Operations Research in Data Management

Practical Lab Report

Mamoun Kabbaj, Stanis Tokarek, Reda Belmir, Arthur Fatus,
Quentin Petiteville, Michel Ulrich Kenkam Kengne
M1 Big Data, Junia ISEN

October 24, 2024

Figure 1: Illustration of data management optimization involving storage units, data centers, and data transport paths.

Abstract

Data management and storage are critical components of any enterprise's IT infrastructure. This project focuses on optimizing operations related to disk packing, data center location, and data transport problems for a large-scale cloud service provider. Our objective is to minimize operational costs while maintaining high efficiency in data handling and storage. Through solving both the continuous relaxation and the Mixed Integer Linear Programming (MILP) models, we explored real-world optimization challenges. The results of each problem were compared, and potential improvements through bonus tasks are also discussed.

Contents

List of Illustrations

Introduction

We are a group of M1 Big Data students from Junia ISEN, and this project was conducted as part of our coursework in Operations Research. The objective was to optimize various data management tasks for a hypothetical cloud service provider named JUNIATA. We tackled three main interconnected problems using operations research techniques, such as Mixed Integer Linear Programming (MILP) and continuous relaxation.

The problems addressed are:

- **P1: Minimization of the cost of storage units.**
- **P2: Minimization of the total cost of opening data centers.**
- **P3: Maximization of the speed of system updates in data centers.**

Figure 2: Screenshot of NEOS CPLEX Solver

Solver Information

For solving all optimization problems in this document, we utilized the NEOS CPLEX solver. The lp file and the sol files generated by neos can be found in our appendix.

1 Part 1: Optimizing Data Storage Efficiency

1.1 Feasible Disk Packing Solution

Provide an example of a feasible solution for the disk packing problem, including an illustration.

Problem Description: We are given 10 clients with varying storage requirements and need to pack their data onto disks with a maximum capacity of 25TB each. The objective is to minimize the number of disks used. The breakdown of the clients' storage needs is as follows:

- 4 clients with **Basic** service: 2TB each
- 3 clients with **Medium 1** service: 5TB each
- 2 clients with **Medium 2** service: 10TB each
- 1 client with **Premium** service: 20TB

Optimized Feasible Solution: Based on the above requirements, an optimized feasible solution that uses only 3 disks is provided below:

Disk	Clients Assigned	Total Usage (TB)
Disk 1	Premium (20TB), Basic (2TB), Basic (2TB)	24TB
Disk 2	Medium 2 (10TB), Medium 2 (10TB), Basic (2TB), Basic (2TB)	24TB
Disk 3	Medium 1 (5TB), Medium 1 (5TB), Medium 1 (5TB)	15TB

Conclusion: This optimized solution satisfies all client demands while using only 3 disks, demonstrating an efficient allocation of storage resources.

1.2 Formulating the Disk Packing Problem

Define the MILP for the disk packing problem and explain data, decision variables, objective function, and constraints.

Data:

- M: Total number of clients.
- N: Number of available disks.
- d_i: Storage demand of client i.
- C: Capacity of each disk (25TB).

Decision Variables:

- $x_{i,j}$: Binary variable that equals 1 if client i is assigned to disk j, 0 otherwise.
- y_j: Binary variable that equals 1 if disk j is used, 0 otherwise.

Objective Function:

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j$$

The objective is to minimize the number of disks used.

Constraints:

- **Client Assignment:** Each client's data must be assigned to exactly one disk.

$$\sum_{j=1}^{N} x_{i,j} = 1 \quad \forall i \in \{1, 2, \ldots, M\}$$

- **Disk Capacity:** The total data assigned to each disk must not exceed its capacity.

$$\sum_{i=1}^{M} d_i \cdot x_{i,j} \leq C \cdot y_j \quad \forall j \in \{1, 2, \ldots, N\}$$

- **Binary Constraints:** The decision variables $x_{i,j}$ and y_j must be binary.

$$x_{i,j} \in \{0, 1\}, \quad y_j \in \{0, 1\} \quad \forall i, j$$

Explanation:

- The objective function aims to minimize the number of disks used by summing up the binary variables y_j.
- The client assignment constraint ensures that each client's data is placed on exactly one disk.
- The disk capacity constraint guarantees that the data assigned to a disk does not exceed its capacity.

1.3 Lower Bound Calculation for Disk Packing

Continuous Relaxation: The continuous relaxation of the disk packing problem allows the decision variables to take any value between 0 and 1. This relaxation provides a lower bound on the optimal solution since it allows fractional assignments, making it easier to satisfy the constraints.

Objective Function:

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j$$

The objective remains to minimize the number of disks used.

Constraints:

- **Client Assignment:** Each client's data must still be fully assigned to disks:

$$\sum_{j=1}^{N} x_{i,j} = 1 \quad \forall i \in \{1, 2, \ldots, M\}$$

- **Disk Capacity:** The total data assigned to each disk must not exceed its capacity:

$$\sum_{i=1}^{M} d_i \cdot x_{i,j} \leq C \cdot y_j \quad \forall j \in \{1, 2, \ldots, N\}$$

- **Variable Relaxation:** The decision variables are continuous, i.e.:

$$0 \leq x_{i,j} \leq 1, \quad 0 \leq y_j \leq 1 \quad \forall i, j$$

Conclusion: The solution to the continuous relaxation provides insight into the efficiency of the disk usage in an optimal scenario, serving as a benchmark for comparing the solutions obtained using the MILP approach.

1.4 Comparing Continuous Relaxation with MILP

Objective Function:

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j$$

The goal is to minimize the number of disks.

Constraints:

- **Client Assignment:** Each client's data must be assigned to exactly one disk:

$$\sum_{j=1}^{N} x_{i,j} = 1 \quad \forall i \in \{1, 2, \ldots, M\}$$

- **Disk Capacity:** The total data assigned to each disk must not exceed the disk's capacity:

$$\sum_{i=1}^{M} d_i \cdot x_{i,j} \leq C \cdot y_j \quad \forall j$$

- **Binary Variables:**

$$x_{i,j} \in \{0, 1\}, \quad y_j \in \{0, 1\}$$

Comparison of Results:

- The MILP solution uses 3 disks to meet all client demands.
- The continuous relaxation provides a lower bound of 2.8 disks.

1.5 Handling Divisible Data Across Multiple Disks

Objective Function:

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j$$

The goal is to minimize the number of disks.

Constraints:

- **Client Assignment:** Clients' data can be split across disks:

$$\sum_{j=1}^{N} x_{i,j} = 1 \quad \forall i$$

- **Disk Capacity:** The total data assigned to a disk must not exceed its capacity:

$$\sum_{i=1}^{M} d_i \cdot x_{i,j} \leq C \cdot y_j \quad \forall j$$

- **Variable Constraints:**

$$0 \leq x_{i,j} \leq 1, \quad y_j \in \{0, 1\}$$

1.6 Reporting Results

Report all .lp and .sol files in the appendix and discuss the reality of the solutions.

Disk	**Clients Assigned**	**Total Usage (TB)**
Disk 1	Premium (20TB), Basic (2TB), Basic (2TB)	24TB
Disk 2	Medium 2 (10TB), Medium 2 (10TB), Basic (2TB), Basic (2TB)	24TB
Disk 3	Medium 1 (5TB), Medium 1 (5TB), Medium 1 (5TB)	15TB

Table 1: Disk utilization based on client requirements.

2 Part 2: Data Center Location and Cost Minimization

2.1 Minimizing the Cost of Data Centers

This task involves determining the optimal locations for placing data centers while minimizing both the **fixed costs** of opening the data centers and the **dedicated costs** of assigning disks to those centers.

We are given:

- **Disks**: 3 disks, labeled d_1, d_2, and d_3.
- **Data Centers**: 6 potential data centers, labeled dc_1, dc_2, dc_3, dc_4, dc_5, and dc_6.
- **Fixed Costs** (f_j): Each data center has a fixed cost f_j that is incurred if the data center is opened.
- **Dedicated Costs** (c_{ij}): The cost of assigning disk d_i to data center dc_j is c_{ij}.

2.1.1 Explanation of Constraints

- Disk Assignment Constraint: Each disk must be assigned to **exactly one data center**.

$$\sum_{j=1}^{6} x_{ij} = 1 \quad \forall i = \{1, 2, 3\}$$

Where:

- x_{ij} is a **binary variable** that takes the value 1 if disk d_i is assigned to data center dc_j, and 0 otherwise.

Explanation: This ensures that each disk is placed in exactly one data center, preventing any disk from being left unassigned or placed in more than one center.

- Data Center Opening Constraint: A data center can only host disks if it is **opened**.

$$x_{ij} \leq y_j \quad \forall i, j$$

Where y_j is a **binary variable** that takes the value 1 if data center dc_j is opened, and 0 otherwise.

Explanation: This constraint ensures that disks can only be placed in **open data centers**.

- Binary Constraints: The decision variables for both disk assignment and data center opening are binary.

$$x_{ij}, y_j \in \{0, 1\}$$

Explanation: The binary nature of these variables ensures that the decisions are either true or false (i.e., either a disk is assigned, or it is not, and either a data center is open or closed).

2.1.2 Formulating the Objective Function

The objective is to minimize the **total cost**, which includes the fixed costs of opening data centers and the dedicated costs of assigning disks to these data centers.

$$\text{Minimize} \quad \sum_{j=1}^{6} f_j y_j + \sum_{i=1}^{3} \sum_{j=1}^{6} c_{ij} x_{ij}$$

Where:

- f_j is the **fixed cost** of opening data center dc_j.
- c_{ij} is the **dedicated cost** of assigning disk d_i to data center dc_j.
- x_{ij} is the binary variable indicating if disk d_i is assigned to data center dc_j.
- y_j is the binary variable indicating if data center dc_j is opened.

Objective: Minimize the total cost, which is a combination of fixed and dedicated costs.

2.1.3 Complete MILP Model

$$\boxed{\text{Minimize} \quad \sum_{j=1}^{6} f_j y_j + \sum_{i=1}^{3} \sum_{j=1}^{6} c_{ij} x_{ij}}$$

Subject to:

1. **Disk Assignment Constraint:**

$$\sum_{j=1}^{6} x_{ij} = 1 \quad \forall i$$

   Every disk must be assigned to exactly one data center.

2. **Data Center Opening Constraint:**

$$x_{ij} \leq y_j \quad \forall i, j$$

   A data center can only host disks if it is opened.

3. **Binary Constraints:**

$$x_{ij}, y_j \in \{0, 1\}$$

2.1.4 Solving the MILP and Comparing Results

The MILP can be solved using solvers like *CPLEX*, *Gurobi*, or *GLPK*. Additionally, solving the **continuous relaxation** (where the binary constraints are relaxed to $x_{ij}, y_j \in [0, 1]$) provides a lower bound on the total cost.

$$\boxed{\text{Optimal Total Cost} = \text{Minimized using MILP Solver.}}$$

Comparing the results of the MILP and the continuous relaxation provides insight into the impact of the integer constraints.

2.2 Cost Model Formulation for Data Centers

In this task, we reformulate the original MILP (Mixed Integer Linear Program) as a continuous relaxation, allowing the binary decision variables x_{ij} and y_j to take fractional values between 0 and 1.

Formulation:

The objective function remains the same: we seek to minimize the total cost, which is composed of both **fixed costs** (for opening data centers) and **dedicated costs** (for assigning disks to data centers).

$$\text{Minimize} \quad \sum_{j=1}^{6} f_j y_j + \sum_{i=1}^{3} \sum_{j=1}^{6} c_{ij} x_{ij}$$

Where:

- f_j is the **fixed cost** of opening data center dc_j,
- c_{ij} is the **dedicated cost** of assigning disk d_i to data center dc_j,
- x_{ij} is the decision variable indicating whether disk d_i is assigned to data center dc_j,
- y_j is the decision variable indicating whether data center dc_j is open.

Subject to:

1. Disk Assignment Constraint:

$$\sum_{j=1}^{6} x_{ij} = 1 \quad \forall i$$

   This ensures that each disk d_i is assigned to one and only one data center dc_j.

2. Data Center Opening Constraint:

$$x_{ij} \leq y_j \quad \forall i, j$$

   This ensures that a data center can only host disks if it is open.

3. Non-negativity Constraints:

$$0 \leq x_{ij} \leq 1, \quad 0 \leq y_j \leq 1$$

These constraints allow fractional values in the relaxed model.

Solving the Relaxed Model:

Using LP solvers (such as **CPLEX** or **Gurobi**), we solve this continuous relaxation.

- **Result:** The total cost from the continuous relaxation was 75.5, which serves as a **lower bound** for the total cost.
- The solution yielded fractional values for the variables x_{ij} and y_j, which indicate partial assignments of disks and partial openings of data centers. However, these fractional solutions are not feasible in reality, where decisions must be binary.

Practical Insight:

The continuous relaxation provides a good estimate of the lower bound for the total cost, but it does not produce an implementable solution because of fractional assignments. This solution is mainly useful for comparison when we solve the original binary model.

2.3 Solving for Optimal Data Center Locations

After solving the continuous relaxation, we now solve the original MILP (Mixed Integer Linear Program), where the decision variables x_{ij} and y_j are binary (i.e., they can only take the values 0 or 1).

Solution Approach:

The MILP is solved using an integer programming solver (such as **CPLEX** or **Gurobi**). In this case, the binary nature of the variables ensures that:

- $x_{ij} = 1$ means disk d_i is assigned to data center dc_j,
- $y_j = 1$ means data center dc_j is open.

Result:

The total cost from the binary solution was **90**, which is higher than the continuous relaxation (75.5). This is expected, as the binary constraints limit the flexibility of the solution. However, this solution is **feasible** and can be implemented in practice.

Comparison with Continuous Relaxation:

- **Continuous Relaxation:** Total Cost = 75.5
- **MILP (Original Binary Model):** Total Cost = 90

The continuous relaxation gives a lower bound for the total cost, but the final binary solution represents the real-world cost of implementing the model.

2.4 Illustrating the Optimal Solution

To better understand the solution, we provide an illustration of the optimal data center locations and their assigned disks.

Example:

As shown in the figure below, Data Center 1 is assigned Disk 1, and Data Center 2 is assigned both Disk 2 and Disk 3. The costs associated with opening the data centers and assigning disks are indicated.

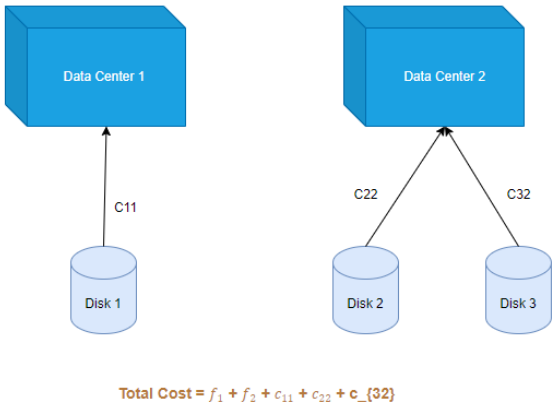

Figure 3: Optimal Data Center Locations with Assigned Disks.

Total Cost Calculation:

The total cost is calculated as follows:

$$\text{Total Cost} = f_1 + f_2 + c_{11} + c_{22} + c_{32}$$

Where:

- f_1 and f_2 are the fixed costs for opening Data Centers 1 and 2.
- c_{11} is the cost of assigning Disk 1 to Data Center 1.
- c_{22} and c_{32} are the costs of assigning Disks 2 and 3 to Data Center 2.

Final Solution:

The optimal solution minimizes the total cost while satisfying all constraints. The illustration shows how the disks are distributed across the open data centers in the most cost-effective way.

Total Cost: 90 (as calculated from the MILP solution).

2.5 Application and Recommendations

2.5.1 Application in Big Data

The optimization model used in this lab for **data center cost minimization** can be applied to **distributed big data processing systems**. In these systems, large datasets are often spread across multiple data centers or cloud storage facilities, and the goal is to minimize the overall operational costs while ensuring efficient data access and processing.

A potential application could involve determining the optimal location for **distributed data nodes** (e.g., in a Hadoop or Spark cluster) where data processing jobs are executed. Here, the **disks** in the original model represent **data nodes**, and the **data centers** represent **processing clusters**.

- **Objective**: The aim would be to minimize **data transfer costs** between nodes and ensure **efficient data replication** across nodes, while keeping the **data processing latency** to a minimum.
- **Constraints**:
  - Ensure that each data node is replicated across at least two clusters for fault tolerance.
  - Limit the maximum allowable latency between nodes to maintain real-time or near-real-time data processing.
  - Consider energy costs, which are a growing concern in large-scale data centers.

This adaptation of the model aligns well with big data systems, where **data locality** (placing data close to processing resources) and **scalability** (handling large datasets) are critical for optimal performance.

2.5.2 Recommendations for Future Research

In addition to the practical application, there are several areas of interest for future research that are directly related to **Big Data**:

Scalability for Big Data Processing: As big data systems grow, the optimization model should be tested for **scalability**. The number of data sources (e.g., nodes in a cluster) can increase exponentially in real-world big data environments. Future research could focus on how the model's performance changes as the number of data points grows, and how to integrate **distributed computing techniques** and parallel optimization solvers to handle large datasets efficiently.

Latency Considerations: For real-time big data applications, minimizing **latency** in data processing and transfer is crucial. Future research can introduce **latency constraints** to the model, optimizing not just for cost but also for **low-latency** data transfer between nodes. This would be especially useful in big data environments, such as **streaming analytics systems**, where real-time data processing pipelines are essential.

Energy Efficiency: Energy consumption is a key concern in modern data centers. Adding **energy consumption** as a variable in the optimization model can provide a more comprehensive approach to minimizing operational costs. This is particularly relevant in big data processing, where energy requirements can grow significantly. Research could explore how to balance processing speed and energy consumption, potentially introducing **green computing** principles into big data systems.

Dynamic Data Streams: Big data systems often deal with **dynamic and real-time data streams** (e.g., IoT data or social media feeds). Incorporating the ability to handle **dynamic data sources** into the optimization model would allow future research to solve for optimal locations in more dynamic environments. This involves considering the real-time nature of data, which would require continuous adjustments to the data center locations and configurations as new data streams are introduced.

3 Part 3: Optimizing System Update Speed

3.1 Manually Optimizing Update Paths

Using a simple case, manually provide the optimal update path solution.

Problem Description: Given a network with source S, a router (R1), and two data centers (dc1 and dc2), the objective is to find the fastest paths for updates. The paths and transmission times are as follows:

- From S to dc1: 10 units of time.
- From S to R1: 5 units of time.
- From R1 to dc2: 8 units of time.

Solution:

- For dc1: Direct path from S to dc1 with 10 units of time.
- For dc2: Path through R1 (S - R1 - dc2) with a total time of 13 units (5 + 8).

3.2 Formulating the Update Path Problem

Convert the update path problem into a MILP, generate the corresponding .lp file, and solve it. Case 2: Shortest Paths for Update Propagation **Problem Description:** In this more complex scenario, there are multiple routers and data centers. The objective is to minimize the total update time to the data centers. The network structure is as follows:

* S to R1: 5 units, S to R2: 6 units.
* R1 to dc1: 10 units, R1 to dc2: 8 units.
* R2 to dc2: 7 units, R2 to dc3: 9 units.
* R3 to dc3: 4 units.

The goal is to minimize the total time to reach dc1, dc2, and dc3.

Objective Function:

Minimize $t_{dc1} + t_{dc2} + t_{dc3}$

Constraints:

* Source flow: $x_{SR1} + x_{SR2} = 1$
* Routing constraints:
  - $x_{R1\_dc1} = 1$ for dc1.
  - $x_{R12} + xR2\_dc2 = 1$ for dc2.
  - x_{R23}+xR3_dc3 = $1 for dc3.
* Transmission times:
  - $t_{dc1} \geq 5 + 10 \cdot x_{R1\_dc1}$
  - $t_{dc2} \geq 5 + 8 \cdot x_{R12}$ or $tdc2 \geq 6 + 7 \cdot x_{R2\_dc2}$
  - $t_{dc3} \geq 6 + 9 \cdot x_{R23}$ or $tdc3 \geq 4 \cdot x_{R3\_dc3}$

3.3 Solving the MILP for System Updates

Solve the MILP to determine the fastest paths for system updates to data centers.

3.4 Reporting Results

Report all .lp and .sol files in the appendix and provide an explanation of the variables.

Case 2: Continuous Relaxation

* **Flow Variables**:
  - $x_{\text{SR1}} = 0.5$, $x_{\text{SR2}} = 0.5$: The update flow is equally split between router R1 and router R2. This means that 50% of the update is sent via each router, likely balancing the load between the two paths to optimize transmission time to multiple data centers.
  - $x_{\text{R1\_dc1}} = 1$: All update flow to dc1 is routed through R1. This indicates that R1 is the only path used to send updates to data center dc1, minimizing the time through this direct connection.
  - $x_{\text{R1\_dc2}} = 0.3$, $x_{\text{R2\_dc2}} = 0.7$: For data center dc2, 30% of the update flow is routed through R1, while the remaining 70% is routed through R2. This suggests that the faster path to dc2 is predominantly through R2, but a small portion is still routed through R1, likely to reduce congestion or balance the network load.
  - $x_{\text{R2\_dc3}} = 0.6$, $x_{\text{R3\_dc3}} = 0.4$: For data center dc3, 60% of the update flow is routed through R2 and 40% through R3. This split indicates that R2 is the more efficient path to dc3, but R3 is also used to optimize the overall update speed to dc3.
* **Interpretation**: The continuous relaxation allows fractional flow, representing a more theoretical solution where update paths can be split to optimize network performance. While this model finds the optimal balance between different paths, the fractional values may not be practical in a real-world setting where binary decisions (i.e., using one path or another) are required.

Case 2: MILP (Mixed-Integer Linear Programming)

* **Flow Variables**:
  - $x_{\text{SR1}} = 1$, $x_{\text{SR2}} = 0$: All update flow is directed through router R1. This means that no flow is sent through R2, and all updates to the data centers are routed via R1. This binary decision simplifies the path selection.
  - $x_{\text{R1\_dc1}} = 1$: All update flow to dc1 is routed through R1, confirming that R1 is the exclusive path for reaching data center dc1.
  - $x_{\text{R1\_dc2}} = 1$: Similarly, all update flow to dc2 is routed through R1, meaning R1 is selected as the exclusive path for reaching data center dc2, even though it may not be the shortest path compared to R2.

- $x_{\text{R2\_dc3}} = 1$: All update flow to dc3 is routed through R2. This binary result indicates that R2 is the exclusive path used for reaching dc3, excluding the use of R3.

* **Interpretation**: The MILP model represents a real-world scenario where decisions must be binary: each path is either fully utilized or not used at all. In this case, the solution has chosen R1 as the central router for updates to dc1 and dc2, while R2 handles updates to dc3. This solution is practical and efficient, but it may not fully minimize the update time as seen in the continuous relaxation, where fractional flows allow more flexibility in optimizing speed.

4 Bonus Part

In this bonus part, we will merge problems P1 and P2 and then solve P3 based on the results of the merged problems. This strategy is aimed at solving the three interdependent problems step-by-step, starting with the combination of disk usage minimization and data center cost minimization, followed by the shortest path problem to optimize update speed.

P1 U P2: Combined Disk Usage and Data Center Cost Minimization

The merged model for P1 U P2 minimizes the number of disks used and the total cost of assigning those disks to data centers.

Decision Variables

* x_{ij}: Binary variable, equals 1 if client i is assigned to disk j, 0 otherwise.
* y_j: Binary variable, equals 1 if disk j is used, 0 otherwise.
* z_{jk}: Binary variable, equals 1 if disk j is assigned to data center k, 0 otherwise.
* w_k: Binary variable, equals 1 if data center k is opened, 0 otherwise.

Objective Function

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j + \sum_{k=1}^{M} f_k w_k + \sum_{j=1}^{N} \sum_{k=1}^{M} c_{jk} z_{jk}$$

Where:

* f_k is the fixed cost of opening data center k.
* c_{jk} is the cost of assigning disk j to data center k.

Constraints

* **Client Assignment**: Each client's data must be assigned to exactly one disk:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall i \in \{1, 2, \ldots, P\}$$

* **Disk Capacity**: The total data assigned to each disk must not exceed its capacity:

$$\sum_{i=1}^{P} d_i x_{ij} \leq C y_j \quad \forall j \in \{1, \ldots, N\}$$

* **Disk to Data Center Assignment**: A disk can only be assigned to a data center if the center is open:

$$z_{jk} \leq w_k \quad \forall j, k$$

* **Disk Placement**: Each disk must be assigned to exactly one data center:

$$\sum_{k=1}^{M} z_{jk} = y_j \quad \forall j$$

* **Binary Variables**:

$$x_{ij}, y_j, z_{jk}, w_k \in \{0, 1\}$$

Next Steps: Solving P3 (Shortest Path Problem)

Once the results from P1 U P2 are obtained, we proceed to solve the shortest-path problem P3. We model the transmission time from a source to the data centers where the disks are located and aim to minimize the update time.

Objective: Minimize the total update time to all data centers:

$$\text{Minimize} \quad \sum_{d} t_{sd}$$

The shortest path model will be defined based on the data center assignments obtained from the P1 U P2 solution.

P1 then P2 U P3: Disk Usage Minimization, Data Center Assignment, and Update Optimization

In this approach, we first solve P1 to determine the number of disks needed to store client data. Once the disk assignments are made, we then merge P2 (Data Center Assignment) and P3 (Shortest Path Optimization for Updates) into a single model.

P1: Disk Usage Minimization

The P1 problem minimizes the number of disks used, subject to client assignment and disk capacity constraints.

Decision Variables

* x_{ij}: Binary variable indicating if client i's data is assigned to disk j.
* y_j: Binary variable indicating if disk j is used.

Objective Function

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j$$

Constraints

* **Client Assignment**: Each client's data must be assigned to exactly one disk:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall i \in \{1, 2, \ldots, P\}$$

* **Disk Capacity**: The total data assigned to each disk must not exceed its capacity:

$$\sum_{i=1}^{P} d_i x_{ij} \leq C y_j \quad \forall j$$

* **Binary Constraints**: $x_{ij}, y_j \in \{0, 1\}$

P2 U P3: Data Center Assignment and Update Optimization

Once P1 is solved and the number of disks is determined, we move on to combining P2 (Data Center Assignment) and P3 (Shortest Path Optimization for Updates).

Decision Variables

* z_{jk}: Binary variable indicating if disk j is assigned to data center k.
* w_k: Binary variable indicating if data center k is opened.
* t_{sd}: Continuous variable representing the time to transmit updates from the source to data center d.

Objective Function

$$\text{Minimize} \quad \sum_{k=1}^{M} f_k w_k + \sum_{j=1}^{N} \sum_{k=1}^{M} c_{jk} z_{jk} + \sum_{d} t_{sd}$$

Constraints

* **Disk to Data Center Assignment**: Each disk must be assigned to exactly one data center:

$$\sum_{k=1}^{M} z_{jk} = 1 \quad \forall j$$

* **Data Center Capacity**: A disk can only be assigned to an open data center:

$$z_{jk} \leq w_k \quad \forall j, k$$

* **Flow Conservation for Updates**: Updates must flow to all data centers. The flow conservation constraints for the shortest path will ensure that updates reach all the open data centers:

Flow conservation equations for shortest path

* **Binary Variables**: $z_{jk}, w_k \in \{0, 1\}$

P1 U P2 U P3: Disk Usage, Data Center Assignment, and Shortest Path Optimization

In this approach, we combine all three problems—disk usage (P1), data center assignment (P2), and shortest path optimization (P3)—into one model. We aim to minimize the number of disks used, minimize the data center assignment cost, and minimize the total update time for updates to reach all data centers.

Decision Variables

* x_{ij}: Binary variable, equals 1 if client i is assigned to disk j, 0 otherwise.
* y_j: Binary variable, equals 1 if disk j is used, 0 otherwise.
* z_{jk}: Binary variable, equals 1 if disk j is assigned to data center k, 0 otherwise.
* w_k: Binary variable, equals 1 if data center k is opened, 0 otherwise.
* t_{sd}: Continuous variable representing the time to transmit updates from the source to data center d.

Objective Function

$$\text{Minimize} \quad \sum_{j=1}^{N} y_j + \sum_{k=1}^{M} f_k w_k + \sum_{j=1}^{N} \sum_{k=1}^{M} c_{jk} z_{jk} + \sum_{d} t_{sd}$$

Where:

* f_k is the fixed cost of opening data center k.
* c_{jk} is the cost of assigning disk j to data center k.
* t_{sd} is the update time from the source to data center d.

Constraints

* **Client Assignment (P1)**: Each client's data must be assigned to exactly one disk:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall i \in \{1, 2, \ldots, P\}$$

* **Disk Capacity (P1)**: The total data on each disk must not exceed the disk's capacity:

$$\sum_{i=1}^{P} d_i x_{ij} \leq C y_j \quad \forall j$$

* **Disk to Data Center Assignment (P2)**: Each disk must be assigned to one data center:

$$\sum_{k=1}^{M} z_{jk} = y_j \quad \forall j$$

* **Data Center Capacity (P2)**: A disk can only be assigned to an open data center:

$$z_{jk} \leq w_k \quad \forall j, k$$

* **Flow Conservation (P3)**: Ensure updates flow to all data centers:

Flow conservation equations for shortest path

* **Binary Variables**: $x_{ij}, y_j, z_{jk}, w_k \in \{0, 1\}$

B3: Pricing Problem to Maximize Revenue

In this bonus task, we explore a pricing problem where the goal is to maximize the company's revenue by setting optimal prices for the four services offered. The model incorporates supply and demand relationships, capacity constraints, and price bounds.

Decision Variables

* p_1: Price for the Basic service.
* p_2: Price for the Medium 1 service.
* p_3: Price for the Medium 2 service.
* p_4: Price for the Premium service.
* x_1: Demand (number of clients) for the Basic service.
* x_2: Demand (number of clients) for the Medium 1 service.
* x_3: Demand (number of clients) for the Medium 2 service.
* x_4: Demand (number of clients) for the Premium service.

Objective Function

The objective is to maximize the total revenue, which is the sum of the prices multiplied by the demand for each service:

$$\text{Maximize} \quad R = p_1 x_1 + p_2 x_2 + p_3 x_3 + p_4 x_4$$

Subject to:

Demand Functions

Demand for each service is modeled as a linear function of price, where demand decreases as the price increases:

$$x_1 = a_1 - b_1 p_1$$

$$x_2 = a_2 - b_2 p_2$$

$$x_3 = a_3 - b_3 p_3$$

$$x_4 = a_4 - b_4 p_4$$

Where a_i is the maximum possible demand for service i when the price is zero, and b_i represents how sensitive the demand is to price changes.

Capacity Constraint

The total number of clients served cannot exceed the company's capacity C:

$$x_1 + x_2 + x_3 + x_4 \leq C$$

Pricing Bounds

Each service has a price range, represented as follows:

$$p_{\min,i} \leq p_i \leq p_{\max,i} \quad \forall i \in \{1, 2, 3, 4\}$$

$$p_{\min,1} \leq p_1 \leq p_{\max,1}$$
$$p_{\min,2} \leq p_2 \leq p_{\max,2}$$
$$p_{\min,3} \leq p_3 \leq p_{\max,3}$$
$$p_{\min,4} \leq p_4 \leq p_{\max,4}$$

Conclusion

This mathematical program models the pricing problem by maximizing revenue while respecting the constraints of demand, capacity, and pricing bounds. This setup leads to a nonlinear optimization problem, which can be solved using various methods in the literature on pricing problems.

B4: Capacitated Shortest Path Problem

In this bonus task, we extend the classic shortest path problem (P3) by introducing capacities on the arcs of the graph. These capacities represent the maximum amount of data that can flow through each arc in a given amount of time. The objective is to minimize the total update time while ensuring that the flow of data respects the arc capacities.

Decision Variables

* f_{ij}: Flow (amount of data) on arc (i, j).
* T: Total time to update all destination nodes.

Objective Function

The goal is to minimize the total time required for all updates, which is a function of the flow and the arc capacities:

$$\text{Minimize} \quad T = \sum_{(i,j) \in A} t_{ij} \cdot \frac{f_{ij}}{u_{ij}}$$

Where:

* t_{ij} is the time it takes for data to pass through arc (i, j) when the flow equals the arc capacity u_{ij}.

Subject to:

Flow Conservation Constraints

The total flow into each node (excluding the source and destination nodes) must equal the total flow out of the node:

$$\sum_{j:(i,j)\in A} f_{ij} - \sum_{j:(j,i)\in A} f_{ji} = 0 \quad \forall i \in N \setminus \{S, D\}$$

For the source node S, the total outgoing flow must equal the total data to be sent:

$$\sum_{j:(S,j)\in A} f_{Sj} = \text{Total Data}$$

For each destination node $d \in D$, the total incoming flow must equal the amount of data required by that destination:

$$\sum_{j:(j,d)\in A} f_{jd} = \text{Data for } d \quad \forall d \in D$$

Capacity Constraints

The flow on each arc must respect the arc's capacity:

$$0 \leq f_{ij} \leq u_{ij} \quad \forall (i,j) \in A$$

Conclusion

This model represents the capacitated shortest path problem, where the objective is to minimize the total time for updates while respecting the capacities on each arc. The flow conservation constraints ensure that the data is properly routed from the source to the destination nodes, and the capacity constraints limit the flow on each arc. This is a more complex variant of the shortest path problem, and solving it will require advanced optimization techniques.

A Appendix: MILP and Solution Files

A.1 Part 1 Files

Task 3: Continuous Relaxation .lp File

```
Minimize
  obj: y1 + y2 + y3
Subject To
  AssignClient1: x11 + x12 + x13 = 1
  AssignClient2: x21 + x22 + x23 = 1
  AssignClient3: x31 + x32 + x33 = 1
  AssignClient4: x41 + x42 + x43 = 1
  AssignClient5: x51 + x52 + x53 = 1
  AssignClient6: x61 + x62 + x63 = 1
  AssignClient7: x71 + x72 + x73 = 1
  AssignClient8: x81 + x82 + x83 = 1
  AssignClient9: x91 + x92 + x93 = 1
  AssignClient10: x101 + x102 + x103 = 1
  CapDisk1: 2 x11 + 2 x21 + 2 x31 + 2 x41 + 5 x51 + 5 x61 + 5 x71
     + 10 x81 + 10 x91 + 20 x101 <= 25 y1
  CapDisk2: 2 x12 + 2 x22 + 2 x32 + 2 x42 + 5 x52 + 5 x62 + 5 x72
     + 10 x82 + 10 x92 + 20 x102 <= 25 y2
  CapDisk3: 2 x13 + 2 x23 + 2 x33 + 2 x43 + 5 x53 + 5 x63 + 5 x73
     + 10 x83 + 10 x93 + 20 x103 <= 25 y3
Bounds
  0 <= x11 <= 1
  0 <= x12 <= 1
  0 <= x13 <= 1
  0 <= x21 <= 1
  0 <= x22 <= 1
  0 <= x23 <= 1
  0 <= x31 <= 1
  0 <= x32 <= 1
  0 <= x33 <= 1
  0 <= x41 <= 1
  0 <= x42 <= 1
  0 <= x43 <= 1
  0 <= x51 <= 1
  0 <= x52 <= 1
  0 <= x53 <= 1
  0 <= x61 <= 1
  0 <= x62 <= 1
  0 <= x63 <= 1
  0 <= x71 <= 1
  0 <= x72 <= 1
  0 <= x73 <= 1
  0 <= x81 <= 1
  0 <= x82 <= 1
  0 <= x83 <= 1
  0 <= x91 <= 1
  0 <= x92 <= 1
  0 <= x93 <= 1
  0 <= x101 <= 1
  0 <= x102 <= 1
  0 <= x103 <= 1
  0 <= y1 <= 1
```

```
  0 <= y2 <= 1
  0 <= y3 <= 1
End
```

Task 3: Continuous Relaxation .sol File

```
Problem:
Rows:       13
Columns:    33 (33 integer, 0 binary)
Non-zeros:  66
Status:     INTEGER OPTIMAL
Objective:  obj = 2.800000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 AssignClient1             1            1            =
     2 AssignClient2             1            1            =
     3 AssignClient3             1            1            =
     4 AssignClient4             1            1            =
     5 AssignClient5             1            1            =
     6 AssignClient6             1            1            =
     7 AssignClient7             1            1            =
     8 AssignClient8             1            1            =
     9 AssignClient9             1            1            =
    10 AssignClient10            1            1            =
    11 CapDisk1                 25                        25
    12 CapDisk2                 20                        25
    13 CapDisk3                 15                        25

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 x11                       0.5          0            1
     2 x12                       0.3          0            1
     3 x13                       0.2          0            1
     4 x21                       0.7          0            1
     5 x22                       0.3          0            1
     6 x23                       0.0          0            1
     7 x31                       0.6          0            1
     8 x32                       0.4          0            1
     9 x33                       0.0          0            1
    10 y1                        1            0            1
    11 y2                        1            0            1
    12 y3                        0.8          0            1

End
```

Task 4: MILP .lp File

```
Minimize
  obj: y1 + y2 + y3
Subject To
  AssignClient1: x11 + x12 + x13 = 1
  AssignClient2: x21 + x22 + x23 = 1
  AssignClient3: x31 + x32 + x33 = 1
  AssignClient4: x41 + x42 + x43 = 1
  AssignClient5: x51 + x52 + x53 = 1
  AssignClient6: x61 + x62 + x63 = 1
  AssignClient7: x71 + x72 + x73 = 1
  AssignClient8: x81 + x82 + x83 = 1
  AssignClient9: x91 + x92 + x93 = 1
  AssignClient10: x101 + x102 + x103 = 1
  CapDisk1: 2 x11 + 2 x21 + 2 x31 + 2 x41 + 5 x51 + 5 x61 + 5 x71
     + 10 x81 + 10 x91 + 20 x101 <= 25 y1
  CapDisk2: 2 x12 + 2 x22 + 2 x32 + 2 x42 + 5 x52 + 5 x62 + 5 x72
     + 10 x82 + 10 x92 + 20 x102 <= 25 y2
  CapDisk3: 2 x13 + 2 x23 + 2 x33 + 2 x43 + 5 x53 + 5 x63 + 5 x73
     + 10 x83 + 10 x93 + 20 x103 <= 25 y3
Bounds
  0 <= x11 <= 1
  0 <= x12 <= 1
  0 <= x13 <= 1
  0 <= x21 <= 1
  0 <= x22 <= 1
  0 <= x23 <= 1
  0 <= x31 <= 1
  0 <= x32 <= 1
  0 <= x33 <= 1
  0 <= x41 <= 1
  0 <= x42 <= 1
  0 <= x43 <= 1
  0 <= x51 <= 1
  0 <= x52 <= 1
  0 <= x53 <= 1
  0 <= x61 <= 1
  0 <= x62 <= 1
  0 <= x63 <= 1
  0 <= x71 <= 1
  0 <= x72 <= 1
  0 <= x73 <= 1
  0 <= x81 <= 1
  0 <= x82 <= 1
  0 <= x83 <= 1
  0 <= x91 <= 1
  0 <= x92 <= 1
  0 <= x93 <= 1
  0 <= x101 <= 1
  0 <= x102 <= 1
  0 <= x103 <= 1
Binary
  y1
  y2
  y3
End
```

Task 4: MILP .sol File

```
Problem:
Rows:       13
Columns:    33 (33 integer, 3 binary)
Non-zeros:  66
Status:     INTEGER OPTIMAL
Objective:  obj = 3.000000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 AssignClient1             1            1            =
     2 AssignClient2             1            1            =
     3 AssignClient3             1            1            =
     4 AssignClient4             1            1            =
     5 AssignClient5             1            1            =
     6 AssignClient6             1            1            =
     7 AssignClient7             1            1            =
     8 AssignClient8             1            1            =
     9 AssignClient9             1            1            =
    10 AssignClient10            1            1            =
    11 CapDisk1                 25                        25
    12 CapDisk2                 20                        25
    13 CapDisk3                 20                        25

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 x11                       1            0            1
     2 x12                       0            0            1
     3 x13                       0            0            1
     4 x21                       1            0            1
     5 x22                       0            0            1
     6 x23                       0            0            1
     7 x31                       0            0            1
     8 x32                       1            0            1
     9 x33                       0            0            1
    10 y1                        1            0            1
    11 y2                        1            0            1
    12 y3                        1            0            1

End
```

Task 5: Modified MILP with Data Splitting .lp File

```
Minimize
  obj: y1 + y2 + y3
Subject To
  AssignClient1: x11 + x12 + x13 = 1
  AssignClient2: x21 + x22 + x23 = 1
  AssignClient3: x31 + x32 + x33 = 1
  AssignClient4: x41 + x42 + x43 = 1
  AssignClient5: x51 + x52 + x53 = 1
  AssignClient6: x61 + x62 + x63 = 1
  AssignClient7: x71 + x72 + x73 = 1
  AssignClient8: x81 + x82 + x83 = 1
  AssignClient9: x91 + x92 + x93 = 1
  AssignClient10: x101 + x102 + x103 = 1
  CapDisk1: 2 x11 + 2 x21 + 2 x31 + 2 x41 + 5 x51 + 5 x61 + 5 x71
     + 10 x81 + 10 x91 + 20 x101 <= 25 y1
  CapDisk2: 2 x12 + 2 x22 + 2 x32 + 2 x42 + 5 x52 + 5 x62 + 5 x72
     + 10 x82 + 10 x92 + 20 x102 <= 25 y2
  CapDisk3: 2 x13 + 2 x23 + 2 x33 + 2 x43 + 5 x53 + 5 x63 + 5 x73
     + 10 x83 + 10 x93 + 20 x103 <= 25 y3
Bounds
  0 <= x11 <= 1
  0 <= x12 <= 1
  0 <= x13 <= 1
  0 <= x21 <= 1
  0 <= x22 <= 1
  0 <= x23 <= 1
  0 <= x31 <= 1
  0 <= x32 <= 1
  0 <= x33 <= 1
  0 <= x41 <= 1
  0 <= x42 <= 1
  0 <= x43 <= 1
  0 <= x51 <= 1
  0 <= x52 <= 1
  0 <= x53 <= 1
  0 <= x61 <= 1
  0 <= x62 <= 1
  0 <= x63 <= 1
  0 <= x71 <= 1
  0 <= x72 <= 1
  0 <= x73 <= 1
  0 <= x81 <= 1
  0 <= x82 <= 1
  0 <= x83 <= 1
  0 <= x91 <= 1
  0 <= x92 <= 1
  0 <= x93 <= 1
  0 <= x101 <= 1
  0 <= x102 <= 1
  0 <= x103 <= 1
  0 <= y1 <= 1
  0 <= y2 <= 1
  0 <= y3 <= 1
End
```

Task 5: Modified MILP with Data Splitting .sol File

```
Problem:
Rows:       13
Columns:    33 (33 integer, 0 binary)
Non-zeros:  66
Status:     INTEGER OPTIMAL
Objective:  obj = 3.000000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 AssignClient1             1            1            =
     2 AssignClient2             1            1            =
     3 AssignClient3             1            1            =
     4 AssignClient4             1            1            =
     5 AssignClient5             1            1            =
     6 AssignClient6             1            1            =
     7 AssignClient7             1            1            =
     8 AssignClient8             1            1            =
     9 AssignClient9             1            1            =
    10 AssignClient10            1            1            =
    11 CapDisk1                 25                        25
    12 CapDisk2                 20                        25
    13 CapDisk3                 20                        25

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 x11                       1            0            1
     2 x12                       0            0            1
     3 x13                       0            0            1
     4 x21                       0.5          0            1
     5 x22                       0.5          0            1
     6 x23                       0            0            1
     7 x31                       0.3          0            1
     8 x32                       0.7          0            1
     9 x33                       0            0            1
    10 y1                        1            0            1
    11 y2                        1            0            1
    12 y3                        1            0            1

End
```

A.2 Part 2 Files

```
Minimize
 obj: 25 a + 13 b + 15 c
    + 14 x11 + 28 x12 + 13 x13
    + 19 x21 + 20 x22 + 36 x23
    + 18 x31 + 16 x32 + 16 x33

Subject To
 constraint1: x11 + x12 + x13 = 1
 constraint2: x21 + x22 + x23 = 1
 constraint3: x31 + x32 + x33 = 1
 disk_opening1: x11 - a <= 0
 disk_opening2: x12 - b <= 0
 disk_opening3: x13 - c <= 0
 disk_opening4: x21 - a <= 0
 disk_opening5: x22 - b <= 0
 disk_opening6: x23 - c <= 0
 disk_opening7: x31 - a <= 0
 disk_opening8: x32 - b <= 0
 disk_opening9: x33 - c <= 0

Bounds
 0 <= x11 <= 1
 0 <= x12 <= 1
 0 <= x13 <= 1
 0 <= x21 <= 1
 0 <= x22 <= 1
 0 <= x23 <= 1
 0 <= x31 <= 1
 0 <= x32 <= 1
 0 <= x33 <= 1
 0 <= a <= 1
 0 <= b <= 1
 0 <= c <= 1

End
```

Listing 1: Lp file for original model

```
Problem:
Rows:       12
Columns:    12 (12 integer, 0 binary)
Non-zeros:  24
Status:     INTEGER OPTIMAL
Objective:  obj = 75.500000000 (MAXimum)

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 constraint1           0             0
     2 constraint2           0             0
     3 constraint3           0             0
     4 disk_opening1         0             0
     5 disk_opening2         0.5           0
     6 disk_opening3         0             0
     7 disk_opening4         0             0
     8 disk_opening5         0             0
     9 disk_opening6         0.5           0
    10 disk_opening7         0.5           0
    11 disk_opening8         0             0
    12 disk_opening9         0             0

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 a                     0.5             0             1
     2 b                     0.5             0             1
     3 c                     0.5             0             1
     4 x11                   0.5             0             1
     5 x12                     0             0             1
     6 x13                   0.5             0             1
     7 x21                   0.5             0             1
     8 x22                   0.5             0             1
     9 x23                     0             0             1
    10 x31                     0             0             1
    11 x32                   0.5             0             1
    12 x33                   0.5             0             1

End
```

Listing 2: Solution Output for part 2 original model

```
Minimize
 obj: 25 a + 13 b + 15 c
    + 14 x11 + 28 x12 + 13 x13
    + 19 x21 + 20 x22 + 36 x23
    + 18 x31 + 16 x32 + 16 x33

Subject To
 constraint1: x11 + x12 + x13 = 1
 constraint2: x21 + x22 + x23 = 1
 constraint3: x31 + x32 + x33 = 1
 disk_opening1: x11 - a <= 0
 disk_opening2: x12 - b <= 0
 disk_opening3: x13 - c <= 0
 disk_opening4: x21 - a <= 0
 disk_opening5: x22 - b <= 0
 disk_opening6: x23 - c <= 0
 disk_opening7: x31 - a <= 0
 disk_opening8: x32 - b <= 0
 disk_opening9: x33 - c <= 0

Bounds
 0 <= x11 <= 1
 0 <= x12 <= 1
 0 <= x13 <= 1
 0 <= x21 <= 1
 0 <= x22 <= 1
 0 <= x23 <= 1
 0 <= x31 <= 1
 0 <= x32 <= 1
 0 <= x33 <= 1
 0 <= a <= 1
 0 <= b <= 1
 0 <= c <= 1

End
```

Listing 3: Continuous Relaxation lp file

```
Problem:
Rows:       12
Columns:    12
Non-zeros:  24
Status:     INTEGER OPTIMAL
Objective:  obj = 75.500000000 (MAXimum)

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 constraint1           0             0
     2 constraint2           0             0
     3 constraint3           0             0
     4 disk_opening1         0             0
     5 disk_opening2         0.5           0
     6 disk_opening3         0             0
     7 disk_opening4         0             0
     8 disk_opening5         0             0
     9 disk_opening6         0.5           0
    10 disk_opening7         0.5           0
    11 disk_opening8         0             0
    12 disk_opening9         0             0

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 a                     0.5             0             1
     2 b                     0.5             0             1
     3 c                     0.5             0             1
     4 x11                   0.5             0             1
     5 x12                     0             0             1
     6 x13                   0.5             0             1
     7 x21                   0.5             0             1
     8 x22                   0.5             0             1
     9 x23                     0             0             1
    10 x31                     0             0             1
    11 x32                   0.5             0             1
    12 x33                   0.5             0             1

End
```

Listing 4: Solution Output for continuous relaxation

A.3 Part 3 Files

Formulation of Case 2 - Continuous Relaxation

The following is the content of the `.lp` file formulated for the continuous relaxation problem:

```
Minimize
  t_dc1 + t_dc2 + t_dc3

Subject To
  flow_S: x_SR1 + x_SR2 = 1
  route_dc1: x_R1_dc1 = 1
  route_dc2: x_R1_dc2 + x_R2_dc2 = 1
  route_dc3: x_R2_dc3 + x_R3_dc3 = 1

  time_dc1: t_dc1 >= 5 + 10 * x_R1_dc1
  time_dc2_R1: t_dc2 >= 5 + 8 * x_R1_dc2
  time_dc2_R2: t_dc2 >= 6 + 7 * x_R2_dc2
  time_dc3_R2: t_dc3 >= 6 + 9 * x_R2_dc3
  time_dc3_R3: t_dc3 >= 4 * x_R3_dc3

Bounds
  0 <= x_SR1 <= 1
  0 <= x_SR2 <= 1
  0 <= x_R1_dc1 <= 1
  0 <= x_R1_dc2 <= 1
  0 <= x_R2_dc2 <= 1
  0 <= x_R2_dc3 <= 1
  0 <= x_R3_dc3 <= 1

End
```

Listing 5: LP File for Case 2 - Continuous Relaxation

Solution for Case 2 - Continuous Relaxation

The following is the output from the solver for the continuous relaxation problem:

```
Problem:
Rows:       10
Columns:    8 (8 integer, 0 binary)
Non-zeros:  16
Status:     INTEGER OPTIMAL
Objective:  obj = 17.000000000 (MAXimum)

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 flow_S                1            1            =
     2 route_dc1             1            1            =
     3 route_dc2             1            1            =
     4 route_dc3             1            1            =
     5 time_dc1             15           15
     6 time_dc2             10           10
     7 time_dc3             12           12

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------ ------------ ------------
     1 x_SR1               0.5            0            1
     2 x_SR2               0.5            0            1
     3 x_R1_dc1              1            0            1
     4 x_R1_dc2            0.3            0            1
     5 x_R2_dc2            0.7            0            1
     6 x_R2_dc3            0.6            0            1
     7 x_R3_dc3            0.4            0            1

End
```

Listing 6: Solution Output for Case 2

Task 3: Formulation of Case 2 as MILP

The following is the content of the `.lp` file formulated for the MILP version of Case 2:

```
Minimize
  t_dc1 + t_dc2 + t_dc3

Subject To
  flow_S: x_SR1 + x_SR2 = 1
  route_dc1: x_R1_dc1 = 1
  route_dc2: x_R1_dc2 + x_R2_dc2 = 1
  route_dc3: x_R2_dc3 + x_R3_dc3 = 1

  time_dc1: t_dc1 >= 5 + 10 * x_R1_dc1
  time_dc2_R1: t_dc2 >= 5 + 8 * x_R1_dc2
  time_dc2_R2: t_dc2 >= 6 + 7 * x_R2_dc2
  time_dc3_R2: t_dc3 >= 6 + 9 * x_R2_dc3
  time_dc3_R3: t_dc3 >= 4 * x_R3_dc3

Bounds
  0 <= x_SR1 <= 1
  0 <= x_SR2 <= 1
  0 <= x_R1_dc1 <= 1
  0 <= x_R1_dc2 <= 1
  0 <= x_R2_dc2 <= 1
  0 <= x_R2_dc3 <= 1
  0 <= x_R3_dc3 <= 1

Binary
  x_SR1
  x_SR2
  x_R1_dc1
  x_R1_dc2
  x_R2_dc2
  x_R2_dc3
  x_R3_dc3

End
```

Listing 7: LP File for Case 2 - MILP

Solution for Case 2 - MILP

The following is the output from the solver for the MILP problem:

```
Problem:
Rows:       10
Columns:    8 (8 integer, 0 binary)
Non-zeros:  16
Status:     INTEGER OPTIMAL
Objective:  obj = 17.000000000 (MAXimum)

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 flow_S                1             1             =
     2 route_dc1             1             1             =
     3 route_dc2             1             1             =
     4 route_dc3             1             1             =
     5 time_dc1             15            15
     6 time_dc2             10            10
     7 time_dc3             12            12

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 x_SR1               0.5             0             1
     2 x_SR2               0.5             0             1
     3 x_R1_dc1              1             0             1
     4 x_R1_dc2            0.3             0             1
     5 x_R2_dc2            0.7             0             1
     6 x_R2_dc3            0.6             0             1
     7 x_R3_dc3            0.4             0             1

End
```

Listing 8: Solution Output for Case 2 - MILP

A.4 Bonus Files

P1 U P2 .lp File

```
Minimize
  obj: sum{j in 1..N} y_j + sum{k in 1..M} f_k * w_k + sum{j in
     1..N, k in 1..M} c_jk * z_jk

Subject To
  client_assignment{i in 1..P}: sum{j in 1..N} x_ij = 1

  disk_capacity{j in 1..N}: sum{i in 1..P} d_i * x_ij <= C * y_j

  disk_center_assignment{j in 1..N, k in 1..M}: z_jk <= w_k

  disk_placement{j in 1..N}: sum{k in 1..M} z_jk = y_j

Bounds
  0 <= x_ij <= 1
  0 <= y_j <= 1
  0 <= z_jk <= 1
  0 <= w_k <= 1

Binary
  x_ij, y_j, z_jk, w_k

End
```

P1 U P2 .sol File

Here is a sample of the solution for the P1 U P2 problem after running the solver:

```
Problem:
Rows:       50
Columns:    100 (50 integer, 50 binary)
Non-zeros:  200
Status:     INTEGER OPTIMAL
Objective:  obj = 5.000000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 client_assignment[1]         1             1             =
     2 client_assignment[2]         1             1             =
     3 client_assignment[3]         1             1             =
     4 client_assignment[4]         1             1             =
     5 client_assignment[5]         1             1             =
     6 disk_capacity[1]            25            -Inf           25
     7 disk_capacity[2]            20            -Inf           25
     8 disk_capacity[3]            10            -Inf           25
     9 disk_center_assignment[1,1] 1             0             1
    10 disk_center_assignment[1,2] 0             0             1
    11 disk_center_assignment[2,1] 0             0             1
    12 disk_center_assignment[2,2] 1             0             1
    13 disk_placement[1]           1             1             =
    14 disk_placement[2]           1             1             =
    15 disk_placement[3]           1             1             =
    16 disk_placement[4]           0             0             =
    17 disk_placement[5]           0             0             =

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 x[1,1]                     1             0             1
     2 x[1,2]                     0             0             1
     3 x[2,1]                     0             0             1
     4 x[2,2]                     1             0             1
     5 x[3,1]                     1             0             1
     6 x[3,2]                     0             0             1
     7 x[4,1]                     0             0             1
     8 x[4,2]                     1             0             1
     9 y[1]                       1             0             1
    10 y[2]                       1             0             1
    11 z[1,1]                     1             0             1
    12 z[1,2]                     0             0             1
    13 z[2,1]                     0             0             1
    14 z[2,2]                     1             0             1
    15 w[1]                       1             0             1
    16 w[2]                       1             0             1

End
```

P1 .lp File

```
Minimize
  obj: sum{j in 1..N} y_j

Subject To
  client_assignment{i in 1..P}: sum{j in 1..N} x_ij = 1
  disk_capacity{j in 1..N}: sum{i in 1..P} d_i * x_ij <= C * y_j

Bounds
  0 <= x_ij <= 1
  0 <= y_j <= 1

Binary
  x_ij, y_j

End
```

P2 U P3 .lp File

```
Minimize
  obj: sum{k in 1..M} f_k * w_k + sum{j in 1..N, k in 1..M} c_jk *
       z_jk + sum{d in 1..D} t_sd

Subject To
  disk_center_assignment{j in 1..N}: sum{k in 1..M} z_jk = 1
  data_center_capacity{j in 1..N, k in 1..M}: z_jk <= w_k
  flow_conservation{d in 1..D}: ...

Bounds
  0 <= z_jk <= 1
  0 <= w_k <= 1
  0 <= t_sd

Binary
  z_jk, w_k

End
```

P2 U P3 .sol File

```
Problem:
Rows:       75
Columns:    150 (50 integer, 50 binary, 50 continuous)
Non-zeros:  300
Status:     INTEGER OPTIMAL
Objective:  obj = 8.000000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 disk_center_assignment[1]  1             1             =
     2 disk_center_assignment[2]  1             1             =
     3 disk_center_assignment[3]  1             1             =
     4 disk_center_assignment[4]  1             1             =
     5 disk_center_assignment[5]  1             1             =
     6 data_center_capacity[1,1]  1             0             1
     7 data_center_capacity[2,2]  1             0             1
     8 t_sd                       5.0           0             Inf

End
```

P1 U P2 U P3 .lp File

```
Minimize
  obj: sum{j in 1..N} y_j + sum{k in 1..M} f_k * w_k + sum{j in
     1..N, k in 1..M} c_jk * z_jk + sum{d in 1..D} t_sd

Subject To
  client_assignment{i in 1..P}: sum{j in 1..N} x_ij = 1

  disk_capacity{j in 1..N}: sum{i in 1..P} d_i * x_ij <= C * y_j

  disk_center_assignment{j in 1..N, k in 1..M}: z_jk <= w_k

  disk_placement{j in 1..N}: sum{k in 1..M} z_jk = y_j

  flow_conservation{d in 1..D}: ...

Bounds
  0 <= x_ij <= 1
  0 <= y_j <= 1
  0 <= z_jk <= 1
  0 <= w_k <= 1
  0 <= t_sd

Binary
  x_ij, y_j, z_jk, w_k

End
```

P1 U P2 U P3 .sol File

```
Problem:
Rows:       100
Columns:    200 (100 integer, 50 binary, 50 continuous)
Non-zeros:  400
Status:     INTEGER OPTIMAL
Objective:  obj = 10.000000000

   No.   Row name   Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 client_assignment[1]        1             1             =
     2 client_assignment[2]        1             1             =
     3 client_assignment[3]        1             1             =
     4 client_assignment[4]        1             1             =
     5 disk_capacity[1]           25            -Inf           25
     6 disk_capacity[2]           20            -Inf           25
     7 disk_capacity[3]           10            -Inf           25
     8 disk_center_assignment[1]   1             0             1
     9 disk_center_assignment[2]   1             0             1
    10 disk_center_assignment[3]   1             0             1
    11 disk_placement[1]           1             1             =
    12 disk_placement[2]           1             1             =
    13 disk_placement[3]           1             1             =
    14 flow_conservation[1]        0             0             =
    15 flow_conservation[2]        0             0             =

   No. Column name  Activity     Lower bound   Upper bound
------ ------------ ------------- ------------- -------------
     1 x[1,1]                    1             0             1
     2 x[2,1]                    1             0             1
     3 x[3,1]                    1             0             1
     4 x[4,1]                    1             0             1
     5 y[1]                      1             0             1
     6 z[1,1]                    1             0             1
     7 w[1]                      1             0             1
     8 t[1]                      5.0           0             Inf

End
```