A Dominating-Set-Based Routing Scheme in Ad Hoc Wireless Networks [1]

Jie Wu and Hailan Li
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

[1]This work was supported in part by NSF grants CCR 9900646 and ANI 0073736.

Abstract

Efficient routing among a set of mobile hosts (also called nodes) is one of the most important functions in ad-hoc wireless networks. Routing based on a connected dominating set is a promising approach, where the searching space for a route is reduced to nodes in the set. A set is dominating if all the nodes in the system are either in the set or neighbors of nodes in the set. In this paper, we propose a simple and efficient distributed algorithm for calculating connected dominating set in ad-hoc wireless networks, where connections of nodes are determined by their geographical distances. We also propose an update/recalculation algorithm for the connected dominating set when the topology of the ad hoc wireless network changes dynamically. Our simulation results show that the proposed approach outperforms a classical algorithm in terms of finding a small connected dominating set and doing so quickly. Our approach can be potentially used in designing efficient routing algorithms based on a connected dominating set.

Keywords: *ad hoc wireless networks, dominating sets, mobile computing, routing, simulation*

1 Introduction

Recent advances in technology have provided portable computers with wireless interfaces that allow networked communication among mobile users. The resulting computing environment, which is often referred to as *mobile computing* [18], no longer requires users to maintain a fixed and universally known position in the network and enables almost unrestricted mobility.

An *ad hoc* wireless network is a special type of wireless network in which a collection of mobile hosts with wireless network interfaces may form a temporary network, without the aid of any established infrastructure or centralized administration. If only two hosts, located closely together within *wireless transmission range* of each other, are involved in the ad hoc wireless network, no real routing protocol or decision is necessary. However, if two hosts that want to communicate are outside their wireless transmission ranges, they could communicate only if other hosts between them in the ad hoc wireless network are willing to forward packets for them. For example, in the network shown in Figure 1, mobile host C is outside the range of host A's wireless transmitter (indicated by the circle around A) and host A is also outside the range of host C's wireless transmitter. If A and C wish to exchange packets, they may use host B to forward packets for them, since B is within the overlap between A's and C's ranges.

We can use a simple graph $G = (V, E)$ to represent an ad hoc wireless network, where V represents a set of wireless mobile hosts and E represents a set of edges. An edge between host pairs $\{v, u\}$ indicates that both hosts v and u are within their wireless transmitter ranges. To simplify our discussion, we assume all mobile hosts are homogeneous, i.e., their wireless transmitter ranges are the same. In other word, if there is an edge $e = \{v, u\}$ in E, it indicates u is within v's range and v is within u's range. Thus the corresponding graph will be an undirected graph. The graph in Figure 2 represents the corresponding ad hoc wireless network in Figure 1.

Routing in ad hoc wireless networks poses special challenges. In general, the main characteristics of mobile computing are low bandwidth, mobility, and low power. Wireless networks deliver lower bandwidth than wired networks, and hence, the information collection (during the formation of a routing table) is expensive. Mobility of hosts, which causes topological changes of the underlying network, also increases the volatility of network information. In addition, the limitation of power leads users disconnect mobile unit frequently in order to save power consumption. This feature may also introduce mobile networks more failures (also called switching on/off), which can be considered as a special form of mobility.

Traditional routing protocols in wired networks, that generally use either *link state* [20, 23] or

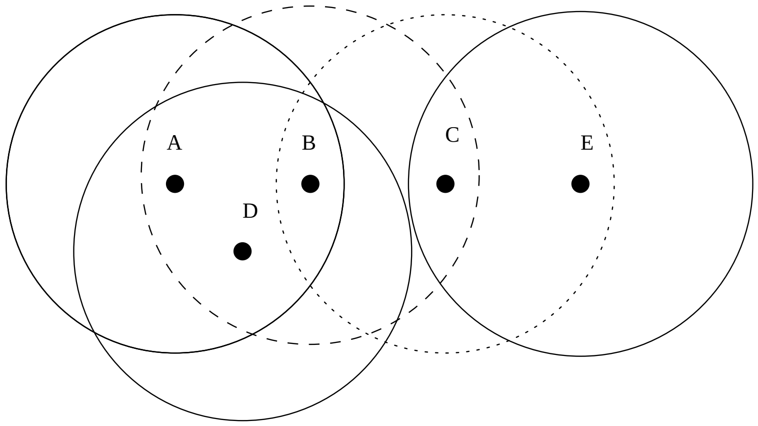

Figure 1: A simple ad hoc wireless network of five wireless mobile hosts.

distance vector [13, 21], are no longer suitable for ad hoc wireless networks. In an environment with mobile hosts as routers, convergence to new, stable routes after dynamic changes in network topology may be slow and this process could be expensive due to low bandwidth. Routing information has to be localized to adapt quickly to changes such as host movements. *Cluster-based routing* [19] is a convenient method for routing in ad hoc wireless networks. In an ad hoc wireless network, hosts within vicinity (i.e., they are physically close to each other) form a cluster, or a *clique*, which is a complete subgraph. Each cluster has one or more *gateway* host to connect other clusters in the network. Gateway hosts (from different clusters) are usually connected. In Figure 2, hosts A, B, and D form one cluster and hosts C and E form another one. B and C are gateway hosts which are connected. *Backbone-based routing* [7] and *spine-based routing* [8] use a similar approach, where a backbone (spine) consists of hosts similar to gateway hosts.

Note that gateway hosts form a *dominating set* [12] of the corresponding ad hoc wireless network. A subset of the vertices of a graph is a dominating set if every vertex not in the subset is adjacent to at least one vertex in the subset. Moreover, this dominating set should be connected for the ease of the routing process within the reduced graph consisting of dominating nodes only. We refer all approaches that use gateway hosts to form a dominating set as *dominating-set-based routing.* The main advantage of connected-dominating-set-based routing is that it simplifies the routing process to the one in a smaller subnetwork generated from the connected dominating set. This means that only gateway hosts need to keep routing information. As long as changes in network topology do not affect this subnetwork there is no need to recalculate routing tables.

Clearly, the efficiency of this approach depends largely on the process of finding a connected

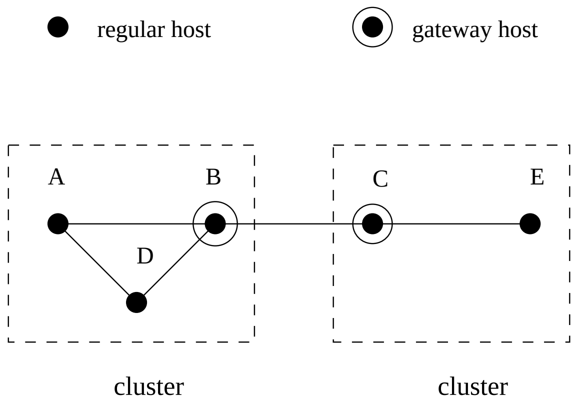

Figure 2: The represented graph of the ad hoc wireless network in Figure 1.

dominating set and the size of the corresponding subnetwork. Unfortunately, finding a minimum connected dominating set is NP-complete for most graphs. In this paper, we propose a simple distributed algorithm that can quickly determine a connected dominating set in a given connected graph, which represents an ad hoc wireless network. Node connections in the ad hoc wireless network are determined based on their geographical distances in a 2-D or 3-D space. That is, two nodes are connected if their geographical distance is within a given wireless transmission range. We study some properties of the derived connected dominating set. We show that proposed approach outperforms a classical approach in terms of finding a small dominating set and does so quickly. We also discuss ways to update/recalculate the connected dominating set when the underlying graph changes with the movement of mobile hosts. Efficient routing in ad hoc wireless networks based on the derived dominating set are described.

This paper is organized as follows: Section 2 overviews related works. Section 3 presents the proposed approach on finding a small connected dominating set. Two rules for improvement are also included. Update/recalculation of the connected dominating set is discussed in Section 4. The routing based on the connected dominating set is briefly discussed in Section 5. Performance evaluation is done in Section 6, where the proposed algorithm is compared with the one proposed by Das et al [7]. Finally, in Section 7 we conclude this paper and discuss ideas for future work.

2 Related Work

There are numerous distributed routing algorithms, mostly shortest path ones, proposed in wired networks represented by simple weighted graphs [9, 10, 14, 21, 20, 27]. In these algorithms, routing information is kept in a *routing table* associated with each host and it is collected through iterative rounds of message exchanges between neighboring hosts. These algorithms differ in the way the routing table is constructed, maintained and updated at each host. In general, routing protocols can be classified into *link state* and *distance vector*.

The *link state* routing algorithms are closer to the centralized version of the shortest path algorithm. Each node maintains a view of the network topology with a cost for each link. To keep these views consistent, each node periodically broadcasts the link costs of its outgoing links to all other nodes using a protocol such as flooding. As a node receives this information, it updates its view of the network topology and applies a shortest path algorithm to choose its next hop for each destination. Some of the link costs in a node's view can be incorrect because of long propagation delays, partitioned network, etc. Such inconsistent views of network topologies might lead to formation of *routing loops*. These loops, however, are short-lived, because they disappear in the time it takes a message to traverse the network [20]. Although the link state routing generally requires each host to have knowledge of the entire network topology, there are link state algorithms [3] in which each host only maintains partial information of the network.

The *distance vector* routing algorithms use the distributed version of Bellman-Ford algorithm (DBF) [1, 13, 17, 21]. Each node v maintains, for each destination d, a set of distances $\{dis^d_{vu}\}$, where u ranges over the neighbors of v. Node v selects neighbor w as a next hop for destination d if dis^d_{vw} equals $min_u\{dis^d_{vu}\}$. The succession of next hops chosen in this manner leads to d along the shortest path. In order to keep the distance estimates up-to-date, each node monitors the cost of its outgoing links and periodically broadcasts, to each one of its neighbors, current estimate of the shortest distance to every other node in the network. Comparing to the link state approaches, the distance vector routing algorithms are computationally more efficient, require much less storage space and much less network bandwidth overhead; however, such an algorithm also has both short lived and long lived loops [4]. The main reason for formation of routing loops is that nodes choose their next hops in a completely distributed fashion based on information which can possibly be stale. Many modifications [14, 20, 22] to the DBF algorithm eliminate the looping problem by forcing all nodes in the network to participate in some form of inter-nodal coordination protocol. However, such coordination might be effective only when network topological changes are rare.

The conventional routing algorithms described above are not suitable for dynamic networks, especially for those quick changing mobile networks, like ad-hoc wireless networks. The main problems are computational burden, bandwidth overhead, and slow convergence of routing information. These problems are especially pronounced in ad-hoc wireless networks that have low power, limited bandwidth, and unrestricted mobility. More detailed discussion on these problems can be found in [15, 26]. Consequently, several routing protocols, particularly for wireless networks, are proposed [2, 5, 6, 8, 7, 9, 15, 16, 19, 24, 26, 25, 28]. Among them, [5, 9, 15, 24, 26] have been reviewed in detail in [19]. In the rest of this section, we will concentrate on approaches that are dominating-set-based [6, 8, 7, 16, 19, 28].

The *cluster-based algorithm* [19] divides a given graph into a number of overlapping, but non-redundant, clusters. Each cluster is a clique which is a complete subgraph. A cluster is non-redundant if it cannot be covered by a set of other clusters. One (or more) representative node, called a *boundary node*, is selected from each cluster to form a connected subnetwork where the routing process proceeds and this subnetwork forms a connected dominating set. Each boundary node has a complete view of the subnetwork captured by a routing table associated. In this way, the routing process centralizes the whole network into a small connected subnetwork so that as long as network topological changes do not affect this centralized part of the network, there is no need to recalculate routing tables in the subnetwork. The routing protocol is completed in two phases: *cluster formation* and *cluster maintenance.* During the cluster formation, the network is viewed as a dynamically growing system, in other words, assume that mobile hosts are inserted into the network sequentially. Therefore, the cluster formation algorithm needs to run $O(\nu)$ rounds, taking $O(\Delta^3)$ time and transmitting $O(\gamma + \Delta)$ messages for each round, where ν is the number of hosts in the network, γ is the number of boundary nodes, and Δ is the maximum node degree. At each round, the newly entered node receives current cluster information from each of its neighbors that are boundary nodes, and based on that information, it determines a new formation of clusters and boundary nodes, then propagates to all the (new) boundary nodes. Once receiving cluster information, each node generates its own routing table, which contains destination identifier, the next hop node, and the number of hops. Intermediate nodes in any route consists of boundary nodes only. However, boundary nodes may not include all the intermediate nodes in a shortest path, i.e., the route generated may not be a shortest path. The cluster maintenance phase consists of non-simultaneous topological changes, such as *switch on*, *switch off*, *link connection*, and *link disconnection.* Note that the initial cluster formation algorithm is fully sequential, causing a high time complexity. The resultant cluster structure also depends on the order in which mobile hosts are added to the network. Each node in the subnetwork always needs to keep up-to-date cluster information, which does not scale well when the network grows. Whenever there is a network topological change, cluster information needs to be recalculated and new information will be propagated to new boundary nodes. Because of the time delay of this propagation, temporary routing loops might be introduced.

Das et al proposed a series of routing algorithms [6, 8, 7, 28] for ad-hoc wireless networks. Similar to the cluster-based routing [19], the idea is to identify a subnetwork that forms a *minimum connected dominating set* (MCDS). Again, each node in the subnetwork keeps a routing table that captures the topological structure of the whole network. Each node in the subnetwork is called a *spine node* or *backbone node* (we call it gateway in our proposed algorithm), so that as long as network topological changes do not affect these MCDS nodes, there is no need to recalculate routing tables. The formation of MCDS nodes is based on Guha and Khuller's *approximation algorithm* [11] and has been used in all the papers by Das et al [6, 8, 7, 28]. This MCDS calculation algorithm has two main advantages over to the cluster-based approach [19]. First, each node only needs 2-distance neighborhood information, unlike [19] in which each node needs information of the entire network topology. Second, the algorithm runs $O(\gamma)$ rounds. The overall complexities are $O(\gamma\Delta^2 + \nu)$ in terms of time and $O(\Delta\nu\gamma + m + \nu\log\nu)$ in terms of messages, where γ is the number of nodes in the resultant dominating set, m is the number of edges, ν is the number of node, and Δ is the maximum node degree. This is an improvement compared to [19] in terms of time complexity, although it has higher message complexity over the one in [19] in the worst case. The main drawback of this algorithm is that it still needs a non-constant number of rounds to determine a connected dominating set.

Methods in [6, 8, 7, 28] differ in the way routing tables are constructed and propagated to non-MCDS nodes. The requirement for shortest paths adds one additional dimension of complexity. Because the set of MCDS nodes (dominating set) may not include all intermediate nodes of a shortest path, the routing process cannot be restricted to MCDS nodes. In other words, in order to compute a routing table, each MCDS node needs to know the entire network topology. An all-pairs shortest path algorithm is actually running on the entire network, not on the reduced subnetwork of MCDS nodes. Therefore, it may lose part of the original goal of reducing the searching space for routing.

Another extreme approach, as proposed by Johnson [16], uses *dynamic source routing* (DSR) without constructing any routing tables. Normally, the resultant routing path is not the shortest. However, this protocol adapts quickly to routing changes when host movement is frequent, yet requires little or no overhead during periods in which hosts move less frequently. The approach consists of *route discovery* and *route maintenance.* Route discovery allows any host to dynamically discover a route to a destination host. Each host also maintains a *route cache* in which it caches source routes that it has learned. Unlike regular routing-table-based approaches that have to perform periodic routing updates, route maintenance only monitors the routing process and informs the sender of any routing errors. One can easily apply Johnson's approach to the dominating-set-based routing, where route discovery is restricted to the subnetwork containing the connected dominating set. A compromising approach has been discussed in [28] which keeps partial global information of the subnetwork.

The *zone-based routing* [25] is another compromising approach, where each routing table keeps information for destinations within a certain distance (the corresponding area is called a *zone*). Information for destinations outside the zone area is obtained on an on-demand basis, i.e., through a route recovery phase as in DSR.

3 Proposed Approach

As mentioned early, we will focus on the formation of a dominating set. Some desirable features for a dominating set are listed below:

- The formation process should be distributed and simple. Ideally, it requires only local information and constant number of iterative rounds of message exchanges among neighboring hosts.
- The resultant dominating set should be connected and close to minimum.
- The resultant dominating set should include all intermediate nodes of any shortest path. In this case, an all-pair shortest paths algorithm only needs to be applied to the subnetwork generated from the dominating set.

3.1 Marking process

We propose a marking process that marks every vertex in a given connected and simple graph $G = (V, E)$. $m(v)$ is a marker for vertex $v \in V$, which is either T (marked) or F (unmarked). We will show later that marked vertices form a connected dominating set. We assume that all vertices are unmarked initially. $N(v) = \{u | \{v, u\} \in E\}$ represents the *open neighbor set* of vertex v. Initially v is equal to $N(v)$.

Marking process:

1. Initially assign marker F to every v in V.

2. Every v exchanges its open neighbor set $N(v)$ with all its neighbors.

3. Every v assigns its marker $m(v)$ to T if there exist two unconnected neighbors.

In the example of Figure 2, $N(A) = \{B, D\}$, $N(B) = \{A, C, D\}$, $N(C) = \{B, E\}$, $N(D) = \{A, B\}$, and $N(E) = \{C\}$. After Step 2 of the marking process, vertex A has $N(B)$ and $N(D)$, B has $N(A)$, $N(C)$, and $N(D)$, C has $N(B)$ and $N(E)$, D has $N(A)$ and $N(B)$, and E has $N(C)$. Based on Step 3, only vertices B and C are marked T.

Clearly, each vertex knows distance-2 neighborhood information after Step 2 of the marking process, i.e., its neighbor's neighbor information. The cost of checking the connectivity of two neighbors is upper bounded by $\Delta^2(G)$ (or simply Δ^2), where Δ is the degree of graph G, i.e., $\Delta(G) = \max\{|N(v)||v \in V\}$. There are $\frac{|N(v)|(|N(v)|-1)}{2}$ possible pairs of neighbors of vertex v, which is upper bounded by Δ^2. Therefore, the cost of the marking process at each vertex is $O(\Delta^2)$. The amount of message exchanges at each vertex is also $O(\Delta)$, which corresponds to the number of neighbors.

3.2 Properties

Assume $V^{'}$ is the set of vertices that are marked T in V, i.e., $V^{'} = \{v | v \in V, m(v) = T\}$. The *reduced graph* $G^{'}$ is the subgraph of G induced by $V^{'}$, i.e., $G^{'} = G[V^{'}]$. The following two theorems show that $G^{'}$ is a dominating set of G and it is connected.

Theorem 1: *Given a graph $G = (V, E)$ that is connected, but not completely connected, the vertex subset $V^{'}$, derived from the marking process, forms a dominating set of G.*

Proof: Randomly select a vertex v in G. We show that v is either in $V^{'}$ (a set of vertices in V that are marked T) or adjacent to a vertex in $V^{'}$. Assume v is marked F, if there is at least one neighbor marked T, the theorem is proved. If all its neighbors are marked F, we consider the following two cases:

- All the other vertices in G are neighbors of v. Based on the marking process and the fact that $m(v)$=F, all these neighbors must be pairwise connected, i.e., G is completely connected. This contradicts the assumption that G is not completely connected.

- There is at least one vertex u in G that is not adjacent to vertex v. Construct a shortest path, $(v, v_1, v_2, ..., u)$, connecting vertices v and u. Such a path always exists since G is a connected graph. Note that v_2 is u when v and u are distance-2 apart in G, i.e., $d_G(v, u) = 2$. Also, v and v_2 are disconnected; otherwise, $(v, v_2, ..., u)$ is a shorter path connecting v and u. Based on the marking process, vertex v_1, with both v and v_2 as its neighbors, must be marked T. Again this contradicts the assumption that v's neighbors are all marked F. □

When the given graph G is completely connected, all vertices are marked F. This make sense, because if all vertices are directly connected, there is no need for gateway hosts.

Theorem 2: *The reduced graph $G' = G[V']$ is a connected graph.*

Proof: We prove this theorem by contradiction. Assume G' is disconnected and v and u are two disconnected vertices in G'. Assume $dis_G(v, u) = k + 1 > 1$ and $(v, v_1, v_2, ..., v_k, u)$ is a shortest path between vertices v and u in G. Clearly, all $v_1, v_2, ..., v_k$ are distinct and among them there is at least one v_i such that $m(v_i) = F$ (otherwise, v and u are connected in G'). On the other hand, the two adjacent vertices of v_i, v_{i-1} and v_{i+1}, are not connected in G; otherwise, $(v, v_1, v_2, ..., v_k, u)$ is not a shortest path. Therefore, $m(v_i) = T$ based on the marking process. This brings up a contradiction. □

Vertices in a dominating set are called *gateway nodes* and vertices outside a dominating set are called *non-gateway nodes*. The next theorem shows that, except for source and destination vertices, all intermediate vertices in a shortest path are contained in the dominating set derived from the marking process.

Theorem 3: *The shortest path between any two nodes does not include any non-gateway node as an intermediate node.*

Proof: We prove this theorem also by contradiction. Assume a shortest path between two vertices v and u includes a non-gateway node v_i as an intermediate node, in other words, this path can be represented as $(v, ..., v_{i-1}, v_i, v_{i+1}, ..., u)$. We label the vertex that precedes v_i on the path as v_{i-1}, similarly, the vertex that follows v_i on the path as v_{i+1}. Because vertex v_i is a non-gateway node, i.e., $m(v_i)$=F, there must be a connection between v_{i-1} and v_{i+1}. Therefore, a shorter path between v and u can be found as $(v, ..., v_{i-1}, v_{i+1}, ..., u)$. This contradicts the original assumption. □

Since the problem of determining a minimum connected dominating set of a given connected graph is NP-complete, the connected dominating set derived from the marking process is normally

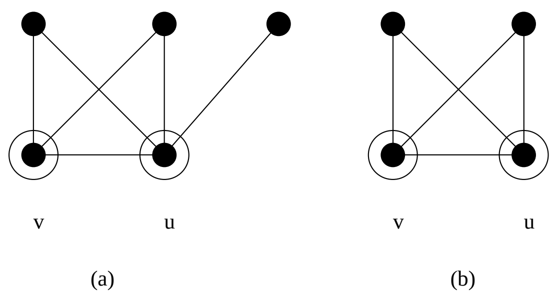

Figure 3: Two samples.

non-minimum. In some cases, the resultant dominating set is *trivial*, i.e., $V^{'} = V$ or $V^{'} = \{\}$. For example, any vertex-symmetric graph will generate a trivial dominating set using the proposed marking process. However, the marking process is efficient for the ad hoc wireless mobile network where the corresponding graph forms a set of localized clusters (or cliques). Our simulation results (to be discussed later) confirm this observation.

3.3 Extensions

In the following, we propose two rules to reduce the size of a connected dominating set generated from the marking process. We first assign a distinct id, $id(v)$, to each vertex v in $G^{'}$. $N[v] = N(v) \cup \{v\}$ is the *closed neighbor set* of v, as oppose to the open one $N(v)$.

Rule 1: *Consider two vertices v and u in $G^{'}$. If $N[v] \subseteq N[u]$ in G and $id(v) < id(u)$, change the marker of v to F if node v is marked, i.e., $G^{'}$ is changed to $G^{'} - \{v\}$,*

The above rule indicates when the closed neighbor set of v is covered by the one of u, vertex v can be removed from $G^{'}$ if v's id is smaller than u's. Note that if v is marked and its closed neighbor set is covered by the one of u, it implies vertex u is also marked. When v and u have the same closed neighbor set, the vertex with a smaller id will be removed. It is easy to prove that $G^{'} - \{v\}$ is still a connected dominating set of G. The condition $N[v] \subseteq N[u]$ implies v and u are connected in $G^{'}$.

In Figure 3 (a), since $N[v] \subset N[u]$, vertex v is removed from $G^{'}$ if $id(v) < id(u)$ and vertex u is the only dominating node in the graph. In Figure 3 (b), since $N[v] = N[u]$, either v or u can be removed from $G^{'}$. To sure one and only one is removed, we pick the one with a smaller id.

Rule 2: *Assume u and w are two marked neighbors of marked vertex v in G'. If $N(v) \subseteq N(u) \cup N(w)$ in G and $id(v) = \min\{id(v), id(u), id(w)\}$, then change the marker of v to F.*

The above rule indicates when the open neighbor set of v is covered by the open neighbor sets of two of its marked neighbors, u and w, if v has the minimum id of the three, it can be removed from G'. The condition $N(v) \subseteq N(u) \cup N(w)$ in Rule 2 implies that u and w are connected. The subtle difference between Rule 1 and Rule 2 is the use of open and close neighbor sets. Again, it is easy to prove $G' - \{v\}$ is still a connected dominating set. Both u and w are marked, because the fact that v is marked and $N(v) \subseteq N(u) \cup N(w)$ in G does not imply that u and w are marked. Therefore, if one of u and w is not marked, v cannot be unmarked (change the marker to F). Therefore, to apply Rule 2, an additional step needs to be added in the marking process:

Marking process:

1-3. (Same as before).

4. If v is marked ($m(v) = T$), send its status to all its neighbors.

Consider the example in Figure 4. Clearly, $N(v) \subseteq N(u) \cup N(w)$. If $id(v) = \min\{id(v), id(u), id(w)\}$, vertex v can be removed from G' based on Rule 2. If $id(u) = \min\{id(v), id(u), id(w)\}$, then vertex u can be removed based on Rule 1, since $N[u] \subseteq N[v]$. If $id(w) = \min\{id(v), id(u), id(w)\}$, no vertex can be removed. Therefore, the id assignment also decides the final outcome of the dominating set. Note that Rule 2 can be easily extended to a more general case where the open neighbor set of vertex v is covered by the union of open neighbor sets of more than two neighbors of v in G'. However, the connectivity requirement for these neighbors is more difficult to specify at vertex v.

The role of id is very important to avoid "illegal simultaneous" removal of vertices in G'. In general, vertex v cannot be removed even if $N[v] \subset N[u]$, unless $id(v) < id(u)$. Consider the example of Figure 2 with $id(v) = \min\{id(v), id(u), id(w)\}$. If the above rule was not followed, vertex u would be unmarked to F (because $N[u] \subset N[v]$ even though $id(v) < id(u)$), and based on Rule 2, vertex v would be unmarked to F. Clearly, the only vertex w in V' does not form a dominating set any more.

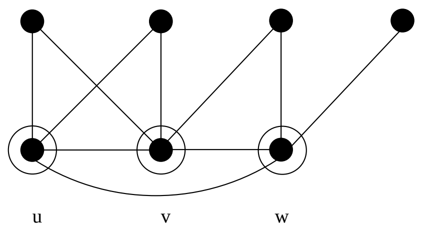

Figure 4: One additional sample.

3.4 Example

Figure 5 shows an example of using the proposed marking process and its extensions to identify a set of connected dominating nodes. Each node keeps a list of its neighbors and sends this list to all its neighbors. By doing so each node has distance-2 neighborhood information, i.e., information about its neighbors and the neighbors of all its neighbors.

Node 1 will not mark itself as a gateway node because its only neighbors 2 and 3 are connected. Node 3 will mark itself as a gateway node because there is no connection between neighbors 1 and 4 (2 and 4). After node 3 marks itself, it sends its status to its neighbors 1, 2, and 4. This gateway status will be used to apply Rule 2 to unmark some gateway nodes to non-gateway nodes. Figure 5 (b) shows the gateway nodes (nodes with double cycles) derived by the marking process without applying two rules.

After applying Rule 1, node 17 will be unmarked to the non-gateway status as shown in Figure 5 (c). The closed neighbor set of node 17 is $N[17] = (17, 18, 19, 20)$, and the closed neighbor set of node 18 is $N[18] = (16, 17, 18, 19, 20)$. Apparently, $N[17] \subseteq N[18]$. Also the id of node 17 is less than the id of node 18, thus node 17 can unmark itself by applying Rule 1.

After applying Rule 2, node 8 will be unmarked to the non-gateway status as shown in Figure 5 (d). Node 8 is aware that its two neighbors 14 and 16 are all marked. This invokes node 8 to apply Rule2 to check if condition $N(8) \subseteq N(14) \cup N(16)$ holds or not. The neighbor set of node 14 is $N(14) = (7, 8, 9, 10, 11, 12, 13, 16)$, the neighbor set of node 8 is $N(8) = (12, 13, 14, 15, 16)$, the neighbor set of node 16 is $N(16) = (8, 14, 15, 18)$, and therefore, $N(14) \cup N(16) = (7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 18)$. Apparently, $N(8) \subseteq N(14) \cup N(16)$. The id of node 8 is the smallest among nodes 8, 14, and 16. Thus node 8 can unmark itself by applying Rule 2.

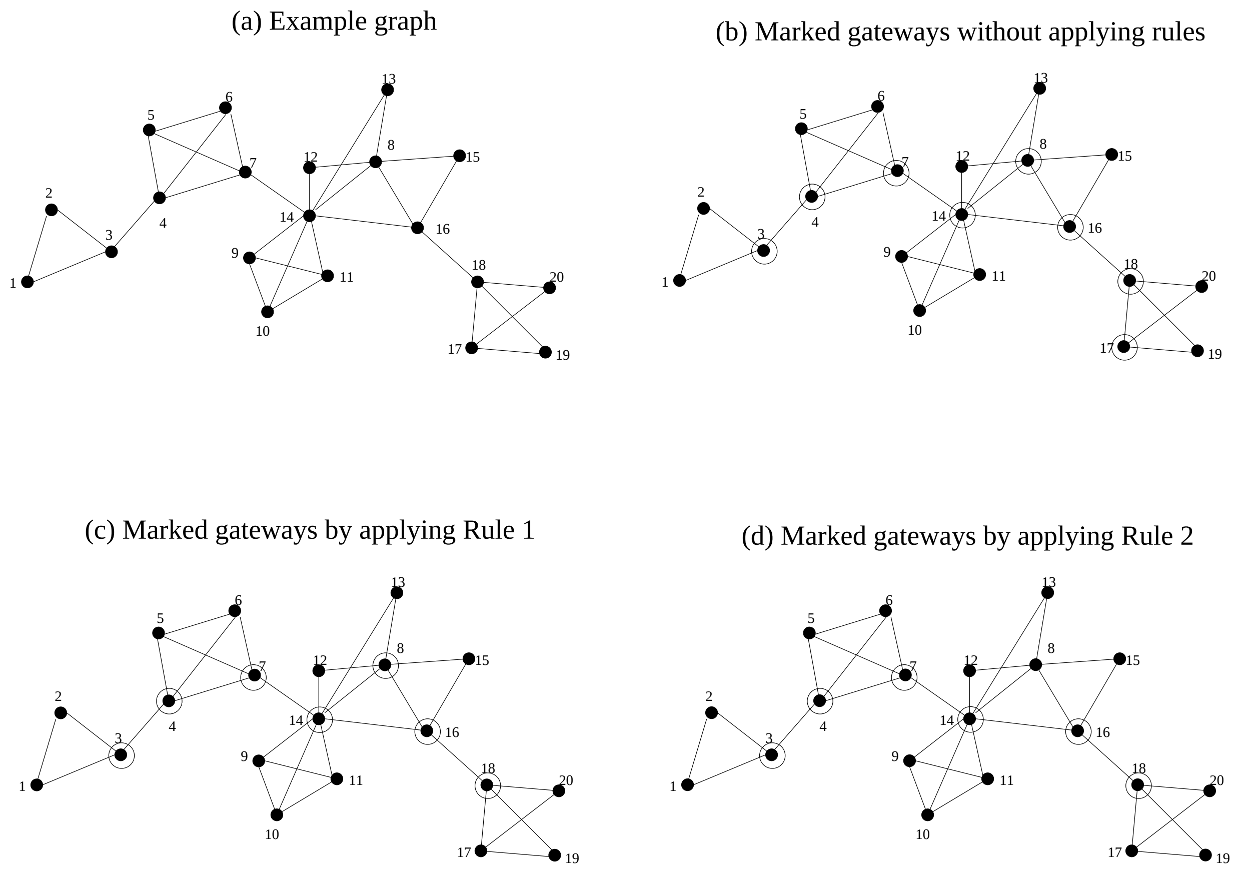
(a) Example graph
(b) Marked gateways without applying rules
(c) Marked gateways by applying Rule 1
(d) Marked gateways by applying Rule 2

Figure 5: An example of the proposed algorithm

4 Update/Recalculation of the Connected Dominating Set

In the ad hoc wireless network each host can move around without speed and distance limitation. Also in order to reduce power consumption, mobile hosts may switch off at any time and switch on later. We can summarize topological changes of an ad hoc wireless network into three different types: *mobile host switching on*, *mobile host switching off*, and *mobile host movement.*

The challenge here is when and how each vertex should update/recalculate gateway information. The gateway *update* means that only individual mobile hosts update their gateway status. The gateway *recalculation* means that the entire network recalculates gateway/non-gateway status. If many mobile hosts in the network are in movement, gateway recalculation might be a better approach, i.e., the connected dominating set is recalculated from scratch. On the other hand, if only few mobile hosts are in movement, then gateway information can be updated locally. The questions arise as exactly when to update gateways and when to recalculate gateways from scratch. Answers to these questions will be part of our future work.

In the following, we will focus only on the gateway update for three types of topology changes mentioned above. Without lost of generality, we assume that the underlying graph of an ad hoc wireless network always remains connected.

4.1 Mobile host switching on

When a mobile host v switches on, only its non-gateway neighbors, along with host v, need to update their status, because any gateway neighbor will still remain as gateway after a new vertex v is added. For example, in Figure 6 (a), when host v switches on, the status of gateway neighbor host u is not affected, because at least two of the u's neighbors u_1, u_2, and u_3 are not connected originally and these connections will not be affected by host v's switch on. On the other hand, in Figure 6 (b), host v's switch on might lead non-gateway neighbor host w to mark itself as gateway, depending on the connection between host v and w's neighbors w_1, w_2, and w_3.

The corresponding update process can be the following:

1. Mobile host v broadcasts to its neighbors about its switching on.
2. Each host $w \in v \cup N(v)$ exchanges its open neighbor set $N(w)$ with its neighbors.
3. Host v assigns its marker $m(v)$ to T if there are two unconnected neighbors.

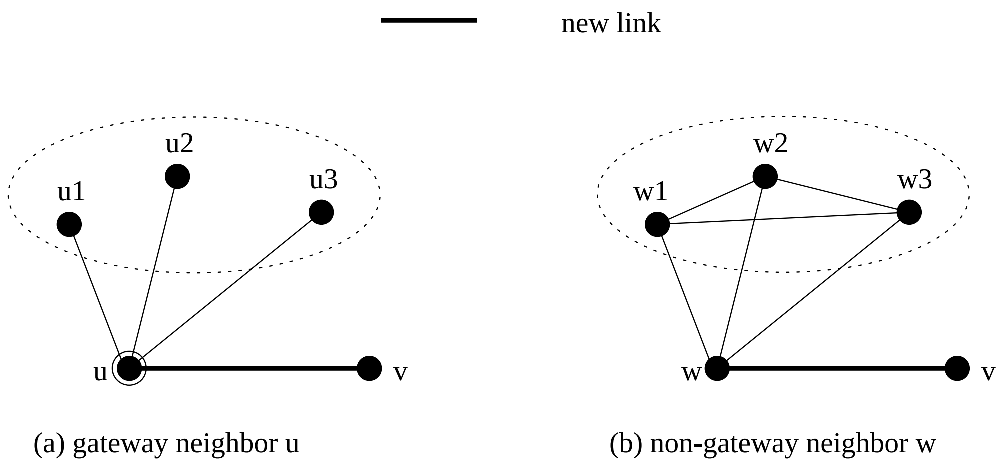

Figure 6: Mobile host v switches on

4. Each non-gateway host $w \in N(v)$ assigns its marker $m(w)$ to T if it has two unconnected neighbors.
5. Whenever there is a newly marked gateway, host v and all its gateway neighbors apply *Rule 1* and *Rule 2* to reduce the number of gateway hosts.

4.2 Mobile host switching off

When a mobile host v switches off, only gateway neighbors of that switched off host need to update their status, because any non-gateway neighbor will still remain as non-gateway after vertex v is deleted. For example, in Figure 7 (a), when v switches off, non-gateway neighbor w is not affected. Host w's neighbors w_1, w_2, and w_3 are pairwise connected originally and these pairwise connections will not be affected by host v's switch off. On the other hand, in Figure 7 (b), host v's switch off might change a gateway neighbor u to non-gateway, depending on the connection between its neighbor hosts u_1, u_2, and u_3.

The corresponding update process can be the following:

1. Mobile host v broadcasts to its neighbors about its switching off.
2. Each gateway neighbor $w \in N(v)$ exchanges its open neighbor set $N(w)$ with its neighbors.

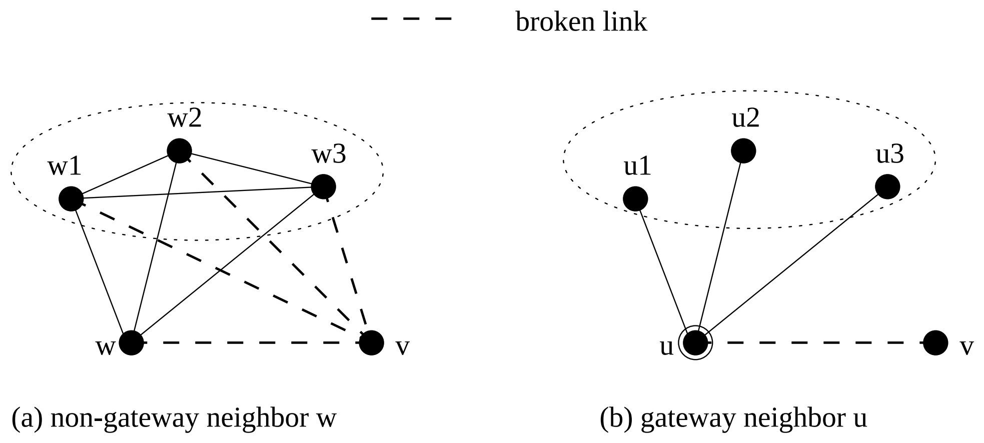

Figure 7: Mobile host v switches off

3. Each gateway neighbor w changes its marker $m(w)$ to F if all neighbors are pairwise connected.

Note that since the underlying graph is connected, we can easily prove by contradiction that the resultant dominating set (using the above marking process) is still connected when a host (gateway or non-gateway) switches off.

4.3 Mobile host movement

A mobile host v's movement can be viewed as several simultaneous or non simultaneous link connections and disconnections. For example, when a mobile host moves, it may lead several link disconnections with its neighbor hosts, and at the same time, it may have new link connections to the hosts within its wireless transmission range, these new links may be disconnected again depending on the way host v moves.

In order to synchronize mobile host's movement in gateway updates, just before mobile host v starts to move, it sends out a special signal {id(v), Start}, then during its movement host v continuously sends out signal {id(v), Heart_Beat} at every τ time interval, and when it stops moving around, host v sends out signal {id(v), Stop}.

When a host u receives signal {id(v), Start}, it starts to monitor host v's movement. If host u continuously receives signal {id(v), Heart_Beat} at every τ time interval, and at the end, it receives signal {id(v), Stop}, then no action is needed at host u. On the other hand, if host u does not receive a {id(v), Heart_Beat} or {id(v), Stop} signal after τ time interval since last time it received a {id(v), Heart_Beat} or {id(v), Start} signal, then host u immediately concludes it has a broken link to host v, and it will perform certain actions (to be discussed later) for broken link $\{u, v\}$.

When a host u receives signal {id(v), Heart_Beat} without receiving signal {id(v), Start} previously, host u can conclude that it has a new link to host v, and it immediately performs certain actions (to be discussed later) for new link $\{u, v\}$. At the same time, host u continuously monitors host v's movement.

In the following subsections, we propose two different update algorithms for the connected dominating set.

4.4 Connected Dominating Set Update I

For mobile host v that is in movement, at every τ time interval during its movement, it performs the following update process:

1. Mobile host v compares its new neighbor set $N'(v)$ with the original neighbor set $N(v)$. If they are the same, host v simply does nothing further; otherwise, it continues the following steps.
2. Host v exchanges the open neighbor set with all its neighbors.
3. Host v marks itself as gateway if it have two unconnected neighbors; otherwise, it marks itself as non-gateway.
4. Further, if host v marked itself as gateway, host v and all its gateway neighbors apply *Rule 1* and *Rule 2* to reduce the number of gateway hosts in the network.

For neighbor u of host v, we consider two subcases: *mobile host u recognizes a new link* $\{u, v\}$ and *mobile host u recognizes a broken link* $\{u, v\}$.

Mobile host u recognizes a new link $\{u, v\}$: When a mobile host u recognizes a new link $\{u, v\}$, two types of mobile hosts need to recalculate their gateway status. One is mobile host u if it is non-gateway host; the other ones are common neighbors of mobile hosts u and v. For example, in Figure 8 (a), non-gateway host u may mark itself as gateway after a new link to v is established, depending on the connection between v and u's neighbors u_1, u_2, u_3, and u_4. On the other hand, in Figure 8 (b), at least two of gateway host u's neighbors are unconnected originally, new link

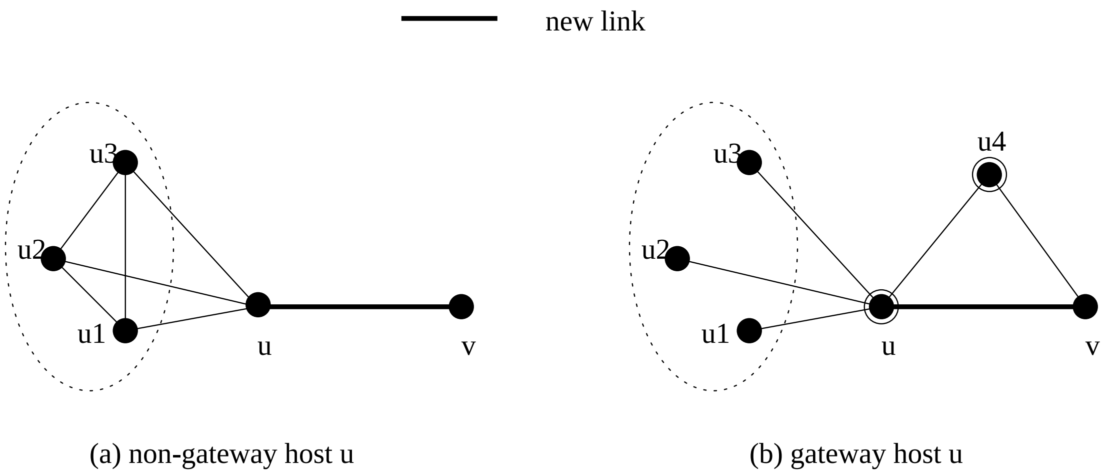

Figure 8: Mobile host u recognizes new link $\{u, v\}$

$\{u, v\}$ has no effect on these unconnected neighbors; therefore, u still remains as a gateway host. In the same figure, common gateway neighbor u_4 of mobile host u and v may unmark itself as non-gateway after new link $\{u, v\}$ is established.

The corresponding update process can be the following:

1. Mobile host u detects a new link to v and it exchanges the open neighbor set with all its neighbors.
2. Upon receiving the open neighbor set $N(u)$ from host u, gateway host w recalculates its status if it is a common neighbor of hosts u and v.
3. If host u is gateway, it simply does nothing further; otherwise, host u marks as gateway if it has two unconnected neighbors.
4. Whenever there is a newly marked gateway, the newly marked gateway host and its gateway neighbors apply *Rule 1* and *Rule 2* to reduce the number of gateway hosts.

Mobile host u recognizes a broken link $\{u, v\}$: When a mobile host u recognizes broken link $\{u, v\}$, two types of mobile hosts need to recalculate their gateway status. One is mobile host u itself if it is a gateway host; the other ones are common neighbors of mobile hosts u and v. For example, in Figure 9 (a), neighbors of non-gateway host u are all pairwise connected and they remain so after link $\{u, v\}$ is broken. On the other hand, in Figure 8 (b), depending on the

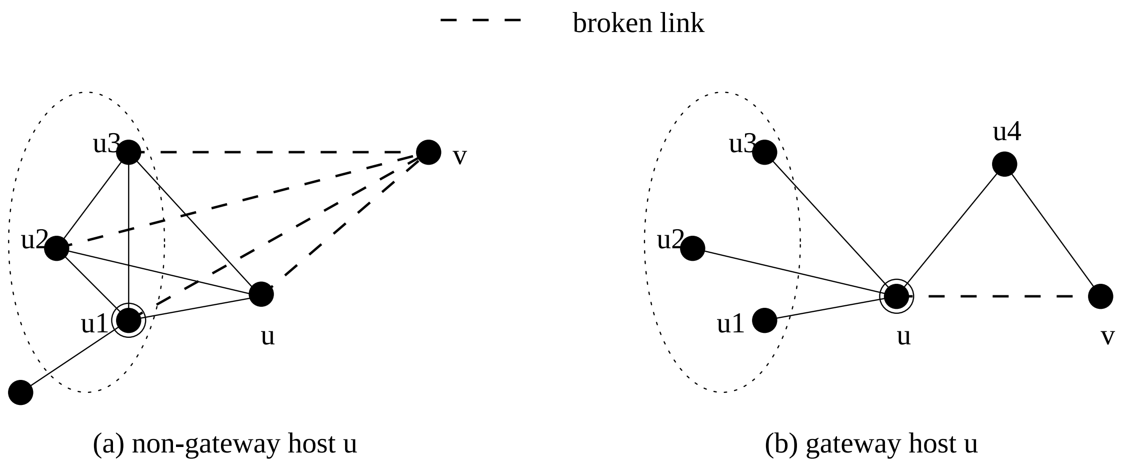

Figure 9: Mobile host u recognizes broken link $\{u, v\}$

connections between neighbors u_1, u_2, u_3, and u_4, gateway host u may unmark itself as non-gateway after link $\{u, v\}$ is broken. In the same figure, mobile hosts u and v's common non-gateway neighbor u_4 may mark itself as gateway after link $\{u, v\}$ is broken.

The corresponding update process can be the following:

1. Mobile host u detects a broken link to v, and it exchanges the open neighbor set with all its neighbors.

2. If host u is non-gateway, it simply does nothing further; otherwise, host u will assign its marker $m(u)$ to F if its neighbors are all pairwise connected.

3. Upon receiving the open neighbor set $N(u)$ from host u, non-gateway neighbor (excluding host v) w recalculates its status if it is a common neighbor of hosts u and v.

4. Whenever there is a newly marked gateway, the newly marked gateway host and its gateway neighbors apply *Rule 1* and *Rule 2* to reduce the number of gateway hosts.

4.5 Connected Dominating Set Update II

The above marking processes are effective only if few nodes have new/broken links with host v. When there are many such nodes, a better way of updating gateways can be similar to the one for host switching on/off. During its movement, host v continuously sends out its open neighbor

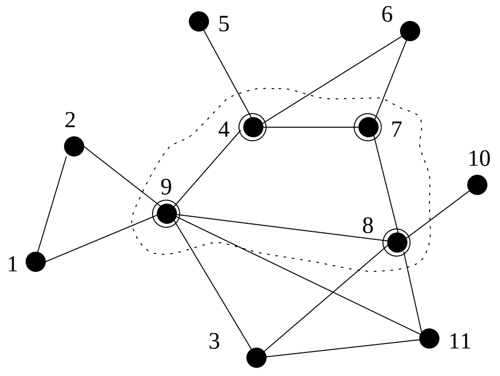

Figure 10: A routing example

set $N(v)$ along with signal {id(v), Heart_Beat}. This triggers each host $w \in v \cup N(v)$ to update it gateway status. The corresponding update process can be the following:

1. Mobile host v periodically exchanges its open neighbor set with all its neighbors at every τ time interval.
2. Each host $w \in v \cup N(v)$ assigns its marker $m(w)$ to T if it has two unconnected neighbors.
3. Whenever there is a newly marked gateway, the newly marked gateway host and its gateway neighbors apply *Rule 1* and *Rule 2* to reduce the number of gateway hosts.

For host u that has a broken link to host v, its gateway status is updated as soon as it recognizes a link disconnection. The corresponding update process can be the following:

1. Mobile host u detects a broken link to v and it exchanges its open neighbor set with all its neighbors.
2. If host u is non-gateway, it simply does nothing further; otherwise, host u assigns its marker $m(u)$ to F if its neighbors are all pairwise connected.

5 Dominating-Set-Based Routing

The routing process in a *dominating-set-based routing* is usually divided into three steps:

3
10
11

(a)

9 (1,2,3,11)	....
4 (5,6)	....
7 (6)	....

(b)

Figure 11: Gateway host routing information: (a) Gateway domain membership list at host 8 and (b) Gateway routing table at host 8

1. If the source is not a gateway host, it forwards the packets to a *source gateway*, which is one of the adjacent gateway hosts.

2. This source gateway acts as a new source to route the packets in the *reduced graph* generated from the connected dominating set.

3. Eventually, the packets reach a *destination gateway*, which is either the destination host itself or a gateway of the destination host. In the latter case, the destination gateway forwards the packets directly to the destination host.

Each gateway host keeps following information: *gateway domain membership list* and *gateway routing table*. Gateway domain membership list is a list of non-gateway hosts which are adjacent to the gateway host. Gateway routing table includes one entry for each gateway host, together with its domain membership list. For example, given an ad hoc wireless network as shown in Figure 10, the corresponding routing information at host 8 are shown as in Figure 11. Figure 11 (a) shows that host 8 has three members 3, 10, 11 in its gateway domain membership list. Figure 11 (b) shows the gateway routing table at host 8, which consists of a set of entries for each gateway together with its membership list. Other columns of this table, including distance and routing information, are not shown.

The way routing tables constructed and updated in the subnetwork generated from the connected dominating set can be different. In the following, we will briefly discuss two extreme cases of routing protocols: *shortest path routing* and *dynamic source routing*.

5.1 Shortest path routing

When a non-gateway host needs to send out a message, it first sends a request to all its source gateways, which are gateway neighbors. Each source gateway, by looking at local routing table, find out a minimum route (connecting the source and destination nodes) among all routes that go through this source gateway. Then the source gateway sends this route back to the source node. The source node will then pick up a global minimum route from all routes sent from source gateways and send the packets to the corresponding source gateway (that generates the global minimum route) as the next hop. The remaining routing process is restricted to the subnetwork generated from the connected dominating set. The gateway routing tables in the subnetwork can be constructed and updated using either *distance vector protocol* or *link state protocol.* For example, in Figure 10, when host 3 needs to send a message to host 5, it first sends a request to its source gateways hosts 8 and 9. Hosts 8 and 9 will look up their local routing tables and sends their minimum routes to source 3. As a result, host 3 will pick up a global minimum route which is $(3, 9, 4, 5)$ in terms of minimum hop count.

5.2 Dynamic source routing

We can also implement the dynamic source routing [16] in the subnetwork generated from the connected dominating set with the following slight modifications. The route request packets will be propagated only within the subnetwork. Also, only gateway hosts can initiate the route discovery procedure. If the source is a non-gateway host, it has maximumly $k_1 \times k_2$ choices of route, on the other hand, if the source is a gateway host, it has maximumly k_2 choice of route, where k_1 is the number of source gateways and k_2 is the number of the destination gateways. The source can pick up any route depends on its transmission criteria. Since a source may have several back up routes, whenever a routing error occurs, instead of initiating a new route discovery procedure immediately, the source gateway will try to use back up routes to transmit packets. Only when all back up routes fail, the source will then initiate a route discovery procedure.

6 Performance Evaluation

In this section, we compare our approaches, for determining a connected dominating set in a static network with and without applying two rules, with Das et al's algorithm in [7] (or simply Das' algorithm), which is based on a classical distributed algorithm [11]. Our comparison was conducted through both analytical study and simulation. Simulation for update/recalculation of the connected dominating set in a dynamic network will be part of our future work.

6.1 Algorithm complexity

In a distributed algorithm, its performance can be measured by computation complexity within each node and communication complexity between nodes. Communication complexity can be measured by the number of rounds needed and the total amount of message exchanges in these rounds.

In our approach, the cost of the marking process (with and without applying two rules) at each vertex is $O(\Delta^2)$, where $\Delta = \max\{|N(v)||v \in V\}$. The total amount of message exchanges is $O(\Delta\nu)$, where $\nu = |V|$ is the total number of vertices in G. In the marking process without using two rules, only one round is needed. In the one applying two rules, two rounds are needed.

On the other hand, Das' algorithm runs $O(\gamma)$ rounds. The overall complexities are $O(\gamma\Delta^2 + \nu)$ in terms of time, where γ is the cardinality of the derived dominating set; $O(\Delta\nu\gamma + m + \nu \log \nu)$ in terms of messages, where m is the cardinality of the edge set.

Clearly, our approach is less complex than Das's algorithm in all measurements, in particular, the number of rounds needed. Note that the number of rounds is an important metrics measuring the performance of the algorithm, because the topology of the ad hoc wireless network changes frequently with the movement of mobile hosts, the dominating set has to be updated and recalculated frequently. Another important measurement is the size of the dominating set generated. This can be done through simulation discussed in the following subsection.

6.2 Simulation

In order to evaluate the performance of our proposed algorithm, we conducted a simulation study for initial gateway calculation. We want to measure the size of the dominating set generated from the marking process and compare it with the one from generated from Das' algorithm. Specifically, simulation is done for three algorithms: the proposed algorithm without applying two rules, the proposed algorithm applying two rules, and Das' algorithm [7].

The simulation is performed using the following parameters. ν represents the number of mobile hosts in the network, and γ represents the number of gateways (the size of the dominating set) in the network, r represents the radius of mobile host's transmission area, New1 is the number of gateway nodes calculated by our algorithm without applying two rules, New2 is the number of gateway nodes calculated by our algorithm with two rules, and MCDS is the number of gateway nodes calculated by Das's algorithm [7].

Random graphs are generated in a 100×100 square units of a 2-D simulation area, by randomly throwing a certain number of mobile hosts. A 2-D simulation area resembles more an actual ad hoc wireless network where mobile hosts usually stay on ground. Assume that each mobile host has the same transmission radius, thus the generated graph is undirected. Then, set the radius of mobile host's transmission area to r. If the distance between any two nodes is less than radius r, then there is a link connection between these two nodes. The connected dominating set is calculated through the following steps.

1. Using a depth-first search algorithm (BFS) to examine if the generated graph is connected or not. If the graph is disconnected, simply discard the graph; otherwise, continue the following step.

2. Applying three different algorithms to the generated random graph and calculate New1, New2, MCDS, respectively.

More precisely, we performed two groups of simulation. In the first group, we first generate random graphs according to the graph generation procedure above. Then, set the radius of the mobile host's transmission area r to four different values: 15, 25, 50, 75. In this way, we can control the density of generated graphs, since the density of generated graphs increases as r increases. If the distance between any two nodes is less than r, then there is a link connection between these two nodes and the resultant graph is a complete graph. For each r, we also vary the number of mobile hosts ν from 0 to 100. For each ν, generate a random connected graph 1000 times. Calculate New1, New2, and MCDS for each case; and at the end, we simply take the average of New1, New2, MCDS. In the second group, we also generate random graphs according to the graph generation procedure described above. Then, set the number of mobile hosts ν in the network to four different values: 20, 40, 60, 80. For each ν, we vary the radius of the mobile host's transmission area r from 0 to 100. For each r, generate a random connected graph 1000 times. Calculate New1, New2, and MCDS for each case, and then, take the average of New1, New2, MCDS. We compare our approaches with Das' algorithm in [7] in terms of number of gateways generated. In general, the less number of gateways, the better the result, because our objective is to generate a small connected dominating set to facilitate a fast routing process.

Figures 12 (a),(b),(c),(d) show the number of gateways versus the number of nodes in the

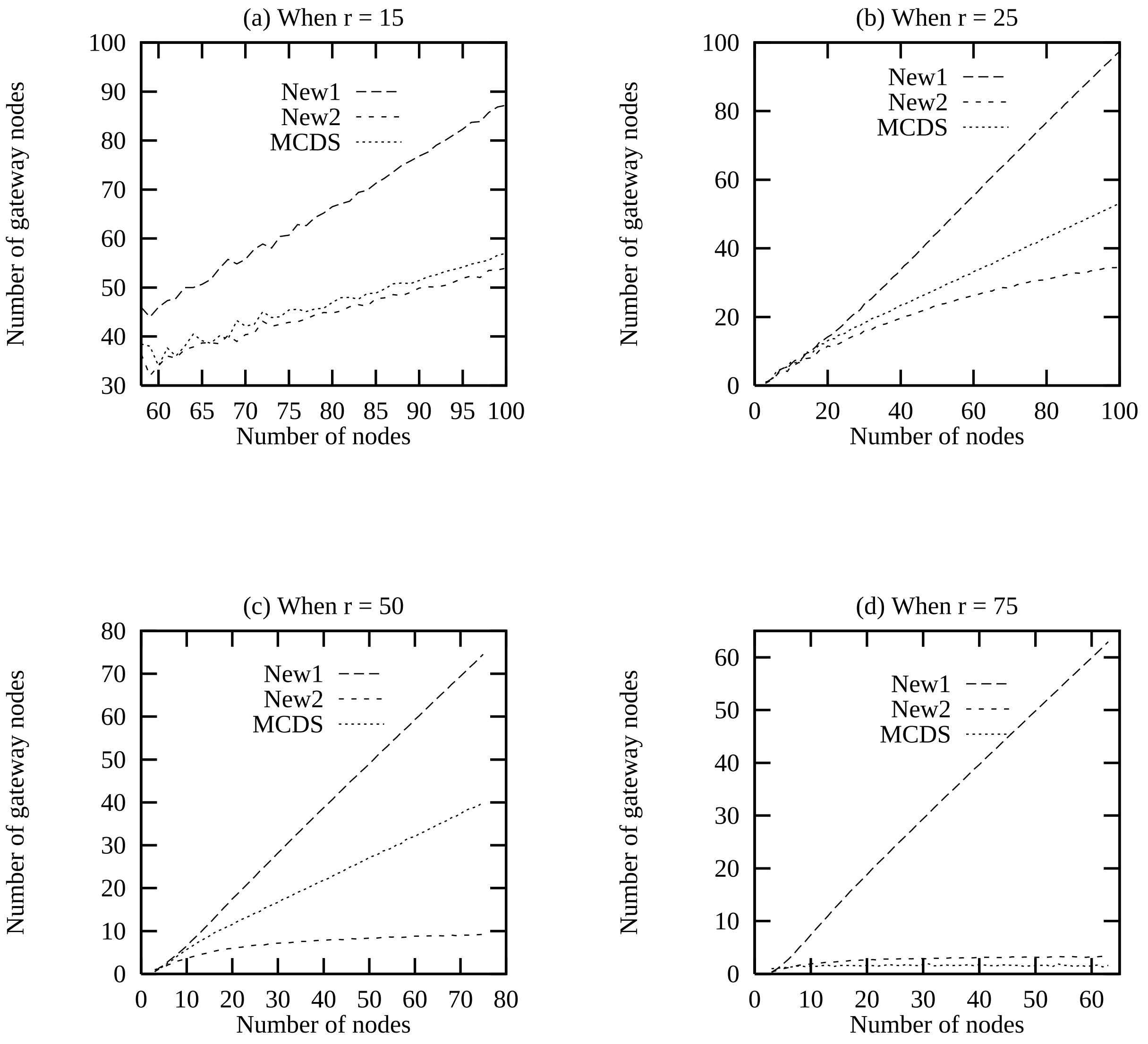

Figure 12: Average number of gateway nodes relative to the number of nodes ν.

network for the increasing order of radius r. We can see that without applying two rules, the performance of our algorithm (the curve for New1) fares poorly. However, by applying two rules, the performance of our approach (the curve for New2) is much better than the one (the curve for MCDS) derived by applying Das' algorithm [7], when the radius of mobile host's transmission area is neither too small nor too large, for example, $r = 15, 25, 50$. We can see the gap between New2 and MCDS increases as r increases. When r=75, our approach (with applying two rules) is outperformed by Das' algorithm. In order to fully understand this sudden change of relative performance between these two approaches, we conducted the second group of simulation.

Figures 13 (a),(b),(c),(d) show the number of gateway nodes with respect to radius r for the increasing order of number of nodes ν. We can see that the number of gateway nodes decreases smoothly as the radius r increases using the proposed algorithm with applying two rules; however, using Das' algorithm the number of gateway nodes remains almost unchanged until the radius r reaches around 60, then it suddenly decreases to stay under the curve for New2. Although for a large radius r, Das' algorithm shows better performance, the difference between New2 and MCDS is no more than 2 gateways.

We have the following summary from simulation results.

- Our approach with applying two rules consistently outperforms Das' algorithm, as long as the wireless transmission range is not too large (with respect to geographical distribution areas of mobile hosts).
- Our approach without applying two rules fares poorly and generates the largest dominating set for any situations among these approaches.
- Das' algorithm outperforms our approach only when the wireless transmission range is very large. On the other hand, their difference, in terms of the number of gateways generated, is at most by 2 gateways. Also, in reality it is unlikely that an ad hoc wireless network forms a dense graph which is close to a complete graph.

7 Conclusions

In this paper, we have proposed a simple and efficient distributed algorithm for calculating connected dominating set in the ad-hoc wireless network. A simulation study has been conducted to compare our proposed algorithm with Das' algorithm [7] in terms of the size of connected dominating set generated. When the mobile host's transmission radius is not too large, the proposed

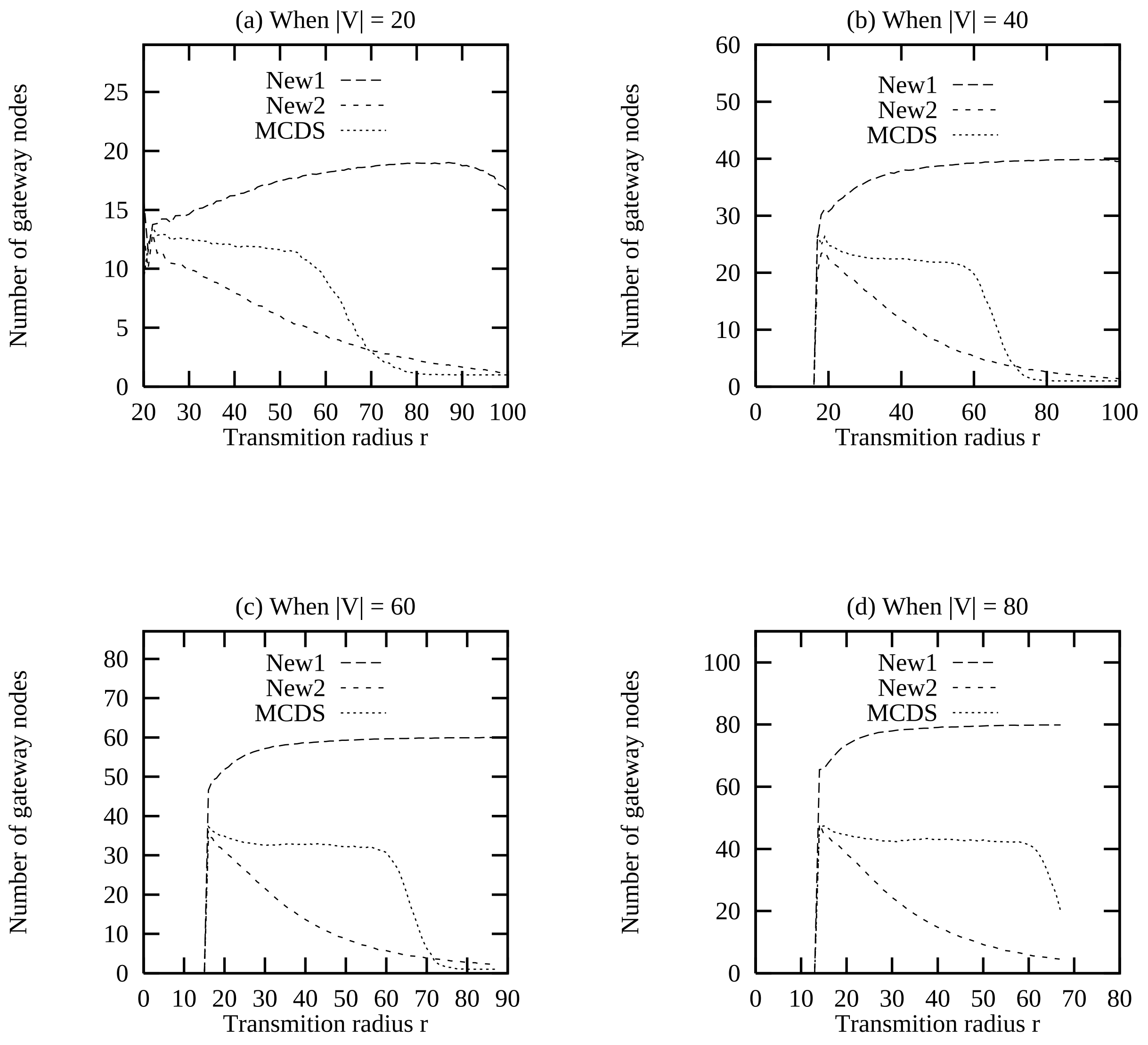

Figure 13: Average number of gateway nodes relative to radius r.

algorithm generates a smaller connected dominating set. Our proposed algorithm calculates connected dominating set in $O(\Delta^2)$ time with distance-2 neighborhood information, where Δ is the maximum node degree in the graph. In addition, the proposed algorithm uses constant (1 or 2) rounds of message exchanges, compared with $O(\gamma)$ rounds of message exchanges in Das' algorithm, where γ is the size of the dominating set. A reduced graph can be generated from the connected dominating set and the searching space for a routing process can be reduced to this reduced graph. Shortest path routing and dynamic source routing have been applied to illustrate this approach.

The future work will extend the proposed algorithm to the ad-hoc wireless networks in which mobile hosts have different transmission radii. Another future research direction is to apply the proposed approach repeatedly to subgraphs generated from connected dominating sets to form a hierarchy of connected dominating sets.

References

[1] D. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall Inc., second edition, 1992.

[2] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Proc. of the 3rd Int'l Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 48–55, 1999.

[3] I. Castineyra, N. Chiappa, and M. Steenstrup. The nimrod routing architecture. *RFC 1992*, August 1996.

[4] C. Cheng, R. Riley, S. P. R. Kumar, and J. J. Garcia-Luna-Aceves. A loop free Bellman-Ford routing protocol without bouncing effect. *ACM SIGCOMM '89*, pages 224 – 237, 1989.

[5] M. S. Corson and A. Ephremides. A distributed routing algorithm for mobile wireless networks. *ACM J. Wireless Networks*, 1(1):61 – 81, 1995.

[6] B. Das and V. Bhargavan. Routing in ad-hoc networks using minimum connected dominating sets. *IEEE International Conference on Communications (ICC '97)*, June 1997.

[7] B. Das, E. Sivakumar, and V. Bhargavan. Routing in ad-hoc networks using a virtual backbone. *Proceedings of the 6th International Conference on Computer Communications and Networks(IC3N '97)*, pages 1 – 20, Sept. 1997.

[8] B. Das, R. Sivakumar, and V. Bhargavan. Routing in ad-hoc networks using a spine. *IEEE International Conference on Computers and Communications Networks '97*, 1997.

[9] E. Gafni and D. P. Bertsekas. Distributed algorithms for generating loop-free routes with frequently changing topology. *IEEE Transactions on Communications*, COM-29(1):11 – 18, 1981.

[10] J. J. Garcia-Luna-Aceves. A unified approach to loop-free routing algorithm using distance vector or link states. *Proceedings of ACM SIGCOMM Symposium on Communication, Architecutures and Protocols*, pages 212 – 213, September 1989.

[11] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374 – 387, April 1998.

[12] T. W. Haynes, Hedetniemi S. T., and P. J. Slater. *Funcamentals of Domination in Graphs.* A Sireis of Monographs and Text books. Marcel Dekker, Inc., 1998.

[13] C. Hedrick. Routing information protocol. *Internet Request For Comments RFC 1058*, June 1988.

[14] J. M. Jaffe and F. H. Moss. A responsive distributed routing algorithm for computer networks. *IEEE Transactions on Communications*, COM-30(7):1758 – 1762, 1979.

[15] D. B. Johnson. Routing in ad hoc networks of mobile hosts. *Proceedings of Workshop on Mobile Computing Systems and Applications*, pages 158 – 163, December 1994.

[16] D. B. Johnson and D. A. Malts. Dynamic source routing in ad-hoc wireless networks. In T. Imielinski, H. Korth, editor, *Mobile Computing.* Kluwer Academic Publishers, 1996.

[17] J. Jubin and J. D. Tornow. The DARPA packet radio network protocols. *Proceedings of the IEEE*, 75(1):21 – 32, January 1987.

[18] H. Koch, H. Krombholz, and O. Theel. A breif introduction to the world of mobile computing. Technical Report Tech. Rep. THD-BS-1993-03, Computer Science Department, University of Darmstadt, Germany, 1993.

[19] P. Krishna, M. Chatterjee, N. H. Vaidya, and D. K. Pradhan. A cluster-based approach for routing in ad-hoc networks. *Proceedings of the Second USENIX Symposium on Mobile and Location-Independent Computing*, pages 1 – 10, 1995.

[20] J. M. McQuillan, I. Richer, and E. C. Rosen. The new routing algorithm for ARPANET. *IEEE Transactions on Communications*, 28(5):711 – 719, 1980.

[21] J. M. McQuillan and D. C. Walden. The ARPA network design decisions. *Computer Networks*, 1(5):243 – 289, August 1977.

[22] P. M. Merlin and A. Segal. A fail safe distributed routing protocol. *IEEE Transactions on Communications*, COM-27(9):1280 – 1287, 1979.

[23] J. Moy. OSPF version 2. *Internet Request For Comments RFC 1247*, July 1991.

[24] S. Murthy and J. J. Garcia-Luna-Aveces. A routing protocol for packet radio networks. *Proceedings of ACM International Conference on Mobile Computing and Networking*, pages 86 – 95, November 1995.

[25] M. R. Pearlman and Z. J. Hass. Determining the optimal configuration for the zone routing protocol. *IEEE Journal On Selected Areas in Communications*, pages 1395 – 1399, August 1999.

[26] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers. *Computer Communications Review (ACM SIGCOMM 1994)*, 24(4):234 – 244, 1994.

[27] M. Schwartz and T. E. Stern. Routing techniques used in communication networks. *Proceedings of IEEE INFOCOM*, pages 218 – 226, May 1987.

[28] R. Sivakumar, B. Das, and V. Bharghavan. An improved spine-based infrastructure for routing in ad hoc networks. *Proceedings of the International Symposium on Computers and Communications (ISCC'98)*, 1998.