A Contention Aware Connected Dominating Set Construction Algorithm for Wireless Ad-Hoc Networks

Chowdhury Nawrin Ferdous
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh
Email: bipashachowdhury03@gmail.com

Ashikur Rahman
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology
Dhaka, Bangladesh
Email: ashikur@cse.buet.ac.bd

***Abstract*—The nodes in wireless ad-hoc networks frequently need to broadcast messages for route discovery and other services. The naive flooding requires every node to forward a message exactly once and causes serious performance bottleneck due to redundant traffic, contention, and collision. One convenient solution to overcome this problem is to construct a connected dominating set (CDS) as virtual backbone. Several centralized and distributed algorithms have been designed to compute CDS in order to reduce the number of rebroadcasts. However, none of the approaches aim at minimizing contentions. Contention is a phenomenon that occurs when a group of nodes want to transmit over a shared channel. During contention only one node gets access to the channel and the others defer their transmission for a later time. In this paper, a new heuristic dubbed as *Contention Aware Connected Dominating Set (CACDS)* has been proposed to construct a CDS that intelligently selects the nodes to reduce contention. A centralized algorithm has been deduced first to construct CDS using the global topology information. Since it is difficult to gather the entire topology information, a distributed counterpart has also been presented where a node selects a subset of nodes from its immediate neighbors as forwarding nodes based on 2-hop neighborhood information and reduces contention. Experimental results show that proposed heuristics significantly reduce contention and outperform other state-of-the-art algorithms in minimizing contention with a little increase in number of forwarding nodes.**

***Keywords*—Contention, Wireless Ad-hoc Network, Connected Dominating Set.**

I. Introduction

A wireless ad-hoc network is a collection of several mobile nodes that dynamically form a network for communication without any pre-existing infrastructure or any centralized control. This network is popularly used in emergency search-and-rescue operations, decision making in the battlefield etc. where network needs to be deployed immediately but the base stations or fixed network or infrastructures are not available. The mobile nodes often cannot communicate with each other directly because of several causes such as limitation of power, improper utilization of channel and many more. So, they rely on several intermediate nodes (known as relays) to exchange data with one another over the network and thus the wireless multi-hop network is formed. The nodes in these networks frequently need to broadcast messages for route discovery, periodic data dissemination, erasing an invalid route, locating a node, or even for sending alarm signals in the entire network.

An effortless approach to perform broadcast is *blind flooding*. Every node forwards a broadcast message exactly once in blind flooding. Though blind flooding ensures full coverage at high mobility but unfortunately, it results in redundant traffic, contention, and collision and finally leads to *broadcast storm problem* [1]. One solution to overcome these problems is to compute a virtual backbone based on the physical topology, and run any existing routing protocol over the virtual backbone [2].

The Connected Dominating Set (CDS) is a widely used approach in this context. A CDS is generally constructed by at first modeling the entire network as a graph G where all nodes form a node set V and the communication links between nodes form an edge set E. Then, a CDS of a graph G contains a subset D of the node set V where any node in D can reach any other node in D by a path that stays entirely within D. That is, D induces a connected sub-graph of G and every node in G either belongs to D or is adjacent to a node in D. Only the nodes belonging to CDS participate in forwarding to convey the message in the entire network.

A notable number of centralized and distributed algorithms [2]–[4] have been devised to create CDS to reduce the number of packet forwarding. Although CDS based algorithms have been proposed to reduce redundancy, none of these works aim at minimizing contention. Contention means *competition for resources*. The term is used especially in networks to describe the situation where two or more nodes attempt to transmit a message across the same medium at the same time. In a mobile ad-hoc network, after broadcasting a message by a mobile host, if more than one neighbors which are within their transmission range want to rebroadcast it, these transmissions may face serious contention with each other.

The main contribution in this paper is to construct a connected dominating set in such a way so that the contention among the nodes in CDS is minimized. A couple of new

heuristics have been proposed to construct the contention aware connected dominating set (CACDS) in this work.

A centralized algorithm has been deduced first to construct CDS using the global topology information of a wireless network. Since it is difficult to gather the entire topology information, a distributed version has also been presented where a node selects a subset of nodes from its immediate neighbors as forwarding nodes based on 2-hop neighborhood information. A demonstration of the efficiency of the distributed algorithm using the centralized algorithm as a benchmark is also provided in this work. Finally, a comprehensive simulation to analyze the behavior of the proposed algorithms and compare their performances with other state-of-the-art algorithms in terms of number of forwarding nodes and amount of contention has been presented. In centralized environment, with the increase of 0-5% forwarding nodes, new heuristic generates almost 90-100% contention free CDS and it increases the number of forwarding nodes by 1-5% to mitigate almost 4-19% contention in CDS in a distributed environment.

The rest of the paper is organized as follows. In Section 2, the basic idea of this work has been discussed. In Section 3, we present the state-of-the-art research works. Section 4 provides the important terminologies and considered state-of-art algorithms. Section 5 deals with the methodology of the new proposed algorithms with suitable examples. Section 6 demonstrates the simulation and performance evaluation and finally Section 7 concludes the paper highlighting the contribution, limitations and possible future works.

II. Basic Idea

Formally, a CDS of a graph G is a set D of nodes with the following two properties:

(i) Any node in D can reach any other node in D using only the nodes that belong to D.

(ii) Every node in G either belongs to D or is adjacent to a node in D. That is why D is called a dominating set of G.

Once the CDS is constructed broadcasting task is performed as follows:- if the source is in CDS then only the nodes in CDS need to forward the packet and the packet reaches all the nodes in the network because the nodes that are not in CDS must be neighbors of some node(s) in CDS. If the source is not in CDS it can forward the packet to its neighbors and at least one of its neighbors is guaranteed to be in CDS. Thus the packet reaches one of the CDS nodes and all other nodes in CDS forward the packet like before.

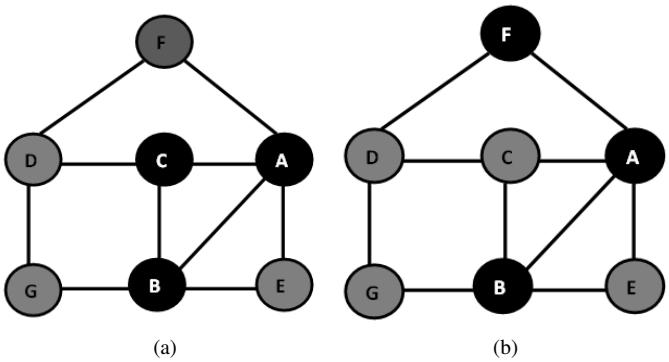
The main goal of this work is to construct a CDS that will intelligently select the nodes to reduce contention. Upon receiving a packet from one node if two or more neighboring nodes try to rebroadcast the packet at the same time, there will occur contention. For example, consider the sample network of 7 nodes illustrated in Figure 1.

Here one can easily see the following CDSs:

$$CDS = \{A,B,C\},\{A,B,F\},\{C,D,B\},\{A,B,G\},\{A,B,C,D\}$$

Among the above CDSs minimum size CDS are as follows:

$$MCDS = \{A,B,C\},\{A,B,F\},\{C,D,B\},\{A,B,G\}$$



(a) (b)

Fig. 1: (a) CDS Construction with Contention, (b) CDS Construction without Contention

However the contention aware CDSs are only the followings:

$$CACDS = \{A,B,F\},\{C,D,B\},\{A,B,G\}$$

In Fig 1(a), if the CDS is constructed in traditional way, node A, B and C can be selected which will lead to contention problem. Because, upon receiving message from node A, if two of its neighbors node B and node C start to *rebroadcast* it, they have to contend with each other first for gaining access to the shared channel as they are within the transmission range of each other. So, the selection of the nodes should be done intelligently to minimize this type of contentions. Figure 1(b) presents a scenario, where node F is selected instead of node C to avoid the contention. The new heuristic designed for minimizing contention will never select A,B,C in order to avoid contention.

III. Related Works

Researchers have proposed several approaches to reduce the *broadcast storm problem*. A conventional solution to mitigate this problem is to construct a virtual backbone as the basis of routing and broadcasting. The idea to create a virtual backbone using connected dominating set (CDS) is first proposed by Ephermides in [5]. Since then, various methods on the CDS construction have been found in the literature which can be classified as centralized algorithms and distributed algorithms based on the network information they used.

Multicasting to all nodes in an ad-hoc network is equivalent to broadcast. The problem of constructing optimal broadcast tree that minimizes the number of packet forwarding is very much similar to MCDS problem [6]. MCDS problem cannot be solved in polynomial time, so the optimal broadcast tree construction based on MCDS is proved to be an NP-complete problem. So, researchers have proposed several approximation algorithms and heuristics to find the optimal broadcast tree using the concept of MCDS.

Guha and Khullar first propose two greedy heuristic algorithms in [7], to construct CDS based on Minimum Connected Dominating Set (MCDS) and Weakly connected dominating set (WCDS). In [8], authors propose a greedy algorithm for MCDS

in unit-disk graphs based on MIS (Maximal Independent Set). Min et al. propose to use a Steiner tree with minimum number of Steiner nodes (ST-MSN) in [9].

Due to the lack of global topology information, the distributed approach is most widely used for CDS construction in wireless multi-hop network. Das and Bharghavan in [10] provide the distributed implementation of the two centralized algorithms given by Guha and Khuller in [7]. Both implementations suffer from high message complexities. The one given by Wu and Li in [11] has no performance analysis. It needs at least two-hop neighborhood information. Lim and Kim [12] propose a reactive algorithm called Self Pruning (SP) that uses direct neighborhood information to decide whether to forward a packet or not and another proactive algorithm named Dominant Pruning (DP) where extended neighborhood information is used. In DP, a node construct its own forwarding list from the subset of its 1-hop neighbors in order to cover all its 2-hop neighbors. They also propose two extensions of DP known as Partial Dominant Pruning (PDP) and Total Dominant Pruning (TDP).

Though there are several methods to construct CDS to reduce redundancy but CDS has never been computed to reduce contention in any of the prior works. Therefore, we are proposing to fill up this notable gap by introducing contention aware minimum connected dominating set in this paper.

IV. Preliminaries

We use a simple graph $G(V,E)$ to represent an ad-hoc network, where V represents the set of wireless mobile nodes and E represents the set of edges. An edge (u,v) indicates that both hosts u and v are within their transmission range. $N(u)$ is defined as a set of adjacent nodes of node u and $N(N(u))$ is the set of nodes that is at most 2-hop away from node u. F_{u} represents the list of 1-hop neighbors of u that are selected for forwarding by node u. B_{u} is the set of 1-hop neighbors of node u that are eligible to be included in the forwarding list F_{u}. U_{u} is the set of nodes that need to be covered by using nodes from B_{u} while u creates its forwarding list F_{u} .

MCDS [7] and Dominant Pruning [12] has been used as state-of-art algorithms to evaluate the new proposed approaches.

(i) MCDS Construction Algorithm: At the start of the algorithm, all nodes in the network are colored white. The node with maximum cardinality is then selected and colored black. All the one-hop neighbors of that node are colored gray. A gray node having maximum number of white neighbors is then selected and colored black. The selection process recursively runs until no white node exits. The set with all the black nodes are the resultant nodes that makes MCDS. In figure 1(a),

$$MCDS = \{A,B,C\}$$

.

(ii) Dominant Pruning: The forwarding list creation process of dominant pruning algorithm is as follows:- suppose, a node v receives a packet from node u. The sender node u also sends a *forwarding list (F_u)* with the packet header. If $v \in F_{\mathrm{u}}$, then the node v will rebroadcast and will create its own forward list (F_{v}). The node then start constructing U_{v} which is all uncovered two-hop neighbors of v. The set B_{v} represents those neighbors of v which are possible candidates for inclusion in F_{v}. Then, in each iteration, v selects a neighbor $w \in B_{\mathrm{v}}$, such that $w \notin F_{\mathrm{v}}$ and the list of neighbors of w covers the maximum number of nodes in U_{v} , i.e $|N(w) \cap U_{\mathrm{v}}|$ is maximized. Next v includes w in F_{v} and sets $U_{\mathrm{v}} = U_{\mathrm{v}} - N(w)$. The iterations continue for as long as U_{v} becomes empty or no more progress can be accomplished.

V. Contention aware Connected Dominating Set (CACDS) Construction Algorithm

A brief description of the proposed algorithms is stated in this section.

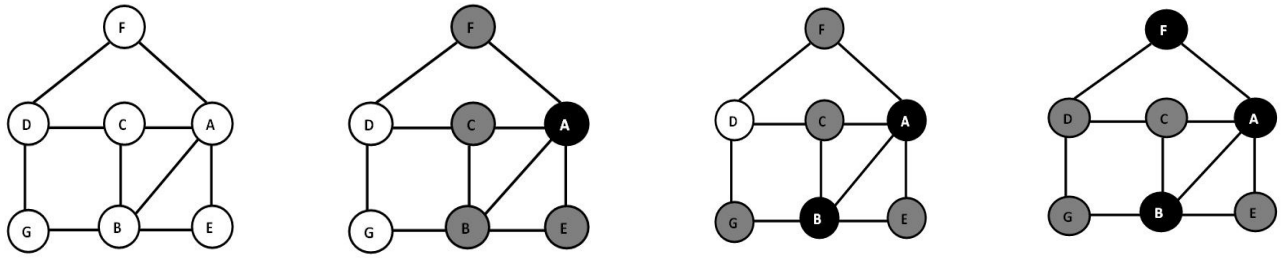
A. Centralized Contention Aware Connected Dominating Set

The centralized algorithm is presented in Algorithm 1. At the start of the algorithm, all the nodes are colored white and placed in *ColorW* set. That means, *ColorW* set consists of all nodes $v \in V$. The node with the maximum cardinality is then selected from *ColorW* set and colored as black and placed in a different *ColorB* set (shown in line 5-6 of Algorithm 1). In case of tie, a node is selected randomly among the ones with maximal cardinality. All the neighbors of that black node are colored as gray and put the in *ColorG* set. Among the gray nodes of *ColorG* set, the nodes that have minimum number of black neighbors are selected and placed in a set called *Candidate_Set* (shown in line 17-22 of Algorithm 1). The black nodes are already in CDS. Therefore, by selecting a gray node with minimum black neighbors reduces the chance of contention. In case of tie, the gray node which has maximum number of white neighbors is selected and colored as black and all its white neighbors are colored as gray (shown in line 23-29 of Algorithm 1) . A recursive selection process runs till there is no white node left in the network.

The entire selection process is shown using the following example. Consider Figure 2, among all the nodes node A has the maximum cardinality (which is 4 here), so, according to the algorithm, node A is colored black and all its neighbors (node B,C,E and F) are colored gray (illustrated in Figure 2(b)). In the next step, all the gray nodes have only 1 black neighbor that is node A and each of the gray nodes B, C and F has 1 white neighbor and node E has none. So we can select either of Node B, C , F. Suppose, node B is selected to cover node G and it is colored as black and node G is colored as gray (illustrated in Figure 2(c)). The only white node remaining in the network at this stage is node D. The gray node C has now two black neighbors (node A and node B), gray node G and node F each has one black neighbor, so either of node G and node F can be selected. Suppose node F is selected to cover node D (In Figure 2(d)). So, the final Contention aware Connected Dominating Set consists of node A,B and F. So,

$$CACDS = \{A,B,F\}$$

.



(a): All nodes are colored WHITE

(b): Node A is selected and colored BLACK

(c): Node B is selected and the process continues

(d): Finally node A,B,F constructs the CACDS

Fig. 2: Step by step construction of Centralized Contention aware Connected Dominating Set (Centralized CACDS).

Algorithm 1 Centralized CACDS

1: INPUT: G(V,E)
2: RESULT {CACDS}
3: $ColorB = \emptyset$, $ColorG = \emptyset$, $CACDS = \emptyset$,
4: $ColorW$ = all nodes $v \in V$;
5: Select a node v ∈ V with max(degree(v));
6: $ColorB = \{v\}$;
7: $ColorG = N(v) - \{v\}$;
8: $ColorW = ColorW - N(v)$;
9: **while** $ColorW \neq \emptyset$ **do**
10: $MaxWhite = 0$, $MinBlack = \|V\|$, $Candidate\_Set = \emptyset$;
11: **for all** node $u \in ColorG$ **do**
12: $BlackCount = \|N(u) \cap ColorB\|$;
13: **if** $BlackCount < MinBlack$ **then**
14: $MinBlack = BlackCount$;
15: **end if**
16: **end for**
17: **for all** node $u \in ColorG$ **do**
18: $BlackCount = \|N(u) \cap ColorB\|$;
19: **if** $BlackCount == MinBlack$ **then**
20: $Candidate\_Set = Candidate\_Set \cup \{u\}$;
21: **end if**
22: **end for**
23: **for all** node $w \in Candidate\_Set$ **do**
24: $WhiteCount = \|N(w) \cap ColorW\|$;
25: **if** $WhiteCount > MaxWhite$ **then**
26: $MaxWhite = WhiteCount$;
27: $selectedNode = w$;
28: **end if**
29: **end for**
30: **if** $MaxWhite > 0$ **then**
31: $ColorB = ColorB \cup \{selectedNode\}$;
32: $ColorW = ColorW - N(selectedNode)$;
33: $ColorG = ColorG \cup (N(selectedNode) - ColorB)$;
34: **else**
35: $ColorG = ColorG - Candidate\_Set$;
36: **end if**
37: **end while**
38: $CACDS = ColorB$;

Complexity Analysis of Centralized CACDS: In the algorithm 1, the iteration of the *WhileLoop* (line 9-37 of Algorithm 1) continues until *ColorW* set does not become empty. At first set *ColorW* consists of all nodes in the network. So, the loop body will run in $O(V)$ times where V is the total number of nodes in the network. Inside the *WhileLoop*, there are loops that are used to find the nodes which have minimum number of black neighbors and maximum number of white neighbors each of which individually runs in $O(V)$ times in the worst case scenario. So, the run time complexity of the Centralized CACDS is $O(V^2)$.

Theoretical Correctness of CACDS: Next, we show the theoretical correctness of CACDS in the following two lemmas.

Lemma 1: CACDS is a connected dominating set.

Proof: The proof is by contradiction. Suppose the vertex set of CACDS is,

$$V_{CACDS} = \{v_1, v_2, v_3 ... v_m\}$$

Assume node $v_i \in V_{CACDS}$ cannot communicate with other nodes in *CACDS*. According to the algorithm, the black nodes are selected among the gray nodes. When a node becomes gray, that means it has at least one black neighbor in the network. As v_i is a member of *CACDS*, the node must be black. When v_i was selected, it was among one of the gray nodes and each gray node is connected to one of the black nodes. Therefore, each black node has at least one path to communicate with other nodes in *CACDS*. The result contradicts with the hypothetic premises. So, v_i must have a path to connect other nodes in the *CACDS*. Hence each node in *CACDS* must be connected.

Lemma 2: The connected dominating set constructed by *CACDS* covers all the nodes in the network.

Proof: Assume that, U is the set consisting of all the nodes in the network, $U = \{x_1, x_2, ... x_n\}$, n is the number of nodes in the network, and vertex set of CACDS is $V_{CACDS} = \{v_1, v_2, v_3 ... v_m\}$, where m is the total number of nodes in the *CACDS*. $N(v_i)$ is the set of nodes which become gray after selecting v_i as a member of *CACDS* i.e., $N(v_i)$ represents the set consisting of all adjacent nodes of v_i.

$$N = N(v_1) \cup N(v_2) \cup N(v_3) \cup ... N(v_m) \quad (1)$$

In order to prove, *CACDS* covers all the nodes in the network, we have to prove that $U = N$. As *CACDS* is a connected dominating set, so every node $x_{\mathrm{i}} \in U$ either same as v_{j} or is adjacent to v_{j} for some j. In other words the following proposition must hold,

$$\forall i \Big[\exists j \big[x_i \in N(v_j) \big] \Big]$$

Thus,

$$\{x_1, x_2, ...x_{\mathrm{n}}\} \subseteq N(v_1) \cup N(v_2) \cup N(v_3) \cup ...N(v_{\mathrm{m}}) \quad (2)$$

From Equation 2,

$$\{x_1, x_2, ...x_{\mathrm{n}}\} \subseteq N \quad (3)$$

By definition,

$$\{x_1, x_2, ...x_{\mathrm{n}}\} \subseteq U \quad (4)$$

From Equation 3 and 4, by the axiom of extensionality, we can say that,

$$\forall x_i (x_i \in U \iff x_i \in N) \implies U = N \quad (5)$$

B. Distributed Contention aware Connected Dominating Set (Distributed CACDS)

For distributed CACDS, it is assumed that, each node knows its 2-hop neighborhood information. After receiving a packet from node u, if node v finds itself in node u's *forward list*, it starts to make its own *forward list* (F_{v}) from a subset of its one-hop neighbors (B_{v}) to cover all of its uncovered two-hop neighbors (U_{v}). The sets U_v and B_v is calculated using the following equations:

$$U_{\mathrm{v}} = N(N(v)) - N(v) - N(u)$$

$$B_{\mathrm{v}} = N(v) - N(u)$$

The new approach is stated in Algorithm 2. In the algorithm, an array named *Black_Count* is used so that each node $p \in B_{\mathrm{v}}$ can keep track of its adjacent nodes in B_{v} that are already selected in the *forwarding list*. At the start of constructing F_{v}, the corresponding value in the *Black_Count* array of all the nodes $p \in B_{\mathrm{v}}$ is set to one. The nodes whose corresponding value of the *Black_Count* is minimum among all the nodes in B_{v} are placed into a *Candidate_Set* (shown in line 13-17 of Algorithm 2). Then in each iteration, a node $x \in Candidate\_Set$ is selected which covers maximum number of nodes in U_{v}, i.e., $\|N(x) \cap U_{\mathrm{v}}\|$ is maximized (shown in line 18-30 of Algorithm 2). Next, node v includes node x in its forwarding list F_{v} and the corresponding values of the *Black_Count* of the nodes in B_{v} adjacent to x are all increased by one (shown in line 32-36 of Algorithm 2). Node v then updates $U_{\mathrm{v}} = U_{\mathrm{v}} - N(x)$ and $B_{\mathrm{v}} = B_{\mathrm{v}} - x$. This process is iterated until all nodes in U_v is covered or no change in B_v set is possible.

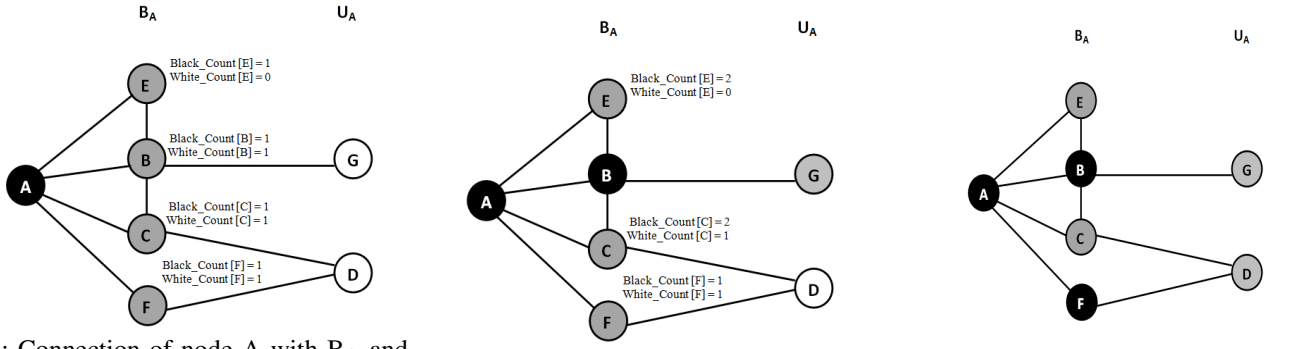
Let us explain the process using an example scenario. Consider Figure 2. Suppose node A is the broadcast initiator. Figure 3 is the redrawing of Figure 2 for better visualization in order to determine the *forward_list* of node A. B_{A} consists of node B, C, E and F and U_{A} consists of node D and G. In DP, node A selects node B and C for forwarding the message to cover node G and D. This selection arises contention between node B and node C when they start to rebroadcast.

The proposed algorithm aims at minimizing this type of contention. The algorithm does not select node C to cover node D if node A decides to choose node B to cover node G. It selects node F instead to cover node D to avoid contention. So, node B and F will be in the forward list of node A. Hence, $F_{\mathrm{A}} = \{B, F\}$. The whole process is shown in Table I.

Algorithm 2 Creation of forward_list of a node v

```
1:  Forwarding_node ← v
2:  F_v = ∅, size_of_forward_list = 0,
3:  for all node p ∈ B_v do
4:      Black_Count[p] = 1;
5:  end for
6:  while U_v ≠ ∅ or B_v remains unchanged do
7:      maximum = 0, minimum = ||V||, Candidate_Set = ∅;
8:      for all node q ∈ B_v do
9:          if Black_Count[q] < minimum then
10:             minimum = Black_Count[q];
11:         end if
12:     end for
13:     for all node r ∈ B_v do
14:         if Black_Count[r] == minimum then
15:             Candidate_Set = Candidate_Set ∪ {r};
16:         end if
17:     end for
18:     for all node s ∈ Candidate_Set do
19:         for all node t ∈ U_v do
20:             if node t ∈ N(s) then
21:                 White_Count[s] = White_Count[s] + 1;
22:             end if
23:         end for
24:     end for
25:     for all node i ∈ Candidate_Set do
26:         if White_Count[i] > maximum then
27:             maximum = White_Count[i];
28:             x = i;
29:         end if
30:     end for
31:     if maximum > 0 then
32:         F_v[size_of_forward_list] = {x};
33:         size_of_forward_list = size_of_forward_list + 1;
34:         for all node y ∈ (B_v ∩ N(x)) do
35:             Black_Count[y] = Black_Count[y] + 1;
36:         end for
37:         U_v = U_v − N(x);
38:         B_v = B_v − {x};
39:     end if
40: end while
```

Complexity Analysis of Distributed CACDS: Constructing the $forwarding\_list$ of a node v is very much similar to set cover problem. Suppose, the maximum degree of a node is Δ.



(a): Connection of node A with B_A and U_A

(b): Node B is selected for F_A

(c): Node F finally selected for F_A

Fig. 3: Step by step construction of Forward_list of node A

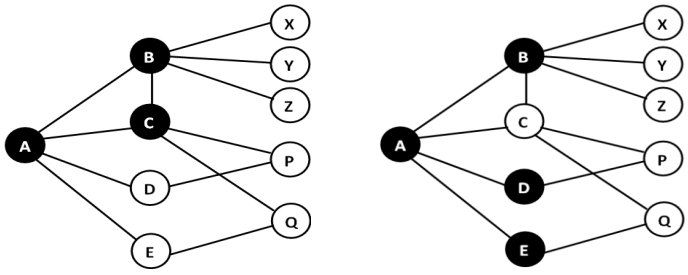
TABLE I: Forwarding lists creation process in Distributed CACDS algorithm for the scenario of Figure 2(a) (node A is the source of broadcast)

Previous node (u)	**Current node (v)**	U_v	B_v	F_v
$\emptyset$	A	{B,C,E,F}	{G,D}	{B,F}
A	B	{G}	{D}	{G}
A	F	{D}	{G}	{D}
B	G	{D}	{F}	{D}
F	D	{C,G}	{B}	{C}
D	C	{A,B}	{E}	{A}

In the worst case scenario the size of B_v set of a node v can be equal to Δ, i.e. $|B_v| = \Delta$. Forwarding nodes should be chosen from the set B_v to cover the nodes that belong to set U_v. The maximum number of nodes in U_v set that can be covered by B_v is Δ^2. So, final run time of constructing the *forwarding_list* of node v is $O(\Delta^3)$.

VI. Experimental Results

A. Simulation Environment



(a) (b)

Fig. 4: (a) CDS to minimize redundancy, (b) CACDS to minimize contention

To evaluate the performance, randomly deployed networks of 100-500 nodes over a fixed 625m X 625m square region has been simulated. The transmission radius is limited between 125m to 225m. For each scenario 10 different networks are generated and the mean value of the number of forwarding nodes and the amount of contention are taken to evaluate the effectiveness of the proposed algorithms. The simulation code-base is built using C++ programming language.

B. Performance Metrics

While evaluating the algorithms, the number of forwarding nodes and the number of contention have been chosen as performance metrics.

(i) Number of forwarding nodes: The total number of forwarding nodes needed to complete a broadcast is same as the size of a CDS. Thus we calculate the size of the constructed CDS to determine number of forwarding. In order to minimize contention, the nodes that can reach the maximum number of nodes in the network may to get selected. Therefore, the number of forwarding nodes might increase in order to mitigate contention. Such a scenario is depicted in Figure 4. The number of forwarding nodes is 3 in traditional MCDS algorithm. The number of forwarding nodes increases to 4 in order to minimize contention.

(ii) Number of Contention: The total number of contention to complete a broadcast is determined as follows. At first we define "per hop contention" which is the number of nodes in the forwarding list of any node v that are neighbors of each other. If one of the members of the F_v is connected (within the transmission range) to another ones, then those nodes face contentions when they try to forward the packet. Thus per hop contention ($\mathcal{P}$) is mathematically determined as:

$$\mathcal{P}(v) = \sum_{w \in F_v} \Big(\big| (N(w) - \{w\}) \cap F_v \big| \Big) / 2$$

Suppose, to complete a broadcast the nodes in $\mathcal{S} = \{v_1, v_2, v_3, ..., v_{cacds}\}$ forward the packet, i.e., $\mathcal{S}$ is the CDS. Let $\mathcal{F}$ represents the set of all the forwarding list that are created at each hop.

$$\mathcal{F} = \{F_{v_1}, F_{v_2}, F_{v_3}, ..., F_{v_{cacds}}\}$$

When we sum up per hop contention of all the forwarding nodes in the network, we get the total number of contention $\mathcal{N}$ that

would occur for a broadcast which is mathematically defined as:

$$\mathcal{N} = \sum_{v_i \in \mathcal{S}} \mathcal{P}(v_i) = \sum_{v_i \in \mathcal{S}} \sum_{w \in \mathcal{F}_{v_i}} \Big(\big| (N(w) - \{w\}) \cap \mathcal{F}_{v_i} \big| \Big) / 2$$

C. Experimental Results regarding Number of Forwarding

For better readability of the results, we have separated the graphs of centralized (MCDS and Centralized CACDS) and distributed (DP and distributed CACDS) algorithms. In Figure 5 and 6, the effect of centralized and distributed algorithms is presented. The number of node has been set to 100. The size of Centralized CACDS and MCDS is almost same for this scenario. In distributed environment, proposed algorithm needs 1-5% additional transmissions which is not very high.

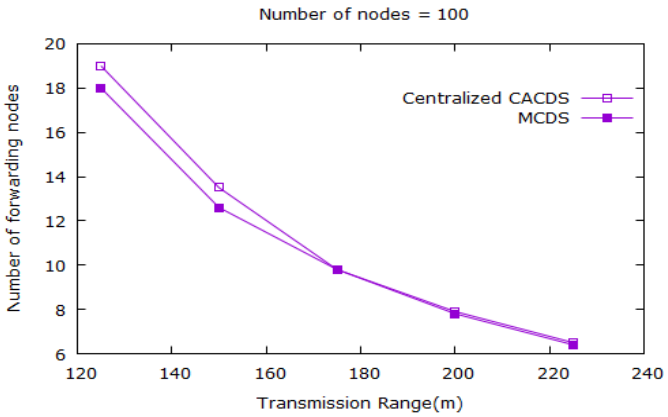


Fig. 5: Performance of Centralized Algorithms for number of required transmissions of networks having 100 nodes by varying transmission range

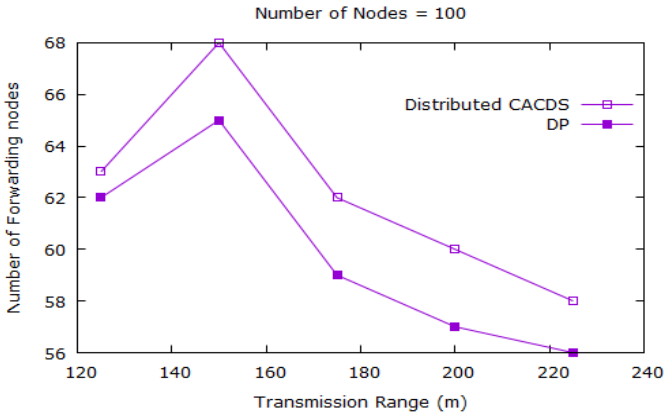


Fig. 6: Performance of Distributed Algorithms for number of required transmissions of networks having 100 nodes by varying transmission range

Figure 7 and 8 shows the required transmissions by adding nodes in a fixed transmission range (225m). Here also the difference in number of forwarding nodes is very small for both centralized and distributed cases.

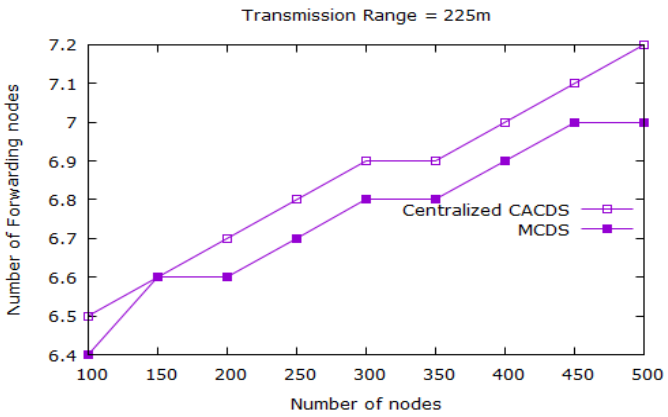


Fig. 7: Performance of Centralized Algorithms for number of required transmissions by varying number of nodes

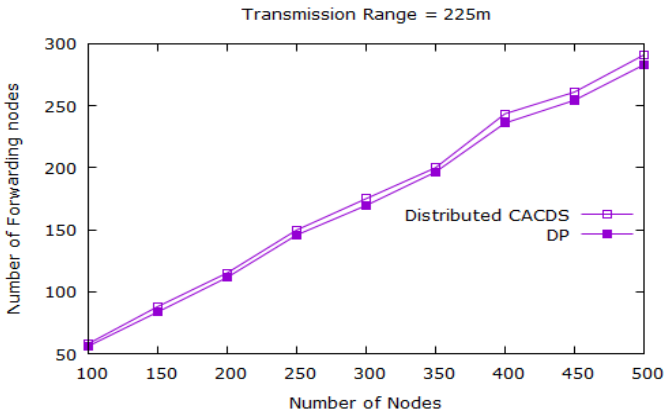


Fig. 8: Performance of Distributed Algorithms for number of required transmissions by varying number of nodes

D. Experimental Results regarding the Number of Contention

Figure 9 and Figure 10 present the state of contention occurrence for a sparse network with 100 nodes for both the centralized and distributed algorithms respectively. Figure 9 shows that the number of contention varies from 0.9 to 3.3 (in average) between the forwarding nodes in MCDS algorithm whereas it is minimized to 0-0.3 (in average) for Centralized CACDS. It is noticeable that for transmission range 225m, new algorithm generates totally contention free CDS. The distributed scenario is presented in Figure 10. In distributed environment also, proposed method performs better than that of DP. It avoids almost 4-9% contention. In the scenario shown in Figure 11 and Figure 12 we vary number of nodes by keeping the transmission range set to 225m. Needless to say that, the new approaches also perform well.

VII. Conclusion and Future Work

In this paper, a new approach has been introduced to construct CDS for minimizing contention. Both centralized and distributed version of the algorithm have been designed and a

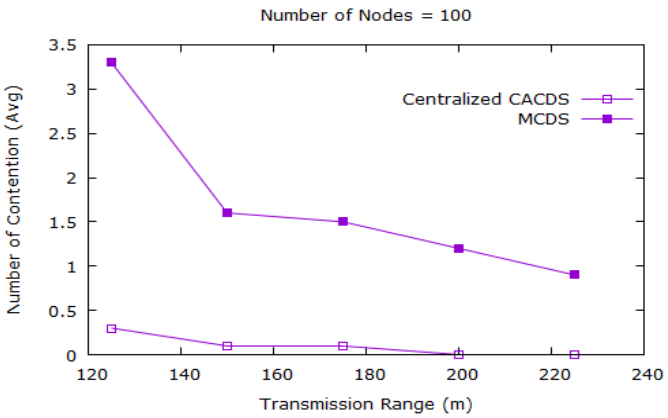


Fig. 9: Performance comparison of Centralized Algorithms in term of contention varying transmission range

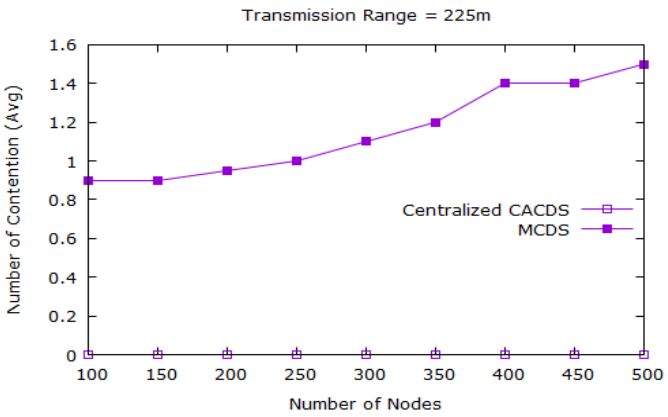


Fig. 11: Performance comparison of Centralized Algorithms in term of contention varying number of nodes

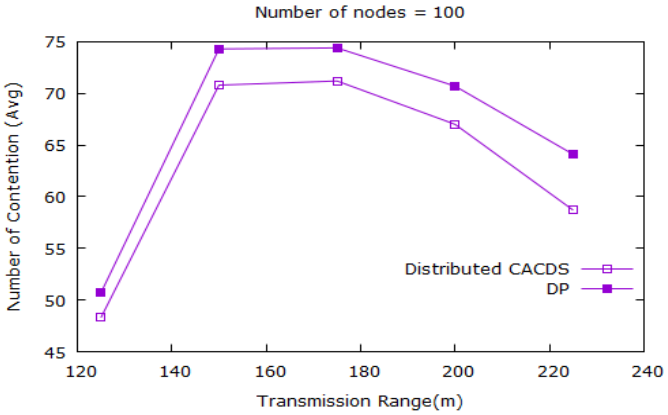


Fig. 10: Performance comparison of Distributed Algorithms in term of contention varying transmission range

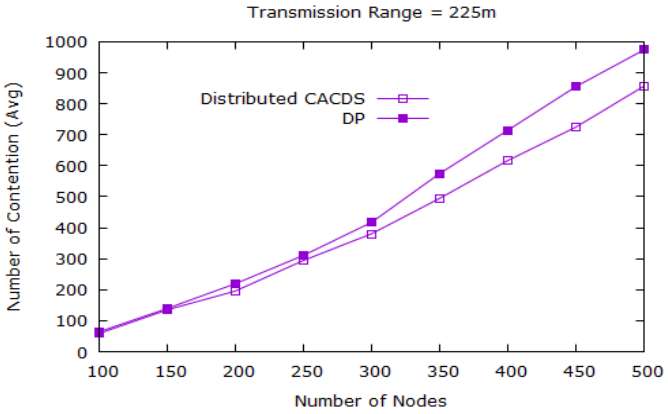


Fig. 12: Performance comparison of Distributed Algorithms in term of contention varying number of nodes

comprehensive simulation has been presented to analyze the behavior of the proposed algorithms. The simulation result shows that, the proposed algorithm works better in terms of minimizing the contention with a slight increase in the number of forwarding. The distributed version experience more contentions than centralized version. We are currently experimenting to minimize the performance gap between the two versions.

REFERENCES

[1] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless networks*, vol. 8, no. 2, pp. 153–167, 2002.

[2] S. Butenko, X. Cheng, C. A. Oliveira, and P. M. Pardalos, "A new heuristic for the minimum connected dominating set problem on ad hoc wireless networks," *Recent developments in cooperative control and optimization*, vol. 3, pp. 61–73, 2004.

[3] J. P. Mohanty, C. Mandal, and C. Reade, "Distributed construction of minimum connected dominating set in wireless sensor network using two-hop information," *Computer Networks*, vol. 123, pp. 137–152, 2017.

[4] J. P. Mohanty, C. Mandal, C. Reade, and A. Das, "Construction of minimum connected dominating set in wireless sensor networks using pseudo dominating set," *Ad Hoc Networks*, vol. 42, pp. 61–73, 2016.

[5] A. Ephremides, J. E. Wieselthier, and D. J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," *Proceedings of the IEEE*, vol. 75, no. 1, pp. 56–73, 1987.

[6] D. Lichtenstein, "Planar formulae and their uses," *SIAM journal on computing*, vol. 11, no. 2, pp. 329–343, 1982.

[7] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20, no. 4, pp. 374–387, 1998.

[8] X. Cheng, M. Ding, and D. Chen, "An approximation algorithm for connected dominating set in ad hoc networks," in *Proc. of International Workshop on Theoretical Aspects of Wireless Ad Hoc, Sensor, and Peer-to-Peer Networks (TAWN)*, vol. 2, no. 4, 2004.

[9] M. Min, H. Du, X. Jia, C. X. Huang, S. C.-H. Huang, and W. Wu, "Improving construction for connected dominating set with steiner tree in wireless sensor networks," *Journal of Global Optimization*, vol. 35, no. 1, pp. 111–119, 2006.

[10] B. Das and V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets," in *Communications, 1997. ICC'97 Montreal, Towards the Knowledge Millennium. 1997 IEEE International Conference on*, vol. 1. IEEE, 1997, pp. 376–380.

[11] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*. ACM, 1999, pp. 7–14.

[12] H. Lim and C. Kim, "Flooding in wireless ad hoc networks," *Computer Communications*, vol. 24, no. 3, pp. 353–363, 2001.