Innovative: International Multi-disciplinary Journal of Applied Technology
(ISSN 2995-486X) VOLUME 03 ISSUE 9, 2025

Solving the K-Cluster Optimization Problems in Combinatorial Optimization

Mouid Abd Alameer Abood
Ministry of Education, Babylon Education Directorate

Abstract:

The optimization problem is demonstrated to be an NP-Hardness problem in this research using a sound methodology. First off, one of the most significant NP-hardness issues in combinatorial optimization is the K-cluster problem. Second, an issue is considered difficult if it cannot be resolved specifically (i.e., in polynomial time) by a workable algorithm. Additionally, the approach taken in this paper is to use a method to demonstrate that the problem is NP-Hard. If any problem from NP can be reduced to it, as shown by means reductions, then the problem is NP-Hard.

Keywords: Graph clustering, K-cluster problem, semidefinite programming, combinatorial optimization problems, and polynomial-time.

INTRODUCTION

A k-cluster is a subgraph on k vertices (nodes) (V) of a graph G=(V, E) that maximizes the number of edges (E) [1,3], which means: The average degree, which is represented by the greatest density [2] of a subgraph G=(V,E) on k vertices, is computed given the graph G=(V,E) and the parameter k. Clustering is a widely used method in multivariate data analysis. Its goal is to look at the inherent natural shapes of data items, where objects are as similar to one another as feasible within a cluster and as dissimilar from one another as possible between clusters. The clusters' equivalence groups offer a means of generalizing the properties of the data objects. Clustering techniques are used. numerous fields, including Furthermore, a discriminant analysis' goal is to strengthen the categorization that has previously been provided by enhancing class demarcations, whereas a cluster analysis' first step is to describe the class structure, making them completely unrelated concepts. Clustering is an examination of the outcomes of exploration. The parameters of the resulting analysis of the cluster may, as a result, be unknown to the explorer [2,4]. Since the k cluster problem is one of the applications of semidefinite programming, we will now give a brief field overview of semidefinite programming before reviewing the fundamental research tools.

Clustering of Combinatorial Optimization

numerous significant combinatorial optimization problems can be cast as problems in NP, usually it would be dealing with approach the task of graph clustering from the common strategy of combinatorial optimization. In other words, the problem is formalized by introducing an objective function that assigns a numerical score to each discrete clustering of a graph, measuring how well it embodies community structure [1]. Optimizing the objective function over all possible clustering will then return "the best" partitioning of the graph with respect to a well-defined measure. As an example, one way to partition a graph G into two pieces is to identify a set of nodes S that minimizes the following function f, referred to as the normalized cut objective [7,11]:

$$f(S) = \frac{\text{The number of edges leaving S}}{\text{The number of edge endpoints in S}} + \frac{\text{The number of edges leaving S}}{\text{The number of edge endpoints not in S}}$$

Minimizing f will produce two clusters (nodes in S, and nodes not in S) that are both nontrivial in size and share few edges with each other [12].

NP-Hard in Clusters

A graph $G = (V, E)$ is referred to as a cluster graph when each connected component of G is a full graph. For instance, a clique is a fully-completed subgraph of a graph (V, E), as shown in Figure (1) [10]. The optimization problem is to identify the maximum clique in the graph, whereas the decision problem is to determine whether the graph has a clique of size k. The following proof shows that a clique choice problem is an NP-hard problem for big graphs. Figuring out the size of a clique is an NP problem because, for large graphs, this answer cannot typically be verified in polynomial time. If we are given a clique, we can quickly verify if it is a clique of size k by counting the number of vertices in the clique.

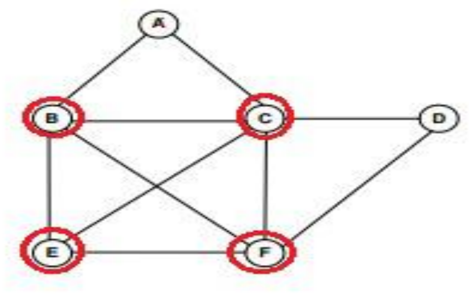


Figure 1: An Example of Cliques in a Graph.

K- Clusters Problem:

Finding a subgraph with the highest weight and precisely k nodes ($1 < k < n - 1$) is required for the K-clusters problem. According to Deogun et al. (1997), [3,8,9,13], this is a classic combinatorial optimization issue that is both NP-hard and difficult to estimate. The semidefinite-based branch-and-bound technique was previously used to solve the K-CLUSTER problem to optimality, and it outperformed the convex quadratic relaxation method. Clustering is often a widely used multivariate data analysis technique. The goal is to look into the data items' innate natural structure, which favors objects being as similar to one another and as diverse from one another as possible inside a cluster. The equivalence classes of the clusters offer a means of generalizing over the data objects and their properties [3]. Numerous disciplines, including health, psychology, economics, and pattern recognition, use clustering techniques. Clustering is frequently mistaken for classification or discriminant analysis. The two methods of data analysis are characterized by the following distinctions and correspond to different concepts:

1. Clustering differs from classification in that, the classification allocates items to

pre-defined classes, whereas clustering requires no previous knowledge of the object classes or their members.

2. The cluster analysis differs from a discriminant analysis in that the former seeks to enhance an existing classification by reinforcing class demarcations, whilst the latter requires first establishing the class structure. (see Figures 2 and 3).

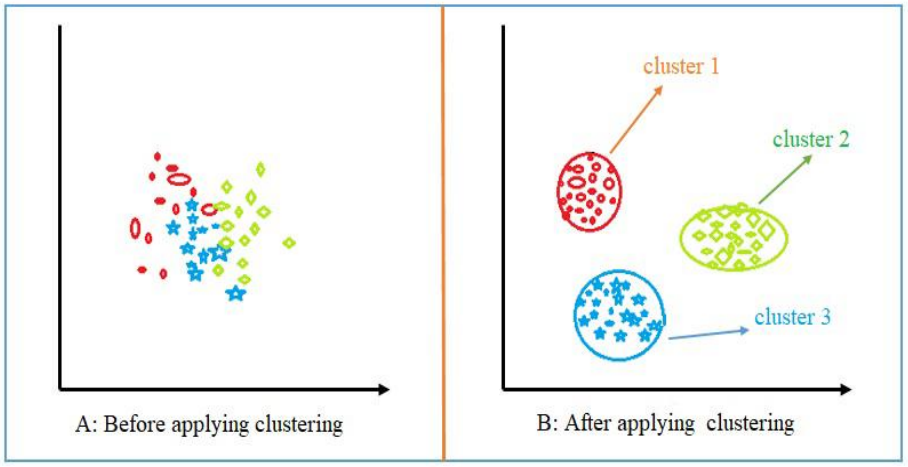


Figure 2: An example of cluster analysis.

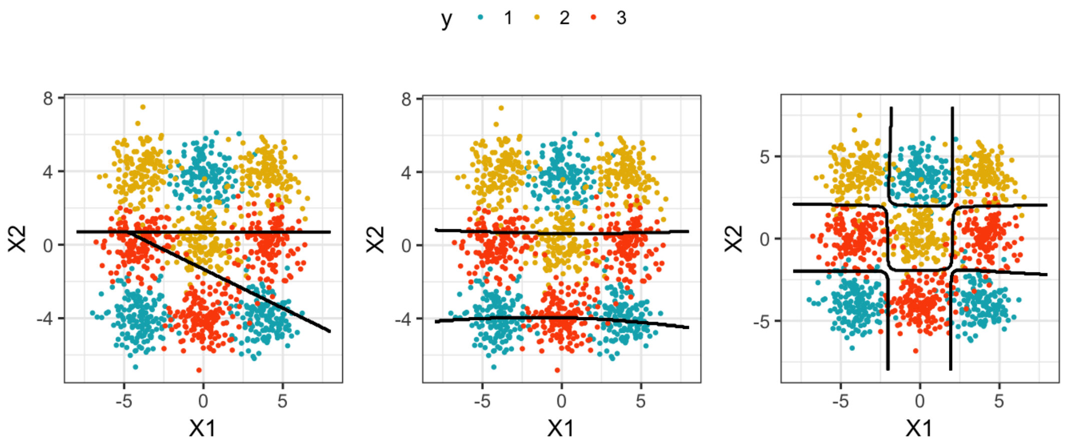


Figure 3: An example of discriminant cluster analysis.

Clustering is a type of data exploration . As a result, the explorer may have no or limited knowledge of the parameters of the cluster analysis that results [5]. Clustering is used in a variety of ways and the objective is to figure out everything of the following:

1. The size of the clusters.

2. The clusters' absolute and relative locations.

3. The number of clusters.

4. Cluster density that is clusters based on density are dense areas in the data space

divided by sparser portions.

Approximate Method Augmented Lagrangian and Penalty methods

An extremely active field in optimization is approximation approaches. Think about the convex function $\mathcal{G} : R^n \rightarrow R$ that minimizes to a convex set X. To replace G and X with approximate $\mathcal{G}^k$ and approximate X^k. is the purpose of approximation methods . The approximation approach will only work if the approximation is easier than the actual problem. We made an effort to determine k for every cycle.

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

Consequently, at the following iteration, the approximation that depends on the new point x^{k+1} generates $\mathcal{G}^{k+1}$ and x^{k+1}. This concept serves as the foundation for several excellent approximation techniques, including the penalty approach, the augmented Lagrangian method, and interior point methods.The penalized and modified Lagrangian methods are the main topics of our research [1]. Constraints generally make algorithmic solutions more difficult and reduce the number of workable solutions for optimization problems. So it only makes sense to attempt to loosen limitations by roughly estimating the pertinent indicator function. with instance, swap out limits with punishment functions that come with high financial penalties [15]. Given by is the linear equality constraint issue.

Min $\langle c, x \rangle$

subject to $\langle a_i, x \rangle = b_i , i = 1, \dots, l,$

$$x \in X.$$

substituted a penalized version for the aforesaid issue.

Min $\langle c, x \rangle + \alpha^k \sum_{i=1}^{l} P_q(\langle a_i, x \rangle - b_i)$

subject to $x \in X.$

The solution X^k of the penalized issue tends to reduce the constraint violation as the scalar α^k approaches zero, which results in an increasingly accurate approximation of the original problem. The optimal answer to each approximating problem should be used to begin iterating the next approximating problem. This is a crucial practical point. One choice for Pq is the quadratic punishment function, and the penalized problem (1.3) looks like this:

([12], [14])

minimize $\langle c, x \rangle + \frac{1}{2\alpha^k} \|A_x - b\|^2$

subject to $x \in X.$

where $A_x = b$ represents the system of equation $\langle a_i, x \rangle = b_i, i = 1, \dots, l.$. The penalty function approach, where we add a linear component to $P_q\ (y)$,, involving a multiplier vector $y^n \in R^n$. is much improved by the augmented Lagrangian method. After that, we resolve the issue in place of problem (1.3).

minimize $\langle c, x \rangle + (y^k)^T (Ax - b) + \frac{1}{2\alpha^k} \|A_x - b\|^2$

subject to $x \in X.$

The multiplier vector y^k is updated by a method that attempts to approximate an optimal dual solution [6], so that after the aforementioned problem's x^k solution is discovered.

$$y^{k+1} = y^k + \frac{1}{\alpha^k} (A_{x^k} - b)$$

This is referred to as It is known as first-order augmented Lagrangian techniques or the first-order method of multipliers. For equality and inequality criteria, the augmented Lagrangian and penalty techniques are both applicable [12].

Optimization Algorithms Design

Finding the optimal design parameters that also meet the design requirements is the goal of design optimization. In real terms, this implies that parametric optimization algorithms automatically test different iterations of your design based on the variables and goals you have set. A mass decrease of a design would be a typical optimization. In order to decrease the block ([8], [11]), specific model parameters are chosen as variables that can be altered by the algorithms. A maximum stress level is also established as an aim that the algorithms must meet.

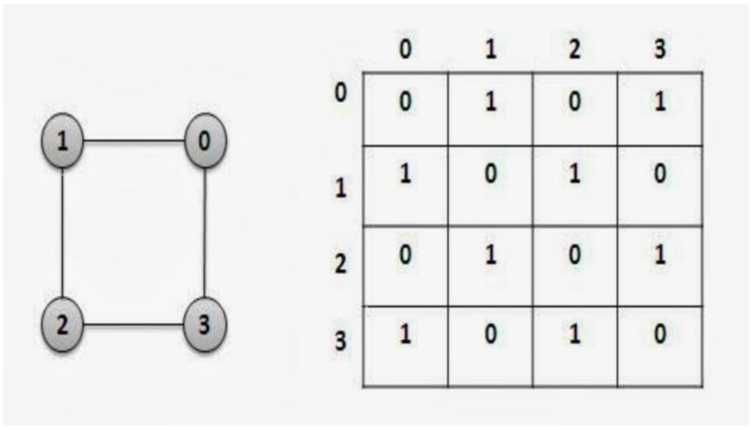
Definitions and Properties

1. (Adjacency matrix) [3] n-vertex graph with the adjacency matrix G $= (V, E)$ is the matrix $x \in M_n$ whose entries are:

a_{ij} = Indicator 1 denotes the existence of an edge between vertices i and j.

a_{ij} = The value 0 denotes the lack of an edge between vertices i and j.

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



	0	1	2	3
0	0	1	0	1
1	1	0	1	0
2	0	1	0	1
3	1	0	1	0

Figure 4: Undirected Graph Adjacency Matrix Representation

Definition 2. [11] The output that you wish to maximize or decrease is referred to as an objective function. For instance, in aeronautical engineering, the goal is frequently to reduce weight while the aim function is typically to maximize portfolio value. In the following illustration, the mountain climber's target function is to reach the highest peak, hence the goal function is to maximize, (see Figure 5).

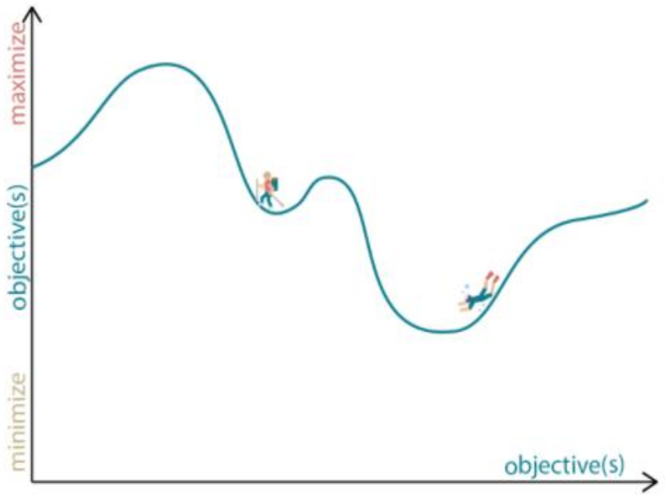


Figure 5: Simple example of objective function

Relaxation for Semidefinite Programming

Let's begin by going over the concept of convex relaxations. One of the most effective methods for creating approximate polynomial-time algorithms for NP-hard optimization problems, such as Chromatic Number, MAX-CUT, and Minimum Vertex Cover, among others, is this one [1]. In order to construct approximation approaches for these issues, the issue is rewritten as an integer program. The integer program is then transformed into a convex program that can be solved in polynomial time, such as a linear program (LP) or a semidefinite program (SDP). Then, by developing a (perhaps randomized) polynomial-time method to convert the result of such a convex relaxation into an integer solution for the combinatorial issue, or "rounding," the solution to the combinatorial problem is discovered. However, semidefnite programming is not a recent concept of combinatorial optimization. In fact, semidefinite programming has recently gained popularity as a method for developing more efficient algorithms for approximating hard combinatorial optimization problems and, more broadly, polynomial optimization problems, which include optimizing a polynomial objective function over a simple closed semi-algebraic set. Since its introduction in the 1990s, the semidefinite relaxation approach has sparked a lot of interest in combinatorial optimization. Finally, semidefinite relaxation is now recognized as an effective method and have close bounds for a wide range of complicated problems and the strategy would be to replace a binary vector variable with a continuous matrix variable,

The General Formula of K-cluster Problem and Bound Procedure

Problem with K-Cluster entails locating a subgraph with the heaviest weight and exactly k nodes ($1 < k < n - 1$) when there is an edge weighted graph with n vertices. This is a classic combinatorial optimization problem, also known as the When all edge weights are equal to 1, the following problems arise: (heaviest k-subgraph problem), (k-dispersion problem), (k-defence-sum problem), and (densest subgraph problem). The k-cluster problem entails evaluating a subset $S \subseteq V$ of k vertices such that the number of the weights of the edges between vertices in S is maximized according to given a graph G = (V,E). Letting n = |V | denote the number of vertices, and w_{ij} denote the edge weight for $ij \in E$ and $w_{ij} = 0$ for $ij \notin E$ the problem can be modeled as the optimization problem:

Max $\frac{1}{2} Z^T W z$

S.T $e^{\top} z = k$

$$z \in \{0,1\}^n, k \in Z.$$

The Penalty and Augmented Lagrangian Methods

The goal of an optimization issue is to maximize or minimize a function while taking certain limitations into account. The overall optimization issue provided by [12, 13,]:

Min f(x)

S.T $x \in X$.

The 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 (4.1) if there exists a neighborhood B(x^*, t) such that $f(x^*) \leq f(x)\, for\, all\, x \in B(x^*, t) \cap X = \{x \in X \mid \| x - x^* \| \leq t\}$

The basic penalty method and augmented Lagrangian method are summarized in

Algorithms (1), (2) respectively.

```
Algorithm 1: Penalty Method
1 Minimize the dual function.
2 Reduce α.
3 Repeat.
```

```
Algorithm 2: Augmented Lagrangian Method
1 For given $X^{\circ} \in S^n, y_0, and \quad \alpha^0 > 0$.
2 Find $y^{k+1}$ such that
      $y^{k+1} = argmin_y L\alpha^k(y, X^k)$.
3 Update the Lagrange multiplier $X^k$.
4 Reduce α.
5 Repeat.
```

Julia Language (JuliaBox)

A dynamic, high-performance, high-level programming language is Julia. Although it can be used to create any application because it is a general-purpose language. A type system with parametric polymorphism in a dynamic programming language with multiple dispatches as its primary programming paradigm is one of Julia's design's distinguishing features. In order to develop a free language that was both high-level and quick, Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman founded Julia in 2009. The team debuted a website on February 14, 2012, along with a blog post describing the goal of the language [2]. Julia is used by setting up a user account, signing into the website, and working on it while connected to the internet. It is possible to work on it without access to the internet, but only if the necessary packages are present.

Numerical Results

In this section, we'll go through our results and evaluate the performance of the suggested development method. These tests were performed using a particular graph that was imported from the Biq Mac library. [8]. There are 50 nodes connected by 1214 edges in these 106 graphs as shown in Figures (6) respectively, as well as on the other types of graphs. we implemented the Augmented Lagrangian, Penalty and Hybrid methods for solving the semidefinte programming. This section's figures display the quantity of function calls made by the different techniques of solving K-CLUSTER PROBLEM. The exact solution to our problem was given by SB and CSDP solvers [11] . Types graphs of various sizes were tested, and the results were extracted and displayed in this part. Finally, our results depend on a new approach by using backtracking line search instead of huge line search.

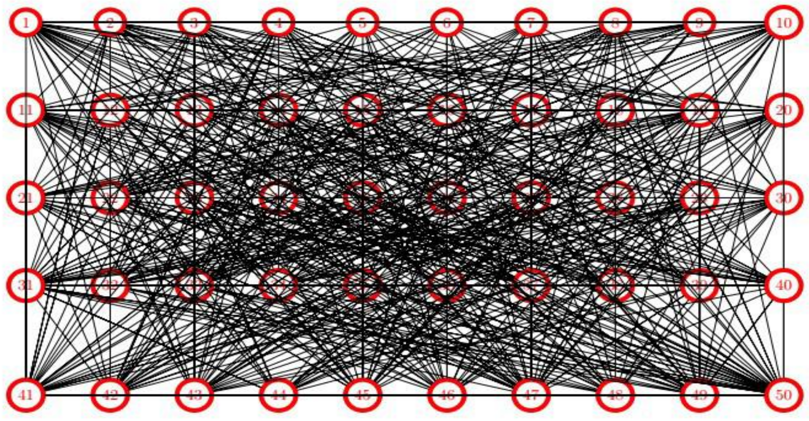


Figure 6: design the graph from Big Mac library and contain (50 vertices and 1214 edges)

Elementary Numerical Results of Graphs g05-60 (Function Calls)

In order to achieve the idea accurately and to obtain results through which it is possible to infer the fastest method of convergence to the bound of optimal solution, we've picked various problems (g05-60) from the Big Mac Library to work on as shown in Table (1) and the graphs in Figure (6). In fact, these problems contain 60 nods and 885 edges. Also, we used three approaches in these tests: Penalty method, Augumented Lagrangian method and Hybrid 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 cluster.

Our results

Problem	PENfcalls	AUG.fcalls	Hybrid fcalls
g05 60.0	753	587	531
g05 60.1	851	443	493
g05 60.2	409	252	224
g05 60.3	890	851	523
g05 60.4	491	351	463
g05 60.5	873	321	444
g05 60.6	503	236	261
g05 60.7	789	514	481
g05 60.8	444	231	213
g05 60.9	517	270	394
Total winner fcalls	0	5	5

Table 1: Numerical results of graphs (g05-60)

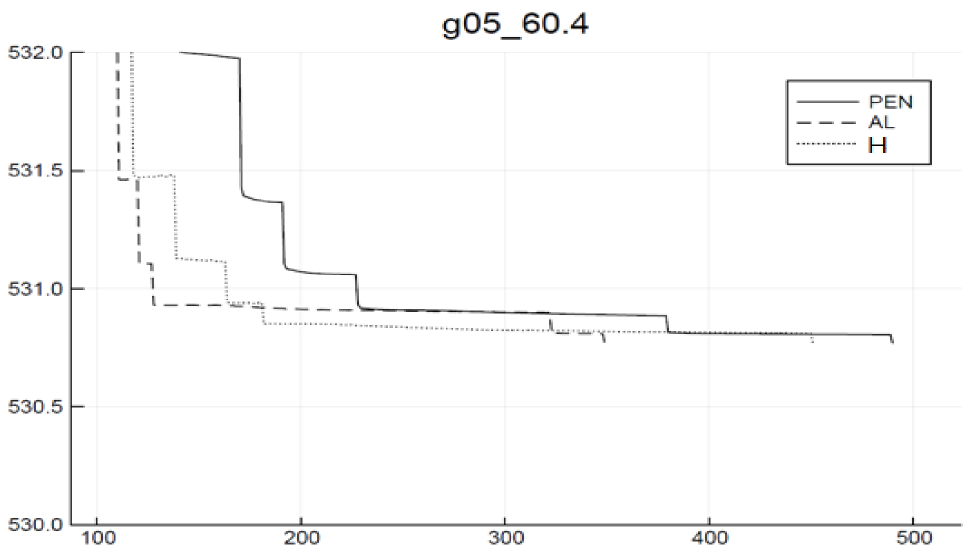


Figure 6: Convergence of the optimal solution between the methods

Conclusions

The peaper goals and objectives were met, and the following information was gleaned:

1. Development anew relaxation of the feasible region to provide the Augmented lagrangian method based on penalty method.
2. The outcomes showed that, in terms of meeting the goal bound, the Augmented Lagrangian Method was superior to the Penalty Method.
3. The Hybrid method, which alternates between the penalty and inequality Augmented Lagrangian methods, is also put to the test. The results showed that the Hybrid approach outperformed the two alternatives separately.
4. It has been proven that the K-CLUSTER combinatorial optimization problems is
5. NP-Hardness problem in graph clustering with large scale of variables.
6. A new algorithm has been developed by which the bound is improved and it works on NP-Hardness problem.
7. Develop theoretical convergence properties.

REFERENCES

1. Ahmed Al-Jilawi. Solving the Semidefinite Programming Relaxation of Max-cut Using an Augmented Lagrangian Method. Northern Illinois University, 2019.
2. Leonardo Jesus Almeida and Alneu de Andrade Lopes. An ultra-fast modularitybased graph clustering algorithm. In Proceedings 14th Portuguese Conference on Artificial Intelligence (EPIA)-Web and Network Intelligence Track, pages 1–9, 2009.
3. Ahmed Alridha and Ahmed Sabah Al-Jilawi. Mathematical programming computational for solving np-hardness problem. In Journal of Physics: Conference Series, volume 1818, page 012137. IOP Publishing, 2021.
4. Ahmed Alridha, Abbas Musleh Salman, and Ahmed Sabah Al-Jilawi. The applications of np-hardness optimizations problem. In Journal of Physics: Conference Series, volume 1818, page 012179. IOP Publishing, 2021.

5. Niclas Andr´easson, Michael Patriksson, and Anton Evgrafov. An introduction to continuous optimization: foundations and fundamental algorithms. Courier Dover Publications, 2020.
6. Neculai Andrei. Penalty and augmented lagrangian methods. In Continuous Nonlinear Optimization for Engineering Applications in GAMS Technology, pages 185–201. Springer, 2017. 152
7. Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. Complexity and approximation: Combinatorial optimization problems and their approximability properties. Springer Science & Business Media, 2012.
8. Hideyuki Azegami. Shape optimization problems of domain variation type. In Shape Optimization Problems, pages 427–566. Springer, 2020.
9. Elham Photoohi Bafghi. Clustering of customers based on shopping behavior and employing genetic algorithms. Engineering, Technology & Applied Science Research, 7(1):1420–1424, 2017.
10. Rangaswami Balakrishnan and Kanna Ranganathan. A textbook of graph theory. Springer Science & Business Media, 2012.
11. Venkataramanan K Balakrishnan. Introductory discrete mathematics. Courier Corporation, 2012.
12. Dimitri Bertsekas. Convex optimization algorithms. Athena Scientific, 2015.
13. Dimitri P Bertsekas,WHager, and O Mangasarian. Nonlinear programming. Athena scientific belmont. Massachusets, USA, 1999.
14. Dario Andrea Bini, Fabio Di Benedetto, Eugene Tyrtyshnikov, and Marc Van Barel. Structured Matrices in Numerical Linear Algebra: Analysis, Algorithms and Applications, volume 30. Springer, 2019.
15. Ernesto G Birgin and Jos´e Mario Mart´ınez. Practical augmented Lagrangian methods for constrained optimization. SIAM, 2014.
16. Michael Birsak. Discrete optimization on graphs and grids for the creation of navigational and artistic imagery. PhD thesis, Wien, 2018.