Republic of Iraq

Ministry of Higher Education & Scientific Research

University of Babylon

College of Education for Pure Sciences

Department of Mathematics

Numerical Optimization On Semidefinite Programming

A Dissertation

Submitted to the Council of College of Education for Pure Sciences,

Babylon University in Partial Fulfillment of the Requirements for

the Degree of Doctor of Philosophy in Education / Mathematics.

By

Abbas Musleh Salman Abbas

Supervised by

Assist. Dr. Ahmed Sabah Al-Jilawi

2022 A.D. **1444 A.H.**

بسم الله الرحمن الرحيم

وَلَقَدْ كَتَبْنَا فِي الزَّبُورِ مِنْ بَعْدِ الذِّكْرِ أَنَّ الْأَرْضَ يَرِثُهَا عِبَادِيَ الصَّالِحُونَ

صدق الله العظيم

سورة الأنبياء
الآية (١٠٥)

The Supervisor Certificate

I certify that this dissertation entitled " Numerical Optimization On Semidefinite Programming" was prepared by the student " Abbas Musleh Salman Abbas " has been prepared under my supervision in Babylon University College of Education for Pure Sciences as a partial fulfillment of the requirement the Degree of Doctor of Philosophy in Education /Mathematics.

Signature:
Name: Dr. Ahmed Sabah Al-Jilawi
Scientific Grade: Assistant Professor
Date: / / 2022

In view of the available recommendation, I forward this dissertation for debate by the examining committee.

Signature:
Name: Dr.Azal Jaafar Musa
Head of Mathematics Department
Date: / / 2022

Certification of Scientific Expert

I certify that I have read the scientific content of this dissertation " Numerical Optimization On Semidefinite Programming" and I have approved this dissertation is qualified for debate.

Signature:

Name: Dr.Ahmad Abbas AL-Adileee

Title: Professor

Date: / / 2022

Certification of Scientific Expert

I certify that I have read the scientific content of this dissertation " Numerical Optimization On Semidefinite Programming" and I have approved this dissertation is qualified for debate.

Signature:

Name: Dr.Basim Karim Kadhir

Title: Asst.Prof.

Date: / / 2022

Certification of Linguistic Expert

I certify that I have read this dissertation entitled " Numerical Optimization On Semidefinite Programming" and corrected its grammatical mistakes;therefor,it has qualified for debate.

Signature:

Name: Dr.Hussain Hamed Mayuuf

Title: Asst.Prof.

Date: / / 2022

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Table 1: **Notations used in the Thesis**

$f(x)$	The Objective function
$Minf(x)$	The Minimum of Objective Function
$Maxf(x)$	The Maximum of Objective Function
$g(x)$	The Inequality Constraint
$h(x)$	The Equality Constraint
IS	Independent Set
MIS	Maximum Independent Set
QP	Quadratic programming
S	The Convex Set
R	Set of Real Numbers
R^n	N-Dimensional Space
x^*	The Minimizer (Local, Global)
x^{k+1}	The Next Iterate
x^k	The Current Iterate
αk	The Step Length
$NLPP$	Non-Linear Programming Problems
KKT	Karush-Kuhn-Tucker
C	A Cone
L	Lagrangian Function
$P(x)$	Penalty Function
SDP	Semidefinite Programming Primal
SDD	Semidefinite programming Dualty
$SPSD$	Symmetric Positive Semidefinite

(MVC)	The Minimum Vertex Cover
VC	Vertex Cover
TSP	Traveling Salesman Problem
ILP	integer linear program
$QCQP$	Quadratically Constrained Quadratic programming
$SOCP$	Second Order Cone programming
CP	Cone programming
NP	Non Deterministic Polynomial Time
$fcalls$	function calls
AUG	Augmented
$COPs$	Combinatorial Optimization Problems
iff	if and only if
$\succ$	Symmetric Positive Definite
$\succeq$	Symmetric Positive Semidefinite

Dedication

The Seal of the Prophets and Messengers, Muhammad, may God bless him and his family and grant them peace.

To my father, may God have mercy on him

To the source of patience, optimism, hope and kindness, my mother, may God be pleased with him prolong her life.

To my supervisor who gives me scientific favor and support throughout my career.

To my wife who bore the burden of responsibility with me.

To the future flowers my children .

I offer this humble work.

Acknowledgements

In the name of Allah, the Most Gracious, the Most Merciful First and above all, Praise be to Allah who gave me the health and patience to conduct this work.

And then, I would like to thank prophet Mohammad and Ahel Al-Bate (peace be upon them).

I am particularly grateful for the assistance given by my supervisor, Asst. Prof. Ahmed Sabah Al-Jilawi, for his encouragement, guidance and support during this work.

I would also like to thank all the faculty and administrative staff and my teachers in the department of mathematics (University of Babylon, College of Education for Pure Sciences), who stand beside me and it was a pleasure to work with them to achieve the thesis and made it possible.

Deep appreciation and love to all my family and everyone who gave me a hand of support.

Abstract

The main objective of this dissertation is to locate a superior method that merges quicker of maximal independent set problem(MIS). In addition, develop a new algorithm technique to solve binary quadratic problems(MIS) with large scale variables (50,80,100 vertices) from biq Mac library. This a new technique of algorithm give us high efficiency, speed that exceeds the previously. One methodology is to the correlation between The Penalty and Augmented Lagrangian Methods. We have likewise built up the hypothetical combination properties of these methods In addition improved the theoretical convergence properties were improved to show that the new algorithm choice for solving the SDP relaxations of the binary quadratic problems. The users of the developed solver for the SDP relaxation will benefit by obtaining faster results. In this study, a new approach has been developed to replace the penalty method with the augmented Lagrangian method. Also, according to the value of the parameter , a hybrid method that switches between the two methods was developed called the combined method. The proposed three approaches of the augmented Lagrangian problem, and the penalty problem were studied for the linear programming (LP) problems. As a result, only two approaches were justified and approved as valid methods to be used for solving the SDP relaxations. we used Julia language in this study.

CHAPTER 1

RELATED WORKS

1.1 Introduction

Optimization is a primary mathematical method aimed at finding the value of variables that provide the minimum or maximum value for a mathematical function. The structure of optimization problems are divided into two types: the first type is continuous problems in which we have continuous variables, and the second type is discrete problems, which are also known as combinatorial optimization [68]. Optimization theory and methods is a young subject in applied mathematics, computational mathematics and operations research which has wide applications in science, engineering, business management, military and space technology. The subject is involved in optimal solution of problems which are defined mathematically, i.e., given a practical problem, the "best" solution to the problem can be found from lots of schemes by means of scientific methods and tools. It involves the study of optimality conditions of the problems, the construction of model problems, the determination of algorithmic method of solution, the establishment of convergence theory of the algorithms, and numerical experiments with typical problems and real life problems. Though optimization might date back to the very old extreme-value problems, it did not become an independent subject until the late 1940s, when G.B. Dantzig presented the well known simplex algorithm for linear programming. After the 1950s, when conjugate gradient methods and quasi-Newton methods were presented, the nonlinear programming developed greatly. Now various modern optimization methods can solve difficult and large scale optimization problems, and become an indispensable tool for solving problems in diverse fields. Optimization algorithms are a basic and efficient technique in mathematical programming, to arrive a solution, generally with the help of a computer . Optimization algorithms start with an first estimate of the value of the variables and by an iterative technique generates a sequence of get better estimates, or iterates, until an optimal solution is reached. A great algorithm should be accurate ,fast , efficient and robust. A good approximation of an optimal solution should be generated

([6], [48], [14]). In our line of research, we will focus on one of the most well-known subjects NP-hard combinatorial optimization problems such as the quadratic assignment problem and maximum independent set problems. Combinatorial Optimization by Cook, Cunningham, Pulley blank, and Schrijver [59]. gives a nice introduction to computational complexity and what it means for a combinatorial optimization problem to be defined as NP-hard. However, linear programming (LP) is vastly used in mathematics and also can be applied to diverse fields of study [76]. LP is commonly used in engineering , business , and economic problems. An LP problem can be defined as the problem of maximizing or minimizing a linear objective function subject to linear equality or inequality constraints. Semidefinite programming (SDP) is the most exciting development in mathematical programming in the 1990's([82], [56]).SDP has applications in such diverse fields as traditional convex constrained optimization, control theory, and combinatorial optimization. Because SDP is solvable via interior point methods, most of these applications can usually be solved very efficiently in practice as well as in theory. Although semidefinite programs are much more general than linear programs, they are just as easy to solve. Most interior-point methods for linear programming have been generalized to semidefinite programs. As in linear programming, these methods have polynomial worst-case complexity, and perform very well in practice. Also the maximum independent set problem (MIS) invokes finding the independent set of the maximum number of principal elements [3]. An exemplary in software engineering and in the premise of diagramming, MIS is known to be NP-hard [24].MIS has numerous significant applications, including combinatorial sales ([15], [25], [65], [84]) chart shading, coding hypothesis, mathematical tiling, issue determination, design acknowledgment, atomic science, and all the more as of late its application in bioinformatics have gotten significant(TSP) [45], [73].

1.2 Related Works

The previous works that are detailed in the framework of this dissertation are split into seven main categories:

1. Semidefinite Programming .
2. Semidefinite Relaxation in Combinatorial Optimization.
3. Multiplier(Augmented Lagrangians)and Penalty Methods.
4. NP-Hardnees.
5. Optimization Algorithms Design.
6. The Maximum Independent Set Problems(MIS).
7. Quadratic Programming (QP).

All literature relevant to each category will be carefully compiled. The pertinent literature for each category will be synthesized thoroughly.

1.2.1 Semidefinite Programming

Semidefinite programming (SDP) has been one of the most exciting and active research areas in optimization during the 1990s([75], [57]). It has attracted researchers with very diverse backgrounds, including experts in convex programming, linear algebra, numerical optimization, combinatorial optimization, control theory, and statistics. This tremendous research activity was spurred by the discovery of important applications in combinatorial optimization and control theory, the development of efficient interior-point algorithms for solving SDP problems, and the depth and elegance of the underlying optimization theory. One of the important problems is the problem of semidefinite programming where solving semidefinite programming problems is still a difficult approach and the difficulty of reaching the optimal solution [78]. In semidefinite programming we minimize a linear function subject to the constraint that an affine combination of symmetric matrices is positive semidefinite. Such a constraint is nonlinear and nonsmooth, but convex, so positive definite programs are convex optimization problems. Semidefinite programming unifies several standard problems (eg, linear and quadratic programming) and finds many applications in engineering. Although semidefinite programs are much more general than linear programs, they are just as easy to solve. Most interior-point methods for linear programming have been generalized to semidefinite programs. It's easy to see that a linear program LP is a special instance of an SDP [7].

1.2.2 Semidefinite Relaxation in Combinatorial Optimization

Combinatorial Optimization Computational problems are not the only things that have to be solved, they are also objects that can be worth studying. Problems and algorithms can be formalized analyzed mathematically. Therefore the semidefinite relaxation technique has received a lot of attention in combinatorial optimization since its emergence in the 1990([67], [70]). Semidefinite relaxation is now known as an efficient approach ([60], [61], [62], [72]). Therefore the semidefinite programming (SDP) relaxations are among the most powerful methods in this family and are surprisingly well suited for a broad range of problems where data take the form of matrices or graphs. It has been observed several times that when the statistical noise is small enough, SDP relaxations correctly detect the underlying combinatorial structures. Generally, semidefinite relaxations provide tight bounds for many difficult problems ([52], [63], [64], [81]). In our topic we use a semidefinite programming relaxation, which replaces a vector valued binary variable with a matrix valued continuous variable. Also, SDP has wide applicability in combinatorial optimization. A number of NP-hard combinatorial optimization problems have convex relaxations that are semidefinite programs [24]. In many instances, the SDP relaxation is very tight in practice, and in certain instances in particular, the optimal solution to the SDP relaxation can be converted to a feasible solution for the original problem with provably good objective value.

1.2.3 Multiplier(Augmented Lagrangian)and Penalty Methods

The method of multipliers or the augmented Lagrangian method this algorithm is related to the quadratic penalty algorithm, but it reduces the possibility of ill conditioning of the sub problems that are generated in this approach by introducing explicit Lagrange multiplier estimates at each step into the function to be minimized, which is known as the augmented Lagrangian function([35] [4]). In contrast to the penalty functions, the augmented Lagrangian function largely preserves smoothness, and implementations can be constructed from standard software for unconstrained or bound-constrained optimization.The beginning in the field of multiplier methods begins with Joh (1943). Kuhn and Tucker (1951) are eminent scientists who have conducted extensive research in the field of multiplier methods. Its results on the necessary conditions and adequate conditions are important in this field. Arrow and Hurwicz (1956) introduced the Lagrangian function. King (1966) developed the augmented Lagrangian algorithms. The augmented Lagrangian method was first proposed by Hestenes [1969] and Powell [1969], and then developed by Rockafellar, Bertsekas and others. The Augmented Lagrangian method has got a powerful theoretical tool for convex programming. Arrow, Gould, and Howe (1971)studied Rockafellar's augmented Lagrangian method and Pierre (1971)introduced a special augmented Lagrangian method with local convergence properties.

1.2.4 The NP-Hard problem

The NP-hard problem is one of the difficult problems that can not be solved in non-deterministic polynomial time. In 1965 [41] , June Hartmanis with Richard Stearns published a Turing award winning paper "On the Computational Complexity of Algorithms." In the 1970's, theoretical computer scientists showed hundreds of more problems NP-complete [39]. Linear programming problems are NP, as the number of steps in the simplex method, invented in 1947 by American mathematician George Dantzig, grows exponentially with the size of the input. in 1979 Russian mathematician Leonid Khachian discovered a polynomial-time algorithm. In August 2010 [26] , Vinay Deolalikar, believes he has solved the riddle of P vs NP in a move that could trans-form mankind's use of computers. as in the Figure 1.1

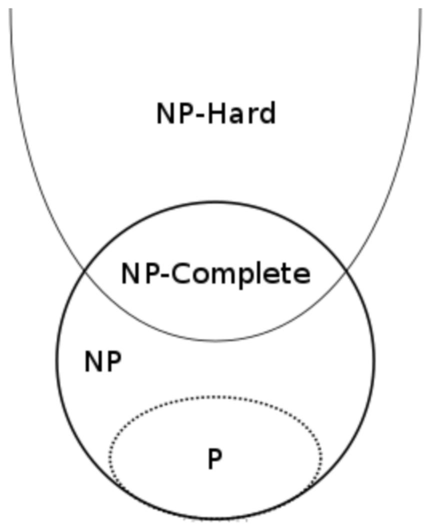


Figure 1.1: NP Contains P and Intersects The Class Of NP-Hard Problems. The Intersection Comprises The NP-Complete Problems

1.2.5 Optimization Algorithms Design

Design optimization is the aim of finding your best possible design parameters which also satisfy the design requirements. Practically, this means that parametric optimization algorithms intelligently test variations of your design automatically based on the variables and objectives you have determined. A common optimization would be a mass reduction of a design. Certain parameters of a particular model are chosen as variables that the algorithms can change to reduce the block([58], [83], [11]). At the same time a maximum level of stress is determined as an objective, that the algorithms must fulfill. as in the Figure 1.2

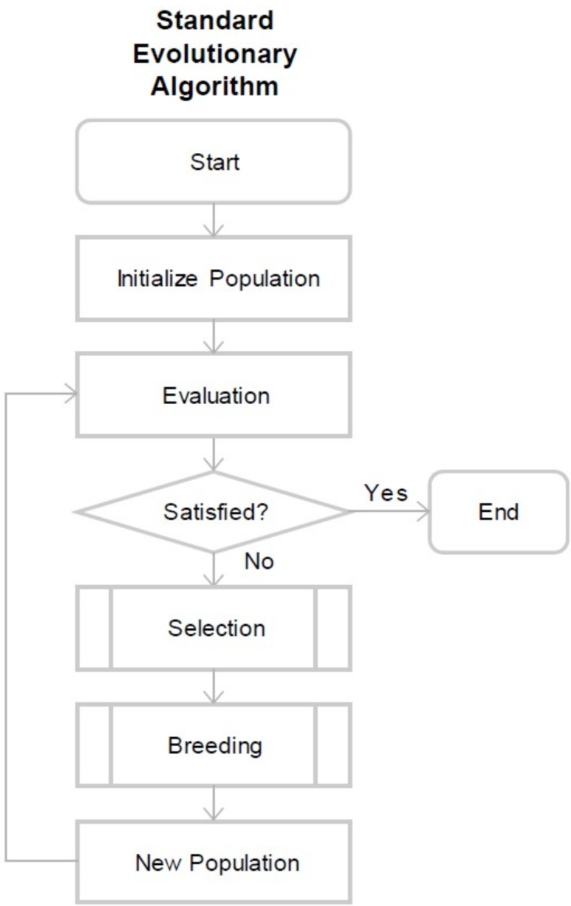


Figure 1.2: A generalized Standard Algorithm Used in Design Optimization Problems.

1.2.6 Maximum Independent Set Problem (MIS)

In a graph theory an independent set is a subset of vertices no two of which are adjacent. The maximum independent set problem consists in finding in a graph an independent set of maximum cardinality [74]. This problem is generally NP-hard The maximum independent set (MIS) problem is an fascinating area of combinatorial optimization and has several applications in different fields, for instance, physics, computer science , and mathematics. which means that is difficult to find the exact solution. Several papers have studied the MIS problem. This line of research started by Goemans-Williamson (1995) [3] with their approximation for the MIS problem based on semidefinite programming relaxation. Poljak showed that linear programming techniques cannot accomlish a better approximation solution.

1.2.7 Quadratic Programming (QP)

Quadratic programming is an important example of optimization with applications to engineering design, combinatorial optimization, game theory, and economics. Garey and Johnson state that quadratic programming is NP-hard.And Quadratic programming is the process of solving certain mathematical optimization problems involving quadratic functions.Specifically, one seeks to optimize (minimize or maximize) a multivariate quadratic function subject to linear constraints on the variables([36], [18], [34]). Quadratic programming is a type of nonlinear programming. Programming in this context refers to a formal procedure for solving mathematical problems. This usage dates to the 1940s and is not specifically tied to the more recent notion of "computer programming." To avoid confusion, some practitioners prefer the term "optimization" e.g., quadratic optimization.

CHAPTER 2

MATHEMATICAL BACKGROUND

Mathematical Background

In this chapter, we discuss the mathematical concepts that will be needed later in this dissertation.

2.1 Optimization Problem

The general basic formula of the optimization problem:

$$\begin{cases} \text{Minimize} & f(x) \\ \text{subject to} & g_i(x) \leq 0, \forall\, i \in \{1,\ \ldots,\ m\} \\ & h_i(x) = 0\ , \forall\, i \in \{1,\ \ldots,\ p\} \\ & x \in R^n \end{cases} \tag{2.1}$$

The function $f : R^n \to R$ that we wish to minimize is a real-valued function called the objective function or "**cost function**". The vector x is an n-vector of independent variables $x = [x_1, x_{2,\ldots}, x_n]^T \in R^n$. The variables $x_1, x_2, \ldots, x_n$ are often referred to as decision variables. The set R^n is a subset of R^n called the "**constraint set**" or "feasible set". The optimization problem above can be viewed as a decision problem that involves finding the "best" vector x of the decision variables over all possible vectors in R^n. By the "best" vector we mean the one that results in the-smallest value of the objective function. This vector is called the minimizer of f over R^n. It is possible that there may be many minimizers. In this case, finding any of the minimizers will suffice. There are also optimization problems that require maximization of the objective function, in which case we seek maximizers. Minimizers and maximizers are also called extremizers.

Maximization problems, however, can be represented equivalently in the minimization form above because maximizing fis equivalent to minimizing $-f$. Therefore, we can confine our attention to minimization problems without loss of generality.
See Figure 2.1

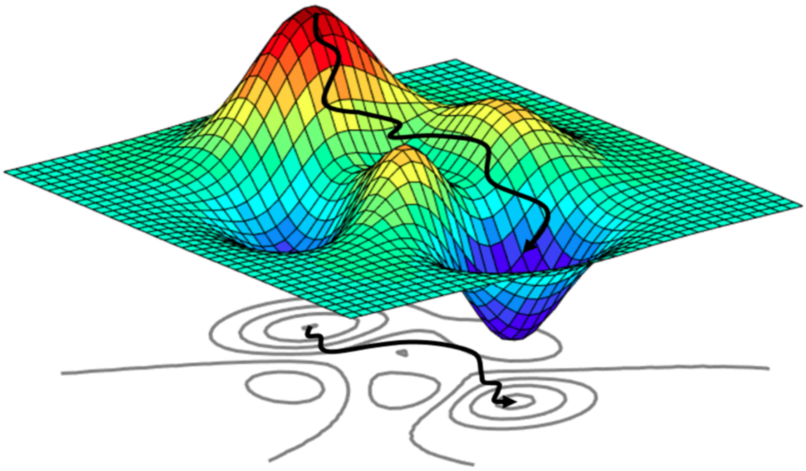


Figure 2.1: Optimization Problem

2.2 Definitions and Properties

In this section, we review definitions, notation, and necessary results of Euclidean space. Also, we give some basic concepts and facts of real mathematical analysis. For more on this substance, see, e.g., ([16], [55], [13], [22], [50], [40], [44], [9]. [49])

2.2.1 Unconstrained Optimization

In many practical applications, unconstrained optimization problems exist explicitly. If the variables have natural constraints,it is often safe to disregard them and conclude that they do not effect the optimum solution. Unconstrained problems often emerge as reformulations of constrained optimization problems,in which the restrictions are substituted by penalty terms in the objective function that has the effect of preventing constraint violations [5].In unconstrained optimization,we minimize an objective function that depends on real variables,without any constraints on the values of these variables.The basic formula of the optimization problem:

$$\begin{aligned} &\text{Minimize} \quad f(x) \\ &\text{subject to} \quad x \in R^n \end{aligned}$$

where x is a real vector, $n \geq 1$ elements and $f : R^n \to R$ is a smooth function [32]. The first thing will be to derive some conditions, which allow to decide whether a point is a minimum or not.

Definition 2.2.1. [22] Let $x \in R^n$ is **a local minimizer** of f if there exist $\varepsilon > 0$ such that $f(x) \leq f(y)$, for all $y \in R^n \cap B(x, \varepsilon)$. Where $B(x, \varepsilon) = \{x \in R^n|, \quad |y - x| \leqslant \varepsilon\}$

Definition 2.2.2. [22] Let $x \in R^n$ is **a global minimizer** of f if there exist $\varepsilon > 0$ such that $f(x) \leq f(y)$, for all $y \in R^n$

Definition 2.2.3. [22] Let $x \in R^n$ is **a strict local minimizer** of f if there exist $\varepsilon > 0$ such that $f(x) < f(y)$, for all $y \in R^n \cap B(x, \varepsilon), x \neq y$. Where $B(x, \varepsilon) = \{x \in R^n|, \quad |y - x| \leqslant \varepsilon\}$

Definition 2.2.4. [22] Let $x \in R^n$ is **astrict global minimizer** of f if there exist $\varepsilon > 0$ such that $f(x) < f(y)$, for all $y \in R^n, x \neq y$.

Definition 2.2.5. [22] Let $x \in R^n$ is **a local maximizer** of f if there exist $\varepsilon > 0$ such that $f(x) \geq f(y)$, for all $y \in R^n \cap B(x, \varepsilon)$. Where $B(x, \varepsilon) = \{x \in R^n|, \quad |y - x| \leqslant \varepsilon\}$

Definition 2.2.6. [22] Let $x \in R^n$ is **a global maximizer** of f if there exist $\varepsilon > 0$ such that $f(x) \geq f(y)$, for all $y \in R^n$

Definition 2.2.7. [22] Let $x \in R^n$ is **a strict local maximizer** of f if there exist $\varepsilon > 0$ such that $f(x) > f(y)$, for all $y \in R^n \cap B(x, \varepsilon), x \neq y$. Where $B(x, \varepsilon) = \{x \in R^n|, \quad |y - x| \leqslant \varepsilon\}$

Definition 2.2.8. [22] Let $x \in R^n$ is **a strict global maximizer** of f if there exist $\varepsilon > 0$ such that $f(x) > f(y)$, for all $y \in R^n, x \neq y$ (Figure2.2)

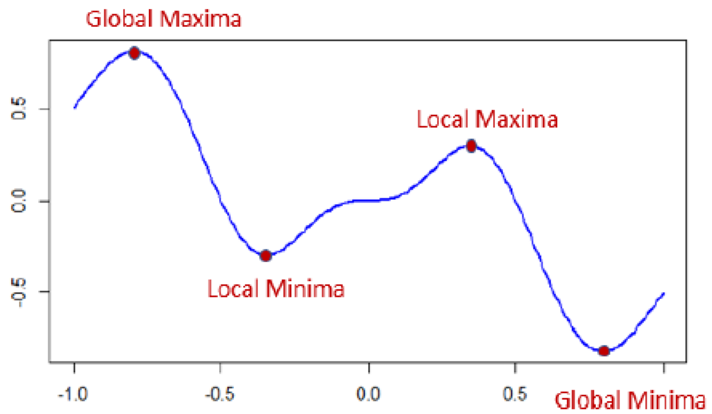


Figure 2.2: Local and Global Minimize

Definition 2.2.9. (**Adjacency matrix**)The adjacency matrix of a graph G=(V,E) with n vertices is the matrix $A \in M_n$ whose entries are

$a_{ij} = 1$ indicates presence of edge between two vertices i and j.

$a_{ij} = 0$ indicates absence of edge between two vertices i and j.

An example of adjacency matrix representation of an undirected and figure is given below:

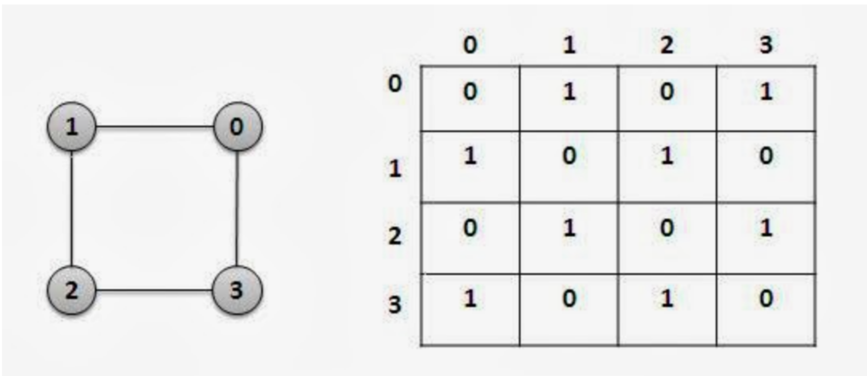


Figure 2.3: Adjacency Matrix Representation of Undirected Graph

Definition 2.2.10. (**a set**) A set is an unordered collection of objects, called elements or members of the set. A set is said to contain its elements. We write $a \in A$ to denote that a is an element of the set A.

Definition 2.2.11. (**Graph**) A set of items connected by edges. Each item is called a vertex or node. Formally, a graph is a set of vertices and a binary relation between vertices, adjacency.

Definition 2.2.12. (**Node**) A unit of reference in a data structure. Also called a vertex in graphs and trees. A collection of information which must be kept at a single memory location.

Definition 2.2.13. (**Edge**) In Figure2.4 a connection between two vertices of a graph. In a weighted graph, each edge has an number, called a "weight". In a directed graph, an edge goes from one vertex, the source, to another, the target, and hence makes connection in only one direction.

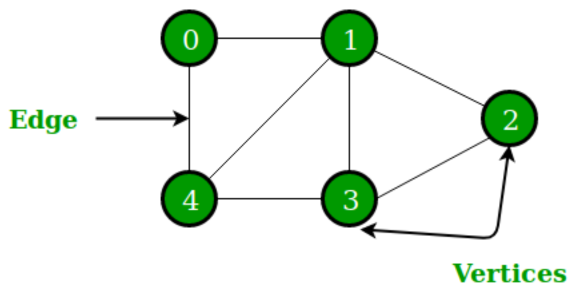


Figure 2.4: Vertices and Edges

Definition 2.2.14. (**Hamiltonian cycle**) A path through a graph that starts and ends at the same vertex and includes every other vertex exactly once.

Definition 2.2.15. (**Traveling Salesman**) The traveling Salesman is find a path through a weighted graph that starts and ends at the same vertex, includes every other vertex exactly once, and minimizes the total cost of edges. which is represented of NP-Hard (Figure 2.5).

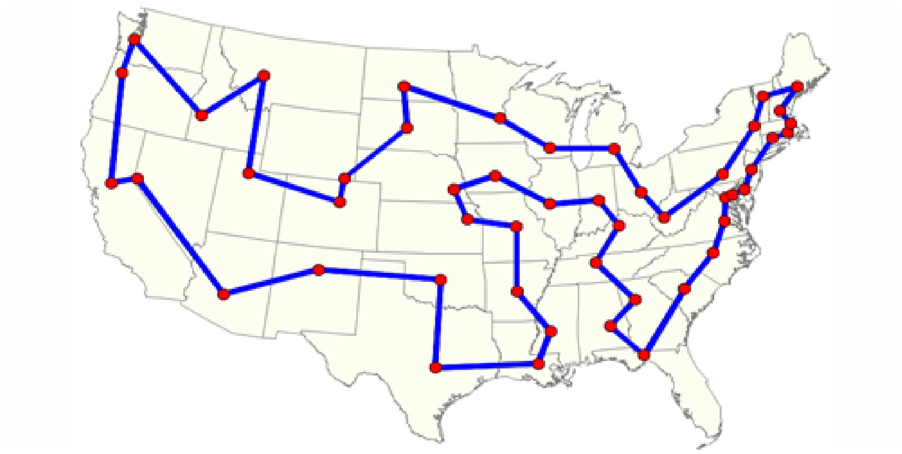


Figure 2.5: Traveling Salesman Problem

Definition 2.2.16. (**The Feasible Region**) The feasible region or the feasible set is the set of all possible values of the problem that satisfies all the constraints of the problem. Otherwise, the problem is said to be infeasible if no solution exists which satisfies all the constraints. The constraints could be equalities and inequalities. (Figure2.6)

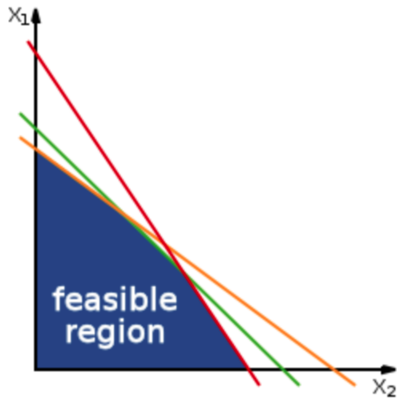


Figure 2.6: Feasible Region

2.2.2 Vector Norm

In this section, we introduce the Euclidean space, which means a finite dimensional vector space with an inner product $\langle ., . \rangle$ and the Euclidean norm defined by $\|x\| = \sqrt{\langle x, x \rangle}$ for all vectors x in the vector space.

Definition 2.2.17. A function $\langle ., . \rangle : R^n \times R^n \to R$ is an inner product if

1. $\langle x, x \rangle > 0, \langle x, x \rangle = 0 \Leftrightarrow x = 0$ (positivity).

2. $\langle x, y \rangle = \langle y, x \rangle$ (symmetric).

3. $\langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle$ (additivity).

4. $\langle \beta x, y \rangle = \beta \langle x, y \rangle$ for all $\beta \in R$ (homogeneity).

Definition 2.2.18. [20] A function $\| \cdot \|: \mathbb{R}^n \to \mathbb{R}$ is called to be a **Norm**, if it satisfies the following properties:

1. $\|a\| \geq 0$, $\forall a \in \mathbb{R}^n$; $\|a\| = 0$ if and only if $a = 0$.

2. $\|\beta a\| = |\beta| \|a\|$, $\forall \beta \in \mathbb{R}$, $\forall a \in \mathbb{R}^n$.

3. $\|a + b\| \leq \|a\| + \|b\|$, $\forall a, b \in \mathbb{R}^n$.

The **Euclidean norm** is defined by as follows :

$$\|a\|_2 = \sqrt{a^T a} = \sqrt{\sum_{i=1}^{n} a_i^2}.$$

The inner product of any two real n-vectors a and b is defined by:

$$\langle a, b \rangle = a^T b = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$$

The Cauchy-Schwarz Inequality: For vectors a and b in R^n, we have:

$$|\langle a, b \rangle| \leq \|a\| \|b\|$$

Definition 2.2.19. [22] (Karush-Kuhn-Tucker (KKT) Conditions) The following four conditions are called KKT conditions (for a problem with differentiable f_i, g_i):

- Primal constraints: $f_i(x) \leq 0, i = 1, \ldots, I$, and $g_i(x) = 0, i = 1, \ldots, E$.

- Complementary slackness: $\beta_i f_i(x) = 0, i = 1, \ldots, I$.

- Dual constraints: $\beta_i \geq 0, i = 1, \ldots, m$.

- Gradient of Lagrangian with respect to x vanishes:

$$\nabla f_0(x) + \sum_{i=1}^{I} \beta_i \nabla f_i(x) + \sum_{i=1}^{E} \alpha_i \nabla g_i(x) = 0$$

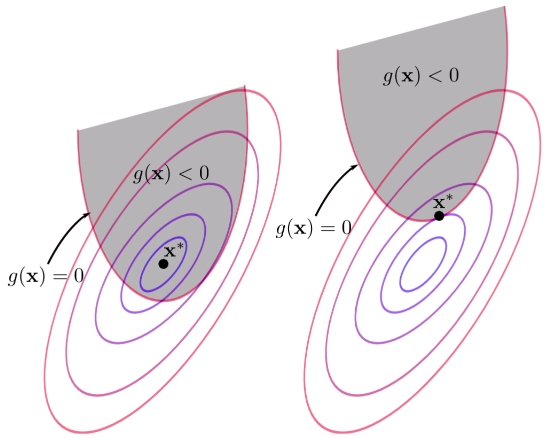


Figure 2.7: Graphical Explanation For The KKT Conditions.

Example 2.2.1.

$$\begin{aligned} \text{Minimize} \quad & 2x_1^2 + x_2^2 \\ \text{subject to} \quad & x_1 + x_2 = 1 \end{aligned}$$

Let us first consider the unconstrained case. Differentiate with respect to x_1 and x_2

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 4x_1,$$
$$\frac{\partial f(x_1, x_2)}{\partial x_2} = 2x_2.$$

These yield the solution $x_1 = x_2 = 0$. Does not satisfy the constraint.

Definition 2.2.20. (**Critical Point**) The critical point is a point in the domain of the function where the function either the derivative is equal to zero or not differentiable.

Definition 2.2.21. [47] For any feasible set x, the active set $A(x)$ of the inequality constraints denoted by:

$$A(x) = E \cup \{j \in L \mid h_j(x) = 0\}$$

Definition 2.2.22. [22] The epigraph of a function f : $\Omega \to R, \Omega \subset R^n$, denoted epi($f$), is a sub set of R^{n+1} and given by

$$\text{epi}(f) = \left\{ \begin{bmatrix} x \\ \\ \alpha \end{bmatrix}, x \in \Omega, \quad \alpha \in \text{R}, \quad \alpha \geq \text{f(x)} \right\}$$

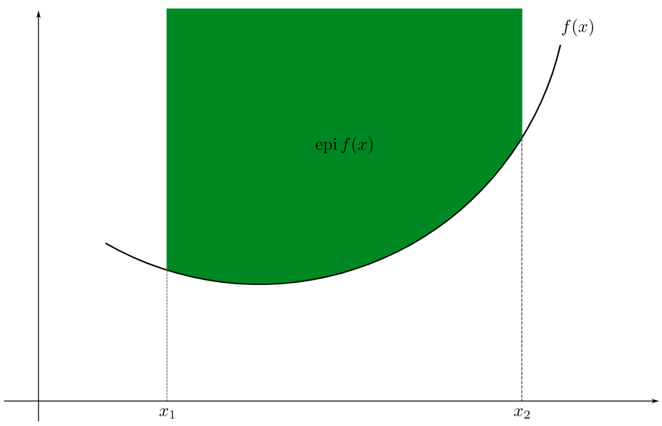


Figure 2.8: Epigraph

The hypograph of f, denoted by hyp(f),

$$\text{hyp}(f) = \left\{ \begin{bmatrix} x \\ \\ \alpha \end{bmatrix} \mid f(x) \geq \alpha, \quad x \in \Omega, \quad \alpha \in R \right\}$$

Definition 2.2.23. (A Linear Transformation)

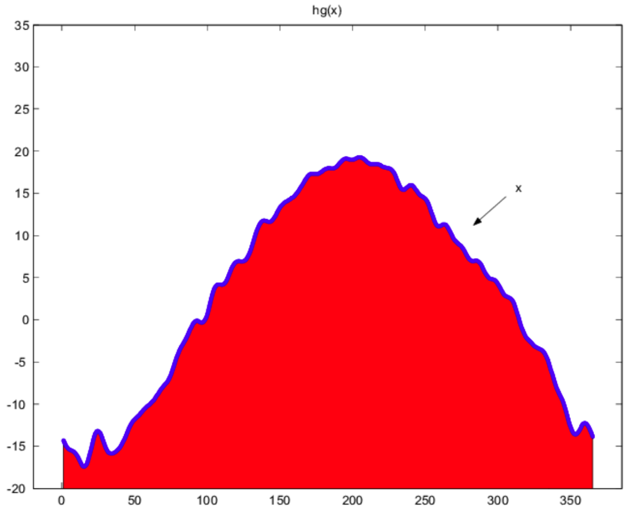


Figure 2.9: Hypograph Of The Function x.

Is a function $L : R^n \to R^m$ that satisfies the following properties:

$$L(a+b) = L(a) + L(b) \quad \text{and} \quad L(\beta a) = \beta L(a),$$

for all vectors $a, b \in R^n$ and for all scalar $\beta \in R$.

Definition 2.2.24. (Adjoint of a Linear Transformation)
Let $A_i \in S^n$, for $i = 1, ..., m$ and define the linear transformation $A : S^n \to R^m$ as $(AX)_i = \langle A_i, X\rangle$, for $i = 1, ..., m$. Then the adjoint of A is defined as the unique linear transformation $A^* : R^m \to S^n$ that satisfies $\langle AX, y\rangle = \langle X, A^*y\rangle$ for each $X \in S^n$ and for all $y \in R^m$. Since

$$\langle AX, y\rangle = \sum_{i=1}^{m} y_i tr(A_i X) = tr\Big(X \sum_{i=1}^{m} y_i A_i\Big) = \langle X, A^*y\rangle$$

we obtain $A^*y = \displaystyle\sum_{i=1}^{n} y_i A_i$.

Definition 2.2.25. Let $f : R^n \to R$ and $x \in R^n$. Then the partial derivative of f at x with respect to x_i is defined as:

$$\frac{\partial f(x)}{\partial x_i} = \lim_{t \to 0} \frac{f(x + te_i) - f(x)}{t}$$

where e_i is i^{th} unit vector. The gradient of f at x is defined as the column vector

$$\nabla f(x) = \begin{bmatrix} \dfrac{\partial f(x)}{\partial x_1} \\ \vdots \\ \dfrac{\partial f(x)}{\partial x_n} \end{bmatrix}$$

The Hessian matrix is defined as the $n \times n$ symmetric matrix:

$$\nabla^2 f(x) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1 \partial x_1} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n \partial x_n} \end{bmatrix}$$

The directional derivative of the function f at x in the direction d given by:

$$f'(x, d) = \lim_{t \to 0+} \frac{f(x + td) - f(x)}{t}$$

We say the function f is differentiable at x if and only if the gradient $\nabla f(x)$ exists and satisfies $\langle \nabla f(x), d \rangle = f'(x, d), \forall d \in R^n$. Moreover, we say the function f is differentiable over a subset S of R^n if it is differentiable at every $x \in S$. Otherwise, is non-differentiable, and$f$is continuously differentiable over S, if $\lim_{d\to 0} \frac{f(x+d)-f(x)-\langle \nabla f(x), d\rangle}{\|d\|} = 0$, for all $x \in S$ where $\|.\|$ is an arbitrary vector norm.

Theorem 2.2.1. *(Taylor Expansion) [45] suppose* $f\colon \mathbb{R}^n \to \mathbb{R}$ *be continuously differentiable. Then, for every* $x_1, x_2 \in R^n$*, there is an* $\beta \in [0,1]$*, such that*

$$f(x_2) = f(x_1) + \nabla f(\beta x_1 + (1-\beta)x_2)^T (x_2 - x_1).$$

Furthermore, if f *is a twice continuously differentiable, then, for every* $x_1, x_2 \in R^n$*, there is* $\beta \in [0,1]$*, such that*

$$f(x_2) = f(x_1) + \nabla f(x_1)^T (x_2 - x_1) + \frac{1}{2}(x_2 - x_1)^T \nabla^2 f(\beta x_1 + (1-\beta)x_2)(x_2 - x_1).$$

In addition, if $x, w \in R^n$ *and* $\rho \in R$*, we have*

$$f(x+\rho w) = f(x) + \rho w^T \nabla f(x) + \frac{\rho^2}{2} w^T \nabla^2 f(x) w + o(\rho^2) \;\; \textit{as } \rho \to 0.$$

Definition 2.2.26. Let $0 \neq S \subseteq R^n$. If the function $f : \Omega \to R$, the argument of the minimum is the set of elements in Ω that achieve the global minimum in Ω, which is defined by

$$\text{argmin}_{x\in\Omega} f(x) = \{x \in \Omega \mid f(y) \geq f(x), \text{ for all } y \in \Omega\}$$

Definition 2.2.27. In an optimization problem, a slack variable is a variable that is added to an inequality constraint to transform it to an equality. Introducing a slack variable replaces an inequality constraint with an equality constraint and a nonnegativity constraint.

2.2.3 Convex Optimization(Convex Sets and Convex Functions)

A convex optimization problem is an optimization problem in which the objective function is a convex function and the feasible set is a convex set. A convex optimization problem is in standard form if it is written as:

$$\begin{cases} \text{Minimize} & f(x) \\ \text{subject to} & g_i(x) \leq 0, \quad \forall\, i \in \{1, \ldots, m\} \\ & h_i(x) = 0\,, \quad \forall\, i \in \{1, \ldots, p\} \\ & x \in R^n \end{cases}$$

Definition 2.2.28. [22] Let $S \subseteq R^n$. If the line segment between any two points in S lies in S, i.e.

$$\partial x_1 + (1-\partial)\, x_2 \in S, \;\; \forall x_1, x_2 \in S\,, \quad 0 \leq \partial \leq 1$$

Then S is said to be ***convex***. It can be indicated that a set $S \subseteq R^n$ is convex iff for any $x_1, \ldots, x_n \in S$, the convex combination $\sum_{i=0}^{n} \partial_i x_i$ Where $\sum_{i=0}^{n} \partial_i = 1$, and $\partial_i \geq 0, i = \{1, \ldots, n\} \subseteq S$.

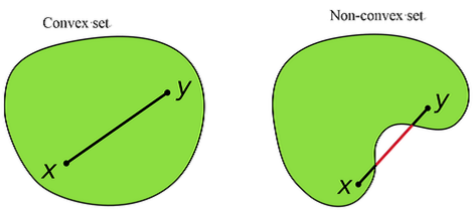


Figure 2.10: Convex Set and Non Convex Set

Definition 2.2.29. Let $S \subseteq R^n$ be a convex set and $S \neq \emptyset$. If $f : S \to R$ satisfies:

$$f\left(\partial x_1 + (1-\partial)\, x_2\right) \leq \partial f\left(x_1\right) + (1-\partial)\, f(x_2)\,, \;\; \forall x_1, x_2 \in S\,,\;\; 0 \leq \partial \leq 1$$

Then f is said to be ***"a convex function*** "on S. If the inequality above is true as a strict inequality for all$x_1 \neq x_2$and for all$0 < \partial < 1$,then f we say is "***a strictly convex function***" on S. If there is a constant a > 0;$\forall\ x_1, x_2 \in S$ and for all $0 \leq \alpha \leq 1$
$f\left(\partial x_1 + (1-\partial)\, x_2\right) \leq \partial f\left(x_1\right) + (1-\partial)\, f(x_2) - \dfrac{1}{2} a\partial\, (1-\partial)\, \|x_1 - x_2\|^2$

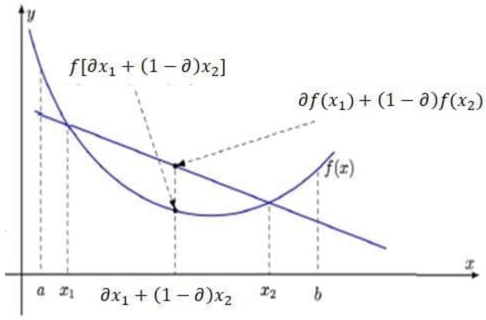


Figure 2.11: Convex Function

Definition 2.2.30. [22] The **convex hull** of a set $C \subseteq \mathbb{R}^n$ is denoted by $conv(C)$ and is the smallest convex set that contains C. The convex hull of a set C is the set of all convex combinations of points in C; i.e.,

$$conv(C) = \left\{ \sum_{i=1}^{n} \beta_i a_i \;\middle|\; a_i \in C,\ \beta_i \geq 0,\ i = 1, \ldots, n,\ \sum_{i=1}^{n} \beta_i = 1 \right\}$$

Definition 2.2.31. The rank of a matrix, rank(D), is the maximum number of linearly independent column vectors in the matrix. Also, rank(D) is the maximum number of linearly independent row vectors in the matrix.

Definition 2.2.32. Suppose $f\colon R^n \to R$ be differentiable. A point $x^* \in R^n$ is said to be a **stationary point** of f if $\nabla f(x^*) = 0$.

Definition 2.2.33. A cone K is a convex cone if $\sigma x + \rho y \in K$ for any positive scalars σ, ρ and any x, y in K. A cone K is convex if and only if $K + K \subseteq K$

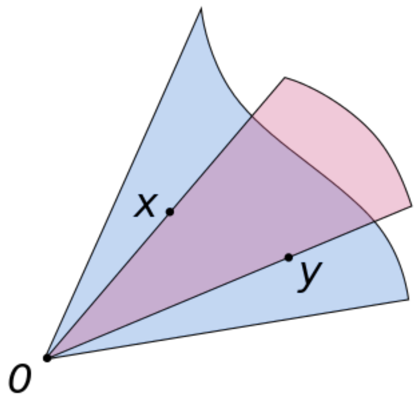


Figure 2.12: A Convex Cone

2.2.4 Matrix Calculus

In this section we will study some matrices and the matrices encountered most frequently in numerical optimization are symmetric.

Definition 2.2.34. ([80], [46], [17]) A matrix, for which horizontal and vertical dimensions are the same (that is, an $n \times n$ matrix), is called a square matrix of order n. A square matrix $B \in R^{n \times n}$ is called symmetric if $B^T = B.$, i.e $\mathrm{a}_{ij} = \mathrm{a}_{ji}$, for all $i, j \in \{1, \ldots, n\}$ and a_{ij} is the element of matrix B in row i of column j. The matrix $B^T \in R^{n \times n}$ is called the transpose of B. A square matrix $B \in R^{n \times n}$ is called positive definite If

$$x^T Bx > 0, \forall x \in R^n, x \neq 0$$

and negative definite If

$$x^T Bx < 0, \forall x \in R^n, x \neq 0$$

Correspondingly, a square matrix $B \in R^{n \times n}$ is called positive semidefinite if

$$x^T Bx \geq 0, \forall x \in R^n$$

and negative semidefinite if

$$x^T Bx \leq 0 \text{ forall } x \in R^n$$

A matrix which is neither positive or negative semidefinite is called indefinite. An inverse of matrix $B \in R^{n\times n}$ is a matrix $B^{-1} \in R^{n\times n}$ such that:

$$BB^{-1} = B^{-1}B = I$$

where $I \in R^{n\times n}$ is the identity matrix. A scalar λ is called an eigenvalue of the matrix $B \in R^{n\times n}$ if

$$Bx = \lambda x$$

for some nonzero vector $x \in R^n$. The vector x is called an eigenvector associated to the eigenvalue λ.

Definition 2.2.35. The trace of a matrix $X \in R^{n\times n}$ is defined by:

$$\mathrm{tr}(X) = \sum_{i=1}^{n} x_{ii}$$

The function $tr : R^{n\times n} I R$ is linear; that is, for any two $n \times n$ matrices X and Y and two scalars γ and δ, we have

$$\mathrm{tr}(\gamma X + \delta Y) = \sum_{i=1}^{n}(\gamma X_{ii} + \delta Y_{ii}) = \gamma\,\mathrm{tr}(X) + \delta\,\mathrm{tr}(Y)$$

For any two symmetric matrices $X, Y \in S^n$; we define the Frobenius inner product

$$\langle X, Y\rangle = \sum_{i=1}^{n}\sum_{j=1}^{n} X_{ij}Y_{ij} = \mathrm{tr}(XY)$$

Definition 2.2.36. Smooth and Non-smooth functions [8]

Smooth functions have a unique defined first derivative (slope or gradient) at every point. Graphically, a smooth function of a single variable can be plotted as a single continuous line with no abrupt bends or breaks. As illustrated in the following graph. (Figure 2.13) Non-smooth functions include non-differentiable and discontinuous functions. Functions with first derivatives with undefined regions are called non-differentiable. Graphs of non-differentiable functions may have abrupt bends. The absolute value of a variable, is an example of a non-differentiable expression, as illustrated in the following graph.

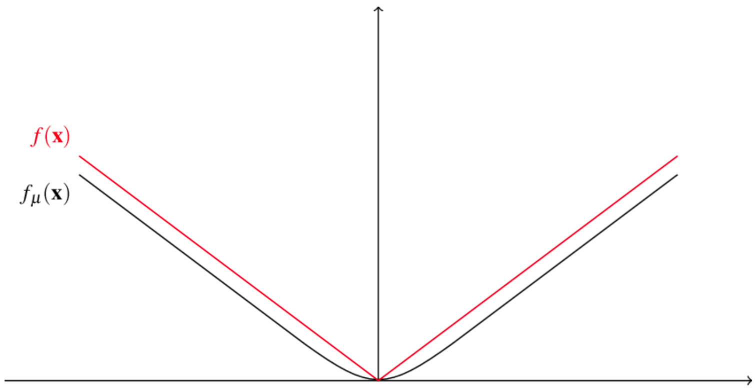


Figure 2.13: Function $f_\mu(x)$ Is a Smooth Approximation to Non-Smooth Function $f(x)$

2.2.5 The Proximal Operator

This section is devoted to the study of the Proximal Operator (proximal mapping), which will be fundamental in many of the algorithms that will be explored later in the thesis.

Definition 2.2.37. [71] the proximal mapping of a closed convex function f is

$$\mathrm{prox}_f(\mathrm{y}) = \underset{u}{\mathrm{argmin}} \left(f(x) + \frac{1}{2}\|x - y\|_2^2 \right)$$

Existence and uniqueness: we minimize a closed and strongly convex function

$$g(x) = \left(f(x) + \frac{1}{2}\|x - y\|_2^2 \right)$$

- minimizer exists because g is closed with bounded sublevel sets.

- minimizer is unique because g is strictly convex. The proximal operator of f with parameter $\mu > 0$ is given by

$$\text{prox}_{\mu f}(\text{x}) = \underset{u}{\text{argmin}} \left(f(x) + \frac{1}{2\mu} \|x - y\|_2^2 \right)$$

From [23] the following properties hold for all $\mu \geq 0$

- The point x^* minimizes f if and only if prox $_{\mu f}$ $(x^*) = x^*$.
- If $f(x) = \frac{1}{2} x^T Ax + b^T x + c$ is a convex quadratic function, then

$$\text{prox}_{\mu f}(\text{y}) = (1 - \mu A)^{-1}(y - \mu b)$$

- If $f(x) = b^T x + c$ is affine function, then prox $_{\mu f}(\text{x}) = (\text{y} - \mu\text{b})$
- If f(x) $= c$ is a constant, then prox $_{\mu f}(\text{y}) = \text{y}$
- If $f(x) = \|x\|_2$ is Euclidean norm then

$$\text{prox}_{\mu \text{f}}(x) = \begin{cases} \left(\dfrac{1 - \mu}{\|x\|_2} \right)^x & \|x\|_2 \geq \mu \\ 0 & \text{otherwise} \end{cases}$$

- If $f(x) = -\sum_{i=1}^{n} \log x_i, \forall x_i > 0$ is Logarithmic function then

$$\text{prox}_{\mu f}(x)_i = \frac{x_i + \sqrt{x_i^2 + 4\mu}}{2}, i = 1, 2, \ldots, n$$

2.2.6 Combinatorial Optimization Problems

Broadly speaking, combinatorial optimization problems are problems that involve finding the "best" object from a finite set of objects [24]. In this context, "best" is measured by a given evaluation function that maps objects to some score or cost, and the objective is to find the object that merits the lowest cost.See figure 2.14

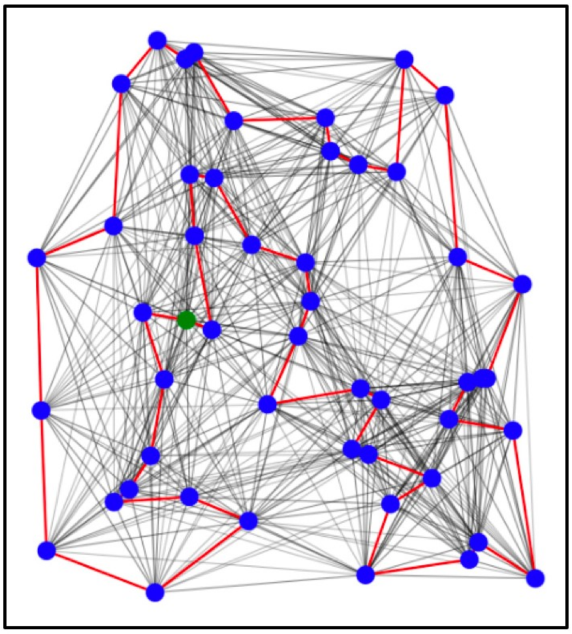


Figure 2.14: Combinatorial Optimization Problems

Most practically interesting combinatorial optimization problems (COPs from now on) are also very hard, in the sense that the number of objects in the set increases extremely fast due to even small increases in the problem size, making exhaustive search impractical. To make things clearer, we will focus on a specific problem, the well-known Traveling Salesman Problem (TSP). In this problem we have N cities, and our salesman must visit them all. However, travelling between cities incurs some cost and we must find a tour that minimizes the total accumulated cost while traveling to all the cities and returning to the starting city. For example, the image in (Figure 2.5) shows an optimal tour of all the capital cities in the US [31].

2.3 Optimality Conditions for Constrained Optimization

The Constrained Optimization problem Consists of equality (or inequality $g_i(x) \leq 0, \quad i = 1, \ldots, I,$) constraints, and it can be expressed as: and

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & h_j(x) = 0, \quad j = 1, \ldots, m \\ & x \in X, \end{aligned} \tag{2.2}$$

where f and h_j defined from $X \subseteq R^n$ into R are assumed to be continuously differentiable functions. The feasible set of the problem (2.2) is denoted by

$$S = \{x \in X \mid h(x) = 0\},$$

where h is the function with component functions $h_1, \ldots, h_m$ In our study we assume $X = R^n$.

The Lagrangian function $L\colon R^n \times R^m \to R$ is represented by

$$L(x, \alpha) = f(x) + \alpha^T h(x),$$

where $\alpha = (\alpha_1, \ldots, \alpha_m)^T$ is called the Lagrange multiplier vector. We have the following optimality conditions.

Theorem 2.3.1. ***(First-Order Necessary Conditions)*** *[30] The first order necessary conditions (KKT conditions) for x^* to be a local minimum of problem* (2.2) *are that x^* must be a feasible point (i.e., $h(x^*) = 0$) and there exists a Lagrange multiplier vector α^* such that*

$$\nabla_x L(x^*, \alpha^*) = 0.$$

Theorem 2.3.2. ***(Second-Order Necessary Conditions)*** *[30] The second order necessary conditions for x^* to be a local minimum of problem (2.2) are that x^* must be a feasible point and*

$$\nabla_x L(x^*, \alpha^*) = 0, \quad w^T \nabla^2_{xx} L(x^*, \alpha^*) w \geq 0,$$

for all w that satisfies $\nabla h(x^*)^T w = 0$.

Theorem 2.3.3. ***(Second-Order Sufficient Conditions)*** *[30] The sufficient conditions for x^* to be a local minimum of problem (2.2) are that x^* is a KKT point (i.e., x^* and α^* satisfy the first order necessary conditions) and that*

$$w^T \nabla^2_{xx} L(x^*, \alpha^*) w > 0,$$

for every nonzero vector w that satisfies $\nabla h(x^*)^T w = 0$.

2.4 Optimality Conditions for Unconstrained Optimization

In this section, we consider unconstrained optimization problem. If $X = R^n$, i.e., minimize f sans constraints [14, 30], it can be expressed by:

$$\underset{x \in R^n}{\text{minimize}} \quad f(x). \tag{2.3}$$

- If f is continuously differentiable, then a necessary condition for $x^* \in R^n$ to be a solution of problem (2.3) is

$$\nabla f(x^*) = 0.$$

- If f is twice continuously differentiable, then a necessary conditions for $x^* \in R^n$ to be a solution of problem (2.3) is

$$\nabla f(x^*) = 0, \;\; \nabla^2 f(x^*) \geq 0.$$

- The sufficient conditions for $x^* \in R^n$ to be a local solution of problem (2.3) are

$$\nabla f(x^*) = 0, \;\; \nabla^2 f(x^*) > 0.$$

Theorem 2.4.1. ***(First-Order Necessary Condition)*** *[14, 30] Suppose $f\colon R^n \to R$ be differentiable. If x^* is a local minimizer of f, then $\nabla f(x^*) = 0$.*

Proof: Define $h\colon R \to R$ as $h(\rho) = f(x^* + \rho w)$ for some $w \in R^n$, then $h'(\rho) = w^T \nabla f(x^* + \rho w)$. If $\rho = 0$, we get $h'(0) = w^T \nabla f(x^*)$. By definition,

$$h'(0) = \lim_{\rho \to 0} \frac{f(x^* + \rho w) - f(x^*)}{\rho}.$$

Since x^* is a local minimizer, there exists $k > 0$, such that $f(x^* + \rho w) \geq f(x^*)$ for every $0 < \rho \leq k$, therefore we get $w^T \nabla f(x^*) \geq 0$. Since w is an arbitrary, we can substitute w by $-w$, and thus $-w^T \nabla f(x^*) \geq 0$. Therefore, $w^T \nabla f(x^*) = 0$, for every $w \in R^n$. Thus, $\nabla f(x^*) = 0$. ■

Theorem 2.4.2. ***(Second-Order Necessary Condition)*** *[14,30] Suppose $f\colon R^n \to R$ be twice differentiable. If x^* is a local minimizer of f, then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite.*

Proof: Suppose $w \in R^n$. We want to show that $w^T \nabla^2 f(x^*) w \geq 0$. By using Taylor expansion of f at x^*, we obtain

$$f(x^* + \rho w) = f(x^*) + \rho w^T \nabla f(x^*) + \frac{\rho^2}{2} w^T \nabla^2 f(x^*) w + o(\rho^2).$$

Since $\nabla f(x^*) = 0$ by First-Order Necessary Condition theorem, we have

$$f(x^* + \rho w) = f(x^*) + \frac{\rho^2}{2} w^T \nabla^2 f(x^*) w + o(\rho^2).$$

Dividing both sides by ρ^2, we have

$$\frac{f(x^* + \rho w) - f(x^*)}{\rho^2} = \frac{1}{2} w^T \nabla^2 f(x^*) w + \frac{o(\rho^2)}{\rho^2}.$$

Taking the limit of both sides, and using the fact that x^* is a local minimizer, we obtain

$$0 \leq \lim_{\rho \to 0} \frac{f(x^* + \rho w) - f(x^*)}{\rho^2} = \lim_{\rho \to 0} \left\{ \frac{1}{2} w^T \nabla^2 f(x^*) w + \frac{o(\rho^2)}{\rho^2} \right\}.$$

Since

$$\lim_{\rho \to 0} \frac{o(\rho^2)}{\rho^2} = 0,$$

we conclude that $w^T \nabla^2 f(x^*) w \geq 0$. Therefore, $\nabla^2 f(x^*)$ is positive semidefinite. ■

Theorem 2.4.3. ***(Second-Order Sufficient Condition)*** *[14] Suppose $f\colon R^n \to R$ be twice continuously differentiable. If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, then x^* is a strict local minimizer.*

Proof: Since $\bigtriangledown^2 f(x^*)$ is positive definite and $\bigtriangledown^2 f$ is continuous, we have that $w^T \bigtriangledown^2 f(x) w > 0$, for every $x \in B(x^*, k)$ for some $k > 0$, and for any $w \neq 0$. Now let $w \neq 0$ such that $x^* + w \in B(x^*, k)$. Since $\nabla f(x^*) = 0$, and by using the Taylor Expansion Theorem (2.2.1), there exists a point p that lies between x^* and $x^* + w$ such that

$$\begin{aligned} f(x^* + w) &= f(x^*) + \nabla f(x^*)^T w + \frac{1}{2} w^T \nabla^2 f(p) w \\ &= f(x^*) + \frac{1}{2} w^T \nabla^2 f(p) w \\ &> f(x^*). \end{aligned} \tag{2.4}$$

Hence $f(x^* + w) > f(x^*)$, so x^* is a strict local minimum. ■

CHAPTER 3

SEMIDEFINITE PROGRAMMING

3.1 Introdiction

In this chapter, we discuss some of the concepts, terms and mathematics that will be required in this thesis. Because this is about a semidefinite programming solution. Semidefinite programming (SDP) is probably the most exciting development in mathematical programming in the last ten years. SDP has applications in such diverse fields as traditional convex constrained optimization control theory and combinatorial optimization [82]. Because SDP is solvable via interior-point methods (and usually requires about the same amount of computational resources as linear optimization), most of these applications can usually be solved fairly efficiently in practice as well as in theory.

3.2 Linear Programming

In this section, we study the linear programming (LP) is one of the simplest ways to perform optimization. It helps you solve some very complex optimization problems by making a few simplifying assumptions. As an analysis, you are bound to come across applications and problems to be solved by Linear Programmingg [14, 38]. Consider the linear programming problem in standard form:

$$(LP)\begin{cases} \text{minimize} & \langle c, x\rangle \\ \text{subject to} & \langle a_i, x\rangle = b_i, \quad i = 1, \dots, m \\ & x_i \geq 0, \quad x \in R^n. \end{cases} \tag{3.1}$$

where $a_i \in R^n$, for $i = 1, \dots, m$, the numbers $b_i \in R^n$, for $i = 1, \dots, m$, and the vector $c \in R^n$, $x_i \geq 0$, for $i = 1, \dots, n$. The Lagrangian is given by

$$L(x, y) = \langle c, x\rangle + \langle y, b_i - \langle a_i, x\rangle\rangle$$

Consider the primal (P) and the dual (D) standard linear programming problem:

$$(P)\begin{cases}\text{maximize} & \langle c, x\rangle \\ \text{subject to} & Ax = b, \\ & x \geq 0,\end{cases} \tag{3.2}$$

and

$$(D)\begin{cases}\text{minimize} & \langle b, y\rangle \\ \text{subject to} & A^T y \geq c\end{cases} \tag{3.3}$$

Example 3.2.1. Consider the following simple linear programming problem:

$$(P)\begin{cases}\text{maximize} & x_1 + 2x_2 + 4x_3 \\ \text{subject to} & x_1 + x_2 + x_3 = 6, \\ & x \geq 0.\end{cases} \tag{3.4}$$

The dual of problem (3.4) is given by

$$(D)\begin{cases}\text{minimize} & 6y_1 \\ \text{subject to} & y_1 \geq 1, \\ & y_1 \geq 2, \\ & y_1 \geq 4.\end{cases} \tag{3.5}$$

The optimal solution of problem (3.4) is $x_1 = x_2 = 0$, $x_3 = 6$ and the optimal value is $x_1 + 2x_2 + 4x_3 = 24$; an optimal solution of problem (3.5) is $y_1 = 4$, and the optimal value is $6y_1 = 24$.

3.3 Symmetric Positive Semidefinite (SPSD) Matrix

- If X is an $n \times n$ matrix, then X is a symmetric positive semidefinite ($SPSD$) matrix if $X = X^T$ and $v^T X v \geq 0$ for any $v \in R^{\mathrm{n}}$

- If X is an $n \times n$ matrix, then X is a symmetric positive definite (SPD) matrix If $X = X^T$ and $v^T X v > 0$ for any $v \in R^n, \quad v \neq 0$

- S^n denote the set of a symmetric $n \times n$ matrices.

- S^n_+ denote the set of positive semidefinite (SPSD)we have $n \times n$ matrices.

- S^n_{++} denote the set of (SPD) we have $n \times n$ matrices. Let X and Y belong to S^n

Let X and Y be any symmetric matrices. We write " X $\succeq$ 0 " to denote that X is symmetric and positive semidefinite, and we write " X $\succeq Y$ " to denote that $X - Y \succeq 0$. We write " X $\succ$ 0 " to denote that X is symmetric and positive definite, etc.

3.4 Duality

Given a particular linear programming problem, there is another (unique) linear program called its dual. In general, duality theory addresses itself to the study of the connection between two related linear programming problems, where one of them, the primal, is a maximization (minimization) problem and the other, the dual, is a minimization (maximization) problem . Moreover, this association is such that the existence of an optimal solution to any one of these two problems guarantees an optimal solution to the other, with the result that their extreme values are equal . To gain some additional insight into the role of duality in linear programming, we note briefly that the solution to any given linear programming problem may be obtained by applying the simplex method to either its primal or dual formulation, the reason being that the simplex process generates an optimal solution to the primal-dual pair of problems simultaneously. So with the optimal solution to one of these problems obtainable from the optimal solution of its dual, there is no reason to solve both problems separately [28]. Throughout this section we consider the primal program in standard form:

$$\begin{cases} \text{Maximize } z = c^T.x \\ \text{subject to } Ax \leq b \\ \qquad x \geq 0 \end{cases} \tag{3.6}$$

and its corresponding dual.

$$\begin{cases} \text{Minimize } z = b^T w \\ \text{subject to } A^T w \geq c \\ \qquad w \geq 0 \end{cases} \tag{3.7}$$

Example 3.4.1. Find the dual of the following linear programming problem.

$$\begin{cases} \text{Maximize} \quad z = 2x_1 - 3x_2 \\ \text{subject to } -2x_1 - x_2 \leq 1 \\ \qquad 3x_1 - 2x_2 \leq -1 \\ \qquad 5x_1 + 2x_2 \leq 2 \\ \qquad x_1 \geq 0, x_2 \geq 0 \end{cases}$$

The dual of the linear programming problem is:

$$\begin{cases} \text{Minimize} \quad z = y_1 - y_2 + 2y_3 \\ \text{subject to } -2y_1 + 3y_2 + 5y_3 \geq 2 \\ \qquad -y_1 - 2y_2 + 2y_3 \geq -3 \\ \qquad y_1 \geq 0, y_2 \geq 0, y_3 \geq 0 \end{cases}$$

3.5 The Lagrangian Duality

The standard form of the nonlinear optimization problem is given by:

$$(P) = \begin{cases} \text{Minimize} \quad f_0(x) \\ \text{subject to} \quad f_i(x) \leqslant 0, \quad i = 1, \ldots, m \\ \qquad h_i(x) = 0, \quad i = 1, \ldots, p \\ \qquad x \in X \end{cases} \tag{3.8}$$

We define the Lagrangian function as:

$$L(x, \lambda, v) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} v_i h_i(x) \tag{3.9}$$

where $\lambda \in R^m$ and $v \in R^p$ are the dual variables (Lagrange multipliers). Then we define the Lagrangian dual function to be

$$\psi(\lambda, v) = \min_{x \in X} L(x, \lambda, v) = inf(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} v_i h_i(x)) \tag{3.10}$$

We denote the optimal value of the Lagranian dual problem by d^*, the optimal value of primal problem by p^*, and x^* is an optimal solution of (P). The Lagrange dual problem with dual variables (λ, v) is given by

$$(\text{D}) = \begin{cases} \text{Maximize} \quad \psi(\lambda, v) \\ \text{subject to} \quad \lambda \geq 0 \end{cases} \tag{3.11}$$

The dual function yields lower bounds on the optimal value p^* of the problem (3.8); that is, for any $\lambda \succeq 0$ and any v we have

$$\psi(\lambda, v) \leq p^* \tag{3.12}$$

To prove this property (3.12) [22], let $\lambda \succeq 0$ and v be given, and let x^* be a feasible point for the problem (3.8) (i.e., $f_i(x^*) \leq 0$, for all i, and $h_i(x^*) = 0$, for all i).Then we have

$$\sum_{i=1}^{m} \lambda_i f_i(x^*) + \sum_{i=1}^{p} v_i h_i(x^*) \leq 0 \tag{3.13}$$

since it is clear each term in the first sum is nonpositive, and each term in the second sum is zero. Thus,

$$L(x^*, \lambda, v) = f_0(x^*) + \sum_{i=1}^{m} \lambda_i f_i(x^*) + \sum_{i=1}^{p} v_i h_i(x^*) \leq f_0(x^*) \tag{3.14}$$

Therefore,

$$\psi(\lambda, v) = \min_{x \in \Omega} L(x, \lambda, v) \leq L\left(x^*, \lambda, v\right) \leq f_0\left(x^*\right) \leq p^* \tag{3.15}$$

for every feasible point x^*, which proves the above property. Since we have a lower bound that depends on the parameters λ, v an important question is: What is the best lower bound that can be obtained from the Lagrangian dual function? This leads to the Lagrangian dual optimization problem(3.11). It is a convex optimization problem, since the objective to be maximized is concave and the constraint is convex [13]. The lower bound (3.12) is illustrated in Figure 3.1, for a simple problem with $x \in R$ and one inequality constraint. The inequality (3.12)holds, but is vacuous, when $\psi(\lambda, v) = -\infty$. The dual function gives a nontrivial lower bound on p^* only when $\lambda \succeq 0$ and $(\lambda, v) \in S_n$, i.e., $\psi(\lambda, v) > -\infty$. We refer to a pair (λ, v) with $\lambda \succeq 0$ and $(\lambda, v) \in S_n$ as dual feasible, for reasons that will become clear later [10].

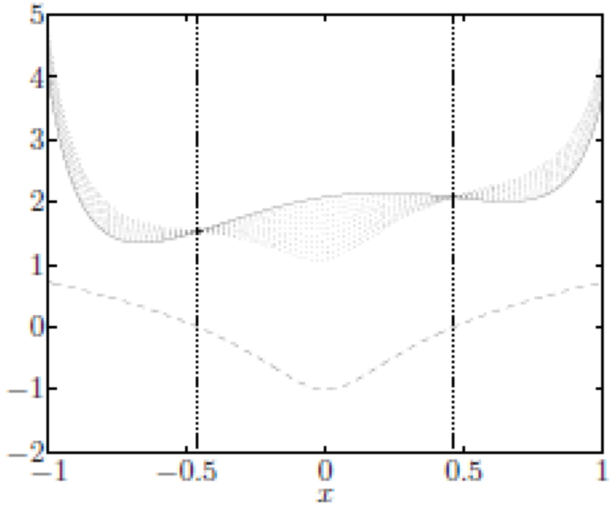


Figure 3.1: Lower Bound From a Dual Feasible Point.

3.6 Semidefinite Programming (SDP)

Semidefinite programming (SDP) is the most exciting development in mathematical programming inthe1990's. SDP has applications in such diverse fields as traditional convex constrained optimization, control theory, and combinatorial optimization [82]. Because SDP is solvable via interior point methods, most of these applications can usually be solved very efficiently in practice as well as in theory. Since Semidefinite programming deals with matrixes and not with vectors, this is considered one of the problems of NP-Hard.

3.7 Semidefinite Programming in Convex Optimization

As stated above, SDP has very wide applications in convex optimization. The types of constraints that can be modeled in the SDP frame work include: linear inequalities, convex quadratic inequalities, lower bounds on matrix norms, lower bounds on determinants of symmetric positive semidefinite matrices, lower bounds on the geometric mean of a nonnegative vector, plus many others. Using these and other constructions, the following problems (among many others) can be cast in the form of a semidefinite program: linear programming, optimizing a convex quadratic form subject to convex quadratic inequality constraints, minimizing the volume of an ellipsoid that covers a given set of points and ellipsoids, maximizing the volume of an ellipsoid that is contained in a given polytope, plus a variety of maximum eigenvalue and minimum eigenvalue problems ([51], [22]). In the subsections below we demonstrate how some important problems in convex optimization can be re-formulated as instances of SDP.

3.7.1 The General Formula of SDP

Let $X \in S^n$. We can think of X as a matrix, or equivalently, as an array of n^2 components of the form $(x_{11}, \ldots, x_{nn})$. We can alsojust think of X as an object(a vector) in the space S^n [38].If $C(X)$ is a linear function of X, then $C(X)$ can be written as $C.X$, where

$$\mathrm{C}.X = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij} \tag{3.16}$$

There is no loss of generality in assuming that the matrix C is also symmetric. The standard general inequality of semidefinite programming (SDP) problem is given by:

$$(SDP) \begin{cases} \min & C.X \\ \text{subject to} & A_i.X \geq b_i, \forall i = 1, \ldots, m \\ & X \succeq 0 \end{cases} \tag{3.17}$$

Notice that in an SDP that the variable is the matrix X,. The objective function is the linear function $C.X$ and there are m linear equations that X must satisfy, namely $A_i.X = b_i,\ i = 1, \ldots, m$. The variable X also must lie in the(closed convex) cone of positive semidefinite symmetric matrices S^n_+. Note that the data for SDP consists of the symmetric matrix C (which is the data for the objective function)and the m symmetric matrices $A_1, \ldots, A_m$, and the m-vector b, which form the m linear equations. Then define:

$$A_i = \begin{pmatrix} a_{i1} & 0 & \ldots & 0 \\ 0 & a_{i2} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{in} \end{pmatrix}, i = 1, \ldots, m, \text{ and } C = \begin{pmatrix} c_1 & 0 & \ldots & 0 \\ 0 & c_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & c_3 \end{pmatrix}$$

Then equality of semidefinite programming (SDP) problem is given by:

$$(SDP) \begin{cases} \text{Minimize} & C.X \\ \text{subject to} & A_i X = b_i, \quad i = 1, \ldots, m \\ & x_{ij} = 0, i = 1, \ldots, n, j = i+1, \ldots, n \\ & X \geq 0 \end{cases}$$

with the association that:

$$X = \begin{pmatrix} x_{11} & 0 & \ldots & 0 \\ 0 & x_{22} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & x_{nn} \end{pmatrix} \succeq 0$$

Of course,in practice one would never want to convert an instance of LP into an instance of SDP. The above construction merely shows that SDP includes linear programming as a special case. Then we get LP$\subset$ SDP (See the Figure3.2)

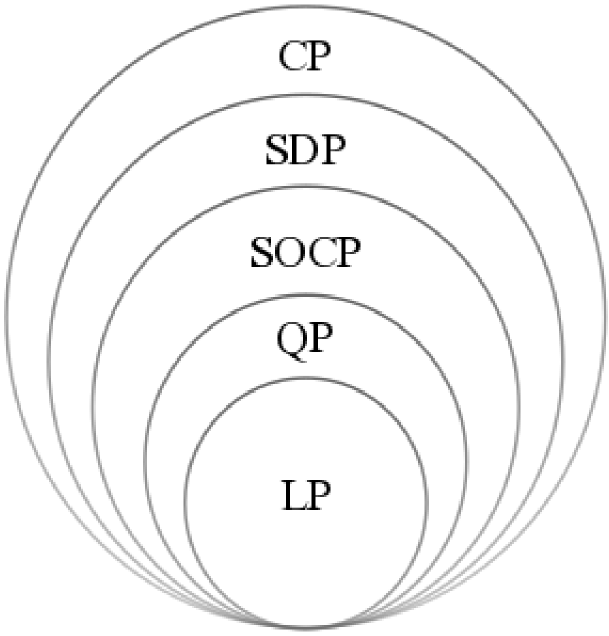


Figure 3.2: A Hierarchy of Convex Optimization Problems.

Let us see an example of an SDP for $n = 3$ and $m = 2$. Define the following matrices:

$$A_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 3 & 7 \\ 1 & 7 & 5 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 2 & 8 \\ 2 & 6 & 0 \\ 8 & 0 & 4 \end{bmatrix}, b = \begin{bmatrix} 11 \\ 9 \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 9 & 0 \\ 3 & 0 & 7 \end{bmatrix}$$

The variable X will be the 3×3 symmetric matrix:

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}$$

and so, for example,

$C.X = x_{11} + 2x_{12} + 3x_{13} + 2x_{21} + 9x_{22} + 0x_{23} + 3x_{31} + 0x_{32} + 7x_{33}$

$= x_{11} + 4x_{12} + 6x_{13} + 9x_{22} + 0x_{23} + 7x_{33}$

Then the semidefinite programming (SDP) problem is :

$$(SDP)\begin{cases} \min & x_{11} + 4x_{12} + 6x_{13} + 9x_{22} + 0x_{23} + 7x_{33} \\ \text{subject to} & x_{11} + 0x_{12} + 2x_{13} + 3x_{22} + 14x_{23} + 5x_{33} = 11 \\ & 0x_{11} + 4x_{12} + 16x_{13} + 6x_{22} + 0x_{23} + 4x_{33} = 9 \end{cases}$$

$$x = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \succeq 0$$

It may be helpful to think of " $X \succeq 0$ " as stating that each of the n eigenvalues of X must be nonnegative. It is easy to see that a linear program LP is a special instance of an SDP.

3.7.2 Semidefinite Programming Duality

The dual problem semidefinite programming of SDP problem is given by: (SDD)

$$\begin{cases} \max & y_i b_i \\ \text{subject to} & \sum_{i=1}^{m} y_i A_i + S = C, \quad C \in R^n and \quad S \in R^{n\times n} \\ & S \succeq 0 \end{cases} \tag{3.18}$$

One convenient way of thinking about this problem is as follows. Given multipliers $y_1, \dots, y_m$, the objective is to maximize the linear function $\sum_{i=1}^{m} y_i b_i$. The constraints of SDD state that the matrix S defined as

$$S = C - \sum_{i=1}^{m} y_i A_i$$

must be positive semidefinite. That is,

$$C - \sum_{i=1}^{m} y_i A_i \succcurlyeq 0$$

We illustrate this construction with the example presented earlier. The dual problem is:

$$(SDD)\begin{cases} \text{Maximize } 11y_1 + 9y_2 \\ \text{Subject to } y_1 \begin{bmatrix} 1 & 0 & 3 \\ 0 & 3 & 7 \\ 1 & 7 & 5 \end{bmatrix} + y_2 \begin{bmatrix} 0 & 2 & 8 \\ 2 & 6 & 0 \\ 8 & 0 & 4 \end{bmatrix} + S \\ = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 9 & 0 \\ 3 & 0 & 7 \end{bmatrix} \\ \text{y} \succeq 0 \end{cases}$$

$$(SDD)\begin{cases} \text{Maximize } 11y_1 + 9y_2 \\ \text{Subject to } y_1 \begin{bmatrix} 1-1y_1-0y_2 & 2-0y_1-2y_2 & 3-1y_1-8y_2 \\ 2-0y_1-2y_2 & 9-3y_1-6y_2 & 0-7y_1-0y_2 \\ 3-1y_1-8y_2 & 0-7y_1-0y_2 & 7-5y_1-4y_2 \end{bmatrix} \succeq 0 \end{cases}$$

It is often easier to ”see” and to work with a semidefinite program when it is presented in the format of the dual SDD, since the variables are the m multipliers $y_1, \ldots, y_m$. As in linear programming, we can switch from one format of SDP (primal or dual) to any other format with great ease, and there is no loss of generality in assuming a particular specific format for the primal orthe dual. The following theorem states that weak duality must hold for the primal and dual of SDP:

Theorem 3.7.1. *Given a feasible solution X of SDP and a feasible solution (y, S) of SDD, the duality gap is $C.X - \sum_{i=0}^{m} y_i b_i = s.\ x \geq 0$ If $C.X - \sum_{i=0}^{m} y_i b_i = 0$, then X and (y, S) are each optimal solutions to SDP and SDD, respectively, and furthermore, $S.X = 0$. In order to prove Proposition It will be convenient to work with the trace of a matrix, defined below:*

$$\operatorname{trace}(M) = \sum_{j=1}^{n} M_{jj}$$

Simple arithmetic can be used to establish the following two elementary identities:

- $\operatorname{trace}(MN) = \operatorname{trace}(NM)$
- $A.B = \operatorname{trace}\left(A^T B\right)$

Proof :

For the first part of the theorem, we must show that if $S \geqslant 0$ and $x \geqslant 0$, then $S.X \geq 0$ Let $S = PDP^T$ and $X = QEQ^T$ where P, Q are orthonormal matrices and D, E are nonnegative diagonal matrices. We have:

$$\begin{aligned} \text{S.X } &= \operatorname{trace}\left(S^T X\right) = \operatorname{trace}(SX) = \operatorname{trace}\left(PDP^T QEQ^T\right) \\ &= \operatorname{trace}\left(DP^T QEQ^T P\right) = \sum_{j=1}^{n} D_{jj}\left(P^T QEQ^T P\right)_{jj} \geq 0 \end{aligned}$$

where the last inequality follows from the fact that all $D_{jj} \geq 0$ and the fact that the diagonal of the symmetric positive semidefinite matrix $P^T QEQ^T P$ must be nonnegative. To prove the second part of the theorem, suppose that trace $(SX) = 0$. Then from the above equalities,we have

$$\sum_{j=1}^{n} D_{jj}\left(P^T QEQ^T P\right)_{jj} = 0$$

However,this implies that for each $j = 1, \ldots, n$, either $D_{jj} = 0$ or the $\left(P^T QEQ^T P\right)_{jj} = 0$ Furthermore, the latter case implies that the j^{th} row of $P^T QEQ^T P$ is all zeros. Therefore $DP^T QEQ^T P = 0$, and so $SX = PDP^T QEQ^T = 0$.■

3.8 The Weak and Strong Duality of Linear Programming

In this section we will discuss the weak and strong duality of the linear programming

3.8.1 The Weak Duality

The optimal value of the Lagrange dual problem, which we denote d^*, is by definition the best lower bound on p^* that can be obtained from the Lagrange dual function [43]. In particular, we have the simple but important inequality.

$$d^* \leq p^* \tag{3.19}$$

which holds even if the original problem is not convex. This property is called weak duality. The weak duality inequality (3.19) holds when d^* and p^* are infinite. For example, if the primal problem is unbounded below, so that $p^* = -\infty$, we must have $d^* = -\infty$, i.e., the Lagrange dual problem is infeasible. Conversely, if the dual problem is unbounded above, so that $d^* = \infty$, we must have $p^* = \infty$, i.e., the primal problem is in feasible. We refer to the difference $p^* - d^*$ as the optimal duality gap of the original problem, since it gives the gap between the optimal value of the primal problem and the best (i.e., greatest) lower bound on it that can be obtained from the Lagrange dual function. The optimal duality gap is always nonnegative. The bound (3.19) can sometimes be used to find a lower bound on the optimal value of a problem that is difficult to solve, since the dual problem is always convex, and in many cases can be solved efficiently, to find d^*.

3.8.2 The Strong Duality

If the equality

$$d^* = p^* \tag{3.20}$$

holds,i.e., the optimal duality gap is zero, then we say that strong duality holds. This means that the best bound that can be obtained from the Lagrange dual function is tight. Strong duality does not, in general hold [19]. But if the primal problem (3.8) is convex, i.e., of the form

$$\begin{cases} \text{Maximize} \quad f_0(x) \\ \text{subject to} \quad f_i(x) \leq 0, \quad i = 1, \ldots, m \\ \qquad\qquad Ax = b for X \in R^n, X \geq 0 \end{cases} \tag{3.21}$$

with $f_0, \ldots, f_m$ convex, we usually (but not always) have strong duality. There are many results that establish conditions on the problem, beyond convexity, under which strong duality holds. These conditions are called constraint qualifications. One simple constraint qualification is Slater's condition: There exists an $X \in R^n$ such that.

$$f_i(x) < 0, \quad i = 1, \ldots, m, \quad Ax = b \tag{3.22}$$

Such a point is sometimes called strictly feasible, since the inequality constraints hold with strict inequalities.If the first k constraint functions $f_0, \ldots, f_k$ are affine function, then strong duality holds provided the following weaker condition holds: There exists an $x \in R^n$ with

$$f_i(x) < 0, \quad i = 1, \ldots, k, \quad f_i(x) < 0, \quad i = k+1, \ldots, m, \quad Ax = b \tag{3.23}$$

In other words, the affine inequalities do not need to hold with strict inequality. Note that the refined Slater condition (3.23) reduces to feasibility when the constraints are all linear equalities and inequalities, and dom f_0 is open. Slater's condition and the refinement (3.23) not only implies strong duality for convex problems. It also implies that the dual optimal value is attained when d* $> -\infty$, i.e. there exists a dual feasible (λ^*, v^*) with $g(\lambda^*, v^*) = d^* = p^*$

3.8.3 Relation Between Weak Duality and Strong Duality:

We know for certain that:

$$q(\lambda^*, \mu^*) \leq f(x^*) \tag{3.24}$$

The maximum of the dual problem is always smaller or equal than the minimum of the primal."What we really want is that both are equal, so that by solving the dual, we have the solution for the primal and not just an approximation. When the duality gap is zero, we say the problem has a strong duality, otherwise it has a weak duality. The illustration below shows the main idea. The primal problem is a minimization with respect to x, while the dual problem is a maximization with respect to another variable λ. When there is strong duality, the optimal values $f(x^*)$ and $q(\lambda^*)$ coincide. Whenever there is weak duality, there is the so called duality gap between the two optimal values [27].

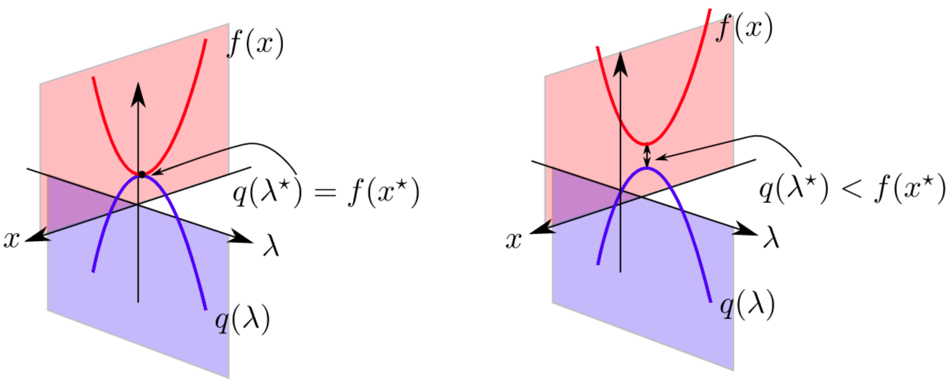


Figure 3.3: Relation Between Weak Duality and Strong Duality

Theorem 3.8.1. *Let* x* *be an optimal solution to the primal and* y* *to the dual where*

$$(\mathrm{P}) = \begin{cases} \mathrm{Max} \quad c.x \\ \\ s.t\ Ax \leq b \\ \\ x \geq 0 \end{cases} \quad and \quad (\mathrm{D}) = \begin{cases} \mathrm{Min}\, b.y \\ \\ A^T y \geq c \\ \\ y \geq 0 \end{cases}$$

Then $c \cdot x^* = b \cdot y^*$

3.9 The Weak and Strong Duality of (SDP):

In this section we will review some basic concepts of primal (SDP)and (SDD) was give the good bound of the Max independent problem.

Theorem 3.9.1. ***Weak duality*** *[22] If* x^* *is a feasible solution for SDP and* y^* *is a feasible solution for SDD, then* $\langle c, x^* \rangle \geq \langle b, y^* \rangle$.

Proof.

$$\begin{aligned} \langle c, x^* \rangle - \langle b, y^* \rangle &= \langle A^* y^* + s, x^* \rangle - \langle b, \mathrm{y}^* \rangle \\ &= \langle A^* y^*, x^* \rangle + \langle s, x^* \rangle - \langle b, y^* \rangle \\ &= \langle A x^*, y^* \rangle + \langle s, x^* \rangle - \langle b, y^* \rangle \\ &= \langle b, y^* \rangle + \langle s, x^* \rangle - \langle b, y^* \rangle \\ &= \langle s, x^* \rangle \end{aligned}$$

□

Where $\langle s, x^* \rangle \geq 0$. It follows $\langle c, x^* \rangle - \langle b, y^* \rangle \geq 0$

Implies $\langle c, x^* \rangle \geq \langle b, y^* \rangle$

Theorem 3.9.2. ***Strong duality for the primal*** *[22] If there exists* x_0 *strictly feasible for (SDP), then there exists an optimal Solution* y* *of (SDD) and* $\langle b, y^* \rangle = p^*$.

Theorem 3.9.3. ***Strong duality for the dual*** *[22] If there exists y_0 strictly feasible for (SDD), then there exists an optimal solution x^* for (SDP) and $\langle c, x^* \rangle = d^*$.*

Theorem 3.9.4. ***Primal and dual strong duality*** *[22] If there exists x_0 which is a feasible for(SDP) and (y_0, S_0) which is a feasible for (SDD) such that $x_0 > 0$ and $S_0 > 0$, then there exists an optimal solution x for (SDP) and an optimal solution (y, S) for (SDD) and $\langle c, x^* \rangle - \langle \mathrm{b}, y^* \rangle = \langle S^*, x^* \rangle = 0$.*

3.10 Quadratic Programming

Quadratic programming (QP) is the problem of optimizing a quadratic objective function and is one of the simplest form of non-linear programming [18]. The objective function can contain bilinear or up to second order polynomial terms, and the constraints are linear and can be both equalities and inequalities. QP is widely used in image and signal processing, to optimize financial portfolios, to perform the least-squares method of regression, to control scheduling in chemical plants, and in sequential quadratic programming, a technique for solving more complex non-linear programming problems. The problem was first explored in the early 1950s, most notably by Princeton University's Wolfe and Frank, who developed its theoretical background,1 and by Markowitz, who applied it to portfolio optimization, a subfield of finance. A

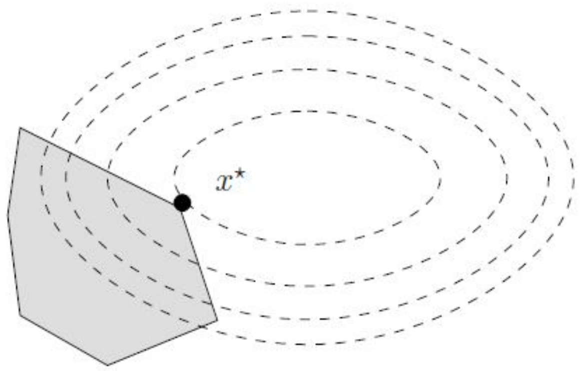


Figure 3.4: Optimization of a Quadratic Function

quadratic program (QP) is written in standard form as:

$$\begin{cases} \text{Minimize} & \frac{1}{2}x^T P x + q^T x \\ \text{Subject to} & Gx \preceq h \\ & Ax = b \end{cases}$$

Primal problem (assume $P \in S_+^n$, so objective is convex quadratic)

$$(\text{P}) = \begin{cases} \text{Minimize} & x^T P x \\ \text{Subject to} & Ax \preceq b \end{cases}$$

Dual function

$$g(\lambda) = \inf_x \left(x^T P x + \lambda^T (Ax - b) \right) = -\frac{1}{4}\lambda^T A P^{-1} A^T \lambda - b^T \lambda$$

Dual problem

$$(\text{D}) = \begin{cases} \text{Maximize} & -\frac{1}{4}\lambda^T A P^{-1} A^T \lambda - b^T \lambda \\ \text{Subject to} & \lambda \succeq 0 \end{cases}$$

3.10.1 Binary Quadratic Optimization Problems

We consider binary quadratic optimization problems, i.e.,(nonconvex) optimization problems with a quadratic objective, quadratic constraints, and 0–1 variables. A binary quadratic problem with m_1 inequality constraints and m_E equality constraints has the following mathematical formulation:

$$
(\text{BQP}) \begin{cases} \text{maximize} & z^T S_0 z + S_0^T z \\ \text{subject to} & z^T S_i z + s_i^T z \le a_i, \quad i \in \{1, \dots, m_I\} \\ & z^T N_i z + n_i^T z = b_i, \quad i \in \{m_I + 1, \dots, m_I + m_E\} \\ & z \in \{0,1\}^n \end{cases} \tag{3.25}
$$

where the S_i and N_i are real symmetric $n \times n$ matrices S_i and n_i are vectors in R^n, and the a_i and b_i are real numbers. Many combinatorial optimization problems can be expressed as (3.25), such as the MAX-CUT, Max-k-Cluster, and Max-Independent Set problems. In our research, we will focus on the Max-Independent Set problems. In the Max-Independent-Set (MIS) problem [53], we are given a graph $G = (V, E)$ with vertex weights w_i and the objective is to maximize the total weight of the vertices in an independent set (a set S of vertices having no two vertices joined by an edge in E); this problem can be stated as:

$$
(\text{MIS}) \begin{cases} \text{Maximiz} & \sum_i w_i z_i \\ \text{subject to} & z_i z_j = 0, \forall (i,j) \in E \\ & z \in \{0,1\}^n \end{cases}
$$

These three problems, and more generally binary quadratic problems, are NP-hard and are often difficult to solve in practice.

3.10.2 Constrained Optimization Problems

A general formulation for nonlinear constrained optimization is

$$\begin{cases} \min\limits_{x \in R^n} & f(x) \\ \text{subject to} & c_i(x) = 0, \quad i = 1, \ldots, m_e \\ & c_i(x) \geq 0, \quad i = m_e + 1, \ldots, m \end{cases} \tag{3.26}$$

where the objective function $f(x)$ and the constrained functions $c_i(x) = 0, i = 1, \ldots, m)$ are all smooth, realvalued functions on $\mathrm{R^n}$ and at least one is nonlinear, m_e and m are nonnegative integers with $0 \leq m_e \leq m$. Sometimes, we set

$$E = \{1, \ldots, m_e\} \text{ and } I = \{m_e + 1, \ldots, m\}$$

as index sets of equality constraints and inequality constraints, respectively. If $m = 0$, the problem (3.26) is an unconstrained optimization problem; if $m_e = m \neq 0$, the problem is called an equality constrained optimization problem; if all $c_i(x), i = 1, \cdots, m$) are linear functions, the problem (3.26) is called a linearly constrained optimization problem. A linearly constrained optimization problem with quadratic objective function $f(x)$ is said to be a quadratic programming problem .

Definition 3.10.1. The point $x \in R^n$ is said to be a feasible point if and only if the problem (3.26) hold. The set of all feasible points is said to be a feasible set.

3.10.3 Optimality Conditions for Quadratic Programming

Quadratic programming is the simplest constrained nonlinear optimization problem. It is a special class of optimization problem (3.26) with a quadratic objective function f(x) and linear constraints $c_i(\mathrm{x}), \forall (i = 1, \cdots, m)$. The general quadratic programming (QP) has the following form

$$\begin{cases} \text{Minimiz} \quad Q(x) = \dfrac{1}{2}x^T G x + g^T x \\ \text{subject to} \quad a_i^T x = b_i, \quad i \in E \\ \qquad\qquad a_i^T x = b_i, \quad i \in I \end{cases} \tag{3.27}$$

where G is a symmetric $n \times n$ matrix, E and I are finite sets of indices, $E = \{1, \cdots, me\}$ and $I = \{m_e + 1, \cdots \ , m \ \}$. If the Hessian matrix G is positive semidefinite, then the problem (3.27) is a convex quadratic programming problem and the local solution x* is a global solution. If G is positive definite, then the problem (3.27) is a strict convex QP and x* is a unique global solution. If G is indefinite, then the problem (3.27) is a non convex QP which is more important and worth emphasizing.

Theorem 3.10.1. *(**Necessary conditions**) [30] Let x^* be a local minimizer of quadratic programming problem* (3.27) *Then there exist multipliers $\lambda^*(i = 1, \cdots, m \ )$ such that*

$$g + Gx^* = \sum_{i=1}^{m} \lambda_i^* a_i \tag{3.28}$$

$$a_i^T x^* = b_i, \quad i \in E \tag{3.29}$$

$$a_i^T x^* \geq b_i, \quad i \in I \tag{3.30}$$

$$\lambda_i^* \left(a_i^T x^* - b_i \right) = 0, \quad i \in I \tag{3.31}$$

$$\lambda_i^* \geq 0, \quad i \in I \tag{3.32}$$

Furthermore,

$$\mathrm{d^T G d \geq 0, \forall d \in G\left(x^*, \lambda^*\right)}$$

$$G\left(x^*, \lambda^*\right) = \left\{ d \neq 0 \;\middle|\; \begin{array}{c} d^T a_i = 0, \quad i \in E \\ \\ d^T a_i \geq 0, \quad i \in I\left(x^*\right) \\ \\ d^T a_i = 0, \quad i \in I\left(x^*\right) \\ \\ \text{and } \lambda_i^* > 0 \end{array} \right.$$

Theorem 3.10.2. *(**Sufficient conditions**) [30] Let* x^* *be a* KKT *point and* λ^* *a corresponding Lagrange multiplier vector. If* $d^T G d > 0, \forall 0 \neq d \in G\left(x^*, \lambda^*\right)$ *then* x^* *is a strict local minimizer to the problem* (3.27) *Next, we give a sufficient and necessary optimality condition for the problem* (3.27).

Theorem 3.10.3. *(**Necessary and sufficient conditions**) [30] Let* x^* *be a feasible point of quadratic programming problem the problem* (3.27)*, then* x^* *is a local minimizer if and only if* $\left(\mathrm{x}^*, \lambda^*\right)$ *is a KKT pair such that the problems* (3.28)*,* (3.32) *hold, and*

$$d^T G d \geq 0, \forall d \in G\left(x^*, \lambda^*\right) \tag{3.33}$$

3.11 NP-Hardnees

In modern computer science, there are many famous problems that are not inevitably solved, such type of problems are called a NP-Hard [77]. A problem is NP-hard if all problems in NP are polynomial time reducible to it, even though it may not be in NP itself. Problem is NP-hard if for all $L^* \in NP, L^* \leq L$. Thus if we can solve L in polynomial time, we can solve all NP problems in polynomial time. NP problems are not only hard to solve but are hard to verify as well. In fact, some of these problems aren't even decidable.

These algorithms have a property similar to NP complete problems.

1. P represents the set of problems that can be answered with a yes or no and that can be solved in polynomial time.
2. NP represents a set of problems that can be answered with yes or no with the following property: If the answer is: Yes, the proof of this property can be verified in polynomial time.
3. NP-hard is one of the important and difficult problems that cannot be resolved in a definitive way.
4. NP-complete problems are from the hardest problems in NP.

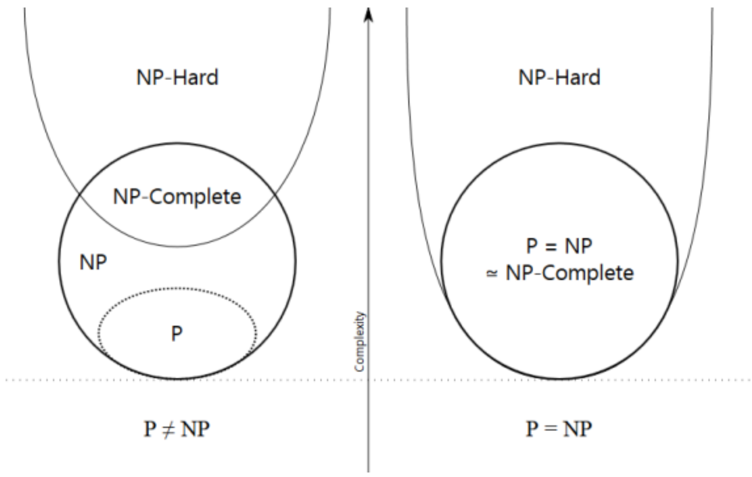


Figure 3.5: NP-Hard Problems

Definition 3.11.1. We say x is NP complete problem if:

$$\mathrm{x} \in \mathrm{NP}, \forall \mathrm{y} \in \mathrm{NP}, \mathrm{y} \leqslant \mathrm{x}$$

If these hold then x can be used to solve every problem in NP. therefore,x is definitely at least as hard as every problem in NP

Theorem 3.11.1. *[77] if x is NP complete problem,then x is solvable in polynomial time if and only if P=NP*

Proof: if P=NP ,then x can be solved in polynomial time. Suppose x is solvable in polynomial time and let Y be any problem in N. we can solve Y in polynomial time :reduce it to x therefore every problem in NP has polynomial time algorithm and P=NP.

Theorem 3.11.2. *[37] If Y is NP-complete problem, and* X *is in NP and*Y $\leqslant X$ *Then x is NP-complete problem In other work. we can prove in a new problem is NP complete problem by reducing some other NP complete problem to it.*

Proof: let Z be any problem in NP. Since Y is NP- complete problem, $Z \leqslant Y$. by assumption $Y \leqslant X$. Therefore $Z \leqslant Y \leqslant X$.

3.12 The Difference Between NP hard and NP Complete Problems

3.12.1 NP-Problems

The NP problems set of problems whose solutions are hard to find but easy to verify and are solved by Non-Deterministic Machine in polynomial time.

3.12.2 NP-Hard Problems

A problem X is NP-Hard if there is an NP-Complete problem Y, such that Y is reducible to X in polynomial time. NP-Hard problems are as hard as NP Complete problems. NP-Hard Problem need not be in NP class.

3.12.3 NP-Complete Problems

A problem X is NP-Complete if there is an NP problem Y, such that Y is reducible to X in polynomial time. NP-Complete problems are as hard as NP problems. A problem is NP-Complete if it is a part of both NP and NP-Hard Problem. A non-deterministic Turing machine can solve NP-Complete problem in polynomial time. Take for example of an NP-hard problem is the optimization problem of finding the least cost cyclic route through each vertices of the graph. This is usually known as a problem as the traveling salesman problem [37]. See the Figure 3.6.

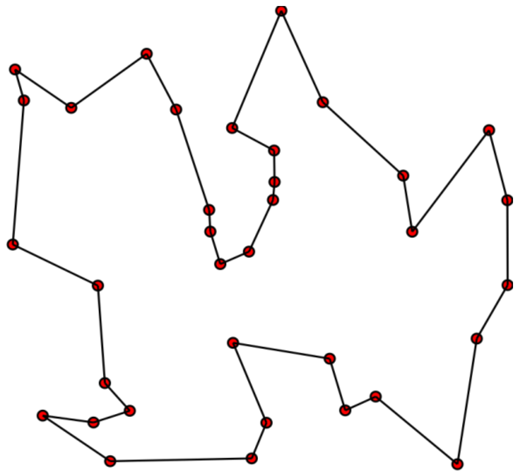


Figure 3.6: Travelling Salesman Problem

Also, there are many examples of NP-hard problem of them Knapsack problem ,graph coloring problem and hamiltonion circuit problem

3.13 The Relaxed Of Linear Programming Problems (LP-relaxation)

This relaxation technique transforms an NP-hard optimization problem (integer programming) into a related problem that is solvable in polynomial time (linear programming); the solution to the relaxed linear program can be used to gain information about the solution to the original integer program [79]. The basic form of linear programming is:

$$(\text{LP}) = \begin{cases} \text{Maximize} \quad \sum x_i \\ \text{Subject to} \quad x_i + x_j \leq 1, \quad \{i,j\} \subseteq E \\ 0 \leq x_i \leq 1 \text{ for } 1 \leq i \leq n \end{cases} \tag{3.34}$$

An integer linear program (ILP) is a linear program (LP) with the additional constraints that the variables must take integer values. For example, the following is an ILP:

$$(\text{LP}) = \begin{cases} \text{Maximize} \quad x_1 - x_2 + 2x_3 \\ \text{Subject to} \quad x_1 - x_2 \leq 1 \\ x_2 + x_3 \leq 2 \\ x_1, x_2, x_3 \in N \end{cases} \tag{3.35}$$

Where N is the set of natural numbers. In the concept of mathematics, relaxation of a linear program is the problem that arises by removing the integral constraint for each variable. For example, in an integer $0-1$ program, all constraints are of form

$$x_i \in \{0,1\}.$$

Instead, the original integer program mitigation uses a set of linear constraints

$$0 \leq x_i \leq 1$$

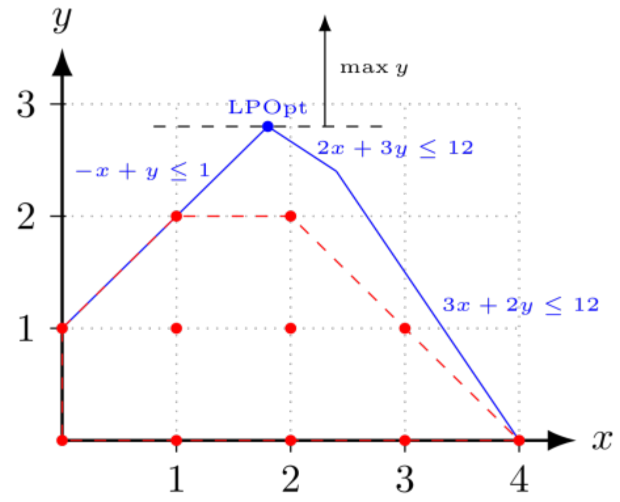


Figure 3.7: A (general) integer program and its LP-relaxation

3.14 The Lagrangian Relaxation of Liner Programming

Most of practical graph or network problems are NP-complete and hard to solve. In such a case, it may be interesting to solve a simplified problem to obtain approximations or bounds on the initial hardest problem [66]. Consider the following optimization problem where f : $R^n \rightarrow R$ and $S \subseteq R^n$:

$$\begin{cases} \text{Minimize} & f(x) \\ \text{subject to} & x \in S \end{cases}$$

A relaxation of the above problem has the following form:

$$\begin{cases} \text{Minimize} & f_R(x) \\ \text{subject to} & x \in S_R \end{cases}$$

where $f_R : R^n \to R$ is such that $f_R(x) \leq f(x)$ for any $x \in S$ and $S \subseteq S_R$. It is clear that the optimal solution f_R^* of the relaxation is a lower bound of the optimal solution of the initial problem. In previous section, the considered problems are such that $S = X \cap \{0,1\}^n$ where $X \subseteq R^n$ (or $X \subseteq Q^n$) and the fractional relaxation corresponds to consider $f_R = f$ and $S_R = X$. A large number of these problems have an underlying network structure. The idea of the Lagrangian Relaxation is to try to use the underlying network structure of these problems in order to use these efficient algorithms. The Lagrangian Relaxation is a method of decomposition: the constraints $S = S_1 \cup S_2$ of the problems are separated into two groups, namely the 'easy' constraints S_1 and the 'hard' constraints S_2. The hard constraints are then removed, i.e., $S_R = S_1$ and transferred into the objective function, i.e., f_R depends on f and S_2. Since S_R is a set of 'easy' constraints, it will be possible to solve the relaxation problem. Moreover, the interest of the Lagrangian relaxation is that, in some cases, the optimal solution of the relaxed problem actually gives the optimal solution of the initial problem.

Definition 3.14.1. decomposition method is a general term for solutions to various problems and algorithm design where the basic idea is to break down the problem into subproblems.

3.14.1 The Lagrangian Dual Relaxation Technique

In this section, we formally define the Lagrangian dual of an optimization problem and show that the solution of the Lagrangian dual provides a lower (resp., upper) bound of the initial minimization (resp., maximization) problem. Moreover, in the case of (convex) linear programmers, the optimal solution of the Lagrangian dual coincides with the optimal solution of the initial problem. Also, the bound obtained thanks to the Lagrangian relaxation is at least as good as the one obtained from fractional relaxation. Consider the following integer linear programme:

$$\begin{cases} \text{Minimize} & c^T x \\ \text{Subject to} & Ax = b \\ & x \in X \end{cases} \tag{3.36}$$

The Lagrangian relaxation procedure uses the idea of relaxing the explicit linear constraints by bringing them into the objective function with associated vector μ called the Lagrange multiplier. We refer to the resulting problem

$$\begin{cases} \text{Minimize} & c^T x + \mu^T (Ax - b) \\ \text{Subject to} & x \in X \end{cases} \tag{3.37}$$

as the Lagrangian relaxation or Lagrangian sub problem or the original problem (3.36), and we refer to the function

$$L(\mu) = \min \left\{ c^T x + \mu^T (Ax - b) \mid x \in X \right\}, \tag{3.38}$$

as the Lagrangian function.

Lemma 3.14.1. For any Lagrangian multiplier μ, the value $L(\mu)$ of the Lagrangian function is a lower bound on the optimal objective function value z^* of the original problem (3.36).

3.14.2 The Bounds and The Optimality Certificates

To obtain the sharpest possible lower bound, we would need to solve the following optimization problem

$$L^* = max_\mu L(\mu) \tag{3.39}$$

which we refer to as the Lagrangian Dual problem associated with the original optimization problem (3.36). The Lagrangian Bounding Principle has the following immediate implication.

Corollary 3.14.0.1. Weak duality the optimal solution L^* of the Lagrangian dual of problem (3.39) is a lower bound on the value z^* of an optimal solution of problem (3.39), i.e. $L^* \leq z^*$

Proof: For any Lagrangian multiplier μ and for any feasible solution x of problem (3.36), we have $L(\mu) \leq L^* \leq c^T x$.

Corollary 3.14.0.2. Optimality test. Let μ be a Lagrangian multiplier. If x is a feasible solution of (3.36) satisfving $L(\mu) = c^T x$. Then

1. $L(\mu)$ is the optimal value of the Lagrangian dual, i.e., $L^* = L(\mu)$, and
2. x is an optimal solution of the primal (3.36).

As indicated by the previous property and its corollary, the Lagrangian Bounding Principle immediately implies one advantage of the Lagrangian relaxation approach. It provides a certificate for guaranteeing that a given solution to the primal (3.36) is optimal.

CHAPTER 4

MAXIMAL INDEPENDENT SET

4.1 Introdiction

The maximum independent set problem is one of the central problems of combinatorial optimization, and the method of augmenting graphs is one of the general approaches to solve the problem [3]. The maximum independent set problem (MIS) is a classic graph optimization NP-hard problem with many real world applications. Given a graph G = (V,E), the independent set problem is that of finding a maximum-cardinality subset S of V such that no two vertices in S are adjacent. The MIS is a classic one in computer science and in graph theory, and is known to be NP-hard [77]. MIS has many important applications, including combinatorial auctions , graph coloring, coding theory [9], geometric tiling, fault diagnosis, pattern recognition, molecular biology, and more recently its application in bioinformatics have become important . The independent set S is the maximum if there is no other independent set containing S. Also the concept of maximum independent sets was used by Tsukiyama, Eddie, Ariyoshi and Shirakawa (1977) , Lawler, Linestra and Renoy Kahn (1980) and Johnson and Yanakakis (1988)..etc. The maximum independent sets problem is NP-hard in general graphs and remains difficult even under substantial restrictions, for instance, for cubic planar graphs (Gareyet al., 1976).

4.2 Independent set

Given an undirected graph G = (V,E) and an integer k does G contain a subset $V^{'}$ of k vertices such that no two vertices in $V^{'}$ are adjacent to one another.In graph theory, an independent set (stable set) is a set of vertices in a graph, no two of which are adjacent. That is, it is a set S of vertices such that for every two vertices in S, there is no edge connecting the two. Equivalently, each edge in the graph has at most one endpoint in S. The size of an independent set is the number of vertices it contains.

Independent sets have also been called internally stable sets.

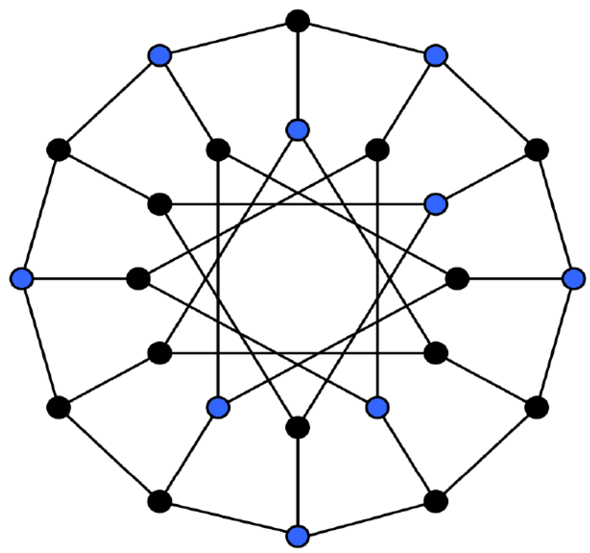


Figure 4.1: Independent Set

The nine blue vertices form a maximum independent set for the Generalized Petersen graph.For example,the graph G shown in Figure4.2 has an independent set (shown with shaded nodes) of size 3, but there is no independent set of size 4.Therefore $(G, 3) \in IS$ but $(G, 4) \notin IS$. The independent set problem arises when there is some sort of selection problem, but there are mutual restrictions pairs that cannot both be selected. (For example, you want to invite as many of your friends to your party, but many pairs do not get along, represented by edges between them, and you do not want to invite two enemies.)

Definition 4.2.1. (Bounded Independence): $G = (V, E)$ is of bounded independence, if for each hub $v \in V$ the biggest independent set in the neighborhood $N(v)$ is limited by a steady.

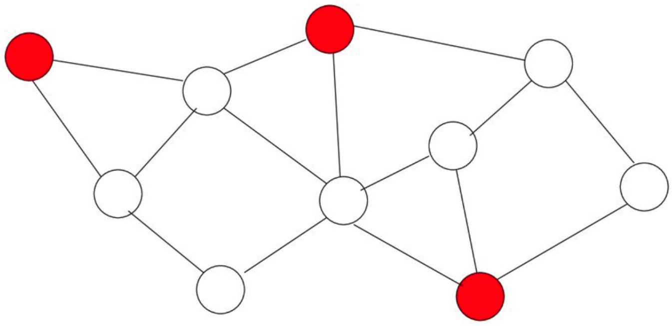


Figure 4.2: A Graph With an Independent Set of Size k = 3

4.3 A Maximal Independent Set (MIS):

Maximum Independent set is an emerging topic of the Independent set from Graph Theory. In Graph Theory, independent set is defined as a set of vertices that does not have an adjacency according to the acknowledged graph. From this definition two types of sets of interests emerge. One is the maximal independent set, and the other is maximum independent set([3], [74]). The maximal independent set is a set to which if you try to add another vertex, it would disrupt its independency. A maximum independent set is a maximal independent set that no other maximal independent set has more vertices in it. In graph theory, a maximal independent set (MIS) or maximal stable set is an independent set that is not a subset of any other independent set. In other words, there is no vertex outside the independent set that may join it because it is maximum with respect to the independent set property.
Mathematically, For a graph $G = (V, E)$ an independent set is S a maximum independent set if for $v \in V$, one of the following is true:

- $v \in S$
- $N(v) \cap S \neq \emptyset$ where $N(v)$ denotes the neighbors of v

The above can be restated as a vertex either belongs to the independent set or has at least one neighbor vertex that belongs to the independent set. As a result, every edge of the graph has at least one endpoint not in S. However, it is not true that every edge of the graph has at least one, or even one endpoint in S.

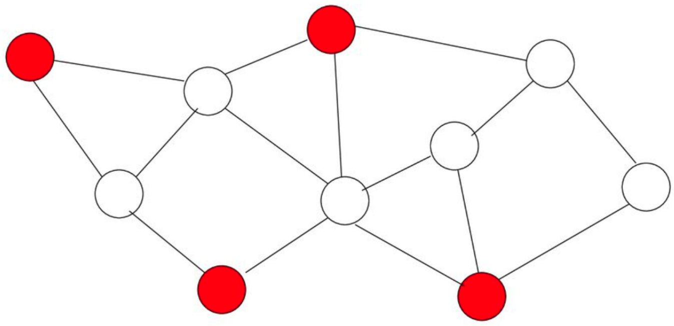


Figure 4.3: An example diagram of a Maximum Independent Set (MIS)

An example of an MIS problem is the poster display scheduling problem. Suppose students will present their work posters, where each poster can have more than one student as author and each student can have more than one poster. The goal is to customize as many posters as possible to adapt the same time slot so that each student can submit at most one poster in the same time period. We create a super graphic with a header for each poster and an edge for each student, then the maximum strong independent group represents the maximum number of posters that can be presented at the same time. In the

Figure 4.3, the red heads represent the stickers and the edges between them are the students so these red heads represent a simple example on the maximum independent sets.

Example 4.3.1. Through the following algorithm, we explain the difference between independent sets and maximal independent sets:

- Input: $V = 3, E = 0$
- Graph:

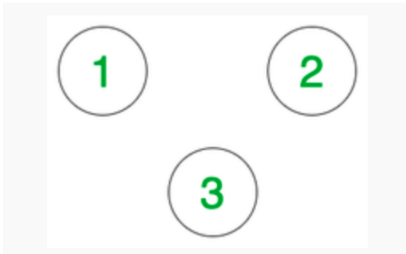


Figure 4.4: Graph for Example1

- Output:

  $L_1 : \{\}, \{1\}, \{12\}, \{13\}, \{2\}, \{123\}, \{3\}, \{23\},$

  $L_2 : \{123\},$

Example 4.3.2. Through the following algorithm, we explain the difference between independent sets and maximal independent sets :

- Input: $V = 4, E = 4$
- Graph:

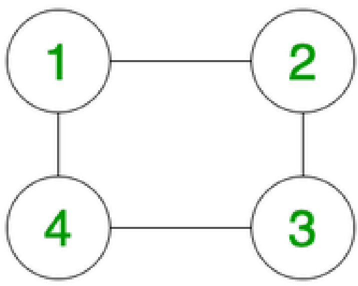


Figure 4.5: Graph for Example 2

- Output:

  $L_1 : \{\}, \{1\}, \{13\}, \{2\}, \{24\}, \{3\}, \{4\}$

  $L_2 : \{13\}, \{24\}$

The first line represents all the possible independent sets for the given graph. The second line has the maximal independent sets possible for the given graph.

4.4 The Minimum Vertex Cover (MVC)

Formally, a vertex cover $V^{'}$ of an undirected graph $G = (V, E)$ is a subset of V such that $uv \in E \Rightarrow u \in V^{'}, v \in V^{'}$ that is to say it is a set of vertices $V^{'}$ where every edge has at least one endpoint in the vertex cover $V^{'}$. Such a set is said to cover the edges of G. The Figures (4.6,4.7,4.8) shows two examples of vertex covers, with some vertex cover $V^{'}$ marked in red [29]. A minimum vertex cover is a vertex cover of smallest possible size. The vertex cover number Ψ is the size of a minimum vertex cover, i.e. $\psi = |V^{'}|$. The figure below shows examples of minimum vertex covers in the previous graphs. And a vertex cover optimization serves as a model for many real-world and theoretical problems [23]. For example, a commercial establishment interested in installing the fewest possible closed circuit cameras covering all hallways (edges) connecting all rooms (nodes) on a floor might model the objective as a vertex cover minimization problem.

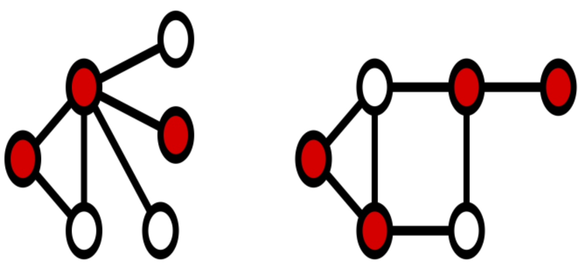


Figure 4.6: Examples of Vertex Covers

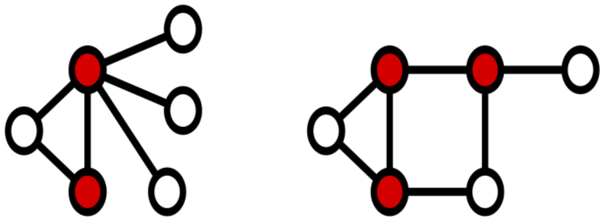


Figure 4.7: Examples of Minimum Vertex Covers

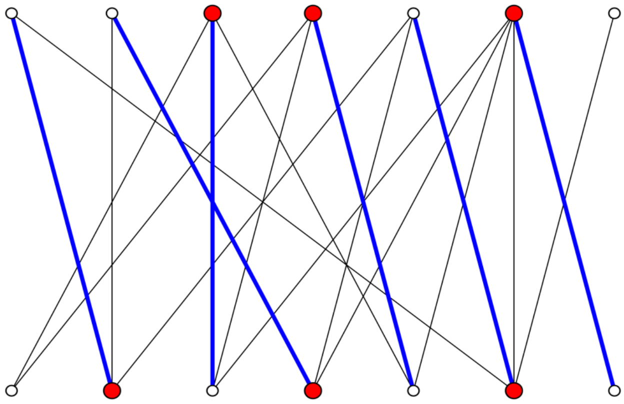


Figure 4.8: Minimum Vertex Covers (red) of Size Six

A set of vertices is a vertex cover if and only if its complement is an independent set. Consequently, the number of vertices of a graph is equal to its minimum vertex cover number plus the size of a maximum independent set. To prove that Vertex Cover is NP Hard, we take some problem which has already been proven to be NP Hard, and show that this problem can be reduced to the Vertex Cover problem. For this, we consider the Clique problem, which is NP Complete (and hence NP Hard). In computer science, the clique problem is the computational problem of finding cliques (subsets of vertices, all adjacent to each other, also called complete sub graphs) in a graph. Here, we consider the problem of finding out whether there is a clique of size k in the given graph. Therefore, an instance of the clique problem is a graph $G(V, E)$ and a non-negative integer k, and we need to check for the existence of a clique of size k in G. Now, we need to show that any instance (G, k) of the Clique problem can be reduced to an instance of the vertex cover problem. Consider the graph $G^{'}$ which consists of all edges not in G, but in the complete graph using all vertices in G. Let us call this the complement of G. Now, the problem of finding whether a clique of size k exists in the graph G is the same as the problem of finding whether there is a vertex cover of size $|V|$ $-k$ in $G^{'}$ .We need to show that this is indeed the case. Assume that there is a clique of size k in G. Let the set of vertices in the clique be $V^{'}$. This means $|V^{'}| = k$. In the complement graph $G^{'}$, let us pick any edge (u, v). Then at least one of u or v must be in the set $V - V^{'}$. This is because, if both u and v were from the set $V^{'}$, then the edge (u, v) would belong to $V^{'}$, which, in turn would mean that the edge (u, v) is in G. This is not possible since (u, v) is not in G. Thus, all edges in $G^{'}$ are covered by vertices in the set V - $V^{'}$. Now assume that there is a vertex cover $V^{''}$of size $|V|$– k in $G^{'}$. This means that all edges in $G^{'}$ are connected to some vertex in $V^{''}$. As a result, if we pick any edge (u, v) from $G^{'}$, both of them cannot be outside the set $V^{''}$. This means, all edges (u, v) such that both u and v are outside the set $V^{''}$ are in G, i.e., these edges constitute a clique of size k. Thus, we can say that there is a clique of size k in graph G if and only if there is a vertex cover of size $|V|$ – k in $G^{'}$, and hence, any instance of the clique problem can be reduced to an instance of the vertex cover problem. Thus, vertex cover is NP Hard. Since vertex cover is in both NP and NP Hard classes, it is NP Complete.See the following example diagram and its complement.

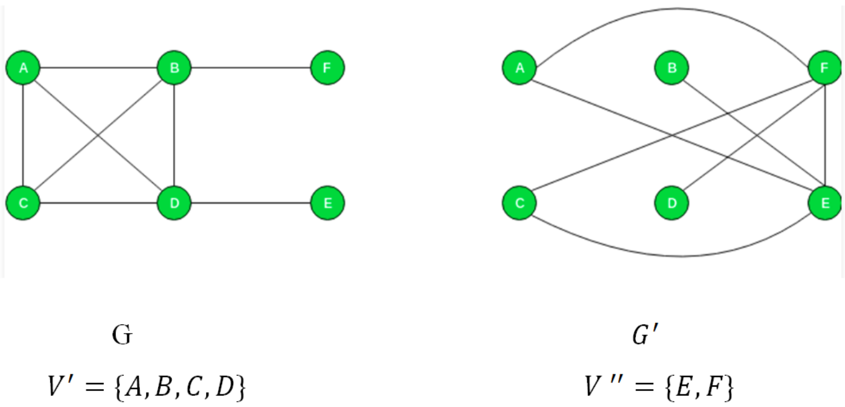
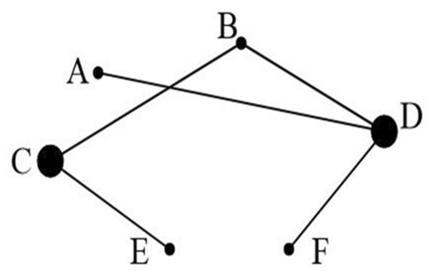


Figure 4.9: Diagram and its Complement.

4.5 The Maximum Independent Set (MIS) and The Minimum Vertex Cover (MVC)

Let $G = (V, E)$ be a subjective undirected diagram, where $V = \{1, 2, \ldots, n\}$ is the set of vertices and $E \subseteq V \times V$ (not in arranged sets) is the set of edges. Two distinct vertices u and v are called adjacent if they are connected by an edge, an independent set S of G is a subset of V whose elements are pairwise non-adjacent.The MIS problem seeks to find an independent set with large number of vertices. The size of the maximum independent set of G is the stability number of G and is denoted by α. A vertex cover for G is a subset V_c of V with the end goal that for each edge $(u, v) \in E$, at least one of u or v or both belongs to (VC). The (MVC) problem consists of identifying the vertex cover V_c of G which has minimum cardinality. The problems of (MIS) and MVC are related in that the maximum independent set S of G contains all those vertices that are not in the minimum vertex cover V_c of G. That is, it is $S = V - V_c$.



MVC = $\{C, D\}$ and MIS = $\{A, B, E, F\}$

Figure 4.10: Graph G(V,E)

Figure 4.10 depicts the above statement briefly. There are two versions of the vertex cover problem: the decision and optimization versions. In the decision version, the task is to verify for a given graph G whether there exists a vertex cover of a specified size but in the optimization version the task is to find a vertex cover of minimum size. In this dissertation we consider the optimization version of the minimum vertex cover with the goal of obtaining optimum solution. Now the minimum vertex cover problem is formulated as an integer programming problem by using the following conditions: Binary variables a_{ij} for $(i = 1, 2, 3, \ldots, n; j = 1, 2, 3, \ldots, n)$ Form the adjacency matrix of the graph G. Each variable has only two values (1 or 0) according as an edge exists or not. In other words, if an edge. In other words, if the edge (v_i, v_j) is in E, then $a_{ij} = 1$, otherwise $a_{ij} = 0$. The result of the program expresses the vertex v_i in the independent set or not. $v_i = 1$ if in the independent set otherwise $v_i = 0$. In this way, the quantity of vertices in the independent set can be expressed by $Z = \sum_{i=1}^{n} v_i$ and any one or none of the vertex of the edge, (v_i, v_j) is included in the independent set, we can write the constrained condition of the MIS as $v_i + v_j \leq 1$. Thus the problem can be mathematically transformed into the following optimization integer programming problem as follows:

$$\text{MIS} \left\{ \begin{array}{ll} \text{Maximum} & Z = \sum_{i=1}^{n} v_i \\ & \\ \text{Subject to} & v_i + v_j \leq 1, \forall \left(v_i, v_j\right) \in \mathrm{E} \\ & \\ & v_i \in \{0, 1\}^n, \forall v_i \in V \end{array} \right\} \tag{4.1}$$

4.6 The Relaxation of Maximum Independent Set Problems

One can formulate a simple of a semidefinite relaxation for the Maximum Independent Set problem where we have a variable $x(v)$ for each node $v \in V$ indicating whether v is chosen in the independent set or not. We have constraints which state that for each edge (u, v) only one of u or v can be chosen.The standard form of the primal of maximum independent set problems:

$$\begin{cases} \text{Maximum} & \displaystyle\sum_{i=1}^{n} w(x)x(v) \\ \text{Subject to} & x(u) + x(v) \leq 1, (u, v) \in E \\ & x(v) \in [0, 1], \quad v \in V \end{cases} \tag{4.2}$$

Although the above is a valid integer programming relaxation of MIS when the variables are constrained to be in $\{0, 1\}$, it is not a particularly useful formulation for the following simple reason. We define the Lagrangian function as:

$$L(V, \alpha) = \sum_{i=1}^{m} V_i + \alpha \langle X, V_i + V_{j-1} \rangle \tag{4.3}$$

$$\psi(X, \alpha) = min \quad L(V, \alpha) \tag{4.4}$$

we found that the Lagrange dual function for the standard form Maximum Independent Set Problems:

$$(\mathrm{D}) \begin{cases} \text{Minimize} & \psi(X, \alpha) \\ \text{subject to} & \alpha \geqslant 0 \end{cases} \tag{4.5}$$

The above problem (4.2) linear program has an exponential number of variables and it cannot be solved in polynomial time in general but for some special cases of interest the above linear program can indeed be solved (or approximately solved) in polynomial time and leads to either exact algorithms or good approximation bounds. A pproximability of Vertex Cover and MIS: The following is a basic fact and is easy to prove.

Corollary 4.6.0.1. In any graph $G = (V, E), S$ is a vertex cover in G if and only if $V \backslash S$ is an independent set in G. Thus $\alpha(G)+ \beta(G) = |V|$ where $\alpha(G)$ is the size of a maximum independent set in G and $\beta(G)$ is the size of a minimum vertex cover in G.The above shows that if one of Vertex Colver or MIS is NP-Hard then the other is as well.

4.7 Relationship between Vertex Cover and Independent Set

Theorem 4.7.1. *[21] if $G = (V,E)$ is a graph, then S is an independent set if and only if $V - S$ is a Vertex Cover.*

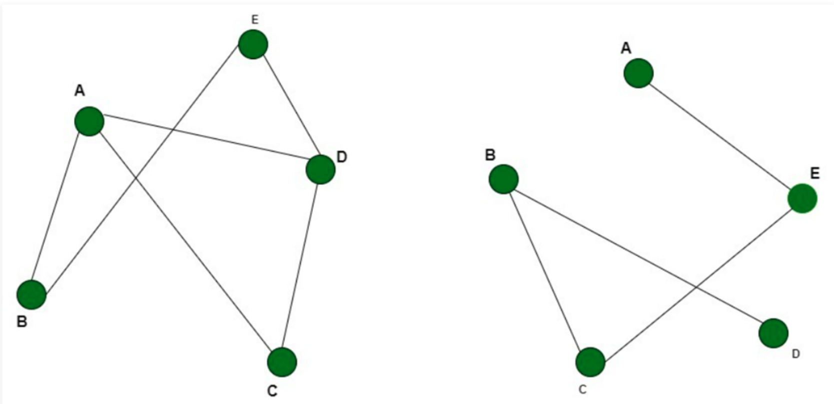
Proof: suppose S is Independent set ,and let $e = (u, v)$ be some edge.Only one of u, v can be in S . Hence, at least one of u, v is in $V - S$. So $V - S$ is Vertex Cover. Now we want to prove S is an Independent set. Suppose V-S is Vertex Cover, let $u, v \in S$ .there can't be in edge between u and v (otherwise, that edge wouldn't be covered in $V - S$) Hence S is Independent set .■

4.8 Independent Set is NP-Hard

In order to prove that the Independent Set problem is NP-Hard, we will perform a reduction from a known NP-Hard problem to this problem. We will carry out a reduction from which the Clique Problem can be reduced to the Independent Set problem. Every instance of the clique problem consisting of the graph $G(V, E)$ and an integer k can be converted to the required graph $G^{'}(V^{'}, E^{'})$ and $K^{'}$ of the Independent Set problem. We will construct the graph $G^{'}$ in the following way: $V^{'} = V$ i.e. all the vertices of graph G are a part of the graph $G^{'}$, $E^{'}$ =complement of the edges E, i.e. the edges not present in the original graph G. The graph $G^{'}$ is the complementary graph of G. The time required to compute the complementary graph $G^{'}$ requires a traversal over all the vertices and edges [2]. The time complexity of this is $O(V + E)$. We will now prove that the problem of computing the independent set indeed boils down to the computation of the clique. The reduction can be proved by the following two propositions:

- Let us assume that the graph G contains a clique of size k. The presence of clique implies that there are k vertices in G, where each of the vertices is connected by an edge with the remaining vertices. This further shows that since these edges are contained in G, therefore they can't be present in $G^{'}$. As a result, these k vertices are not adjacent to each other in $G^{'}$ and hence form an Independent Set of size k.

- We assume that the complementary graph $G^{'}$ has an independent set of vertices of size $G^{'}$. None of these vertices shares an edge with any other vertices. When we complement the graph to obtain $G^{'}$, these k vertices will share an edge and hence, become adjacent to each other. Therefore, the graph G will have a clique of size $G^{'}$.



G G'

clique of size 3 $\{A, C, D\}$ independent set of size 3 $\{A, C, D\}$

Figure 4.11: Clique and Independent Set

Thus, we can say that there is an independent set of size k in graph G if there is a clique of size k in $G^{'}$ (complement graph). Therefore, any instance of the independent set problem can be reduced to an instance of the clique problem. Thus, the independent set is NP-Hard. Therefore, we conclude that the independent set problem is NP and NP-Hard, it is NP-complete problem.

4.9 Algorithms

The following algorithm is designed to find the maximum independent set of a graph G. Adjacency matrix $A = (a_{ij})$ of the given graph G of n vertices and m edges is given as the input of the program and the degree $d\left(v_i\right)$, support $s\left(v_i\right)$ of the each vertex $v_i \in V$ of the graph G are calculated. Support of the vertex $v \in V$ is found by the relation $\sum\limits_{u \in N(v)} d(u)$.Add the vertex which has the maximum value of the support into the vertex cover V_c. If one or more vertices have equal maximum value of the support, in this case if (degree $(v_i) \geqslant$ degree (v_j)), add the vertex v_i into the vertex cover V_c otherwise add v_j into V_c. Update the adjacency matrix by putting zero in to the row and column entries of the corresponding vertex $v \in V_c$. Proceed the above process until G has no edges. i.e., up to $a_{ij} \neq 0, \forall i, j$. Finally the maximum independent set S of the graph G is extracted from the minimum vertex cover V_c of the graph G by $S = V$ - V_c. The pseudo-code of the proposed algorithm is described in Algorithm 1.

Definition 4.9.1. (Neighborhood of a vertex)
Let $G = (V, E), V$ is a vertex set and E is an edge set, be an undirected graph and let $V = n$ and $E = m$. Then for each $v \in V$, the neighborhood of v is defined by $N(v) = \{u \in V \setminus u$ is adjacent to v $\}$ and $N\left[v\right] = v \cup N(v)$.

4.10 Applications of Maximum Independent Set

Practical applications of the independent maximum problem appear in information retrieval, signal transmission analysis, classification theory, economics, scheduling, experimental design, and computer vision([69], [54], [12], [70]).In this section, we will briefly describe the selected applications.

Algorithm 1: Vertex Cover Algorithm (VCA)

1. The pseudo-code of the proposed algorithm:
2. Input: $G(V, E)$
3. Output: MIS $S(G) = V - V_c$ where V_c is the minimum vertex cover of G
4. $V_c \leftarrow \varphi$, selection $= 0$
5. $\forall v_i \in V$
6. do
7. $d(v_i) = \sum_j a_{ij}; s(v_i) = \sum_{v_j \in N(v_i)} d(v_j);$
8. Max $= s(v_1), k = 1$
9. for $i \leftarrow 2$ to n
10. If max $< s(v_i)$ then
11. $t = i$
12. else if max $= s(v_i)$ and $d(v_{i-k}) \leq d(v_i)$ then
13. $t = i$
14. else if max $=$ s(v_i) and $d(v_{i-k}) > d(v_i)$ then
15. $t = i - k$
16. $k = k + 1$
17. end for
18. $V_c = V_c \cup \{v_t\}$
19. selection $=$ selection $+1$
20. assign the value zero to the t^{th} row and t^{th} column of the matrix $A = (a_{ij})$;
21. while $A \neq (0)$
22. $|V_c| =$ selection;
23. end

4.10.1 Macromolecular Docking

Given two proteins, the protein docking problem is to find whether they interact to form a stable complex, and, if they do, then how. This problem is fundamental to all aspects of biological function, and development of reliable theoretical protein docking techniques is an important goal. One of the approaches to the macromolecular docking problem consists in representing each of two proteins as a set of potential hydrogen bond donors and acceptors and using a clique-detection algorithm to find maximally complementary sets of donor/acceptor pairs. To take a simple example of molecular anchoring, see the figure 4.12, so that the red heads represent a group of proteins, and we denote them by the x_1, x_2, x_3, symbols, which represent the maximum independent groups and can be mathematically described according to the Formula (4.6)

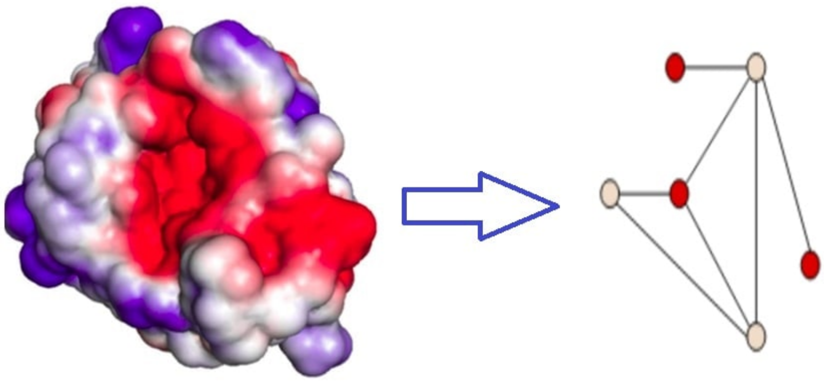


Figure 4.12: Macromolecular Docking

$$\begin{cases} \text{maximum} & f(x_1, ..., x_n) \\ \text{subject to} & g_i(x_1, x_2, ..., x_n) \leq 0, i = 1, ..., m \\ & x_i \geq 0, x_i \in R \end{cases} \tag{4.6}$$

4.10.2 Integration of Genome Mapping Data

One major technological development is being able to map genomes for products. A genome is all of a living thing's genetic material. It is the entire set of hereditary instructions for building, running, and maintaining an organism, and passing life on to the next generation. In most living things, the genome is made of a chemical called DNA. The genome contains genes, which are packaged in chromosomes and affect specific characteristics of the organism. A genome map helps scientists navigate around the genome. Like road maps and other familiar maps, a genome map is a set of landmarks that tells people where they are, and helps them get where they want to go. The landmarks on a genome map might include short DNA sequences, regulatory sites that turn genes on and off, and genes themselves. Often, genome maps are used to help scientists find new genes.Genome mapping is a capability that grew out of the biotech industry, and which is proliferating, largely due to the dropping cost of genome sequencing. The cost of genome sequencing in 2001 was dollar 100 million per sequence. By 2007 it was dollar 10 million per, and it has now dropped to dollar 1000.

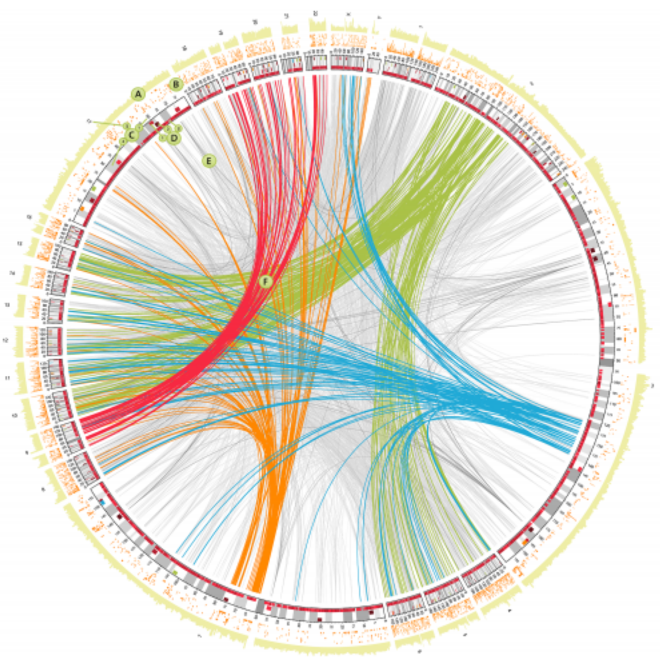


Figure 4.13: Genome Mapping

Due to differences in the methods used for analyzing overlap and probe data, the integration of these data becomes an important problem. It appears that overlap data can be effectively converted to probe-like data elements by finding maximal sets of mutually overlapping clones. Each set determines a site in the genome corresponding to the region which is common among the clones of the set; therefore these sets are called virtual probes. Finding the virtual probes is equivalent to finding the maximal cliques of a graph.

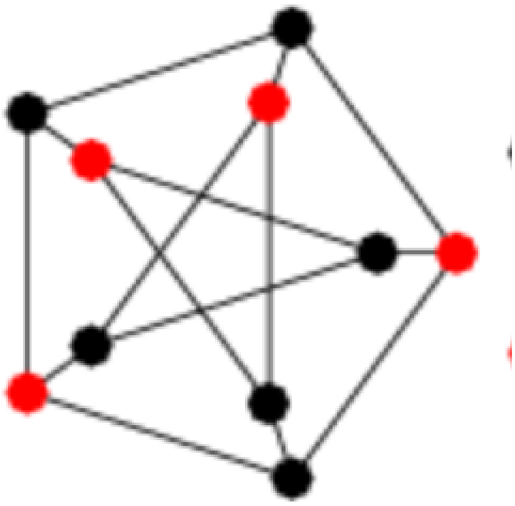


Figure 4.14: Integration of Genome Mapping Data

$$\begin{cases} \text{maximum} & C.X \\ \text{subject to} & g_i(x_1, x_2, ..., x_n) = 0 \\ & X \succeq 0, \quad X \in S^n, X \in [0,1]^n \end{cases} \tag{4.7}$$

4.10.3 Comparative Modeling of Protein Structure

The rapidly growing number of known protein structures requires the construction of accurate comparative models. This can be done using a clique detection approach as follows.We construct a graph in which vertices correspond to each possible conformation of a residue in an amino acid sequence, and edges connect pairs of residue conformations (vertices) that are consistent with each other (i.e., clash-free and satisfying geometrical constraints).Based on the interaction between the atoms corresponding to the two vertices, weights are assigned to the edges. Then the cliques with the largest weights in the constructed graph represent the optimal combination of the various main-chain and side-chain possibilities, taking the respective environments into account. and it is considered an important application of the applications of the maximum independent.sets as in the Figure 4.15

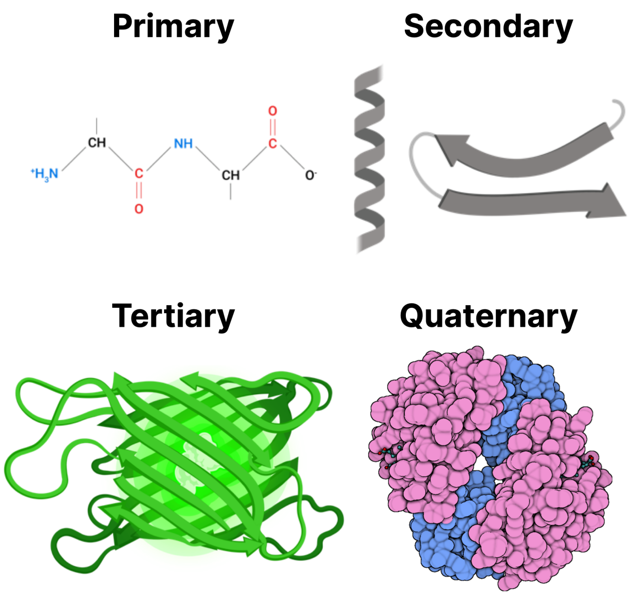


Figure 4.15: Comparative Modeling of Protein Structure

4.11 Nonlinear Optimization

Nonlinear optimization (or nonlinear programming) is the term used to describe an optimization problem when the objective or constraint functions are not linear, but not known to be convex. But there are no effective methods for solving the general nonlinear programming problem (4.8). Even simple looking problems with as few as ten variables can be extremely challenging, while problems with a few hundreds of variables can be intractable. Methods for the general nonlinear programming problem therefore take several different approaches, each of which involves some compromise. A general optimization problem is to select n decision variables $x_1, x_2, ..., x_n$ from a given feasible region in such a way as to optimize (minimize or maximize) a given objective function $f(x_1, x_2, ..., x_n)$ of the decision variables. The problem is called a nonlinear programming problem (NLP) if the objective function is nonlinear and/or the feasible region is determined by nonlinear constraints.Thus, in maximization form, the general nonlinear program is stated as:

$$\begin{cases} \text{Maximize} & f\left(x_1, x_2, \ldots, x_n\right) \\ \text{subject to:} & g_1\left(x_1, x_2, \ldots, x_n\right) \leq b_1, \\ & \vdots \\ & \vdots \\ & g_m\left(x_1, x_2, \ldots, x_n\right) \leq b_m \end{cases} \tag{4.8}$$

where each of the constraint functions g_1 through g_m is given. A special case is the linear program that has been treated previously. The obvious association for this case is

$$f\left(x_1, x_2, \ldots, x_n\right) = \sum_{j=1}^{n} c_j x_j$$

and $g_i\left(x_1, x_2, \ldots, x_n\right) = \sum_{j=1}^{n} a_{ij} x_j, \quad (i = 1, 2, \ldots, m)$. Note that nonnegativity restrictions on variables can be included simply by appending the additional constraints: $g_{m+i}\left(g_1\left(x_1, x_2, \ldots, x_n\right)\right) = -x_i \leq 0, \quad (i = 1, 2, \ldots, n)$. Sometimes these constraints will be treated explicitly, just like any other problem constraints. At other times, it will be convenient to consider them implicitly in the same way that nonnegativity constraints are handled implicitly in the simplex method. For notational convenience, we usually let x denote the vector of n decision variables $x_1, x_2, \ldots, x_n$ that is, $x = (x_1, x_2, \ldots, x_n)$ and write the problem more concisely as

$$\begin{array}{ll} \text{Maximize} & f(x) \\ \text{subject to} & g_i(x) \leq b_i, \quad (i = 1, 2, \ldots, m) \end{array}$$

As in linear programming, we are not restricted to this formulation. To minimize $f(x)$, we can of course maximize $-f(x)$. Equality constraints $h(x) = b$ can be written as two inequality constraints $h(x) \leq b$ and $-h(x) \geq -b$. In addition, if we introduce a slack variable, each inequality constraint is transformed to an equality constraint. Thus sometimes we will consider an alternative equality form :

$$\begin{cases} \text{Maximize} & f(x) \\ \text{subject to} & h_i(x) = b_i, \quad (i = 1, 2, \ldots, m) \\ & x_j \geq 0, \quad (j = 1, 2, \ldots, n) \end{cases} \tag{4.9}$$

Usually the problem context suggests either an equality or inequality formulation (or a formulation with both types of constraints), and we will not wish to force the problem into either form.

Example 4.11.1. **Two-dimensional model:**

A simple problem (Figure 4.16) .It can be characterized more simply by restrictions.

In Figure 4.16 the blue area is the feasible area.The intersection of the line with the

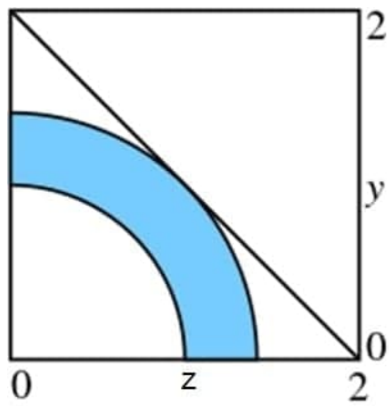


Figure 4.16: Two Dimation

feasible area the solution. The line is the best achievable shape line (locus with a given value of the goal function).

$$z_1 \geq 0$$

$$z_2 \geq 0$$

$$z_1{}^2 + z_2{}^2 \geq 1$$

$$z_1{}^2 + z_2{}^2 \leq 2$$

With a goal function to be maximal

$f(z) = z_1 + z_2$,when $z = (z_1, z_2)$.

Three- dimensional model:

The intersection of the top surface with the constrained space in the center the solution. (See the Figure 4.17) .It can be defined more simply by restrictions.

$$x_1{}^2 - x_2{}^2 + x_3{}^2 \leq 2$$

$$x_1{}^2 + x_2{}^2 + x_3{}^2 \leq 10$$

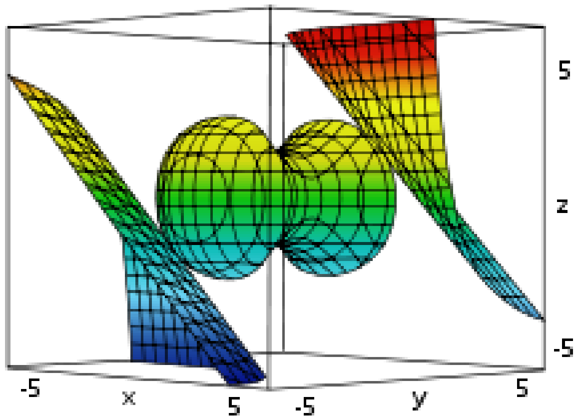


Figure 4.17: Three-Dimensional

With a goal function to be maximal

$$f(x) = x_1x_2 + x_2x_3$$

When $\quad x = (x_1, x_2, x_3).$

4.12 Approximation Methods

In this section, we begin by discussing methods of approximation, concepts, and The terms we used. Furthermore, the Penalty and Augmented Lagrange methods, a methods for both nonlinear programming problem and we offer the settlement.

Approximation methods are a very active area of improvement.Consider the minimizing convex function $\mathcal{G}\colon \mathbb{R}^n \to \mathbb{R}$ on a convex set X. The goal of approximation methods is to replace $\mathcal{G}$ and X with approximation $\mathcal{G}^k$ and X^k. The approximation method works only if the approximation is easier than the original problem [53], [14]. For each iteration k we tried to find

$$x^{k+1} = \arg\min_{x \in X^k} \mathcal{G}^k(x),$$

then at the next iteration, $\mathcal{G}^{k+1}$ and X^{k+1} are generated by the approximation which depends on the new point x^{k+1} .

Many great approximation methods are based on this idea, such as polyhedral approximation, the bundle method, the augmented Lagrangian method and interior point methods. In this research, we focused on the augmented Lagrangian method.

4.13 The Penalty and Augmented Lagrangian Methods

In an optimization problem, one wishes to maximize or minimize some function subject to some constraint. The general optimization problem given by :

$$\begin{cases} \text{maximum} & f(x) \\ \text{subject to} & x \in X \end{cases} \tag{4.10}$$

he function f is defined from a convex set $X \subseteq R^n$ into R. A point $x^* \in X$ is a local solution of problem (4.10) if there exists a neighborhood $B(x^*, t)$ such that $f(x^*) \geqslant f(x)$ for all $x \in B(x^*, t) \cap X = \{x \in X \mid \|X - x^*\| \leq t\}$

4.13.1 The Penalty Function Definition

The fundamental idea of the penalty function is a combination of an objective function and a penalty parameter to controls restrictions violations by adding a penalty term [15].Penalty approaches provide an easy and straight-forward approach for processing constrained problems [1].They are also particularly interesting because they are capable of overcoming potential slow convergence. Our aim is to replace a restricted problem with a sequence of unconstrained problems that are simpler to solve [15]. Consider the problem below:

$$\begin{cases} \text{maximum} & f(x) \\ \text{subject to} & h(x) = 0 \\ & g(x) \leqslant 0 \\ & x \in R^n \end{cases} \tag{4.11}$$

Assume this problem is substituted by unconstrained problem, where $\mu > 0$ and $\rho > 0$ are a large number [5] :

$$\psi(x, \rho, \mu) = f(x) + \rho \|h(x)\|^2 + \|[g(x)]_+\|^2$$
$$\text{subject to} \quad \mathrm{x} \in \mathrm{R^n}$$

Where, for a given inequality constraint $g(x) \leq 0$, the function $[g(x)]+ = \max\{0, g(x)\}$ is defined as follows:

$$[g(x)]_+ = \begin{cases} 0 & \text{if } g(x) \leq 0 \\ g(x) & \text{if } g(x) > 0 \end{cases}$$

We can intuitively see that the optimum solution on the equality constraint problem should be$\|h(x)\|^2$,

approach to zero. If that is not the situation, a great penalty would be incurred.It is clear that, in the state of using the form $f(x) + \mu \|g(x)\|^2$ will not be appropriate for the inequality constraint$g(x) \leq 0$; the penalty will have been incurred, whether $g(x) < 0$ or $g(x) > 0$. But, we do not need penalty for feasible points. Observe that if $g(x) \leq 0$, then $\|[g(x)]_+\|^2 = 0$, and no penalty is incurred. However, if $g(x) > 0$, then$\|[g(x)]_+\|^2 > 0$, and the penalty term $\mu g(x)$ is realized. On the other hand, it is noted that at points x where $g(x)= 0$, the previous objective function may not be differentiable, although g is differentiable. Consider the general form of constrained optimization with equality and inequality restrictions:

$$\begin{cases} \text{maximum} & f(x) \\ \text{subject to} & h_i(x) = 0, \qquad i = 1, ..., n \\ & g_j(x) \leq 0, \qquad j = 1, ..., m \\ & x \in R^n \end{cases} \tag{4.12}$$

Where the feasible region is denoted by:

$S = \{x \in R^n | h_i(x) = 0, i = 1, ..., n \quad \text{and} \quad g_j(x) \leq 0, \quad j = 1, ..., m\}$

4.13.2 The Augmented Lagrangian Method:

In this section, we present an approach known as the augmented Lagrangian method. We implemented the equality augmented Lagrangian, the inequality augmented Lagrangian, and the penalty methods for the linear programming (LP) problem. The initial results will provide an idea for working in semidefinite programming. The penalty method and augmented Lagrangian methods can be used for equality and inequality constraints. The standard LP problem is given by

$$\begin{cases} \text{maximum} & \langle c, x\rangle \\ \text{subject to} & \langle a_i, x\rangle = b_i, i = 1, ..., m \\ & x \geq 0 \quad \text{and} \quad x \in R^n \end{cases}$$

The problem data are given by the vectors$a_i \in R^n$ for $i = 1, \ldots, m$, the numbers $b_i \in R^n$ for $i = 1, \ldots, m$, and the vector $c \in R^n$. The notation $x \geq 0$ means that $x_i \geq 0$, for $i = 1, \ldots, n$. Furthermore, we let R^n_+ be the set of vectors $x \in R^n$ such that $x_i \geq 0$. The Lagrangian is given by:

$$L(\mathrm{x}, \mathrm{y}) = \langle \mathrm{c}, \mathrm{x}\rangle + \langle \mathrm{y}, \mathrm{b_i} - \langle \mathrm{a_i}, \mathrm{x}\rangle\rangle$$

The general form for the augmented Lagrangian is

$$L_\alpha(x, y) = f(x) + \langle y, b_i - \langle a_i, x\rangle\rangle + \frac{1}{2\alpha} \|b_i - \langle a_i, x\rangle\|^2 \tag{4.13}$$

The quadratic penalty term makes the new objective strongly convex if α is positive. Represent the primal (P) and the dual (D) general linear programming problem:

$$(\mathrm{P}) \begin{cases} \text{Maximize} & \langle c, x\rangle \\ \text{subjectto} & Ax = b \\ \mathrm{x} \geq 0 \end{cases} \tag{4.14}$$

and

$$(\mathrm{D}) \begin{cases} \text{Minimize} & \langle b, y\rangle \\ \text{subjectto} & A^T y \geq c \end{cases} = \begin{cases} \text{Minimize} & \langle b, y\rangle \\ \text{subjectto} & \left[c - A^T y\right]_+ = 0 \end{cases} \tag{4.15}$$

We consider the equality augmented Lagrangian function given by

$$L_\alpha(x,y) = \langle b, y\rangle + \left\langle x, \left[c - A^T y\right]_+ \right\rangle + \frac{1}{2\alpha}\left\| \left[c - A^T y\right]_+ \right\|_2^2$$

and the inequality augmented Lagrangian given by

$$L_\alpha(x,y) = \langle b, y\rangle + \frac{1}{2\alpha}\left(\left\| \left[c - A^T y\right]_+ \right\|_2^2 - \|\mathrm{x}\|_2^2 \right)$$

and consider the penalty function given by

$$P_\alpha(y) = \langle b, y\rangle + \frac{1}{2\alpha}\left\| \left[c - A^T y\right]_+ \right\|^2$$

4.14 The Penalty Function Methods

A penalty method is an approach that uses a sequence of unconstrained problems. This method replaces a constrained problem with unconstrained problems by adding a penalty term to the objective function [1]. In 1943, the first penalty function was studied by Courant. However, sequential unconstrained minimization approaches began to be widely used for solving optimization problems in the 1960 [33], and such approaches were applied to get the solution of the original constrained problems. Following [30], we consider the standard penalty function (quadratic penalty function) of problem (4.14) given by

$$P_q(x,\alpha) = f(x) + \frac{1}{2\alpha}\|g(x)\|^2 \tag{4.16}$$

Where $\alpha > 0$.

Let $x(\alpha)$ be a minimizer of $P_q(x,\alpha)$ for $\alpha > 0$ .The penalty methods produce a sequence of infeasible solutions. Each iterate $x\alpha^k$ is either necessarily infeasible or a local optimal solution of problem (4.14). These methods are beneficial because of their comparative simplicity as they can use powerful methods for solving unconstrained problems.

Although they have a solid theoretical background, they are not efficient in practice since the sequence of unconstrained problems does not produce an exact solution [33]. The penalty function method is summarized in Algorithm 2

Algorithm 2: The Penalty Function Method

1- Given x^0,and $\alpha^0 > 0$
2- Find x^{k+1} such that
$x^{k+1} = \text{argmin} Pq(x, \alpha^k)$
3- Choose $\alpha^{k+1} \leq \alpha^k$
4- Set $k = k + 1$ and repeat.

You may have noticed that the addition of constraints to an optimization problem has the effect of making it much more difficult. The goal of penalty functions is to convert constrained problems into unconstrained problems by introducing an artificial penalty for violating the constraint. Consider this example: Suppose there is a freeway (like a toll freeway) that monitors when you enter and exit the road. Instead of establishing a speed limit of 65 km in the hour, they could use these rules:

- You can drive as fast as you want.
- If you drive under 100 km in the hour, you can use our road for free.
- Every km you drive over 100 will cost you 500.

The previous example had no constraints ,you can drive as fast as you want! But the effect of these rules would still be to limit drivers to 100 km in the hour. You can also control the likelihood of speeding by adjusting the fine. Penalty functions work in exactly this way.

4.15 The Augmented Lagrangian Method (Multiplier Methods)

This method has stated to be used in the 1970s. Initially, it was called the method of multipliers. Now, this method is known as the augmented Lagrangian method. The objective this method is to solve constrained optimization problems. This is done by replacing a constrained problem with a sequence of unconstrained problems [1] [30]. The augmented Lagrangian method is similar to the penalty method since in both of them a penalty term is added to the objective. The only diference in the augmented Lagrangian method is that a Lagrange multiplier term is added to it [1] [30]. The augmented Lagrangian method was introduced by Hestenes [42]. To introduce the augmented Lagrangian method, we can change the constraint $g_i(x)$ of problem (4.14) to the constraint $g_i(x) + 2\alpha\beta = 0$, Thus, we obtain the problem

$$\begin{cases} \text{maximum} & f(x) \\ \text{subject to} & g_i(x) + 2\alpha\beta = 0, i = 1, ..., E \\ & x \in X \end{cases} \tag{4.17}$$

We apply the penalty method to problem (4.17), then we can obtain the augmented Lagrangian function as follows. We begin with the penalty problem for problem (4.17)

$$\text{argmin}\, f(x) + \frac{1}{2\alpha}(g(x) + 2\alpha\beta)^T(g(x) + 2\alpha\beta)$$

which expands to

$$\text{argmin}\, f(x) + \frac{1}{2\alpha}\left(g(x)^T g(x) + 4\alpha\beta^T g(x) + 4\alpha^2\beta^T\beta\right)$$

and simplifies to

$$\operatorname{argmin} f(x) + 2\beta^T g(x) + \frac{1}{2\alpha}\|g(x)\|^2$$

Therefore, the augmented Lagrangian function is

$$L(x,\beta,\alpha) = f(x) + 2\beta^T g(x) + \frac{1}{2\alpha}\|g(x)\|^2 \tag{4.18}$$

The multiplier method consists of updating an estimate of the Lagrange multiplier β and sometimes the penalty parameter α at each iteration [30]. The multiplier method is summarized in Algorithm 3

Algorithm 3: The Augmented Lagrangian Methods

1 Choose x^0, and $\alpha^0 > 0$, choose β^0.

2 Find x^{k+1} such that

$$x^{k+1} = \underset{x}{\operatorname{argmin}}\, L(x, \beta^k, \alpha^k).$$

3 Update β^k and α^k.

4 Set $k = k + 1$ and repeat.

In penalty methods, $x(\alpha)$ approaches x^* as $\alpha \to 0$. And the relaxation augmented Lagrangian method, we change $g_i(x) = 0$ to $g_i(x) + 2\alpha\beta = 0$, so we can attain x^* with a finite value of α .(See Figure 4.18)

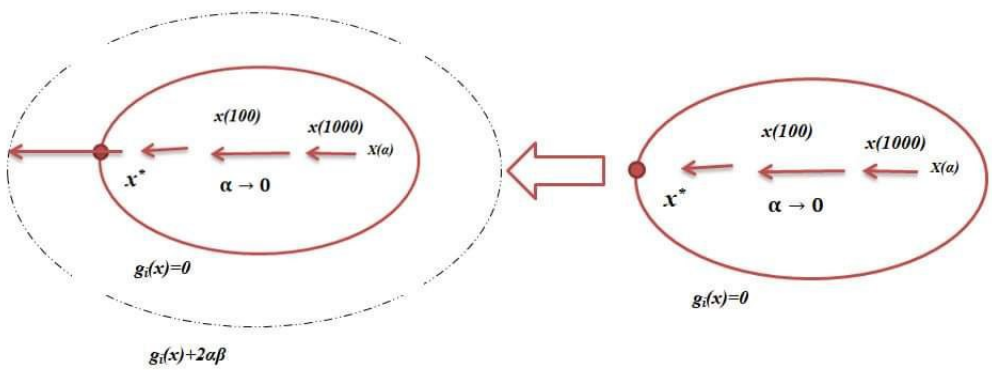


Figure 4.18: The Penalty and Relaxation Augmented Lagrangian Method

4.16 The Equality and Inequality Augmented Lagrangian for Semidefinite Programming

The standard general semidefinite programming (SDP) problem is given by:

$$\text{(SDP)}\begin{cases}\min & \langle C.X\rangle \\ \text{subject to} & \langle A_i.x\rangle \geq b_i, \forall i=1,\ldots,m, \\ & x\succeq 0\end{cases}\tag{4.19}$$

The augmented Lagrangian function is given by

$$L_\alpha(X,y)=\langle C\cdot X\rangle+y^T(b-\mathrm{Ax})+\frac{1}{2\alpha}\|b-\mathrm{Ax}\|_F^2$$

The Lagrangian function for (4.19) is given by

$$\begin{aligned}L(X,y)&=\langle C\cdot X\rangle+\langle y,b-\mathrm{Ax}\rangle\\&=b^Ty+\langle C-A^*y,X\rangle\end{aligned}$$

where $y\in R^n$ is the vector of Lagrangian multipliers and$\alpha>0$ is the penalty parameter. The problem data are given by the matrices $A_i\in S^n$, for $i=1,\ldots,m$, the numbers $b_i\in R$, for $i=1,\ldots,m$, and the matrix $C\in C^n$, define $(AX)_i=tr(A_iX)$: and the references thereinfor detailed studies. The dual problem of SDP problem (4.19) is given by S

$$\text{(SDD)}\begin{cases}\max & \sum_{i=1}^m y_iA_i+S=C \\ \text{subject to} & S\succeq 0\end{cases}\tag{4.20}$$

The augmented function for the dual to the linear constraints is defined as

$$L_{\alpha}(X, y, S) = \langle b, y\rangle + \langle x, A^{*}y + S - C\rangle + \frac{1}{2\alpha}\left\|A^{*}y + S - C\right\|_{F}^{2}$$

where the adjoint operator $A^{*}y = \sum_{i=1}^{m} y_i A_i, \alpha > 0$ satisfies $\langle A^{*}y, X\rangle = \langle y, AX\rangle$ and $X \in S^{n}$

Definition 4.16.1. A point X is strictly feasible for (SDP) if it is feasible for (SDP) and satisfies $X \succ 0$; and (y, S) is strictly feasible for (SDD) if it is feasible for (SDD) and satisfies $S \succ 0$: We define the optimal values of the primal problem (4.19) and its dual problem (4.20) as:

$$\mathrm{p}^{*} = \inf\left\{\langle c, \mathrm{x}\rangle; \langle \mathrm{A_i}, \mathrm{x}\rangle \geq \mathrm{b_i}, \forall \mathrm{i} = 1, \ldots, \mathrm{m} \text{ and } \mathrm{X} \succeq 0\right\}$$
$$\mathrm{d}^{*} = \sup\left\{\langle \mathrm{b}, \mathrm{y}\rangle; \sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{y_i A_i} + \mathrm{S} = \mathrm{C} \text{ and } \mathrm{S} \succeq 0\right\}$$

Consider the algorithm

Algorithm 4: The bounding MIS for augmented lagrangian method

1 Input $\alpha_0 > 0, \epsilon_0 > 0, y_0 = 0, z_0 = \theta$, Intial y and z variables
2 (y^{k}, z^{k})=argmin $H(x, y, s, z, \alpha)$
By using approximate method (penalty and augmented)
3 (y^{k+1}, z^{k+1})=max $\{[c - A^{T}y]_{+}, y^{T}(b - AX)\}$
$x^{k+1} = x^{k} + \frac{1}{\alpha_k}[A^{*}y + s + c]_{+}$
4 Update the bound $H^{*}(x, y, s, z, \alpha) = H(x, y, s, z, \alpha)$
where $H^{*}(x, y, s, z, \alpha)$ =min $L_{\alpha}(x, y, s)$
5 Set $k = k + 1$ and repeat and $\alpha^{k+1} = \alpha^{k}, \epsilon^{k+1} = \epsilon^{k}$.

The bounding procedure:

$H(x, y, s, z, \alpha) = minL_{\alpha}(x, y, s)$

Then the dual give as the tight bound which concider as:

$$
(\mathrm{D}) \begin{cases} \max \quad H(x,y,s,\alpha) \\ \\ \qquad\quad \alpha \geq 0 \end{cases} \tag{4.21}
$$

The dual (the best bound)

$H(x,y,s,z,\alpha) \leq p^*$

CHAPTER 5

NUMERICAL RESULTS

5.1 Numerical Results

In this Chapter, we present the numerical results of the algorithms being introduced throughout Chapter four. The augmented Lagrangian method is used to generate the numerical results, which were validated with the penalty method. A novel combined algorithm in the augmented Lagrangian method and the the penalty method is introduced and the results seemed to be more valuable than the separated two methods. Comprehensive comparisons between the results of the three numerical models are explained.

5.1.1 Julia Language (JuliaBox) :

Julia is a high-level, high-performance, dynamic programming language. While it is a general-purpose language and can be used to write any application. Distinctive aspects of Julia's design include a type system with a parametric polymorphism in a dynamic programming language; with multiple dispatches as its core programming paradigm. Julia was started in 2009, by Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman, who set out to create a free language that was both high-level and fast. On 14 February 2012, the team launched a website with a blog post explaining the language's mission.The using Julia is by creating an account for the user and logging in to the site and working on it needs the availability of the internet. It is possible to work on it without the internet, but the condition that the packages that need constant updating are available.

5.1.2 Algorithms and Numerical Computation

We now discuss our numerical results. In Figure 5.1, we plotted the bounds of the inequality augmented Lagrangian methods and the penalty methods. This test was accomplished on a specific graph imported from the Biq Mac library [81]. This graph includes 100 nodes connected with 2475 edges,also this graph includes 80 nodes connected with 1940 edges and 50 nodes connected by 1214 edges See Figures (5.1,5.2). It turns out that the penalty method approaches faster. However, when the solution comes closer to the SDP bound, it is clear that the augmented Lagrangian method provides a faster convergence.

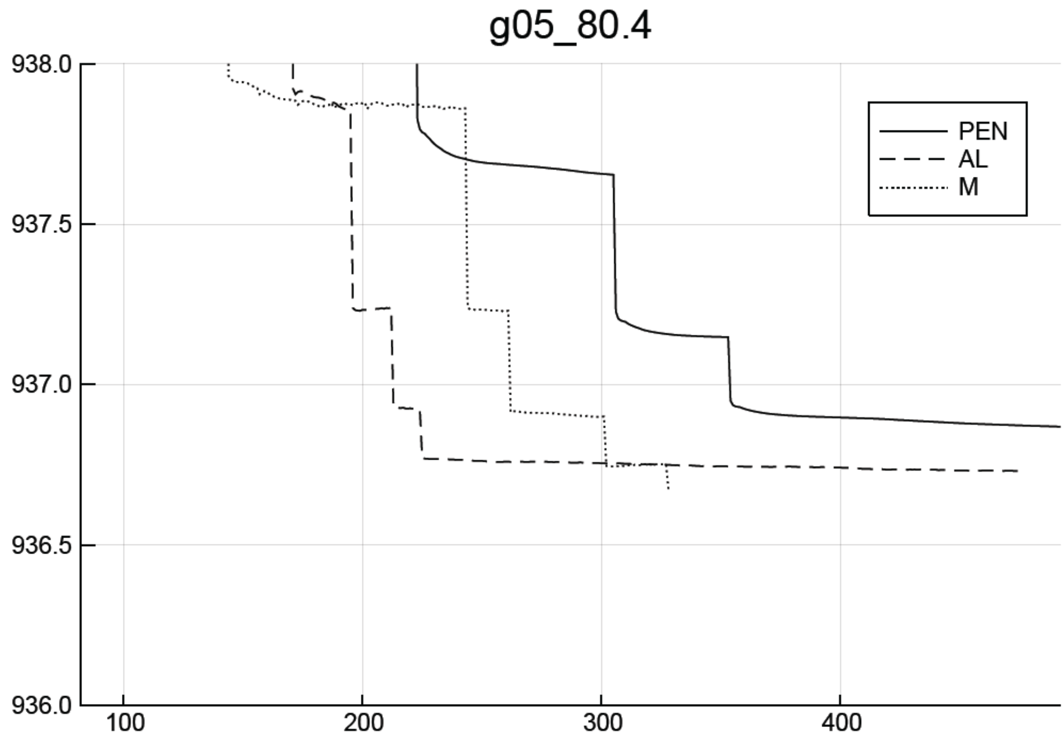


Figure 5.1: Comparison of the penalty and augmented Lagrangian test method with 80 nodes connected with 1940 edges where y-axis represented the bound and x-axis represented the function calls .

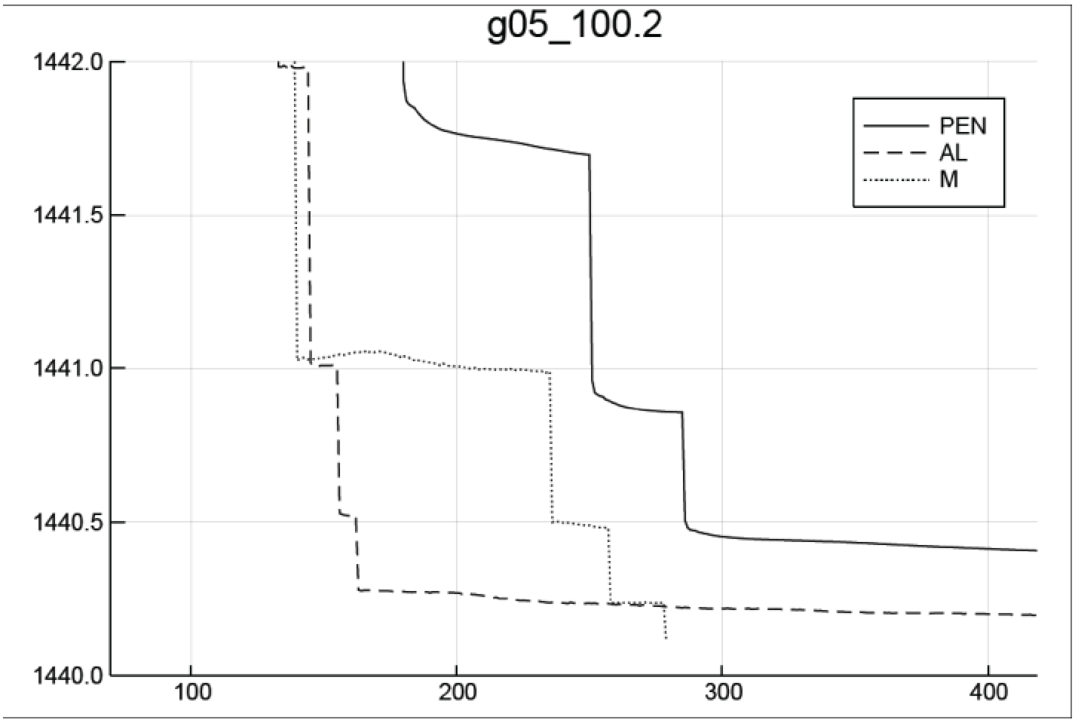


Figure 5.2: Comparison of the penalty and augmented Lagrangian test method with 100 nodes connected with 2475 edges where y-axis represented the bound and x-axis represented the function calls .

In fact, these problems contain 80 nods and 1940 edges [81].See Table 5.1 Also, we used three approaches in these tests: Penalty method, Augumented Lagrangian method and combined(Mixed) method. It was shown that the Augmented Lagrangian Method is more accurate than the method of Penalty Method, which means that the Augmented lagrangian method is the best, therefore it is provide the optimal solution of this problem.

Graphs	Penalty fcalls	Augmented fcalls	Mixed fcalls
g05_80.0	566	319	**261**
g05_80.1	943	489	**375**
g05_80.2	623	**292**	333
g05_80.3	421	165	**142**
g05_80.4	666	490	**330**
g05_80.5	323	**185**	200
g05_80.6	459	256	**238**
g05_80.7	353	311	**262**
g05_80.8	554	334	**181**
g05_80.9	435	316	**290**

Table 5.1: Test results between the two methods the penalty and augmented Lagrangian method with 80 nodes connected with 1940 edges.

In Figure 5.2, we plotted the bounds against the number of function calls for the same graph in the above figure, Actually, there are 100 nods and 2475 edges in these problems. In addition, these tests employed two approaches: penalty and Lagrangian methods. Furthermore, we compared the results and developed a new approach that combined the two methods and has shown promising results in terms of speed and accuracy. Also,tables 5.2) demonstrate that the Augmented lagrangian method is more accurate than the Penalty method, implying that the Augmented lagrangian method is the best. It is clear from the results that the penalty method starts faster and meets with the inequality augmented Lagrangian at the point at which the bound is approximately 1440. After this point, it is evident that the augmented Lagrangian method performed faster and required 150 function calls to converge, while the penalty method required 401 function calls.see Table 5.2

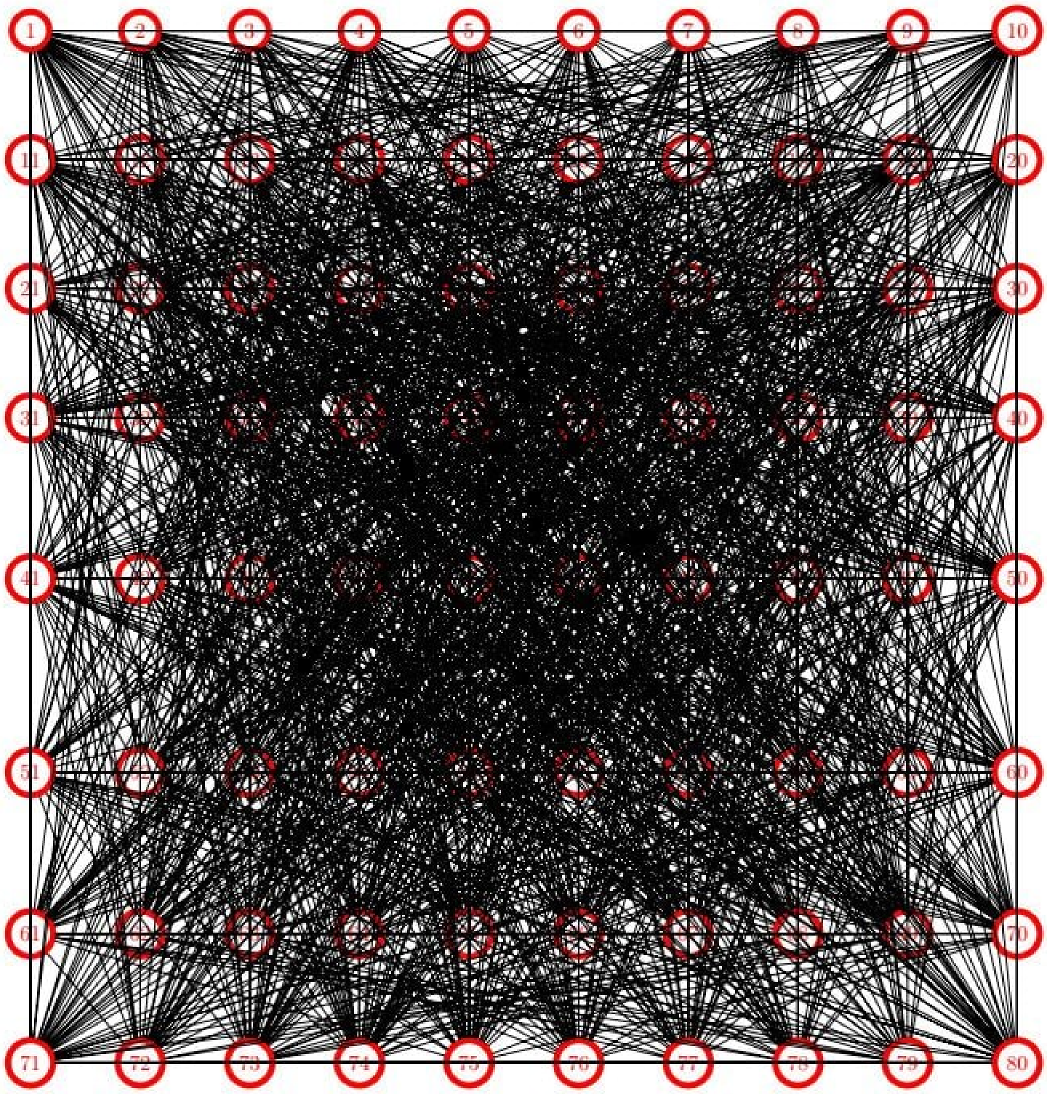


Figure 5.3: The Problem contain of 80 nodes

Graphs	Penalty fcalls	Augmented fcalls	Mixed fcalls
g05_100.0	401	**150**	200
g05_100.1	354	201	**197**
g05_100.2	517	450	**285**
g05_100.3	371	193	**170**
g05_100.4	409	**240**	270
g05_100.5	502	202	**179**
g05_100.6	441	391	**256**
g05_100.7	626	472	**391**
g05_100.8	549	**390**	401
g05_100.9	375	300	**289**

Table 5.2: Test results between the two methods the penalty and augmented Lagrangian method with 100 nodes connected with 2475 edges.

5.2 Numerical Models

In this section, we evaluate the performance of the new proposed algorithm that we explained throughout the dissertation. The figures in this section illustrate the number of function calls of the three approaches being used to solve the binary quadratic problems. Different sizes of instances were tested, and results were extracted and are shown in this section.

5.2.1 Results of Different Graphs with Dimensions of 100 Nodes

Graphs with higher number of nodes were also tested using our proposed methods. Four graphs containing 100 nodes with the same number of edges 2475 edges and considered as maximum independent set problems were solved, as shown in Figures 5.5

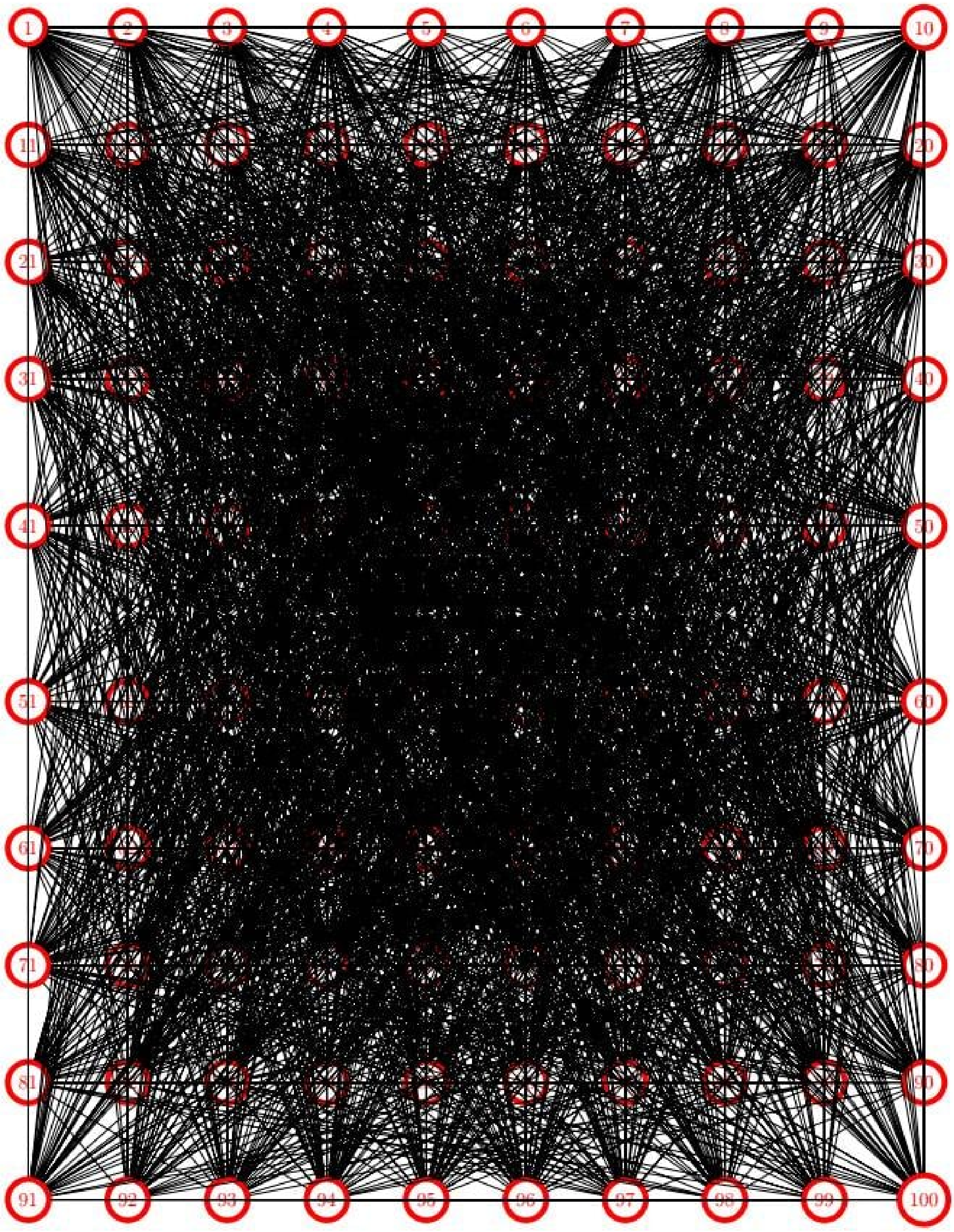


Figure 5.4: The Problem contain of 100 nodes

5.6 and 5.7 . Fast convergence was seen when utilising the combined method with around 188 function calls as shown in Figure 5.5. As can be seen in Figure 5.6, the combined method required approximately 428 function calls when the bound was around 8117. The augmented Lagrangian method performed better and was fast to converge to the preferable bound.It is important to identify that the semidefinite bound 8117 and the optimal cut is 8045. At the point close to the required bound, we noticed a good convergence of the augmented Lagrangian method to meet with the semidefinite bound after nearly 431 function calls and the penalty method was the slowest with 460 function calls.

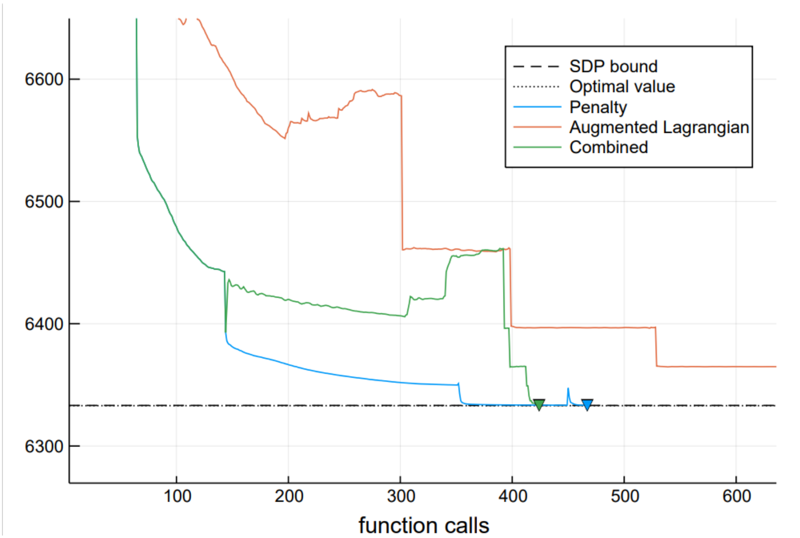


Figure 5.5: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph gk1d, where y-axis represented the bound and x-axis represented the function calls

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
gka10b	1208.67272	154	11620	**1280**	1481
gka10d	19162.83824	19102	2020	**1870**	2277
gka1a	3414.41551	3414	1031	1023	**699**
gka1b	133.99965	133	2681	**2571**	3000
gka1c	5058.95772	5058	**309**	566	381
gka1d	6333.17726	6333	471	860	**433**
gka1e	16464.11796	16464	2011	**1165**	1287
gka1f	63012.37510	61194	2880	2755	**2698**
gka2a	6063.12603	6063	305	**132**	364
gka2b	204.15121	121	1540	**1250**	1418
gka2c	6213.97887	6213	975	**606**	715
gka2d	6579.09478	6579	**935**	1050	1167
gka2e	23664.57541	23395	1820	**1756**	1849
gka2f	104379.99224	100161	2439	3845	**2423**
gka3a	6037.13660	6037	1122	1065	**956**

Table 5.3: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
gka3b	289.76896	118	1814	**1390**	1834
gka3c	6665.93382	6665	**340**	650	370
gka3d	9261.28638	9261	**878**	1000	999
gka3e	25529.09562	25243	1845	2300	**1834**
gka3f	144920.36872	138035	2311	**2242**	3745
gka4a	8598.13797	8598	985	1160	**947**
gka4b	299.94767	129	2545	**2376**	2545
gka4c	7398.63055	7398	**498**	890	566
gka4d	10730.95505	10727	1456	**1309**	2078
gka4e	35821.08965	35594	1656	2000	**1541**
gka4f	180981.11092	172771	3121	**2956**	3023
gka5a	5737.99856	5737	**382**	589	751
gka5b	431.59163	150	1640	1573	**1548**
gka5c	7362.99129	7362	861	1260	**723**
gka5d	11721.96721	11626	1700	**1567**	1925
gka5e	36068.61159	35154	2400	2439	**2150**
gka5f	200748.68965	190507	4300	**3667**	3878
gka6a	3980.99348	3980	**220**	230	270
gka6b	553.22900	146	1744	1890	**1311**
gka6c	5825.45322	5824	420	**352**	522
gka6d	14207.48330	14207	1522	**1047**	1298

Table 5.4: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

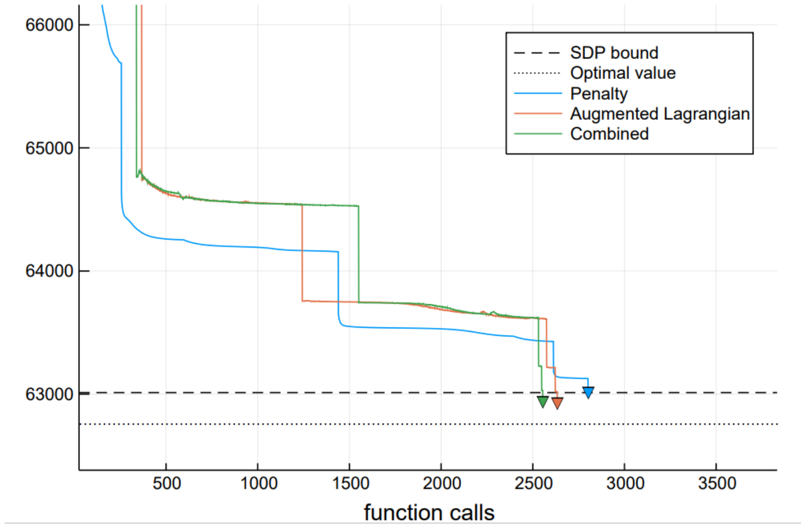


Figure 5.6: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph gk1f

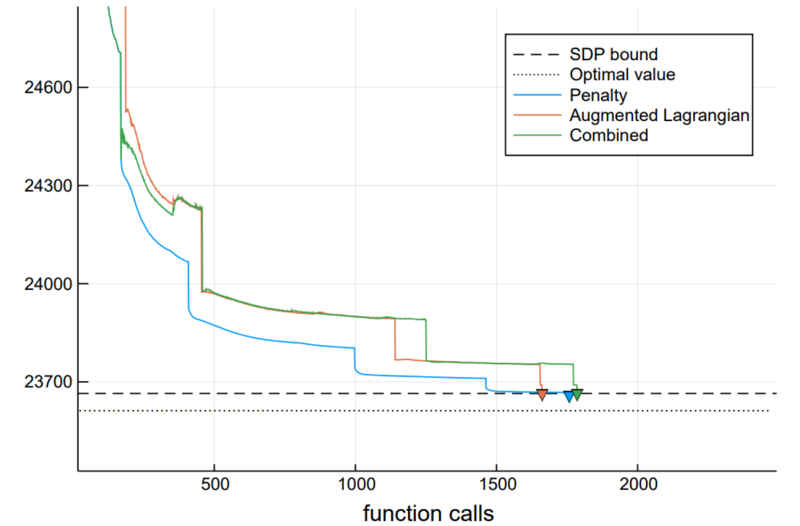


Figure 5.7: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph gk2e

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
gka7a	4541.92594	4541	**245**	356	367
gka7b	635.43440	160	1678	**1109**	1434
gka7c	7225.98017	7225	**315**	700	413
gka7d	14567.10427	14476	1742	**1057**	1100
gka8a	11109.43487	11109	460	**350**	650
gka8b	849.89441	145	**1200**	1262	1270
gka8d	16352.38459	16352	**1056**	1434	1176
gka9b	955.31397	137	1156	**1074**	1214
gka9d	15722.60254	15656	1978	**1456**	1500

Table 5.5: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

5.2.2 Numerical Results of Graph g3.0-100

A graph of 100 nodes with 2475 edges was used for testing various methods, and the results are shown in Figure 5.12. As can be seen, we compared the results of three methods (penalty method, augmented Lagrangian method, and a combined method). It is noticeable from this figure that the combined method provided a faster convergence to the optimal semidefinite bound. It is clear that the combined method approaches the semidefinite bound after approximately 215 function calls. However, the augmented Lagrangian and penalty methods approach the bound after 230 and 446 function calls, respectively. The augmented Lagrangian method test is shown in Table 5.2 The same g3.0-100 graph was used. Also, similar parameters were used as 101 initial conditions, as can be seen in the last row in this table. All initial values of the main parameters are illustrated in Table 5.6 On the other hand, the combined method, which combines the penalty method and the augmented Lagrangian method was also proposed and used to solve the problem as shown in Table 5.6 As discussed, the idea is to take advantage of the fast reduction feature of the penalty method at the beginning of the test, and then at a specific alpha value (reduced alpha twice) switches to the augmented Lagrangian and keeps solving to approach to the optimal bound. This new algorithm seems to be performing better than both methods when they work separately.

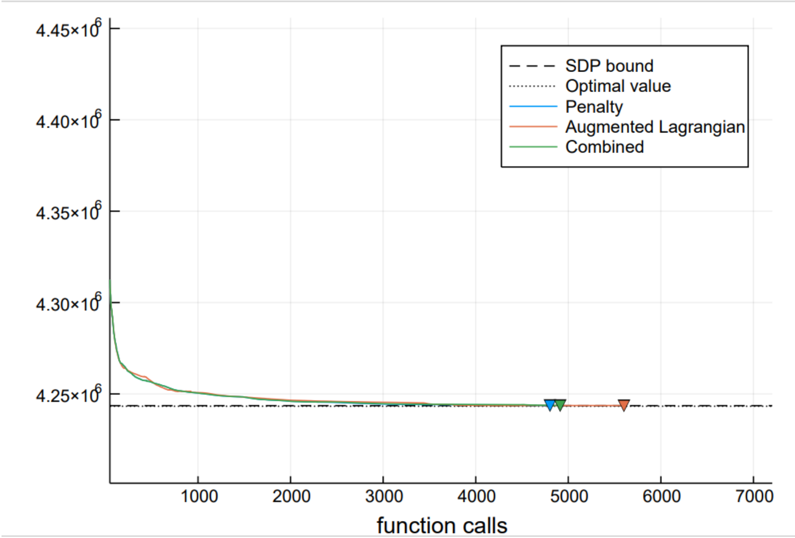


Figure 5.8: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising2.5-150-7777

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
ising3.0_100_5555	2448430.69591	2448189	**1430**	1679	1987
ising3.0_100_6666	1984413.23917	1984099	**1265**	1313	1511
ising3.0_100_7777	3336237.95581	3335814	3256	2561	**2341**
ising3.0_150_5555	4279776.81357	4279261	3965	**3256**	4019
ising3.0_150_6666	3949879.77770	3949317	4031	3267	**2341**
ising3.0_150_7777	4211510.92030	4211158	3000	2801	**2253**
ising3.0_200_5555	6216108.37879	6215531	**5341**	6128	6731
ising3.0_200_6666	6756637.30748	6756263	5671	**3456**	6532
ising3.0_200_7777	5561271.33233	5560824	6000	**5231**	7562
ising3.0_250_5555	7823982.65514	7823791	7756	**6731**	8942
ising3.0_250_6666	6903664.75993	6903351	**4561**	6234	6731
ising3.0_250_7777	6418428.57285	6418276	8712	763	**5871**
ising3.0_300_5555	8493505.06088	8493173	8231	7999	**7912**
ising3.0_300_6666	8915458.39818	8915110	**5671**	6789	7567
ising3.0_300_7777	8243043.78682	8242904	**4561**	7531	8962

Table 5.6: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

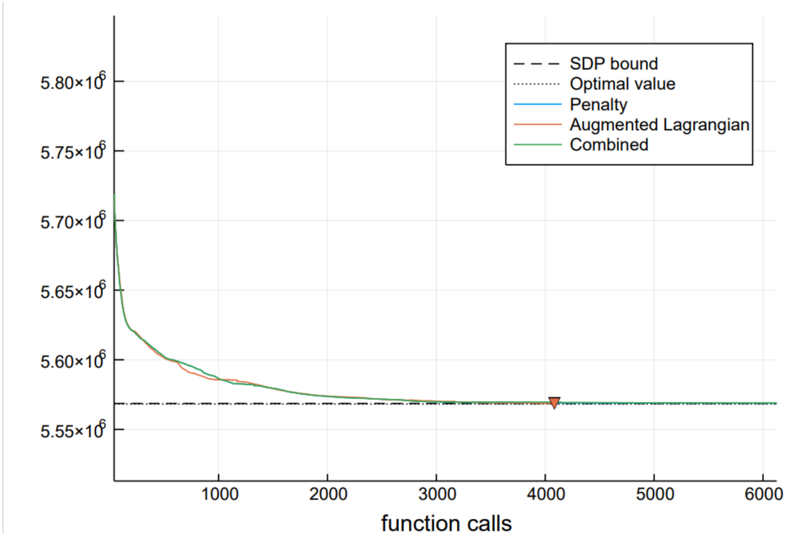


Figure 5.9: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising2.5-200-7777

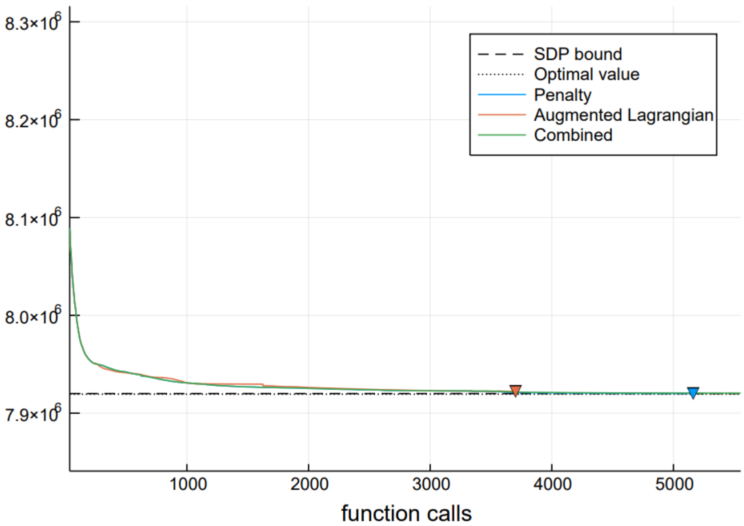


Figure 5.10: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising2.5-250-5555

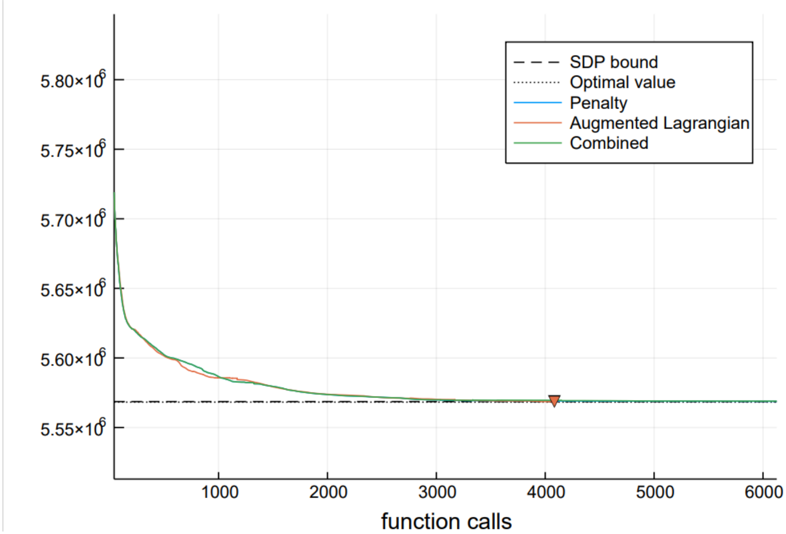


Figure 5.11: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising2.5-300-5555

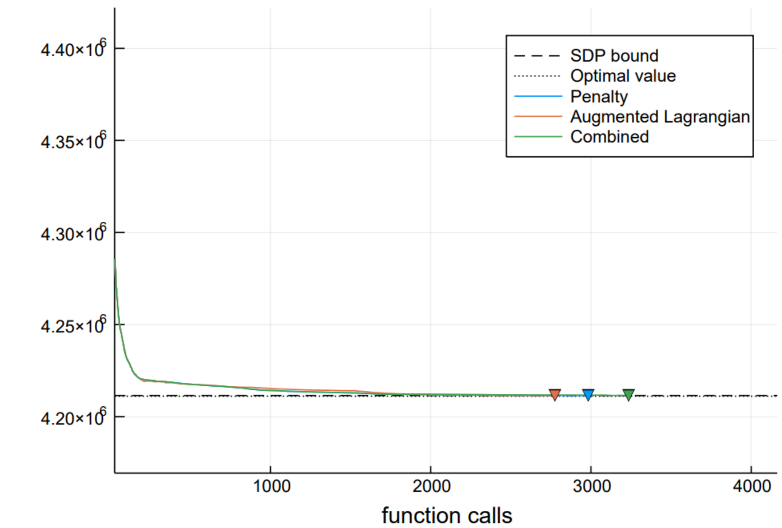


Figure 5.12: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising3.0-150-7777

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
ising2.5_100_5555	2460322.89700	2460049	**1361**	2118	2197
ising2.5_100_6666	2031519.63986	2031217	2021	**1342**	1560
ising2.5_100_7777	3363843.16961	3363230	**2666**	3451	4531
ising2.5_150_5555	4364075.63155	4363532	2341	2876	**819**
ising2.5_150_6666	4057614.24529	4057153	4896	4543	**4495**
ising2.5_150_7777	4243590.02345	4243269	**4081**	5666	4089
ising2.5_200_5555	6295121.43256	6294701	6146	**5531**	5961
ising2.5_200_6666	6795833.18191	6795365	6860	7563	**5586**
ising2.5_200_7777	5568657.53679	5568272	9594	4022	**4020**
ising2.5_250_5555	7920010.00204	7919449	5331	**3745**	5441
ising2.5_250_6666	6926249.90794	6925717	**4532**	5321	7451
ising2.5_250_7777	6597181.23130	6596797	6987	8964	**4532**
ising2.5_300_5555	8579651.53957	8579363	9765	**4291**	8045
ising2.5_300_6666	9102368.69243	9102033	9021	**8671**	8771
ising2.5_300_7777	8324051.74896	8323804	8250	8340	**7541**

Table 5.7: Function calls and nodes to solve the problems Different problems of Biq Mic Library.

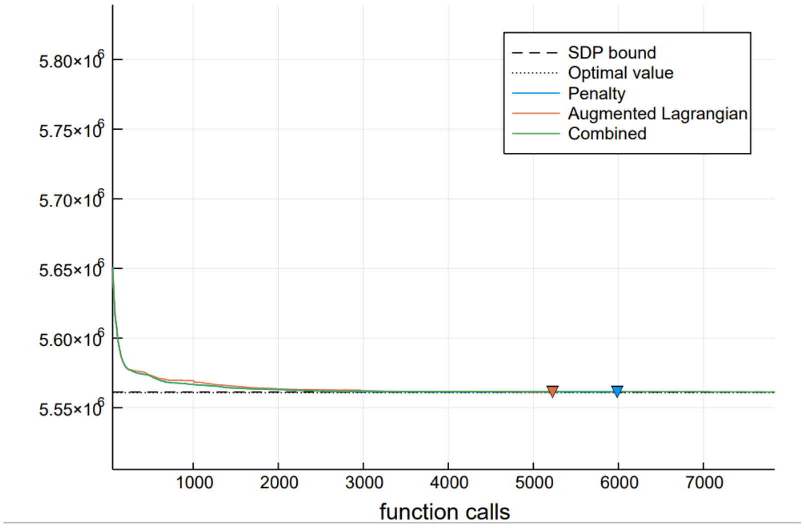


Figure 5.13: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph ising3.0-200-7777

5.3 Numerical Results of Graphs pm1d - 80.0 , pm1d - 100.0 and pm1s - 80.1

More graphs were also tested by using the three methods in order to give a better idea that the proposed algorithms can be successful for problems that have higher number of 108 nodes. The graph called pm1d 80.5 was used, which has 80 nodes with 3128 edges. The target in this case study is to approach the bound 257.5. Figures 5.14 and 5.15 demonstrate the performnce of the three methods. The calculated semidefinite bound for instance pm1s 80.4 is equal to 70, as shown in Figure 5.14. The augmented Lagrangian method converged faster than both methods. It required 829 function calls, which is nearly 100 function calls lower than the combined method to approach to the preferable bound. The penalty method performed slower than the other two because it required 323 function calls

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
pm1d_100.0	364.82456	340	323	183	**163**
pm1d_100.1	352.71143	324	256	**200**	276
pm1d_100.2	416.24235	389	365	**180**	465
pm1d_100.3	418.98334	400	556	500	**401**
pm1d_100.4	388.92342	363	533	298	**256**
pm1d_100.5	461.07720	441	436	**233**	245
pm1d_100.6	385.32408	367	706	**234**	276
pm1d_100.7	374.00485	361	845	590	**367**
pm1d_100.8	397.09047	385	576	278	**251**
pm1d_100.9	425.87169	405	476	**236**	287
pm1d_80.0	233.81283	227	620	**300**	321
pm1d_80.1	259.16657	245	**411**	417	536
pm1d_80.2	294.86177	284	587	**256**	376
pm1d_80.3	297.76394	291	787	**267**	326
pm1d_80.4	258.86377	251	787	456	**272**
pm1d_80.5	257.53359	242	349	**298**	345
pm1d_80.6	217.09151	205	530	**287**	309
pm1d_80.7	260.39022	249	720	316	**209**
pm1d_80.8	300.37542	293	669	259	**233**
pm1d_80.9	264.63628	258	492	301	**265**

Table 5.8: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
pm1s_100.0	129.04253	127	740	**380**	451
pm1s_100.1	130.29090	126	534	435	**352**
pm1s_100.2	126.60467	125	632	**355**	371
pm1s_100.3	115.33249	111	528	394	**275**
pm1s_100.4	131.42055	128	713	**277**	441
pm1s_100.5	129.82046	128	552	405	**268**
pm1s_100.6	125.54201	122	521	317	**264**
pm1s_100.7	112.00673	112	923	**780**	870
pm1s_100.8	121.17854	120	686	470	**379**
pm1s_100.9	127.81774	127	1238	840	**686**
pm1s_80.0	79.00552	79	840	862	**720**
pm1s_80.1	85.02522	85	1172	**800**	1066
pm1s_80.2	84.20944	82	640	700	**356**
pm1s_80.3	81.46720	81	1977	1498	**951**
pm1s_80.4	70.02195	70	**970**	1667	1349
pm1s_80.5	87.21569	87	1002	1162	**780**
pm1s_80.6	74.23661	73	850	**375**	687
pm1s_80.7	83.41438	83	876	740	**430**
pm1s_80.8	81.86462	81	1077	**476**	676
pm1s_80.9	70.23645	70	1189	**860**	1287

Table 5.9: Function calls and nodes to solve the problems Different problems of Biq Mic Library.

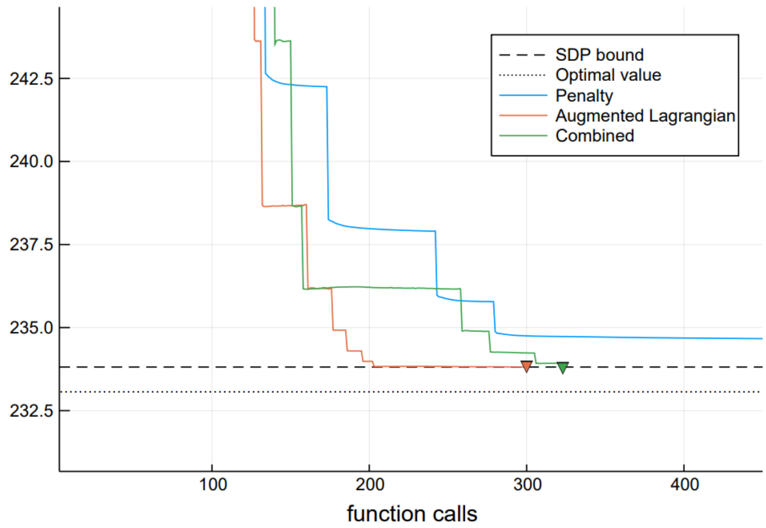


Figure 5.14: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph pm1d-80.0

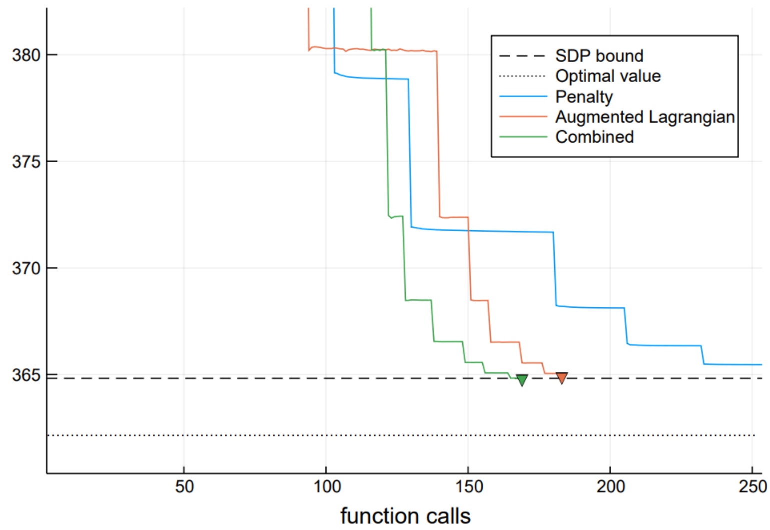


Figure 5.15: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph pm1d-100.0

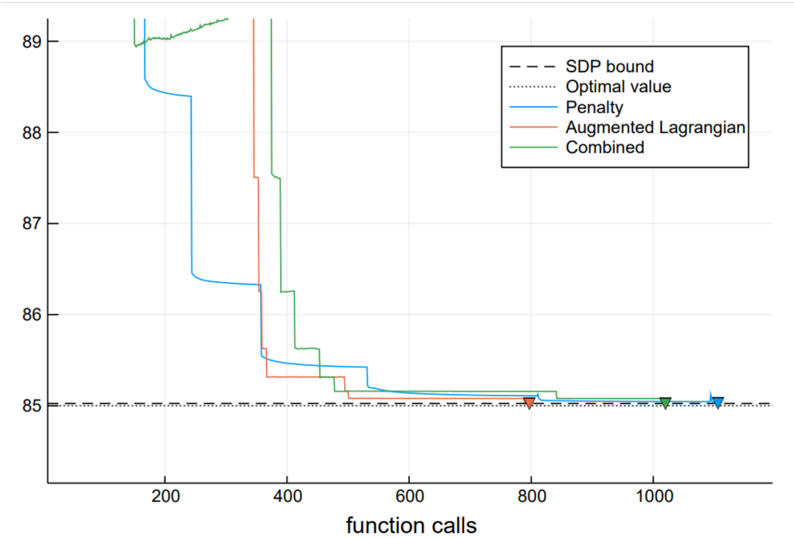


Figure 5.16: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph pms-80.1

5.4 Numerical Results of Graph pw01-100.0 ,pw05-100.5 and t2g10-5555

It is clear from all graphs we tested that the penalty method provided a lesser speed of convergence to achieve the same bound. The success of convergence for our combined algorithm comes from the augmented Lagrangian method. Figure 5.17 illustrates the convergence of the solution for a graph containing 80 nodes with 1580 edges. It is clear that the combined algorithm has better characteristics of convergence than the other two methods.As shown in Table 5.13 the penalty method required 686 function calls. The other two methods, such as the augmented Lagrangian and the combined methods, required 476 function calls and 389 function calls, respectively as demonstrated in Tables 5.15 and 5.14.

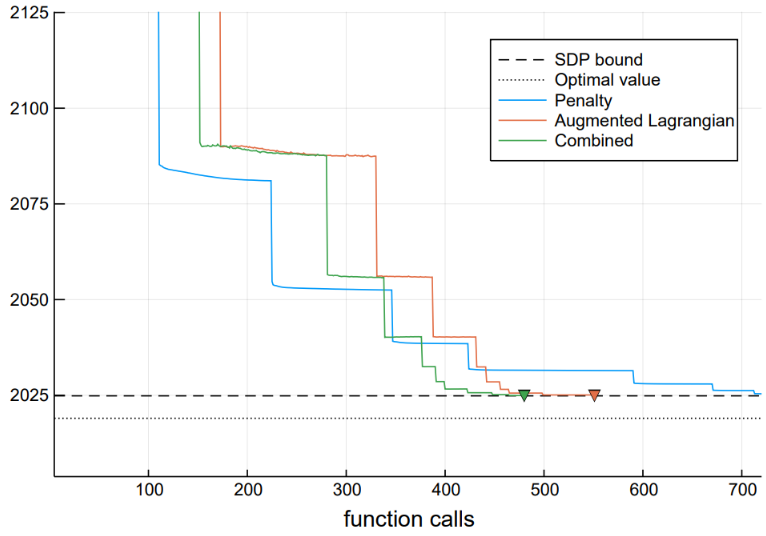


Figure 5.17: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph pw01-100.0

We can notice from the results that our novel combined method works very well for the instances that have higher number of nodes. Figure 5.18 demonstrates the solution of one of the instances that have 250 nodes. It is obvious that the penalty and the combined methods have two identical solutions in the region between approximately 8700 to 8400 bound values. However in the same region, the augmented Lagrangian method performs slower than the two other methods. The penalty method shows a fairly good performance when approaching to the optimal bound but it does not approach to the bound faster than the other two methods in the region very close to the bound when it required 480 function calls

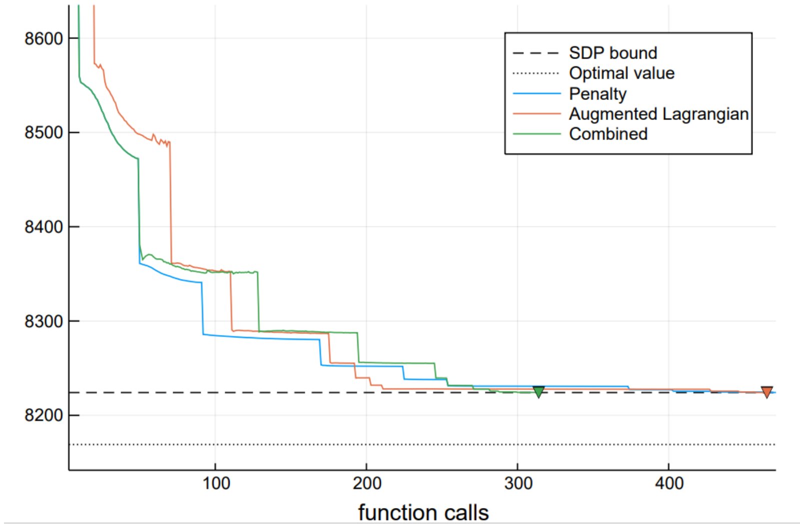


Figure 5.18: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph pw05-100.5

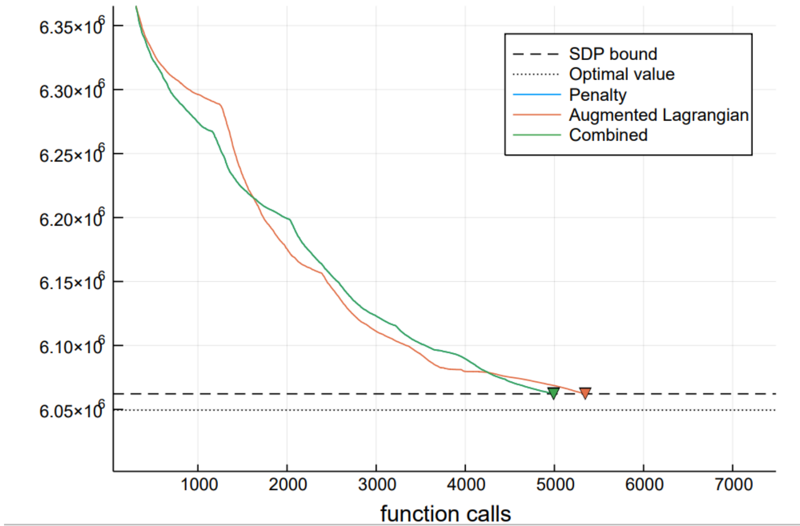


Figure 5.19: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph t2g10-5555

Our results					
Graphs	The goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
gka10b	1208.67272	154	1719	**1389**	1589
gka10d	19162.83824	19102	2014	**1860**	2285
gka1a	3414.41551	3414	**631**	1027	691
gka1b	133.99965	133	**2381**	2476	3033
gka1c	5058.95772	5058	**308**	552	371
gka1d	6333.17726	6333	**505**	854	506
gka1e	16464.11796	16464	2000	**1188**	1264
gka1f	63012.37510	61194	2872	2741	**2655**
gka2a	6063.12603	6063	**303**	2132	396
gka2b	204.15121	121	1547	**1254**	1617
gka2c	6213.97887	6213	**275**	486	602
gka2d	6579.09478	6579	**931**	1020	1165
gka2e	23664.57541	23395	1814	**1750**	1864
gka2f	104379.99224	100161	**2420**	3839	2476
gka3a	6037.13660	6037	1117	1071	**978**

Table 5.10: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

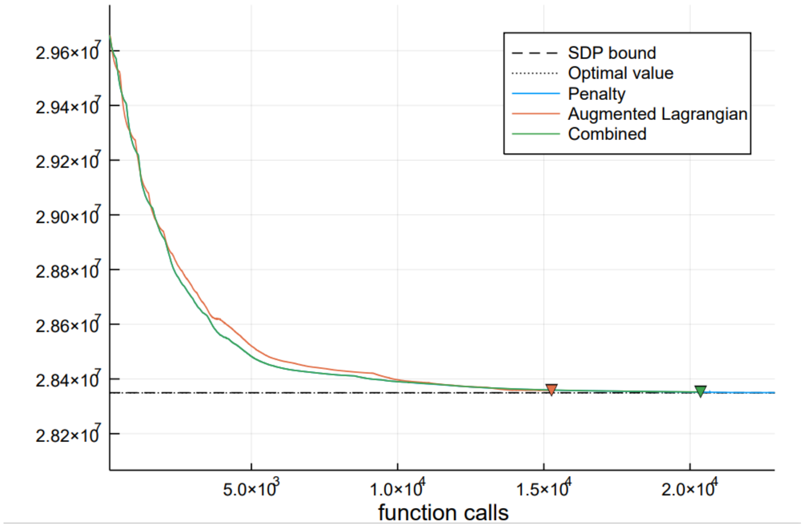


Figure 5.20: Bounds vs function calls for inequality augmented Lagrangian, penalty methods, and combined technique on graph t2g20-7777

Our results					
Graphs	The goal bound	Optimal value	Penalty	Aug	Combined
gka3b	289.76896	118	1813	**1292**	1832
gka3c	6665.93382	6665	**330**	600	366
gka3d	9261.28638	9261	**896**	1084	918
gka3e	25529.09562	25243	**1828**	2303	1866
gka3f	144920.36872	138035	2300	**2203**	3762
gka4a	8598.13797	8598	**786**	1169	827
gka4b	299.94767	129	2572	**2332**	2592
gka4c	7398.63055	7398	**415**	894	512
gka4d	10730.95505	10727	1471	**1385**	2056
gka4e	35821.08965	35594	1629	2081	**1347**
gka4f	180981.11092	172771	3122	**2929**	3066

Table 5.11: Combined penalty and augmented Lagrangian test Different problems of Biq Mic Library.

Our results					
Graphs	The goal bound	Optimal value	Penalty	Aug	Combined
gka5a	5737.99856	5737	**286**	489	650
gka5b	431.59163	150	**1740**	1773	1748
gka5c	7362.99129	7362	**465**	1160	610
gka5d	11721.96721	11626	1708	**1596**	1915
gka5e	36068.61159	35154	2405	2419	**2159**
gka5f	200748.68965	190507	4302	**3688**	3889
gka6a	3980.99348	3980	**224**	338	279
gka6b	553.22900	146	1764	1893	**1312**
gka6c	5825.45322	5824	**421**	851	518
gka6d	14207.48330	14207	1224	1744	**1195**
gka7a	4541.92594	4541	**232**	360	337
gka7b	635.43440	160	1693	2239	**1683**
gka7c	7225.98017	7225	**315**	700	413
gka7d	14567.10427	14476	**1849**	2058	2140
gka8a	11109.43487	11109	464	850	**459**
gka8b	849.89441	145	1207	1263	**1172**
gka8d	16352.38459	16352	**1091**	1426	1106
gka9b	955.31397	137	**1170**	1464	1312
gka9d	15722.60254	15656	1960	1546	**1508**

Table 5.12: Function calls and nodes to solve the problems Different problems of Biq Mic Library.

Our results					
Graphs	goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
pw01_100.0	2024.86045	2211	780	558	**491**
pw01_100.1	2068.33902	2060	1655	**890**	998
pw01_100.2	2040.44551	2032	951	**741**	897
pw01_100.3	2079.97438	2067	781	**584**	689
pw01_100.4	2039.01451	2039	1500	1279	**1036**
pw01_100.5	2108.05997	2108	1610	965	**900**
pw01_100.6	2040.06693	2032	1065	898	**798**
pw01_100.7	2082.95324	2074	991	**554**	987
pw01_100.8	2022.01521	2022	850	**687**	698
pw01_100.9	2022.18848	2005	740	**550**	604
pw05_100.0	8284.62754	8190	488	**333**	390
pw05_100.1	8117.05291	8045	478	498	**440**
pw05_100.2	8116.48548	8039	598	878	**570**
pw05_100.3	8189.39981	8139	787	**459**	837
pw05_100.4	8201.37564	8125	487	**387**	390
pw05_100.5	8224.13918	8169	598	498	**321**
pw05_100.6	8308.47846	8217	798	**370**	498
pw05_100.7	8313.06874	8249	522	657	**209**
pw05_100.8	8230.34665	8199	840	**498**	859
pw05_100.9	8160.27337	8099	687	**467**	687

Table 5.13: Function calls and nodes to solve the problems pw05_100

Our results					
Graphs	goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
pw09_100.0	13657.78513	13585	421	**289**	398
pw09_100.1	13494.91415	13417	478	**376**	465
pw09_100.2	13519.47082	13461	687	**300**	301
pw09_100.3	13706.17606	13656	745	**451**	587
pw09_100.4	13575.50698	13514	578	**366**	387
pw09_100.5	13639.05029	13574	590	687	**500**
pw09_100.6	13703.98302	13640	600	**398**	400
pw09_100.7	13579.47914	13501	476	334	**300**
pw09_100.8	13650.86654	13593	900	602	**590**
pw09_100.9	13713.99428	13658	882	**476**	598

Table 5.14: Function calls and nodes to solve the problems pw09_100

Our results					
Graphs	goal bound	Optimal value	Penalty fcalls	Aug fcalls	Combined fcalls
t2g10_5555	6062205.71512	6049461	**1789**	5330	5000
t2g10_6666	5784631.60681	5757868	311	1276	**678**
t2g10_7777	6515593.09284	6509837	**349**	432	2100
t2g15_5555	15051925.83740	15051133	8766	1109	**1456**
t2g15_6666	15764232.66991	15763716	1454	**1276**	6781
t2g15_7777	15269672.73355	15269399	5671	6789	**4542**
t2g20_5555	24839432.22377	24838942	2376	**1996**	2108
t2g20_6666	29304094.44929	29290570	2789	**2679**	2867
t2g20_7777	28349824.47416	28349398	3231	**1561**	2112
t3g5_5555	10998774.65556	10933215	8021	9087	**7543**
t3g5_6666	11745835.79499	11582216	987	**765**	8865
t3g5_7777	11840631.06716	11552046	4532	5652	**1342**
t3g6_5555	17534113.79105	17434469	**1589**	1608	1756
t3g6_6666	20315687.52881	20217380	2145	1945	**1897**
t3g6_7777	19493828.63175	19475011	**1548**	1897	1976
t3g7_5555	28303395.63876	28302918	2769	2867	**2543**
t3g7_6666	33612636.40663	33611981	3211	**2984**	3015
t3g7_7777	29264458.79157	29118445	678	**2445**	2598

Table 5.15: Function calls and nodes to solve the Different problems of Biq Mic Library.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

The objectives of this thesis are accomplished, and the following points were explained:

1. We tested the comparison between two methods, the penalty and the Lagrange augmentated method .
2. We demonstrate the properties of hypothetical aproximation and also we studied algorithms of the three methods.
3. We also used the graphs provided in the Big Mac library to evaluate the methods. Also, these figures included various features and properties with no small number of edges and vertices.
4. The augmented Lagrange method was created by developing a duality related to the near point problem. Numerical Experiments were conducted to test the enhanced binding procedure using graphs from the Biq Mac library [81].
5. The switch was developed between the penalty kick and the enhanced Lagrange method.
6. Improvement theoretical convergence properties.
7. The Improvement shows, the Lagrange augmentated is better that penalty method.

6.2 Future Work

The future line of research for this project can be explained in the following developments:

1. According to our results, a combined algorithm can be formulated that combines the penalty method and augmented Lagrangian method to reduce the convergence time.This would be possible by presenting a new method to switch between the two methods during particular times of the bounding procedure.
2. This work can be improved by considering other problems, such as generic binary quadratic problems and different problems from Big.Mac library.
3. More graphs can be tested, and results can be extracted to provide a better understanding of the algorithm performance.
4. Large scale binary quadratic problems can be studied.
5. The bounding procedure can be further understood by investigating theoretical convergence properties.
6. We comporse with different method.

REFERENCES

[1] Ahmed Al-Jilawi. *Solving the Semidefinite Programming Relaxation of Max-cut Using an Augmented Lagrangian Method.* Northern Illinois University, 2019.

[2] Vladimir E Alekseev. On easy and hard hereditary classes of graphs with respect to the independent set problem. *Discrete Applied Mathematics*, 132(1-3):17–26, 2003.

[3] Diogo V Andrade, Mauricio GC Resende, and Renato F Werneck. Fast local search for the maximum independent set problem. *Journal of Heuristics*, 18(4):525–547, 2012.

[4] Roberto Andreani, Ernesto G Birgin, José Mario Martínez, and María Laura Schuverdt. On augmented lagrangian methods with general lower-level constraints. *SIAM Journal on Optimization*, 18(4):1286–1309, 2008.

[5] Neculai Andrei. An unconstrained optimization test functions collection. *Adv. Model. Optim*, 10(1):147–161, 2008.

[6] Miguel F Anjos and Anthony Vannelli. Computing globally optimal solutions for single-row layout problems using semidefinite programming and cutting planes. *INFORMS Journal on Computing.*

[7] Sanjeev Arora, Béla Bollobás, and László Lovász. Proving integrality gaps without knowing the linear program. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 313–322. IEEE, 2002.

[8] Adil Bagirov, Napsu Karmitsa, and Marko M Mäkelä. *Introduction to Nonsmooth Optimization: theory, practice and software.* Springer, 2014.

[9] Alampallam Venkatachalaiyer Balakrishnan. *Introduction to optimization theory in a Hilbert space*, volume 42. Springer Science & Business Media, 2012.

[10] Grey Ballard, Erin Carson, James Demmel, Mark Hoemmen, Nicholas Knight, and Oded Schwartz. Communication lower bounds and optimal algorithms for numerical linear algebra. *Acta Numerica*, 23:1–155, 2014.

[11] Thomas Bartz-Beielstein, Marco Chiarandini, Luís Paquete, and Mike Preuss. *Experimental methods for the analysis of optimization algorithms.* Springer, 2010.

[12] Claude Berge. *Graphs and Hypergraphs (North-Holland Mathematical Library; V. 6).* Elsevier, 1973.

[13] Dimitri Bertsekas. *Convex optimization theory.* Athena Scientific, 2009.

[14] Dimitri Bertsekas. *Convex optimization algorithms.* Athena Scientific, 2015.

[15] Dimitri Bertsekas, Angelia Nedic, and Asuman Ozdaglar. *Convex analysis and optimization*, volume 1. Athena Scientific, 2003.

[16] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[17] Ewald Bodewig. *Matrix calculus.* Elsevier, 2014.

[18] Paul T Boggs and Jon W Tolle. Sequential quadratic programming. *Acta numerica*, 4:1–51, 1995.

[19] RI Boţ, G Kassay, and Gert Wanka. Strong duality for generalized convex optimization problems. *Journal of Optimization Theory and Applications*, 127(1):45–70, 2005.

[20] Nicolas Bourbaki. *Topological vector spaces: Chapters 1–5*. Springer Science & Business Media, 2013.

[21] Nicolas Bourgeois, Bruno Escoffier, and Vangelis Th Paschos. Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms. *Discrete Applied Mathematics*, 159(17):1954–1970, 2011.

[22] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[23] Boštjan Brešar, František Kardoš, Ján Katrenič, and Gabriel Semanišin. Minimum k-path vertex cover. *Discrete Applied Mathematics*, 159(12):1189–1195, 2011.

[24] Alberto Colorni, Marco Dorigo, Francesco Maffioli, Vittorio Maniezzo, GIOVANNI Righini, and Marco Trubian. Heuristics from nature for hard combinatorial optimization problems. *International Transactions in Operational Research*, 3(1):1–21, 1996.

[25] AO Converse and CI Huber. Communication. effect of holdup on batch distillation optimization. *Industrial & Engineering Chemistry Fundamentals*, 4(4):475–477, 1965.

[26] Stephen Cook. The p versus np problem. *The millennium prize problems*, pages 87–104, 2006.

[27] Colleen D Cutler. Strong and weak duality principles for fractal dimension in euclidean space. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 118, pages 393–410. Cambridge University Press, 1995.

[28] W Erwin Diewert. Applications of duality theory. 1974.

[29] Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pages 439–485, 2005.

[30] Mahmoud Mahmoud El-Alem. *A global convergence theory for a class of trust region algorithms for constrained optimization.* PhD thesis, Rice University, 1988.

[31] Dominique Feillet, Pierre Dejax, and Michel Gendreau. Traveling salesman problems with profits. *Transportation science*, 39(2):188–205, 2005.

[32] Silvia Ferrari and Robert F Stengel. Smooth function approximation using neural networks. *IEEE Transactions on Neural Networks*, 16(1):24–38, 2005.

[33] AV Fiacco and GP McCormick. Sequential unconstrained minimization techniques for nonlinear programming. *A primal dual method. Munagement Sci*, 10, 1968.

[34] Roger Fletcher. A general quadratic programming algorithm. *IMA Journal of Applied Mathematics*, 7(1):76–91, 1971.

[35] Michel Fortin and Roland Glowinski. *Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems.* Elsevier, 2000.

[36] Marguerite Frank, Philip Wolfe, et al. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[37] Michael R Garey, David S Johnson, and Larry Stockmeyer. Some simplified np-complete problems. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63, 1974.

[38] Saul I Gass. *Linear programming: methods and applications.* Courier Corporation, 2003.

[39] Raymond Greenlaw, H James Hoover, Walter L Ruzzo, et al. *Limits to parallel computation: P-completeness theory.* Oxford University Press on Demand, 1995.

[40] Osman Güler. *Foundations of optimization*, volume 258. Springer Science & Business Media, 2010.

[41] Eitan M Gurari and Oscar H Ibarra. The complexity of decision problems for finite-turn multicounter machines. *Journal of Computer and System Sciences*, 22(2):220–229, 1981.

[42] MR Hestenes. Multipliers and gradient methods—in computing met h ods in optimization problems, vol. 2, la zadeh, lw neustadt, av balakrishnan eds, 1969.

[43] Michael Hintermüller, Michael Hinze, and Ronald HW Hoppe. Weak-duality based adaptive finite element methods for pde-constrained optimization with pointwise gradient state-constraints. *Journal of Computational Mathematics*, pages 101–123, 2012.

[44] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of convex analysis*. Springer Science & Business Media, 2004.

[45] Sha Hu et al. *Semidefinite relaxation based branch-and-bound method for nonconvex quadratic programming*. PhD thesis, Massachusetts Institute of Technology, 2006.

[46] Thorsten Joachims. Making large-scale svm learning practical. Technical report, Technical report, 1998.

[47] H Th Jongen, F Twilt, and GW Weber. Semi-infinite optimization: structure and stability of the feasible set. *Journal of optimization theory and applications*, 72(3):529–552, 1992.

[48] Christian Kanzow, Daniel Steck, and Daniel Wachsmuth. An augmented lagrangian method for optimization problems in banach spaces. *SIAM Journal on Control and Optimization*, 56(1):272–291, 2018.

[49] Stefan E Karisch and Franz Rendl. Semidefinite programming and graph equipartition. *Topics in Semidefinite and Interior-Point Methods*, 18(77-95):25, 1998.

[50] Bernhard H Korte, Jens Vygen, B Korte, and J Vygen. *Combinatorial optimization*, volume 1. Springer, 2011.

[51] Robert L Kosut and Daniel A Lidar. Quantum error correction via convex optimization. *Quantum Information Processing*, 8(5):443–459, 2009.

[52] Nathan Krislock, Jérôme Malick, and Frédéric Roupin. Improved semidefinite bounding procedure for solving max-cut problems to optimality. *Mathematical Programming*, 143(1):61–86, 2014.

[53] Nathan Krislock, Jérôme Malick, and Frédéric Roupin. Biqcrunch: A semidefinite branch-and-bound method for solving binary quadratic problems. *ACM Transactions on Mathematical Software (TOMS)*, 43(4):1–23, 2017.

[54] Peter Kuchment. Analysis on graphs and its applications. In *Proc. Symp. Pure. Math*, volume 77, pages 0802–3442, 2008.

[55] Harold W Kuhn and Albert W Tucker. Nonlinear programming. In *Traces and emergence of nonlinear programming*, pages 247–258. Springer, 2014.

[56] Gert RG Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine learning research*, 5(Jan):27–72, 2004.

[57] Monique Laurent and Franz Rendl. Semidefinite programming and integer programming. *Handbooks in Operations Research and Management Science*, 12:393–514, 2005.

[58] Yu Liu, Xiaoxi Ling, Zhewen Shi, Mingwei Lv, Jing Fang, and Liang Zhang. A survey on particle swarm optimization algorithms for multimodal function optimization. *J. Softw.*, 6(12):2449–2455, 2011.

[59] Bruno F Lourenço, Ellen H Fukuda, and Masao Fukushima. Optimality conditions for nonlinear semidefinite programming via squared slack variables. *Mathematical Programming*, 168(1):177–200, 2018.

[60] Zhi-Quan Luo, Wing-Kin Ma, Anthony Man-Cho So, Yinyu Ye, and Shuzhong Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20–34, 2010.

[61] Wing-Kin Ma, Chao-Cheng Su, Joakim Jaldén, Tsung-Hui Chang, and Chong-Yung Chi. The equivalence of semidefinite relaxation mimo detectors for higher-order qam. *IEEE Journal of Selected Topics in Signal Processing*, 3(6):1038–1052, 2009.

[62] Wing-Kin Ma, Ba-Ngu Vo, Timothy N Davidson, and Pak-Chung Ching. Blind ml detection of orthogonal space-time block codes: Efficient high-performance implementations. *IEEE Transactions on Signal Processing*, 54(2):738–751, 2006.

[63] Jérôme Malick. The spherical constraint in boolean quadratic programs. *Journal of Global Optimization*, 39(4):609–622, 2007.

[64] Jérôme Malick and Frédéric Roupin. On the bridge between combinatorial optimization and nonlinear optimization: a family of semidefinite bounds for 0–1 quadratic problems leading to quasi-newton methods. *Mathematical Programming*, 140(1):99–124, 2013.

[65] Garth P McCormick. Nonlinear programming; theory, algorithms, and applications. Technical report, 1983.

[66] Ofer Meshi and Amir Globerson. An alternating direction method for dual map lp relaxation. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 470–483. Springer, 2011.

[67] Camilo Ortiz and René Meziat. Semidefinite relaxations of dynamical programs under discrete constraints. *Optimization Letters*, 4(4):567–583, 2010.

[68] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[69] J Pardalos and M Resende. On maximum clique problems in very large graphs. *DIMACS series*, 50:119–130, 1999.

[70] Panos M Pardalos and Henry Wolkowicz. *Novel approaches to hard discrete optimization*. Number 37. American Mathematical Soc., 2003.

[71] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[72] Svatopluk Poljak and Franz Rendl. Nonpolyhedral relaxations of graph-bisection problems. *SIAM Journal on Optimization*, 5(3):467–487, 1995.

[73] Gerhard Reinelt. *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag, 1994.

[74] Robert Endre Tarjan and Anthony E Trojanowski. Finding a maximum independent set. *SIAM Journal on Computing*, 6(3):537–546, 1977.

[75] Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü. Sdpt3—a matlab software package for semidefinite programming, version 1.3. *Optimization methods and software*, 11(1-4):545–581, 1999.

[76] Paolo Toth. Optimization engineering techniques for the exact solution of np-hard combinatorial optimization problems. *European journal of operational research*, 125(2):222–238, 2000.

[77] Jeffrey D. Ullman. Np-complete scheduling problems. *Journal of Computer and System sciences*, 10(3):384–393, 1975.

[78] Lieven Vandenberghe and Stephen Boyd. Applications of semidefinite programming. *Applied Numerical Mathematics*, 29(3):283–299, 1999.

[79] Hui Wang and George N Rouskas. Traffic grooming in optical networks: Decomposition and partial linear programming (lp) relaxation. *Journal of Optical Communications and Networking*, 5(8):825–835, 2013.

[80] JD Weber and TJ Overbye. A two-level optimization problem for analysis of market bidding strategies. In *1999 IEEE Power Engineering Society Summer Meeting. Conference Proceedings (Cat. No. 99CH36364)*, volume 2, pages 682–687. IEEE, 1999.

[81] Angelika Wiegele. Biq mac library—a collection of max-cut and quadratic 0-1 programming instances of medium size. *Preprint*, 51, 2007.

[82] Henry Wolkowicz, Romesh Saigal, and Lieven Vandenberghe. *Handbook of semidefinite programming: theory, algorithms, and applications*, volume 27. Springer Science & Business Media, 2012.

[83] Ali R Yildiz. Comparison of evolutionary-based optimization algorithms for structural design optimization. *Engineering applications of artificial intelligence*, 26(1):327–333, 2013.

[84] Uwe Zimmermann. Discrete optimization (r. gary parker and ronald l. rardin), 1990.

الخلاصة

الهدف الرئيسي من هذه الاطروحة هو تحديد طريقة متفوقة تدمج بشكل أسرع مشكلة المجموعة المستقلة القصوى (MIS). بالإضافة إلى ذلك ، قمنا بتطوير تقنية خوارزمية جديدة لحل المشكلات الثنائية التربيعية (MIS) بمتغيرات واسعة النطاق (100،80،50 من الرؤوس) من مكتبة Biq Mac. هذه تقنية جديدة من الخوارزمية تعطينا كفاءة عالية ، وسرعة تفوق السابقة. تتمثل إحدى المنهجيات في الارتباط بين طريقة الجزاء وطريقة لاغرانج المعززة. لقد قمنا أيضًا ببناء خصائص الجمع الافتراضية لهذه الطرق بالإضافة إلى تحسينها تم تحسين خصائص التقارب النظري لتوضيح أن الخوارزمية الجديدة اختيار لحل استرخاء SDP للمشاكل الثنائية التربيعية. سيستفيد مستخدمو برنامج الحل المطور لاسترخاء SDP من خلال الحصول على نتائج أسرع. في هذه الدراسة تم تطوير نهج جديد لاستبدال طريقة الجزاء بطريقة لاغرانج المعززة. أيضًا ، وفقًا لقيمة المعلمة ، تم تطوير طريقة هجينة تقوم بالتبديل بين الطريقتين تسمى الطريقة المدمجة. تمت دراسة الطرق الثلاثة المقترحة لمسألة لاغرانج الموسعة ، ومشكلة الجزاء لمشكلات البرمجة الخطية (LP). نتيجة لذلك ، تم تبرير نهجين فقط والموافقة عليهما كطرق صالحة لاستخدامها في حل استرخاء SDP. استخدمنا لغة جوليا في هذه الدراسة.

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعة بابل

كلية التربية للعلوم الصرفة

قسم الرياضيات

التحسين العددي على البرمجة شبه المحددة

أطروحة

مقدمة الى مجلس كلية التربية للعلوم الصرفة/ جامعة بابل كجزء من متطلبات

نيل درجة الدكتوراه فلسفة في التربية / الرياضيات

من قبل

عباس مصلح سلمان عباس

بأشراف

أ.م.د. احمد صباح احمد الجيلاوي

1444 هـ

2022م