

Enhanced Backbone Net Synthesis for Mobile Wireless Ad Hoc Networks

Huejiun Ju and Izhak Rubin
Electrical Engineering Department
University of California, Los Angeles (UCLA)
Los Angeles, CA, USA
{hju, rubin}@ee.ucla.edu

Abstract—In this paper, we present an enhanced mobile backbone network topology synthesis algorithm for constructing and maintaining a dynamic backbone structure in mobile wireless ad hoc networks. The scalability and efficiency of backbone-based routing in ad hoc networks depend on the overhead introduced by the formation of a connected backbone network and the size of the backbone network. We prove that the algorithm presented in this paper has time complexity of the order of $O(1)$, and yields a communication overhead factor of the order of $O(1)$ per node. Two rules are introduced for regulating the election of backbone nodes that are shown to enable an asynchronous, distributed and stable operation of the algorithm.

Keywords—wireless ad hoc network; backbone; connected dominating set; clustering.

I. INTRODUCTION

An ad hoc network can be represented by a unit disk graph in the following manner: Assume network nodes to have equal maximum transmission range. The topology of such an ad hoc network is modeled as a graph $G = (V, E)$ where vertices (in V) represent individual mobile stations, and where an edge (in E) is placed between two vertices if the corresponding stations are within range of each other. A dominating set problem in graph theory entails the finding of a subset of nodes with the following property: each node is either in the dominating set, or is adjacent to a node in the dominating set. A wide range of heuristic algorithms have been proposed to construct a Minimum Connected Dominating Set (MCDS) [1]–[10].

The implementation of Connected Dominating Set (CDS) formation algorithms in ad hoc networks can lead to drastic performance degradation induced by the following issues:

1) Control message losses: For a system with high nodal density, control message overhead and user data traffic can lead to heavy MAC contention and thus a high rate of Hello message losses (which gets worse with large Hello messages).

2) Asynchronous timers: In a mobile ad hoc network, there is no central controller. A node can join the network at any time from any location. Hence, to distribute network state changes to nodes that are located in an h -hop neighborhood of the focal point (where the value of h is selected as a design parameter), several Hello message cycles are required.

For conventional CDS construction algorithms to achieve their theoretical performance, perfect neighborhood data is needed. However, control message losses and asynchronous timers can both lead to incorrect neighborhood information, and in turn, cause a node to miscalculate its role in acting as a dominator or as a dominee. Thus, the resulting CDS may be disconnected or consist of an excessive number of nodes.

The mobile backbone networking architecture employed herein was first introduced in [11]–[12]. The concept, operation and characteristics of a Mobile Backbone Network (MBN) were presented. Under this approach, a multi-tier hierarchical architecture is constructed and employed for routing messages. Under the MBN protocol, nodes belong to one of two classes: regular nodes (RNs) and backbone capable nodes (BCNs). A Backbone Network (BNet) is formed by dynamically electing BCNs to act as backbone nodes (BNs). In this paper, we assume all nodes to have sufficient resources to enable them to act as BNs. Thus, all nodes are assumed to be in BCN types. Each node is assumed to be equipped with a single radio. All nodes share the same frequency band. Under these assumptions, the topology synthesis algorithm presented here provides for the asynchronous construction of a CDS. Nodes that are asynchronously elected to convert into BN state act as dominators and the remaining nodes (that are in BCN state) are dominees.

In this paper, we present an enhanced MBN topology synthesis algorithm (ETSA) that employs two BCN-to-BN conversion restricting rules to regulate excessive BCN to BN conversions induced by imperfect neighborhood information. These two rules contribute to the robustness and scalability of the MBN. At the same time, this ETSA also exhibits: constant $O(1)$ convergence time, constant $O(1)$ nodal degree in the BNet, and constant $O(1)$ control message length. As such, in addition to its asynchronous manner of operation, this algorithm is superior to other such CDS construction heuristic algorithms that have appeared in the literature.

II. RELATED WORK

A basic way to construct (at polynomial complexity) a CDS is to use a centralized greedy algorithm [2], [3]. These algorithms may often construct a CDS whose size is smaller than that resulting from the application of a distributed algorithm, but their implementation complexity generally makes them impractical. Therefore, the design of efficient

This work was supported by Office of Naval Research (ONR) under Contract No. N00014-01-C-0016, as part of the AINS (Autonomous Intelligent Networked Systems) project, by the National Science Foundation (NSF) under Grant No. ANI-0087148, and by University of California/Conexant MICRO Grant No. 04-100.

distributed algorithms has attracted recent interest. *Distributed* CDS formation algorithms can be divided into two categories: (1) *size-efficient algorithms*, and (2) *time-efficient algorithms*.

Size-Efficient Algorithms [1] [7] [8] [10]: In general, size-efficient algorithms use two phases to construct a CDS: clustering, and finding gateways (to connect the cluster-heads). In the first phase, the basic idea of the clustering approaches used by such algorithms is as follows: Initially all nodes are white. When a white node finds itself having the highest degree/ID among all its white neighbors, it becomes a cluster-head and colors itself black. All its white neighbors join in the cluster and change their color to grey. The process continues until there is no white node. The black nodes form the set of cluster-heads/dominators, forming an independent set, i.e., a dominating set in which any pair nodes are non-adjacent. This process suffers from a sequential propagation problem, which leads to long convergence time of the order of $O(n)$, where n represents the total number of nodes in the network.

The second phase is to connect the cluster-heads. Under certain protocols [1], [8], every node only has to include in the periodically broadcasted Hello message, its neighboring cluster-head list instead of the whole neighbor list. Thus, the Hello message length is of the order of $O(1)$, i.e., the message length is bounded by a value that does not depend on the number of network nodes. Actually, in [1], GPS information is required to achieve all the desired features the authors claim to achieve. In comparison, the message length is of the order of $O(\log n)$ for the protocols presented in [7] and of the order of $O(\Delta)$ (where Δ is the maximum nodal degree in the network) for the protocol described in [10]. The algorithms presented in [1], [7], [8] were shown to asymptotically approach the relative dimensionality of a minimum CDS. However, the noted long convergence time and the synchronized sequential phase-by-phase character of the operation impair the practicality of this type of CDS construction algorithms.

Time-Efficient Algorithms [4] [5] [6] [9] [12]: Some time-efficient algorithms, e.g., [4], [5], are also executed in two phases: clustering and connecting the cluster-heads. The main difference is that in the clustering phase, a node claims itself as a cluster-head if it finds itself to have the highest degree/ID among its 1-hop neighborhood or if it has the highest weight in comparison with weights exhibited by nodes that reside in one of its 1-hop neighbor's 1-hop neighborhood. Thus, the elected cluster-heads do **not** form an independent set. This design ensures the clustering phase converges in constant $O(1)$ time.

The CDS construction algorithms proposed in [6] and [9] take a different approach. The construction has two phases: a marking process to generate a CDS with rich connectivity followed by the application of pruning rules (Rule 1, Rule 2 [9], and Rule k [6]). Execution of the marking process and pruning rules can be done in $O(1)$ time. In general, 1-hop neighbor list exchanges are required for time-efficient algorithms (such as those presented in [5], [6], and [9]), inducing message lengths of the order of $O(\Delta)$. A distributed mobile backbone maintenance algorithm is proposed in [12], which also exhibits Hello message length of the order of $O(\Delta)$. The "core network" topology management algorithm proposed in [4]

requires 2-hop neighborhood data exchange, yielding $O(\Delta^2)$ message complexity per node.

We observe that the time-efficient algorithms noted above do not construct a CDS whose size has a constant approximation ratio to the size of a MCDS. The size of the CDS derived by the algorithm presented in [6] is proved there to be characterized by a "probabilistic bound," so that the average size is bounded by a constant value; yet, such a bound does not apply for certain outlier cases. The existence of outlier layouts that violate the average size bound is also the case for the algorithms presented in this paper. It is however noted that the probability that such an outlier layout is very low.

III. ENHANCED TOPOLOGY SYNTHESIS ALGORITHM

A. MBN Topology Synthesis Algorithm

Every node has two timers: *Short_Timer* and *Long_Timer* (in our design, *Long_Timer* is three times *Short_Timer*). There is no time synchronization between nodes; every node maintains its own time.

Whenever the *Short_Timer* expires at a node, the node broadcasts a *Hello* message to its direct neighbors. The Hello message contains the node's ID, status, *weight*, associated BN ID, BN-to-BCN indicator, and its "BN neighbor list". The *weight* of a node can be based on its ID, degree, capability, congestion level, or on some stability measure. Through periodic Hello message exchange, each node learns its 1-hop neighborhood and 2-hop BN neighborhood. *A node does not learn its complete 2-hop neighborhood, as assumed by typical CDS construction algorithms.* The notations used in the rest of this paper are summarized in Table 1.

Whenever the *Long_Timer* expires at a node, the node updates its neighbor list based on the number of Hello messages received within the previous period. In accordance with its type, this node then executes the following operations: A BCN runs the association and the BCN-to-BN conversion algorithms; a BN runs the BN-to-BCN conversion algorithm.

1) Association Algorithm:

BCNs will try to find a BN v with highest *weight* in its 1-hop neighborhood to associate with. If no neighboring BN is detected, the node attempts to associate with a BCN v —selecting among all its neighboring BCNs, including itself, the one with the highest *weight* (node ID is used for tie breaking). This selected node v is identified as its associated BN in the Hello messages that it subsequently periodically issues. When a BCN or BN receives a Hello message indicating itself as the associated BN, it proceeds to include this node in its client list.

2) BCN to BN Conversion Algorithm:

Such a conversion will take place if **any** of the following conditions are satisfied at a BCN u :

- (1) Client coverage: $N_{BN}(u) = \emptyset$ and $wt(v) < wt(u)$, $\forall v \in N_{BCN}(u)$.
- (2) Local 2-hop BNet connectivity: (illustrated in Fig. 1 (a))
 $\exists \{v, w\} \subseteq N_{BN}(u)$, $\{v \in N_{BN}(v)\} \{w \in N_{BN}(w)\} = \emptyset$ and
 $wt(x) < wt(u)$, $\forall x \in N_{BCN}(u)$, $\{v, w\} \subseteq N_{BN}(x)$.

- (3) Local 3-hop BNet connectivity: (illustrated in Fig. 1 (b))
 $\exists v \in N_{BN}(u), w \in N_{BCN}(u), \{v \in N_{BN}(v)\} \quad N_{BN}(w) = \emptyset$
 and there does not exist a BCN x , such that
 $x \in N_{BCN}(u), v \in N_{BN}(x), N_{BN}(x) \quad N_{BN}(w) \neq \emptyset$.

3) *BN to BCN Conversion Algorithm:*

Such a conversion will take place if **all** of the following conditions are satisfied at a BN u :

- (1) Client coverage condition: $\forall v \in C(u), |N_{BN}(v)| > 1$.
- (2) Local 2-hop BNet connectivity: $\forall \{v, w\} \subseteq N_{BN}(u)$ either
 - i) $(v, w) \in E$, and at least one of the following four conditions are satisfied: $wt(v) > wt(u), wt(w) > wt(u), ind(v) = 0, ind(w) = 0$ (Fig. 2 (a)), or,
 - ii) $\exists x, x \in N_{BN}(v) \quad N_{BN}(w)$, and either $wt(x) > wt(u)$ or $ind(x) = 0$ (Fig. 2 (b)).
- (3) Local 3-hop BNet connectivity: $\forall \{v, w\}, v \in N_{BN}(u), w \in N_{BCN}(u)$ either
 - i) $v \in N_{BN}(w)$, and either $wt(v) > wt(u)$ or $ind(v) = 0$ (Fig. 2 (c)), or,
 - ii) $\exists x, x \in N_{BN}(v) \quad N_{BN}(w)$, and either $wt(x) > wt(u)$ or $ind(x) = 0$ (Fig. 2 (d)).

For BN u , if condition (1) is not satisfied or either condition (2) or condition (3) are not satisfied because there does not exist a sufficiently short alternate path between at least one pair of u 's BN neighbors, or between a pair of BN and BCN neighbors, BN u sets $ind(u)$ equal to "0". This indicates that u converting from BN to BCN will definitely break the network connectivity. If condition (1) is satisfied and either condition (2) or condition (3) are not satisfied because the BNs on the alternative routes have higher weights, $ind(u)$ is set to "1".

TABLE I. NOTATION

$wt(u)$	The weight of node u . The relation $wt(u) > wt(v)$ is defined to indicate that u 's weight is higher than v 's or that u and v have the same weight but u ' ID is higher than v 's ID.
$ind(u)$	BN-to-BCN indicator.
$N_{BN}(u)$	The set of 1-hop neighbors of node u that are in BN status.
$N_{BCN}(u)$	The set of 1-hop neighbors of node u that are in BCN status.
$C(u)$	The set of clients of node u .

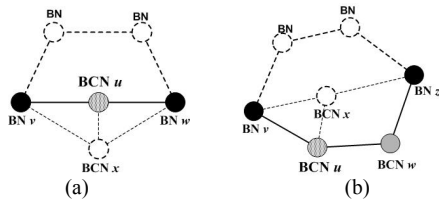


Figure 1. BCN to BN conversion conditions

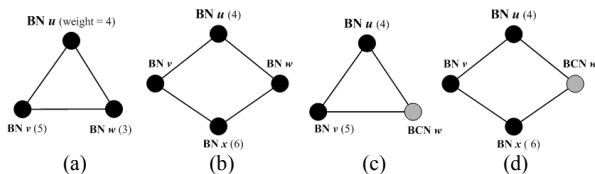


Figure 2. BN to BCN conversion conditions

B. *Restricting Conversions of BCN to BN*

Rule 1: A BCN should not convert to a BN if the number of its BN neighbors is higher than a threshold level, denoted as the *BN_Neighbor_Limit*.

Our principle of operation is that if a BCN is surrounded by a large number of BNs, it is unlikely that the network will become disconnected if the BCN does not convert to a BN. It is possible that, as a result of this, several of its BN neighbors will not be connected by a path whose length is ≤ 2 hops; yet, due to the high local density of nodes, it is very likely that the BNs will remain connected.

Theorem 1: If a BCN has more than 9 BN neighbors, these BN neighbors must all belong to a single connected BNet.

Proof: Let R represent the transmission range of nodes. As illustrated in Fig. 3 (a), in order to separate the peripheral BNs (BN 1 ~ 9) into two separated BNets, there must be two pairs of neighboring BNs out of each other's radio transmission range, e.g. (BN 1, BN 9) and (BN 5, BN 6). At the same time, the distance between every other BN (e.g., BN 1 and BN 3) needs to be larger than R . Thus, based on geometry, the maximum number of BN neighbors that a BCN can have is 9, for the case under which the peripheral BNs may be divided into separate groups. If there are more than 9 BN neighbors, these BN neighbors must all belong to the same connected BNet so that the underlying BCN (located at the center of the figure) does not have to consider converting to BN status. ■

Theorem 2: If a BCN has at least 9 BN neighbors, it does not have to convert to a BN for client coverage purposes.

Proof: Noting from the geometrical layout depicted in Fig. 3 (b) that a client node located in BCN u 's coverage disk is within a distance R from at least one of the 9 peripheral BNs—thus, all of the non-backbone neighbors of BCN u are already covered by these BNs, indicating that it does not have to convert to BN status for client coverage purposes. ■

We thus conclude that the *BN_Neighbor_Limit* threshold level should be set to a value that is no smaller than 9.

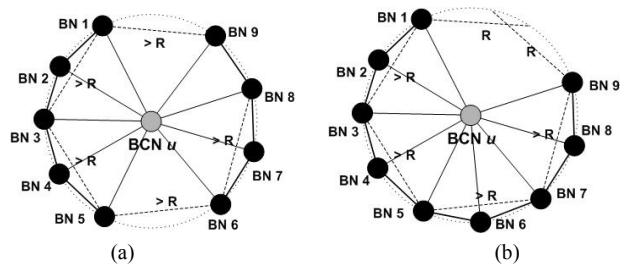


Figure 3. Maximum number of BN neighbors a BCN can have when considering converting to a BN

Rule 2: A BCN should not convert to a BN if the number of its BN neighbors increases by at least one within the previous *Short_Timer* period.

When a BCN u has just converted to a BN, its 1-hop neighbors will recognize its new status once they receive its next Hello message. However, u 's neighbors have to wait at most an additional *Short_Timer* period before they send out their next Hello message with the updated BN neighbor list.

Since we assume the actions taken by different nodes proceed in an asynchronous manner, if one of u 's BCN neighbor acts on its conversion to a BN before receiving the updated Hello messages, its conversion operation may be unnecessary. Some of its neighbors' conversions from BCN to BN may have enhanced the network connectivity to a sufficient level. Rule 2 was created to reduce the occurrence of such conversions.

IV. PERFORMANCE ANALYSIS

A. Size of the Backbone Network

Theorem 3: The number of BN neighbors that a node can have is upper-bounded by a constant value that is, with high probability, independent of the number of nodes in the network.

Proof: According to our algorithm, a backbone node is elected to provide for either coverage of a client node and/or to provide for BNet connectivity. We prove that when nodes are randomly and uniformly distributed across the area, the probability that the size of the BNet (for client coverage) is higher than a given level K is very low. To obtain a more practical characterization of the BNet size resulting from the application of our scheme, we carry out the analysis here on the upper bound of the backbone network size in considering the election of nodes to provide BNet connectivity.

We randomly select a backbone node u as shown in Fig. 4 (a). The BN neighbors of BN u must be located on or inside the circle whose center is at u and whose radius is equal to R . For obtaining an upper bound, we consider the extreme situation under which all BN neighbors of u are located on this circle. We note that if the radius of this circle is smaller than R , then a smaller number of such BNs will necessarily be elected because of the overlapping coverage areas. The distance between every other BN on the ring (e.g. "BN 1 and BN 3") must be larger than R . Otherwise a BN will have to convert to BCN status because of redundancy in connectivity, according to BN-to-BCN conversion condition (2). If there is a BN inside the ring, e.g. BN v , it will convert to BCN status, since BN u already provides a 2-hop path between any pair of BNs located within its coverage area. Observing the geometry represented in Fig 4 (a), we conclude that the number of BN neighbors of any BN is upper-bounded by 11. Note that Rule 1 does not affect this bound because BN u may exist before the peripheral BNs (BN 1 ~ 11) show up.

Consider the case where the node under consideration is a BCN, as shown in Fig. 4 (b). If a BN is located inside the ring, such as BN v in Fig. 4 (b), the latter will stay as a BN because no other BN can provide a 2-hop path between BN 3 and BN 11. If there is another BN located inside the ring, such as BN w , which can provide a 2-hop path between any two BNs among BN 11, BN 1, BN 2, BN 3 and BN 4, then BN w will stay as a BN while BN v will convert to BCN status. On the other hand, if there is another BN, such as BN x , which provides a 2-hop path for any two BNs among BN 8, BN 9, BN 10, BN 11, then BN x can co-exist with either BN v or BN w . Thus, there can be up to 11 BNs in total located inside the ring. We conclude that for any non-backbone node, the number of BN neighbors of this selected node is upper-bounded by 22. ■

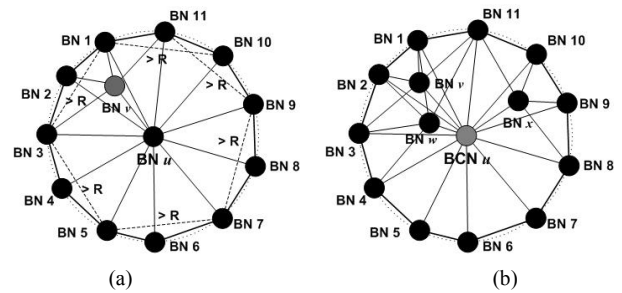


Figure 4. Maximum Number of BN Neighbors

B. Message Overhead

Theorem 4: The message complexity of the MBN topology synthesis algorithm is of the order of $O(1)$ per node.

Proof: The Hello messages include only the "BN Neighbor List" instead of the full neighbor list. Noting the number of BN neighbors of a BN or BCN to be bounded by a constant number (11) or (22), we conclude that the size of each Hello message is of the order of $O(1)$. Each node sends only one Hello message per *Short_Timer* period. Thus, we conclude that the message complexity is of the order of $O(1)$ per node. ■

C. Convergence and Time Complexity

Theorem 5: The enhanced MBN topology synthesis algorithm converges in $O(1)$ time.

Proof: Assume that all nodes are initially set to be in the BCN state. Following the second *Long_Timer* period, every node has learned the identity of the nodes in its 1-hop neighborhood, as well as the identity of the nodes in its 2-hop neighborhood that are BNs. Using this information, each BCN decides if it should convert itself into a BN or rather act to associate with a neighboring BCN. According to Rule 2, a BCN cannot convert to BN status if the number of its BN neighbors increases by at least one within the previous *Short_Timer* period. In the worst case, a BCN that needs to convert to BN status waits for its neighboring nodes to convert from BCN to BN one-by-one (one per *Long_Timer* period) before the original BCN can proceed with its own conversion. In the end, this BCN may have up to 22 BN neighbors before it converts to a BN. Thus, in the worst case scenario, the BCN to BN conversion process can take 22 cycles. However, restricted by Rule 1, a BCN will stop considering conversion to a BN when it has more than 9 (for a *BN_Neighbor_Limit* that is set to 9) BN neighbors, which takes 9 cycles.

After the basic backbone topology is formed, the backbone network reduction processes take place. Some BNs may convert back to BCNs, as dictated by the specified BN redundancy check condition. This process takes only one *Long_Timer* period. Furthermore, the reduction operation does not disrupt the connectivity of the backbone network. Hence, no further BCN to BN conversions will be triggered. We conclude that the ETSA scheme converges in at most 12 update cycles, corresponding to the 12 underlying *Long_Timer* periods noted above. ■

V. PERFORMANCE BEHAVIOR

The simulation models used for performance evaluation were implemented in QualNet v3.6.1. The Distributed Coordination Function (DCF) of IEEE 802.11 is used as the MAC layer protocol. The channel data rate is 2 Mbps. Each simulation has been run for 300 seconds, and the results are averaged over 5 randomly generated topologies. We use nodal degree as the weight for each node. *Short_Timer* is set to 2 seconds. We simulate a static ad hoc wireless network that consists of 100 ~ 500 nodes, randomly placed in a 1500m x 1500m area. The simulation results are shown in Fig. 5.

We observe that when employing the ETSA scheme with Rule 1 and Rule 2, the size of the backbone network stays almost the same while the nodal density increases. On the other hand, the algorithm without any restricting rules produces a BNet that is about 6 times larger when there are 500 nodes in the network, due to high nodal density induced control message losses. Though theoretically Rule 1 and Rule 2 do not affect the backbone network size, these restricting rules are very effective in controlling the backbone network size in practical implementation. The average number of BN neighbors per node obtained under ETSA is about 6, nicely under the theoretical bound, 22. Thus, the per node Hello message rate of ETSA stays below 0.2 kbps, and is independent of the nodal density. The ETSA scheme with two restricting rules converges in less than 8 cycles. When only Rule 2 is applied, the convergence time is longer. Without Rule 2, the algorithm converges in 5 cycles but generates a bigger BNet.

For throughput performance evaluation, we impose 50 simultaneous UDP traffic flows with randomly selected disjoint source and destination nodes. The inter-arrival time of packets follows an exponential distribution with an average inter-arrival time of 0.21 sec; the packet size is set to 512 Bytes, leading to a total offered traffic rate of 487 kbps. Based on the MBN structure described above, we modify the Ad hoc On-demand Distance Vector (AODV) routing algorithm—yielding a *Mobile Backbone Network Routing (MBNR)* protocol—by imposing the following requirement: only BNs forward route request (RREQ) packets. In this way, a source node that becomes active will search for a route by distributing RREQ packets only across the BNet. We observe that as the nodal density increases, the throughput performance of AODV deteriorates and so does MBNR without any restricting rules. The high rate of generated RREQ packets imposes a network overload, which leads to the observed throughput degradation. The excessive MAC contention induced by this heavy loading further worsens the situation.

VI. CONCLUSIONS

In this paper, an enhanced Mobile Backbone Network (MBN) topology synthesis algorithm (ETSA) is presented. Two BCN-to-BN conversion restricting rules are introduced to regulate excessive BCN-to-BN conversions induced by imperfect neighborhood information from control message losses and asynchronous operation. The same desirable analytical features are retained—constant $O(1)$ convergence time, constant $O(1)$ nodal degree in the BNet, and constant $O(1)$ control message length.

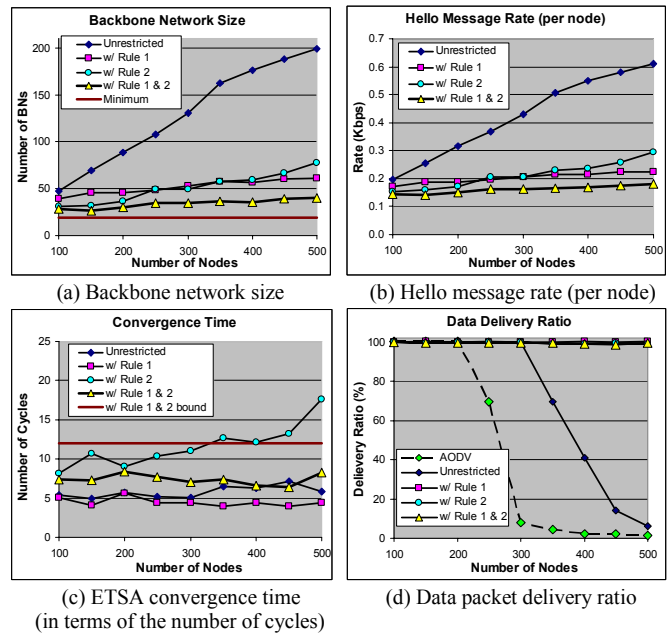


Figure 5. Performance behaviors of a static ad hoc network consisting of 100 ~ 500 nodes with total 487 kbps traffic load

REFERENCES

- [1] K. Alzoubi, X.-Y. Li, Y. Wang, P.-J. Wan and O. Frieder, "Geometric Spanners for Wireless Ad Hoc Networks", *IEEE Trans. Parallel and Distributed Systems*, vol. 14, pp. 408–421, Apr. 2003.
- [2] S. Guha and S. Khuller, "Approximation Algorithm for Connected Dominating Sets", *Algorithm*, 1998
- [3] B. Das and V. Bharghavan, "Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets", in *Proc. IEEE Int. Conf. Communications (ICC)*, June 1997, pp. 376–380.
- [4] R. Sivakumar, P. Sinha, and V. Bharghavan, "CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm", *IEEE Journal on Selected Area in Communications*, vol. 17, pp. 1454–1465, Aug. 1999.
- [5] L. Bao and J. J. Garcia-Luna-Aceves, "Topology Management in Ad Hoc Networks", in *Proc. ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, June 2003, pp.129–140.
- [6] F. Dai and J. Wu, "An Extended Localized Algorithm for Connected Dominating Set Formation in Ad Hoc Wireless Networks", *IEEE Trans. Parallel and Distributed Systems*, vol. 15, pp. 908–920, Oct. 2004.
- [7] K. Alzoubi, P. Wan, and O. Frieder, "Message-Optimal Connected Dominating Sets in Mobile Ad Hoc Networks", in *Proc. ACM Int. Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2002.
- [8] W. Lou and J. Wu, "A Cluster-Based Backbone Infrastructure for Broadcasting in MANETs", in *Proc. IEEE Int. Parallel and Distributed Processing Symposium*, Apr. 2003.
- [9] J. Wu and H. Li, "On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks", in *Proc. ACM Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999, pp. 7–14.
- [10] U. C. Kozat, G. Kondylis, B. Ryu and M. K. Marina, "Virtual Dynamic Backbone for Mobile Ad Hoc Networks", in *Proc. IEEE Int. Conf. Communications (ICC)*, June 2001, pp. 250–255.
- [11] I. Rubin, A. Behzad, H. Ju, R. Zhang, X. Huang, Y. Liu, and R. Khalaf, "Ad Hoc Wireless Networks with Mobile Backbones", in *Proc. IEEE Int. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2004, vol.1, pp. 566–573.
- [12] I. Rubin, X. Huang, Y. Liu, and H. Ju, "A Distributed Stable Backbone Maintenance Protocol for Ad Hoc Wireless Networks", in *Proc. IEEE Vehicle Technology Conference (VTC)*, 2003, pp. 2019–2022.