Mathematical Modeling and Algorithm Application
ISSN: 3006-0842 | Vol. 3, No. 3, 2024

Research on Debt Network Repayment Problem Based on Mixed Integer Linear Programming Modeling

Lichen Pan

Xi' an Jiaotong-Liverpool University, Suzhou, 215123, China

Abstract: This study aims to optimize the payment path in the debt network by constructing a mathematical model, thereby minimizing the number of payments in the debt repayment process. The key to the study is to reduce the number of edges with non-zero weights in the network and optimize the debt repayment path. To this end, this paper divides the nodes in the debt network into two categories: debtors and creditors, and transforms the problem into a classic transportation problem. However, unlike the linear programming model of the transportation problem, the goal of this study is to minimize the number of paths in the network. Therefore, a mixed integer linear programming model is established by introducing binary variables. Finally, the model is verified to be effective through Python simulation experiments and the gurobi optimization solver is used to solve the model, demonstrating its advantages in optimizing debt networks.

Keywords: Linear Programming, Debt Network, Transportation Problem, Python Programming

1. Introduction

1.1. Research Background

The repayment problem in debt networks is widely present in the financial system, especially in the debt management of complex economies and large enterprises, involving the capital flow and repayment arrangements of multiple creditors and debtors. How to simplify the repayment process, reduce transaction costs, and reduce potential default risks while maintaining the stability of the financial system has become an important research topic in the intersection of finance and operations research. Past research has focused on optimizing the debt repayment process through traditional linear programming (LP) models or heuristic algorithms, but when faced with more complex multi-party networks, these methods are difficult to handle multi-path payment problems, resulting in too many payments and long paths. In order to solve this problem, it is necessary to build a more sophisticated optimization model.

1.2. Research Significance

1.2.1. Theoretical Significance

The mixed integer linear programming (MILP) model proposed in this paper transforms the debt repayment problem into a classic transportation problem, thereby reducing the complexity of the payment path in the debt network. This new model provides a more efficient mathematical framework for solving multi-party debt repayment problems and expands the theoretical tools in the field of operations research and financial network optimization.

1.2.2. Practical significance

By optimizing the number of payments in the debt network, this study not only helps to reduce the transaction costs of financial institutions and enterprises, but also reduces the systemic risks in the process of multi-party debt repayment and improves the efficiency of debt repayment. Especially in situations such as financial crises or corporate bankruptcy reorganization, this model can provide practical and effective solutions.

1.3. Research core

The core of this study is to optimize the payment path in the debt network by constructing a mixed integer linear programming (MILP) model to reduce the number of non-zero weight edges in the network, thereby simplifying and optimizing the debt repayment process. Specifically, this core goal can be explained at the following levels:

1.3.1. Structural analysis in debt networks

In a typical debt network, nodes represent debtors or creditors, and edges represent the debt relationship between the two parties. The weight of the edge represents the amount of debt. When the debt relationship in the network becomes complicated, the path involved in payment may contain a large number of non-zero weight edges, that is, multiple debt payment paths, which increases the complexity and cost of repayment.

1.3.2. Optimization goal - reduce non-zero weight edges

The focus of the study is on how to reduce the number of edges with non-zero weights. In the debt network, each non-zero weight edge means a debt payment. Through optimization, the debt repayment in the entire network can be completed through fewer payment paths, thereby reducing complexity and improving payment efficiency. That is, reduce unnecessary debt flow paths and concentrate payments on the least paths.

1.3.3. Minimize the number of debt repayments by optimizing payment paths

The payment paths in the debt network are optimized through the mixed integer linear programming (MILP) model. The MILP model uses integer decision variables to indicate whether to pay on a certain path, uses linear constraints to ensure the balance of debt repayment, and minimizes the number of edges with non-zero weights by optimizing the objective function, thereby achieving the simplest debt payment plan.

1.3.4. The problem is transformed into a classical transportation problem

To achieve the above goals, this paper divides the nodes in the debt network into two categories: debtors (borrowers) and creditors (lenders). This division allows the debt repayment problem of the entire network to be transformed into a classic transportation problem, where the debtor is equivalent to the "supply point" and the creditor is equivalent to the "demand point". By solving the optimal solution to the transportation problem, the minimum payment path for debt repayment is determined, thereby reducing the repayment cost and complexity.

1.3.5. Core Overview of the Research

The core of this study is achieved through the following steps:

1. Construct a debt network and divide the nodes into debtors and creditors.
2. Use the mixed integer linear programming (MILP) model to determine the optimal payment path.
3. Optimize the debt repayment process by reducing the number of edges with non-zero weights in the network, so that the payment paths involved in the network are minimized.
4. By converting the problem into a transportation problem, the complexity of the model solution is further simplified and the efficiency is improved.

1.4. Main content

The main content of this paper includes: first, briefly introduce the relevant definitions and theoretical foundations of linear programming (LP), integer linear programming (ILP), mixed integer linear programming (MILP), transportation problems and debt networks; then, propose the corresponding optimization model; finally, verify the effectiveness of the proposed model through Python experiments, and show the optimization results and their performance in reducing the number of payments.

2. Preliminary knowledge

2.1. Linear Programming (LP)

2.1.1. Definition:

Linear programming is a type of optimization problem whose goal is to maximize or minimize a linear objective function subject to a set of linear inequality or equality constraints.

2.1.2. Objective function:

Usually in the form of maximize/minimize $c^T x$, where c is the coefficient vector and x is the decision variable vector.

2.1.3. Constraints:

In the form of Ax≤b, where A is the coefficient matrix and b is the constraint vector.

2.1.4. Variable values:

In linear programming, the decision variable x can be a continuous real number, that is, x∈ R^n .

2.2. Integer Linear Programming (ILP)

2.2.1. Definition:

Integer linear programming is similar to linear programming, but requires that the decision variables must be integers.

2.2.2. Objective function:

Same as linear programming, in the form of maximize/minimize $c^T x$, where c is the coefficient vector and x is the decision variable vector.

2.2.3. Constraints:

Also the same as linear programming, in the form of Ax≤b.

2.2.4. Variable values:

In integer linear programming, all or some of the decision variables x must be integers, that is, x∈ Z^n or x∈ $Z^m \subseteq R^n$.

2.3. Mixed Integer Linear Programming (MILP)

2.3.1. Definition:

The mixed integer linear programming problem is to maximize or minimize a linear objective function, subject to a set of linear inequalities or equality constraints, and some decision variables must be integers, and some decision variables can be continuous real numbers.

2.3.2. Objective function:

In the form of maximize/minimize $c^T x$, where c is the coefficient vector and x is the decision variable vector.

2.3.3. Constraints:

In the form of Ax ≦ b, where A is the coefficient matrix and b is the constraint vector.

2.3.4. Variable values:

In MILP, some decision variables $x_1, x_2, \ldots, x_k$ must be integers, and some decision variables $x_{k+1}, \ldots, x_n$ can be continuous real numbers. That is:

$x_i \in Z$ (Integer) for some i

$x_j \in R$ (Continuous variable) for other j

2.3.5. Features:

Mixture of continuous and integer variables: Compared with integer linear programming, mixed integer linear programming allows some variables to be continuous values, which increases the flexibility of the problem, but the complexity of the problem also increases.

2.3.6. Difficulty of solution:

The difficulty of solving MILP is between linear programming (LP) and pure integer linear programming (ILP). It is still an NP-hard problem. Commonly used solution methods include branch and bound method, cutting plane method, etc., which usually require the help of solvers such as CPLEX, Gurobi or other optimization tools.

2.3.7. Summary:

In summary, MILP is a linear programming problem that handles both integer variables and continuous variables. It is widely used in modeling complex optimization problems, such as production planning, logistics, scheduling and other fields.

2.4. Debt network

Debt network refers to the network structure formed by different entities (such as individuals, enterprises, financial institutions, governments, etc.) in the financial system or economy through debt relationships. The network reflects the debt interactions between different nodes (debtors and creditors), which are usually composed of financial instruments such as debt contracts, loans, and bonds. The debt network studies how these debt relationships spread risks within the economic system, affect stability, and generate systemic risks.

2.4.1. Basic elements of the debt network

2.4.1.1. Nodes:

The nodes in the debt network represent different economic entities, which can be individuals, companies, banks, countries, etc. Each node may exist as a debtor (borrower) and a creditor (lender) at the same time.

2.4.1.2. Edges:

The edges in the network represent debt relationships, usually with directed edges to indicate the direction of the debt (who owes whom), from the debtor to the creditor. The weight of the edge represents the size or amount of the debt.

2.4.1.3. Debt relationships:

Bilateral debt: refers to a direct debt relationship between two nodes. For example, A borrows money from B, A is the debtor, and B is the creditor.

Multilateral debt: refers to an indirect debt network formed through multiple debt relationships. For example, A borrows money from B, and B borrows money from C, thus forming a multi-level debt chain.

2.5. Transportation Problem

2.5.1. Definition:

The transportation problem is a classic linear programming problem, which aims to determine how to transport goods from multiple supply points (such as factories) to multiple demand points (such as warehouses or markets) to minimize transportation costs. It is a typical problem in logistics and operations research, and is widely used in optimizing resource allocation, cost management, etc.

2.5.2. Basic composition of the transportation problem:

2.5.2.1. Source:

- Represents the departure point of the goods, such as factories, warehouses, etc.

- Each supply point has a fixed supply s_i, which represents the total amount of goods that the point can provide.

2.5.2.2. Destinations:

- Represents the arrival point of the goods, such as stores, customers, etc.

- Each demand point has a fixed demand d_j, which represents the total amount of goods required by the point.

2.5.2.3. Transportation costs:

- Each pair of supply points and demand points has a unit transportation cost c_{ij}, which represents the cost of transporting one unit of goods from supply point i to demand point j.

2.5.2.4. Transportation quantities:

- The variable x_{ij} represents the quantity of goods transported from supply point i to demand point j.

2.5.2.5. Objective:

- The objective of the transportation problem is to determine the transportation quantity x_{ij} from each supply point to each demand point, so that the total transportation cost is minimized while satisfying the supply capacity of the supply point and the demand requirements of the demand point.

2.5.3. Mathematical model of the transportation problem:

2.5.3.1. Objective function:

$$Minimize \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

Where c_{ij} is the unit transportation cost from supply point i to demand point j, and x_{ij} is the transportation quantity from i to j.

2.5.3.2. Supply constraints:

The total transportation quantity of each supply point cannot exceed its supply capacity:

$$\sum_{j=1}^{n} x_{ij} \le s_i, \forall i = 1,2,..., m \tag{2}$$

2.5.3.3. Demand constraints:

The quantity of goods received by each demand point must meet its demand:

$$\sum_{i=1}^{m} x_{ij} \ge d_j, \forall j = 1,2,..., n \tag{3}$$

2.5.3.4. Non-negativity constraints:

The transportation quantity cannot be negative:

$$x_{ij} \ge 0, \forall i, j \tag{4}$$

2.5.4. Three classifications of transportation problems:

2.5.4.1. Balanced transportation problem:

This is the type mainly discussed in this article. The total supply is equal to the total demand, which is called "balance", that is, $\sum_{i=1}^{m} s_i = \sum_{j=1}^{n} d_j$

2.5.4.1.1. Objective function:

$$Minimize \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{5}$$

Where, c_{ij} is the unit transportation cost from supply point i to demand point j, x_{ij} and is the transportation volume from i to j.

2.5.4.1.2. Constraints:

Supply restriction: The total transportation volume of each supply point is equal to its supply capacity:

$$\sum_{j=1}^{n} x_{ij} = s_i, \forall i = 1,2,..., m \tag{6}$$

Demand restriction: The amount of goods received by each demand point is equal to its demand:

$$\sum_{i=1}^{m} x_{ij} = d_j, \forall j = 1,2,..., n \tag{7}$$

Non-negativity constraint: The transportation volume cannot be negative:

$$x_{ij} \geq 0, \forall i, j \tag{8}$$

2.5.4.2. Oversupply problem:

If the total supply is greater than the total demand, it is called "oversupply problem", that is $\sum_{i=1}^{m} s_i > \sum_{j=1}^{n} d_j$, for the oversupply problem, a virtual demand point can be introduced to absorb the excess supply, and the transportation cost is zero.

2.5.4.3. Excess demand problem:

If the total supply is less than the total demand, it is called "excess demand problem", that is $\sum_{i=1}^{m} s_i < \sum_{j=1}^{n} d_j$, for the excess demand problem, a virtual supply point can be introduced to provide the missing demand with zero transportation cost.

3. Problem description and model establishment

3.1. Background of multilateral debt relations

In the actual financial system or business environment, there are often complex debt relationships between multiple debtors (borrowers) and creditors (lenders). This relationship is not limited to one-to-one debt repayment, but may involve multiple debtors and multiple creditors, forming a multilateral debt network. For example, enterprise A owes money to enterprise B, enterprise B owes money to enterprise C, and enterprise C may also owe money to enterprise A. This complex chain of intertwined debts increases the complexity of repayment and transaction costs.

3.2. Structure of debt network

The debt network can be represented in the form of a graph:

(1) Node: represents the debtor and creditor.

(2) Edge: represents the debt relationship between nodes. The direction of the edge points from the debtor to the creditor, and the weight of the edge represents the amount of the debt.

This network structure reflects the debt repayment needs of all parties. As the number of debtors and creditors increases, the payment path in the network becomes complicated, and too many payment paths not only increase the difficulty of repayment, but also increase systemic risks.

A relatively simple example is shown in Figure 1.

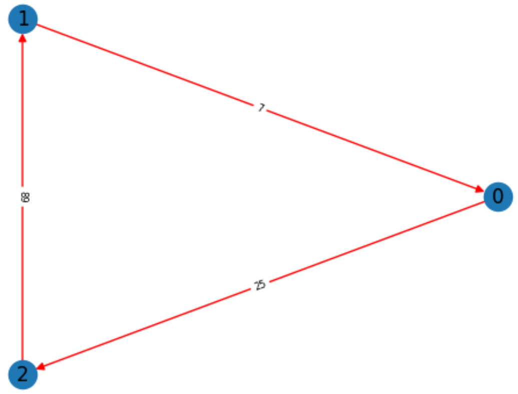

Figure 1 Simple example of debt network

There are nodes 0, 1, and 2.

Node 0 owes 25 to 2, 1 owes 7 to 0, and 2 owes 89 to 1.

3.3. The problem is analogous to the transportation problem.

By analogy, the debt repayment process in the debt network can be regarded as a transportation problem, where:

(1) The debtor (borrower) can be regarded as a supply point. They need to repay a certain amount of debt to the creditor, which is equivalent to providing a certain amount of "goods" in the transportation problem.

(2) The creditor (lender) can be regarded as a demand point. They expect to receive money from the debtor, which is equivalent to the demand in the transportation problem.

(3) The debt amount (edge weight) corresponds to the transportation volume, that is, the specific amount that a debtor needs to repay to a creditor.

(4) In the transportation problem, the goal is usually to transport the goods from the supply point to the demand point at the lowest cost. Similarly, in the debt repayment problem, the goal is to repay all debt relationships through the least payment path.

3.4. According to Figure 1

No. 0 and No. 2 are debtors (borrowers), and No. 1 is the creditor (lender).

3.5. Mathematical model

3.5.1. Mathematical variables, constraints and objective function

(1) Node set N = {1, 2, …, n}: represents all nodes in the debt network, including debtors and creditors.

(2) Edge set E = {(i, j) | i, j∈N}: represents the edge in the network, the direction of the edge is from debtor i to creditor j, and each edge represents a debt relationship.

(3) Debtor: Each debtor has a fixed payable debt s_i, which represents the total debt payable at that point.

(4) Creditor: Each demand point has a fixed receivable debt d_j, which represents the total debt receivable at that point.

(5) Payment amount: x_{ij} variable represents the amount of debt paid from debtor i to creditor j.

(6) Payment relationship: y_{ij} variable represents whether debtor i pays creditor j.

3.5.2. Proposition: The debt network settlement problem must be a balanced transportation problem

3.5.2.1. Proposition:

Suppose there is a debt network with several debtors and creditors. If the total debt of the debtors in the network is equal to the total claims of the creditors, then the debt settlement problem can be equivalently transformed into a balanced transportation problem.

3.5.2.2. Proof:

3.5.2.2.1. Definition:

① Debtors: Each debtor has a fixed payable debt, which represents the total debt payable at that point

② Creditors: Each demand point has a fixed receivable debt s_i, which represents the total debt receivable at that point

③ Payment amount: The variable x_{ij} represents the amount of debt paid from debtor i to creditor j.

3.5.2.2.2. Total debt and total claims:

In the debt settlement problem, the total debt of all debtors

$$\sum_{i=1}^{m} s_i = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n}\sum_{i=1}^{m} x_{ij} = \sum_{j=1}^{n} d_j ,$$

is equal to the total claims of all creditors

This means that the total demand (creditor demand) and total supply (debtor debt) in the debt settlement problem are balanced.

3.5.2.3. Balanced transportation problem:

By definition, a balanced transportation problem means that the total supply at the supply point is equal to the total demand at the demand point. Because the total debt of the debtor is equal to the total claims of the creditor, the debt repayment problem meets the conditions of a balanced transportation problem.

Therefore, the debt repayment problem of the debt network can definitely be transformed into a balanced transportation problem.

3.5.3. Detailed model

$$Minimize \sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} \tag{9}$$

s.t

For each debtor i, its total payment amount is equal to the total debt it owes, that is:

$$\sum_{j=1}^{n} x_{ij} = s_i, \forall i = 1,2,\ldots,m \tag{10}$$

For each creditor j, the repayment amount it receives must be equal to the total debt it should receive, that is:

$$\sum_{i=1}^{m} x_{ij} = d_j, \forall j = 1,2,\ldots,n \tag{11}$$

Only when the decision variable $y_{ij} = 1$, will payment be made, and the payment amount x_{ij} will not exceed the total debt of debtor i, that is:

$$x_{ij} \le y_{ij} \cdot s_i \tag{12}$$

4. Model solution and interpretation

4.1. Optimization rate

In this experiment, Python is used to solve the linear programming model. The optimization rate is calculated by comparing the original number of payments with the optimized number of payments. Specifically, the calculation formula for the optimization rate is:

$$optimaization\ rate = \frac{\text{original number } of\ payments - \text{optimized number of payments}}{\text{original number of payments}} \tag{13}$$

4.2. The structure of the debt relationship of a relatively simple experiment is shown in Figure 2:

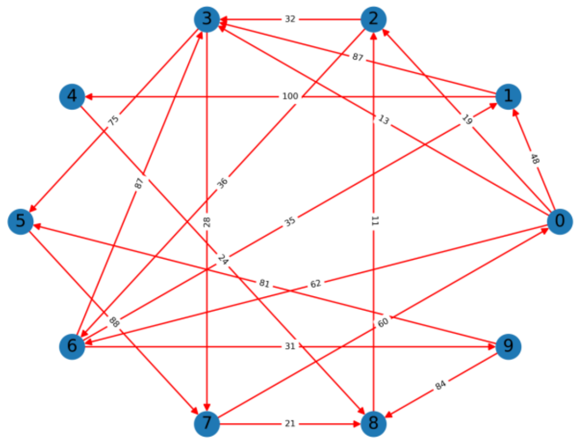

Figure 2 Experiment 1

The total number of supply points and demand points in this experiment, that is, the total number of debtors and creditors, is 10, and the number of debt relationships is 10.

The payment relationship in the figure is as follows:

0	48	19	13	0	0	62	0	0	0
0	0	0	87	100	0	0	0	0	0
0	0	0	32	0	0	36	0	0	0
0	0	0	0	0	75	0	28	0	0
0	0	0	0	0	0	0	0	24	0
0	0	0	0	0	0	0	88	0	0
0	35	0	87	0	0	0	0	0	31
60	0	0	0	0	0	0	0	21	0
0	0	11	0	0	0	0	0	0	0
0	0	0	0	0	81	0	0	84	0

The optimized payment relationship is as follows:

0	0	0	0	0	0	0	82	0	0
0	0	0	0	68	0	0	36	0	0
0	0	0	38	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	38	0	0	17	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	116	0	0	0	18	0	0	0

Using Gurobi optimization to solve the linear programming model, the results show that the optimized number of payments is 8 times, and the calculated optimization rate is 0.6.

4.3. The following is a larger debt network and the optimized results:

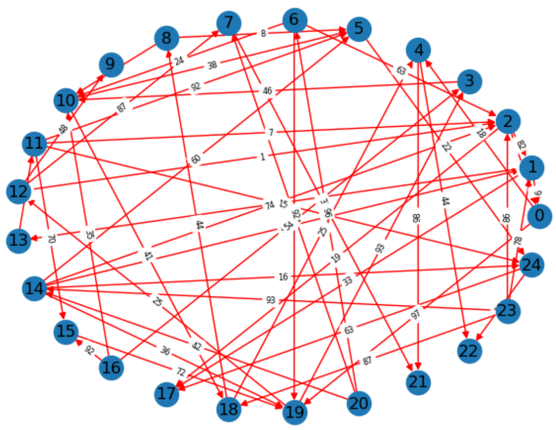

Figure 3 Experiment 1 on larger debt network

The total number of supply points and demand points in this experiment, that is, the total number of debtors and creditors, is 25, and the number of debt relationships is 50.

Model size: 308 variables, 179 constraints, number of payments required after optimization: 18.0, optimization rate 0.64.

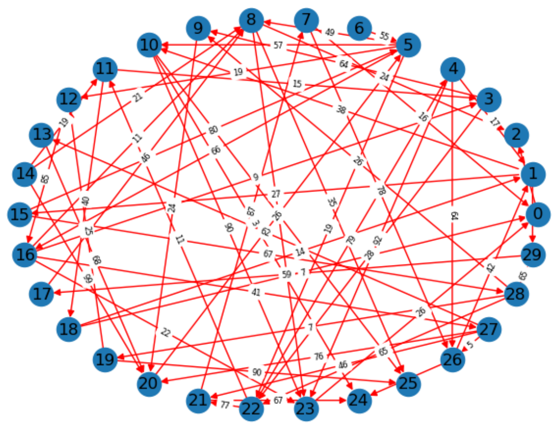

Figure 4 Experiment 2 on larger debt network

The total number of supply points and demand points in this experiment, that is, the total number of debtors and creditors, is 30, and the number of debt relationships is 60.

Model size: 442 variables, 251 constraints. Number of payments required after optimization: 22.0, optimization rate 0.63.

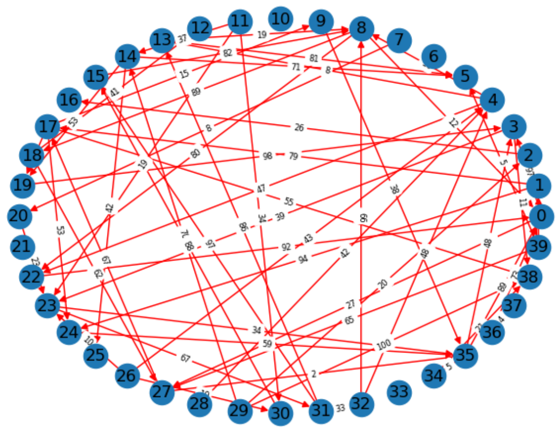

Figure 5 Experiment 2 on larger debt network

The total number of supply points and demand points in this experiment, that is, the total number of debtors and creditors, is 40, and the number of debt relationships is 60.

Model size: 616 variables, 344 constraints, number of payments required after optimization: 25.0, optimization rate 0.58.

5. Conclusion

In the case of a small number of debtors and creditors, such as 20 debtors and creditors, the optimization rate will increase with the increase of the complexity of the debt network. However, as the number of debtors and creditors increases and the complexity of the debt network increases, the optimization rate shows a significant downward trend. At the same time, the solution time also increases significantly, which may have an adverse impact on the actual application efficiency of the model. The increase in the number of debtors and creditors has a more significant impact on the solution time of the model than the increase in the complexity of the debt network. This shows that the performance of the model in complex networks may be limited by scale and complexity.

References

[1] https://jxpt.whut.edu.cn/meol/analytics/resPdfShow.do;jsessionid=98ABE8CA4ADBA6D96379B0086F4555B1?resId=171030&lid=16101

[2] https://books.google.co.uk/books?hl=zh-CN&lr=&id=ewM5_t5gtNsC&oi=fnd&pg=PA1&dq=operational+research%E6%95%99%E6%9D%90&ots=irtj-o-jKy&sig=PXErYTSLgulrk1NhdHxqy8mtIx0#v=onepage&q&f=false

[3] https://www.pdfiles.net/pdf/view/Head-First-Python

[4] https://pulpproject.org/pulpcore/

[5] https://www.gurobi.com/resources/gurobi-licenses-and-resources-for-academics/