ORTA DOĞU TEKNİK ÜNİVERSİTESİ
MIDDLE EAST TECHNICAL UNIVERSITY

MÜHENDİSLİK FAKÜLTESİ
FACULTY OF ENGINEERING
ENDÜSTRİ MÜHENDİSLİĞİ BÖLÜMÜ
DEPARTMENT OF INDUSTRIAL ENGINEERING

IE 554 Term Project

2024-2025 Spring

"Academic integrity is expected of all students of METU at all times, whether in the presence or absence of members of the faculty. Understanding this, we declare that we shall not give, use, or receive unauthorized aid in this project."

Student ID Number	***Full Name***	***Signature***
2445013	Ömer Turan Şahinaslan	
2444693	Emek Irmak	

CONTENTS

1 Introduction

Graph partitioning problems arise in various applications, including social network analysis and data clustering. One such problem is the *Dominator Partition Problem* (DPP), which involves partitioning the vertex set of a graph into k disjoint partitions such that every vertex dominates at least one partition.

The goal of this work is twofold: first, to provide a formal integer programming formulation of the DPP, and second, to develop and analyze valid inequalities that improve solver performance. Computational experiments are conducted to evaluate the impact of these inequalities on solution time.

2 Formulation of Dominator Partition Problem

Sets & Parameters

$G = (V, E)$: a graph with vertex set V and edge set E

π: dominator partition of size k of G

Decision Variables

x_{vi}: 1 if vertex v is assigned to the i^{th} block

d_{vi}: 1 if vertex v dominates block i

Objective Function & Constraints

Since k is fixed, we only seek a feasible assignment satisfying all constraints.

$$\min \quad 0 \qquad \text{subject to:}$$

Each vertex must be assigned to exactly one of the k blocks.

$$\sum_{i=1}^{k} x_{vi} = 1, \quad \forall v \in V$$

Every block in the partition must contain at least one vertex; no block is left empty.

$$\sum_{v \in V} x_{vi} \geq 1 \quad \forall i \in \pi$$

If two vertices are not adjacent, then one cannot be in a block while the other dominates it.

$$x_{ui} + d_{vi} \leq 1 \quad \{\forall u, v \in V \mid \{u, v\} \notin E\}, i \in \pi$$

Each vertex must dominate at least one block to satisfy the dominator partition condition.

$$\sum_{i=1}^{k} d_{vi} \geq 1 \quad \forall v \in V$$

To eliminate symmetric solutions, the blocks are required to be used in order.

$$\sum_{v \in V} x_{vi} \geq \sum_{v \in V} x_{v,i+1} \quad \forall i \in \{1, 2, \ldots, k-1\}$$

All decision variables are binary.

$$x_{vi}, d_{vi} \quad \forall v \in V, i \in \pi$$

3 Valid Inequalities

To investigate the integrality properties of the model, we initially relaxed the binary constraints and solved the resulting linear programming formulation. This preliminary analysis was conducted on simple graph instances, beginning with a tree consisting of three nodes, as illustrated in *Figure 1.* The relaxed problem was solved for $k = 2$, and the solution obtained was fractional as shown below.

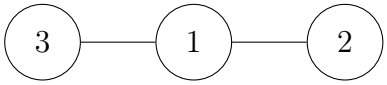

Figure 1: The tree with 3 nodes to be solved

$$\mathbf{x} = \begin{pmatrix} x_{1,1} \\ x_{2,1} \\ x_{3,1} \\ x_{1,2} \\ x_{2,2} \\ x_{3,2} \end{pmatrix} = \begin{pmatrix} 0 \\ 1.0 \\ 0.5 \\ 1.0 \\ 0 \\ 0.5 \end{pmatrix} \qquad \mathbf{d} = \begin{pmatrix} d_{1,1} \\ d_{2,1} \\ d_{3,1} \\ d_{1,2} \\ d_{2,2} \\ d_{3,2} \end{pmatrix} = \begin{pmatrix} 1.0 \\ 0.5 \\ 0 \\ 0 \\ 0.5 \\ 1.0 \end{pmatrix}$$

Subsequently, we aimed to strengthen the formulation by introducing a valid inequality intended to eliminate the observed fractional solution and better approximate the convex hull of the feasible integer region. Through this process, we derived the valid inequality:

$$\sum_{v \in V} x_{v,1} \geq \left\lceil \frac{|V|}{k} \right\rceil \tag{1}$$

where $|V|$ denotes the total number of vertices and k is the number of partitions. The rationale behind this valid inequality is to ensure that the number of vertices assigned to the first partition—which is expected to contain the most due to the ordering assignment constraints—is not below the average number of vertices per partition. To illustrate the effect of this valid inequality, consider the previously obtained fractional solution in which

$$x_{2,1} = 1.0, \qquad x_{3,1} = 0.5, \qquad x_{1,2} = 1.0, \qquad x_{3,2} = 0.5.$$

Here, the sum of assignments to the first partition is

$$\sum_{v \in V} x_{v,1} = 1.0 + 0.5 = 1.5.$$

For the case where $|V| = 3$ and $K = 2$, the right-hand side of the inequality evaluates to

$$\left\lceil \frac{3}{2} \right\rceil = 2.$$

Since $1.5 < 2$, the aforementioned solution violates the newly introduced constraint, and is thus excluded from the feasible region of the relaxed model.

As a result of adding the proposed valid inequalities to the model, it was observed that, for $k = 2$, the linear programming relaxation produced only integral extreme points. This finding indicates that, for this instance, the convex hull of feasible integer solutions was obtained. A

similar result was observed for the same tree structure with $k = 3$, where the relaxation again yielded only integral solutions, suggesting that the formulation describes the convex hull in these cases. To reach this conclusion, random weights between -1 and 1 were assigned to the x and d variables, and the model was resolved multiple times under varying objective functions. In all cases, the optimal solutions remained integral.

Following a similar rationale to the valid inequality (1), we further strengthened the formulation by introducing an upper bound on the number of vertices assigned to each partition. Specifically, for each $i \in \Pi$, the following valid inequality was added:

$$\sum_{v \in V} x_{v,i} \leq \left\lfloor \frac{|V| - k + i}{i} \right\rfloor \quad \forall i \in P \tag{2}$$

Here, $|V|$ denotes the number of vertices, k is the number of partitions, and i is the index of the partition under consideration. This constraint ensures that the number of vertices assigned to the i-th partition does not exceed the calculated upper bound, thereby eliminating infeasible or overly large assignments in the relaxed model.

The analysis was then extended to a tree with four nodes, as illustrated in *Figure 2*. For $k = 2$, the solver returned the following fractional solution:

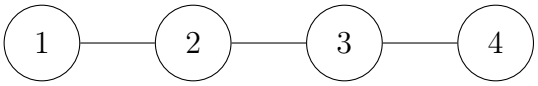

Figure 2: The tree with 4 nodes to be solved

4 Impact of Valid Inequalities to Solver Performance

To assess the computational effectiveness of the valid inequalities introduced in the previous section, we conducted a series of experiments on large graph instances.

We generated random undirected graphs with $n = 200$ nodes using an Erdős–Rényi model, where each edge is independently included with probability $p = 0.2$. We also applied a connectivity post-process to retain only connected graphs.

For the generated graph, the Dominator Partition Problem was solved for $k = 50$ under three different configurations:

(i) No valid inequalities

(ii) With valid inequalities *Equation 1* and *Equation 2* only

(iii) With all valid inequalities

The solution times (in seconds) for each case are reported in Table 1. Additionally, we present the relative percentage improvement compared to the baseline case without valid inequalities.

Table 1: Impact of Valid Inequalities on Solver Performance ($n = 200$, $p = 0.2$, $k = 50$)

Model Variant	**Time (s)**	**Improvement (%)**
No valid inequalities	1047.77	–
Valid inequalities 1 & 2	680.52	35.05%
All valid inequalities	651.63	37.81%

As seen in the results, the addition of valid inequalities significantly enhances solver performance. Incorporating just the first two inequalities reduces runtime by more than one-third. When all proposed inequalities are applied, the model achieves a total runtime reduction of approximately 38%, demonstrating the power of carefully crafted inequalities in accelerating the solution process on large-scale instances.

These results highlight that, beyond theoretical tightness, valid inequalities can substantially reduce computation time in practice—a crucial consideration for solving real-world graph partitioning problems efficiently.

5 Conclusion