Journal of Computer Science
https://dergipark.org.tr/en/pub/bbd
Anatolian Science

ISSN,e-ISSN: 2548-1304
Volume:10, Issue:1, pp:101-115, 2025
https://doi.org/10.53070/bbd.1660231
Research Paper

A New Approach for Minimum Dominating Set Problem: A Three-Stage Solution with Malatya Centrality Metrics

Şeyda Karcı[1], Fatih Okumuş[2], İhsan Tuğal[*3], Murat Demir[3], Ali Karcı[2]

[1]Department of Computer Engineering, Marmara University, İstanbul, Türkiye

[2] Department of Software Engineering, İnönü University, Malatya, Türkiye

[3] Department of Software Engineering, Muş Alparslan University, Muş, Türkiye

(seyda.karci@marmara.edu.tr, fatih.okumus@inonu.edu.tr, i.tugal@alparslan.edu.tr, m.demir@alparslan.edu.tr, ali.karci@inonu.edu.tr)

Received:Mar.18, 2025 Accepted:May 26, 2025 Published:Jun.1, 2025

Abstract— The Minimum Dominating Set (MDS) problem is a fundamental challenge in graph theory, with wide-ranging applications across various domains. This study proposes a novel three-phase approach for solving the MDS problem, generating solutions at each stage of the process. The proposed method leverages three interconnected centrality metrics, offering an effective way to address the MDS problem at different problem scales. These centrality measures capture the prominence of a target node in relation to its neighbors. In the initial iterations, the algorithm prioritizes nodes with high dominance and clustering, gradually incorporating those with lower clustering into the dominating set in subsequent phases. Notably, the computational cost (node selection cost) is higher during the early iterations and decreases over time. The empirical evaluation of the study encompasses a variety of problem scenarios, including synthetic datasets generated by specific strategies, networks based on the Erdős–Rényi model, and authentic datasets from network science applications. The results show that the proposed algorithm consistently produces robust solutions under various constraints. In many cases, its performance surpasses that of existing algorithms. The findings of this study contribute to the evolving landscape of MDS problem-solving techniques and offer valuable insights for future research and practical applications in areas such as network design and resource allocation.

Keywords : *Minimum dominating set, Domination number, Centrality, Greedy heuristics, Graph algorithms*

1. Introduction

The Minimum Dominating Set (MDS) problem entails the quest for a dominant set within a network or graph, while minimizing the number of elements required for domination. The identification of dominant nodes represents a prominent domain of study within graph theory and constitutes a classic combinatorial optimization challenge.

Fast and effective access to all network nodes is crucial, especially since some nodes can significantly influence the network and reflect its topological structure. Identifying such influential nodes is essential in network analysis, as they can be used to improve network performance through various strategies. Moreover, efficiently reaching the entire network through direct communication with a small group of key individuals is a crucial problem. The challenge lies in finding the most effective way to connect with all nodes using as few intermediaries as possible (Truta et al., 2020). Solving the MDS problem within a finite number of steps and polynomial time is widely recognized as a formidable challenge (Wang et al., 2011). Consequently, common strategies for addressing this problem encompass the utilization of greedy, heuristic, and metaheuristic algorithms (Potluri & Singh, 2013; Sanchis, 2002). However, existing methods, while useful, do not always provide exact solutions for all network types. There is a need for algorithms that can deliver optimal results across different network structures. This study aims to present an effective and optimal algorithm for MDS to fill this gap.

The MDS problem is important for solving various real-world challenges. It helps identify the most critical nodes in a network, allowing for better protection and faster access to the entire system. In communication or energy networks, MDS can reveal nodes vulnerable to attacks, improving security. It also supports efficient

resource allocation and strengthens the network's ability to withstand failures. These benefits are especially relevant for social networks like Facebook, Twitter, or Instagram, which play a key role in modern communication and information sharing. Online social networks are rapidly growing and have a strong impact on daily life. One important feature of these networks is the ability to enable fast communication between users, which can be achieved using MDS.

This study aims to produce an effective solution based on Malatya centrality values (Karcı et al., 2022). It also supports and continues the study previously done with Malatya centrality for MDS (Okumuş & Karcı, 2024). A solution to the Minimum Dominating Set problem is proposed using Malatya Centrality approaches. A greedy heuristic method was chosen, although it does not guarantee finding the exact Minimum Dominating Set. The method selects nodes through a three-stage calculation based on newly introduced centrality metrics. In each iteration, the node with the highest Malatya centrality among the unselected nodes is chosen to dominate the network. The algorithm was tested on various scenarios, including synthetic graphs generated by a specific algorithm, large random graphs created with the Erdös-Renyi model, and real-world graphs. The results show that the algorithm produces consistent outcomes when run multiple times on the same graph.

Casadoa et al. proposed the Iterative Greedy algorithm to reach the minimum dominating set and low dominance number. They presented a meta-heuristic approach that aims to provide high-quality solutions in short computational times. Iterative Greedy uses heuristic procedures to obtain an initial solution, an efficient local search method to refine it, and strategies to destroy and reconstruct a solution during the search (Casado et al., 2023).

Karcı obtained a special spanning tree to solve the problem of obtaining the minimum dominating set and called it Kmax tree. The fundamental cut sets of the graph are obtained using this spanning tree. The dominance of each node is calculated based on these basic cut sets, node degrees in the graph, and the corresponding Kmax tree. The method proposed in this study confirmed that this problem can be solved with deterministic algorithms (Karci, 2020).

Chalupa introduced a novel row-based random local search algorithm (RLSo) designed to address the Minimum Dominating Set problem in large graphs. The algorithm underwent testing across diverse scenarios, including a wireless network model, unit disk graphs representing arbitrary scale-free networks, and real-world graphs. RLSo demonstrated superior performance when compared to both the classical greedy approximation algorithm and two metaheuristic algorithms based on ant colony and local search. (Chalupa, 2018).

A polynomial time approximation algorithm for the MDS problem was proposed by Mira et al. The algorithm works in two stages. In the first stage, a dominant set is created by a greedy algorithm. In the second stage, this dominant set is purified (reduced). The reduction is achieved by analyzing the flowchart of the algorithm in the first stage and a special type of set of the dominating set created in the first stage (Hernández Mira et al., 2022).

Li et al. conducted a performance assessment of several approximation algorithms for computing the Minimum Dominating Set of a graph. Their evaluation encompassed a comparison of the standard greedy algorithm, contemporary Linear Programming (LP) rounding algorithms, and a hybrid algorithm they formulated by combining the greedy and LP rounding techniques. The results of their experiments demonstrated that each of these approaches holds distinct advantages depending on the characteristics of the dataset. (Li et al., 2020).

Connolly et al. calculated the minimum dominating set of a random graph using edge probabilities. By examining the dominance numbers of dense random graphs; Studies have been conducted to obtain one or two point concentrations (Connolly et al., 2016).

An algorithm for minimum dominating set was developed by Zhou et al. This proposed algorithm presented a 4-stage method. They included results comparing their own algorithms with the fastest algorithms known in the literature (Zhou et al., 2014).

Jovanovic and Tuba proposed the ant colony algorithm to solve the minimum dominating set problem. With the help of this new approach, a strategy for regulating the amount of pheromone has been developed. Thus, they solved the problem of the one-step ant colony algorithm getting stuck in local optimum (Jovanovic & Tuba, 2013)

Abdel-Rahman and Rashad proposed a new hybrid method based on genetic algorithm to solve the minimum dominating set problem. The proposed method uses local search and condensation schemes along with genetic algorithm search methodology to achieve better performance (Hedar & Ismail, 2010).

Potluri et al. presented a hybrid genetic algorithm that incorporates local search techniques along with an ant colony optimization algorithm for addressing the Minimum Dominating Set problem. The study conducted a performance comparison between these two hybrid algorithms and the solutions derived from a greedy heuristic approach and another hybrid genetic algorithm previously proposed in the literature. (Potluri & Singh, 2011).

Albuquerque and Vidal introduced a hybrid metaheuristic method that combines tabu search and integer programming to tackle the Minimum Dominating Set problem. This approach fuses tabu search techniques with an integer programming solver to provide a comprehensive solution. (Albuquerque & Vidal, 2018).

Abed and Rais proposed a stochastic search methodology that leverages a hybrid swarm intelligence algorithm for identifying the minimum node set capable of dominating a given graph. The approach incorporates a population-based technique known as the bat algorithm, which facilitates an extensive exploration of the search space, thereby enhancing the diversification process (Abed & Rais, 2017).

The content of this study begins with an introduction to the Minimum Dominating Set problem, followed in the second section by the explanation of centrality and the proposed Malatya Centrality techniques. The third section of the study is devoted to a comprehensive evaluation of these methods on various data sets. Finally, the results obtained, and relevant recommendations are presented.

1.1 Minimum Dominating Set Problem (MDSP)

The dominant set problem is the problem of finding the smallest set of nodes that dominates all nodes in a graph. Minimum Dominating Set (MDS) problem is the problem of finding a dominant minimum set in a graph (Hedar & Ismail, 2010). Let $G = (V, E)$ be a graph. V is the set of nodes, E is the set of edges. Let $n = |V|$ be the number of nodes and $m = |E|$ be the number of edges. In a graph, each node in the dominating set is expressed as $V_D \subseteq V$. If all the remaining nodes in the graph called V_N (VN - Vertex Neighbors - vertice(s) of dominating set neighbors) can be reached by the elements of the V_D (VD - Vertex Dominating - vertice(s) of dominating set) set, V_D can be called the dominant set. If the V_D set is created to contain the minimum number of elements, this dominant set is called the minimum dominating set (Kapoor et al., 2013) (Casado et al., 2023).

MDS problem: a minimum-dimensional subset in a graph. The elements in this subset have at least one neighbor, and this subset is created to include all graph nodes together with their neighbors (Nguyen et al., 2020). There is no deterministic effective algorithm in the literature to find the minimum dominating set. Algorithms that try to find an exact solution generally use the greedy algorithm. The downside of this approach is that the solution time is long (Karcı et al., 2022).

The size of G's minimum dominating set is called the domination number ($\gamma(G)$). The minimum achieved domination number of a graph is the minimum size of the dominant node set. It can also be said to be in the minimally dominating set. In a graph with n nodes and maximum node degree Δ, the domination number value is $\frac{n}{1+\Delta} \leq \gamma(G) \leq n$. In a fully connected star or wheel graph, the number of dominating set elements is 1. In graphs that do not have isolated nodes, that is, nodes with a minimum node degree of 1, this value is $\gamma(G) \leq \frac{n}{2}$ (Bujtás & Klavžar, 2016; Ore, 1962).

The aim is to reach all elements outside the dominating set in a graph. The problem of identifying a good minimum dominating set consists of finding the smallest group of nodes such that nodes not in that group have a person in that group with whom they are in a known relationship. The solution is a dominant set with the smallest cardinality. MDS is a difficult process to find and has entered the literature as an NP-hard problem (Li et al., 2020). A fixed algorithm does not produce a healthy solution for all kinds of graphs. For large and complex examples, it is necessary to be willing to find approximate results rather than exact results. Algorithms developed for MDS problem solution are mostly based on heuristic or greedy approaches (Casado et al., 2023).

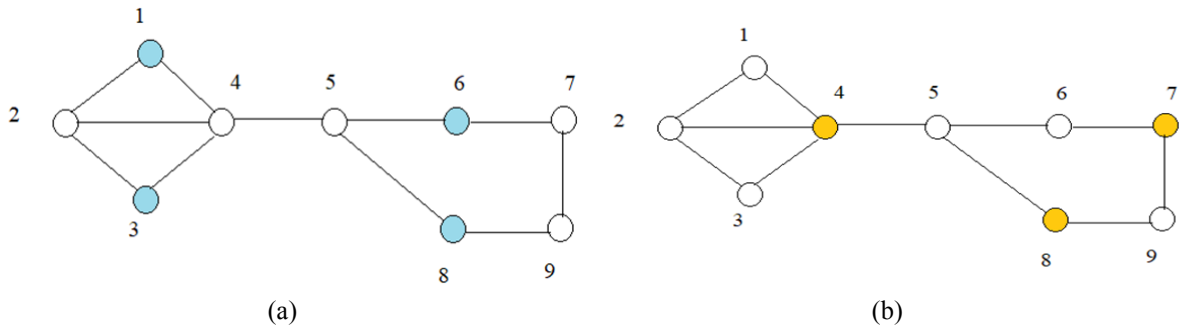


Figure 1. Example of dominating set and minimum dominating set

Figure 1 (a) is an example of a dominating set, Figure 1 (b) is an example of MDS. In (a), the dominating set nodes are 1, 3, 6, 8. In (b), the minimum dominating set nodes are 4, 7, 8. Although all nodes are covered in both, in the MDS this coverage process is carried out with a minimum number of elements.

2.Methods

2.1. Centrality

The selection of nodes for the Minimum Dominating Set (MDS) problem is based on their centrality values. Centrality measures how important or effective a node is within a network. These metrics help reveal the position, influence, and overall importance of nodes, making it easier to identify key nodes in large and complex networks. Centrality also helps understand how information flows through the network. Therefore, centrality values provide valuable insights for solving many real-world problems (Estrada, 2011; Newman, 2003; İ. Tuğal & Karcı, 2019; İhsan Tuğal & Karcı, 2020). There are many different proposed methods used for centrality calculations.

This study employed degree centrality and neighbor node degree as centrality measurement methods. Degree centrality serves as the fundamental and widely applied centrality metric. It quantifies the degree of a node, representing the count of edges directly linked to that specific node. Nodes characterized by higher degrees are typically perceived as more central, owing to their increased number of connections within the network (Gao et al., 2013; Hu et al., 2016).

In addition to the node's degree, the degrees of its neighboring nodes represent another critical parameter influencing centrality. While originally designed as an algorithm to rank web pages, PageRank has found applications as a centrality metric in network analysis. Its fundamental principle revolves around assessing a node's importance based on its connections with other influential nodes. PageRank computes a node's centrality through a process involving link traversal. Each node commences with a predefined initial value, and at each iteration, it updates its value by calculating a weighted average of the PageRank values of its neighboring nodes, with the weighting determined by the number of connections. This iterative process continues until equilibrium is achieved (Brin & Page, 1998; Page et al., 1998). In this study, an algorithm that finds the minimum dominating set according to PageRank centrality values was used. Comparison was made with the proposed method.

2.2. First Malatya Centrality

In the first step of the proposed algorithm, the First Malatya Centrality (FMC) values of the nodes in the graph are used. In the FMC calculation, the degree of each node and the degrees of its neighboring nodes are used. FMC values of nodes in any graph are the sum of the ratio of the degree of the node to the degrees of the neighboring nodes for each node. The degree of a node is the number of edges (i.e. number of neighbors) connected to it. In determining the FMC values, the degree of the neighboring nodes is as effective as the degree of the node. The main factor affecting the centrality of a node is that its degree is higher than the degrees of its neighboring nodes. In the first stage, how central each node is evaluated by looking at this value.

In Equation 1, $\Psi_1(v_i)$, is the FMC value of the node i. $d(v_i)$, is the number of neighbors of the node i, that is, its degree. $N(v_i)$, is the set of neighbors of the node i. v_j is one of the neighbor nodes of the node i (Karcı et al., 2022; Yakut et al., 2023).

$$\Psi_1(v_i) = \sum_{\forall v_j \in N(v_i)} \frac{d(v_i)}{d(v_j)} \tag{1}$$

2.3. Second Malatya Centrality

In the Second Malatya Centrality (SMC) calculation, as shown in Equation 2, the FMC values we obtained in the first step are used. $\Psi_2(v_i)$, is the second Malatya centrality value of the node i. Flowchart of FMC and SMC are given in Figure 2.

$$\Psi_2(v_i) = \sum_{\forall v_j \in N(v_i)} \frac{\Psi_1(v_i)}{\Psi_1(v_j)} \frac{1}{d(v_i)} \tag{2}$$

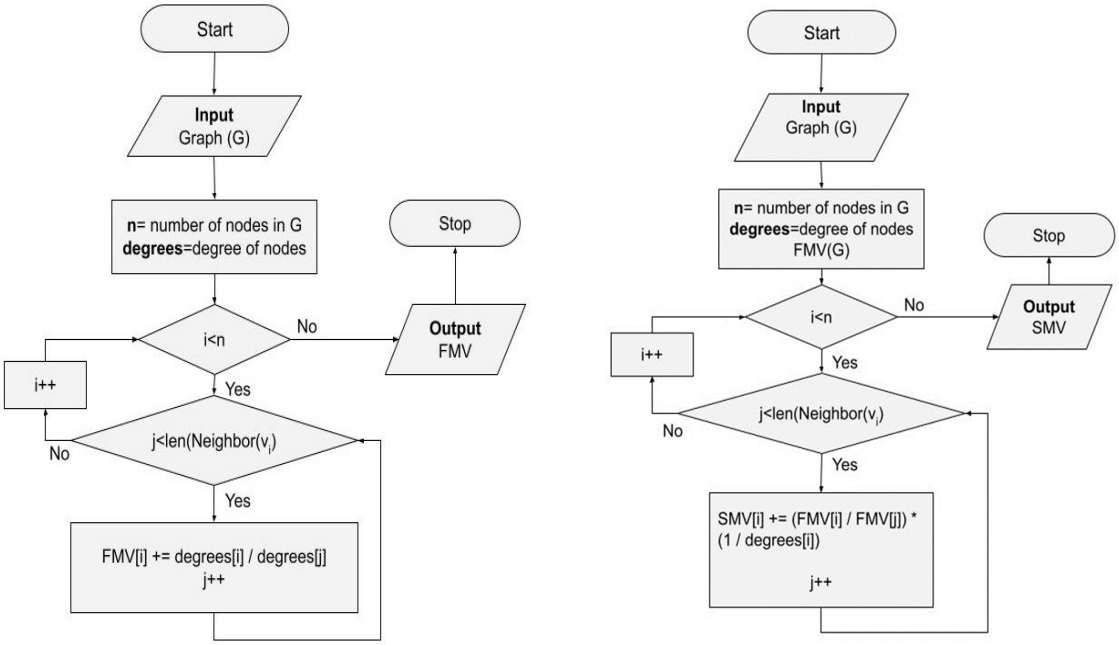


Figure 2. Flowchart of FMC and SMC

2.4. Third Malatya Centrality

As shown in Equation 3, in the Third Malatya centrality (TMC) calculation, the SMC values are used. $\Psi_3(v_i)$, is the TMC value of the node i. The k value is the number of nodes that are neighbors with v_i but are not in the $VD \cup VN$ set. The nodes in the graph are classified as the set of VD-dominant nodes and the set of VN-non-dominant nodes. In this way, the weight of such nodes is increased when calculating the TMC value. Flowchart of TMC is given in Figure 3.

$$\Psi_3(v_i) = \sum_{\forall v_j \in N(v_i)} \frac{\Psi_2(v_i)}{\Psi_2(v_j)} \frac{k}{d(v_i)} \tag{3}$$

2.5. Malatya Minimum Dominating Set (MMDS)

The core of node selection for building the MDS is a key focus of the proposed algorithm. This method uses a three-step Malatya centrality to identify the MDS in the graph. The algorithm starts by taking the input graph G. For each node, it calculates the FMC and SMC values. These first two centrality values are computed only once for all nodes in the graph. The algorithm initializes the sets V_D and V_N as empty sets.

Subsequently, in the following iterations of the algorithm, an evaluation ensues to ascertain whether the combination of V_D and V_N equates to the total number of nodes in the graph. If this condition is met, the algorithm terminates. If not, the algorithm continues to execute. In the process of selecting the initial elements for V_D and V_N, the selection is contingent upon the node harboring the highest SMC value. Then, in the subsequent iterations, nodes are chosen iteratively based on the highest TMC value. This process entails the addition of the neighbors of the node with the highest TMC value to V_N, while the selected node itself is added to the V_D set.

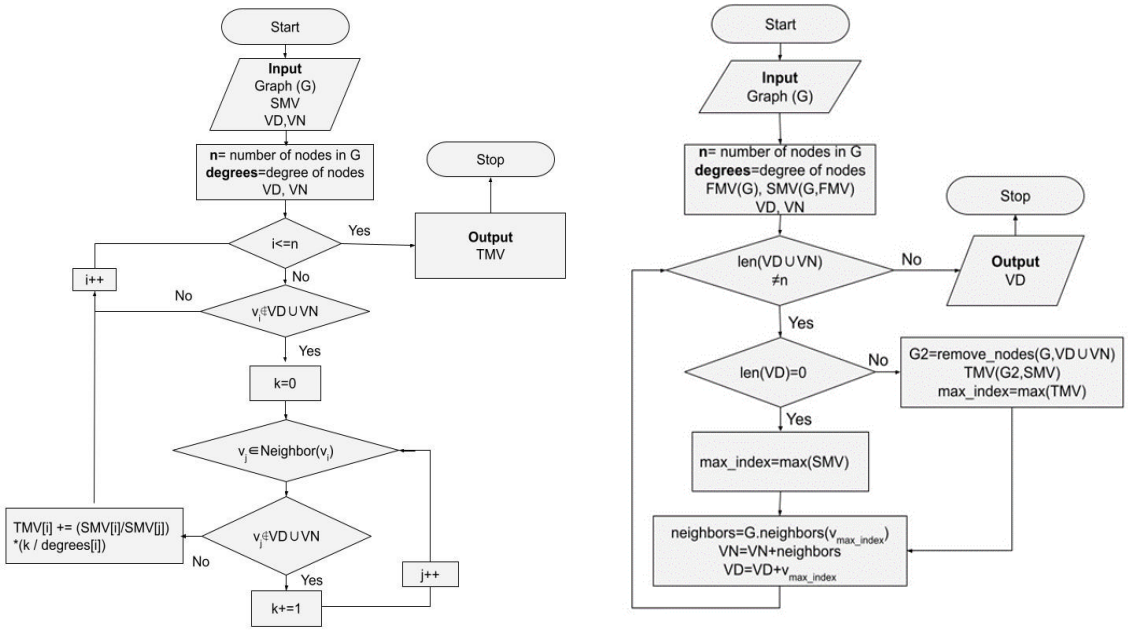
Following the addition of nodes into the V_D and V_N sets, these selected nodes are subsequently removed from the graph. After each removal, TMC values are recomputed. In cases where these recalculated TMC values are equivalent, nodes that have a greater number of neighbors within the combined set of V_N and V_D are afforded priority during the selection process. This iterative procedure persists until the number of nodes encompassed by V_D and V_N equals the total number of nodes in the graph. A visual representation of the algorithm's workflow is depicted in Figure 3, while the pseudocode for MMDS is presented in Algorithm 1.

Greedy algorithms work by repeatedly choosing the best option at each step to build the solution set. For the MDS problem, finding the exact optimal solution is often very difficult or sometimes impossible. Greedy methods, like the one proposed here, aim to find a near-optimal solution by always picking the most promising node, but

they don't guarantee the absolute best result. Although this approach can have high time complexity, it is often preferred because it provides solutions that are close to optimal.

Algorithm1: Pseudocode of MMDS

```
MalatyaMinimumDominatingSet(G)
        n=number of nodes in G
        degrees=degree of nodes
        first_malatya_values=FMV(G)
        second_malatya_values=SMV(G, FMV)
        VD = Empty Set     //Vertices of MDS
        VN = Empty Set      //Neighbors of MDS elements
        While (|VD ∪ VN| ≠ n):
          If |VD| = 0:
              node = max(second_malatya_values)
          Else:
              G2 = G - (VD ∪ VN)
              remaining_nodes = G - VN - VD
              If remaining_nodes is empty:
                 VD = G - VN
                 Exit Loop
              TMV (G2, second_malatya_values, VD, VN, n)
              node = max([third_malatya_values[x] for x in remaining_nodes])
           VN = VN ∪ neighbors(node)
           VD = VD ∪ {node}
        End Loop
return VD
```



Third Malatya Centrality Malatya Minimum Dominating Set

Figure 3. Flowchart of TMC and MMDS

To address time constraints, one option is to reduce the number of steps required from the default three. The MDS identification can be executed with either FMC or TMC. The fundamental operational principles of the proposed MMDS algorithm are encapsulated in Figure 4.

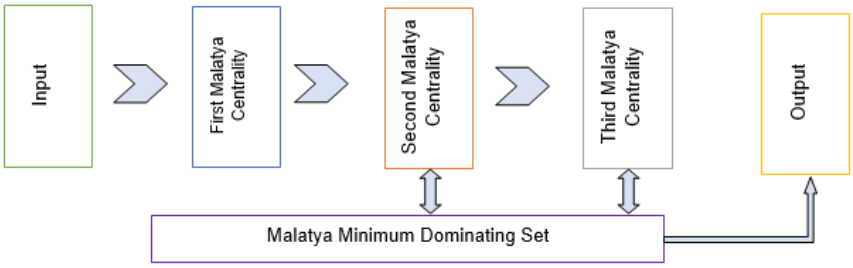


Figure 4. Malatya Minimum Dominating Set Algorithm

2.6. Sample Application

The Zachary Karate Club dataset was used as a case study to demonstrate the steps of the proposed algorithm. This dataset, originally compiled by Wayne W. Zachary in the 1970s, encapsulates the interrelationships among members of a Karate Club (Batool & Niazi, 2014). The dataset clearly shows how internal conflicts in the club caused a split, changing the relationships among members in the new groups. It is a foundational study often referenced by researchers in social network analysis and graph theory.

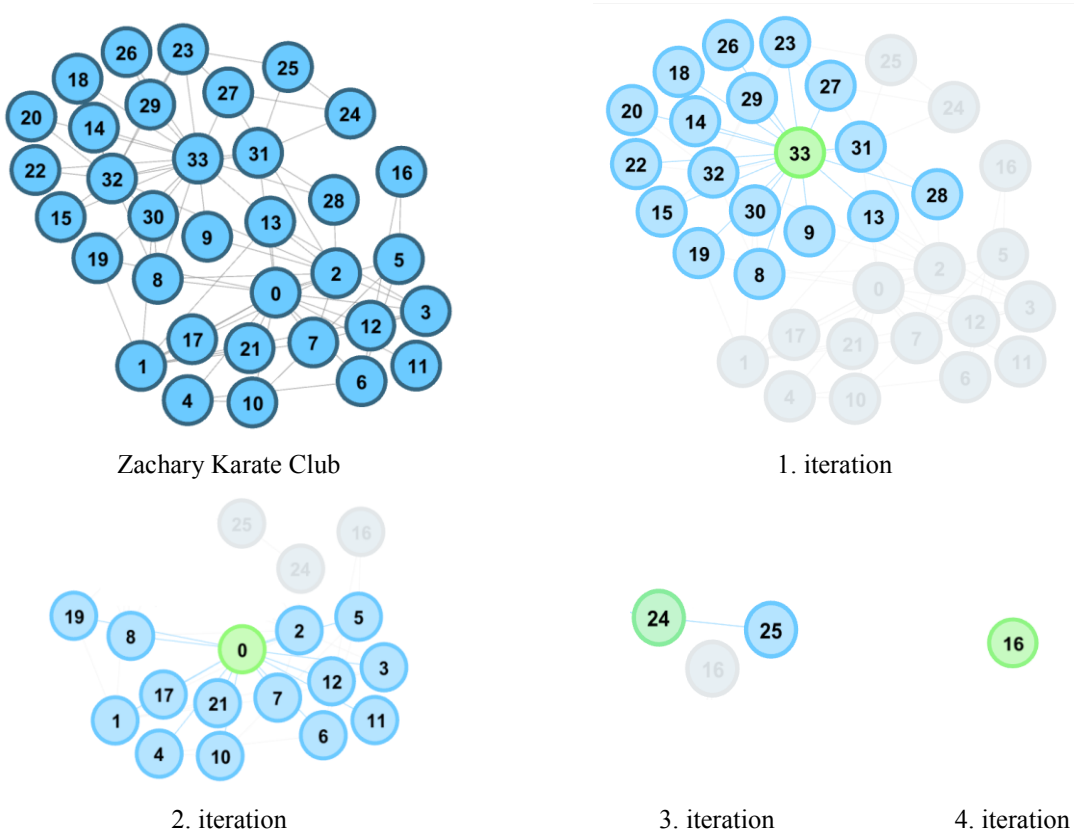


Figure 5. MMDS for Zachary Karate Club

When the algorithm is applied to the Zachary Karate Club as shown in Figure 5., the first node to be selected for the V_D set according to the SMC value is node 33. The SMC value of node 33 is 156.555. The closest value to this is 148.258, which is the SMC value of node 0. The neighbors of node 33 are added to the V_N set. These are $\{8, 9, 13, 14, 15, 18, 19, 20, 22, 23, 26, 27, 28, 29, 30, 31, 32\}$ nodes. When node 33 and its neighbors are removed from the graph, the resulting graph is as shown in the second iteration of Figure 5.

In the second iteration, calculations will be made according to the TMC values. As can be seen in the Figure 5., the node with the highest TMC value is node 0. It is added to the V_D set. Its neighbors are $\{1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 17, 21\}$ nodes. These nodes are added to the V_N set. Node 0 and its neighbors are removed from the graph.

In the third iteration, 3 nodes remain. TMC values for these nodes are calculated. This iteration was examined in a little more depth to understand the functioning of the algorithm. The neighbors of 24 are the set $\{25, 27, 31\}$. The neighbors of 25 are the set $\{23, 24, 31\}$. Its different neighborhoods are nodes 23 and 27. When the neighbors of node 23 $\{ 25, 27, 29, 32, 33 \}$ and the neighbors of node 27 $\{ 2, 23, 24, 33 \}$ are examined, node 27 is associated with the first selected dominant node 33 and its neighbors, as well as with the neighbors of the second selected dominant node 0. The neighbors of 23 are only associated to the first selected dominated node 33 and its neighbors. That is, it is less associated with dominant nodes. Therefore, the algorithm prioritizes node 24 and adds it to the V_D set. Node 25 is also added to the V_N set.

In the fourth iteration, only node 16 remains single in the graph. Since it remains single and has no relationship, node 16 is added to the V_D set.

After the fourth iteration, the algorithm is terminated since the number of V_N union and V_D set elements is equal to the number of nodes in the graph. The number of iterations actually equals the number of V_D sets. The V_D set consisted of $\{33, 0, 24, 16\}$ nodes. V_N set consisted of $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32\}$ nodes.

The algorithm quickly identifies nodes with high clustering values in its early iterations. As a result, the nodes selected in the first two steps often have a dominant influence over most members. While this pattern may vary depending on the graph structure, the algorithm often solves a significant part of the problem at these early stages.

3. Results and Discussions

In a fully connected graph, the proposed algorithm effectively establishes the V_D and V_N sets within a single iteration. The size of the V_D set, which corresponds to the number of set elements, as well as the number of iterations, plays a pivotal role in the algorithm's performance. The duration of each iteration is predominantly influenced by the number of neighbors associated with the node newly incorporated into the V_D set.

Consequently, as nodes with a greater number of neighbors are typically chosen in the initial iterations, the processing time for these initial phases tends to be more prolonged.

In the context of exceptionally large graphs, the algorithm's time complexity assumes a pivotal role. This complexity can exhibit variation contingent upon the unique structural attributes of the graph and the distribution of its edges. Since every node and edge is subject to processing, the algorithm's time complexity is inherently dependent on the specific characteristics of the graph under consideration.

The time complexity associated with the calculation of centralities predominantly stems from the necessity to iterate over all nodes (n iterations) and, within each iteration, to iterate over the neighboring nodes (avg_degree). The exact time complexity is contingent on the distinctive properties inherent to the graph. Consequently, the average time complexity for each iteration pertaining to each Malatya centrality metric can be approximated as $O(|VD| * n * avg\_degree)$. In cases where there is no alteration in the graph structure, the algorithm yields consistent results across multiple applications, devoid of any inherent randomness.

When assessing algorithmic computational costs, a thorough analysis takes into account critical elements such as the selected data structures, procedural sequences, and the inherent intricacies of the algorithm's core components. In the pursuit of our primary research objective, which centers on enhancing performance and achieving superior outcomes, the proposed algorithms naturally evolve towards greater complexity. This evolution leads to an increase in computational overhead that is commensurate with the sophistication of the algorithms.

It is worth noting that the calculation of TMC values imposes a higher computational demand compared to SMC values. In the case of SMC computations, the process entails the utilization of FMC values for the neighbors of each node, with the computational intensity contingent upon the number of neighbors associated with each node. Conversely, the complexity of Malatya Centrality exhibits a pronounced correlation with the number of edges present in the graph.

In contrast, the calculation of TMC values involves the consideration of nodes that are adjacent to the neighbors of a given node, resulting in more intricate computational processing and necessitating interaction with a greater number of nodes. Consequently, the complexity of TMC is significantly heightened. It is imperative to highlight that the computational cost of the algorithm tends to be elevated in graphs characterized by a substantial node count, a limited number of edges, and a reduced network diameter.

3.1. Results with Synthetic Graphs

In this section, graphs with approximately known dominance numbers were created and the proposed method was tested. The algorithm in (Sanchis, 2002) was used to generate a graph with a certain number of nodes, density and dominance number. The graph generation algorithm steps are as follows: The set V was created. The total

number of nodes is equal to n in V. To create the graph, V nodes were divided into d subsets: $V_1, V_2, \ldots\ldots, V_d$. Each subset $V_1, V_2, \ldots\ldots, V_d$ represents the dominance number of the graph. For each i, x_i and y_i were selected from the nodes in the set V_i. These nodes will represent the edges of the graph to be created with x_i and y_i. For each i, edges with x_i were added to all other nodes in V_i. This step helps identify dominant nodes by connecting nodes in each subset V_i with x_i. Finally, for each i, we ensure that y_i is not connected to any node other than V_i. If not, additional edges are added to achieve the desired density.

The graph's dominance number precisely corresponds to the value denoted as 'd'. Furthermore, to introduce an element of uncertainty and complexity, a controlled level of noise was introduced by randomly adding edges to the generated graph. This was done with the intent of enhancing variability in the domination number across the graph, thereby facilitating a comprehensive evaluation of the applied methodologies. The noise ratio was deliberately set at 0.2. This algorithm holds potential utility within the realm of graph theory applications, particularly when the objective is to modulate crucial properties such as the dominance number and network density.

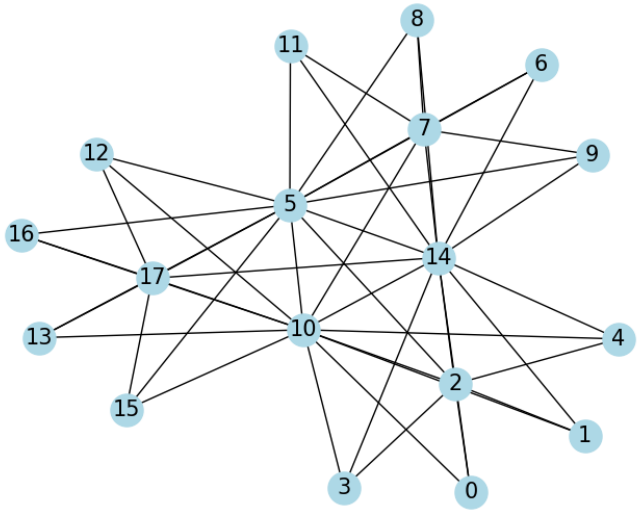


Figure 6. A sample generated graph

Figure 6 illustrates a graph characterized by specific parameters, namely $n = 18$, $p = 0.01$, and $d = 3$, in the absence of any noise. It is established that the domination number in this instance is 3. This setup served as the basis for testing the efficacy of the various methods on graphs created with differing values of n, p, and d. The findings, summarized in Table 1, encompass the results obtained from graphs generated under diverse parameter settings. It is notable that, in accordance with the results, the value of d remained relatively high; however, it's worth noting that introducing noise into the graphs tends to diminish this parameter.

Our proposed method's results were subjected to a comparative analysis against values generated through several established approaches, including the maximum degree, PageRank, a networkx library function, and minimum weighted method. The maximum node degree and PageRank methodologies entail the iterative selection of nodes with the highest centrality values, subsequently incorporating them into the V_D set, along with the addition of their neighbors to the V_N set. The dominating_set function employed in the Python networkx library is based on the 7th algorithm described in (Esfahanian, 2013). In practice, node selection is random. In the method called minimum weight dominating set proposed by Vazirani (Vazirani, 2003), the cost of selecting a node is calculated. This selection cost is inversely proportional to the weight of the node and the number of nodes dominated by its neighbors. At each step, the iteration selects the most cost-efficient among the non-dominated nodes and adds them to the V_D set.

Upon analyzing the outcomes of the techniques applied to the generated graphs, it became evident that Min_weighted, FMC, SMC, and TMC produced comparable results. Notably, the most favorable outcomes were consistently achieved with SMC and TMC, both of which yielded highly similar results. Furthermore, our methods exhibited superior performance, particularly on networks characterized by high density.

Table 1. Results of methods for generated graphs

n	p	d	Max degree	Pagerank	Networkx	Min Weighted	FMC	SMC	TMC
100	0.2	5	8.3	9.3	13.6	**5.0**	7.7	6.9	7.1
400	0.2	20	15.9	15.4	17.4	11.5	12.0	**11.2**	11.6
800	0.2	40	17.6	18.4	20.6	14.0	14.3	**13.2**	13.3
1000	0.2	50	19.2	19.0	22.0	14.7	15.3	14.5	**14.4**
1500	0.2	75	19.9	20.9	23.2	16.0	16.2	15.7	**15.5**
2000	0.2	100	21.7	20.8	23.9	16.8	16.8	**16.3**	16.5
100	0.3	5	7.2	7.0	9.4	**5.2**	5.9	5.4	5.9
400	0.3	20	10.7	10.6	12.9	8.7	**8.4**	8.5	**8.4**
800	0.3	40	12.6	12.8	14.2	10.0	10.3	9.9	**9.8**
1000	0.3	50	13.1	13.1	14.5	10.3	10.4	**10.1**	10.6
1500	0.3	75	14.0	14.2	16.0	11.4	11.9	**11.3**	11.4
2000	0.3	100	15.1	15.2	16.6	12.3	12.1	**11.6**	11.7

3.2. Results with Erdos-Renyi Graphs

In order to better understand the behavior of the proposed method, the model was tested on large-scale, dense datasets produced with Erdös-Renyi. The Erdös-Renyi graph generator creates a graph by randomly connecting nodes according to the specified number of nodes and probability. Each edge is included in the graph with probability p, independent of the other edges. The probability p determines the density of the graph (Erdös & Rényi, 1959).

Understanding the characteristics and structure of a network requires a grasp of fundamental network properties. Table 2 presents an array of network properties pertaining to graphs generated using the Erdős-Rényi model, which prove invaluable in comprehending how the proposed methodology behaves within diverse network contexts. These network attributes offer insights into the algorithm's performance in terms of time complexity, computational costs, and the challenges posed by attaining optimal results. Among these properties, the number of nodes and edges assumes paramount importance, with the former signifying the network's overall size. The density quantifies the ratio of actual edges between nodes to the maximum potential edges in the network, providing an indicator of network compactness. The average degree shows how well-connected the network is by indicating how many edges a node has on average. The network diameter measures the shortest path between the most distant nodes, affecting how easily and quickly information can travel. The clustering coefficient reflects how likely nodes are to form tightly knit groups, and the average clustering coefficient gives an overall measure of this tendency across the network. Together, these properties impact the algorithm's performance and results.

As the quantity of edges diminishes, so too do the interconnections between nodes. Consequently, there are fewer nodes available for inclusion in the V_N set at any given iteration. This scenario necessitates a greater number of algorithmic iterations and traversal of a larger number of nodes.

Table 2. Properties for Erdös-Renyi Graphs with different edges

n	m	p	Average degree (m / n)	Density	Diameter	Average Clustering	Time (minute)
1000	499500	1	499.5	1	1	1	0.05721
1000	99375	1/5	99.375	0.1989	2	0.1990	0.05376
1000	50103	1/10	50.103	0.1003	3	0.1004	0.05981
1000	24874	1/20	24.874	0.0498	3	0.0500	0.05664
1000	10082	1/50	10.082	0.0202	4	0.0202	0.06851
1000	6262	1/80	6.262	0.0125	5	0.0130	0.08877
2000	1999000	1	999.5	1	1	1	0.25643
2000	399622	1/5	199.811	0.1999	2	0.1999	0.46925
2000	199912	1/10	99.956	0.1000	2	0.0999	0.47112
2000	100102	1/20	50.051	0.0501	3	0.0502	0.48024
2000	39913	1/50	19.956	0.0200	3	0.0202	0.50723
2000	24792	1/80	12.396	0.0124	4	0.0123	0.58904

In order to comprehend the time complexity of the MMDS algorithm, we conducted experiments utilizing Erdős-Rényi graphs containing 1000 and 2000 nodes while varying the number of edges. We meticulously

recorded the computational time required to execute these calculations. It is evident from Table 2 that both the number of nodes and edges significantly influence the computational overhead of the algorithm. Specifically, as observed in the table, a decrease in the number of edges, while keeping the number of nodes constant, results in an increase in processing costs. Furthermore, we observed that reductions in the average number of connections, graph density, diameter, and average clustering coefficient correspond to an elevated computational cost.

While density is recognized as an insufficient determinant for assessing MDS instances, it is generally expected that dense graphs tend to exhibit fewer MDS domination number, whereas sparse graphs are more likely to yield a greater number of MDS domination number. Each graph possesses unique characteristics that may influence this aspect, necessitating a separate analysis of the MDS problem for each specific graph. The structure of the graph, its density, and the size of the MDS are intricately linked, albeit devoid of a universal guiding principle. Notably, density emerges as a pivotal parameter governing the prevalence of dominating sets within the generated datasets. As depicted in Figure 7, under a consistent number of nodes, an increase in the number of edges leads to a decrease in the count of MDS domination number.

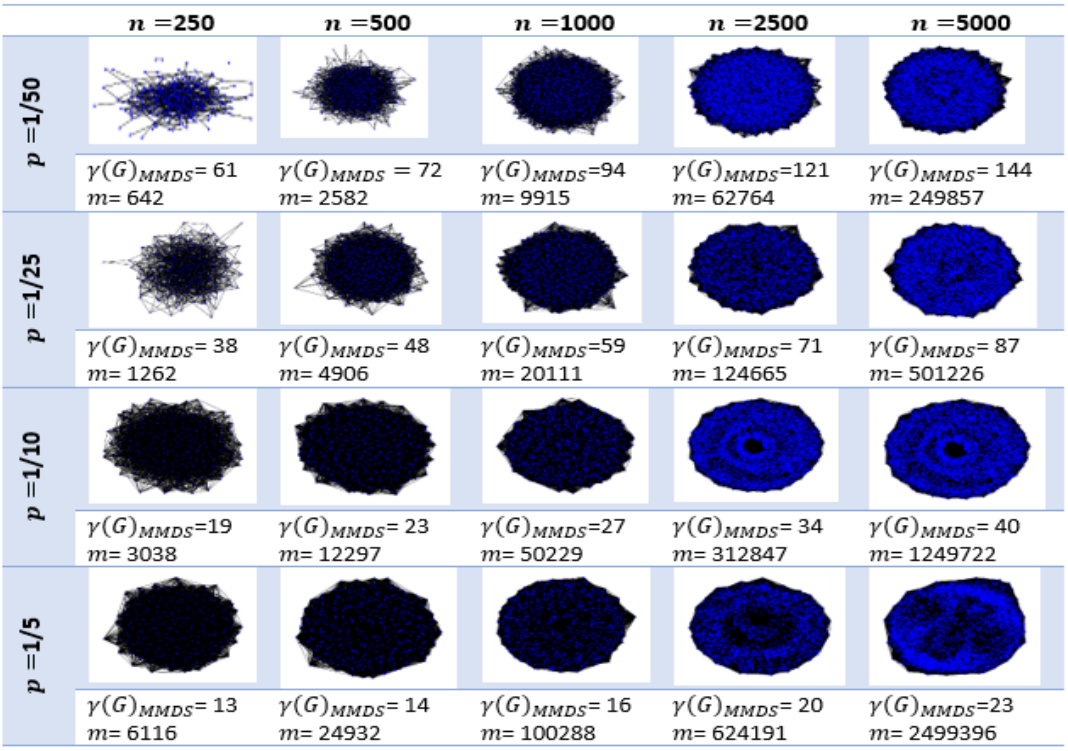


Figure 7. MMDS with Erdös-Renyi graphs

3.3. Results with Real Datasets

In this section of the study, results were obtained on real data sets (Aggarwal et al., 2014)(Kunegis, 2013). The characteristics of the graphs are given in Table 3.

The algorithm operates by selecting nodes based on their TMC values, affording priority to those nodes with the highest TMC values. In the initial stages of the algorithm, as the high-value nodes are incorporated into the V_D set and their neighboring nodes are added to the V_N set, there is a rapid expansion in the number of elements within the V_N set. Consequently, nodes characterized by both high dominance and clustering properties are typically selected during the initial iterations. In contrast, nodes appended to the dominating set towards the conclusion of the algorithm exhibit lower clustering characteristics. In scenarios where time constraints are pertinent, it is feasible to terminate the iterations prematurely to swiftly identify the most dominant nodes. While this may not encompass all nodes as in the case of a MDS, it ensures the inclusion of the most pivotal set elements. For instance, this approach can be effectively employed to address tasks such as identifying the 'k' most dominant elements. Importantly, the algorithm commences its iterative process from the node of utmost dominance.

Table 3. Properties of graphs

	Node numbers (n)	Edge numbers (m)	Average degree (m / n)	Density	Diameter	Average Clustering
Karate	34	78	2.29	0.139037	5	0.5706
Dolphins	62	159	2.56	0.084082	8	0.2590
Lesmiserable	77	254	3.30	0.086808	5	0.5731
Adjnoun	112	425	3.80	0.068372	5	0.1728
Football	115	613	5.33	0.093516	4	0.4032
Netscience	1589	2742	1.73	0.002173	17	0.6378
ego-Facebook	4039	88234	21.85	0.010819	8	0.6056
Power	4941	6594	1.33	0.000540	46	0.0801
Hep-th	8361	15751	1.88	0.000451	19	0.4420
As-22july06	22963	48436	2.11	0.000184	11	0.2304
Email-Enron	36692	183831	5.01	0.000273	11	0.4970

When the Figure 8 is examined, it is seen that the proposed method generally obtains smaller values.

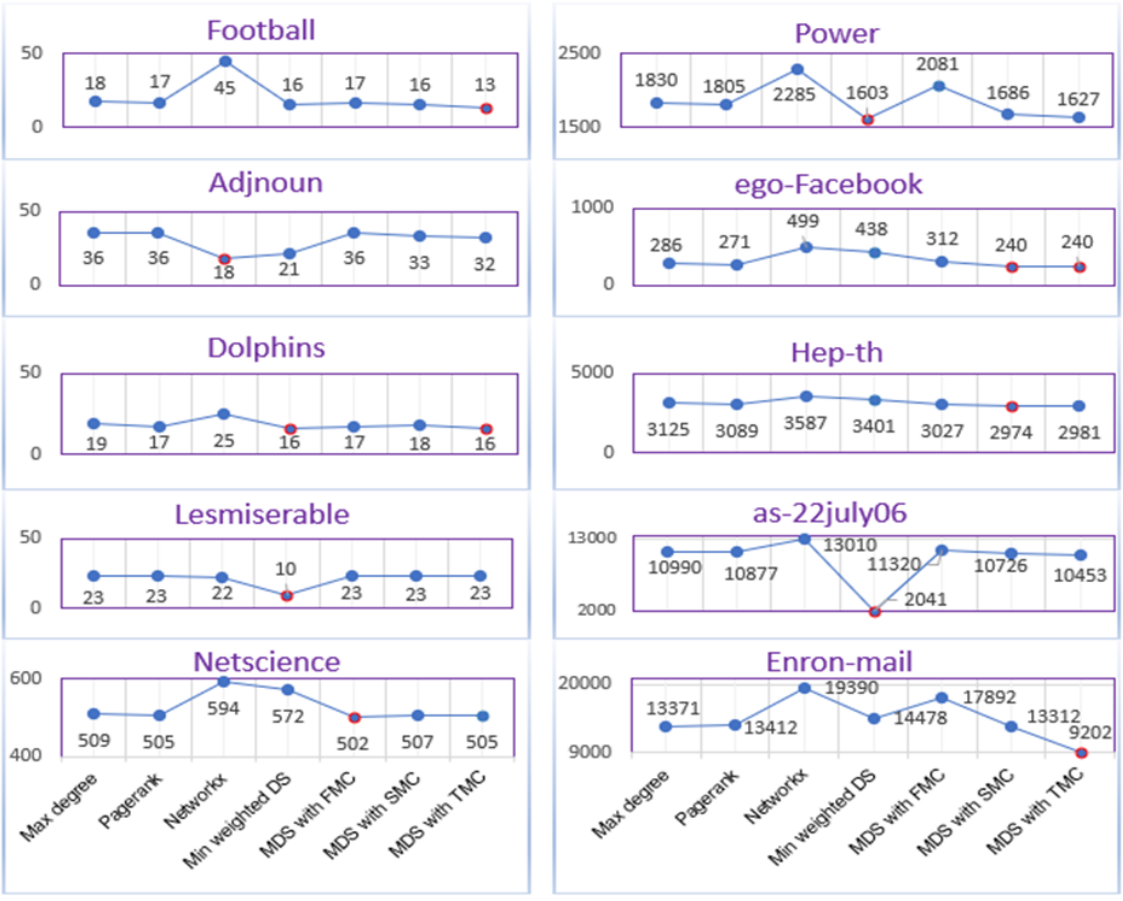


Figure 8. Domination numbers with real dataset

When we analyze the as-22july06 dataset, our method consistently yields solutions in regions characterized by high node connectivity with a notably small number of set elements. In a network structure featuring 48,436 edges, the majority of nodes, save for a limited subset, are reachable with the inclusion of approximately 1,400 nodes into the V_D set. This quantity increases progressively as we augment the algorithm by incorporating nodes lacking direct connections into the V_D set. Thus, when the objective is to identify the initial 1,400 most dominant nodes, it is indeed feasible to efficiently span a significant portion of the network while keeping the count of dominant set elements at a minimum. By employing this judicious approach, a substantial portion of the network's intended

tasks can be effectively achieved. Ultimately, our method provides a solution that optimizes the trade-off between computational costs and benefits.

4. Conclusions

This study introduces an efficient algorithm designed to deliver approximate solutions for the Minimum Dominating Set (MDS) problem. The approach herein leverages a three-step computation of Malatya centrality values, initiating from node degrees and neighbor node degrees, all while eliminating the element of randomness. The algorithm's accuracy and efficacy are rigorously examined through comparative evaluations across various datasets and methodologies. Upon reviewing the results, it is evident that the proposed algorithm consistently generates a robust solution set that remains resilient under diverse constraints. These outcomes underscore the method's effectiveness. Although the algorithm may incur higher time costs in sparse networks, its ability to produce near-optimal results affirms its practical value and effectiveness.

When selecting the most suitable algorithm for the efficient computation of the MDS, it is imperative to consider both graph properties and the time required to identify such a set. Throughout the quest for a solution, key factors that influence time complexity include the graph's connectivity density and the clustering characteristics of its nodes. Notably, the algorithm demonstrates expedited performance in the context of dense graphs, thus mitigating issues pertaining to memory constraints.

Building upon the current findings, future research can explore strategies to reduce the algorithm's time complexity, particularly in sparse and large-scale networks. This may involve the integration of parallel computing techniques or the development of hybrid methods that combine Malatya centrality with other heuristic or machine learning-based approaches to accelerate node selection. Additionally, adapting the algorithm for dynamic networks could significantly enhance its applicability to real-world scenarios such as social media, transportation, and communication systems. Investigating the trade-offs between solution quality and computational cost in resource-constrained environments also remains a promising direction. Lastly, further analytical work can be conducted to formalize performance bounds and deepen the theoretical understanding of the algorithm’s behavior across different graph topologies.

References

Abed, S. A., & Rais, H. M. (2017). Hybrid bat algorithm for minimum dominating set problem. *Journal of Intelligent & Fuzzy Systems*, *33*(4), 2329–2339. https://doi.org/10.3233/JIFS-17398

Aggarwal, C., Subbian, K., Butler, K., Stephens, M., Stephens, M., Chakrabarti, D., Kumar, R., Tomkins, A., Clauset, A., Moore, C., Newman, M. E. J., Csardi, G., Nepusz, T., Decelle, A., Krzakala, F., Moore, C., Zdeborov??, L., Eisinga, R., Te Grotenhuis, M., … Cov, E. R. (2014). {SNAP Datasets}: {Stanford} Large Network Dataset Collection. *Physical Review Letters*, *Complex Sy*(1).

Albuquerque, M., & Vidal, T. (2018). An efficient matheuristic for the minimum-weight dominating set problem. *Applied Soft Computing Journal*, *72*. https://doi.org/10.1016/j.asoc.2018.06.052

Batool, K., & Niazi, M. A. (2014). Towards a methodology for validation of centrality measures in complex networks. *PLoS ONE*, *9*(4). https://doi.org/10.1371/journal.pone.0090283

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, *30*(1–7), 107–117. https://doi.org/10.1016/S0169-7552(98)00110-X

Bujtás, C., & Klavžar, S. (2016). Improved Upper Bounds on the Domination Number of Graphs With Minimum Degree at Least Five. *Graphs and Combinatorics*, *32*(2), 511–519. https://doi.org/10.1007/s00373-015-1585-7

Casado, A., Bermudo, S., López-Sánchez, A. D., & Sánchez-Oro, J. (2023). An iterated greedy algorithm for finding the minimum dominating set in graphs. *Mathematics and Computers in Simulation*, *207*. https://doi.org/10.1016/j.matcom.2022.12.018

Chalupa, D. (2018). An order-based algorithm for minimum dominating set with application in graph mining. *Information Sciences*, *426*. https://doi.org/10.1016/j.ins.2017.10.033

Connolly, S., Gabor, Z., Godbole, A., Kay, B., & Kelly, T. (2016). Bounds on the maximum number of minimum dominating sets. *Discrete Mathematics*, *339*(5). https://doi.org/10.1016/j.disc.2015.12.030

Erdös, P., & Rényi, A. (1959). On random graphs. *Publicationes Mathematicae*, *6*, 290–297. https://doi.org/10.2307/1999405

Esfahanian, A. (2013). Connectivity algorithms. In Topics in structural graph theory (pp. 268–281). Cambridge University Press. https://www.cse.msu.edu/~cse835/Papers/Graph_connectivity_revised.pdf

Estrada, E. (2011). The Structure of Complex Networks. In *SIAM Review* (Vol. 45, Issue 2). https://doi.org/10.1093/acprof:oso/9780199591756.001.0001

Gao, C., Lan, X., Zhang, X., & Deng, Y. (2013). A Bio-Inspired Methodology of Identifying Influential Nodes in Complex Networks. *PLoS ONE*, *8*(6). https://doi.org/10.1371/journal.pone.0066732

Hedar, A. R., & Ismail, R. (2010). Hybrid genetic algorithm for minimum dominating set problem. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6019 LNCS*(PART 4). https://doi.org/10.1007/978-3-642-12189-0_40

Hernández Mira, F. Á., Parra Inza, E., Sigarreta Almira, J. M., & Vakhania, N. (2022). A polynomial-time approximation to a minimum dominating set in a graph. *Theoretical Computer Science*, *930*, 142–156. https://doi.org/10.1016/j.tcs.2022.07.020

Hu, J., Du, Y., Mo, H., Wei, D., & Deng, Y. (2016). A modified weighted TOPSIS to identify influential nodes in complex networks. *Physica A: Statistical Mechanics and Its Applications*, *444*, 73–85. https://doi.org/10.1016/j.physa.2015.09.028

Jovanovic, R., & Tuba, M. (2013). Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem. *Computer Science and Information Systems*, *10*(1). https://doi.org/10.2298/CSIS110927038J

Kapoor, K., Sharma, D., & Srivastava, J. (2013). Weighted node degree centrality for hypergraphs. *Proceedings of the 2013 IEEE 2nd International Network Science Workshop, NSW 2013*. https://doi.org/10.1109/NSW.2013.6609212

Karci, A. (2020). New Algorithms for Minimum Dominating Set in Any Graphs. *Computer Science*, *5*(2).

Karcı, A., Yakut, S., & Öztemiz, F. (2022). A New Approach Based on Centrality Value in Solving the Minimum Vertex Cover Problem: Malatya Centrality Algorithm. *Computer Science*. https://doi.org/10.53070/bbd.1195501

Kunegis, J. (2013). KONECT - The koblenz network collection. *WWW 2013 Companion - Proceedings of the 22nd International Conference on World Wide Web*.

Li, J., Potru, R., & Shahrokhi, F. (2020). A Performance Study of Some Approximation Algorithms for Computing a Small Dominating Set in a Graph. *Algorithms*, *13*(12), 339. https://doi.org/10.3390/a13120339

Newman, M. E. J. (2003). The Structure and Function of Complex Networks. *SIAM Review*, *45*(2), 167–256.

Nguyen, M. H., Hà, M. H., Nguyen, D. N., & Tran, T. T. (2020). Solving the k-dominating set problem on very large-scale networks. *Computational Social Networks*, *7*(1). https://doi.org/10.1186/s40649-020-00078-5

Okumuş, F., & Karcı, Ş. (2024). MDSA: A Dynamic and Greedy Approach to Solve the Minimum Dominating Set Problem. *Applied Sciences*, *14*(20), 9251. https://doi.org/10.3390/app14209251

Ore, O. (1962). Theory of Graphs. American Mathematical Society Colloquium Publications.

Page, L., Brin, S., Motwani, R., & Winograd, T. (1998). The PageRank Citation Ranking: Bringing Order to the Web. *World Wide Web Internet And Web Information Systems*, *54*(1999–66), 1–17. https://doi.org/10.1.1.31.1768

Potluri, A., & Singh, A. (2013). Hybrid metaheuristic algorithms for minimum weight dominating set. *Applied Soft Computing Journal*, *13*(1). https://doi.org/10.1016/j.asoc.2012.07.009

Potluri, A., & Singh, A. (2011). Two hybrid meta-heuristic approaches for minimum dominating set problem. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7077 LNCS*(PART 2). https://doi.org/10.1007/978-3-642-27242-4_12

Sanchis, L. A. (2002). Experimental Analysis of Heuristic Algorithms for the Dominating Set Problem. *Algorithmica*, *33*(1), 3–18. https://doi.org/10.1007/s00453-001-0101-z

Truta, T. M., Campan, A., & Beckerich, M. (2020). Efficient Approximation Algorithms for Minimum Dominating Sets in Social Networks. In *Research Anthology on Artificial Intelligence Applications in Security* (pp. 1120–1153). IGI Global. https://doi.org/10.4018/978-1-7998-7705-9.ch052

Tuğal, İ., & Karcı, A. (2019). Comparisons of Karcı and Shannon entropies and their effects on centrality of social networks. *Physica A: Statistical Mechanics and Its Applications*, *523*, 352–363. https://doi.org/10.1016/j.physa.2019.02.026

Tuğal, İhsan, & Karcı, A. (2020). Determination of Influential Countries by Cultural and Geographical Parameters. *Anemon Muş Alparslan University Social Sciences Journal*. https://doi.org/10.18506/anemon.563211

Vazirani, V. V. (2003). Approximation Algorithms. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-04565-7

Wang, F., Du, H., Camacho, E., Xu, K., Lee, W., Shi, Y., & Shan, S. (2011). On positive influence dominating sets in social networks. *Theoretical Computer Science*, *412*(3), 265–269. https://doi.org/10.1016/j.tcs.2009.10.001

Yakut, S., Öztemiz, F., & Karci, A. (2023). A new robust approach to solve minimum vertex cover problem: Malatya vertex-cover algorithm. *The Journal of Supercomputing*. https://doi.org/10.1007/s11227-023-05397-8

Zhou, Y., Lv, G., Xiu, B., Zhang, W., & Cheng, Q. (2014). A faster approximate method to identify minimum dominating set. *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*. https://doi.org/10.1109/ICSESS.2014.6933601