CLUSTERING ALGORITHMS
FOR AD HOC WIRELESS NETWORKS

YUANZHU PETER CHEN[†], ARTHUR L. LIESTMAN[†], AND JIANGCHUAN LIU[‡]

Abstract. An ad hoc network is a multihop wireless communication network supporting mobile users without any existing infrastructure. To become commercially successful, the technology must allow networks to support many users. A complication is that addressing and routing in ad hoc networks does not scale up as easily as in the Internet. By introducing hierarchical addresses to ad hoc networks, we can effectively address this complication. Clustering provides a method to build and maintain hierarchical addresses in ad hoc networks. Here, we survey several clustering algorithms, concentrating on those that are based on graph domination. In addition, we describe results that show that building clustered hierarchies is affordable and that clustering algorithms can also be used to build virtual backbones to enhance network quality of service.

1. Introduction. In a speculative paper, Kleinrock [32] described ad hoc networking technology as a blend of nomadicity, embeddedness, and ubiquity. In a network of the future, users and computing devices will be able to connect to such a network conveniently and even transparently. Computing and communication capabilities will not only be restricted to standard electronic devices, but every gadget can afford to embed a considerable amount of intelligence. On a global basis, devices in the network will be able to rely on other devices to relay packets for them if necessary. The entire world will be heterogeneously networked by a vast "invisible global infrastructure".

The idea of ad hoc networking has been around for over 30 years. As early as 1972, DARPA started the pioneering PRNet (Packet Radio Network) project [31]. Subsequently, various projects sponsored by the military, such as SURAN (Survivable Radio Networks), TI (Tactical Internet), and GloMo (Global Mobile Information Systems), were launched to implement the ad hoc networking paradigm [20]. In the meantime, many enabling technologies, such as wireless signal processing and encoding, distributed computing, VLSI circuit design and manufacturing, cryptography, positioning services, et al. have been invented and developed that can address various problems confronting the ad hoc network community. Given the successful commercial use of the Internet, one cannot help asking why there are no cost-effective off-the-shelf commercial ad hoc networking systems. Among the many challenges for ad hoc network designers and users, scalability is a critical issue. In particular, when a flat-topology network contains a large number of nodes, control overhead, such as routing packets, requires a large percentage of the limited wireless bandwidth.

A technology can be sustainably viable only if it can find widespread use. In order to allow ad hoc networks to achieve commercial success, we must solve the scalability problem. One promising approach is to build hierarchies among the nodes, such that the network topology can be abstracted. This process is commonly referred to as *clustering* and the substructures that are collapsed in higher levels are called *clusters*.

In this chapter, we first explain why scalability is a hindrance for ad hoc networks and why the scaling techniques used successfully by the Internet are not directly applicable. We then survey some of the clustering algorithms for building network hierarchies. Finally, we consider the costs associated with using clusters in hierarchical

---

[†] School of Computing Science, Simon Fraser University, Canada.

[‡] Department of Computer Science and Engineering, The Chinese University of Hong Kong, China.

routing and how QoS in ad hoc networks can benefit from clustering.

2. Scalability. Perkins [41] observed that "aggregating routing information is the key to Internet scalability". In particular, a node's IP address contains hierarchical information related to its location that can be used in routing. Due to the mobility of nodes in an ad hoc network, this is not as simple to accomplish.

In a multihop packet-switched network, intermediate nodes are required to route packets between the source and destination if they (the source and the destination) are not directly connected. For example, in a distance-vector routing protocol, each node participating in the route calculation stores a routing table and shares it with all neighboring nodes. If the network has a flat topology (that is, all nodes are treated equally), the size of the routing table is proportional to the number of nodes in the entire network. Further, as network size increases, communication costs tend to consume a larger proportion of the bandwidth. Furthermore, as the rate of the network topology change increases, the exchange of routing tables between neighboring nodes must be more frequent to keep the routing information up to date. Other network parameters, such as network node density and traffic load, can also impair network scalability. Arpacioglu, Small, and Haas [6] have begun a study of the scalability issue of multihop networks and, in particular, ad hoc networks.

The Internet, a multihop packet-switched communication network, manages to function with approximately 10^9 nodes. Each node in the Internet is given a 32-bit IP address that is assigned in a way such that all the nodes in the same subnet share the same address prefix. This very important property allows us to build a hierarchy in the Internet topology. Routing nodes do not need to store the IP addresses of all the nodes in the network; address prefixes are sufficient to direct packets to the proper subnets for further local routing.

Unfortunately, due to mobility, nodes in an ad hoc network can not be assigned such aggregate addresses. This is an obstacle for scaling up ad hoc networks.

However, we believe that many substructures in a large-scale, even global, ad hoc network are relatively stable. Users can indeed be mobile, but their movements are usually confined within a specific geographical area. For example, students may wander around a campus during the day and commute within a metropolitan area on a daily basis. These movements cause local topology changes but do not drastically alter the overall structure of the network. Since many of these changes are confined to a relatively small region, one can abstract the network to obtain a simpler topology and avoid the need to inform the entire network of these topology changes. Local portions of the network are represented by super-vertices in the abstracted topology and connections between them are super-edges. Clustering is a process of defining such an abstracted structure of a network. It can be applied recursively to obtain a multi-level hierarchy. We will give a more formal definition of clustering in Section 3.

After clustering, each node in the hierarchy can be assigned a hierarchical address that indicates its position in each level of the hierarchy. Routing can easily be carried out using such addresses. We use an example from Sucec and Marsic [50] to explain this. Figure 2.1 depicts an n-node network with three hierarchy levels created by recursive clustering. We use the terms Level-0 to refer to the original network, Level-1 to refer to the structure obtained by clustering once, and Level-2 to refer to the structure obtained by clustering Level-1. Each node in the network can be assigned a 3-level hierarchical address. For example, in the figure node 63 is a member of the level-1 cluster represented by node 68. Node 68, in turn, is a member of the level-2 cluster represented by node 97. Thus, node 63's hierarchical address is $(97_2, 68_1, 63_0)$,

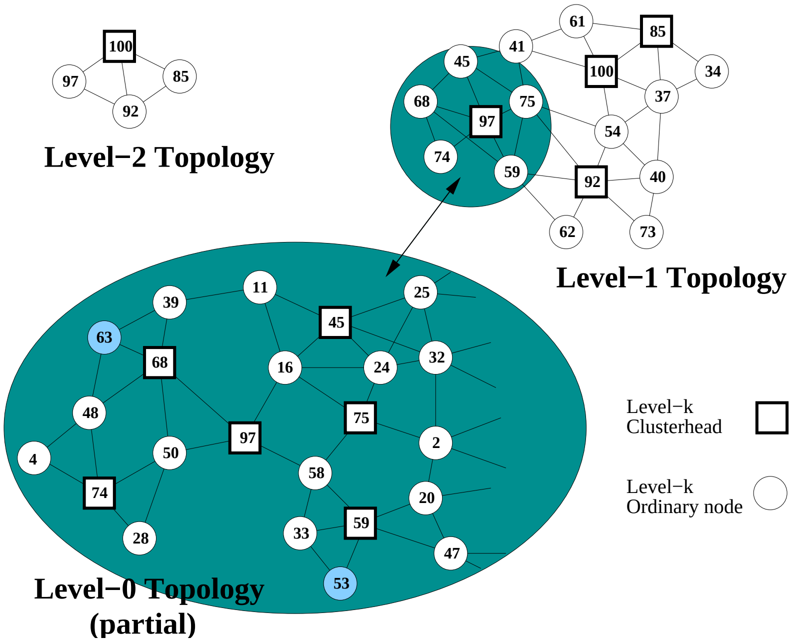

Fig. 2.1. *Hierarchical routing [50].*

where the subscripts indicate address levels. In such a hierarchy, a node only needs to store a $3 \times c$ matrix to route packets, where c is the number of sub-clusters within a cluster of the next higher level. Suppose source node 53 $(97_2, 59_1, 53_0)$ wants to send a packet to destination node 63 $(97_2, 68_1, 63_0)$. The packet is first routed to a node in node 63's level-1 cluster, say node 68 $(97_2, 68_1, 68_0)$, and then routed to node 53 within the cluster. In general, if c is of a constant order in each level, then the hierarchy level $L = O(\log n)$. Therefore, each node only needs to store a routing table of size $O(\log n)$ rather than of size $O(n)$. With this exponential savings from clustering, it is possible for an ad hoc network to scale. (See Steenstrup [45] for more details on hierarchical routing.)

3. Topology Abstraction and Clustering Algorithms. *Topology control* is the problem of determining an appropriate topology for ad hoc networks. The physical capabilities of the network devices, node locations, and peripheral settings provide a potential topology of an ad hoc network. However, other constraints may be placed on the network that make it desirable to use a simpler substructure. To produce such a substructure, we use a topology control process. One approach to this is to reduce the power levels of the nodes to obtain a subgraph of the network. Another approach, which we adopt, is to build and maintain substructures in ad hoc networks. Readers are referred to Li [35] and Rajaraman [43] for more information on the topic of topology control.

The *clustering problem* can now be defined formally. We are given an undirected graph $G = (V, E)$ representing a communication network where the vertices are the nodes in the network and the edges are the communication links. The clustering process first divides V into a collection of (not necessarily disjoint) subsets $\{V_1, V_2, \ldots, V_k\}$, where $V = \bigcup_{i=1}^{k} V_i$, such that each subset V_i induces a connected subgraph of G. Note that these induced subgraphs can overlap. Each such vertex

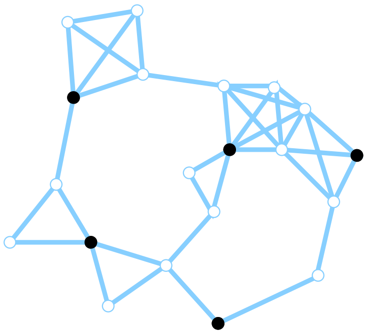

Fig. 3.1. *Dominating set.*

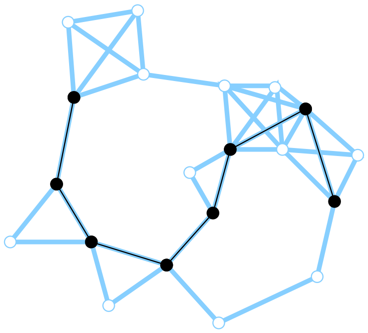

Fig. 3.2. *Connected dominating set.*

subset is a cluster. Ideally, the size of the clusters falls in a desired range and the induced subgraphs have small diameters. An abstracted graph $G' = (V', E')$ is constructed, where each vertex $v'_i \in V'$ corresponds to a subset V_i. There is an edge from v'_i to v'_j if and only if there is an edge of E from some vertex $u_i \in V_i$ to some vertex $u_j \in V_j$. Typically, a particular vertex in each cluster is elected to represent the cluster. This vertex is commonly called the *cluster-head* or *cluster-leader*. The abstracted network G' can also be clustered leading to a multi-level hierarchy.

A natural way to cluster an ad hoc network is to use the notion of graph domination or one of its variants (see Section 3.1 for formal definitions). The members of a dominating set are chosen as cluster-heads and the neighborhood of each cluster-head comprises a cluster. In this section, we describe some variants of graph domination and then review some of the clustering algorithms for ad hoc networks based on these variants.

3.1. Graph Domination. Given a graph $G = (V, E)$, the *closed neighborhood* $N[v]$ *of a vertex* v in G consists of the vertices adjacent to v plus vertex v itself. The *closed neighborhood* $N[S]$ *of the set* $S \subseteq V$ is the union $\bigcup_{v \in S} N[v]$. The *closed distance-*k *neighborhood*, $N_k[v]$, of a vertex $v \in V$ is the set of vertices that are within distance-k from v. The *open neighborhood* $N(v)$ *of a vertex* v in graph G consists of the vertices adjacent to v is $N(v) = N[v] \setminus v$ with other open neighborhood terms defined analogously. For other graph theoretic definitions, see West [51].

A *dominating set* of a graph $G = (V, E)$ is a subset $S \subseteq V$, such that every vertex $v \in V$ is either in S or adjacent to a vertex of S. The solid black vertices in Figure 3.1 form a dominating set of the graph. A vertex of S is said to *dominate* itself and all adjacent vertices. We say that an edge is *dominated* if either of its endpoints is in S and refer to other edges as *free*. In general, a vertex subset S is called a *distance-*k *dominating set* if every vertex v is within the closed distance-k neighborhood of some vertex of S.

A dominating set is an *independent dominating set* if no two vertices in the dominating set are adjacent. The dominating set of Figure 3.1 is an independent dominating set. A *connected dominating set* S of a given graph G is a dominating set whose induced subgraph, denoted $\langle S \rangle$, is connected. An example is shown in Figure 3.2. Another important variant is the weakly-connected dominating set. For any subset $S \subseteq V$, the subgraph *weakly induced* by S is the graph $\langle S \rangle_w = (N[S], E \cap (N[S] \times S))$. That is, the weakly induced subgraph $\langle S \rangle_w$ contains the vertices of S, their neighbors, and all edges of the original graph with at least one endpoint in S. Figure 3.3 shows a

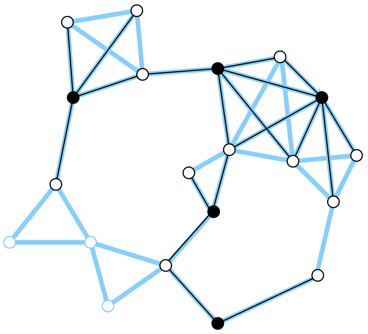

Fig. 3.3. *Weakly induced subgraph*

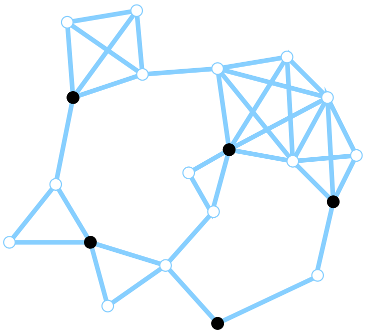

Fig. 3.4. *A weakly-connected dominating set.*

subset S of vertices in solid black with the edges of $\langle S \rangle_w$ indicated by black lines. Note that, in this example, $\langle S \rangle_w$ is not a spanning subgraph since S is not dominating. A vertex subset S is a *weakly-connected dominating set* if S is dominating and $\langle S \rangle_w$ is connected. The black vertices of Figure 3.4 form a weakly-connected dominating set S of the graph.

Intuitively, in the network clustering context, the vertices of the dominating set represent cluster-heads. Their neighborhoods (or, perhaps, distance-k neighborhoods) are the clusters. These clusters can be viewed as vertices in an abstracted network and the connections between them as edges. In general, to create a small abstracted graph, one wishes to find the smallest possible set of cluster-heads for the network. Unfortunately, the related decision problems on dominating set variants for general graphs are all NP-complete. Therefore, to use small dominating sets to form the abstracted network structure one will need to resort to approximation algorithms and heuristics. See Haynes, Hedetniemi, and Slater's comprehensive monographs for more information on graph domination [27, 26].

3.2. Clustering Algorithms. In the following survey, we focus on clustering algorithms based on graph domination. We briefly mention other notions used to define clusters.

3.2.1. Clustering with independent dominating sets. One can produce a relatively small number of clusters of a given graph by insisting that the dominating set is also an independent set.

Baker and Ephremides [7] devised one of the earliest clustering algorithms for ad hoc networks, the linked cluster algorithm. This algorithm is executed in a synchronous ad hoc network, where each node has a dedicated TDMA time slot to avoid collisions. It takes $|V|$ time slots for a node to learn the structure of its neighborhood. A vertex v is chosen as a cluster-head by a neighbor u if v has the highest vertex ID within $N(u)$. The chosen vertices form an independent dominating set of G.

Gerla and Tsai [22] proposed two clustering algorithms based on vertex ID and vertex degree. In the lowest-ID algorithm, each vertex with the lowest ID within its closed neighborhood is selected as a cluster-head. In the highest degree algorithm, each vertex with the highest degree in its closed neighborhood is selected. The cluster-heads chosen by these two algorithms form an independent set. However, as noted by Chen, et al. [11], these algorithms do not work on all graphs. In particular, for some graphs the cluster-heads do not form a dominating set and, thus, not every vertex has a cluster-head. Lin and Gerla [37] corrected this flaw and proposed a modified lowest-

ID algorithm that constructs independent dominating sets. Extending this result, Chen, et al. [11] presented an algorithm for constructing distance-k dominating sets with the additional property that the members of the dominating set are at distance at least $k+1$ from each other. This algorithm selects vertices based on the highest-degree within distance-k neighborhoods with the lowest-ID used to break ties. Basagni generalized this result to show that any meaningful measure can be used to determine cluster-heads [9].

For some other results on clustering with independent dominating sets, see the papers of An and Papavassiliou [5], Hou and Tsai [29], and Gerla, Kwon, and Pei [21].

3.2.2. Clustering with dominating sets. The use of independent dominating sets as cluster-heads is problematic when the network topology changes. In particular, when two cluster-heads move within transmission range of each other, one of them must defer to the other which can trigger cluster-head changes that may propagate throughout the network. Such an effect is called *chain reaction* [21]. By relaxing the independence condition on dominating sets, this chain reaction effect does not occur. Thus, it may be of interest to simply consider dominating sets.

Liang and Haas [36] presented a distributed greedy algorithm for dominating sets that mimics a centralized greedy algorithm. In the centralized algorithm, a dominating set is constructed by adding, in each iteration, the vertex with the largest number of free neighbors. This yields a dominating set with approximation ratio $O(\log \Delta)$, where Δ is the maximum vertex degree. The authors showed that this algorithm can be distributed so that, in each iteration, vertices only need to know about the structure of their distance-2 neighborhood. Consequently, both algorithms have the same logarithmic approximation ratio. Jia, Rajaraman and Suel [30] devised a randomized version of this algorithm that terminates in $O(\log |V| \log \Delta)$ rounds with high probability. The approximation ratio is expected to be $O(\log \Delta)$ and is $O(\log |V|)$ with high probability.

For some other results on clustering with dominating sets, see the papers of Amis, Prakash, Vuong, and Huynh [4], Belding-Royer [10], and Sivakumar, Sinha, and Bharghavan [44].

3.2.3. Clustering with connected dominating sets. Some researchers argue that better connectivity among the cluster-heads is an advantage for applications such as message broadcasting. The vertices of a connected dominating set induce a connected subgraph that can be used as a *virtual backbone* so that broadcast redundancy is reduced significantly [47].

As the minimum connected dominating set decision problem is NP-complete in general graphs, Guha and Khuller [23, 24] proposed two centralized greedy algorithms for finding suboptimal connected dominating sets in arbitrary connected graphs. In one algorithm, vertices are added to a connected set so as to maximize the number of newly dominated vertices. In the other, the connectivity of the subgraph induced by adding each candidate to the current set is also considered. Both algorithms have an approximation ratio of $O(\log \Delta)$. Due to the close similarity between the connected dominating set problem and the set cover problem, it is unlikely that an approximation ratio asymptotically better than $O(\log \Delta)$ can be found for the connected dominating set problem [19].

Das and Bharghavan [16] provided distributed implementations of the algorithms of Guha and Khuller [23, 24] for constructing connected dominating sets in ad hoc networks. These distributed algorithms generate the same connected dominating sets as their centralized counterparts and, thus, have exactly the same approximation ratio since they utilize central coordinators to oversee the entire execution.

To address the issue of non-localized computation in the distributed algorithms of Das and Bharghavan, Wu and Li [53, 54] presented a localized distributed algorithm for finding small connected dominating sets in which each node only needs to know its distance two neighborhood. The algorithm consists of two marking phases. Initially, each vertex is marked F to indicate that it is not in the connected dominating set. In phase one, a vertex marks itself T if any two of its neighbors are not directly connected. This process marks all vertices that can be potentially included in a connected dominating set. In phase two, a T vertex v changes its mark to F if either of the following conditions is met:

1. $\exists u \in N(v)$ which is marked T such that $N[v] \subseteq N[u]$ and $id(v) < id(u)$;
2. $\exists u, w \in N(v)$ which are both marked T with $N[v] \subseteq N[u] \cup N[w]$ and $id(v) < \min\{id(u), id(w)\}$.

The left and right examples shown in Figure 3.5 illustrate conditions 1 and 2, respectively. The vertices colored black are T vertices and those colored white are F vertices. In both cases, the vertex v will change its mark to F provided the identity condition holds. This algorithm constructs a connected dominating set in a localized fashion. However, there is no known non-trivial upper bound on the size of the connected dominating set generated.

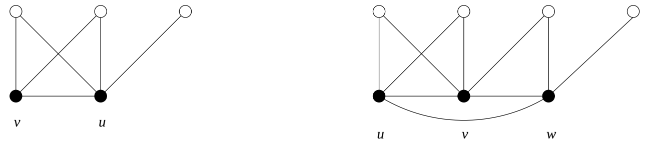

FIG. 3.5. *Unmarking conditions.*

In a more recent paper, Dubhashi, et al. [17] presented a distributed algorithm for constructing small connected dominating sets and weakly-connected dominating sets with an $O(\log \Delta)$ approximation ratio. The connected dominating set algorithm first constructs a dominating set and then adds extra vertices in an economical way such that the resultant dominating set is connected and has a provable performance ratio. Specifically, the algorithm utilizes the randomized algorithm of Jia, Rajaraman and Suel [30] to construct a dominating set that is expected to be within a $O(\log \Delta)$ factor of the minimum size. The algorithm utilizes the observation that a simple undirected graph G on n vertices has at most $n^{1+\frac{2}{g-1}} + n$ edges (Lemma 15.3.2 in Matousek [40]), where g is the girth (length of the shortest cycle) of G. Given a dominating set S of G, an auxiliary graph G' is constructed on S. By removing cycles of length less than $\lfloor 1 + 2\log|S| \rfloor$ in G', G' has no more than $2|S|$ edges left. An edge $e = (u, v)$ of G' corresponds to a set of paths $\{P_i(u, v)\}$ in G of length at most three. For each pair of such vertices u and v, additional vertices are added to S only if u and v are neither adjacent nor joined by a path comprised of only dominated vertices in G. At most two additional vertices on a path are added to S. As this is done for each edge in G', the total number of vertices added to S is at most $4|S|$. Therefore, S becomes a connected dominating set of expected size at most $O(\log \Delta)$ times the minimum.

In an obstacle-free two-dimensional space where all vertices have the same transmission range, ad hoc networks can be modeled using unit disk graphs (UDG's) [15]. UDG's are the intersection graphs of equal sized circles in the plane, that is, there is an edge between two vertices if their corresponding circles intersect.

Alzoubi, Wan, and Frieder [1, 2] proposed a localized algorithm for finding small connected dominating sets in UDG's. Initially, a maximal independent set of the given UDG G is chosen. This set is also a minimal dominating set of G. Other vertices are added to guarantee that the set is connected. The algorithm takes advantages of some particular geometric properties of UDG's that guarantees that the size of the chosen connected dominating set is within a constant factor of the minimum.

3.2.4. Clustering with weakly-connected dominating sets. Chen and Liestman [13] introduced the use of weakly-connected dominating sets for clustering ad hoc networks. This inherently sparser structure models the scatternet configuration of Bluetooth [25]. In the Bluetooth specification, devices can form two types of master-slave structure: piconet and scatternet. A piconet has a star topology with a single master device at the center and a set of slave devices around. Several piconets can be joined to form a scatternet. A weakly-connected dominating set of a graph faithfully captures the scatternet topology with the vertices in the dominating set being the master devices.

A series of algorithms was presented by Chen and Liestman [13] with an approximation ratio of $\ln \Delta + O(1)$ based on the algorithms of Guha and Khuller [23, 24] for connected dominating sets. These greedy algorithms construct weakly-connected dominating sets incrementally by adding a vertex to the current set. As in the connected dominating set case, a $\ln \Delta + O(1)$ approximation ratio upper bound can be proved for this algorithm and it is asymptotically optimal.

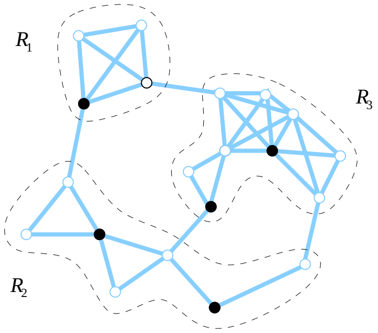

Fig. 3.6. *Zonal clustering scheme.*

In order to decentralize these algorithms, Chen and Liestman [14] proposed a *zonal* version of these algorithms. In the zonal clustering algorithm, given a *zone size control parameter* x, each zone is a connected subgraph of the input network with no more than $2x$ vertices. A zone has a dedicated vertex known by all zone members as *root*. The zonal construction algorithm has two levels: intrazonal and interzonal. In the intrazonal level, a weakly-connected dominating set is independently constructed for each zone. In the interzonal level, the root of a zone adds additional vertices to its weakly-connected dominating set to guarantee that the union of the dominating sets for the individual zones is a weakly-connected dominating set for the entire network. As an illustration, the network in Figure 3.6 is partitioned into three zones. The solid black vertices are the dominators of each zone. The hollow black vertex in zone Z_1 is added to guarantee the weak connectivity of the dominating set. The advantage of the zonal approach is that the zone size control parameter x can be used to control the zone granularity, providing a trade-off between the extent of network structure

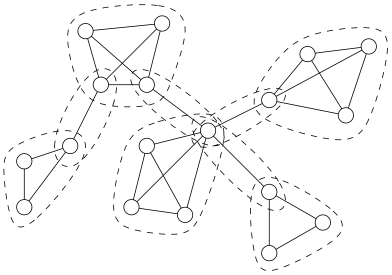

FIG. 3.7. *Clique-based clustering.*

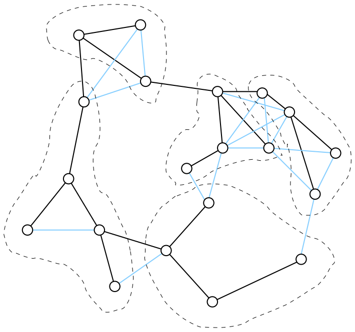

FIG. 3.8. *Spanning tree-based clustering.*

simplification and the locality of algorithm execution.

An algorithm of Dubhashi, et al. [17] similar to the one presented in Section 3.2.3 constructs small weakly-connected dominating sets with an $O(\log \Delta)$ approximation ratio.

Alzoubi, Wan and Frieder [3] proposed an algorithm related to their algorithm for connected dominating sets in unit disk graphs (UDG's) [2] that generates weakly-connected dominating sets of size within a constant factor of the minimum for UDG's (see Section 3.2.3).

3.2.5. Clustering by methods other than graph domination. Other algorithms for clustering ad hoc networks have been proposed that are not based on graph domination.

Krishna, Vaidya, Chatterjee, and Pradham [33] presented a clustering algorithm where clusters are formed without cluster-heads. A *clique* in graph $G = (V, E)$ is a subset S of V, whose induced subgraph is complete. The authors use maximal cliques as clusters as illustrated in the example of Figure 3.7. A node is called a *boundary node* if it belongs to more than one cluster. Nodes in the same cluster can communicate directly with each other, while nodes in different clusters must rely on boundary nodes to relay messages. This is similar to the concept of Internet BGP routing.

Banerjee and Khuller [8] proposed a protocol based on a *spanning tree*. In their scheme, a cluster is a subset of vertices whose induced graph is connected. These subsets are chosen with consideration to cluster size and the maximum number of clusters to which a node can belong. The idea is to group branches of a spanning tree into clusters of an approximate target size. The resulting clusters can overlap and nodes in the same cluster may not be directly connected. Figure 3.8 shows an example of clusters obtained by this method. In this figure, the spanning tree is shown in black.

4. Utilizing the Clustered Structure. In this section, we present two related case studies. The first considers the costs due to cluster-based hierarchical routing. The other studies how to use clustering to build virtual backbones for service discovery in ad hoc networks. We will see that clustering is not only affordable but also beneficial in various aspects.

4.1. Hierarchical routing overhead. As we have seen in the example of Section 2, multi-level hierarchical routing can introduce exponential savings in the amount of routing information to be stored and exchanged in a large-scale ad hoc network. Sucec and Marsic [48, 50] analytically assessed the communication overhead incurred in hierarchal routing and showed that the *number of control packets transmissions per second per node*, denoted ϕ, is only polylogarithmic in the number of nodes of the network.

4.1.1. Network settings. Sucec and Marsic [48, 50] studied an ad hoc network represented by a connected unit disk graph $G = (V, E)$. All nodes were placed randomly in a bounded two-dimensional space according to a uniform distribution. Each node has a fixed transmission range. Scalability can be considered with respect to various meaningful parameters [6], such as number of nodes, node density, and node velocity. The authors considered only the number N of nodes in the network while holding density and velocity constant.

4.1.2. Hierarchy construction and addressing. The authors use the max-min d-hop clustering algorithm of Amis, Prakash, Vuong, and Huynh [4] recursively to build a hierarchy of clusters. By setting d to one, the localized algorithm constructs and maintains a dominating set of the graph representing the topology of one level lower. A node is selected as a cluster-head by a neighbor v if it has the highest ID in $N[v]$. The resulting clusters do not overlap, that is, every vertex has exactly one dominator (a neighbor or itself). The clustering process is then applied to the clustered network to build another level of the hierarchy. Assuming constant network density, the average vertex degree is bounded by a constant c. Thus, the clustering ratio, the average number of sub-clusters in a cluster, is bounded by c and the number of levels in the hierarchy is $L = \log_c N = O(\log N)$.

Each node v in the network maintains a *hierarchical topology map*. Such a data structure is an L-row table. The i^{th} row $(i = 1, 2, \ldots, L)$ lists the hierarchical addresses of the $(i-1)^{\text{st}}$ level clusters within the ith level cluster that v belongs to. A source node needs to translate the flat node address of the destination node to an appropriate hierarchical address. The authors apply the hierarchical location management protocol [49]. By using a set of hashing functions from the flat address space to the hierarchical address space, a vertex registers its L-level hierarchical address at L different *address servers* in the network. A source vertex can look up the hierarchical address of an intended destination by sending queries to the address servers calculated using the same set of hashing functions.

4.1.3. Communication overhead. In the hierarchy constructed above, node mobility can cause topology changes (link/cluster additions/deletions) to propagate up to any level. Despite this complication, the authors derived the following breakdowns of the control overhead. The the values are expressed as packet transmissions per node per second unless otherwise specified.

- $\phi_{\text{HELLO}} = \Theta(1)$ – “Hello” protocol.
- $\phi_{\text{CL}} = O(\log N)$ – cluster formation and maintenance messaging.
- $\phi_{\text{ACQ}} = O(\log N)$ – acquisition of local data when node migrates from one cluster to another.
- $\phi_{\text{FLOOD}} = O(\log N)$ – flooding of cluster topology updates to cluster members.
- $\phi_{\text{REG}} = \Theta(\log N)$ – location registration events.
- $\phi_{\text{HANDOFF}} = \Theta(\log^2 N)$ – handoff or transfer of location management data.
- $\phi_{\text{QRY}} = \Theta(h)$ – location query events.

- $\phi_{\text{CTRL-HEADER}} = \Theta(\log N)$ bits per control message datagram – addressing information required in datagram headers.

In summary, the total communication overhead in such a hierarchical routing protocol is $\Theta(\log^2 N)$ packet transmissions per node per second and $\Theta(\log^3 N)$ bits per node per second, assuming that the node density, mobility, and traffic load remain constant. This is a substantial (exponential) improvement over the linear costs of routing in flat networks.

4.2. Virtual backbone based service discovery. In this section, we discuss an application of clustering algorithms in service discovery in ad hoc networks. With the rapid increase of available services and accessing requests for ad hoc networks, an efficient service discovery system that enables clients to search and utilize desired services is necessary. There have been extensive studies of service discovery in the Internet [18, 56]. They often suggest a tree organization of service directory servers which achieves scalability by partitioning the network into domains. The maintenance of the tree structure, however, is not an easy task in ad hoc networks and statically configured domains do not reflect the dynamic relations among mobile nodes. In addition, providing desirable Quality-of-Service (QoS) is an important design objective for service discovery. While there are many proposals for QoS enhancements in underlying layers [12, 34, 55, 57], a more general solution is to deploy multiple replicated providers for the same service. In this case, locating the best provider according to QoS metrics such as path latency becomes an important issue. In Internet-based discovery systems, probing is often employed. Probing measures QoS in a relatively short period and is thus suitable for quasi-static link conditions. However, simulation results have suggested that it is not very effective for highly dynamic wireless links in mobile ad hoc networks [38].

To overcome these problems, a cluster-based QoS-aware service discovery algorithm was proposed by Liu, et al. [38, 39]. The system relies on a virtual backbone (VB) consisting of a small set of nodes, among which all of the control messages for service discovery are exchanged. The VB nodes are dynamically selected. They maintain the directory information of the services using a hash indexing which provides fast query response and fault-tolerance. These nodes dynamically partition the whole network into *virtual domains*. Each domain has a home VB node which responds to discovery queries from clients in the domain. Finally, all service registration and query messages are exchanged only among VB nodes. These frequently exchanged messages are also used to continuously estimate path QoS, such as the path latency between two VB nodes, which is then used to direct the selection of service providers.

A virtual domain is essentially a cluster with the home VB being the clusterhead. The clustering algorithms described in the previous sections can thus be used to generate the VB. Liu, et al. [38] proposed a simple distance-based broadcast algorithm for VB formation as its overhead is relatively low. A similar approach using an advanced clustering algorithm was proposed by Helmy [28].

We now give a brief overview of the operations in this cluster-based system as well as its performance in QoS-aware service discovery.

4.2.1. Organization of Service Directory. In this system, each service has a unique ID and its provider registers the ID to the system. The registration process is as follows: the provider first sends a registration request to its home VB node; the request includes the service ID, valid time, provider's address, and other related information. Assume the service ID is P. A uniform hash function F is used to produce a index $Q = F(P)$ in the set of $\{1, 2, \ldots, M\}$. The home VB node will then distribute the registration request to VB node Q to store the directory information of the service.

In this organizing scheme, the directory information of a service can be quickly located through hash indexing. Moreover, if there are multiple providers for the same service, all of the providers' addresses can be located in only one VB node. It is thus easy to obtain the list of the providers for QoS-aware selection. To provide fault-tolerance, one can either use multiple hash functions and replicate the directory information in more than one VB node, or use a distributed hash table (DHT) [46].

4.2.2. Monitoring of Path QoS. For replicated services, the path QoS between the client and the providers needs to be estimated to select the best provider. Note that short-term probing might be inaccurate in such a dynamic network. It may also trigger high cost route discovery operations of the on-demand routing protocol [42] as non-VB nodes communicate with each other less frequently. Hence, instead of using probing, a cluster-based approximation method is proposed by Liu, et al. [38], assuming that the nodes in a virtual domain have similar QoS. Here, the home VB node serves as a representative, and the path QoS of two non-VB nodes is approximated by the path QoS of their respective home VB nodes.

Assume that path latency is used as the path QoS measure and that all the VB nodes are equipped with the Global Positioning System (GPS). Their local clocks can be synchronized by the Coordinated Universal Time service provided by GPS. Thus, by adding a time-stamp to all the packets exchanged through the VB, the VB nodes can easily estimate expected path latencies among themselves by a statistical predictor based on those frequently exchanged packets [52].

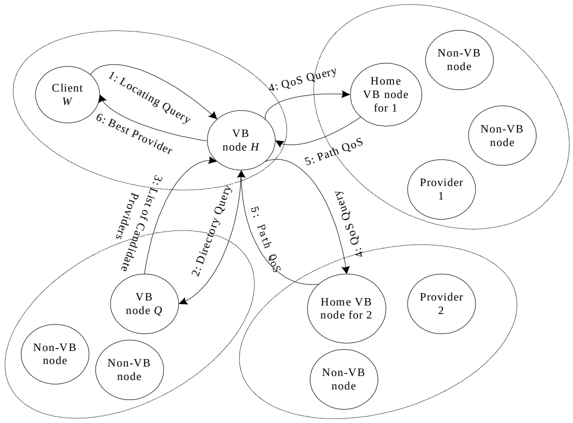

FIG. 4.1. *State transition for service query.*

4.2.3. Service Query and Selection. Given the directory organization, a client node that needs to locate a service can directly query its home VB node and the VB node will then search the service directory information and return the address of the service provider based on QoS measures. The detailed process is shown in Figure 4.1. First, the client node W sends a query including the service ID (P) to its home VB node H (Step 1). If no local directory record matches the query, the

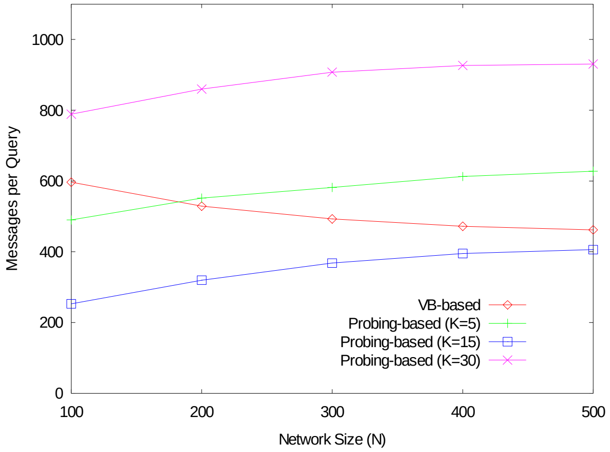

FIG. 4.2. *Discovery cost as a function of network size for different discovery systems.*

home VB node will calculate the hashing index (Q) of the service and then forward query to VB node Q (Step 2). A full list of candidate providers' addresses is then returned to the H (Step 3). If node Q fails, H will try to forward the query to other qualified VB nodes that have the directory replication of this service until the query is successful or there is no other qualified VB node.

Upon receiving the service providers' addresses, H should query the home VB nodes for these candidate providers to estimate the path QoS. For simplicity, H will send the QoS query to all the VB nodes simultaneously (Step 4). The query includes the list of the candidate providers and the type of QoS of interest. If there are one or more candidates in a VB node's domain, it will respond to H by providing the addresses of the candidates and the corresponding QoS information (Step 5). The QoS of all candidate providers are then compared at H, and the directory information of the best candidate is returned to client W(Step 6). Then it can access the service by launching appropriate protocols.

4.2.4. Sample Results. To have some quantitative understanding, in Figure 4.2 we show the discovery costs (messages per query) for the cluster(VB)-based system in a typical ad hoc network. For the sake of comparison, the results for a traditional discovery system are also presented. In this system, service discovery is based on a centralized directory server and, when there are replicated providers, a client sends K consecutive probing packets to each provider to estimate path latency and selects the one with the minimum average latency. It can be seen that, with the increase of network size, the cost of the cluster-based system decreases. This is because the queries involve only message exchanges among the VB nodes and the QoS (path latency) information is shared by the nodes in the same virtual domain; for denser node distributions, more nodes could be accommodated in one cluster and the average cost per query is thus reduced. On the contrary, the cost of the traditional system even increases in this case as path probing is executed for each query. Figure 4.3 shows the QoS gain for the two systems in terms of the latency reduced (compared to a random replica selection). It can be seen that the QoS gain of the cluster-based system does not decrease significantly with the increase of network size. In contrast, the gain of the traditional system is noticeable only if there are enough probing packets.

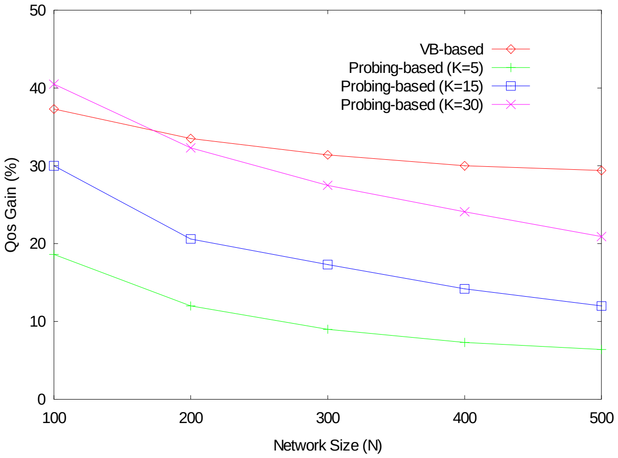

Fig. 4.3. *QoS gain as a function of network size for different discovery systems. Baseline: random provider selection.*

However, in this case, the cost is much higher than that of the cluster-based system. More importantly, for the probing-based selection, the QoS gain diminishes quickly with the increase of network size.

Consequently, the cluster-based service discovery system scales well and is particularly suitable for QoS-aware service discovery in large-scale ad hoc networks.

5. Conclusion. In order for ad hoc networks to achieve widespread use, the ad hoc networking community has been working on improving its scalability. By building and maintaining hierarchies among network devices, large-scale networks can scale up affordably. Various clustering algorithms have been devised as building blocks for this purpose. Both deterministic and probabilistic distributed algorithms have been developed to construct clusters.

It has be shown that the average communication costs incurred in building, maintaining, and utilizing such hierarchies can be logarithmic in the network size. Furthermore, many other aspects of ad hoc networks, such as message broadcasting and quality-of-service, can also benefit from network clustering.

REFERENCES

[1] K. M. Alzoubi, P.-J. Wan, and O. Frieder, *Distributed heuristics for connected dominating set in wireless ad hoc networks*, KICS Journal of Communications and Networks, 4 (2002).

[2] ———, *Message-optimal connected dominating sets in mobile ad hoc networks*, in The Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02), June 2002, pp. 157–164.

[3] ———, *Weakly-connected dominating sets and sparse spanners in wireless ad hoc networks*, in The 23rd International Conference on Distributed Computing Systems (IEEE, ICDCS), May 2003.

[4] A. Amis, R. Prakash, T. Vuong, and D. Huynh, *Max-min d-cluster formation in wireless ad hoc networks*, in Proceedings of IEEE INFOCOM, Tel Aviv, March 2000.

[5] B. An and S. Papavassiliou, *A mobility-based clustering approach to support mobility management and multicast routing in mobile ad-hoc wireless networks*, International Journal of Network Management, 11 (2001), pp. 387–395.

[6] O. Arpacioglu, T. Small, and Z. Haas, *Notes on scalability of wireless ad hoc net-*

works, IETF Internet draft, draft-irtf-ans-scalability-notes-01.txt, December 2003, (Work in Progress).

[7] D. J. Baker and A. Ephremides, *The architectural organization of a mobile radio network via a distributed algorithm*, IEEE Transactions on Communications, COM-29 (1981), pp. 1694–1701.

[8] S. Banerjee and S. Khuller, *A clustering scheme for hierarchical routing in wireless networks*, Tech. Report CS-TR-4103, University of Maryland, College Park, February 2000.

[9] S. Basagni, *Distributed clustering for ad hoc networks*, in Proc. ISPAN'99 Int. Symp. on Parallel Architectures, Algorithms, and Networks, 1999, pp. 310–315.

[10] E. M. Belding-Royer, *Multi-level hierarchies for scalable ad hoc routing*, Wireless Networks, 9 (2003), pp. 461–478.

[11] G. Chen, F. G. Nocetti, J. S. Gonzalez, and I. Stojmenovic, *Connectivity-based k-hop clustering in wireless networks*, in Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS-35), January 2002.

[12] S. Chen and K. Nahrstedt, *Distributed quality-of-service routing in ad-hoc networks*, IEEE Journal on Special Areas in Communications, 17 (1999).

[13] Y. P. Chen and A. L. Liestman, *Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks*, in The Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02), June 2002, pp. 165–172.

[14] ———, *A zonal algorithm for clustering ad hoc networks*, International Journal of Foundations of Computer Science, 14 (2003), pp. 305–322.

[15] B. N. Clark, C. J. Colbourn, and D. S. Johnson, *Unit disk graphs*, Discrete Mathematics, 85 (1990), pp. 165–177.

[16] B. Das and V. Bharghavan, *Routing in ad-hoc networks using minimum connected dominating sets*, in IEEE International Conference on Communications (ICC'97), Vol.1, June 1997, pp. 376–380.

[17] D. Dubhashi, A. Mei, A. Panconesi, J. Radhakrishnan, and A. Srinivasan, *Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons*, in Proc. ACM-SIAM Symposium on Discrete Algorithms (SODA), 2003, pp. 717–724.

[18] C. P. E. Guttman, J. Veizades, and M. Day, *Service location protocol, version 2*, RFC 2608, June 1999.

[19] U. Feige, *A threshold of ln n for approximating set cover*, in 28th ACM Symposium on Theory of Computing, 1996, pp. 314–318.

[20] J. A. Freebersyser and B. Leiner, *A DoD perspective on mobile ad hoc networks*, in Ad Hoc Networking, C. Perkins, ed., Addison-Wesley, 2001.

[21] M. Gerla, T. J. Kwon, and G. Pei, *On demand routing in large ad hoc wireless networks with passive clustering*, in Proceedings of IEEE WCNC, September 2000.

[22] M. Gerla and J. T.-C. Tsai, *Multicluster, mobile, multimedia radio network*, Wireless Networks, 1 (1995), pp. 255–265.

[23] S. Guha and S. Khuller, *Approximation algorithms for connected dominating sets*, Tech. Report 3660, University of Maryland, College Park, June 1996.

[24] ———, *Approximation algorithms for connected dominating sets*, Algorithmica, 20 (1998), pp. 374–387.

[25] J. Haartsen, *Bluetooth – the universal radio interface for ad hoc, wireless connectivity*, Ericsson Reviews, (1998), pp. 110–117.

[26] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, *Domination in graphs, Advanced Topics*, Marcel Dekker, Inc., 1998.

[27] ———, *Fundamentals of Domination in Graphs*, Marcel Dekker, Inc., 1998.

[28] A. Helmy, *Parse: Power-efficient architecture for resource discovery and small-transfers in large scale ad hoc networks*, tech. report, Technical Report, USC-CS-TR-02-780, December 2002.

[29] T.-C. Hou and T.-J. Tsai, *An access-based clustering protocol for multihop wireless ad hoc networks*, IEEE Journal on Selected Areas in Communications, 19 (2001), pp. 1201–1210.

[30] L. Jia, R. Rajaraman, and T. Suel, *An efficient distributed algorithm for constructing small dominating sets*, in Proceedings of the 20th ACM Symposium on Principles of Distributed Computing (PODC'01), 2001.

[31] R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kunzelman, *Advances in packet radio technology*, Proceedings of the IEEE, 66 (1978), pp. 1468–1496.

[32] L. Kleinrock, *An internet vision: the invisible global infrastructure*, Ad Hoc Networks, 1 (2003), pp. 3–11.

[33] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan, *A cluster-based approach for routing in dynamic networks*, Computer Communication Review, 49 (1997), pp. 49–64.

[34] B. Li, *QoS-aware adaptive services in mobile ad-hoc networks*, in Proceedings of IWQoS'01, June 2001.

[35] X.-Y. Li, *Topology control in wireless ad hoc networks*, in Ad Hoc Networking, S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, eds., IEEE Press, 2003.

[36] B. Liang and Z. J. Haas, *Virtual backbone generation and maintenance in ad hoc network mobility management*, in INFOCOM, 2000, pp. 1293–1302.

[37] C. R. Lin and M. Gerla, *Adaptive clustering for mobile wireless networks*, IEEE Journal on Selected Areas in Communications, 15 (1997), pp. 1265–1275.

[38] J. Liu, Q. Zhang, B. Li, W. Zhu, and J. Zhang, *A unified framework for resource discovery and QoS-aware provider selection in ad hoc networks*, ACM Mobile Computing and Communications Review, 6 (2002), pp. 13–21.

[39] J. Liu, Q. Zhang, W. Zhu, and B. Li, *Service location for large-scale mobile ad hoc networks*, Kluwer International Journal of Wireless Information Networks, 10 (2003), pp. 33–40.

[40] J. Matousek, *Lectures on discrete geometry. Graduate Texts in Mathematics*, Springer, New York, 2002.

[41] C. Perkins, ed., *Ad Hoc Networking*, Addison-Wesley, 2001.

[42] C. E. Perkins and E. M. Royer, *Ad hoc on-demand distance vector routing*, in Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, February 1999.

[43] R. Rajaraman, *Topology control and routing in ad hoc networks: A survey*, SIGACT News, 33 (2002), pp. 60–73.

[44] R. Sivakumar, P. Sinha, and V. Bharghavan, *CEDAR: a core-extraction distributed ad hoc routing algorithm*, IEEE Journal on Selected Areas in Communications, 17 (1999), pp. 1454–1465.

[45] M. Steenstrup, *Cluster-based networks*, in Ad Hoc Networking, C. Perkins, ed., Addison-Wesley, 2001.

[46] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, *Chord: A scalable peer-to-peer lookup service for internet applications*, in Proceedings of ACM SIGCOMM, September 2001.

[47] I. Stojmenovic, M. Seddigh, and J. Zunic, *Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks*, IEEE Transactions on Parallel and Distributed Systems, 13 (2002), pp. 14–25.

[48] J. Sucec and I. Marsic, *Clustering overhead for hierarchical routing in mobile ad hoc networks*, in Proceedings of INFOCOM, New York, NY, June 2002, pp. 1698–1706.

[49] ———, *Location management for hierarchically organized mobile ad hoc networks*, in Proceedings of IEEE WCNC, Orlando, FL, March 2002, pp. 603–607.

[50] ———, *Hierarchical routing overhead in mobile ad hoc networks*, IEEE Transactions on Mobile Computing, 3 (2004), pp. 46–56.

[51] D. West, *Introduction to Graph Theory – Second edition*, Prentice Hall, Upper Saddle River, N.J., 2001.

[52] R. Wolski, *Forecasting network performance to support dynamic scheduling*, in Proceedings of the 6th IEEE International Symposium on High Performance Distributed Computing, 1997.

[53] J. Wu and H. Li, *On calculating connected dominating set for efficient routing in ad hoc wireless networks*, in DIAL M'99, Seattle, 1999, pp. 7–14.

[54] ———, *A dominating-set-based routing scheme in ad hoc wireless networks*, Special Issue on Wireless Networks, Telecommunication Systems Journal, 3 (2001), pp. 63–84.

[55] H. Xiao, W. Seah, A. Lo, and K. Chua, *A flexible quality of service model for mobile ad-hoc networks*, in Proceedings of IEEE VTC2000-spring, May 2000.

[56] D. Xu, K. Nahrstedt, and D. Wichadakul, *QoS-aware discovery of wide-area distributed services*, in Proceedings of IEEE International Symposium on Cluster Computing and the Grid, May 2001.

[57] C. Zhu and M. Corson, *QoS routing for mobile ad hoc networks*, tech. report, CSHCN Technical Report, 2001.