] Y.-D. Lin and Y.-C. Hsu. Mulithop cellular: A new architecture for wireless communications. In *Proceedings of IEEE INFOCOM*, 2000.

[15] H. Luo, R. Ramjee, P. Sinha, L. E. Li, and S. Lu. Ucan: a unified cellular and ad-hoc network architecture. In *Proceedings of the 9th MOBICOM*, pages 353–367. ACM Press, 2003.

[16] M. Miller, W. List, and N. Vaidya. A hybrid network implementation to extend infrastructure reach. Technical report, University of Illinois at Urbana Champaign, January 2003.

[17] C. Perkins and E. Royer. Ad-hoc on demand distance vector routing, 1999.

[18] A. Raniwala, K. Gopalan, and T. cker Chiueh. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8(2):50–65, 2004.

[19] T. Rousse, I. Band, and S. McLaughlin. Capacity and power investigation of opportunity driven multiple access (odma) networks in tdd-cdma based systems. 2002.

[20] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *J. Comput. Sci.*, 26:362–391, 1983.

[21] R. E. Tarjan. *Data structures and network algorithms*. Society for Industrial and Applied Mathematics, 1983.

[22] X. Wu, S.-H. Chan, and B. Mukherjee. Madf: A novel approach to add an ad-hoc overlay on a fixed cellular infrastructure. In *Proceedings of IEEE WCWN*, 2000.