

# Solving Connected Dominating Set Problem in Unit Disk Graphs by Genetic Algorithms

Maryam Gholami

Department of Computer Engineering, Azad University of  
Qazvin Branch, Islamic Azad University  
Qazvin, Iran  
maryamgholami83@yahoo.com

Mohammad Reza Meybodi

Faculty of IT and Computer Engineering  
Amirkabir University of Technology  
Tehran, Iran  
mmeybodi@aut.ac.ir

**Abstract**— In this paper, we use Genetic Algorithms to find the Minimum Connected Dominating Set (MCDS) of Unit Disk Graphs (UDG). UDGs are used for modeling ad-hoc networks and finding MCDS in such graphs is a promising approach to construct an efficient virtual backbone in wireless ad-hoc networks. The MCDS problem is proved to be NP-complete. The simulation results show that the proposed algorithm outperforms the existing CDS-based backbone formation algorithms in terms of the backbone size.

**Keywords**- *Wireless ad-hoc networks, Backbone formation, Genetic algorithm, Connected dominating set*

## I. INTRODUCTION

Wireless ad-hoc networks can be quickly deployed for many applications. Unlike wired networks, there is no physical backbone infrastructure in wireless ad hoc networks. In such networks two hosts can directly communicate when they are within the range of each other, and they communicate indirectly through relaying packets by the intermediate hosts. Constructing a virtual network backbone in such networks significantly reduces the communication overhead. The CDS formation is a promising approach for constructing this virtual backbone [2].

A wireless ad-hoc network can be modeled as a Unit Disk Graph (UDG) [1]. A graph  $G = (V, E)$  is a UDG if and only if its vertices can be put in one-to-one correspondence with equalized circles in a plane in such a way that an edge connects two nodes if and only if the corresponding circles intersect. In modeling a network by UDG, nodes represent the individual hosts and an edge connects two nodes if the corresponding hosts are within the transmission range of each other.

For graph  $G$ , Dominating Set,  $S$ , is defined as a subset of  $V$  such that each node in  $V-S$  is adjacent to at least one node in  $S$ . Each node in dominating set  $S$  is called a dominator node, otherwise it is called a dominee node. A node of  $S$  is said to dominate itself and all its neighbors. A minimum DS (MDS) is a DS with the minimum cardinality. Finding a MDS is NP-Hard [7]. A Connected Dominating Set (CDS)  $C$  of a graph  $G$  is a Dominating Set whose induced sub graph is connected. CDS with minimum cardinality is called Minimum CDS (MCDS) which forms a virtual backbone [3, 4, 5, and 6] in the

graph. Restricting the routing to the CDS results in a significant reduction in message overhead associated with routing updates [1] by which the routing overhead can be reduced. Finding MCDS is also NP-Hard [7]. A sample UDG and one of its virtual backbones induced by the CDS have been shown in Figs. 1 and 2 respectively.

Genetic Algorithm (GA) is a stochastic search method which is inspired by natural biological evolution. It was first proposed by John Holland [8]. D.E. Goldberg has given a new dimension to GA using it in search, optimization and machine learning [9]. The basic concept of GA is designed to simulate the processes in natural system necessary for evolution, specifically for those that follow the principle of survival of the fitness, laid down by Charles Darwin. The basic steps of GA are shown in Fig. 3.

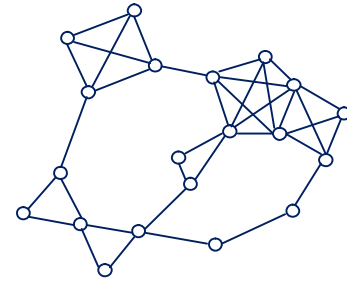


Figure 1. sample unit disk graph

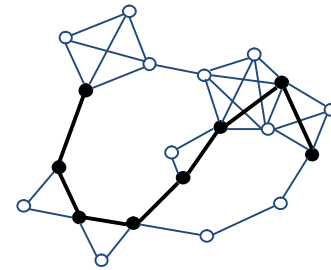


Figure 2. The virtual backbone induced by CDS

### Genetic Algorithms Outline

Compute initial population  $P_0$   
**While** stopping condition not fulfilled **Do**  
    Select individuals for reproduction;  
    Create offsprings by crossing individuals;  
    Eventually mutate some individuals;  
    Compute new generation;  
**End.**

Figure 3. The basic steps of a GA

In this paper, a genetic algorithm based method is proposed to form a virtual backbone for wireless ad-hoc networks by finding a near optimal solution to the minimum CDS problem in the graph of the network. In the energy constrained ad hoc and sensor networks, the proposed method helps to extend the network lifetime due to its smaller size CDS compared to other CDS schemas, in terms of 1) giving better energy conservation and 2) reducing the network traffic. The rest of this paper is organized as follows. The next section reviews the related work. Section 3 describes the proposed Genetic Algorithm and finally our simulation results are given in section 4, and we draw our conclusions in section 5.

## II. RELATED WORKS

MCDS problem is an NP-hard problem, and so several approximation algorithms have been proposed to find a near optimal solution to this problem in a reasonable time. Guha and Khuller [10] proposed two greedy centralized algorithms to construct a CDS in general graph  $G$ . The idea of the first algorithm is to build a spanning tree  $T$  rooted at the nodes with maximum degree and grow  $T$  until all nodes are added to  $T$  and the non-leaf nodes in  $T$  form a CDS. The second algorithm is an improvement of the first one and consists of two phases. The first phase is to construct a DS and the second phase is to connect DS nodes using a Steiner tree algorithm.

Butenko et al. [11] also proposed a prune-based heuristic algorithm for constructing the MCDS. In this algorithm, the connected dominating set is initialized to the whole nodes of the graph and then each node is examined to determine whether it should be removed or retained. If eliminating a given node disconnects the induced sub graph of the connected dominating set, then it is retained and otherwise removed.

Wu and Li [12] proposed an algorithm that determines a CDS using a marking process. In particular, if a node has two unconnected neighbors, then it marked true. At last all the marked nodes form a CDS. The authors also introduce some pruning rules to reduce the size of the CDS. Several distributed algorithms also proposed for solving MCDS problem such as the algorithm proposed by Alzoubi et al. in [13] that forms a minimum CDS in a wireless ad hoc network in two phases. In the first phase a maximum independent set (MIS) is constructed and then in the second phase, CDS is generated by adding intermediate nodes into MIS.

Li et al. [14] proposed a greedy algorithm for constructing CDS in network topology graph of a wireless ad hoc network. The first step of the proposed algorithm focused on

constructing an MIS of the graph. At the second step, a Steiner tree is approximated with the minimum number of Steiner nodes to interconnect the nodes in the MIS. In [15], Xie et al. proposed a distributed approximation algorithm to construct the MCDS in wireless sensor networks. In their algorithm, the network is modeled as a hierarchical graph that at each level of this graph, a selected set of nodes is served as the message hubs (to route the messages) for the other nodes in the next level of the hierarchical graph. This algorithm uses a competition-based strategy to select the nodes at each level. Misra et al. in [17] proposed a new heuristic called collaborative coder using two principles: 1) domatic number of a connected graph is at least two and 2) optimal substructure defined as subset of independent dominator preferably with a common connector. They obtain a partial Steiner tree for the independent set of  $G$ . Many algorithms also have been proposed for solving the CDS problem. One can find a good survey of these algorithms in [18].

## III. THE PROPOSED GENETIC ALGORITHM FOR FINDING MCDS

We described main features of our genetic algorithm (GA) for the CDS problem as follows:

### A. Representation

The chromosome used for this problem consists of a string of bits which length is equal to the number of the nodes in the graph. Each gene (one bit) in the chromosome corresponds to a node in the considered graph and the value of each gene (allele) determines whether that node is selected as a dominator node or not. For an example a typical chromosome for GA that be applied to graph  $G$  with 5 vertices is in form 01001 that denote vertices 2 and 5 are selected as dominator.

### B. Fitness

Previous works on solving graph problems by genetic algorithm, have considered definitions of the fitness function which combine information on various properties of the sub graph (feasible solution) represented by a chromosome, like its density and size. We use a rather simple fitness function, which describes just one property of the subgraph and embed a penalizing mechanism for preventing the production of infeasible solutions.

We define the fitness value of an individual, say  $S$ , in GA population as follows. If  $S$  be a feasible solution, the fitness value is equal to the number of vertices in the solution (i.e. the number of 1's occurring in the string). If  $S$  be an infeasible solution, the fitness function gives it a bad fitness such as the number of all nodes of the graph. This is to avoid selection of the chromosomes which their corresponding subgraph is not CDS, for next generation. Fitness function is shown in (1), where  $n$  is the number of nodes in the graph:

$$f(S) = \begin{cases} \sum_{j=1}^n S[j] & \text{If } S \text{ is a feasible solution} \\ n & \text{Otherwise} \end{cases} \quad (1)$$

### C. Selection

This operator is motivated by the evolutionary mechanism implied by the well-known phrase "survival of the fittest." In the context of the genetic algorithm, it means that greater representation is given to the fitter strings in the current population of solutions. For the purpose of this paper, we use elitist selection mechanism which copies the two best individuals of a population to the population of the next generation.

### D. Crossover

This operator is motivated by the biological process of meiosis. Suppose that  $x$  and  $y$  are parent chromosomes. We say that  $z$  is a potential child of  $x$  and  $y$ , if  $z_i = x_i$  or  $z_i = y_i$  for  $i \in \{1, 2, \dots, n\}$ . In our proposed algorithm after two parents (chromosomes) have been selected for crossover, the GA uses stochastic uniform crossover for generating offspring.

### E. Mutation Operators

This operator motivated by evolutionary mechanism of mutation, where a chromosome undergoes a small but important modification. We implemented and compared three type of mutation operator in proposed GA algorithm. The mutation operators which we used are described below:

#### 1) The simple mutation operator

The simple mutation operator simply selects one vertex in the subset of vertices and changes its corresponding allele to its complement, means that one node to be added to the subgraph or one node to be deleted from it.

#### 2) Hypermutation (HM) operator

The basic hypermutation heuristic that proposed by Correa, et al. [16] starts by randomly selecting a percentage of the chromosomes of the population. The algorithm then tries to improve the fitness of each of the selected chromosomes as follows. For a given node in the chromosome, the algorithm performs the change that most improves the chromosome's fitness. Note that chromosome representation in [17] is different from our representation in which the chromosome is a list of vertices in the subset. For an example of their representation, assume the vertex set of a given graph be labeled by  $\{1, 2, 3, 4, 5, 6, 7\}$ , a typical chromosome is  $\{2, 3\}$  means that nodes 2, 3 selected as dominator. We customize the hypermutation operator for our algorithm with its particular representation. The implementation of our hypermutation algorithm is shown in Fig. 4. We investigate two cases:

Case 1: if the number of 1s in the chromosome is more than or equal to half of the length of the chromosome, we use hypermutation type 1 called HM1. This algorithm deletes dominator nodes one by one via inverting its value to 0 in the chromosome, and checks that which of these changes results most improvement in the fitness of chromosome and then performs that change on the chromosome. The implementation of HM1 is shown in Fig. 5.

Case 2: if the number of 1s in the chromosome is less than half of the length of the chromosome, we use hypermutation type 2 called HM2. HM2 changes a dominator node to non dominator node and instead selects another non dominator

node to be dominator. The implementation of HM2 is shown in fig. 6.

```

Procedure Hypermutation
Step1:
Randomly select a subset of 10% of the chromosomes from the
entire population
Step2:
For each chromosome X selected in step 1
    If the number of 1s in X >= n/2
        HM1(X)
    Else
        HM2(X)
    End if
End for
End procedure

```

Figure 4. The hypermutation algorithm

```

Procedure HM1(X)
Best = X
For each node j with value 1 in the chromosome X
    Let Y be a new chromosome with value 0 at j'th gene
    Calculate the fitness of Y
    If fitness(Y) < fitness(best)
        Best = Y
    End if
End for
If fitness(best) < fitness(X)
    X = best
End if
Insert the new X into the population replacing the old X
End procedure

```

Figure 5. The HM1 algorithm

```

Procedure HM2(X)
Let H be the set of nodes in the graph that are currently 1 in the
chromosome X
For each node i included in set H
    Best = X
    For each node j that has value 0 in X
        Let Y be a new chromosome with value 1 at
        j'th gene and value 0 at i'th gene
        Calculate the fitness of Y
        If fitness(Y) < fitness(best)
            Best = Y
        End if
    End for
    If fitness(best) < fitness(X)
        X = best
    End if
End for
Insert the new X into the population replacing the old X
End procedure

```

Figure 6. The HM2 algorithm

We illustrate the use of these operators in our proposed algorithm by an example. Consider a graph with vertex set {1, 2, 3, 4, 5, 6, 7}, a typical chromosome is 1011011 means that nodes 1, 3, 4, 6, 7 are in the dominator set. Since the number of 1's is more than half of the length of this chromosome, HM1 is applied. This algorithm inverts the 1's one by one and so the chromosomes should be evaluated are 0011011, 1001011, 1010011, 1011001 and 1011010. Finally, the algorithm selects the change that results in maximum improvement in the fitness of the chromosome.

Let's consider another chromosome such as 1010010. In this case the HM2 should be applied. The algorithm first considers the first position containing 1 and changes its value to 0 and then searches for a non dominator node which making it a dominator causes maximum improvement in the fitness of the chromosome. The chromosomes that should be examined are 0110010, 0011010, 0010110, and 0010011. The drawback of this algorithm is that it takes the entire chromosome and mutates its node by node with all non dominator nodes. So it takes a lot of time and it is computationally expensive.

### 3) KN algorithm

We developed another heuristic which is an enhancement of algorithms described above. This algorithm uses a local optimization rather than a global optimization technique. The idea behind this algorithm is to mutate every gene with only its neighbors. The number of neighbors that are investigated (K) depends on the average degree of the network nodes. We define average degree of the network nodes as (2).

$$\alpha = \frac{\sum_{i=1}^n \delta_i}{n} \quad (2)$$

In which  $\alpha$  denotes the average degree of the network.  $\delta_i$  is degree of node  $i$  and  $n$  is the number of nodes in the network.

In fact  $k$  neighbors of a given node are investigated that  $k$  is at most  $\alpha$ , so we call this algorithm  $k$  neighbors (KN). The implementation of KN is shown in figure 6. Note that set  $H$  is calculated for each node in the chromosome.

## IV. COMPUTATIONAL RESULTS

To study the performance of our GA algorithm for solving CDS problem, we have conducted simulation experiments in two groups. The first experiment involves comparing impact of applying three different mutation operators in implementation of proposed genetic algorithm on solving CDS problem. The second group evaluates the results of the proposed GA with those of the best-known CDS formation algorithms. The performance measures of experiment 1 are CDS size and run time and in experiment 2, the performance measure is only CDS size.

In our experiments we generate random connected graphs repeatedly and run the algorithms, measuring the size of the CDS. The size of the graph ranges from 60 to 200 nodes. To simulate the structure of ad hoc networks, we place nodes (hosts) randomly in a square simulation area of size  $100 \times 100$  units. The coordinates of the nodes are chosen uniformly in each dimension. It is assumed that the transmission range for each host is 20. The parameters of our genetic algorithm are setting as bellow: population size is equal to 100, crossover rate is 0.8 and the number of the iterations is 100.

**Experiment 1:** In this experiment we study the impact of applying three different mutation operators in our genetic algorithm for solving the CDS problem. In this experiment we set the stalltime of the GA algorithm to 8 seconds and compare the size of the CDSs produced by GA with three different mutation algorithms. The results are depicted in fig. 9. As shown in figure 8, the GA with KN heuristic performs better than the simple mutation and hypermutation in the same amount of time. This occurs because hypermutation is trying all of the options rather than KN algorithm; therefore better results can be achieved in the same amount of time by applying KN algorithm.

#### Procedure KN

Step1:

Randomly select a subset of 10% of the chromosomes from the entire population

Step2:

For each chromosome X selected in step1

For each node  $i$  in X that has value 1

Best = X

Let H be the set of neighbors of node  $i$  that are currently 0 in the chromosome X (maximum cardinality of this set is average degree of the graph)

For each node  $j$  included in H

Let Y be a new chromosome with value 1 at  $j$ 'th gene and value 0 at  $i$ 'th gene

Calculate the fitness of Y

If fitness(Y) < fitness(best)

Best = Y

End for

If fitness(best) < fitness(X)

X = best

End for

Insert the new X into the population replacing the old X

End for

End procedure

Figure 7. The KN algorithm

**Experiment 2:** In this experiment, we compare the results obtained from our proposed algorithm with the results of Butenko et al.'s algorithm [11], Li et al.'s algorithm [14], Xie et al.'s algorithm [15] and Misra's algorithm [17]. In terms of induced CDS size. The results are shown in fig. 10. Our algorithm is labeled as GA. From the figure it is obvious that the size of CDS constructed by our proposed algorithm is comparable with those constructed by the best existing algorithms.

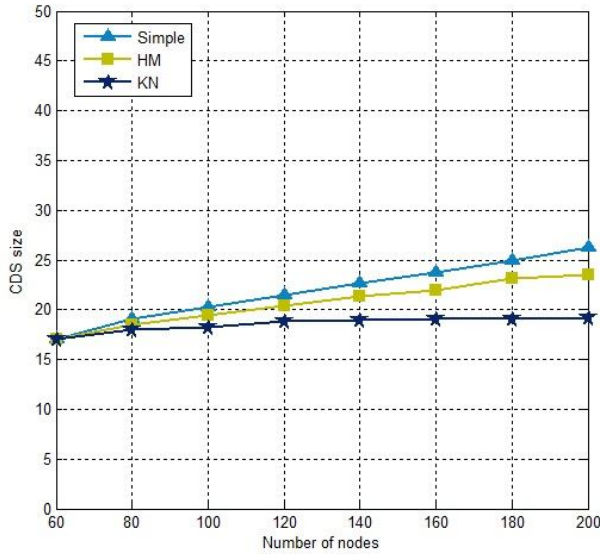


Figure 8. Comparison of three mutation algorithms on the resulting backbone size

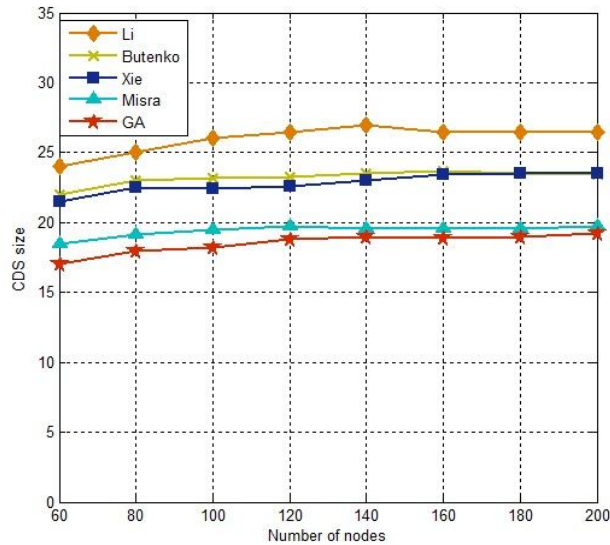


Figure 9. Comparison of the backbone size for the CDS-based backbone formation algorithms

## V. CONCLUSION

In this paper, we presented a method based on genetic algorithms for solving the MCDS problem in unit disk graphs. Most of the components of proposed GA are comparable to those used in a standard GA. We implemented the hypermutation heuristic concept presented by Correa, et al. [16] with some modifications to tune it for our problem. We also developed a heuristic called KN (K Neighbors) heuristic for mutation operator. Experimental results show that GA with KN heuristic works better than simple mutation and hypermutation. Furthermore we found that the proposed GA with KN heuristic also outperforms the best existing algorithms for finding MCDS in UDGs.

## REFERENCES

- [1] V. Bharghavan and B. Das, "Routing in Ad Hoc Networks Using Minimum Connected Dominating Sets", International Conference on Communications'97, Montreal, Canada, June 1997, pp. 376-380.
- [2] Y. Z. Chen, A. L. Liestman, "Approximating inimum size weakly connected dominating sets for clustering mobile ad hoc networks", in: Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'2002), 2002, pp. 157-164.
- [3] K. M. Alzoubi, P. J. Wan, O. Frieder, "Maximal independent set, weakly connected dominating set, and induced spanners for mobile ad hoc networks", International Journal of Foundations of Computer Science, Vol. 14, No. 2, 2003, pp. 287-303.
- [4] P. J. Wan, K. Alzoubi, O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks", in: Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002), Vol. 3, 2002, pp. 1597-1604.
- [5] J. Wu, B. Wu, I. Stojmenovic, "Power-aware broadcasting and activity scheduling in ad hoc wireless networks using connected dominating sets", Journal of Wireless Communications and Mobile Computing, 2003, pp. 425-438.
- [6] H. Lim, C. Kim, "Flooding in wireless ad hoc networks", Journal of Computer Communications 24, 2001, pp. 353-363.
- [7] M. R. Garey and D. S. Johnson, "Computers and Intractability: A guide to the theory of NP-completeness", Freeman, San Francisco, 1978.
- [8] J. H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, MI, 1975.
- [9] D. E. Goldberg, Genetic Algorithm in Search, "Optimization and Machine Learning", Addison-Wesley, Reading, MA, 1989.
- [10] S. Guha, S. Khuller, "Approximation algorithms for connected dominating sets", Algorithmica, Vol. 20, No. 4, 1998, pp. 374-387.
- [11] S. Butenko, X. Cheng, C. Oliveira, P.M. Pardalos, "A new heuristic for the minimum connected dominating set problem on ad hoc wireless networks", in: Recent Developments in Cooperative Control and Optimization, Kluwer Academic Publishers, 2004, pp. 61-73.
- [12] J. Wu, H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks", in: Proceedings of the Third ACM International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (ACM DIALM'1999), 1999, pp. 7-14.
- [13] K.M. Alzoubi, X.Y. Li, Y. Wang, P.J. Wan, O. Frieder, "Geometric spanners for wireless ad hoc network", IEEE Transactions on Parallel and Distributed Systems, Vol. 14, No. 4, 2003, pp. 408-421.
- [14] Y. Li, M.T. Thai, F. Wang, C.W. Yi, P.J. Wang, D.Z. Du, "On greedy construction of connected dominating sets", in: Wireless Networks, Special Issue of Wireless Communications and Mobile Computing (WCMC), 2005.
- [15] R. Xie, D. Qi, Y. Li, J.Z. Wang, "A novel distributed MCDS approximation algorithm for wireless sensor networks", Journal of Wireless Communications and Mobile Computing, 2007.

- [16] E. S. Correa, M. T. A. Steiner, A. A. Freitas, C. Carnieri, "A Genetic Algorithm for the P-median Problem In: Genetic and Evolutionary Computation Conference - GECCO 2001, 2001, San Francisco, California. Proceedings of the Genetic and Evolutionary Computation Conference GECCO 2001. San Francisco, California: Morgan Kaufmann Publishers, 2001.
- [17] R. Misra, Ch. Mandal, "Minimum Connected Dominating Set using a Collaborative Cover Heuristic for Ad Hoc Sensor Networks", IEEE Transactions on parallel and distributed systems, Vol. 21, No. 3, 2010.
- [18] Z. Liu, B. Wang , L. Guo, "A survey on connected Dominating Set Construction Algorithm for Wireless Sensor Networks", Information Technology Journal, Vol. 9, No. 6, 2010.