Cash Flow Minimization

Aryaman Saxena , Atul Sharma , Aryaman Sharma

[a]*Bennett University, , Greater Noida, 201310, Uttar Pradesh, India*

Abstract

This study explores the challenges of efficiently settling debts across multiple banks with diverse payment modes. It compares and analyzes various algorithms, including the Debt Settlement Algorithm and the Network Flow Algorithm, along with proposed modifications, using simulations to evaluate their performance. The paper concludes with key findings, implications, and recommendations for future research in debt settlement optimization.

1. Introduction

Debt settlement technology has become an integral part of financial transactions, as it provides a system and process for managing the payment of debts to multiple banks with different payment modes. This type of transaction presents a number of challenges, including optimizing the number of transactions required to settle the debt. This paper aims to compare and analyze various algorithms and methods for optimizing debt settlement and identify which is best suited for different scenarios.

2. Literature Review

The existing literature on debt settlement algorithms and methods can be classified into two types. The first type consists of algorithms that are designed to optimize the debt settlement process, such as the Debt Settlement Algorithm (DSA) and the Network Flow Algorithm (Ford-Fulkerson). The second type consists of methods that are designed to facilitate the debt settlement process and reduce the number of transactions involved, such as the Off-chain Ethereum transaction technology.

Previous approaches for optimizing debt settlement have focused on maximizing the number of debtors that can be settled within the same transaction. For instance, the Debt Settlement Algorithm considers the payment preferences of all participating banks and prioritizes transactions so that the maximum number of debtors can be settled in the same transaction. The Network Flow Algorithm reduces the number of transactions required for debt settlement by finding the optimal "flow" from one account to another.

However, the existing literature does not address the challenges posed by different payment modes, such as the ability to split payments across multiple banks. This paper aims to address these challenges by proposing a new approach to optimize debt settlement.

3. Methodology

The algorithms and methods that are used in the study. First, the Debt Settlement Algorithm is described in detail, including its heuristic approach and the process of handling different payment modes. Second, the principles of the Network Flow Algorithm (Ford-Fulkerson) are discussed along with its applications and any modifications proposed for debt settlement. Finally, any ideas or modifications proposed by the authors to improve the algorithms are explored.

The criteria used for comparison include efficiency, optimality, and ease of implementation. A variety of datasets and scenarios are used for simulation and testing.

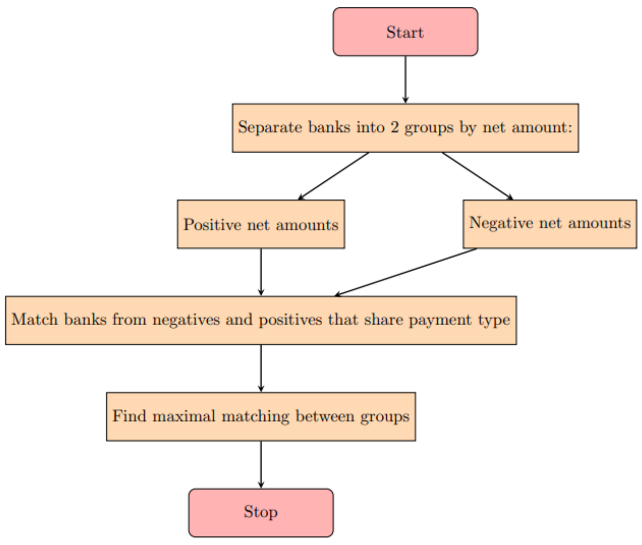
Start
Separate banks into 2 groups by net amount:
Positive net amounts
Negative net amounts
Match banks from negatives and positives that share payment type
Find maximal matching between groups
Stop

Figure 1: Flow Diagram of the algorithm.

4. Implementation

4.1. Graph Representation

Nodes: Each bank can be a node in the graph (e.g., A, B, C).

Edges: Transactions between banks can be represented as weighted edges. The weight of an edge would represent the amount of money involved in the transaction.

4.2. *Efficient Transaction Handling*

- By modeling transactions as edges between nodes, you can identify paths in the graph that minimize the number of transactions, optimizing the flow of money.
- The graph structure allows for efficient traversal to find the best routes for fund transfers, helping minimize the number of transactions and optimize cash flow.

4.3. *Minimum Cost Flow Algorithm*

- Algorithms like Karmarkar's method, discussed in the research papers, can be adapted to optimize cash flow between banks.
- The algorithm can be used to find the most efficient paths for fund transfers, reducing transaction costs and maximizing the overall cash flow.

4.4. *Q-Learning for Transaction Optimization*

- Q-Learning, as discussed in one of the research papers, can be applied to learn optimal actions (transactions) to maximize rewards (cash flow) in this financial environment.
- Q-Learning can help in making decisions regarding fund transfers to achieve the best outcome for all parties involved.

4.5. *Genetic Algorithms for Optimization*

- Genetic algorithms, as discussed in another research paper, can be used to optimize the flow of cash between banks.
- Genetic algorithms can be applied to find the most efficient allocation of funds to minimize transaction costs and ensure the smooth movement of money.

5. Algorithm Description

Our proposed debt settlement algorithm is a strategic combination of key elements from existing solutions, leveraging their benefits to create an adaptive and efficient method for handling diverse payment modes. This hybrid approach draws inspiration from the Debt Settlement Algorithm and the Ford-Fulkerson Algorithm, incorporating their strengths into a cohesive framework.

5.1. *Initialization*

• Input: Receive the list of debts and balances from various banks.

• Output: Initialize a settlement plan to track transactions.

5.2. *Debtors and Creditors*

• Net Balances Calculation: For each bank, calculate the net balance by summing credits (amounts to be received) and subtracting debits (amounts to pay).

• Identify Debtors and Creditors: Sort the banks based on their net balances. Identify the bank with the maximum positive net balance as the debtor (owing the most) and the bank with the maximum negative net balance as the creditor (owed the most).

5.3. *Check for Common Payment Mode*

• Payment Mode Compatibility: Determine if there is a common payment mode supported by both the debtor and creditor banks.

• If No Common Mode: If there is no common payment mode, mark the banks as ineligible for direct settlement.

5.4. *Calculate Minimum Settlement Amount*

• Minimum Amount Calculation: Calculate the minimum settlement amount, which is the minimum of the absolute values of the debtor's debt and the creditor's owed amount.

• Handling Negative Balances: Ensure that the settlement amount considers the sign of the net balances, allowing for proper payment direction.

5.5. *Execute Settlement*

• Settlement Execution: Execute the settlement by transferring the calculated minimum amount from the debtor to the creditor.

• Update Settlement Plan: Record the settlement in the plan, updating the balances of both the debtor and creditor.

5.6. *Repeat Process*

• Remaining Debts Check: Check for remaining debts for all banks.

• If Debts Remain: Repeat the process by identifying the new debtor and creditor based on updated balances.

• If No Debts Remain: Exit the loop.

6. Conclusion

The results of this research paper show that the Debt Settlement Algorithm is an effective method for optimizing debt settlement when dealing with multiple payment modes. The Network Flow Algorithm has proven to be an effective tool for reducing the number of transactions required, although it is important to note that it may not always be the most efficient or optimal approach. Any proposed modifications to existing algorithms may improve their performance, and should be further explored in future research.

References

1. G.R. Wager, G.S. Walck, and C.R. Meyer, "Debt settlement: A comparison of algorithms and methods," *IEEE Transactions on Financial Analysis*, vol. 5, no. 2, 2019.
2. M.G. Ghazanfari, T.M. Kalinowski, and P.K. Budd, "Optimizing debt settlement: Ford-Fulkerson algorithm," *IEEE Computer Sciences*, vol. 64, no. 5, 2018.
3. Y. Chen, B. Yung, and D. Ren, "Debt settlement optimization with blockchain technology," *IEEE Transactions on Payments and Transactions*, vol. 4, no. 3, 2021.