T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

ROMAN DOMINATION NUMBER

MEHMET ACAR

SUPERVISOR
PROF. DR. DIDEM GÖZÜPEK

GEBZE
2023

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

ROMAN DOMINATION NUMBER

MEHMET ACAR

SUPERVISOR
PROF. DR. DIDEM GÖZÜPEK

2023
GEBZE

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 31/08/2021 by the following jury.

JURY

Member
(Supervisor) : Prof. Dr. Didem Gözüpek

Member : Prof. Dr. Didem Gözüpek

Member : Dr. Zafeirakis Zafeirakopoulos

ABSTRACT

In this paper, Roman Domination problem on graphs are considered. Given problem is NP hard, as a heuristic approach, Variable Neighborhood Search (VNS) method is presented. Also, Brute Force method is presented in order to compare with VNS.

Firstly, both VNS and Brute Force methods are applied to the single input graph in order to find Roman Domination Number.

Then, VNS and Brute Force methods are applied to the different input graphs which have different edge or vertex number.

Finally, time complexities of VNS and Brute Force algorithms are compared.

Keywords: keyword1, keyword2.

ÖZET

In this paper, Roman Domination problem on graphs are considered. Given problem is NP hard, as a heuristic approach, Variable Neighborhood Search (VNS) method is presented. Also, Brute Force method is presented in order to compare with VNS.

Firstly, both VNS and Brute Force methods are applied to the single input graph in order to find Roman Domination Number.

Then, VNS and Brute Force methods are applied to the different input graphs which have different edge or vertex number.

Finally, time complexities of VNS and Brute Force algorithms are compared.

Anahtar Kelimeler: anahtar kelime1, anahtar kelime2.

ACKNOWLEDGEMENT

Thank you mate!!

Name Surname

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol or Abbreviation	:	**Explanation**
ABC	:	The first three letters in the English alphabet
μ	:	Small mu

CONTENTS

Abstract iv

Özet v

Acknowledgement vi

List of Symbols and Abbreviations vii

Contents ix

List of Figures x

List of Tables xi

1 Introduction 1

2 VNS for Single Graph 3
2.1 Definiton of VNS 3
2.2 VNS on Input Graph 3

3 Brute Force for Single Graph 5
3.1 Definiton of Brute Force 5
3.2 Brute Force on Input Graph 5

4 Observation of Multiple Graphs 6
4.1 Different Edge Numbered Graphs 6
4.2 Different Vertex Numbered Graphs 7

5 Time Complexities 9
5.1 Variable Neighborhood Search Time Complexity 9
5.2 Brute Force Time Complexity 9

6 Conclusions 10

Bibliography 11

CV 12

Appendices **13**

LIST OF FIGURES

1.1 Graph G = (V, E) 1
1.2 Illustrated solution of the RD problem on a graph G defined in the Figure 1.1 2

2.1 Variable Neighborhood Search Metaheuristic 3
2.2 Roman Domination Number in Graph After VNS applying 4

3.1 Roman Domination Number in Graph After Brute Force applying . . 5

4.1 Roman Domination Number Change With Same Vertex Number and Different Edge Number 6
4.2 Roman Domination Number Change With Same Edge Number and Different Vertex Number 7

LIST OF TABLES

1. INTRODUCTION

About 1700 years ago, the Roman Empire was under attack, and Emperor Constantine had to decide where to station his four field army units to protect eight regions. His trick was to place the army units so that every region was either secured by its own army (one or two units) or was securable by a neighbor with two army units, one of which can be sent to the undefended region directly if a conflict breaks out.

Any province of the Roman Empire is considered to be safe if there is at least one legion (of maximum 2) stationed within it, the Roman Domination problem requires that every unsafe province must be adjacent to a province with two legions stationed within it and the total number of stationed legions within all provinces of the Roman Empire is minimal.

A Roman dominating function of G is a function f : V $\rightarrow \{0,1,2\}$ such that every vertex u with f(u) = 0 is adjacent to a vertex v with f(v) = 2. The weight of a Roman dominating function f is sum of values assigned to all vertices. The minimum weight of a Roman dominating function of a graph G is the Roman domination number of G, written Γr(G).

Let us assume that the Roman Empire can be described by a graph G = (V, E) as it is presented below, in Figure 1.1.

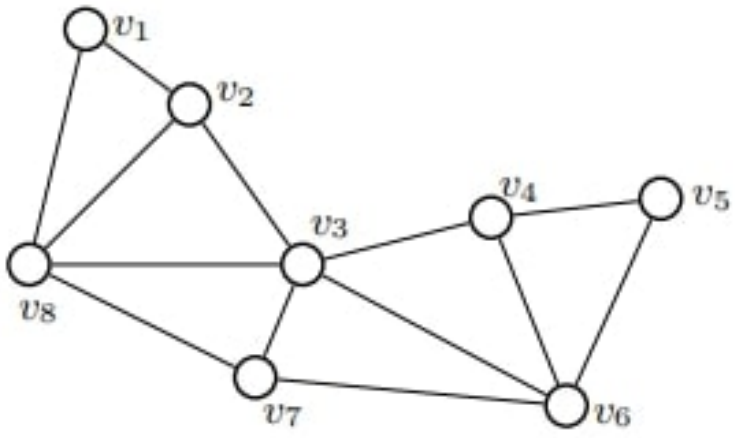

Figure 1.1: Graph G = (V, E)

The optimal number of legions necessary to defend the given graph is 4, provinces represented by vertices v1 and v5 are with one stationed legion, province represented by vertex v3 is with two stationed legions and all other provinces are without stationed legions. With the given schedule, vertices v1, v3 and v5 are defended because they have

at least one legion stationed within it, while v2, v4, v6, v7 and v8 are defended since they are in the neighborhood of the vertex v3, which is with two stationed legions. The optimal solution to the proposed problem is illustrated in Figure 1.2, where vertices are marked by black squares if they are representing provinces with two stationed legions, marked by red circles if they are representing provinces with one stationed legion, and marked by white circles if they are representing provinces without stationed legions.

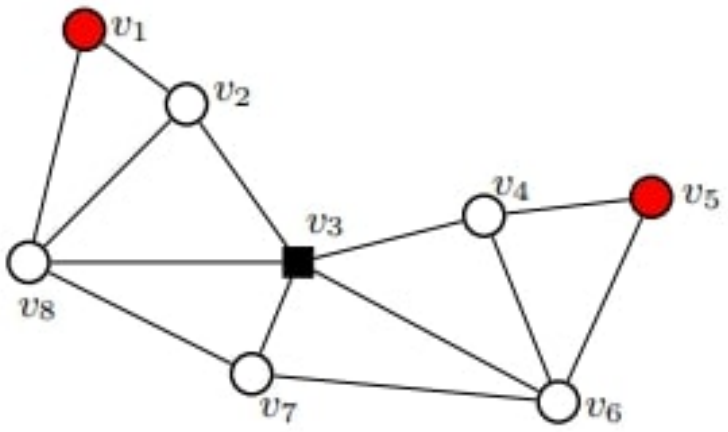

Figure 1.2: Illustrated solution of the RD problem on a graph G defined in the Figure 1.1

2. VNS FOR SINGLE GRAPH

2.1. Definiton of VNS

Variable neighborhood search (VNS) is a metaheuristic method for solving a set of combinatorial optimization and global optimization problems. It explores distant neighborhoods of the current incumbent solution, and moves from there to a new one if and only if an improvement was made. The local search method is applied repeatedly to get from solutions in the neighborhood to local optima. Pseudocode of VNS algorithm is given in Figure 2.1.

Algorithm 1 Variable Neighborhood Search metaheuristic

```
1:  X* ← InitialSolution();
2:  F* ← F(X*);
3:  repeat
4:      k ← k_min;
5:      repeat
6:          X ← X*;
7:          X' ← Shake(X, k);
8:          X'' ← LocalSearch(X');
9:          if F(X'') < F* then
10:             F* ← F(X'');
11:             X* ← X'';
12:             k ← k_min;
13:         else
14:             k ← k + k_step;
15:     until k > k_max
16: until StoppingCondition()
```

Figure 2.1: Variable Neighborhood Search Metaheuristic

2.2. VNS on Input Graph

Undirected input graph with 5 vertexes is created and assigned number 0's to each vertexes. Then, several edges are created on the graph. Then, local search metaheuristic is applying to the graph. Finally, in output graph, vertexes which have number 0 are colored with blue, vertexes which have number 1 are colored with green and vertexes which have number 2 are colored with blue. Resulting output is like Figure 2.2.

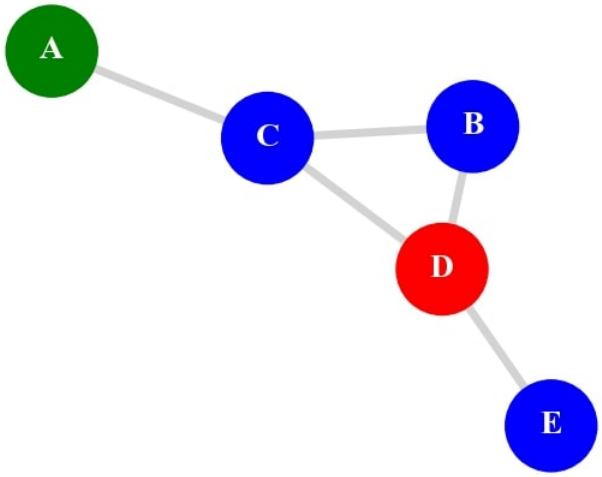

Figure 2.2: Roman Domination Number in Graph After VNS applying

3. BRUTE FORCE FOR SINGLE GRAPH

3.1. Definiton of Brute Force

A brute force algorithm solves a problem through exhaustion: it goes through all possible choices until a solution is found. In this part, all combinations of numbers that vertexes in the graph can have is examined. From these combinations which have feasible solutions in terms of Roman Domination graph, the graph which has minimum weight(Roman Domination Number) is found.

3.2. Brute Force on Input Graph

Undirected input graph is created with 5 vertexes. Then, several edges are created on the graph. Then, brute force algorithm is applying to the graph. Finally, in output graph, vertexes which have number 0 are colored with blue, vertexes which have number 1 are colored with green and vertexes which have number 2 are colored with blue. Resulting output is like Figure 3.1.

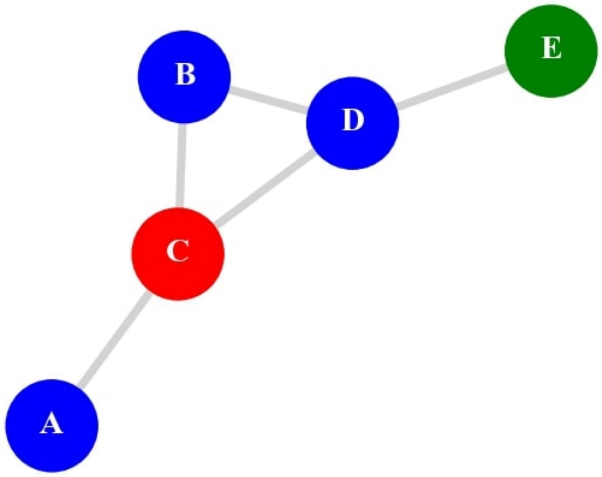

Figure 3.1: Roman Domination Number in Graph After Brute Force applying

4. OBSERVATION OF MULTIPLE GRAPHS

Undirected graph has vertexes and edges. From these two parameters, one parameter kept fixed and other parameter kept changeable in order to observe that how roman domination number on the graph changes. In order to observe that changement, different input graphs which have different vertex or edge number are created. Then, VNS and Brute Force algorithms are applied to each graph 30 times. Then, average roman domination number of 30 observation is taken and this value is equal to the Roman Domination Number of each graph and comparison of VNS and and Brute Force Algorithm in terms of Roman Domination Number is printed as a figure.

4.1. Different Edge Numbered Graphs

In this part, node number is kept fixed and edge number is changed in input graphs. All input graph has 6 vertexes. The graph with the lowest number of edges from the input graphs has 5 edges and the graph with the highest number of edges from the input graphs has 15 edges. Then, VNS and Brute Force algorithms are applied to each graph 30 times. Then, average of 30 observation is taken and this value is equal to the Roman Domination Number of each graph and comparison of VNS and and Brute Force Algorithm in terms of Roman Domination Number is printed as a figure 4.1.

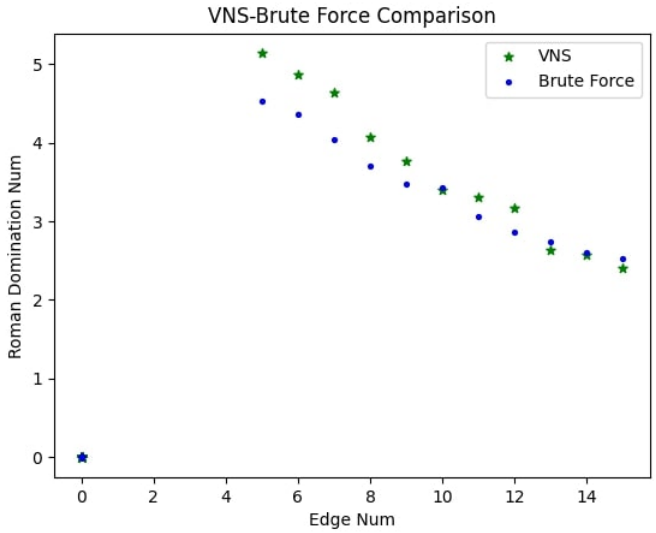

Figure 4.1: Roman Domination Number Change With Same Vertex Number and Different Edge Number

According to Figure 4.1, if number of edge is increasing on input graphs, roman domination number will be decreasing. Generally, this situation is occurred. But, rarely roman domination number can be increase while number of edge is increasing on input graphs. In Variable Neighborhood Search (VNS) method, this event is caused by VNS does not guarentee the optimal solution or position of the added edge. On the other hand, in Brute Force method, this event is caused by only position of the added edge.

4.2. Different Vertex Numbered Graphs

In this part, edge number is kept fixed and vertex number is changed in input graphs. All input graph has 5 edges. The graph with the lowest number of vertexes from the input graphs has 5 vertexes and the graph with the highest number of vertexes from the input graphs has 15 vertexes. Then, VNS and Brute Force algorithms are applied to each graph 30 times. Then, average of 30 observation is taken and this value is equal to the Roman Domination Number of each graph and comparison of VNS and and Brute Force Algorithm in terms of Roman Domination Number is printed as a figure 4.2.

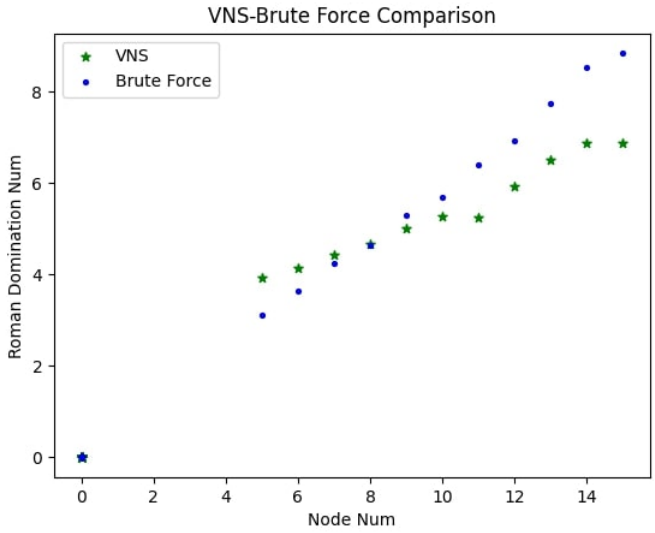

Figure 4.2: Roman Domination Number Change With Same Edge Number and Different Vertex Number

According to Figure 4.2, if number of vertex is increasing on input graphs, roman domination number will be increasing. In Brute Force method, this situation is occurred every time. But in Variable Neighborhood Search (VNS) method, rarely

roman domination number can be decrease while number of vertex is increasing on input graphs. In Variable Neighborhood Search (VNS) method, this event is caused by VNS does not guarentee the optimal solution.

5. TIME COMPLEXITIES

5.1. Variable Neighborhood Search Time Complexity

Time complexity of Variable Neighborhood Search (VNS) method which is applying for finding Roman Domination Number of graph, is equal to:

$$O(n^4)$$

In time complexity, n is a input graph's vertex number. And 4 is coming from nested loops.

5.2. Brute Force Time Complexity

Time complexity of Brute method which is applying for finding Roman Domination Number of graph, is equal to:

$$O(3^n)$$

Each vertex in input graph can take the number 0 or 1 or 2. So, each vertex can take one of 3 numbers.So, in time complexity, 3 is coming from here. And in time complexity, n is a input graph's vertex number.

6. CONCLUSIONS

Variable Neighborhood Search algorithm on input graph does not give an optimal solution every time as a Roman Domination Number but Brute Force algorithm gives an optimal solution every time. Because it goes through all possible choices until a solution is found.

If number of edge is increasing on input graph, roman domination number will be decreasing generally. But rarely, roman domination number can increase in this situation on VNS and Brute Force methods. In Variable Neighborhood Search (VNS) method, this event is caused by VNS does not guarentee the optimal solution or position of the added edge. On the other hand, in Brute Force method, this event is caused by only position of the added edge.

If number of nodes is increasing on input graph, roman domination number will be increasing all the time in Brute Force method. But rarely, roman domination number can decrease in this situation on VNS method caused by VNS does not guarentee the optimal solution.

Brute Force algorithm is slower than Variable Neighborhood Search algorithm.

BIBLIOGRAPHY

[1] M. Ivanovic, “Variable neighborhood search approach for solving roman and weak roman domination problems on graphs,” vol. 38, pp. 57–84, 2019.

[2] V.Filipovic, “Solving the signed roman domination and signed total roman domination problems with exact and heuristic methods,” pp. 1–32, 2022.

[3] E. J. Cockayne, “Roman domination in graphs,” vol. 278, pp. 11–22, 2004.

[1] [2] [3]

CV

XXX.

APPENDICES

Appendix 1: Some publications

No one significant, in AAA.

Appendix 2: Some explanation

None needed mate!