Mobility Control and Its Applications in Mobile Ad Hoc Networks

Jie Wu and Fei Dai, Florida Atlantic University

Abstract

Most existing localized protocols in mobile ad hoc networks, such as data communication and topology control, use local information for decentralized decision among nodes to achieve certain global objectives. These objectives include determining a small connected dominating set for virtual backbone and topology control by adjusting transmission ranges of nodes. Because of asynchronous sampling of local information at each node, delays at different stages of protocol handshake, and movement of mobile nodes, the local view captured at different nodes may be inconsistent and/or outdated and may not reflect the actual network situation. The former may cause "bad" decisions that fail to keep the given global constraint such as global domination and connectivity, and the latter may incur "broken" links, which in turn will ultimately cause the failure of the constraint. In this article we review some techniques that handle inconsistent and outdated local views. These techniques are illustrated using several well-known protocols in data communication and topology control.

In mobile ad hoc networks (MANETs), all nodes cooperate to achieve a global task, such as data gathering, communication, or area monitoring. MANETs are characterized by unit disk graphs where two nodes are connected only if their geographical distance is within a given transmission range (as shown in Fig. 1a where the transmission range is 2.5). To design protocols that are simple and quick to converge, many protocols in MANETs rely on *localized algorithms*. The localized algorithm running at each node makes its local decision based on local information within one or two hops. Collectively, nodes running the localized algorithm achieve some desirable global objectives. Two widely used applications of the localized algorithm are:

- Determining a connected dominating set (CDS) for efficient routing [1–4]
- Selecting an appropriate transmission range of each node for topology control [5–8]

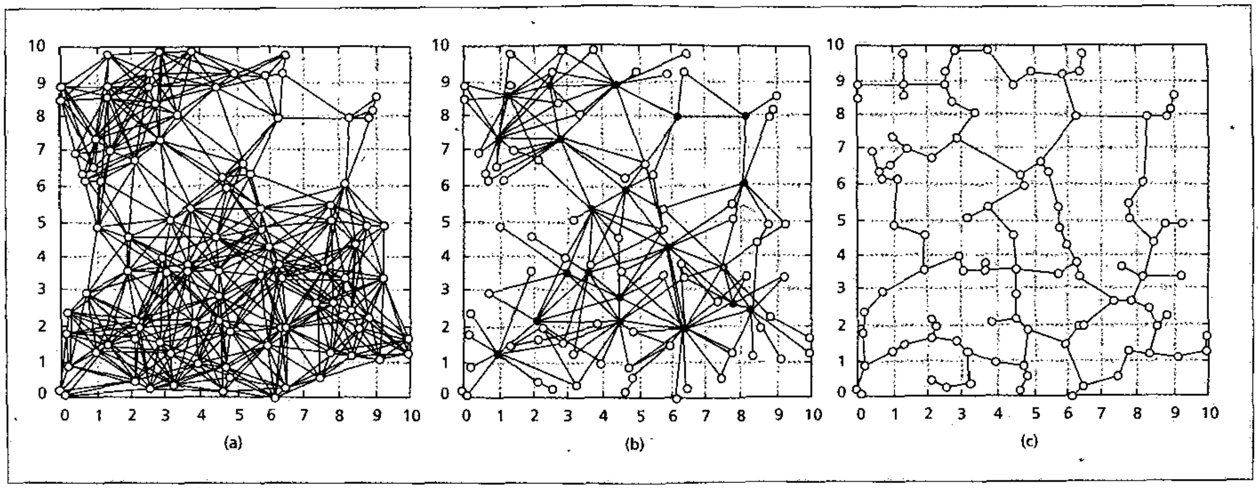
A CDS is a subset such that each node in the system is either in the set or the neighbor of a node in the set. The CDS has been used widely to support the notion of a *virtual backbone* in MANETs. Another application of CDS is in broadcasting, where nodes and only nodes in the CDS forward the broadcast message to reduce message collision. However, finding a minimum CDS is NP-complete. Most practical approaches in MANETs use localized algorithms to find a small CDS. In a typical localized CDS protocol, each node uses local information to determine its status, *dominator* or *dominatee*. Figure 1b shows a CDS constructed via a localized algorithm [1]. In this diagram the connections between dominatees are not shown. As dominators (black nodes) form a backbone of the MANET, any dominatee (white node) can switch to sleep mode for energy saving without causing network partition. Most localized CDS algorithms rely on two-hop information of the current node v, which includes information of v's neighbors and neighbors of v's neighbors.

In MANETs, in order to reduce energy consumption and signal interference, it is important to select an appropriate transmission power for each node, also called *topology control*, while still satisfying certain global constraints, including connectivity and other reliability and throughput related measures. In localized topology control, each node uses local information to select a subset of physical neighbors, called *logical neighbors*, and its transmission range is reduced to reaching only as far as the farthest logical neighbor. Figure 1c shows the result of a localized topology control algorithm [5], where both the average number of neighbors and transmission range are reduced significantly, while the network is still connected. Localized topology control algorithms usually rely on one-hop location information of the current node v, which includes information on v's neighbors and their location information. Some algorithms require less information where distance or angle of arrival information of neighbors is sufficient.

Compared with their centralized counterparts, localized algorithms are lightweight, fast to converge, and resilient to node movement. However, without a mobility control mechanism, global domination and connectivity may still be compromised by node movement. In most existing localized algorithms, each node in a MANET emits a periodic Hello message to advertise its presence and position (if needed) at a fixed interval Δ. Hello intervals at different nodes are asynchronous to reduce message collision. Each node uses received Hello messages as samples to construct a local view

---

This work was supported in part by NSF grants CCR 0329741, ANI 0073736, and EIA 0130806.



■ Figure 1. *Virtual networks constructed via localized algorithms: a) the original network; b) the connected dominating set; c) topology control. The original MANET has 100 nodes and a transmission range of 2.5.*

of its one- or two-hop neighborhood. In a MANET with mobile nodes, the limited sample frequency, asynchronous Hello intervals, and delays at different stages of protocol handshake will cause a mismatch between the *virtual network* constructed from the collection of local views sampled at different nodes and the *actual network*. This mismatch will cause the *link availability* issue, where a neighbor in a virtual network is no longer a neighbor in the actual network, because the virtual network is constructed from *outdated* information. Therefore, special mechanisms are needed to address the following issue:

- *Delay and mobility management*: How protocols deal with imprecise neighborhood information caused by node mobility and various delays introduced at different stages of protocol handshake

One solution in [9], discussed later in detail, uses two transmission ranges to address the link availability issue. First, a transmission range r is determined based on the selected protocol. This transmission range is either the same as the Hello message range r', as in the CDS protocol or shorter than r', as in the topology control protocol. The actual transmission uses a long transmission range set to $r + l$. The difference, l, between these two ranges is based on update frequency and speed of node movement.

The mismatch between the virtual network and actual network will cause a more serious problem: *inconsistent local* views. Inconsistent local views may cause "bad" decisions that fail to keep global constraints such as global domination and connectivity. Again, special mechanisms are needed to address the following issue:

- *Node synchronization and consistent local view*: How each node knows when to sample its local view; how each node collects and uses local information in a consistent way

We examine two different approaches that address the consistency issue: enforcing consistent views and making conservative decisions. The first approach was initially proposed in [10] to construct consistent one-hop information for topology control. This approach can also be extended to support the construction of two-hop information. The second approach was originally proposed in [9] for CDS formation, but the same principle can be used in topology control.

The main objective of this article is to expose to the reader the challenging issue related to mobility control. Through discussion on the effects of mobile nodes on several important protocols, we present some problems, provide possible solutions, and discuss several open issues. By this we hope to stimulate more research in this important area.

The Link Availability Issue and Its Solution

In MANETs, because of asynchronous Hello messages and various protocol handshake delays, neighborhood information and/or position used in decision making may be outdated. For example, a previously sampled neighbor can move out of transmission range during actual transmission. In order to apply existing protocols without having to redesign them, the notion of *buffer zone* is used in [9], where two circles with radii r and $r + l$ are used. r corresponds to the transmission range determined by a selected protocol, whereas $r + l$ corresponds to the actual transmission range used. $l = d \times 2t$ is defined as a buffered range depending on the moving speed t of mobile nodes and the maximum time delay d. To simplify the discussion, both Hello intervals and moving patterns/speeds are homogeneous; hence, l is uniform for each node.

The above requirement of buffered range guarantees link availability in the worst case situation. However, probabilistic study in [9] reveals that the worst case rarely happens. In MANETs with very high moving speed (t), it is either impossible or too expensive to use such a large l. Both probabilistic analysis and simulation results in [9] show that link availability is preserved in most cases with a buffered range much smaller than $d \times 2t$. There is a wide range of potential trade-offs between efficiency and connectivity.

Specifically, suppose r' is the normal Hello message range. A typical CDS protocol works as follows:

- Select $r = r'$ for neighborhood information exchange.
- Apply the selected localized CDS protocol to determine the status of each node.
- Use $r + l$ for each dominator in the actual transmission.

The second step of the above process varies from protocol to protocol. Here we use Wu and Li's marking process and rules 1 and 2 [4] to illustrate:

At each node u:

- Marking process: u is marked true (i.e., becomes a dominator) if there are two unconnected neighbors.
- Rule 1: u is unmarked (i.e., becomes a dominatee) if its neighbor set is covered by another node with a higher id.
- Rule 2: u is unmarked if its neighbor set is covered jointly by two connected nodes with higher ids.

In rules 1 and 2, we say u's neighbor set, $N(u)$, is covered by one or two covering nodes, if every node w in $N(u)$ is either a covering node or a neighbor of a covering node. Figure 2 shows a sample ad hoc network with nine nodes. Node r is unmarked by the marking process because its neighbors u and z are directly connected. Node w is unmarked by rule 1 because its neighbor set is covered by node x. Here node id x is higher than w according to alphabetical order. Node u is unmarked by rule 2 because its neighbor set is covered by two connected nodes, x and z. Clearly, the marked nodes v, x, and z form a CDS of the sample network.

Originally, rules 1 and 2 use only marked nodes as covering nodes, and involve overhead in communicating dominating set status. Stojmenovic *et al.* [3] showed that unmarked nodes can also be covering nodes, and there is no need to exchange dominating set status. Dai and Wu [1] proposed a generalized rule (called rule k) to construct a smaller CDS. Based on the generalized rule, u is unmarked if its neighbor set is covered by several connected nodes with higher ids. The number of covering nodes is allowed to be more than two.

When the network is static or local views are consistent (say, all nodes see only a solid line) in Fig. 3, both nodes u and v are marked after the marking process. u will be unmarked using rule 1.

A typical topology control protocol works as follows:

- Select r' to collect neighborhood information.
- Apply a selected localized topology control protocol to select $r(u)$, $r(u) \leq r'$, for node u to cover its farthest logical neighbor.
- Use $r(u) + l$ for actual transmission.

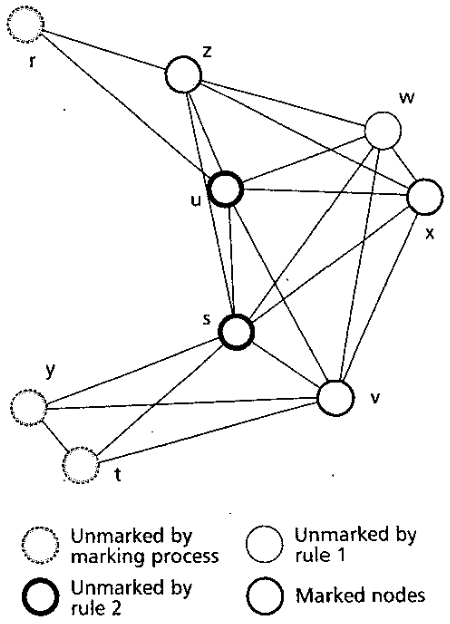
We use Li, Hou, and Sha's topology control algorithm based on a local minimum spanning tree (MST) [5] to illustrate the second of the above process.

At each node u:

- Build a local MST using Prim's algorithm based on one-hop location information. The resultant MST covers all one-hop neighbors of u.
- Select neighbors in MST as logical neighbors of u.
- Set the transmission range of u to the distance to the farthest logical neighbor.

When the network is static or local views are consistent (say, all nodes see only solid line) in Fig. 4a, the MST includes two links (u,v) and (w,v). Node u has one logical neighbor v and sets its range to 4. Node w has one logical neighbor v and sets its range to 5. Node v has two logical neighbors u and w, and sets its range to 5 to reach the farthest node w.

When the network contains mobile nodes, such as node w in Fig. 4, the transmission range of each node is increased to maintain link availability. For example, if it is known that the maximum relative movement between two nodes during one Hello interval is $l = 2$, the actual transmission range of nodes u, v, and w are adjusted to 8, 7, and 8, respectively. Therefore, link (v,w) is still available even if node w moves upward, and the distance between v and w becomes 6. It is also observed in [9] that the buffer zone width $l = 2$ is conservative and not always necessary. The probability is high that all links can be maintained with a smaller l.

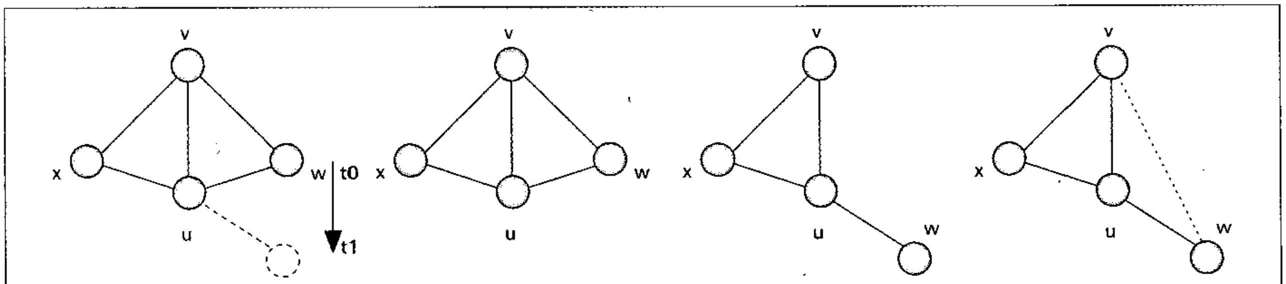


□ Figure 2. *Wu and Li's CDS algorithm. Black nodes are marked (i.e., in the CDS).*

The View Consistency Issue

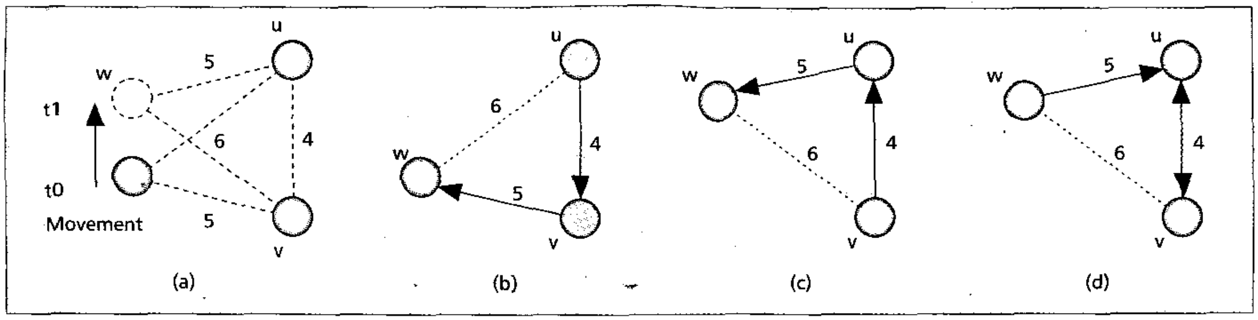
Again, we use two localized algorithms as examples to demonstrate how inconsistent local views cause "bad" decisions in MANETs: Wu and Li's marking process [4] for CDS construction, and Li, Hou, and Sha's topology control algorithm based on local MST [5].

In the CDS construction example (Fig. 3) we assume that node w moves southward. Link (v,w) exists at time t_0 and is broken at time t_1. We also assume t_0 and t_1 belong to two intervals. Since link (v,w) is two hops away from node u, when node u decides its status, it uses the outdated information (lagging by one interval) that link (v,w) still exists. The local view u is shown in Fig. 3b. Based on rule 1, node u is unmarked because its neighbor set is covered by node v. However, when node v decides its status, it has the fresh informa-



(a) w's position at t0 and t1
(b) u's decision as an unmarked node at t1
(c) v's decision as unmarked at t1
(d) v's conservative local view at t1

■ Figure 3. *s node* w *moves away from node* v *(a), both nodes* u *and* v *are unmarked due to inconsistent local views sampled at nodes* u *and* v *(b,c). Based on the conservative view, node* v *is still marked for a little while after it detected the broken link* (v,w) *(d). The dotted line represents a virtual link in* v*'s view.*



■ Figure 4. *Node* w *becomes unreachable from nodes* u *and* v *due to inconsistent local views sampled at nodes* u *and* v*: a)* w*'s positions at* t_0 *and* t_1*; b) LMST(*u*) built at time* t_0*; c) LMST(*v*) built at time* t_1*; d) network topology at time* t_1.

tion that link (w,v) is broken since it is adjacent to the link (Fig. 3c). Based on the marking process, the only two neighbors of v, x and u, are connected, so node v is also marked false. As a consequence, none of the nodes in the network are marked!

In the topology control example (Fig. 4), assume node u's view reflects the topology at t_0 (as shown in Fig. 4b), whereas node v's view corresponds to the topology at t_1 (Fig. 4c). This happens when the recent Hello message from w is sent at t, where $t_0 < t < t_1$. In this case, u has only one logical neighbor, v, and v has only one logical neighbor, u. Based on the protocol, a link is selected only if both end nodes select each other. As a result, only one link (u,v) exists after topology control (Fig. 4d). A network partition occurs!

In the above examples, individual nodes make "bad" decisions based on inconsistent local views. Two views are inconsistent if their common parts do not match. In the CDS example, link (v,w) exists in node u's view but not in node v's view. In the topology control example, w is closer to v in u's view but closer to u in v's view. There are two solutions to this problem:

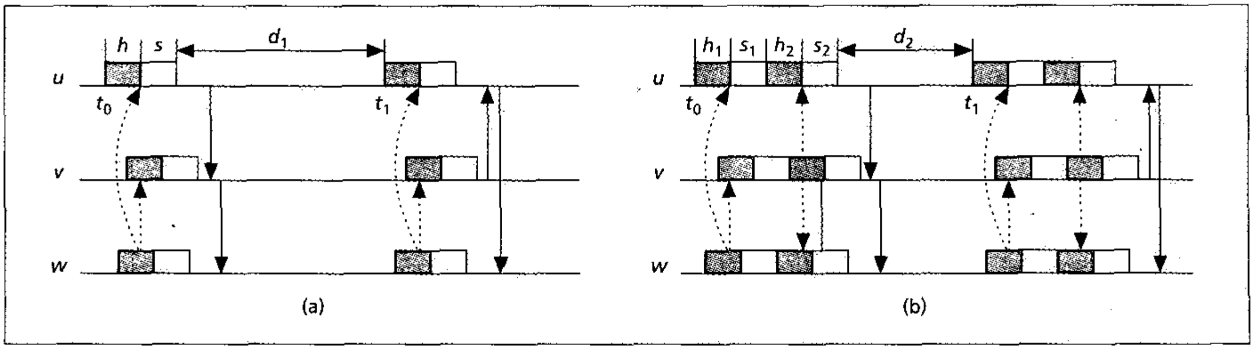
- Enforcing consistent local views
- Making conservative decisions that maintain the global property, as discussed in the next two sections

Consistent Local Views

We first consider one-hop (location) information used in topology control. Originally, each node receives Hello messages from its one-hop neighbors, and updates its local view upon the arrival of every Hello message. If all nodes have synchronized clocks, this scheme actually works. In the topology control example, if both nodes u and v make their decisions at t_0, they will agree that v is closer to w; at t_1 they will agree that u is closer. Here we omit the propagation delay and assume that a Hello message is received by all neighbors at the same time. However, it is impossible to have totally synchronized clocks in a MANET without centralized control. If u makes its decision slightly earlier than v, and w's Hello message arrives after u's decision and before v's decision, the two nodes have inconsistent views. This inconsistency cannot be avoided no matter how small the asynchrony is.

The traditional solution to this problem is to build local views only once at the beginning of each Hello interval. As shown in Fig. 5a, each Hello interval is divided into three time periods, $\Delta = h + s + d_1$. Because of asynchronous clocks, different nodes may start their Hello intervals at different times. That is, some nodes have "faster" clocks than other nodes. However, we assume the difference between two clocks is bounded by s. In the construction of consistent views, each node sends its Hello message during period h, waits for a period s, and conducts normal activities (e.g., sending data packets) in period d_1. As the h period of the "slowest" node ends before the s period of the "fastest" node, every node receives all Hello messages before the end of its s period. Local views built in the end of s are consistent. It is safe to route data packets in period d_1 based on these local views.

This scheme can be extended to build two-hop information. As shown in Fig. 5b, each Hello interval is divided into five periods, $\Delta = h_1 + s_1 + h_2 + s_2 + d_2$. Normally $h_1 = h_2$ and $s_1 = s_2$. Again, we assume the clock difference is bounded by both s_1 and s_2. Each node first advertises its 0-hop information (i.e., its id and/or location) in period h_1, builds one-hop information at the end of period s_1, and then advertises the newly constructed one-hop information in period h_2. At the end of period s_2, every node constructs its consistent local view, which is ready for use in period d_2. The drawback of this scheme is that two Hello messages are sent during each interval Δ, and the effective communication period d_2 is further reduced.



■ Figure 5. *Build consistent local views at the beginning of each Hello interval; dotted lines represent Hello messages, solid lines data packets: a) consistent one-hop information; b) consistent two-hop information.*

The traditional solution relies on the assumption that the maximal difference among local clocks, s, is predictable and $s \leq \Delta$. In a totally asynchronous system, $s = \Delta$ and the above simple approach cannot be applied. Note that even if $s < \Delta$ at a particular network, delays accumulate unless some clock synchronization protocol is applied. Although various solutions exist to adjust clock values, frequent clock synchronization is costly. When maintaining a (partially) synchronous Hello interval becomes too expensive or impossible, we propose using timestamped asynchronous Hello messages to enforce application specific consistent local views.

The basic idea is to maintain a sequence number, i_v, at each node v, and attach the sequence number to each Hello message from this node. The sequence number serves as a timestamp. Consistent local views are obtained from Hello messages with the same timestamps. This can be done by carrying a timestamp in each data packet (including control packets from a higher-level protocol). The timestamp is chosen by the originator of the data packet, and all nodes relaying this packet must determine their logical neighbors based on information of the same version (i.e., with the same timestamp). In this scheme, each node keeps several local views, each corresponding to a recently used timestamp. Similarly, several logic topologies coexist in the same network. Each logic topology corresponds to a timestamp and is connected. The logic time (i.e., the *timestamp* of the latest local view) of the originator of the data packet is used as a selector. It indicates in which logical topology this data packet is travelling. This approach can tolerate a larger *time skew* among different local views and therefore involves less synchronization overhead.

In Fig. 4, suppose the first Hello message from node w has timestamp 0, and the second has timestamp 1. When the above method is applied, two parallel logic topologies exist. The logical topology corresponding to timestamp 0 includes two bidirectional links, (u,v) and (v,w). The logic topology corresponding to timestamp 1 includes (u,v) and (u,w). When a data packet, p, is sent from u to w, the source node u selects a recent timestamp and forwards p on the corresponding logical topology. If p has timestamp 0, it is first forwarded to v. Based on v's local view with timestamp 0, w is a logical neighbor of v, and p is forwarded along the logical link (v,w). If p has timestamp 1, it is sent to w directly via logical link (u,w). In both cases, p arrives safely at its destination.

Conservative Local Views

Both solutions for enforcing consistent local views require a certain degree of internode synchronization, which introduces extra overhead. When maintaining consistent local views becomes too expensive or impossible, another approach called the *conservative local view* [10] can be applied, which makes conservative decisions based on inconsistent views. No synchronization is necessary. A conservative decision is one that maintains the global property with the penalty of lower efficiency. This means selecting more logical neighbors in a topology control algorithm, which in turn generates a larger average transmission range, and marks more nodes as dominators in a CDS formation process. We use Wu and Li's marking process as an example to illustrate the conservative approach.

In Wu and Li's marking process, a node v may be unmarked incorrectly if:

- v no longer views a node w as its neighbor
- Another node u still views w as v's neighbor and unmarks itself based on this view

As the broken link (v,w) is first detected by v and then propagated to u via periodic Hello messages, local views of nodes u and v are inconsistent for a short period. During that period, u and v may be unmarked simultaneously, and the CDS is temporarily compromised. In order to prevent the above conditions from happening together, each node must use a conservative local view, instead of its most recent local view, to make conservative decisions. In this case, the conservative local view $View_c(v)$ of node v is constructed from k most recent local views $View_1(v)$, $View_2(v)$, ..., $View_k(v)$ based on the following rule: a link (u,w) exists in $View_c(v)$ if and only if (1) (u,w) exists in the most recent local view $View_1(v)$, or (2) $u = v$ and (u,w) exists in at least one recent local view $View_i(v)$ for $1 \leq i \leq k$. That is, a broken link is preserved longer in the conservative views of its two end nodes than in those of all other nodes.

As shown in Fig. 3d, after node v detects a broken link (v,w), it will keep a virtual link corresponding to the broken link in its local view for a short time period. Based on this conservative view, v is still a dominator. Note that the virtual link (v,w) is still available during this time period, if v uses a large actual transmission range to create a buffer zone, as discussed earlier. The virtual link stays in v's view until all other nodes have removed this link from their views. When two-hop information is used, link (v,w) exists in local views of v's one-hop neighbors and w's one-hop neighbors, which will remove link (v,w) from their local views after receiving a "Hello" message from v or w. Node v will send its next Hello message within a Hello interval (Δ). Node w may detect the broken link and send its Hello message later than v, but the difference is bounded by Δ. Therefore, it is safe to remove the virtual link (v,w) for v's local view after 2Δ.

This approach can also be applied to other localized CDS and topology control algorithms. However, the conservative decisions are different from algorithm to algorithm, and the construction of conservative views depends on the specific algorithm. For example, in Li, Hou, and Sha's local MST algorithm, a conservative view of node v can be defined as follows: given k most recent local views $View_1(v)$, $View_2(v)$, ..., $View_k(v)$, which contain distance values $d_i(u,w)$ $(1 \leq i \leq k)$ between any two nodes u and w within v's transmission range (including v), their distance in the conservative view is:

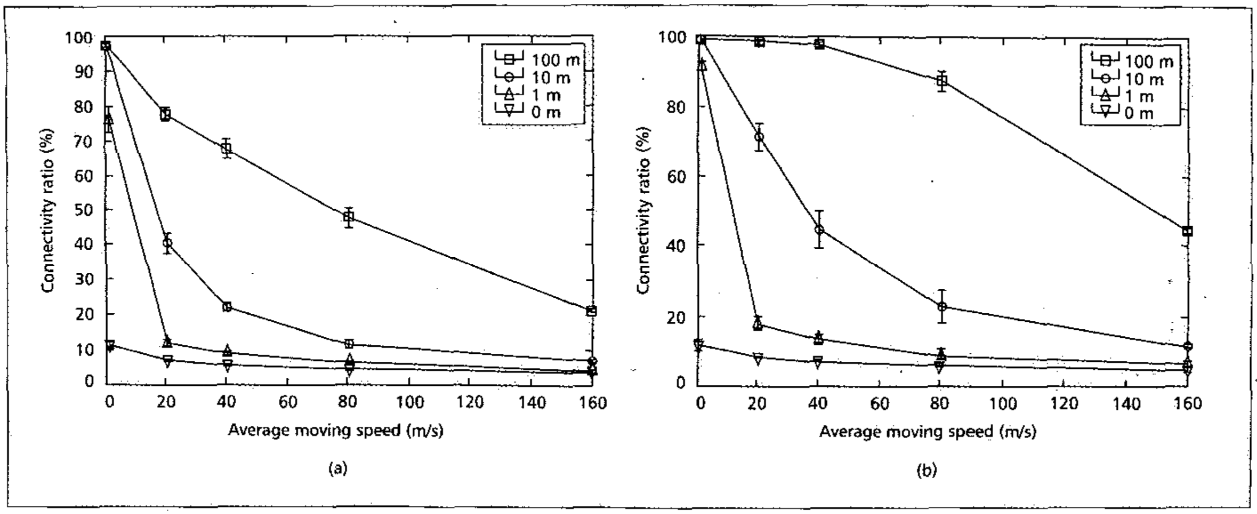
- $\max_i d_i(u,w)$, if $u \neq v$ and $w \neq v$
- $\min_i d_i(u,w)$ otherwise

That is, the virtual distance between v and a neighbor w in its conservative view may be smaller than the actual distance, and the virtual distance between two neighbors may be larger than the actual distance. When conservative local views are used in Fig. 4, both nodes u and v select w as a logical neighbor, and network connectivity is preserved.

Simulation Results

We illustrate sample results from simulations of the mobility control mechanisms. For more results, the readers can refer to [9, 10]. All simulations are conducted using *ns*-2, with 100 nodes, a 900 × 900 m^2 deployment area, normal transmission range $r' = 250$ m, 1 s Hello interval, and a random waypoint mobility model. Network connectivity is measured in terms of the connectivity ratio, which is defined as the ratio of pairs of connected nodes to the total number of pairs. In Dai and Wu's original CDS algorithm [1], the connectivity ratio drops rapidly as the average moving speed increases. When a small (20 m) buffer zone is used to tolerate broken links, the delivery ratio improves significantly under low (1 m/s) to moderate (40 m/s) mobility. With a 100 m buffer zone, the algorithm has almost 100 percent connectivity ratio under very high (160 m/s) mobility.

Figure 6a shows the connectivity ratio of Li, Hou, and Sha's topology control algorithm [5]. When there is no buffer zone



■ Figure 6. *The connectivity ratio of a topology control algorithm under different buffered ranges: a) with inconsistent local views; b) with consistent local views.*

(0 m), the connectivity ratio is very low (10 percent) under an average moving speed of 1 m/s. The connectivity ratio increases significantly after a very small (1 m) buffer zone is used. On the other hand, 100 percent connectivity ratio is not achieved under low mobility. Moderate and high mobility cause low connectivity ratio. Figure 6b shows the effect of using consistent views. When using a 20 m buffer zone in MANETs with a 10 m/s average moving speed, the connectivity ratio is 40 percent without consistent views and 70 percent with consistent views. When using a 100 m buffer zone under a 40 m/s average moving speed, the connectivity ratio reaches 98 percent with consistent views, while the original connectivity ratio without consistent views is only 70 percent.

Overall, simulation results confirm that global connectivity can be compromised by both link availability and view consistency issues. Both issues can be overcome with mobility control mechanisms, and the global property can be preserved with high probability and relatively small overhead.

Conclusion

We have addressed issues related to mobility control in mobile ad hoc networks. To illustrate the importance of the negative impact of mobile nodes on various protocols, we focus on two types of protocols, one for CDS construction and the other for topology control. It has been shown that most existing protocols on CDS construction and topology control will generate incorrect results in the presence of mobile nodes. We discuss two major problems caused by mobility control, link availability and view consistency, and provide several solutions. Mobility control in MANETs is still in its infancy. Many open issues exist:

- How does mobility affect protocols at other layers?
- Can approaches for view consistency in distributed systems be applied in mobile ad hoc networks?
- How should various kinds of cost and efficiency trade-offs be done?

More efforts are needed to address these issues before various protocols can be applied in MANETs with mobile nodes.

References

[1] F. Dai and J. Wu, "Distributed Dominant Pruning in Ad Hoc Wireless Networks," *Proc. ICC*, May 2003, p. 353.

[2] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Message in Mobile Wireless Networks," *Proc. HICSS-35*, Jan. 2002.

[3] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating Sets and Neighbor Elimination based Broadcasting Algorithms in Wireless Networks," *IEEE Trans. Parallel and Distributed Sys.*, vol. 13, no. 1, Jan. 2002, pp. 14–25.

[4] J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks," *Proc. DialM*, 1999, pp. 7–14.

[5] N. Li, J. C. Hou, and L. Sha, "Design and Analysis of an MST-based Topology Control Algorithm," *Proc. INFOCOM*, vol. 3, Mar./Apr. 2003, pp. 1702–12.

[6] V. Rodoplu and T. H. Meng, "Minimum Energy Mobile Wireless Networks," *IEEE JSAC*, vol. 17, no. 8, Aug. 1999, pp. 1333–44.

[7] Y. Wang *et al.*, "Distributed Spanners with Bounded Degree for Wireless Ad Hoc Networks," *Int'l. J. Foundations of Comp. Sci.*, vol. 14, no. 2, 2003, pp. 183–200.

[8] R. Wattenhofer *et al.*, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks," *Proc. INFOCOM*, Apr. 2001, pp. 1388–97.

[9] J. Wu and F. Dai, "Mobility Management and Its Applications in Efficient Broadcasting in Mobile Ad Hoc Networks," *Proc. INFOCOM*, Mar. 2004.

[10] J. Wu and F. Dai, "Mobility-sensitive Topology Control in Mobile Ad Hoc Networks," *Proc. IPDPS*, Apr. 2004.

Biographies

JIE WU [SM] (jie@cse.fau.edu) is a professor at the Department of Computer Science and Engineering, Florida Atlantic University. He has published over 200 papers in various journal and conference proceedings. His research interests are in the area of mobile computing, routing protocols, fault-tolerant computing, and interconnection networks. He served as a program vice chair for the 2000 International Conference on Parallel Processing and 2001 IEEE International Conference on Distributed Computing Systems. He is a program co-chair for the IEEE 1st International Conference on Mobile Ad Hoc and Sensor Systems. He was a co-guest editor of a special issue of *IEEE Computer* on Ad Hoc Networks. He was editor for several special issues of the *Journal of Parallel and Distributing Computing* and *IEEE Transactions on Parallel and Distributed Systems*. Currently, he serves as an Associated Editor of *IEEE Transactions on Parallel and Distributed Systems* and three other international journals. He was a recipient of the 1996-97 and 2001-2002 Researcher of the Year Awards at Florida Atlantic University.

FEI DAI (fdai@fau.edu) is a Ph.D. candidate at the Department of Computer Science and Engineering, Florida Atlantic University. His current research focuses on the design and application of localized algorithms in wireless ad hoc and sensor networks. Other research areas include interconnection networks, fault-tolerant routing, and software architecture. He received his B.S. and M.S. degrees in computer science in 1990 and 1993, respectively, from Nanjing University, China. He worked as a senior programmer at Greatwall Computer, China, from 1993 to 1996, and as a software architect and team leader in J&A Securities, China, from 1996 to 2000.