Quorum-Based Match-Making Services for Wireless Mesh Networks

Ilknur Aydin, Chaiporn Jaikaeo, and Chien-Chung Shen

***Abstract*— We propose a quorum-based approach to facilitating match-making (content-based routing) service for wireless mesh networks. Given a wireless mesh network, mesh nodes first autonomously select a set of backbone nodes via a backbone selection protocol. These backbone nodes then participate in a symmetric coterie construction process to create a quorum system. A quorum system has the property that the interaction of every two quorums is not empty and the union of all quorums results in the set of all backbone nodes. By using these two properties, the approach facilitates a match-making service that matches producer nodes' advertisements and consumer nodes' subscriptions at intersecting nodes between quorums.**

***Index Terms*— Wireless Mesh Networks, Service Discovery, Match-Making, Quorum System**

I. Introduction

Multi-hop wireless mesh networks are an emerging disruptive networking architecture. In contrast to current broadband Internet access paradigm which uses either cable or DSL, mesh networks facilitate free flow of information among nodes who contribute network resources and cooperates without any centralized control. For instance, neighbors could form a community-based multi-hop wireless network where they cooperate and forward each other's packets to share Internet gateways, disseminate local information, and backup each other's information. In contrast to mobile ad hoc networks, nodes of multi-hop wireless mesh networks are stationary and not constrained by energy.

Active research has been pursued for mesh networks to address issues of capacity, fairness, MAC and routing protocols, among others. In this paper, we address *services* needed to facilitate high-level applications. In particular, we propose a quorum-based approach to facilitating a *match-making* service for large-scale wireless mesh networks.

Match-making (also termed content-based routing) is different from traditional multicast and any-cast routing schemes in the sense that routing is performed based on the data (contents) in the messages, rather than specialized address attached to the messages. Sources (producers) of data in the network generate messages and advertise them without any particular destination in mind. Destinations (consumers) are determined based on their interests (via subscriptions) in receiving the produced data. Advertisements from producers and subscriptions from consumers are effectively and efficiently *matched* by the underlying network routing system [1]. Match-making capability may be used to facilitate *service/resource discovery* where the potential communicating parties do not know the identities (addresses) of each other in the first place. Once identified, conventional (mesh) routing protocols will then be used by parties to communicate directly. Without efficient/effective match-making capability, service discovery would resort to either (centralized or distributed) directory systems or query/event flooding will be needed. However, the directory systems suffer from performance bottleneck, single point of failure, or the need of manual configuration, and flooding suffers from excessive messaging overhead.

This work is supported in part by National Science Foundation under grant CNS-0347460.

I. A. is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, U.S.A. (e-mail: aydin@cis.udel.edu).

C. J. is with the Department of Computer Engineering, Kasetsart University, Bangkok, Thailand (e-mail: fengchj@ku.ac.th)

C-C. S. is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, U.S.A. (e-mail: cshen@cis.udel.edu).

Match-Making Service
Quorum System Layer
Backbone Layer
Mesh Network Layer

Fig. 1. The architecture

The proposed solution adopts a layered architecture as depicted in Figure 1. Mesh nodes in the Mesh Network Layer *autonomously* execute a backbone selection protocol to elect a set of backbone nodes to form the Backbone Layer. Every mesh node not belonging to the set of backbone nodes will associate itself with one backbone node. Backbone nodes then participate in a quorum system construction process to build the Quorum System Layer. The quorum system has the property that the interaction of every two quorums is not empty and the union of all quorums results in the set of all backbone nodes. By using these two properties, the approach facilitates a match-making service that matches producer nodes' advertisements and consumer nodes' subscriptions at intersecting nodes of two quorums. With this approach, due to the much smaller quorum size in comparison with the size of the mesh network, the messaging overhead of match-making will be drastically reduced in contrast to query flooding. Furthermore, backbone nodes and quorums could be formed autonomously to avoid any manual configuration required in existing service/resource discovery protocols.

Organization of this paper is as follows. In the next section, we describe the backbone selection protocol and its properties. In Section III, we describe the definition of a symmetric coterie quorum system, and describe the adopted quorum construction and maintenance algorithm, which provides a

TABLE I
PERCENTAGE OF BACKBONE NODES WITH DIFFERENT HOP LIMITS BETWEEN A REGULAR MESH NODE AND ITS BACKBONE NODE

hop limit	% of mesh nodes as backbone nodes
1	23.465
2	11.373
3	8.694
4	7.174
5	6.674

robust quorum layer for the match-making service. In Section IV, we describe the match-making protocol together with an illustrative example. Section V describes related work, and Section VI concludes the paper.

II. BACKBONE SELECTION

In the Backbone Layer, the selection of backbone nodes allows a smaller subset of mesh nodes to participate in the quorum construction and help reducing the messaging overhead of the match-making process. Our backbone selection algorithm, termed Adaptive Dynamic Backbone (ADB), was originally designed to support multicast operations in highly mobile ad hoc networks [2]. It has been derived from the Virtual Dynamic Backbone Protocol (VDBP) [3]. VDBP is based on the concept of *connected dominating set*, where a node is either a backbone node or an immediate neighbor of a backbone node. In contrast to VDBP, ADB relaxes the property of dominating set and allows a mesh node to be associated with a backbone node that could be more than one hop away. Table I lists the percentage of mesh nodes (obtained from simulation) selected as backbone nodes with different hop limits between a mesh node and a backbone node. Higher hop limit (for instance, 2 and higher) will select less than 12% of the mesh nodes to be the backbone nodes to support scalability.

ADB consists of three major components: (1) neighbor discovery process, (2) backbone selection process, and (3) backbone connection process, which are executed simultaneously. The neighbor discovery process is responsible for keeping track of immediate neighboring nodes via HELLO packets, as well as measuring the stability of wireless communication. The backbone selection process then uses this information to determine a set of nodes who will become backbone nodes. Since backbone nodes may not be reachable by each other in one hop, the backbone connection process will determine a subset of intermediate nodes to connect these backbone nodes together. The following data structures are maintained by each mesh node i.

- parent_i: keeps the ID of the upstream node toward the backbone from i. If i is a backbone node itself, parent_i is set to i. Initially, every node starts as a backbone node.
- NIT_i (Neighboring Information Table at node i): maintains information of all the i's immediate neighbors. Fields of each table entry are described in Table II. The entry corresponding to the neighbor j is denoted by $\mathit{NIT}_i(j)$. This table is maintained by the neighbor discovery process.
- $\mathit{stability}_i$: keeps track of stability of wireless connections node i is experiencing. Specifically, it estimates the probability that a link from i to any of its neighbor will still be usable within the next specific time window. The computation of this value is left flexible and can be obtained from various measurements such as signal strengths, signal-to-noise ratio, link failure frequency, as well as combinations of those.

A. Neighbor Discovery Process

Every mesh node is required to broadcast a HELLO packet every certain time period to discover each other in the surrounding area, as well as to allow nodes to know when some of their neighbors have already withdrawn from (or left) the mesh network. In addition, HELLO packets also carry extra information to be used by other components of ADB, especially the backbone selection process. A HELLO packet h_i sent out by the node i contains the following fields:

- $h_i.\mathit{nodeId}$: the unique ID of the sending node, i.e., i.
- $h_i.\mathit{bnodeId}$: the ID of the backbone node with which i is currently associated, or its own ID (i.e., i) if i is a backbone node.
- $h_i.\mathit{hops}$: the number of hops away from the backbone node. This value is zero if i is a backbone node. Otherwise, it is set to $\mathit{NIT}_i(\mathit{parent}_i).\mathit{hops} + 1$.
- $h_i.\mathit{degree}$: the degree of connectivity (the number of neighbors), set to $|\mathit{NIT}_i|$.
- $h_i.\mathit{nodeStability}$: the estimated stability of wireless connectivity in the area surrounding i. This value is set to $\mathit{stability}_i$.
- $h_i.\mathit{pathStability}$: the estimated stability of the path from i to its current backbone node in terms of the probability that this path will still exist within the next time window.

$$h_i.\mathit{pathStability} = \begin{cases} 1 & \text{if } i \text{ is a backbone node;} \\ \mathit{NIT}_i(\mathit{parent}_i).\mathit{pathStability} & \\ \quad \times \mathit{nodeStability}_i & \text{otherwise.} \end{cases}$$

Upon receiving a HELLO packet, the receiving node updates the corresponding entry in its own *NIT*. Due to the fact that nodes may withdraw, each entry is also associated with a timestamp recording the time at which it was last updated. When an entry has not been update for longer than a specific time, it is removed from the table.

B. Backbone Selection Process

Once mesh nodes have learned about their neighboring nodes, the backbone selection process decides whether a node should still be serving as a backbone node, or become a child of an existing backbone node by checking for its local optimality. First, each node computes its own *height* from its current status, as well as the heights of all of its neighbors. Similar to VDBP, we used $(\mathit{nodeStability}_i, degree_i, i)$ as the height metric for node i in ADB. If a node has the highest height among its neighbors, it is considered a local optimal node and should continue serving as a backbone node. If not, it picks the local optimal node as a backbone node and updates

TABLE II
FIELDS USED BY THE NEIGHBOR INFORMATION TABLE (*NIT*)

Field	**Description**
id	Neighbor's node ID
bnodeId	ID of the backbone node with which this neighbor is associated
hops	Number of hops from this neighbor to its current backbone node
degree	Degree of connectivity
nodeStability	Estimated dynamics of this neighbor's surrounding area
pathStability	Estimated stability of the path from this neighbor to its backbone node

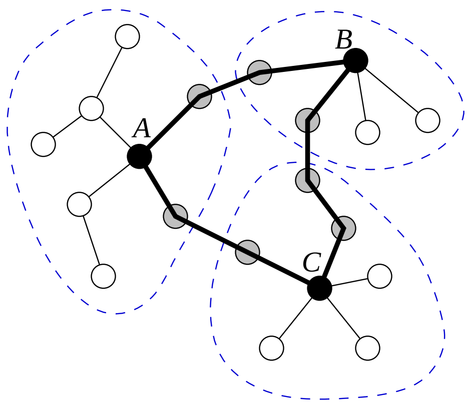

Fig. 2. Sample mesh network and the selected backbone nodes A, B, and C. Solid lines denote parent-child relationship; dashed circles denote backbone node coverage boundaries.

its *parent* variable. All consequent HELLO packets will be changed accordingly.

While keeping the number of backbone nodes as low as possible, the backbone selection process must ensure that the current backbone configuration satisfies two constraints: the hop count limit and the path stability. These two constraints require that the hop count and the path stability from every node to its backbone node must not exceed and drop below the parameters *HOP_THRESHOLD* and *STABILITY_THRESHOLD*, respectively. Therefore, once the local optimality check has been made, each node must keep listening to the incoming HELLO packets. In the case of topology change, if a node detects that its parent has withdrawn (from the neighbor discovery process), it will do the same thing as if its parent violates the constraints.

C. Backbone Connection Process

Mesh nodes which are selected to become backbone nodes by the backbone selection process cannot form a connected backbone by themselves, since they may not be reachable by one another within a single hop. The backbone connection process is responsible for connecting these backbone nodes together by designating some nodes to take the role of intermediate nodes. Consequently, the backbone nodes and intermediate nodes jointly comprise a virtual backbone, as illustrated in Figure 2. To do so, each backbone node relies on nodes that are located along the border of its coverage, called *border nodes*, to collect information about surrounding nearby backbone nodes. A node is said to be a border node if and only if it is able to hear HELLO packets from other nodes that are associated with different backbone nodes. Each border node then builds a list of backbone nodes reachable by itself, including the next hop and the number of hops to each of them, and reports this information to its own backbone node so that a backbone connection request can be performed. To reduce overhead, a border node only needs to collect and report information regarding nearby backbone nodes with IDs lower than that of its own backbone node. Hence, a backbone connection request between a pair of backbone nodes is always initiated by the backbone node with the higher ID.

III. QUORUM SYSTEM CONSTRUCTION AND MAINTENANCE

The Quorum Layer is constructed by organizing backbone nodes into sets called *quorums* where every two quorums intersect and no quorum includes another quorum. *Coterie* is a type of quorum system that requires additional properties. In particular, a *Symmetric Coterie* $\{Q_1, Q_2, ..., Q_n\}$ is defined as follows [4].

Given a finite set $S = \{b_1, b_2, ..., b_n\}$ representing the backbone nodes in the mesh network, find n (= $|S|$) subsets $Q_i \subset S$, where each Q_i represents a quorum and $Q_i \neq \emptyset$, such that:

- **Covering**: $\bigcup_{i=1}^{n} Q_i = S$ (*i.e.*, the union of all the quorums covers all the backbone nodes)
- **Minimality**: $Q_i \not\subset Q_j$, $1 \leq$ i, j $\leq n$ and $i \neq j$ (*i.e.*, no quorum is a subset of another)
- **Mutual Intersection**: $Q_i \bigcap Q_j \neq \emptyset$, $1 \leq$ i, j $\leq n$ and $i \neq j$ (*i.e.*, any pair of quorums have non-empty intersection. But it is not required that any pair of quorums will have the same common number of backbone nodes)
- **Equal size**: $|Q_i| = K$, $1 \leq$ i $\leq n$ (*i.e.*, quorums have the same size)
- **Equal Effort**: $|\{Q_j|\ b_i \in Q_j\}| = K$, $1 \leq$ i $\leq n$ (*i.e.*, each backbone node is included in the same number of quorums)

For instance, given a set of 7 backbone nodes $S = \{b_1, b_2, ..., b_7\}$, one symmetric coterie is the set $\{\{b_1, b_2, b_4\}, \{b_2, b_3, b_5\}, \{b_3, b_4, b_6\}, \{b_4, b_5, b_7\}, \{b_5, b_6, b_1\}, \{b_6, b_7, b_2\}, \{b_7, b_1, b_3\}\}$ [4].

The mechanism to construct the coterie in our architecture is as follows. First, each backbone node acquires the identities of all the other backbone nodes. (This can be achieved in the Backbone Layer by disseminating the backbone node IDs among the backbone nodes). After that, each backbone node orders the backbone node IDs and forms the same *initial* *quorum matrix* to execute the symmetric coterie construction algorithm *QGEN* [4]. While other coterie construction algorithms can also be used to construct the coterie, we use the *QGEN* algorithm which has O($logn$) time complexity and generates near-optimal O($n^{0.63}$) quorum size. After each backbone node completes executing the *QGEN* algorithm, each backbone node knows which quorums it belongs to.

IV. Quorum-Based Match-Making

The Match-Making Service is built on top of the Quorum Layer. We assume the lower layers in the architecture provide the following information.

- The Mesh Network Layer supplies each node with a unique ID called *mesh node ID* in the mesh network.
- The Backbone Layer provides the Backbone Node function BBN(f) = b, where f is a mesh node and b is the backbone node that mesh node f is associated with.
- The Quorum Layer provides the function QUORUM(b) = S where b is a backbone node and S is the set of quorums that b belongs to.

We first introduce the notations used in the algorithm, and then describe the algorithm in three steps below.

- $\mathrm{S}_c(I) =$<mesh node ID of C, 'name' of information I>: Subscription message for information I by consumer C.
- $\mathrm{A}_p(I) =$<mesh node ID of P, 'name' of information I>: Advertisement message for information I by producer P.

(1) How a consumer subscription is handled:

- At the Mesh Network Layer, consumer C requests information I, *i.e.*, C prepares the subscription message $\mathrm{S}_c(I)$.
- At the Backbone Layer,
  - consumer C delivers $\mathrm{S}_c(I)$ to BBN(C) = v_c (v_c is the backbone node where the mesh node C is associated with), and
  - v_c selects a quorum $Q_c \in$ QUORUM(v_c) *randomly* and sends $\mathrm{S}_c(I)$ to every member of Q_c via multicast.
- At the Quorum Layer, all members of quorum Q_c become subscribers of information I, and each member of Q_c adds the entry $<\mathrm{S}_c(I), Q_c>$ to its subscription database.

(2) How a producer advertisement is handled:

- At the Mesh Network Layer, producer P advertises information I, *i.e.*, P prepares the advertisement message $\mathrm{A}_p(I)$.
- At the Backbone Layer,
  - producer P delivers $\mathrm{A}_p(I)$ to BBN(P) = v_p (v_p is the backbone node where the mesh node P is associated with), and
  - v_p selects a quorum $Q_p \in$ QUORUM(v_p) *randomly* and sends $\mathrm{A}_p(I)$ to every member of Q_p via multicast.
- At the Quorum Layer, all members of quorum Q_p become advertisers of information I and each member of Q_p adds the entry $<\mathrm{A}_p(I), Q_p>$ to its advertisement database.

(3) How to match a subscription with an advertisement: Matching of subscriptions with advertisements is achieved via the *mutual intersection* property of the coterie system. After consumer C sends its subscription to quorum Q_c and producer P sends its advertisement to quorum Q_p, any backbone node $\in Q_c \bigcap Q_p$ will be able to match $\mathrm{S}_c(I)$ and $\mathrm{A}_p(I)$. Then, one of these common nodes forwards this matching information to the proper *BBN* node $\in Q_c$ (or Q_p), which in turn forwards this matching information to the consumer (or producer).

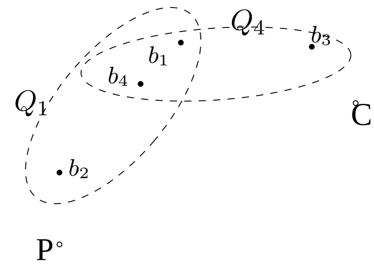

Fig. 3. The example network

Formally, let b be the backbone node with the smallest mesh node ID among all the nodes $\in Q_c \bigcap Q_p$.

- At the Quorum Layer, b sends the message $<\mathrm{A}_p(I)>$ to v_c.
- At the Backbone Layer, v_c forwards the message $<\mathrm{A}_p(I)>$ to consumer C.
- At the Mesh Network Layer, consumer C *directly* contacts producer P by relying on the underlying routing protocol and obtains information I.

Illustrative Example: We illustrate the operations of the quorum-based match-making service by using the example network shown in Figure 3. In the figure, b_1, b_2, b_3, and b_4 are the selected backbone nodes, and P and C are two other mesh nodes representing producer and consumer, respectively. After the coterie is constructed, there are four quorums formed: $Q_1 = \{b_1, b_2, b_4\}$, $Q_2 = \{b_1, b_2, b_3\}$, $Q_3 = \{b_2, b_3, b_4\}$, and $Q_4 = \{b_1, b_3, b_4\}$. To prevent confusion, only two quorums, Q_1 and Q_4, are shown in the figure.

The sequence of messages sent in the match-making process are denoted by m_i, where multicast messages are shown with lighter lines and unicast messages are shown with darker lines. In Figure 4(a), consumer C first subscribes information I and delivers $\mathrm{S}_c(I) =$<C, I> to its *BBN* node b_3 (m_1). Node b_3 then *randomly* selects, say Q_4, among the quorums Q_2, Q_3, Q_4 that it belongs to, and multicasts $\mathrm{S}_c(I)$ to members of quorum Q_4 (m_2). Each member of Q_4 learns that C subscribes information I by adding the entry $<S_c, Q_4>$ to its subscription database.

Later, producer P advertises the information I by delivering $\mathrm{A}_p(I)$ =<P,I> to its *BBN* node b_2 as shown in Figure 4(b) (m_3). Node b_2 then *randomly* selects, say Q_1, among the quorums Q_1, Q_2, Q_3 that it belongs to, and multicasts $\mathrm{A}_p(I)$ to members of quorum Q_1 (m_4). Now each member of Q_1 learns that P advertises information I by adding entry $<A_p, Q_1>$ its to advertisement database.

Since $Q_1 \cap Q_4 = \{b_1, b_4\}$, both b_1 and b_4 can match C's subscription with P's advertisement by examining their advertisement and subscription databases. Since b_1 has a smaller mesh node id, b_1 is selected to send $\mathrm{A}_p(I)$ to the

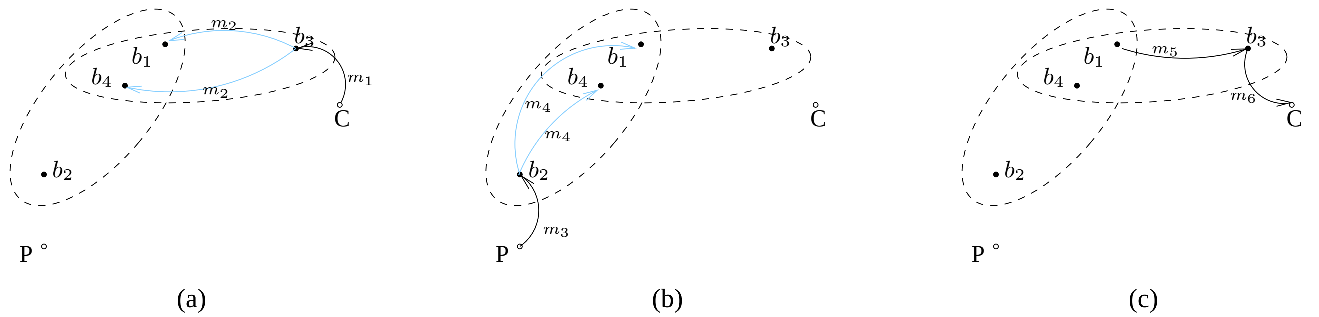

Fig. 4. Match-making example: (a) Handling consumer's subscription, (b) Handling producer's advertisement, and (c) Matching the consumer and the producer

BBN node b_3 of C, as shown in Figure 4(c) (m_5). Thus, node b_3 forwards $\mathrm{A}_p(I)$ to node C, and consumer C learns the identity of producer P and can directly contact P to obtain I.

As it is seen in the match making example above, the producer P and the consumer C floods their advertisements and subscriptions only to quorums Q_1 and Q_4, respectively. Assume that there are 1000 mesh nodes in the network. According to Table I with hop limit 2, the total number of backbone nodes will be 11.4% of the total nodes, which is about 114 nodes. According to *QGEN*'s O($n^{0.63}$) quorum size, the size of each quorum will be around 20 backbone nodes. As a result, instead of flooding to the entire mesh network or to all the backbone nodes, a producer and a consumer could be matched by flooding only to 2*20 backbone nodes.

Adapt to Topology Changes: In a mesh network, new nodes may join the network and existing nodes may (voluntarily) withdraw from participating in the mesh network or fail, which change the topology. The joining and exiting of (non-backbone) mesh nodes are transparent to the Backbone and the Quorum System Layers. The ADB protocol is continuously running to discover the joining and exiting of (non-backbone) mesh nodes. The exiting or failure of a backbone node will trigger the re-election of new backbone node(s), which triggers the construction of a new coterie quorum system. At the same time, producers and consumers follow the principle of *soft-state* protocol by periodically refreshing their advertisements and subscriptions, respectively.

V. Related Work

While our quorum-based match-making system is intended for (low mobility) wireless mesh networks, there are related work in the literature for ad hoc and sensor networks context. In particular, [5] uses cross(+) shaped *pseudo* quorums in ad hoc and sensor networks to match consumer subscriptions and producer advertisements by using different heuristics. [6] uses a Uniform Quorum System(UQS) to facilitate mobility management service in ad hoc networks, where a location database system is constructed to store the location information of each node at all the backbone nodes belonging to the same quorum. The cost analysis of the UQS is also performed to investigate the trade-off between the system reliability and the cost of location updates in the UQS scheme. [7] describes a Content Based Multicast (CBM) scheme for ad hoc networks, where the dissemination of information is performed by taking the type and velocity of the threads/resources into account to determine the region the information should be spread. Finally, [8] introduces Rumor Routing for sensor networks which describes and analyzes a method for routing queries to the nodes who observed a particular event. Each sensor node witnessing an event creates a corresponding agent (i.e., long lived packet) to be forwarded along a straight path in a randomly selected direction to spread the event, depending on a probability. Queries are also forwarded in straight paths in a randomly selected direction, until the queries meet the paths leading to the corresponding events.

VI. Conclusion

Wireless mesh networks are formed by nodes that contribute network resources and cooperates. In addition to the routing function, service/resource discovery capability is needed to facilitate high-level applications and services over large scale wireless mesh networks. In this paper, we described a layered architecture to facilitate the match-making (content-based routing) service in wireless mesh networks. In this architecture, mesh nodes autonomously elect a set of backbone nodes that participate in a quorum system construction process. By taking advantage of the mutual intersection and covering properties of the coterie quorum system, match-making could be made when producer nodes' advertisements and consumer nodes' subscriptions are matched at intersecting nodes of two quorums. With this approach, due to the much smaller quorum size in comparison with the size of the mesh network, the messaging overhead of match-making will be significantly reduced in contrast to query flooding. Moreover, backbone nodes and quorums could be formed autonomously to avoid any manual configuration required in existing service/resource discovery protocols.

References

[1] Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf, "Content-based Addressing and Routing: A General Model and its Application," Tech. Rep. CU-CS-902-00, Department of Computer Science, University of Colorado, January 2000.

[2] Chaiporn Jaikaeo and Chien-Chung Shen, "Adaptive Backbone-based Multicast for Ad hoc Networks," in *IEEE International Conference on Communications (ICC 2002)*, New York, NY, April 28-May 2 2002, vol. 5, pp. 3149–3155.

[3] Ulas C. Kozat, George Kondylis, Bo Ryu, and Mahesh K. Marina, "Virtual Dynamic Backbone for Mobile Ad Hoc Networks," in *IEEE International Conference on Communications (ICC)*, Helsinki, Finland, June 11–15 2001.

[4] Wee K. Ng and Chinya V. Ravishankar, "Coterie Templates: A New Quorum Construction Method," in *15th International Conference on Distributed Computing Systems (ICDCS)*, Vancouver, Canada, May 30–June 02 1995.

[5] Ilknur Aydin and Chien-Chung Shen, "Facilitating Match-Making Service in Ad hoc and Sensor Networks Using Pseudo Quorum," in *11th International Conference on Computer Communications and Networks (ICCCN)*, Miami, Florida, October 14–16 2002.

[6] Zygmunt J. Haas and Ben Liang, "Ad Hoc Mobility Management with Uniform Quorum Systems," *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 228–240, April 1999.

[7] Hu Zhou and Suresh Singh, "Content based multicast (CBM) in ad hoc networks," in *the ACM/IEEE Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Boston, MA, August 2000, pp. 51–60.

[8] D. Braginsky and D. Estrin, "Rumor routing algorithm for sensor networks," in *First ACM International Workshop on Sensor Networks and Applications (WSNA)*, Atlanta, Gerogia, September 28 2002.