An Extended Localized Algorithm for Connected Dominating Set Formation in Ad Hoc Wireless Networks *

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

Abstract

Efficient routing among a set of mobile hosts 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 search space for a route is reduced to the hosts in the set. A set is dominating if all the hosts in the system are either in the set or neighbors of hosts in the set. The efficiency of dominating-set-based routing mainly depends on the overhead introduced in the formation of the dominating set and the size of the dominating set. In this paper, we first review a localized formation of a connected dominating set called *marking process* and dominating-set-based routing. Then we propose a dominant pruning rule to reduce the size of the dominating set. This dominant pruning rule (called Rule k) is a generalization of two existing rules (called Rule 1 and Rule 2 respectively). We prove that the vertex set derived by applying Rule k is still a connected dominating set. Rule k is more effective in reducing the dominating set derived from the marking process than the combination of Rules 1 and 2, and surprisingly, in a restricted implementation with local neighborhood information, Rule k has the same communication complexity and less computation complexity. Simulation results confirm that Rule k outperforms Rules 1 and 2, especially in networks with relatively high vertex degree and high percentage of unidirectional links. We also prove that an upper bound exists on the average size of the dominating set derived from Rule k in its restricted implementation.

Keywords: *Ad hoc wireless networks, dominant pruning, dominating sets, routing, probabilistic analysis, simulation.*

---

1 Introduction

An ad hoc wireless network, or simply ad hoc network, can be represented by a *unit disk graph* [6], where every vertex (host) is associated with a disk centered at this vertex with the same radius (also called transmission range). Two vertices are neighbors (i.e., there is an edge between them) if and only if they are covered by each other's disk. For example, both vertices v and w in Figure 1 (a) are neighbors of vertex u because they are covered by disk u; while vertices v and x in Figure 1 (b) are not neighbors because their disks cannot cover each other. In an ad hoc network, some links (edges) may be unidirectional due to either the disparity of energy levels of hosts or the hidden terminal problem [21]. Therefore, a general ad hoc network can be considered as a *general disk graph* with both bidirectional and unidirectional links.

Routing protocol design is one of the challenging issues in ad hoc networks. Among various existing routing protocols, *dominating-set-based routing* [9, 20, 24, 25] is a promising approach. This approach was first proposed for undirected graphs only using the notion of *dominating set* [9, 25] and was later extended to cover directed graphs by introducing another notion called *absorbent set* [24]. A subset of vertices in an undirected 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 ease of the routing process within the induced graph of dominating vertices. The main advantage of dominating-set-based routing is that it simplifies the routing process to the one in a smaller subnetwork generated from the connected dominating set (CDS). Only dominating vertices (also called *gateways*, as shown in Figure 1 as doubly-cycled vertices) need to keep routing information in a *proactive approach* and the search space is reduced to the dominating set in a *reactive approach*.

Clearly, the efficiency of this approach depends largely on the process of finding and maintaining a CDS and the size of the corresponding subnetwork. It is desirable to find a small CDS without compromising the functionality, reliability, and efficiency of an ad hoc network. In addition, the CDS formation algorithm should be *localized* (i.e., based on local information) for low overhead and fast convergence, two essential requirements for a routing protocol in ad hoc networks. Unfortunately, finding a minimum CDS is NP-complete for most graphs, even if global information is available and no constraint, such as preserving the shortest paths, is enforced.

Wu and Li [24, 25] proposed a simple and efficient localized algorithm that can quickly determine a CDS in ad hoc networks. This approach uses a *marking process* where hosts interact with others in the neighborhood. Specifically, each host is marked true if it has two unconnected neighbors. It is shown that collectively these hosts achieve a desired global objective – a set of marked hosts forms a
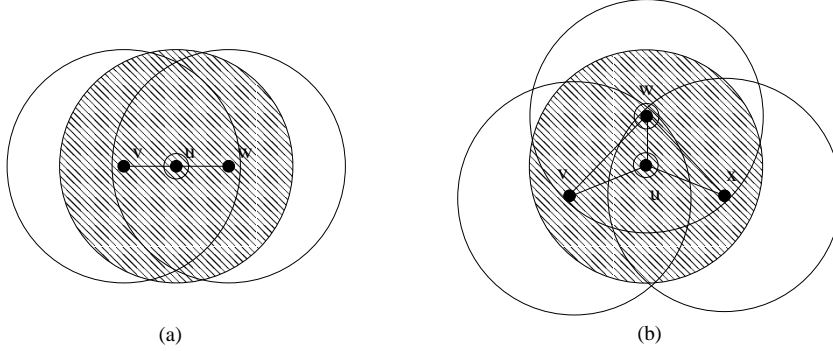
Figure 1: Two examples of ad hoc networks without unidirectional links.

small CDS. In Wu and Li's approach [24, 25], the resultant dominating set derived from the marking process is further reduced by applying two *dominant pruning rules*. According to dominant pruning Rule 1, a marked host can unmark itself if its neighbor set is covered by another marked host; that is, if all neighbors of a gateway are connected with each other via another gateway, it can relinquish its responsibility as a gateway. In Figure 1 (b), either u or w can be unmarked (but not both). According to Rule 2, a marked host can unmark itself if its neighborhood is covered by two other directly connected marked hosts. The combination of Rules 1 and 2 is fairly efficient in reducing the number of gateways while still maintaining a CDS. This approach also outperforms several classic approaches in ad hoc networks, such as the cluster approach [10, 15], in terms of finding a small CDS, and MCDS (minimum connected dominating set) [13, 20], in terms of doing so quickly [24].

Dominant pruning rules with more than two covering hosts were not considered in early study due to the following two assumptions: (1) testing the coverage of multiple hosts could be costly, and (2) only a few hosts' neighbor sets need to be covered by three or more other hosts. However, further study in this paper will show that these assumptions are not always true. In this paper, we propose a generalized dominant pruning rule, called Rule k, which can unmark gateways covered by k other gateways, where k can be any number. We also show that Rule k can be implemented in a restricted way with local neighborhood information that has the same complexity as Rule 1, and surprisingly, less complexity than Rule 2.

Note that a gateway that can unmark itself according to Rule k is not necessarily "unmarkable" according to Rules 1 and 2. For example, suppose hosts in Figure 1 are evenly distributed and very dense. It is almost impossible to find two hosts v and w to cover the neighborhood of host u (see the shadowed area in Figure 1 (a)). However, it is much easier to find three or more hosts to cover the same shadowed area (see Figure 1 (b)). Simulation results of this paper show that Rule k is better than

the combination of Rules 1 and 2 in terms of generating a small CDS. Rule k is especially suitable for ad hoc networks with relatively high density (more than 10 neighbors for each host) and considerable percentage (10–20%) of unidirectional links, where its superiority over Rules 1 and 2 is obvious.

Like Rules 1 and 2, Rule k does not guarantee a constant approximation ratio; however, we show the existence of a "probabilistic bound" on the size of the CDS derived from Rule k. Suppose in a random unit disk graph, the CDS derived from Rule k is R times as large as the minimum CDS; the upper bound of R is also called approximation ratio. We prove that (1) the probability that R is infinitely large is very small, specifically, $Pr(R > x) < \alpha e^{-\beta x}$, and (2) the average value of R is upper bounded by a constant. We also show the same results for the restricted Rule k in unit disk graphs and the non-restricted Rule k in general disk graphs. To the best of our knowledge, this is the first bound given to a pure localized algorithm without resorting to location information. We believe that our proof can be extended to other localized algorithms.

The remainder of this paper is organized as follows: Section 2 summarizes related work. Section 3 reviews the formation of a CDS using the marking process for directed graphs, which include the general disk graph as a special case, two dominant pruning rules, and basic ideas of dominating-set-based routing. Section 4 gives the definition of Rule k and an efficient implementation; the performance analysis is also given in this section. Section 5 discusses the impact of several implementation issues and possible optimizations. Section 6 presents simulation results on the pruning efficiency of Rule k on the general disk graph. The paper concludes in Section 7. The proof of the probabilistic bound is in the appendix.

2   Related Work

Algorithms that construct a CDS in ad hoc networks can be divided into two categories: *centralized* algorithms that depend on network-wide information or coordination and *decentralized* that depend on local information only. Centralized algorithms usually yield a smaller CDS than decentralized algorithms, but their application is limited due to the high maintenance cost. Das et al. [9] proposed a centralized algorithm to find a small CDS. This algorithm is based on Guha and Khuller's first approximation algorithm [13], which can be viewed as the process of growing a spanning tree T in several sequential rounds. In the first round, a vertex with the maximum vertex degree is selected as the root of T. In each following round, a vertex v in T that has the maximum number of neighbors not in T is selected. Selecting v also adds edges to T from v to all its neighbors not in T. Finally, a spanning tree is constructed and the non-leaf vertices form a CDS. This so called *MCDS algorithm* has

an $O(\log \Delta)$ approximation ratio in regular graphs, where Δ is the maximum number of neighbors of a vertex. Another algorithm based on a spanning tree was proposed by Wan et al [23]. In this scheme, a maximal independent set (MIS) is elected such that each vertex in the MIS can be connected to the spanning tree via an extra vertex. Since in unit disk graphs, the size of an independent set is at most 4 times that of the minimum CDS, this algorithm has an approximation ratio of 8. However, this algorithm usually produces a larger CDS than the MCDS algorithm in random unit disk graphs.

Decentralized algorithms can be further divided into cluster-based algorithms and pure localized algorithms. Cluster-based algorithms have a constant approximation ratio in unit disk graphs and relatively slow convergence ($O(n)$ in the worst case). Pure localized algorithms take constant steps to converge, produce a small CDS on average, but have no constant approximation ratio. A cluster-based algorithm usually contains two phases. In the first phase, the network is partitioned into clusters, and a *clusterhead* is elected for each cluster. In the second phase, clusterheads are interconnected to form a CDS. Several clustering algorithms have been proposed [4, 10, 12, 15] to elect clusterheads that have the minimal id, maximal degree, or maximal weight. A host v is a clusterhead if it has the minimal id (or maximal degree or weight) in its 1-hop neighborhood. A clusterhead and its neighbors form a cluster and these hosts are *covered*. The election process continues on uncovered hosts, and finally all hosts are covered. The resultant set of clusterheads is an MIS. Kwon and Gerla [14] proposed passive clustering (PC) to reduce the control overhead. In PC, the control information is piggybacked in normal packets, and neighbors compete to be the clusterhead in a first-come-first-serve manner.

Several approaches were proposed to construct a CDS by connecting clusterheads via non-clusterheads called *connectors*; that is, both clusterheads and connectors are gateways here. In early schemes [3, 15], every non-clusterhead that has a neighbor in another cluster is designated as a connector, which results in a larger CDS. The objective here is to maximize the throughput and reliability, rather than to reduce the CDS size. Alzoubi et al [2] proposed growing a tree to reduce the number of connectors. The root of this tree is the winner of a distributed election among clusterheads, and other clusterheads are connected to the tree via at most two connectors per clusterhead. This algorithm is an early version of [23]; it has an approximation ratio of 12 and a slow converging speed. Most approaches [10, 14, 26] use a mesh structure, which is much faster to construct than a tree. In the mesh scheme, each clusterhead designates one or two connectors to form a path to each neighboring clusterhead (i.e., a clusterhead 2 or 3 hops away). The mesh scheme also has a constant approximation ratio, but this constant is much larger than 12.

In pure localized algorithms [1, 5, 16, 19, 24, 25], the status of each node depends on its h-hop topology only, where h is a small constant, and usually converges after at most h rounds of information

exchange among neighbors. Chen et al [5] proposed an approach similar to the marking process, called Span, to select a set of special hosts called *coordinators*. Ideally coordinators form a CDS such that other hosts can switch to the energy saving mode without compromising the routing capability of the network. A host v becomes a coordinator if it has two neighbors that are not directly connected, indirectly connected via one intermediate coordinator, or indirectly connected via two intermediate coordinators. Before a host changes its status from non-coordinator to coordinator, it waits for a backoff delay which is computed from its energy level and 2-hop neighborhood topology. The backoff delay can be viewed as a priority value, such that nodes with shorter backoff delay have a higher chance of becoming coordinators. Span cannot ensure a CDS when two coordinators simultaneously change back to non-coordinators. We use in the simulation an enhanced version of Span, where a host becomes a coordinator if it has two neighbors that are not directly connected or indirectly connected via one or two intermediate coordinators with higher priority values. This enhanced Span uses 3-hop information and takes three rounds to converge.

Qayyum et al [16] proposed an efficient broadcast scheme called mutipoint relaying (MPR). In MPR, each host designates a small set of 1-hop neighbors (MPRs) to cover its 2-hop neighbors. In the broadcasting, a host u forwards a packet p from the last hop v only if (1) u has not received p before, and (2) u is a MPR of v. For each broadcasting, forwarding hosts form a *source-dependent CDS* (i.e., a dynamic CDS depends on the broadcast process). By taking advantage of the broadcast history information, a source-dependant CDS is usually smaller than a *source-independent CDS* constructed by above algorithms. It was proved in [16] that MPRs selected by a single host has $\log \Delta$ approximation ratio. However it is unknown if a global approximation ratio exists for the entire CDS. Tseng et al [22] proposed several efficient broadcasting schemes for ad hoc networks, but none of them forms a CDS.

3   Preliminaries

In this section, we review the marking process and two dominant pruning rules that reduce the size of a dominating set [24], and give a brief description of a routing scheme based on CDS.

3.1   Localized dominating set formation

Given a simple directed graph $G = (V, E)$, where V is a set of vertices (hosts) and E is a set of directed edges (unidirectional links), a directed edge from u to v is denoted by an ordered pair (u, v). If (u, v) is an edge in G, we say that u dominates v and v is an absorbent of u. The *dominating neighbor*

set $N_d(u)$ of vertex u is defined as $\{w : (w, u) \in E\}$. The *absorbent neighbor set* $N_a(u)$ of vertex u is defined as $\{v : (u, v) \in E\}$. $N(u) = N_d(u) \cup N_a(u)$ represents the neighbor set of vertex u. For example, in Figure 2 (a), vertex x dominates vertex u, y is an absorbent of u, and v is a dominating and absorbent neighbor of u. The dominating neighbor set of vertex u is $N_d(u) = \{v, x\}$, the absorbent neighbor set of u is $N_a(u) = \{v, y\}$, and the neighbor set of u is $N(u) = \{v, x, y\}$. The general disk graph and unit disk graph are special cases of directed graphs.

A set $V' \subset V$ is a *dominating set* of G if every vertex $v \in V - V'$ is dominated by at least one vertex $u \in V'$. Also, a set $V' \subset V$ is called an *absorbent set* if for every vertex $u \in V - V'$, there exists a vertex $v \in V'$ which is an absorbent of u. For example, vertex set $\{u, v\}$ in Figures 2 (a) and (b) and $\{u, v, w\}$ in Figure 2 (c) are both dominating and absorbent sets of the corresponding directed graphs. In this paper, unless otherwise specified, we use the term "(connected) dominating set" to represent "(strongly connected) dominating and absorbent set". The following marking process can quickly find a strongly connected dominating and absorbent set in a given directed graph.

Algorithm 1 Marking process [24]

1: Initially assign marker F to each u in V.

2: Each u exchanges its neighbor set $N_d(u)$ and $N_a(u)$ with all its neighbors.

3: u changes its marker $m(u)$ to T if there exist vertices v and w such that $(w, u) \in E$ and $(u, v) \in E$, but $(w, v) \notin E$.

The marking process is a localized algorithm, where hosts only interact with others in the neighborhood. Unlike clustering algorithms, there is no "sequential propagation" of information. The marking process marks every vertex in G. $m(v)$ is a marker for vertex $v \in V$, which is either T (marked) or F (unmarked). Suppose the marking process is applied to the network represented by Figure 2 (a), host u will be marked because $(x, u) \in E$ and $(u, y) \in E$, but $(x, y) \notin E$; host v will also be marked because $(u, v) \in E$ and $(v, z) \in E$, but $(u, z) \notin E$. All other hosts will remain unmarked because no such pair of neighbor hosts can be found. For the same reason, only hosts u and v in Figure 2 (b) and hosts u, v, and w in Figure 2 (c) will be marked by the marking process. Assume that V' is the set of vertices that are marked T in V; that is, $V' = \{v : v \in V \wedge m(v) = T\}$. The induced graph G' is the subgraph of G induced by V'; that is, $G' = G[V']$. Wu [24] showed that marked vertices form a strongly connected dominating and absorbent set, and furthermore, can connect any two vertices with minimum hops.

An important issue in implementing the marking process is how to collect the neighbor set information. For each host u, its dominating neighbor set, $N_d(u)$, can be established by monitoring the *beacon*
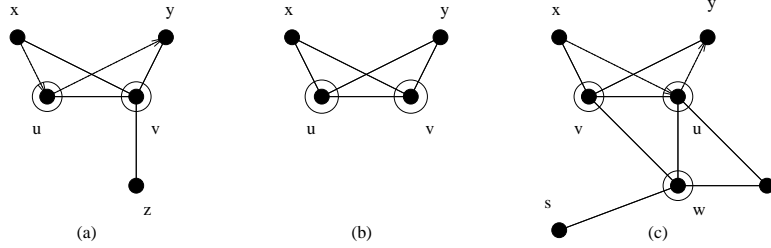
Figure 2: Three examples of dominating set reduction.

packet sent periodically by each dominating neighbor. Its absorbent neighbor set, $N_a(u)$, however, cannot be established in this way. Suppose host v is an absorbent neighbor of u; that is, $v \in N_a(u)$ but $v \notin N_d(u)$, u will not receive the 1-hop beacon sent by v and, therefore, cannot recognize v as its neighbor. This problem is handled in [24] by the means of k-hop beacons: If host v finds out it is an absorbent neighbor of host u, but currently not in u's neighbor set, host v will broadcast a beacon packet to notify u of its existence. Each broadcast packet has a TTL (time-to-live) value set to k to limit its propagation range. Simulation results in [24] show that with 2-hop beacon packets, more than 99.9% of the absorbent neighbors can be detected in a random network with an average node degree of 18 and 20% unidirectional links.

3.2 Dominating set reduction

In the marking process, a vertex is marked T because it may be the only connection between its two neighbors. However, if there are multiple connections available, it is not necessary to keep all of them. We say a vertex is covered if its neighbors can reach each other via other connected marked vertices. Two dominant pruning rules are proposed in [25] and then extended in [24] to reduce the size of the connected dominating set. The idea is the following: If a vertex is covered by no more than two connected vertices, removing this vertex from V' will not compromise its functionality as a CDS. To avoid simultaneous removal of two vertices covering each other, a vertex is removed only when it is covered by vertices with higher id's. Node id $id(v)$ of each each vertex $v \in V$ serves as a priority. Nodes with high priorities have high probability of becoming gateways. Id uniqueness is not necessary, but equal id's will produce more gateways.

Rule 1: *Consider two vertices u and v in G'. If $N_d(u) - \{v\} \subseteq N_d(v)$ and $N_a(u) - \{v\} \subseteq N_a(v)$ in G and $id(u) < id(v)$, change the marker of u to F; that is, G' is changed to $G' - \{u\}$.*

Rule 2: *Assume that v and w are bidirectionally connected in G'. If $N_d(u) - \{v, w\} \subseteq N_d(v) \cup N_d(w)$*

and $N_a(u) - \{v, w\} \subseteq N_a(v) \cup N_a(w)$ in G and $id(u) < \min\{id(v), id(w)\}$, then change the marker of u to F.

In Figure 2 (a), since $N_d(u) - \{v\} \subseteq N_d(v)$, $N_a(u) - \{v\} \subseteq N_a(v)$ and $id(u) < id(v)$, vertex u is removed from V' and vertex v is the only dominating vertex in the graph. In Figure 2 (b), u and v cover each other, but only u is removed from V' because $id(u) < id(v)$. In Figure 2 (c), since $N_d(u) - \{v, w\} \subseteq N_d(v) \cup N_d(w)$, $N_a(u) - \{v, w\} \subseteq N_a(v) \cup N_a(w)$, and $id(u) < \min\{id(v), id(w)\}$, vertex u can be removed from V' based on Rule 2. It is proved in [24] that the reduced set $V'_* \subseteq V'$ generated from applying Rule 1 and/or Rule 2 to V' is still a strongly connected dominating and absorbent set of G. If vertex u in Rule 1 and vertices u and w in Rule 2 are neighbors of vertex v, the corresponding dominant pruning rules are called the *restricted* Rule 1 and Rule 2; otherwise, they are *non-restricted*.

3.3 Dominating-set-based routing

Assume that a CDS has been determined for a given ad hoc network. *Dominating-set-based routing* usually consists of three steps: (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 in its absorbent set. (2) This source gateway acts as a new source to route the packets in the *induced 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 in the dominating neighbor set of the destination host. In the later case, the destination gateway forwards the packets directly to the destination host.

There are in general two ways to perform routing within the induced graph: proactive routing and reactive routing. In [25], DSDV is used as a sample proactive routing to illustrate the dominating-set-based routing. In reactive routing protocols such as DSR and AODV, a connected dominating set can serve as a *forward node set* to forward routing request (RREQ) packets.

4 Dominant Pruning Through k-Neighbor Coverage

In this section, we propose a generalized dominant pruning rule (called Rule k). The new Rule k can reduce the size of a dominating set at least as much as Rules 1 and 2 do. We will show that a restricted version of Rule k can be implemented in a localized way with the same complexity as the restricted version of Rule 1, and surprisingly, with less complexity than Rule 2.
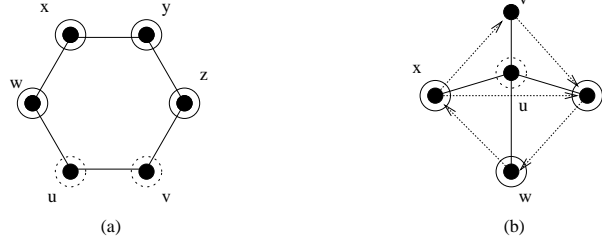
Figure 3: Limitation of Rules 1 and 2 in networks (a) without or (b) with unidirectional links.

4.1 Generalized pruning rule

Assume $G' = (V', E')$ is the induced subgraph of a given directed graph $G = (V, E)$ from marked vertex set V'. In the following dominant pruning rule, we use $N_d(V'_k)$ $(N_a(V'_k))$ to represent the dominating (absorbent) neighbor set of a vertex set V'_k; that is, $N_d(V'_k) = \bigcup_{v_i \in V'_k} N_d(v_i)$ and $N_a(V'_k) = \bigcup_{v_i \in V'_k} N_a(v_i)$.

Rule k: *Assume that $V'_k = \{v_1, v_2, ..., v_k\}$ is the vertex set of a strongly connected subgraph in G'. If $N_d(u) - V'_k \subseteq N_d(V'_k)$ and $N_a(u) - V'_k \subseteq N_a(V'_k)$ in G and $id(u) < \min\{id(v_1), id(v_2), \cdots, id(v_k)\}$, then change the marker of u to F.*

Rules 1 and 2 are special cases of Rule k where $|V'_k|$ is restricted to 1 and 2, respectively. Note that V'_k may contain two subsets: V'_{k_1} that really covers u's neighbor set, and V'_{k_2} that acts as the glue to make them a connected set. Obviously, if a vertex can be removed from V' by applying Rule 1 or Rule 2, it can also be removed by applying Rule k. On the other hand, a vertex removed by Rule k is not necessarily removable via Rule 1 or Rule 2. For example, in Figure 3 (a), both vertices u and v can be removed using Rule k (for $k \geq 3$) because they are covered by vertices w, x, y, and z; in Figure 3 (b), vertex u can be removed because it is covered by vertices w, x, and y. Note that although x and y are not bidirectionally connected, they can reach each other via vertex w. However, none of these vertices can be removed via Rule 1 or Rule 2.

Theorem 1 *If V' is a strongly connected dominating and absorbent set of a directed graph G, and V'_R is the set of vertices removable under Rule k, then $V'_* = V' - V'_R$ is also a strongly connected dominating and absorbent set of G.*

Proof: First we prove V'_* is a dominating set. This claim holds when $|V'| = 1$, because $V'_* = V'$. If $|V'| > 1$, for every vertex u in G, it is either in V' or not in V'. If $u \notin V'$, it is dominated by at least one vertex in V', because V' is a dominating set of G. If $u \in V'$, it is also dominated by
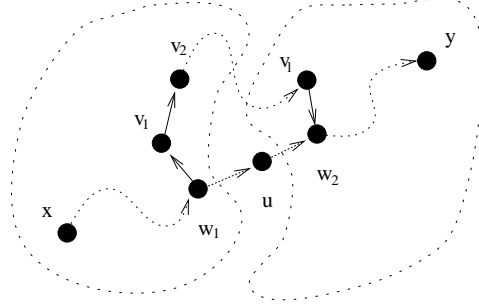
10

Figure 4: An impossible case: u is the first vertex that re-connects the partitioned network, but any path from x to y through u can detour via other vertices with higher id's.

a vertex in V', because V' is strongly connected. In addition, there always exists a vertex $v \in V'$ satisfying $id(v) = \max\{id(w) : w \in N_d(u)\}$, which cannot be removed by applying Rule k. Therefore u is dominated by at least one vertex $v \in V'_*$. By analogy we can prove V'_* is also an absorbent set.

Then we prove $G[V'_*]$ is strongly connected. Suppose $G[V'_*]$ is not strongly connected, if we put back the removed vertices one by one in descending order of vertex id's, we shall find the first vertex u that "re-connects" V'_*; that is, after the removal of u, at least one pair of vertices (x, y) in $G[V']$ loses its last connecting path. However, this is impossible: If u is removed from V' by applying Rule k, its dominating and absorbent neighbor sets are covered by a strongly connected set of vertices with higher id's than $id(u)$. As we can see in Figure 4, for any (x, y)-path through u, there always exists another (x, y)-path with the following three segments: (1) from source x to vertex w_1 before u, (2) from w_1 to the vertex after u, w_2, through vertices $v_1, v_2, \cdots, v_l$ covering u, and (3) from w_2 to destination, which is not through u. Therefore, removal of u cannot eliminate all (x, y)-paths, which is a contradiction. $\square$

4.2 An efficient pruning algorithm

Similar to restricted Rules 1 and 2, if $v_1, v_2, \ldots, v_k$ are all neighbors of u in Rule k, the corresponding dominant pruning rule is called the restricted Rule k; otherwise, it is non-restricted. In the non-restricted dominant pruning rules, a host can be covered by a group of hosts 1 or 2 hops away, self-connected or connected by other marked hosts. For example, hosts u and v in Figure 3 (a) and u in Figure 3 (b) can unmark themselves via the non-restricted Rule k, but only host u in Figure 3 (b) can unmark itself via the restricted Rule k. Host v in Figure 3 (a) cannot unmark itself because one of the covering hosts, w, is not a neighbor of v. The restricted Rule k is easier to implement,

because it demands only 2-hop neighborhood information. The non-restricted Rule k demands global information, which is quite unrealistic in ad hoc networks. Our simulation shows that the number of hosts unmarked by restricted and non-restricted rules are very close.

Algorithm 2 Restricted k-dominant pruning (executed on each marked host $u \in V'$)

1: Broadcasts its id and marker $(id(u), T)$ to all its neighbors.
2: Builds a subgraph $G[V'_+]$, where $V'_+ = \{w | w \in (V' \cap N(u)) \wedge (id(u) < id(w))\}$.
3: Computes the set of strongly connected components $\{V'_1, V'_2, \cdots, V'_l\}$ of $G[V'_+]$.
4: Changes its marker $m(u)$ to F if there exists $V'_i, 1 \le i \le l$, such that $N_d(u) - V'_i \subseteq N_d(V'_i)$ and $N_a(u) - V'_i \subseteq N_a(V'_i)$.

Algorithm 2 gives an implementation of the restricted Rule k. This procedure is invoked only when the current host is marked T by the marking process. First, all marked hosts advertise their id's to their neighbors (step 1). By collecting the advertised information, each marked host can build the set V'_+ of marked neighbors with higher id's and the induced graph $G[V'_+]$ that includes all those neighbors (step 2). Because during the marking process, each host has collected the information of its neighbors and links among its neighbors, $G[V'_+]$ can be built without further information exchange. Then the condition of Rule k is tested and a marked host is unmarked if the rule applies (steps 3 and 4). Note that the computation in steps 3 and 4 is based on local information and does not involve inter-host communication.

In step 3, each host decomposes the induced graph of its marked neighbor set with higher id's, V'_+, into several *strong components*. The strong components [7] of a directed graph are the equivalence classes of vertices under the "mutually reachable" relation. Two vertices of V'_+ belong to the same strong component if and only if they are strongly connected in $G[V'_+]$. For example, the directed graph in Figure 5 has three strong components: $\{t, v, x\}$, $\{w\}$, and $\{y, z\}$. A directed graph is strongly connected if it has only one strong component. Note that although we always assume that G' is a strongly connected graph, $G[V'_+]$ is not necessarily strongly connected. For any marked host u, if it can be unmarked by applying the restricted Rule k, it must be covered by a subset of a strong component, V'_i, which also covers u. If u is not covered by any V'_i, it cannot be covered by any strongly connected vertex set. Therefore, it is not necessary to test the coverage of every combination of u's marked neighbors: *Testing every strongly connected component shall be sufficient.*

Several linear-time algorithms can decompose a directed graph into strong components [7, 17]. They are all based on the *depth-first search* (DFS) algorithm and have a complexity of $O(|E| + |V|)$. A DFS process grows a *DFS tree* from a given starting vertex (root). All vertices reachable from a
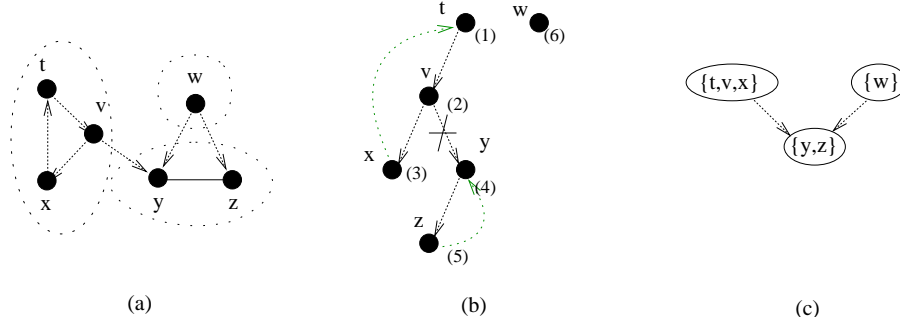
Figure 5: After applying depth-first search (DFS) on a directed graph (a), a DFS forest including two DFS trees is generated, where vertices are numbered with the visiting order (b). A branch is pruned from the first DFS tree below node v. Finally, the directed graph is decomposed into three strong components (c).

root are *visited* (i.e., added to the DFS tree) in pre-order. After the construction of a DFS tree, if there are still vertices *unvisited* (i.e., unreachable from root), one unvisited vertex is selected as the root to grow another DFS tree. This process continues until all vertices are visited. Each visited vertex u is labeled with an ordering number $ord(u)$; that is, for any two visited vertices $u, v \in V$, $ord(u) < ord(v)$ if and only if u is visited before v. Figure 5 (b) shows the result of a DFS process starting from vertex t (i.e., $ord(t)=1$). Note that different DFS processes may have different order assignments. Each DFS tree contains one or several strong components. The following algorithm, originally proposed by Gabow [11], is considered as the most efficient algorithm that partitions a DFS tree into strong components. Algorithm 3 utilizes two stacks: Stack A stores *visited but unsettled* vertices (i.e., their strong components are still open for new joiners) in the ascending order of ord. Vertices in stack $A = [v_1 v_2 \ldots v_n]$ are partitioned into several sections $[S_1 S_2 \ldots S_m]$, where each S_i is a subsequence of $[v_1 v_2 \ldots v_n]$ with consecutive elements. Vertices in the same section are strongly connected with each other, and the first vertex of each section is stored in stack B in the ascending order of ord. Initially, both stacks A and B are empty, vertices enter and leave these stacks during the DFS process, and finally, both stacks are empty again when the algorithm terminates.

There are two key operations in the above algorithm: merging several sections into one larger section and closing a section to form a strong component. A newly visited vertex u is itself a section S_m (step 1). If this vertex has a link to another section S_i with smaller ord in stack A, sections $S_i, S_{i+1}, \ldots, S_m$ are merged into one section S_i (step 2). When u's descendants in the DFS tree are visited, u's section may be further merged into more sections (step 3). u's section is closed when all its descendants in the DFS tree have been visited; that is, no more merge is possible. Therefore, u's section

13

Algorithm 3 Sc-Dfs(u)

1: Push u into A and B.

2: For each *visited but unsettled* absorbent neighbor v of u, pop B until $ord(top(B)) \leq ord(v)$.

3: While there exists an *unvisited* absorbent neighbor v of u, recursively call Sc-Dfs(v).

4: If $top(B) = u$, pop u out of B, and pop A until u is out of A. The newly *settled* vertices (those popped out of A) form a strong component.

---

is popped out of stack A and forms a strong component (step 4). For example, corresponding to the DFS forest in Figure 5 (b), the status of stack A after Sc-Dfs is applied on each vertex is: (1) $[t^*]^1$, (2) $[t^*v^*]$, (3) $[t^*v^*x^*] \rightarrow [t^*vx]$ (x merges sections t, v), (4) $[t^*vxy^*]$, (5) $[t^*vxy^*z^*] \rightarrow [t^*vxy^*z] \rightarrow [t^*vx]$ (section yz is closed at vertex y) $\rightarrow []$ (section tvx is closed at vertex t), (6) $[w^*] \rightarrow []$ (section w is closed immediately). Note that Algorithm 3 generates the same set of strong components when a different DFS process is applied (and a different DFS forest is generated).

The correctness of Algorithm 3 lies in the facts that (1) only mutually reachable vertices are put in the same section, (2) all vertices mutually reachable via visited vertices are in the same section, and (3) if a vertex cannot reach any ancestor in the DFS tree after all its descendants have been visited, then it cannot reach any unvisited vertex either. Note that in step 2 of Algorithm 3, facts (1) and (2) hold even if the directed edge (u, v) points to another branch in the DFS tree. Suppose an extra edge (z, x) is added to the directed graph in Figure 5, the status of stack A after vertex z is visited becomes: (5) $[t^*vxy^*z^*] \rightarrow [t^*vxy^*z] \rightarrow [t^*vxyz] \rightarrow []$, still giving the correct result.

4.3 Performance analysis

This subsection discusses the efficiency and overhead of Rule k. First we give an upper bound on the average size of the dominating set derived from Rule k in unit disk graphs. Let V'_* be the connected dominating set derived from Rule k, and V'_{opt} be the minimal connected dominating set. We define the *efficiency ratio* $R = |V'_*|/|V'_{opt}|$. It is clear that there is no upper bound on R. In the worst case, $|V'_*|$ is proportional to $|V|$, when vertices in V are placed in a 2-D space, and their coordination satisfies that $id(u) > id(v)$ if and only if $(x_u, y_u) > (x_v, y_v)$. As the network density grows, $|V'_*|$ grows while $|V'_{opt}|$ is upper bounded by $O(S_V/r^2)$, where S_V is the area of the 2-D space. Here R can be infinitely large. Fortunately, this is not the average case.

---

[1] A vertex with superscript "*" is also stored in stack B and marks the beginning of a section in stack A.

Theorem 2 *If Rule k is applied on a unit disk graph where vertices are randomly and uniformly distributed in a rectangular region, then there exist constants $\alpha, \beta, \gamma > 0$, such that $Pr(R > x) < \alpha e^{-\beta x}$ and $E[R] \leq \gamma$.*

Theorem 2 is proved in the appendix for both restricted and non-restricted Rule k. This theorem states that (1) the probability that R is infinitely large is very small, and (2) the average value of R is upper bounded by a constant that is independent of network size and density. For general disk graphs, suppose $\lambda = r_{max}/r_{min}$ is the ratio between the maximal disk radius r_{max} and the minimal disk radius r_{min}, the following theorem is also proved in the appendix.

Theorem 3 *If the non-restricted Rule k is applied on a general disk graph where vertices are randomly distributed in a rectangular region with a given λ, then there exist $\alpha_\lambda, \beta_\lambda, \gamma_\lambda > 0$, such that $Pr(R > x) < \alpha_\lambda e^{-\beta_\lambda x}$ and $E[R] \leq \gamma_\lambda$.*

Next we show that the restricted Rule k has the same complexity as the restricted Rule 1 and less complexity than the restricted Rule 2. The computation complexity of the restricted Rule 1 for each marked host is $O(\Delta^2)$, because a host compares its neighbor set with Δ neighbors in the worst case, and the neighbor set comparison has a complexity of $O(\Delta)$. The complexity of the restricted Rule 2 for each marked host is $O(\Delta^3)$, because a host compares its neighbor set with $\Delta(\Delta - 1)/2$ pairs of marked neighbors in the worst case. The following theorem shows that the complexity of restricted Rule k is $O(\Delta^2)$, same as the restricted Rule 1, better than the restricted Rule 2.

Theorem 4 *The computation complexity of Algorithm 2 is $O(\Delta^2)$, where Δ is the maximum vertex degree in the network.*

Proof: The complexity of Algorithm 2 can be derived from the complexity of its steps. Obviously the complexity of step 1 is $O(\Delta)$. The complexity of step 2 is $O(\Delta^2)$, because subgraph $G[V'_+]$ has at most Δ vertices, each with at most Δ links. The complexity of step 3 is $O(\Delta^2)$. Because it has been proven in [7] that a/ graph $G = (V, E)$ can be decomposed into strong components with $O(|E| + |V|)$ complexity (every vertex is visited only once in the depth-first search), and for $G[V'_+]$ which contains marked neighbors with higher id's, $|E| \leq \Delta^2$ and $|V| \leq \Delta$. The complexity of step 4 is also $O(\Delta^2)$. Note that each vertex in V'_+ ($V'_+ \leq \Delta$) contributes at most two neighbor set subtractions in step 4, and the complexity of each substraction is $O(\Delta)$. Overall, the computation complexity of Algorithm 2 is $O(\Delta^2)$. $\qquad\square$

Contrary to intuition, Rule 2 is theoretically slower than Rule k. Even when it is known that vertex u can be covered by k $(k > 3)$ of its marked neighbors, it still cannot decide whether u can be covered by any two of them. Therefore, the neighbor set of each $v \in V'_+$ needs to be compared with u's neighbor set many times in Rule 2, but only once in Rule k. Nevertheless, the actual difference of execution time is hard to observe, if the network is relatively sparse, or made sparse via clustering or power control techniques. The following theorem shows that the restricted Rule k has the same communication overhead and latency (in terms of the rounds of information exchange) as the restricted Rules 1 and 2.

Theorem 5 *In bi-directional networks, the combination of the marking process and restricted Rule k takes 3 rounds to complete. Each host sends at most 1 message of $O(\Delta)$ bits.*

Proof: The 2-hop information used by the marking process can be collected via two rounds of information exchanges. In round 1, each host advertises its id and builds its 1-hop neighbor set based on the advertisement of its neighbors. In round 2, each host advertises its 1-hop neighbor set and identifies links among its 1-hop neighbors. After the marking process, each marked host advertises its marker in round 3. The restricted Rule k is applied based on the 2-hop information and the list of marked neighbors. In rounds 1 and 3, each host sends a $O(1)$ message; in round 2, each host sends a $O(\Delta)$ message. □

5   Implementation Issues

5.1   Mobility

Topology changes caused by mobility are handled in a localized way by the marking process and Rule k. Basically each host is sensitive to four types of topological changes: a new neighbor appears (host-on), an old neighbor disappears (host-off), two neighbors move close enough to each other (link-on), and two neighbors move far enough from each other (link-off). When a topological change is detected by a host, the marking process and restricted Rule k is applied to compute the new status of this host. For any host, the marking process can only be triggered by changes within 1 hop (host-on/off) and 2 hops (link-on/off). The restricted Rule k can only be triggered by changes within 1 hop (host-on/off), 2 hops (link-on/off), and 3 hops (status change of neighbor hosts). Therefore, the propagation range of any topological change is no more than 3 hops.

The above bound of propagation range can be reduced to 2 hops by slightly altering the restricted

Rule k algorithm. Algorithm 2 is still correct if we remove step 1 and make a subtle change of step 2 to "Build a subgraph $G[V'_+]$, where $V'_+ = \{w|w \in N(u) \wedge (id(u) < id(w))\}$ is u's neighbor set with higher id's". If u's neighbor set is covered by V_i, which is a strong component of its neighbors, then $V'_i = V' \cap V_i$ is also strongly connected, because any vertex that connects two otherwise separated vertices must be marked by the marking process. Furthermore, V'_i covers u's neighbor set, because any vertex that connects a covered vertex and vertices in V'_i must be marked by the marking process. The altered algorithm depends only on the link state within its 2-hop neighborhood, and therefore, will not be affected by any topological change more than 2-hop away. The altered algorithm has lower communication cost and converges faster, but it has higher computation cost because of the larger V'_+.

The computation cost can be reduced by handling each type of topological change differently. For example, for the host-on events, if the current host is marked and the new neighbor has a lower id, it is easy to tell that the current host will still be marked. No computation is needed. If the current host is unmarked, and the new neighbor is covered by the last V'_i, which covers the old neighbor set, the current host can remain unmarked. The drawback of this approach is the complicated updating algorithm. Usually, we assume that saving computation power is less critical than saving communication bandwidth and fast convergence. Therefore, a compute-from-scratch scheme with the altered Rule k algorithm is appropriate. If the computation power is the bottleneck, and the average vertex degree is large, an incremental updating scheme with the original algorithm is appropriate.

5.2 Restricted implementation based on h-hop neighborhood information

For restricted implementations of Rule 1, Rule 2, or Rule k, collecting 2-hop information ($N(u)$ and $N(w), \forall w \in N(u)$) is sufficient and only partial 2-hop information ($N(u)$ and $N(w) \cap N(u), \forall w \in N(u)$) is actually used. For non-restricted implementations, Rule 1 still needs 2-hop information, Rule 2 needs 3-hop information, and Rule k needs information of the entire network, which is impractical for ad hoc networks. However, Algorithm 2 can be extended to use 2- to 3-hop neighbors to cover u's neighbor set.

Coverage based on 2-hop information can be computed by changing step 2 of Algorithm 2 to include 2-hop neighbors $N(N(u))$ into V'_+. This extension requires no extra communication cost, but has higher computation cost. However, the 2-hop information collected in step 1 does not include edges between any two 2-hop neighbors. Unidirectional links cause another problem: If v is the downstream host of unidirectional link (u, v), u will not see those 2-hop neighbors connected with v. Rule k based on 2-hop information can cover non-restricted Rule 1 and some non-restricted Rule 2.
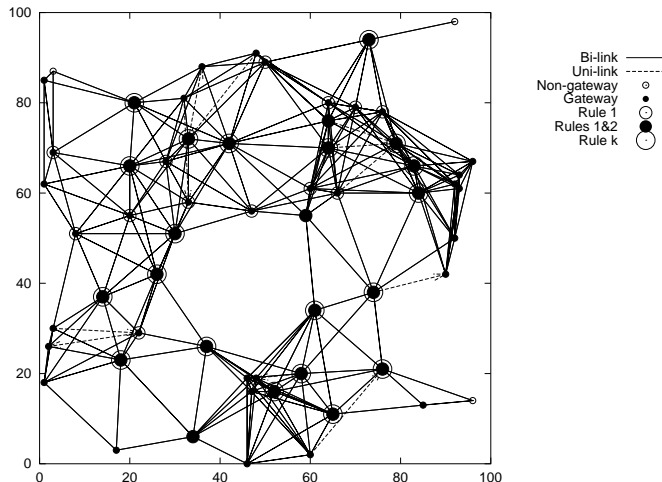
Figure 6: An ad hoc network generated by *ds*. There are three hosts unmarked by Rule k, but not by Rules 1 and 2.

Coverage based on 3-hop information can be computed by changing steps 1 and 2 of Algorithm 2 to first exchange 2-hop neighborhood information with neighbors and then include 3-hop neighbors $N(N(N(u)))$ into V'_+. A benefit of 3-hop information collection is that some unidirectional links can be detected as a by-product. Rule k based on 3-hop information can cover both non-restricted Rule 1 and Rule 2. The main drawback of this extension is the increased communication cost. In order to collect 3-hop information, 2-hop information ($O(\Delta^2)$), instead of merely a list of neighbors ($O(\Delta)$), is exchanged among neighboring hosts. Simulation study in the next section shows that results of the restricted, 2-hop, and 3-hop implementations of Rule k are very close. Therefore, unless special reason exists, we adopt the restricted implementation.

6   Simulation

We conducted a simulation study to compare the performance of Rule k and several existing algorithms that construct a connected dominating set. All algorithms are simulated on a custom simulator *ds* [8]. To generate a random ad hoc network, n hosts (with pre-assigned unique id's 1 to n) are randomly placed in a confined square area. For a given transmission range r, a wireless link is added between each pair of hosts that has a distance smaller than r. Note that for a constant r, the network density, in terms of the average vertex degree d, will increase rapidly as the network size (n) increases. In most scenarios, r is adjusted as n increases to maintain a constant d, such that the impact of network size can

be observed independent of density. In order to observe the impact of network density, each simulation is repeated on both relatively sparse ($d = 6$) and relatively dense ($d = 18$ or 30) networks. The marking process and various dominant pruning rules are simulated on both directed and undirected networks. In directed networks, most wireless links are bidirectional, but a small portion ($p\%$) of them may be randomly designated as unidirectional links. Networks that cannot form a strongly connected graph are discarded. Figure 6 shows a sample network generated by *ds*. All simulations are conducted in static ad hoc networks, where a simulation completes after a CDS formation algorithm converges after several rounds of information exchanges. Each simulation is repeated until the confidence interval of the average result is sufficiently small ($\pm 1\%$, for 90% probability).

First the performance of the restricted Rule k, in terms of the size of the resultant connected dominating set, is compared with a centralized algorithm (MCDS [13]), two cluster-based algorithms (Tree [2] and Mesh [10]), and a pure localized algorithm, i.e., the variation of Span [5] that ensures a connected dominating set. MCDS is a very good approximation to the optimal solution. We use it as a rough estimation to the real minimal connected dominating set, as the brute force method to find the optimal solution is too slow to provide the result for $n > 40$. For the two cluster-based algorithms, Tree is actually a centralized algorithm, as all clusterheads are connected to a global infrastructure (i.e., a tree) controlled from a central point (i.e., the root). This algorithm avoids the redundancy in connecting clusterheads with multiple paths and usually designates fewer gateways than Mesh. In the Mesh method, each clusterhead is connected with every neighboring clusterhead in 3 hops; that is, 3-hop information is collected at each clusterhead. Besides, the cluster structure must be maintained before gateways can be designated. Span, on the other hand depends on 3-hop information only. The pruning rule of Span can be viewed as an extension of non-restricted Rules 1 and 2: if a vertex can be pruned via Rules 1&2, then every pair of its neighbors is connected by one or two vertices with higher priorities (i.e., id's). In a clustering process, either vertex id or vertex degree can be used as a priority value in selecting clusterheads. Those priority types can also be used by the Span variation. Similarly, we can change Rule k and still maintain its correctness, such that a vertex can be pruned if its neighbor set is covered by several connected vertices with higher vertex degrees. Usually, using vertex degree yields a smaller connected dominating set than using vertex id, but takes an extra round to converge. In the remainder of this section, unless otherwise specified, we assume vertex id's are used as priority values.

The upper half of Figure 7 shows the performance of these algorithms when vertex id's are used as priority values. In MCDS, the size of the connected dominating set is about 30% of the network size in sparse ($d = 6$) networks, and 7% in dense ($d = 30$) networks. This performance is much better than other algorithms. Tree has a performance of 50% in sparse networks and 12% in dense networks.
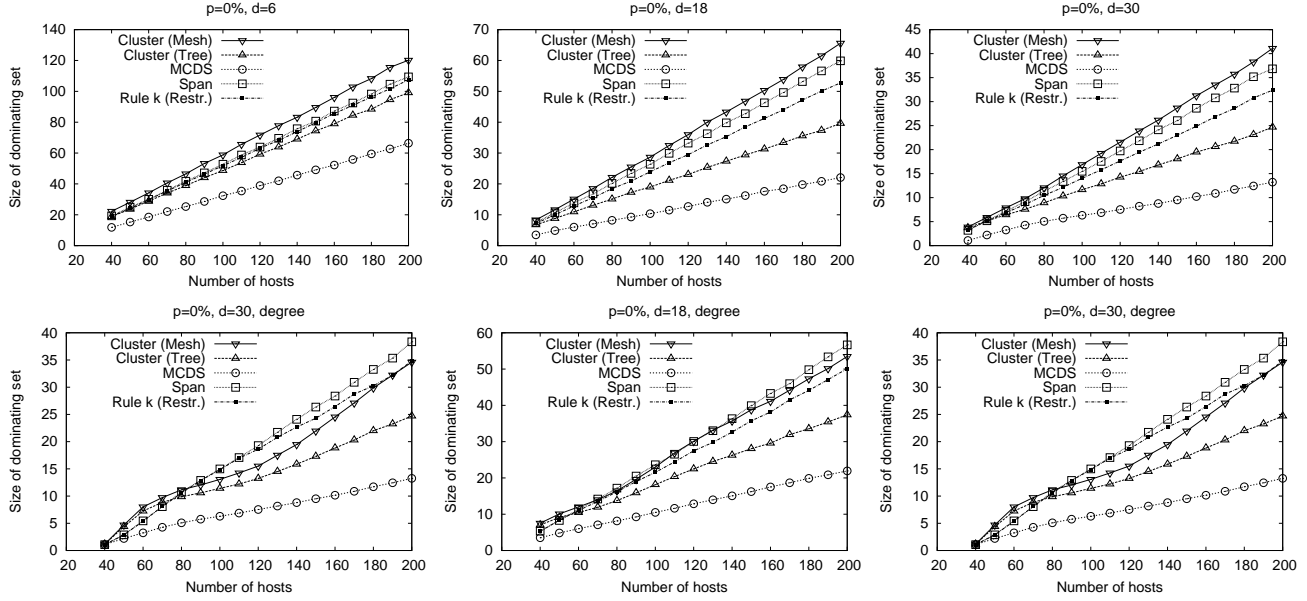
Figure 7: Comparison with existing algorithms.

Mesh produces a dominating set that is about 20% larger than Tree in sparse networks, and about 80% larger in dense networks. The performance of the restricted Rule k is about the average of those of Mesh and Tree. That is, a pure localized algorithm, the restricted Rule k, is actually more efficient than a cluster-based algorithm when vertex id's are used as priority values. Another pure localized algorithm, Span, is also better than Mesh, but is not as good as the restricted Rule k, which implies that the restricted Rule k performs better than non-restricted Rules 1 and 2.

The lower half of Figure 7 shows scenarios when vertex degrees serve as priority values. In sparse networks, the performances of Tree, Span, Rule k, and Mesh are still very close, as in the upper row. The only difference is that Span is slightly better than the restricted Rule k this time. The strength of Rule k is its ability to cover a large neighbor set with more than two high priority vertices, which is not very helpful in a sparse network. On the other hand, Span may take advantage of the 3-hop topology information collected at each host. In dense networks, the restricted Rule k performs better than Span, but occasionally worse than Mesh. However, the difference is very small ($< 10\%$). Tree performs much better (40% difference) than these three algorithms in dense networks. This result is not surprising, as the corresponding connected dominating sets are constructed via network-wide coordination.

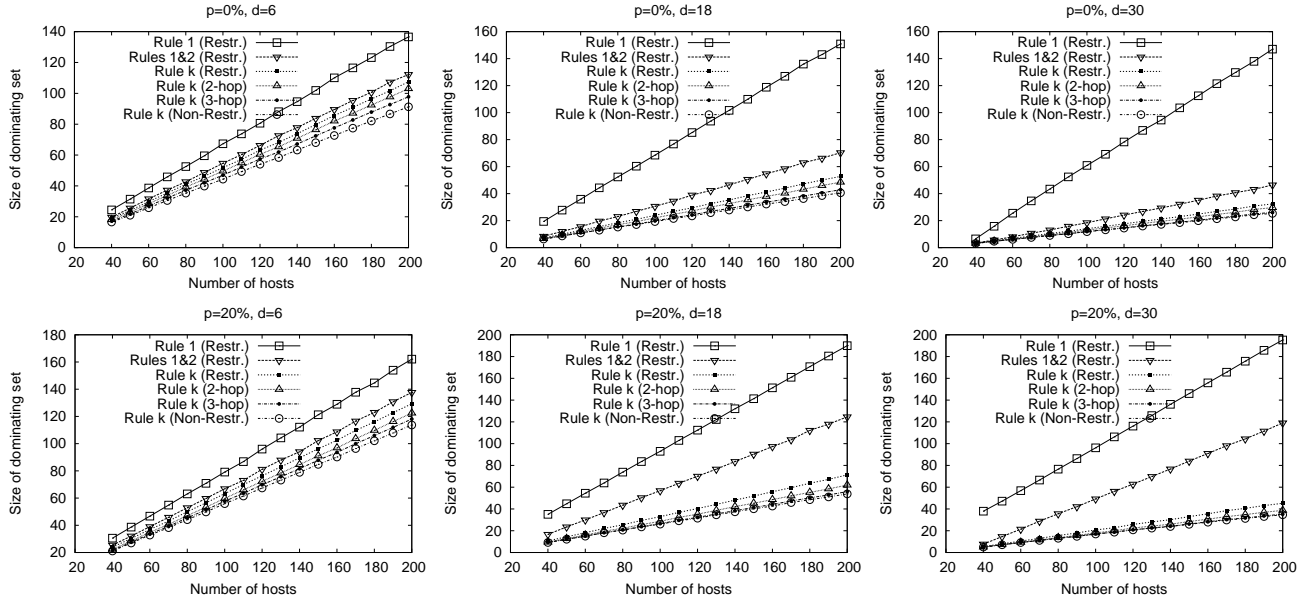The second group of simulations compares performances of different dominant pruning rules, in-

20

Figure 8: Comparison of various dominant pruning rules.

cluding the restricted Rule 1, the combination of restricted Rules 1 and 2, the restricted Rule k, Rule k based on 2-hop and 3-hop information, and the non-restricted Rule k. The restricted Rule k is guaranteed to outperform restricted Rules 1 and 2; the question is how much and under which circumstances Rule k will outperform Rules 1 and 2 significantly. Applying Rule k based on more than 2-hop information will also enhance its pruning performance. In the extreme case, testing Rule k with global information has the highest performance. However, collecting more than 2-hop information is also very expensive and, therefore, should be justified by its contribution to the overall performance.

Figure 8 shows that situations are different in sparse and dense networks. When the network is sparse ($d = 6$), performances of various pruning rules are relatively close. Specifically, Rule 1 alone yields a dominating set about 30%–50% larger than other rules; the combination of restricted Rules 1 and 2 is slightly ($< 5\%$) worse than the restricted Rule k, which in turn, is about 10% worse than the non-restricted Rule k. In dense networks ($d = 18$ or 30), the performance of the restricted Rule 1 is much worse. In such networks, it is nearly impossible for the neighbor set of a vertex to be covered by another vertex, unless in the border area of a network. The combination of restricted Rules 1 and 2 performs much better than the restricted Rule 1 only, but is significantly worse than the restricted Rule k. In undirected graphs ($p = 0\%$), the size of dominating sets derived from the restricted Rule k is about two thirds of that derived from Rules 1 and 2. That is, as illustrated in Figure 1 (b), the neighbor sets of many vertices can be covered only by more than two vertices. In directed graphs

21

($p = 20\%$), the difference is more significant. Restricted Rules 1 and 2 yield a dominating set 100–200% larger than the one produced by the restricted Rule k. This phenomena can be explained with the example in Figure 3 (b). Neighbor sets of many vertices can be covered by two vertices that are connected via a unidirectional link. These vertices cannot be pruned by Rule 2, but can probably be pruned by Rule k. That also explains why the performance of Rule k is about the same in both directed and undirected graphs. Another observation in dense networks is that the contribution of extra neighborhood information becomes trivial as the network becomes denser. Therefore, collecting more than 2-hop information is not appropriate in dense ($d > 18$) networks.

Simulation results can be summarized as follows: (1) The connected dominating set produced by the marking process and the restricted Rule k is about the same size as those produced by the cluster-based schemes, and this is achieved in a localized way without sequential propagation. (2) The restricted Rule k performs slightly better than another pure localized algorithm, Span, with lower cost and a faster converging speed. (3) The restricted Rule k is more efficient than the combination of Rules 1 and 2, restricted or non-restricted, and can be implemented without increasing complexity. (4) Rule k outperforms Rules 1 and 2 significantly in networks with relatively high density and/or high percentage of unidirectional links.

7 Conclusions

A major challenge in dominating-set-based routing it to construct a small connected dominating set, and to do it rapidly in a localized way under communication and computation restraints. Wu and Li [24, 25] have proposed a distributed marking process to rapidly construct a connected dominating set, and then reduce the dominating set with two dominant pruning rules. In this paper, a new dominant pruning rule, Rule k, has been proposed to replace Rule 1 and Rule 2. Given any strongly connected dominating set, if a vertex can be removed by applying Rule 1 or Rule 2, it can also be removed by applying Rule k; a vertex removable by Rule k is not necessarily removable by Rules 1 and 2. An efficient algorithm is proposed to implement the restricted Rule k with the same communication and computation complexity as the restricted Rule 1, and the same communication complexity as the restricted Rule 2, but lower computation complexity than the restricted Rule 2. A constant upper bound is given for the average value of R, the ratio of the size of the dominating set derived from the restricted Rule k to the minimal connected dominating set. We believe this is the first bound given to a pure localized algorithm, and can be applied to other localized algorithms.

Simulation study verifies that the restricted Rule k is a more efficient dominant pruning rule than

the combination of the restricted Rules 1 and 2, especially in dense networks with a relatively high percentage of unidirectional links. For these networks, the resultant dominating set can be greatly reduced by Rule k without any performance or resource penalty. One advantage of the marking process and the dominant pruning rules is their capability to support unidirectional links. For networks without unidirectional links, the marking process and the restricted Rule k is as efficient as several cluster-based schemes and another pure localized algorithm, Span, in terms of the size of the dominating set; this is achieved with lower cost and higher converging speed. Our future research includes performance evaluation of CDS-based routing protocols, and applying the dominant pruning rules to the k-hop dominating set to make dominating-set-based routing more scalable.

References

[1] C. Adjih, P. Jacquet, and L. Viennot. Computing connected dominated sets with multipoint relays. Technical Report 4597, INRIA-Rapport de recherche, Oct. 2002.

[2] K. M. Alzoubi, P.-J. Wan, and O. Frieder. Distributed heuristics for connected dominating sets in wireless ad hoc networks. *Journal of Communications and Networks*, 4(1):22–29, Mar. 2002.

[3] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh. Max-min d-cluster formation in wireless ad hoc networks. *Proc. of IEEE INFOCOM 2000*, pages 32–41, 2000.

[4] S. Basagni. Distributed clustering for ad hoc networks. In *Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN'99)*, pages 310–315, June 1999.

[5] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM Wireless Networks Journal*, 8(5):481–494, Sep. 2002.

[6] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86:165–177, 1990.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* McGraw Hill, 2001.

[8] F. Dai. Dominating set simulation program. http://www.cse.fau.edu/~fdai/adhoc, 2001.

[9] B. Das, R. Sivakumar, and V. Bhargavan. Routing in ad hoc networks using a spine. *Proc. of 6th IEEE Int'l Conf. on Computers Communications and Networks (IC3N '97)*, pages 1–20, Sep. 1997.

[10] A. Ephremides, J. E. Wjeselthier, and D. J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proceedings of IEEE*, 71(1):56–73, 1987.

[11] H. N. Gabow. Path-based depth-first search for strong and biconnected components. *Information Processing Letters*, 74:107–114, 2000.

[12] M. Gerla and J. Tsai. Multicluster, mobile, multimedia radio network. *ACM/Baltzer Journal of Wireless Networks*, 1(3):255–265, 1995.

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

[14] T. J. Kwon and M. Gerla. Efficient flooding with passive clustering (PC) in ad hoc networks. *ACM SIGCOMM Computer Communication Review*, 32(1):44–56, Jan. 2002.

[15] C. R. Lin and M. Gerla. Adaptive clustering for mobile wireless networks. *IEEE Journal on Selected Areas in Communications*, 15(7):1265–1275, 1996.

[16] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast message in mobile wireless networks. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS)*, volume 9, page 298, Jan. 2002.

[17] R. Sedgewick. *Algorithms in C.* Addison-Wesley, 3rd edition, 2002.

[18] M. Shaked and J.G. Shantikumar. *Stochastic Orders and Their Applications.* Academic Press, California, 1994.

[19] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing ad hoc routing with dynamic virtual infrastructures. *Proc. of IEEE INFOCOM 2001*, pages 1763–1772, 2001.

[20] R. Sivakumar, B. Das, and V. Bharghavan. An improved spine-based infrastructure for routing in ad hoc networks. *Proc. of the Int'l Symp. on Computers and Communications (ISCC'98)*, 1998.

[21] A. Tanenbaum. *Computer Networks.* Prentice Hall, Inc., 1996.

[22] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Wireless Networks*, 8(2/3):153–167, Mar.-May 2002.

[23] P.-J. Wan, K. Alzoubi, and O. Frieder. Distributed construction of connected dominating set in wireless ad hoc networks. *Proc. of IEEE INFOCOM'2002*, 3:1597–1604, June 2002.

[24] J. Wu. Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links. *IEEE Transactions on Parallel and Distributed Systems*, 9(3):189–200, Sep. 2002.

[25] J. Wu and H. Li. On calculating connected dominating sets for efficient routing in ad hoc wireless networks. *Proc. of 3rd Int'l Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 7–14, 1999.

[26] J. Wu and W. Lou. Forward-node-set-based broadcast in clustered mobile ad hoc networks. *accepted by Wireless Communications and Mobile Computing*, 2003.

Appendix (Proof of Theorems 2 and 3)

Let R be the efficiency ratio. We need to prove that

$$Pr(R > x) < \alpha e^{-\beta x} \quad \text{and} \quad E[R] < \gamma \tag{1}$$

for both restricted and non-restricted Rule k in unit disk graphs, and for non-restricted Rule k in general disk graphs, where $\alpha, \beta, \gamma > 0$ are constants independent of network size and density. For the sake of clarity, we break our proof into several steps. First we prove (1) with assumptions that (a) G is a unit disk graph, (b) V'_* is derived from the non-restricted Rule k, and (c) vertices in G are randomly and uniformly distributed in a boundless area. Then we extend this proof by removing these assumptions, one by one, and prove that (1) still holds.

Consider a boundless 2-D space, which is partitioned into small square regions (called *cells*) with side $d = r/2\sqrt{2}$ (diagonal line $r/2$) aligned in grid pattern.

Definition 1 *Given a cell C, its minimal coverage region, $C_{min}(C)$, is the intersection of all disks centered within C; its maximal coverage region, $C_{max}(C)$, is the union of all disks centered within C.*

Consider a unit disk graph $G = (V, E)$ with a disk radius r where vertices in V are randomly and uniformly distributed in the boundless 2-D space. Clearly, for any $v \in C$ and $u \in C_{min}(C)$, u is within v's disk, and $(u, v) \in E$. As shown in Figure 9 (a), all eight neighboring cells of C are within $C_{min}(C)$. That is, the $3d \times 3d$ grid can be covered by one vertex in C. Figure 9 (b) shows the $C_{max}(C)$, which is a $(d + 2r) \times (d + 2r)$ square with four rounded corners generated from four $90°$ cones centered at each corner of C with radius r. The area of $C_{max}(C)$ is $\pi r^2 + 4rd + d^2 = (8\pi + 8\sqrt{2} + 1)d^2 < 38d^2$. If
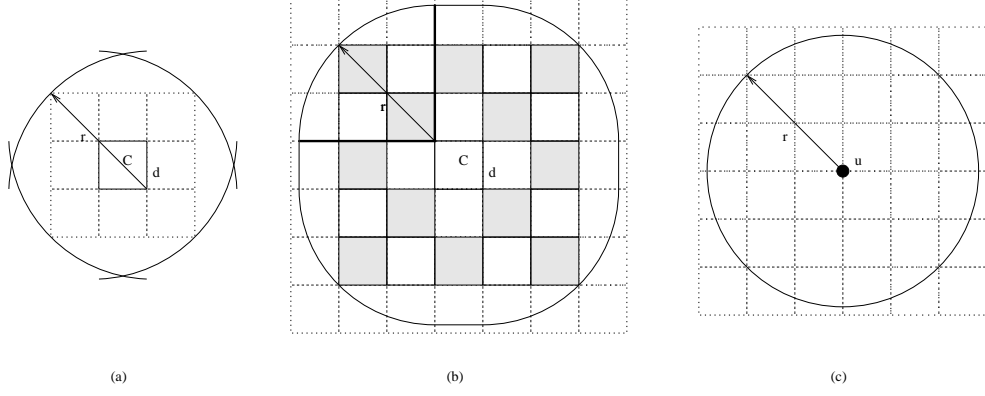
25

Figure 9: Properties of square cells with side $d = r/2\sqrt{2}$: (a) all eight neighboring cells of C are within its minimal coverage region $C_{min}(C)$, (b) the maximal coverage region of C, $C_{max}(C)$ is covered by 49 cells, (c) the unit disk of a vertex in V_{opt}' is covered by 36 cells.

$v \in C$ and $(u,v) \in E$, then $u \in C_{max}(C)$. As shown in Figure 9 (b), $C_{max}(C)$ is contained in a $7d \times 7d$ region consisting of 49 cells. The inner 25 cells contained in $C_{max}(C)$ are *complete*. The probability that any vertex in $C_{max}(C)$ is within a specific complete cell is at least $1/38$.

Lemma 1 *If the non-restricted Rule k is applied on a unit disk graph in a boundless 2-D space, then there exist constants $\alpha_1, \beta_1 > 0$, such that $Pr(R > x) < \alpha_1 e^{-\beta_1 x}$.*

Proof: First we construct a probabilistic "upper bound" for the number of gateways in any cell C, $|V_*' \cap C|$, after Rule k has been applied. Let V_k be the set of k vertices in $C_{max}(C)$ with the largest id's. If every complete cell in $C_{max}(C)$ contains at least one vertex in V_k, then the induced graph $G(V_k)$ is connected and $C_{max}(C)$ is covered by disks of vertices in V_k. According to the non-restricted Rule k, any $v \in V$ located in C that is not in V_k can be pruned from V_*'; that is, $(V_*' \cap C) \subseteq V_k$ and, therefore, $|V_*' \cap C| \leq |V_k| = k$. In other words, if $|V_*' \cap C| > k$, then at least one complete cell has no vertex from V_k. Let A represent "at least one complete cell has no vertex from V_k", and A_i represent "the i-th complete cell in $C_{max}(C)$ has no vertex from V_k", we have $A = A_1 \cup A_2 \cup \ldots \cup A_{25}$ and $Pr(A_i) < (1 - \frac{1}{38})^k = (\frac{37}{38})^k$. Therefore,

$$Pr(|V_*' \cap C| > k) \leq Pr(A) \leq \sum_{i=1}^{25} Pr(A_i) < 25(\frac{37}{38})^k \tag{2}$$

Then we consider the optimal solution $V_{opt}' = \{v_1, v_2, \ldots, v_m\}$. Since every vertex in V is covered by the disk of at least one $v_i \in V_{opt}'$, $|V_*'| \leq \sum_{i=1}^{m} n_i$, where n_i is the number of gateways in v_i's disk. As shown in Figure 9 (c), the disk of each vertex in V_{opt}' can be covered by 36 cells. If we label cells

26

covering v_i's disk as $C_{i,1}, C_{i,2}, \ldots, C_{i,36}$ and let $n_{i,j} = |V'_* \cap C_{i,j}|$, then $n_i \leq \sum_{j=1}^{36} n_{i,j}$. From (2),

$$Pr(n_{i,j} > k) < 25(\frac{37}{38})^k \qquad (i = 1, 2, \ldots, m, \ j = 1, 2, \ldots, 36)$$

and therefore,

$$Pr(|V'_*| > 36km) \leq Pr(\sum_{i=1}^{m}\sum_{j=1}^{36} n_{i,j} > 36km) \leq Pr(n_{i,j} > k, \forall i, j) < 25(\frac{37}{38})^k$$

Note that $R = |V'_*|/|V'_{opt}| = |V'_*|/m$. Let $x = 36k$, we have

$$Pr(R > x) < 25(\frac{37}{38})^{x/36} \tag{3}$$

Let $\alpha_1 = 25, \beta_1 = \ln(\frac{38}{37})/36$, we get $Pr(R > x) < \alpha_1 e^{-\beta_1 x}$ from (3). $\qquad\square$

Note that in Lemma 1, a smaller α_1 and a larger β_1 yield a smaller x under the same probability. However, our focus here is to prove the existence of a "probabilistic bound" rather than to find the tightest one. For example, α_1 can be reduced to 12, because $C_{max}(C)$ can be covered by vertices in 12 gray cells in Figure 9 (b). But that could cause extra complexity in the proof. If we view R as a random variable, then its distribution function is $F_R(x) \geq 1 - \alpha_1 e^{-\beta_1 x}$, and the following lemma show that its average value, $E[R]$, has a constant upper bound.

Lemma 2 *If the non-restricted Rule k is applied on a unit disk graph in a boundless 2-D space, then there exists a constant γ_1, such that $E[R] \leq \gamma_1$.*

Proof: Let R' be a non-negative random variable with distribution function $F_{R'}(x) = 1 - \alpha_1 e^{-\beta_1 x}$, then its density function is

$$f_{R'}(x) = \frac{dF_{R'}(x)}{dx} = \alpha_1 \beta_1 e^{-\beta_1 x}$$

and

$$E[R'] = \int_0^\infty x f_{R'}(x)dx = \frac{\alpha_1}{\beta_1} \tag{4}$$

Since $F_R(x) \geq F_{R'}(x)$ for all $x \geq 0$, $E[R] \leq E[R']$ (strong stochastic ordering [18]). Let $\gamma_1 = \alpha_1/\beta_1$, we get $E[R] \leq \gamma_1$ from (4). $\qquad\square$

Lemma 3 *Lemmas 1 and 2 still hold when vertices of G are randomly and uniformly distributed in a confined rectangular region.*

Proof: Let S be the confined rectangular region where all vertices in V are randomly and uniformly distributed. It is sufficient to prove that (2) still holds when $C_{max}(C)$ is not totally contained in
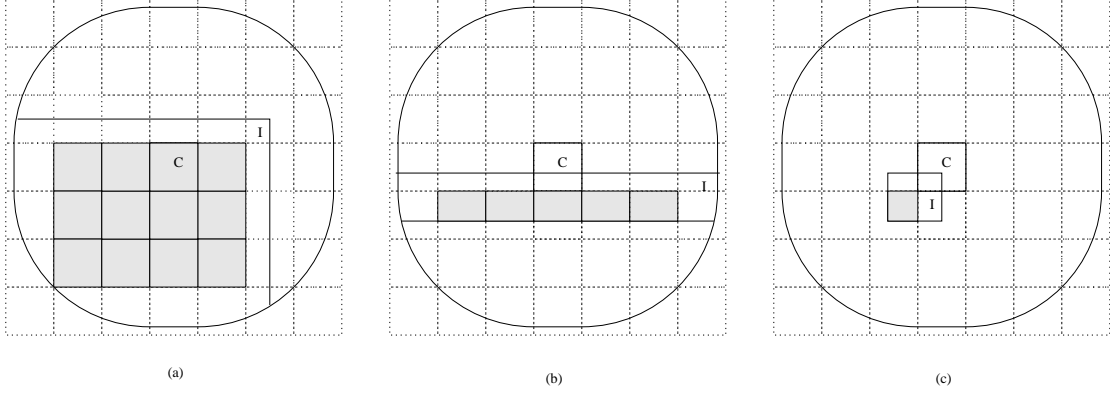
Figure 10: Three types of intersection of $C_{max}(C)$ and a rectangular region. One vertex in each gray cell is enough to cover the intersection region.

S. Here we assume that C is not outside S; otherwise, the number of gateways in C is always 0. Without loss of generality, we assume that the width of S is no less than the height. Let I denote the intersection region of S and $C_{max}(C)$, if C is not outside S, only three cases are possible, as shown in Figure 10.

Case (a): There are complete cells in I. Because every incomplete cell in I has at least one neighboring complete cell, if V_k has at least one vertex in each complete cell, I is covered by V_k. Since there are at most 25 complete cells, and the area of I is less than $38d^2$, (2) still holds in this case.

Case (b): There is no complete cell in I, but there are some cells C_i, such that the width of $C_i \cap I$ is d. We call these cells *full width cells*. In the horizontal direction, I expands at most 7 columns. In the vertical direction, I expands at most 2 rows. If I is within one row, we mark all full width cells in gray color; the area of each gray cell is larger than $1/7$ of I. If I occupies two rows, we only mark full width cells in one row that has the larger intersection with I; the area of each gray cell is larger than $1/14$ of I. If V_k has at least one vertex in each gray cell, I is covered by V_k. Since there are at most 5 gray cells, and the ratio of the area of such a full width cell to the area of I is larger than $1/14$, (2) still holds in this case.

Case (c): There is neither complete cell nor full width cell in I. I expands at most 4 cells and any vertex in V_k can cover I. That is, $Pr(A) = 1$ for $k \geq 1$, and (2) still holds. □

Lemma 4 *If the restricted Rule k is applied on a unit disk graph randomly and uniformly distributed in a confined rectangular region, then there exist constants $\alpha_2, \beta_2, \gamma_2 > 0$, such that $Pr(R > x) < \alpha_2 e^{-\beta_2 x}$ and $E[R] \leq \gamma_2$.*

Proof (sketch): The difficulty here is that the restricted Rule k requires vertices in V_k be neighbors
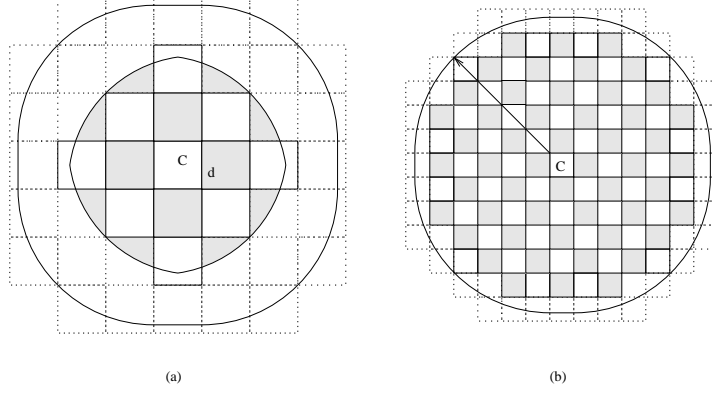
Figure 11: Coverage scheme for (a) the restricted Rule k in a unit disk graph and (b) non-restricted Rule k in a general disk graph.

of a vertex $u \in C$ for u to be pruned. Since some complete cells within $C_{max}(C)$ are not within $C_{min}(C)$, they are not suitable for covering $C_{max}(C)$. The solution is to use the partition scheme as shown in Figure 11 (a), where $C_{max}(C)$ is covered by one vertex at each of the 12 gray regions. Using the process similar to Lemmas 1 and 3, we can prove that this lemma is also true. Since the area of some gray regions is smaller than a complete cell, β_2 is smaller than β_1, but it is still a constant. $\square$

Theorem 2 can be deduced from Lemmas 3 and 4, and Theorem 3 is from the following lemma.

Lemma 5 *If the non-restricted Rule k is applied on a general disk graph randomly and uniformly distributed in a confined rectangular region, then there exist $\alpha_3, \beta_3, \gamma_3 > 0$, such that $Pr(R > x) < \alpha_3 e^{-\beta_3 x}$ and $E[R] \leq \gamma_3$.*

Proof (sketch): Similar to Lemma 1, but use a cell size $d' = r_{min}/2\sqrt{2}$ and define $C_{min}(C)$ ($C_{max}(C)$) with r_{min} (r_{max}), where r_{min} (r_{max}) is the minimal (maximal) disk radius of all vertices. Since $C_{max}(C)$ is a convex region, it can still be covered by one vertex at each of the complete cells. The resultant α_3 will be much larger than α_1, and β_3 much smaller than β_1; but they are still positive constants. $\square$

Figure 11 (b) shows a general graph, where $r_{max} = 2r_{min}$, and a scheme to cover the neighbor set of the center cell C using vertices from 52 gray cells.