Meghanathan, Natarajan; Terrell, Michael
An Algorithm to Determine Stable Connected Dominating Sets for Mobile Ad hoc Networks using Strong Neighborhoods
International Journal of Combinatorial Optimization Problems and Informatics
Morelos, México

Available in: http://www.redalyc.org/articulo.oa?id=265224454009
An Algorithm to Determine Stable Connected Dominating Sets for Mobile Ad hoc Networks using Strong Neighborhoods

Natarajan Meghanathan¹ and Michael Terrell²
¹ Jackson State University, USA
² Grambling State University, USA
Corresponding Author E-mail: natarajan.meghanathan@jsums.edu

Abstract. We propose an algorithm to determine stable connected dominating sets (CDS) for mobile ad hoc networks using the notion of strong neighborhood (SN). The SN-CDS algorithm takes an input parameter called the Threshold Neighborhood Distance Ratio (TNDR); for an edge to be part of a strong neighborhood-based topology, the ratio of the physical Euclidean distance between the end nodes of the edge to that of the transmission range per node has to be less than or equal to the TNDR. The algorithm prefers to include nodes (into the SN-CDS) in the decreasing order of the number of uncovered strong neighbors until all nodes in the network are covered. We observe the SN-CDS (TNDR < 1) to have a significantly longer lifetime than a maximum density-based CDS (MaxD-CDS with TNDR = 1.0); the tradeoff being a slightly larger CDS Node Size and hop count per path.

Keywords: Strong Neighborhood, Connected Dominating Set (CDS), Stability, Mobile Ad hoc Networks, Maximum Density, Algorithm.

1 Introduction

A mobile ad hoc network (MANET) is a dynamic distributed system of wireless nodes that operate with limited battery charge and resource constraints (like memory and processing capacity). The bandwidth of a MANET is limited compared to that of a wired network and the communication medium is shared. Nodes often move randomly and independent of each other. MANET routes are multi-hop in nature due to the limited transmission range of the wireless nodes. Due to node mobility, these paths have a limited lifetime and have to be often reconfigured to continue communication. Due to the resource and mobility constraints, MANET routing protocols prefer to discover a route only when needed rather than proactively determining and maintaining them [1,2]. On-demand route discovery is conducted through a flooding-based route-request-reply cycle wherein a source node (that has data to send to a destination or to a multicast group) broadcasts a Route Request (RREQ) message to its neighbors [3]. Each node broadcasts the RREQ message exactly once in its neighborhood. This way, if the underlying network is connected, the RREQ message definitely reaches the targeted destination node(s) through one or more paths; the path that best satisfies the route selection principles of the routing protocol is chosen and a Route Reply (RREP) message is sent by the destination on the selected path to the source.

Flooding maximizes the chances that at least one RREQ message reaches the destination (including on the minimum-hop path to the destination). However, flooding is often considered to trigger the “broadcast storm problem” [4] as it is associated with redundant retransmissions; because, each node in the network broadcasts the RREQ message once to its neighborhood. Several strategies have been considered to minimize the broadcast storm problem and the most significant and effective among them is to use a connected dominating set (CDS) of the underlying network graph at the time of route discovery (e.g., [5][6][7]). A CDS of a network of nodes is the subset of nodes such that every node in the network is either in the CDS or is a neighbor node (that is located within the transmission range) of a node in the CDS [8]. A non-CDS node is said to be covered if it has at least one neighbor node in the CDS. If we could find a CDS that has the least number of constituent nodes (called the Minimum CDS, MCDS) to cover all the nodes in the network, we can incur the minimum number of retransmissions if only the nodes constituting the CDS broadcast a message (such as the RREQ message) in the neighborhood. However, the problem of determining a MCDS is considered to be NP-complete [8] and hence several heuristics have been proposed to approximate a MCDS. The Maximum density-based CDS (MaxD-CDS) algorithm [9] that was earlier proposed by us is one such heuristic. The common thread among the heuristics proposed for approximating the MCDS (including the MaxD-CDS algorithm) is to include a covered node that has the largest number of uncovered neighbor nodes into the CDS, during each of the iterations of
the algorithm until all the nodes in the network are covered. This way, it has been observed [9] that the approximated MCDS will have the lowest or closer to the lowest number of constituent CDS nodes.

The motivation for the research leading to this paper stemmed from observations in our earlier research works (e.g. [9][10]) that an MCDS (like MaxD-CDS) is significantly unstable in the presence of node mobility when used in the context of a MANET. This could be attributed to the requirement to cover all the nodes of a network with the lowest possible number of constituent MCDS nodes and with node mobility, the chances of a non-CDS node not having a MCDS node as its neighbor in the near future is quite high. Also, since the MCDS spans the entire network and is composed of the least number of constituent nodes, the physical Euclidean distance between any two MCDS nodes is often close to the transmission range of the nodes at the time of the CDS construction and is vulnerable to break (i.e., the two MCDS nodes move out of their transmission range) at any time in the near future.

In this paper, we propose the notion of considering only the “Strong Neighborhood” (SN) of a node, rather than considering all neighbor nodes within the transmission range of the node (referred to as the “Open Neighborhood”), as the neighbor nodes that will be covered due to the inclusion of a node into the CDS. Accordingly, our CDS is referred to as the SN-CDS. The Strong Neighborhood of a node is defined based on a parameter called the Threshold Neighbor Distance Ratio (TNDR) that can be at most 1. A node j at a physical Euclidean distance of r from node i is said to be in the strong neighborhood of node i if r/R ≤ TNDR where R is the fixed transmission range of all nodes in the network (i.e., we assume a homogeneous MANET). The construction of the SN-CDS starts with the inclusion of the node with the maximum number of strong neighbors into the CDS. As usual for any CDS algorithm, a non-CDS node is said to be covered if at least one of its neighbor nodes is in the SN-CDS. The set of covered nodes are considered to be the candidate nodes for inclusion into the SN-CDS and they are preferred in the decreasing order of the number of uncovered nodes in their strong neighborhood. We continue the inclusion of the covered nodes into the SN-CDS, one node at a time, until all the nodes in the network are covered by at least one node in the SN-CDS. If TNDR = 1, then SN-CDS corresponds to the MaxD-CDS. Hence, for purpose of clarity in the rest of the paper, we refer to an SN-CDS as a CDS formed using TNDR < 1 and a CDS formed using TNDR of 1 is referred to as MaxD-CDS.

The rest of the paper is organized as follows: In Section 2, we describe the related work on CDS algorithms for MANETs. Section 3 describes the SN-CDS construction algorithm, analyzes its run-time complexity as well as describes the procedure used to validate the existence of a pre-determined SN-CDS during subsequent time instants. Section 4 presents the simulation results comparing the performance of an SN-CDS with that of the MaxD-CDS and interprets them with respect to metrics such as the CDS lifetime, Hop count per path between any two nodes in the network, CDS node size and the CDS edge size. We also identify the values observed for the above performance metrics when SN-CDS is operated with a TNDR value of 0.9 (the largest value of TNDR that is less than 1.0 used in our simulations) vis-à-vis a MaxD-CDS (TNDR = 1.0) and quantify the tradeoffs between the two CDS construction strategies (strong neighborhood vs. open neighborhood). We consider a MCDS-based heuristic to use an “Open Neighborhood” as it considers all nodes within the transmission range of a node to be the potential neighbor nodes that are covered if the node gets included into the CDS. Section 5 discusses the extension of the SN-CDS algorithm for heterogeneous systems. Section 6 concludes the paper. Throughout the paper, the terms ‘heuristic’ and ‘algorithm’, ‘route’ and ‘path’, ‘node’ and ‘vertex’, ‘edge’ and ‘link’ are used interchangeably. They mean the same.

2 Related Works

Very few algorithms are available in the MANET literature to determine stable connected dominating sets. The MCMIS (Maximal Independent Set with Multiple Initiators) algorithm proposed in Wang et al. [11] constructs a stable virtual backbone through a localized approach, consisting of two phases: In the first phase, a forest consisting of multiple dominating trees, each rooted at a different initiator and comprising of a subset of nodes in the network, is constructed. The dominating trees, each started by its initiator, are concurrently constructed (i.e., in parallel). The candidacy of the nodes to initiate the construction of the dominating trees is decided based on the decreasing order of values of a weighted tuple comprising of the three metrics (stability, effective degree, ID) associated with the nodes. In the second phase, the dominating trees are interconnected, through the overlapping edges, to form a complete virtual backbone.

In Sheu et al. [12], the authors present a link-stability based algorithm that ranks nodes based on the number of non-weak links associated with them and prefers nodes with a larger number of non-weak links for inclusion into a CDS. The algorithm categorizes a link as non-weak, if the strength of the beacon signal received on that link is above a threshold. In Meghanathan [10], the authors had proposed a minimum velocity-based algorithm that prefers slow-moving nodes to be the constituent nodes of a CDS, the tradeoff being an apparent increase in the number of nodes constituting the CDS. In Fly and Meghanathan [13],
In addition to those defined above, the SN-CDS algorithm uses the following auxiliary variables and functions:

- **Physical Euclidean distance**: The physical Euclidean distance between two nodes \(i\) and \(j\) located at \((X_i, Y_i)\) and \((X_j, Y_j)\) respectively is given by \(\text{dis}(i, j) = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}\).
- **Open Neighborhood**: The open neighborhood of a node \(i\) (denoted \(\text{ON}_i\)) is the set of neighbor nodes such that the physical Euclidean distance between node \(i\) and each node \(j\) in this set is \(s \leq R\). Every node \(j \in \text{ON}_i\) is simply referred to as a “neighbor” of node \(i\).
- **Threshold Neighbor Distance Ratio (TNDR)**: The TNDR is defined as the maximum value for the ratio of the physical Euclidean distance between a node \(i\) and a node \(j\) in its Open Neighborhood and the transmission range \(R\) so that node \(j\) is considered a “strong” neighbor of node \(i\). The value of TNDR can be at most 1.0.
- **Strong Neighborhood**: The strong neighborhood of a node \(i\) (denoted \(\text{SN}_i\)) is the subset of nodes in its open neighborhood, \(\text{ON}_i\), such that the ratio of the physical Euclidean distance between node \(i\) and any node \(j\) in \(\text{SN}_i\) is less than or equal to the TNDR.
- **Static Network Graph**: A snapshot of the network topology at a particular time instant.

### 3 Algorithm to Construct Strong Neighborhood-based Connected Dominating Set (SN-CDS)

#### 3.1 Definitions and Assumptions

- All nodes operate under an identical transmission range, \(R\).
- Physical Euclidean distance: The physical Euclidean distance between two nodes \(i\) and \(j\) located at \((X_i, Y_i)\) and \((X_j, Y_j)\) respectively is given by \(\text{dis}(i, j) = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}\).
- Open Neighborhood: The open neighborhood of a node \(i\) (denoted \(\text{ON}_i\)) is the set of neighbor nodes such that the physical Euclidean distance between node \(i\) and each node \(j\) in this set is \(s \leq R\). Every node \(j \in \text{ON}_i\) is simply referred to as a “neighbor” of node \(i\).
- Threshold Neighbor Distance Ratio (TNDR): The TNDR is defined as the maximum value for the ratio of the physical Euclidean distance between a node \(i\) and a node \(j\) in its Open Neighborhood and the transmission range \(R\) so that node \(j\) is considered a “strong” neighbor of node \(i\). The value of TNDR can be at most 1.0.
- Strong Neighborhood: The strong neighborhood of a node \(i\) (denoted \(\text{SN}_i\)) is the subset of nodes in its open neighborhood, \(\text{ON}_i\), such that the ratio of the physical Euclidean distance between node \(i\) and any node \(j\) in \(\text{SN}_i\) is less than or equal to the TNDR.
- Static Network Graph: A snapshot of the network topology at a particular time instant.

#### 3.2 Auxiliary Variables, Functions and Initialization

In addition to those defined above, the SN-CDS algorithm uses the following auxiliary variables and functions:

- **SN-CDS-Node-List** – includes all the nodes that are part of the strong neighborhood based CDS, initialized to \(\Phi\).
- **Covered-Nodes-List** – includes nodes that are either in \(\text{SN-CDS-Node-List}\) or covered by a node in the \(\text{SN-CDS-Node-List}\), initialized to \(\Phi\).
- **Uncovered-Nodes-List** – includes nodes that are not covered by any node in the \(\text{SN-CDS-Node-List}\), initialized to the Vertex set of the input static graph.
- **Priority-Queue** – includes nodes that are in the \(\text{Covered-Nodes-List}\) and not in the \(\text{SN-CDS-Node-List}\). The nodes in the priority queue are probable candidates for inclusion to the \(\text{SN-CDS-Node-List}\). The list is sorted in the decreasing order of the number of uncovered strong neighbors of the nodes. A dequeue operation returns the node with the largest number of uncovered strong neighbors; if there is a tie, then a node is randomly chosen from among the contending nodes. The priority queue is initialized to \(\Phi\).
- **MaxUncoveredStrongNeighbors** – the maximum value for the number of uncovered strong neighbors of any node in the network at the beginning of the algorithm, initialized to \(- \infty\) to start with and is computed while determining the strong neighbors of the nodes in the network.
- **UncoveredStrongNeighbors\((u)\)** – the set of all strong neighbors of node \(u\) that are not yet covered. Initially, \(\forall u, \text{UncoveredStrongNeighbors}(u) = \Phi\).
3.3 Function to Compute the Strong Neighborhood of the Nodes

To determine the set of strong neighbors of a node $u$, we basically compute the physical Euclidean distance between $u$ and every other node $v$ in the network graph; if the ratio of this distance to the transmission range $R$ is less than or equal to the TNDR value (both $R$ and TNDR are passed as input parameters to the SN-CDS algorithm), then $v$ is added to the Strong Neighborhood of $u$, $\text{SN}_u$. After evaluating all possible vertices $v$, the value of the auxiliary variable $\text{maxUncoveredStrongNeighbors}$ is compared with size of the set $\text{SN}_u$. If the value of $\text{maxUncoveredStrongNeighbors}$ is less than $|\text{SN}_u| > 0$, then it is updated to $|\text{SN}_u|$. The pseudo code of the function to compute the strong neighborhood of the nodes and the maximum value for the number of uncovered strong neighbors is given below in Figure 1.

```
Begin Compute-Strong-Neighborhood
for every vertex $u \in V$ do
  for every vertex $v \in V$ do
    if $\frac{\text{dis}(u,v)}{R} \leq \text{TNDR}$ then
      $\text{SN}_u = \text{SN}_u \cup \{v\}$
    end if
  end for
  // loop for every possible neighbor node $v$ of $u$
  if $\text{maxUncoveredStrongNeighbors} < |\text{SN}_u|$ then
    $\text{maxUncoveredStrongNeighbors} < |\text{SN}_u|$ end if
end for
// loop for vertex $u$
return $\text{maxUncoveredStrongNeighbors}$
End Compute-Strong-Neighborhood
```

Fig. 1. Pseudo Code to Compute the Strong Neighborhood of the Nodes in the Network

3.4 Selection of the Starting Node to Include into the SN-CDS

The node that has the $\text{maxUncoveredStrongNeighbors}$ is chosen the start node (i.e., the first node) to be included into the SN-CDS. If more than one node has the same value for $\text{maxUncoveredStrongNeighbors}$, then to be fair to all the nodes in the network, we randomly choose one among these contending nodes as the starting node (the first node) to be included in to the SN-CDS-Node-List. We form a candidate list of all the contending nodes and randomly choose one among these nodes as the Start Node of the SN-CDS algorithm. The pseudo code for this randomization function is given in Figure 2. A similar approach is used to break the tie among the nodes contending to be extracted from the Priority Queue for inclusion into the SN-CDS-Node-List during the subsequent iterations of the algorithm.
3.5 Description of the Algorithm to Construct the SN-CDS

The SN-CDS construction algorithm (pseudo code in Figure 3) primarily works as follows: The algorithm inputs a snapshot of the network (referred to as the static graph) at the time instant during which we want to determine the SN-CDS. The algorithm also inputs the Threshold Neighborhood Distance Ratio (TNDR) and transmission range (R) for the network. For every node in the vertex set V, the nodes constituting the strong neighborhood is constructed by calling the Compute-Strong-Neighborhood function (pseudo code in Figure 1). This function also returns the maximum value for the number of uncovered strong neighbors for any node in the network, referred to as maxUncoveredStrongNeighbors. The node that has this maximum value of uncovered strong neighbors (i.e., the size of the Strong Neighborhood set of the node equals maxUncoveredStrongNeighbors) is selected as the Start Node (the first node to be included into the SN-CDS-Node-List) by calling the Choose-Start-Node function (pseudo code in Figure 2). If there is a tie, the Choose-Start-Node function randomly returns one among the contending nodes as the Start Node.

When the Start Node is added to the SN-CDS-Node-List, all of its strong neighbors are said to be covered; these nodes are removed from the Uncovered-Nodes-List and added to the Covered-Nodes-List and to the Priority-Queue. If both the Uncovered-Nodes-List and the Priority-Queue are not empty, we dequeue the Priority-Queue to extract a node s that is not yet in the SN-CDS-Node-List and has the largest number of uncovered strong neighbor nodes. All the uncovered strong neighbor nodes of s are now removed from the Uncovered-Nodes-List and added to the Covered-Nodes-List as well as to the Priority-Queue. The number of uncovered strong neighbors of each node in the network is then updated based on the additional node coverage obtained during the iteration and accordingly, the Priority-Queue is re-sorted in the decreasing order of the number of uncovered strong neighbors of the nodes in the queue. The above procedure is repeated for several iterations until the Uncovered-Nodes-List becomes empty or the Priority-Queue becomes empty.

Note that during an iteration, if the node s extracted from the Priority-Queue has all its strong neighbor nodes already covered, then it implies that all the other nodes, if any, in the Priority-Queue also have zero uncovered strong neighbor nodes. However, we have not yet broken from the while loop (i.e. the Uncovered-Nodes-List is not yet empty), indicating that the underlying network based on the strong neighborhood of the nodes is not connected and hence the algorithm returns NULL (i.e. a SN-CDS for the entire network does not exist). Also, even after exiting from the while loop, if the Priority-Queue becomes empty and the Uncovered-Nodes-List has at least one node, then the underlying network is considered to be disconnected (based on the strong neighborhood of the nodes) and the algorithm returns NULL. If the underlying network is connected based on the strong neighborhood of the nodes, then the algorithm does not return NULL and returns the SN-CDS-Node-List after all the nodes in the network are included to the Covered-Nodes-List.
Input: Static Network Graph $G = (V, E)$, where $V$ is the set of vertices and $E$ is the set of edges
TNDR and $R$ (the transmission range of the nodes in the network)

Output: SN-CDS-Node-List // contains list of nodes part of the strong neighborhood based CDS.

Begin SN-CDS Algorithm

maxUncoveredStrongNeighbors = Compute-Strong-Neighborhood($V$) // pseudo code in Figure 1
Start Node = Choose-Start-Node($V$, maxUncoveredStrongNeighbors) // pseudo code in Figure 2
Uncovered-Nodes-List = Uncovered-Nodes-List – {Start Node}
Covered-Nodes-List = Covered-Nodes-List U {Start Node}
Priority-Queue = Priority-Queue U {Start Node}

$\forall u \in V$, uncoveredStrongNeighbors($u$) = {$v | v \in SN_u$}

while (Uncovered-Nodes-List $\neq \Phi$ and Priority-Queue $\neq \Phi$) do  // the loop runs in $O(|E| + |V| \log |V|)$ time

node $s$ = Dequeue(Priority-Queue)
if (uncoveredStrongNeighbors($s$) = $\Phi$) then
  return NULL; // the underlying network is not connected
end if

SN-CDS-Node-List = SN-CDS-Node-List U {$s$}

for all node $u$ $\in$ Neighbors($s$) do
  if ($u \in$ Uncovered-Nodes-List) then
    Uncovered-Nodes-List = Uncovered-Nodes-List – {$u$}
    Covered-Nodes-List = Covered-Nodes-List U {$u$}
    Priority-Queue = Priority-Queue U {$u$}
  end if
end for

$\forall u \in V$, uncoveredStrongNeighbors($u$) = {$v | v \in SN_u$ AND $v \in$ Uncovered-Nodes-List }

Re-sort the entries in Priority-Queue in the decreasing order of the number of uncovered strong neighbors for the nodes in the queue
end while

if (Uncovered-Nodes-List $\neq \Phi$ and Priority-Queue = $\Phi$) then
  return NULL; // the underlying network is not connected
end if

return SN-CDS-Node-List

End SN-CDS Algorithm

Fig. 3. Pseudo Code for the SN-CDS Construction Algorithm

3.6 Algorithm to Construct the MaxD-CDS

The algorithm to construct the MaxD-CDS would be similar to that described in the previous sections for SN-CDS. The difference primarily lies in using the open neighborhood, determined based on the fixed transmission range ($R$) of the nodes, rather than the strong neighborhood. To begin with, the algorithm gives preference to the nodes that have the maximum number
of uncovered neighbors for inclusion to the MaxD-CDS. Any tie could be broken randomly or by choosing the node with the lowest or the largest ID among the contending nodes. In this paper, we break the tie by randomly choosing a neighbor node among the contending nodes with the maximum number of uncovered neighbors (the same way ties are broken with the SN-CDS algorithm). During subsequent iterations, the covered node that has the largest number of uncovered neighbors is chosen for inclusion to the MaxD-CDS.

3.7 Time Complexity of the MaxD-CDS and SN-CDS Algorithms

The time complexity of the MaxD-CDS and SN-CDS algorithms depends on whether we maintain the \textit{Priority-Queue} of \( |V| \) nodes as an array or a binary heap. When stored as an array, it takes \( O(|V|) \) time to extract, insert an element or re-sort the array. On the other hand, all of these operations can be done on a binary heap of \( |V| \) elements in \( O(\log |V|) \) time. Both the MaxD-CDS and SN-CDS algorithms require all the \( O(|V|) \) nodes and the \( O(|E|) \) edges in the underlying network to be explored for inclusion into the CDS. Assuming that the \textit{Priority-Queue} is implemented as a binary heap (as is done in our simulations), the overall time complexity of both the MaxD-CDS and SN-CDS algorithms is \( O(|E| + |V| \times \log |V|) \).

3.8 Example to Illustrate the Construction of MaxD-CDS and SN-CDS

Figures 4 and 5 illustrate the examples to demonstrate the working of the algorithms to determine the MaxD-CDS and SN-CDS respectively. In these figures, the nodes are represented with a circle and the integer outside the circle represents the node ID. The real-number on an edge indicates the ratio of the physical Euclidean distance between the two nodes constituting the edge to the fixed transmission range of the nodes. In Figure 4, the integer inside the circle indicates the number of uncovered neighbors (determined based on the fixed transmission range \( R \) of the nodes) of the corresponding node. In Figure 5, the integer inside the circle indicates the number of uncovered strong neighbors (determined based on a TNDR value of 0.5 when applied to the edges of the initial network in Figure 5.1) of the corresponding node. The nodes determined to be part of the MaxD-CDS and the SN-CDS are shown in bold circles. The circles of nodes that are covered (but are not part of the CDS) are shaded. The circles of nodes that are not yet covered are neither made bold nor shaded.
On a 24-node network example considered in Figures 4 and 5, we see that it takes 10 iterations to determine the MaxD-CDS; whereas, it takes 12 iterations to determine the SN-CDS. The number of iterations for both the CDS algorithms corresponds to the number of nodes constituting the CDS. The MaxD-CDS includes 10 nodes (and 10 edges) while the SN-CDS includes 12 nodes (and 11 edges). The SN-CDS incurs only slightly more nodes and edges than the MaxD-CDS, owing to a reduced number of neighbors per node due to the imposition of the strong neighborhood constraint. However, as also observed in the simulations, with only a slightly larger number of nodes and edges, the SN-CDS sustains for a significantly longer lifetime. The strength of the SN-CDS is attributed to the selection of edges (with distance ratio less than or equal to the TNDR value) that have greater chances of existence for a longer time compared to that of the MaxD-CDS.

3.9 Algorithm to Validate the Existence of a CDS

We follow the LORA (Least Overhead Routing Approach) [3,15] for CDS usage and maintenance. Accordingly, after a CDS is constructed (represented using a CDS-Node-List), we intend to use it as long as it exists. A CDS is considered to exist if the following two conditions are satisfied: (i) The nodes constituting the CDS (i.e., the nodes in the CDS-Node-List) should be connected and (ii) For every non-CDS node, there should be at least one CDS node as a neighbor node. To validate the existence of a CDS at subsequent time instants, since its construction, we adopt the following algorithm: the algorithm returns true if the CDS still exists; otherwise, it returns false.

We construct the CDS-Edge-List among the nodes in the CDS-Node-List. To do so, for any two nodes in the CDS-Node-List, we determine the physical Euclidean distance between the two nodes and if it is less than the fixed transmission range of the nodes, then we add a link between the two nodes into the CDS-Edge-List. After the CDS-Edge-List is constructed, we run a Breadth First Search (BFS) [8] on a network sub graph featuring the CDS-Node-List and the CDS-Edge-List. If BFS is able to explore all the nodes of the CDS by starting from an arbitrarily chosen node, then the CDS is considered connected (i.e., the first of the two requirements for the existence of a CDS is satisfied); otherwise the algorithm returns false. If the first requirement is satisfied, then we check for every non-CDS node whether there exists a CDS node (in the CDS-Node-List) as a neighbor node. If this
requirement is also satisfied for all non-CDS nodes, then the algorithm returns true; otherwise, it returns false. The above algorithm to validate existence of a CDS is very generic and can be used for any CDS in MANETs, including the SN-CDS and MaxD-CDS, simulated and studied in this paper.

4 Simulations

We conduct our simulations in a discrete-event simulator (developed in Java), which has been also successfully used in recent studies (e.g. [9,10,13,14]). The dimensions of the network topology are 1000m x 1000m. The fixed transmission range per node is 250m. The number of nodes in the network is varied by conducting simulations with 50, 100 and 150 nodes to represent networks of low, moderate and high density respectively, corresponding to an average neighborhood size of approximately 10, 20 and 30 nodes if the transmission range per node is 250m.

The mobility model used in the simulations is the Random Waypoint model [16], one of the most widely used models for simulating mobility in MANETs. According to this model, the nodes are initially uniform-randomly distributed throughout the network. Each node randomly chooses a destination location (within the network boundary) to move to with a velocity randomly chosen from the range \([0, \ldots, v_{\text{max}}]\). Once a node reaches the targeted location, it continues to move to a different randomly chosen destination location with a velocity again randomly chosen from the above range. Each node continues to move like this for the simulation time of 1000 seconds. The movement of a node is independent of the other nodes in the network. The values of \(v_{\text{max}}\) chosen are 5 m/s, 25 m/s and 50 m/s representing scenarios of low, moderate and high node mobility respectively.

We construct snapshots of the network topology for every 0.25 seconds, starting from time 0 to the simulation time of 1000 seconds. The underlying network graph for both the MaxD-CDS and the SN-CDS algorithms is the topology generated according to the unit disk graph model according to which there is an edge between any two nodes in the network if the physical Euclidean distance between the two nodes is less than or equal to the fixed transmission range of the nodes in the network. However, to construct the network topology for the SN-CDS algorithm, an additional constraint (corresponding to the strong neighborhood) is imposed on the unit disk graph topology and the constraint is to include an edge in the topology only if the distance ratio (ratio of the physical Euclidean distance to the transmission range of the nodes) is less than or equal to the Threshold Neighborhood Distance Ratio (TNDR). If a CDS is not known or does not exist for the network snapshot (generated either way as described above) at a particular time instant \(t\), we run the appropriate CDS algorithm on that network snapshot. The CDS determined during a particular time instant is validated for existence during the subsequent time instants (as described in Section 3.9) until the CDS ceases to exist. The above procedure is continued until the end of the simulation time.

For the SN-CDS, the TNDR values are varied from 0.5 to 1.0 (where a TNDR value of 1.0 also corresponds to MaxD-CDS). The number of strong neighbors for a node decreases with decrease in the TNDR value as the ratio of the physical Euclidean distance to that of the fixed transmission range (250m) has to be less than or equal to TNDR. Since the SN-CDS is constructed on a network graph with constituent edges included based on the strong neighborhood of the nodes, the connectivity of the network gets lower as the TNDR value is lowered (observed in Figure 6). Since a CDS exists as long as the underlying network is connected, we report the network connectivity as the CDS connectivity (Figures 6 and 7). The CDS connectivity is the ratio of the number of time instants a CDS exists for the network to that of total number of time instants considered during the simulation.
4.1 Performance Metrics

In addition to CDS connectivity, the following are the performance metrics measured in our simulations. Each data point in Figures 6 through 13 is an average computed over 10 mobility trace files generated for every combination of network density and node mobility values considered in the simulations.

- **CDS Lifetime**: We keep track of the duration of existence of each of the CDS used for the entire simulation time and compute the average value of the CDS lifetime, considering all the mobility profiles for the particular simulation condition. Refer to section 4.2 wherein we distinguish between the Absolute CDS Lifetime and the Effective CDS Lifetime.

- **CDS Node Size**: This is a time-averaged value of the number of nodes that are part of the CDS used for every time instant over the entire simulation. For example, if there exists a sequence of three CDS of size 20 nodes, 30 nodes and 40 nodes in the network for 4, 6 and 10 seconds respectively, then the average CDS Node Size for a total of 20 seconds is \((20 \times 4 + 30 \times 6 + 40 \times 10) / (4 + 6 + 10) = 33.0\) and not simply the average of 20, 30 and 40 nodes = 30.

- **CDS Edge Size**: This is a time-averaged value of the number of edges that exist between any two CDS nodes for every time instant over the entire simulation.

- **Hop Count per s-d Path**: This is a time-averaged value for the number of edges (hops) in the paths determined for every s-d pair on the CDS-induced sub graphs, considered over the entire simulation time and all the s-d pairs.

To measure the hop count per path, we run the Breadth First Search algorithm on a CDS-induced sub graph for 15 source-destination (s-d) pairs – the role of the source or destination could be assigned to any node (CDS node or non-CDS node) in the network. The CDS-induced sub graph for a particular time instant comprises of all the nodes in the network and edges that may exist between any two CDS nodes and between a CDS node and a non-CDS node. Two non-CDS nodes have to communicate through one or more CDS nodes as intermediate nodes, even if the two non-CDS nodes are neighbors of each other. However, two CDS nodes can communicate directly if they are neighbors of each other.

For better accuracy, we report only the performance results for network connectivity of 0.5 or above. Also, we observe appreciable SN-CDS connectivity (i.e., 50% connectivity) for all three network densities (50, 100 and 150 nodes) only for TNDR values of 0.8 and 0.9. For the rest of the paper, we will refer to the SN-CDS performance obtained with a TNDR value of 0.9 while comparing with that of MaxD-CDS (TNDR = 1.0). The performance metric values at these two TNDR values are displayed in Figures 7 and 9 through 13. In Figures 6 and 7, we notice that for a given node density, the percentage CDS connectivity for the SN-CDS improves significantly with increase in the maximum node velocity, and approaches to that of the MaxD-CDS. This could be attributed to be the dynamically changing topology at high node velocities. The lack of connectivity observed at a particular time instant may not continue for a long time; the nodes might move around quickly and the connectivity among them could be restored.

4.2 Absolute CDS Lifetime and Effective CDS Lifetime

We measure the Absolute CDS Lifetime as the duration of time a particular CDS actually exists and is averaged over the sequence of CDS that are used during the simulation time. Since the connectivity of the SN-CDS depends on the value of the TNDR values used, we calculate another metric called the Effective CDS Lifetime, defined as the product of the absolute CDS lifetime observed and the percentage CDS connectivity. This way, we take into consideration not only the stability of a particular CDS, when it exists, we also consider the percentage of times such CDS can be determined across the entire simulation period. For example, for a particular node density and maximum node velocity, if the connectivity values observed for an SN-CDS (at TNDR of 0.8) and a MaxD-CDS are 0.6 and 0.9 respectively and the absolute CDS lifetimes are 15 seconds and 5 seconds respectively, then the effective CDS lifetimes for the SN-CDS and MaxD-CDS are 15*0.6 = 9 seconds and 5*0.9
Such a comparison (using the effective CDS lifetime) would be fair than using the absolute CDS lifetimes for evaluating the stability of the two CDS. In Figure 8, we report the absolute CDS lifetimes observed for SN-CDS (at different TNDR) values and the MaxD-CDS; whereas, Figure 9 reports the corresponding effective CDS lifetimes, taking into consideration the CDS connectivity values reported in Figure 6. The absolute lifetime and the effective lifetime of the SN-CDS (for all values of TNDR reported) are larger than that of the MaxD-CDS. Also, as observed in Figures 8 and 9, the difference between the absolute and effective SN-CDS lifetimes is much more in networks with 50 nodes compared to networks with 100 and 150 nodes. This could be attributed to the relatively lower SN-CDS connectivity in 50-node networks, especially at TNDR values less than 0.8.

We notice from Figure 10, the effective lifetime of the SN-CDS (@ TNDR = 0.9) is always significantly larger than that of the MaxD-CDS, especially with increase in network density as well as node mobility. The effective lifetime of SN-CDS is 10% (at $v_{max} = 5$ m/s and 50 nodes) to 170% (at $v_{max} = 50$ m/s and 150 nodes) larger than that of the MaxD-CDS. The relatively high stability of the SN-CDS could be primarily attributed to the TNDR constraint and the resulting side-effect of requiring slightly more nodes (as part of the SN-CDS) to cover the rest of the nodes in the network. With the TNDR restriction, the physical Euclidean distance between the constituent end nodes of the edges included into the SN-CDS is not closer to the transmission range of the nodes; and as a result, the chances that these edges are likely to break in the near future is not high. The tradeoff is a slightly larger number of constituent nodes in the SN-CDS (i.e., the CDS Node Size) and an accompanying increase in the number of edges between the SN-CDS nodes (i.e., the CDS Edge Size). The MaxD-CDS algorithm, in pursuit of minimizing the CDS Node Size, ends up including the bare minimum number of nodes into the CDS and the edges between these CDS nodes are highly vulnerable to break in the immediate future (due to the physical Euclidean distance between the end nodes of these edges being closer to the transmission range of the nodes). We notice from the simulation results of Figures 10, 11 and 12, with at most a 22% increase in the CDS Node Size and at most a 45% increase in the CDS Edge Size, the lifetime of the SN-CDS could be increased to as large as 170% more than that of the MaxD-CDS. Thus, even though we may say that there is a tradeoff between the CDS Lifetime vs. the CDS Node Size and CDS Edge Size, the tradeoff is more favorable towards SN-CDS and it is worth including slightly fewer nodes into the SN-CDS in order to sustain a significantly longer lifetime.
4.3 CDS Node Size and CDS Edge Size

The MaxD-CDS algorithm is 100% density-based and is designed to minimize the number of constituent nodes of the CDS. On the other hand, even though the SN-CDS algorithm gives preference to include (into the CDS) nodes with a relatively larger number of uncovered neighbors, the neighborhood of a node is decided based on the TNDR. As explained in Section 4.2, the edges constituting the strong neighborhood topology on which the SN-CDS is determined are relatively more stable than the edges constituting the open neighborhood topology on which MaxD-CDS is determined. Thus, SN-CDS is more stability-oriented and minimizing the number of constituent nodes of the CDS is only a secondary objective. As a result, we do observe an increase in the CDS Node Size (refer Figure 11); but, it is not a significant increase since the algorithm is still density-based and prefers to include nodes (into the SN-CDS) that have more uncovered strong neighbors.

In the case of a MaxD-CDS, with the CDS nodes likely to be far away from each other (to cover all the nodes in the network with the minimal CDS Node Size), the number of edges between the CDS nodes is barely the minimum required to keep the CDS nodes connected. This contributes to the fragile nature of the MaxD-CDS; even if one or two edges (between the CDS nodes) break, the whole MaxD-CDS could get disconnected. On the other hand, the SN-CDS incorporates relatively more edges (attributable to the larger CDS Node Size) between the CDS nodes. However, since the SN-CDS also has a (secondary) objective of minimizing the CDS Node Size, there is not a plethora of nodes and edges constituting the CDS (refer Figure 12).

4.4 Hop Count per Path

The SN-CDS incurs a slightly average larger hop count per path (Figure 13) between any two nodes in the network. This could be attributed to the lower physical Euclidean distance of the edges constituting the SN-CDS, compared to that of a MaxD-CDS. As a result, on average, more intermediate CDS nodes are required to connect any two randomly chosen source and destination nodes in the network. As mentioned earlier, the source and destination nodes could be any two nodes in the network and need not be the CDS nodes. The hop count per path incurred with an SN-CDS is almost the same as that incurred with a MaxD-CDS in low-density networks and the difference between the hop count per path incurred with the two CDSs increase with increase in the network density. The hop count per path determined through the SN-CDS is at most 24% more than that determined using MaxD-CDS (noticed for network density of 150 nodes). Like the CDS Lifetime-CDS Node Size tradeoff observed earlier, the performance of the SN-CDS and the MaxD-CDS vis-à-vis the hop count per path could be accounted as the tradeoff between CDS Lifetime and the Hop Count per Path. In both cases, the tradeoff is more favorable towards SN-CDS.
5 Extension of the SN-CDS Algorithm for Heterogeneous Systems

The SN-CDS algorithm can be easily adapted for heterogeneous MANETs, where each node could operate at a different transmission range, most of the time chosen according to the residual (i.e., available) battery charge at the nodes, facilitated through transmission power control. If all nodes have the same TNDR value, then the number of nodes in the strong neighborhood of a node will be proportional to the transmission range of the node (i.e., nodes with a larger available battery charge will have a larger number of strong neighbor nodes) and hence nodes with longer transmission ranges will more likely end up being the CDS nodes. To honor the commitment to its covered nodes, a CDS node has to operate at a fixed transmission range (chosen at the time of construction of the CDS) until the CDS breaks. To be fair to all nodes in the network and extend their lifetime, each time a CDS is constructed, the network should facilitate transmission power control so that a node can dynamically adapt its transmission range based on the available battery charge at the node and operate at the chosen range until the current CDS breaks.

The only open question would be to decide on the nature of the TNDR values assigned to the nodes – there are three options: (i) assigning the same TNDR value for all nodes; (ii) assigning larger TNDR values for nodes with longer transmission range and vice-versa and (iii) assigning smaller TNDR values for nodes with longer transmission range and vice-versa. From the point of view of enhancing the fairness of node usage and maximizing the time of first node failure due to exhaustion of battery charge, the first strategy of assigning the same TNDR value for all nodes in the network appears to be more logical and practical than assigning different TNDR values for the nodes. Also, with the first option of operating all nodes using the same TNDR value, the number of nodes in the strong neighborhood of a node will be proportional to the transmission range of the node (i.e., nodes with longer transmission range will have a larger number of strong neighbor nodes) and hence nodes with relatively larger available battery charge will more likely end up being the CDS nodes. Under the second and third options, the TNDR value assigned to a node could be directly or inversely proportional to the transmission range of the node; either way, from the point of view of maximizing both node lifetime and coverage, it does not appear to be a favorable strategy for nodes with limited battery charge. As part of future work, we will investigate the required modifications to the SN-CDS and other CDS algorithms for use in heterogeneous MANETs and study their performance.

6 Conclusions

In this paper, we have proposed an algorithm to determine long-living stable CDS without incurring a significant increase in the CDS Node Size. The proposed algorithm is based on the notion of “Strong Neighborhood” and has been appropriately referred to as the SN-CDS algorithm; the algorithm has the same run-time complexity as the standard maximum density based MaxD-CDS algorithm. We observe the MaxD-CDS to be quite unstable (i.e., incur a lower CDS Lifetime) in the presence of node mobility. The CDS Lifetime vs. CDS Node Size tradeoff and the CDS Lifetime vs. Hop Count per Path tradeoff are both favorable towards the SN-CDS. The SN-CDS lifetime can be as large as 170% more than MaxD-CDS lifetime; with the increase (for the SN-CDS) in the CDS Node Size, CDS Edge Size and the Hop Count per path being at most 22%, 45% and 24% respectively. The tradeoff increases with increase in network density. At the same time, the relatively larger number of constituent nodes and edges (with increase in network density) makes the SN-CDS to be more robust to node mobility and link failures. Since, the hop count per path is not excessively high and there are more nodes that are part of the SN-CDS, we anticipate routing through the SN-CDS would better balance the forwarding load among the CDS nodes (compared to routing through the MaxD-CDS); thus, improving the fairness of node usage. On the other hand, since the MaxD-CDS induced sub graph has fewer nodes, the nodes that are part of the MaxD-CDS can be significantly over-working compared to the rest of the nodes in the network and this can lead to premature node failure of the nodes being part of the MaxD-CDS.
Acknowledgments

The work leading to this paper was partly funded through the U.S. National Science Foundation (NSF) grants DUE-0941959 and CNS-0851646. The views and conclusions contained in this paper are those of the authors and do not represent the official policies, either expressed or implied, of the funding agency.

References