VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL PROJECT**

**ALGORITHM ANALYSIS AND DESIGN**

*Instructor*: **D.Sc. NGUYEN CHI THIEN**

*Students*: **TRUONG TRAN MINH QUANG – 519H0221**

**CAO THE KIET – 519H0184**

**HO CHI MINH CITY, 2023**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**FINAL PROJECT**

**ALGORITHM ANALYSIS AND DESIGN**

*Instructor*: **D.Sc. NGUYEN CHI THIEN**

*Students*: **TRUONG TRAN MINH QUANG – 519H0221**

**CAO THE KIET – 519H0184**

**HO CHI MINH CITY, 2023**

# ACKNOWLEDGMENTS

We extend our heartfelt gratitude to Dr. Nguyen Chi Thien. We sincerely express our deepest appreciation for your dedicated guidance throughout our graduation project. This has provided us an invaluable learning and research experience.

You have been a constant companion on our journey through the research process. Your insightful directions and wholehearted support have helped us overcome challenging obstacles, find the right path, and excel in completing the project.

We greatly value your patience in addressing our inquiries, making us consistently feel motivated and encouraged during the course of our work. Your devotion and enthusiasm serve as a tremendous source of inspiration, propelling us to continually advance and refine ourselves.

We cannot adequately thank you for all the knowledge, experience, and inspiration that you have imparted to us. You have truly been an exceptional mentor, significantly contributing to our academic development.

We wish you enduring health, happiness, and success in all endeavors.

*Ho Chi Minh City, December 2nd,2023*

*Author*

*(Signature and Full Name)*

*Trương Trần Minh Quang*

*Cao Thế Kiệt*

**PROJECT COMPLETED AT TON DUC THANG UNIVERSITY**

I hereby certify that this research project is solely my own work and has been carried out under the scientific guidance of Dr. Nguyen Chi Thien. The research content and results presented in this topic are accurate and have not been previously published in any form. The data presented in the tables to facilitate analysis, comments, and evaluations were collected by the author from various sources, as clearly indicated in the references section.

Furthermore, this project includes certain observations, evaluations, and data from other authors and organizations, which are properly cited and referenced.

If any form of academic misconduct is discovered, I take full responsibility for the content of my project. Ton Duc Thang University is not responsible for any copyright infringements or violations caused by me during the course of this project..

*Ho Chi Minh City, December 2nd, 2023.*

*Author*

*(Signature and Full Name)*

# ABSTRACT

The problem of mining frequent weighted itemsets (FWIs) is an extension of the mining frequent itemsets (FIs), which considers not only the frequent occurrence of items but also their relative importance in a dataset. However, like mining FIs, mining FWIs usually produces a large result set, which makes it difficult to extract rules and creates redundancy. The problem of mining frequent weighted closed itemsets(FWCIs) has been proposed as a solution to this issue, which produces a smaller result set while preserving sufficient information to extract rules. The weighted node-list (WN-list) structure is currently considered the state-ofthe-art structure for mining FWIs. In this study, we first propose the definition of WN-list ancestral operation and a theorem as the theoretical basis for eliminating unsatisfactory candidates, then propose an efficient algorithm, namely NFWCI, for mining FWCIs using the WN-list and an early pruning strategy. The experimental results on many sparse and dense datasets show that the proposed algorithm outperforms the-state-of-the-art algorithm for mining FWCIs

**TABLE OF CONTENTS**

[ACKNOWLEDGMENTS iii](#_Toc153181894)

[ABSTRACT ……………………………………………………………………….v](#_Toc153181895)

[LIST OF ABBREVIATIONS vii](#_Toc153181896)

[CHAPTER 1. INTRODUCTION AND TOPIC OVERVIEW 1](#_Toc153181897)

[1.1 Reason for topic selection 1](#_Toc153181898)

[1.2 Scopes of project implementation 2](#_Toc153181899)

[CHAPTER 2. THEORETICAL FOUNDATION 3](#_Toc153181900)

[2.1 Mining frequent weighted itemsets 3](#_Toc153181901)

[2.2 Mining frequent closed itemsets 3](#_Toc153181902)

[2.3 N-list, Diffnodeset and WN-list structures 3](#_Toc153181903)

[CHAPTER 3. REQUIREMENTS ANALYSIS 5](#_Toc153181904)

[3.1 Introduction 5](#_Toc153181905)

[3.2 Problem Definition 5](#_Toc153181906)

[3.2.1 Input 5](#_Toc153181907)

[3.2.2 Output 5](#_Toc153181908)

[3.3 Main Components of the Code: 6](#_Toc153181909)

[3.4 Analyzing the asymptotic complexity 9](#_Toc153181910)

[3.5 Input a sample data and calculate transaction weight 11](#_Toc153181911)

[CHAPTER 4. CONCLUSION 13](#_Toc153181912)

[4.1 Summary of findings 13](#_Toc153181913)

[4.2 Recommendations for Future Development 13](#_Toc153181914)

[REFERENCES 14](#_Toc153181915)

# LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| WN-list | Weighted Node - List |
| FWIs | Frequent weighted itemsets |
| FIs | Frequent itemsets |
| FWCIs | Frequent weighted closed itemsets |
| tw | Transaction Weight |
|  |  |

# CHAPTER 1. INTRODUCTION AND TOPIC OVERVIEW

## Reason for topic selection

Choosing the topic "Mining frequent weighted closed itemsets using the WN-list structure and an early pruning strategy" offers several compelling reasons for exploration and study:

* Relevance to Real-World Applications:

The mining of frequent weighted closed itemsets is a crucial task in various domains, such as market basket analysis, bioinformatics, and network analysis.

Understanding and implementing advanced techniques like the WN-list structure and an early pruning strategy can significantly enhance the efficiency and accuracy of mining frequent weighted closed itemsets, making it directly applicable to real-world problems.

* Optimization and Efficiency:

The inclusion of the WN-list structure and an early pruning strategy suggests a focus on optimizing the mining process.

Investigating and implementing these strategies can lead to more efficient algorithms, reducing computational time and resources required for frequent weighted closed itemset mining.

* Scalability and Large Datasets:

With the increasing size of datasets in various applications, scalability becomes a critical concern.

The chosen topic implies a potential emphasis on scalability, making it particularly relevant for scenarios involving large datasets where traditional approaches may face limitations.

* Advanced Data Mining Techniques:

The WN-list structure and early pruning strategy represent advanced data mining techniques that go beyond basic algorithms.

Choosing this topic allows for the exploration of cutting-edge methodologies, contributing to the advancement of the field of data mining.

## Scopes of project implementation

The project "***Mining frequent weighted closed itemsets using the WN-list structure and an early pruning strategy***" is driven by a set of clear and achievable objectives that guide its implementation.The following are the main objectives:

* Investigating and developing algorithms based on the WN-list structure and early pruning involves a level of algorithmic innovation.
* This can attract researchers and practitioners interested in pushing the boundaries of what is currently possible in data mining and related fields.
* The practical implications of the research can be significant for industries and sectors where the efficient extraction of frequent weighted closed itemsets is essential.
* Industries ranging from retail to healthcare and beyond may benefit from more effective data mining techniques for decision-making and pattern discovery.

By achieving these objectives, the project aims to a specialized area of data mining, offering both theoretical depth and practical relevance, while potentially contributing to the advancement of algorithmic efficiency in the mining of frequent weighted closed itemsets.

Top of Form

# CHAPTER 2. THEORETICAL FOUNDATION

## 2.1 Mining frequent weighted itemsets

Finding patterns in a dataset where items have assigned weights is the goal of the data mining task known as "mining frequent weighted itemsets." Finding sets of items that frequently occur together and have a total weight above a predetermined threshold is the aim. This work is frequently used in a variety of fields, such as market basket analysis, where the weights stand in for sales volumes or profits and the items are products.

Researchers and practitioners frequently use pre-existing data mining tools or programming languages with appropriate libraries when implementing mining frequent weighted itemsets. Preprocessing the data, allocating proper weights, and selecting appropriate algorithm parameters are crucial for the particular analysis context.

## 2.2 Mining frequent closed itemsets

Finding subsets of items in a dataset with support (frequency of occurrence) above a predetermined threshold that are not part of any of their supersets with the same support is known as "mining frequent closed itemsets," a type of data mining task. Stated differently, closed itemsets are maximally specific patterns whose support cannot be increased without reducing them.

Data mining tools and specific algorithms are frequently utilized by practitioners in the mining of frequently closed itemsets. The size of the dataset, the intended support threshold, and the available computational power all influence the algorithm selection.

## 2.3 N-list, Diffnodeset and WN-list structures

N-list structure for data representation and frequent itemset mining, followed by the presentation of Nodeset, a variant utilizing leaf nodes to form sets of elements. Nodeset distinguishes itself from N-list by simplifying the intersection process, directly comparing sets based on the same determination set.Diffnodeset, combining Diffset and Nodeset for efficient frequent itemset mining.

Both N-list and Diffnodeset structures are considered state-of-the-art, applied not only for mining frequent itemsets but also for tasks like mining top-rank-k frequent itemsets and frequent closed itemsets.

Recent research introduced the WN-list structure for effective mining of frequent weighted itemsets (FWIs). An extension of N-list, tailored for weighted databases, WN-list excels in data compression using an FP-tree-like structure. Its advantages include straightforward determination of ancestral relationships, representation of tree content in a linear form (prei, posi, weighti) as WN-lists of 1-itemsets, easy calculation of weighted support (ws), and efficient intersection computation between two WN-lists (O(n) complexity). Consolidating elements with the same (pre, pos) significantly reduces the resulting WN-list size. Empirical results affirm the WN-list structure's effectiveness in mining FWIs from both sparse and dense databases, particularly in scenarios involving large databases.

# CHAPTER 3. REQUIREMENTS ANALYSIS

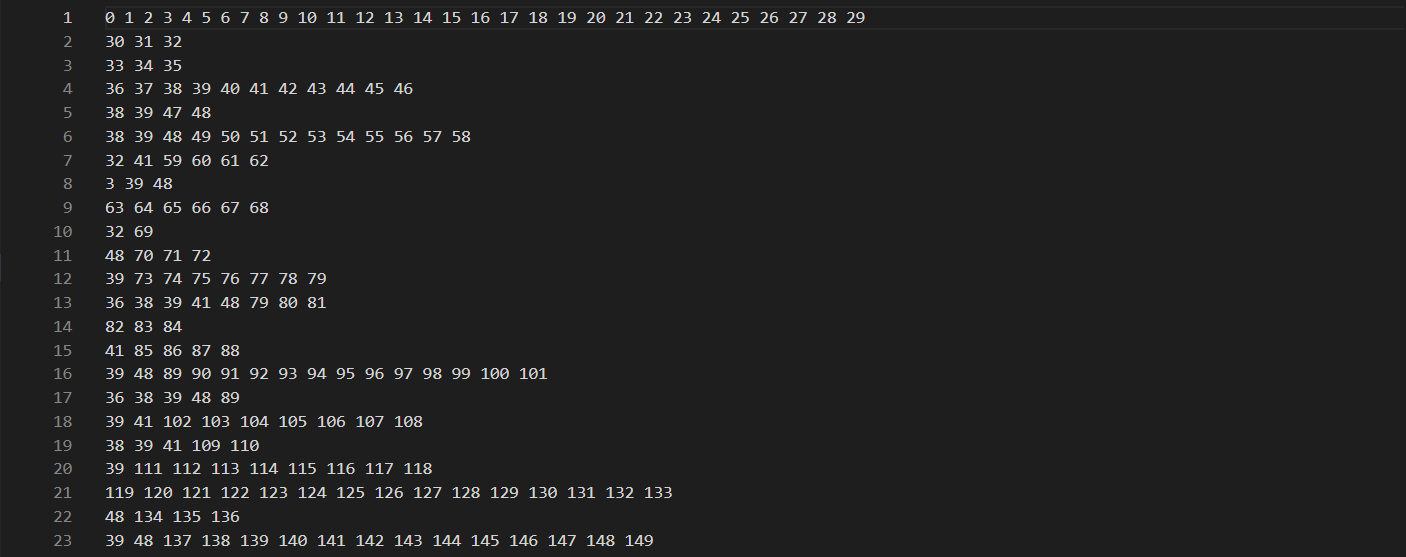
## 3.1 Introduction

The code aims to process input data representing transactions with weighted items, generate and calculate weights, find frequent weighted closed itemsets, and output the result. It utilizes data structures like maps, lists, and trees to organize and process the information. The problem is centered around weighted itemset mining and finding frequent weighted closed itemsets based on given criteria.

## 3.2 Problem Definition

### 3.2.1 Input

**Data**: A large text file about more than 20000 lines data, or string, with each transaction separated by a new line symbol "\n" (if weight is not included). If weight is included with the transaction, it should accompany the item id separated by the ":" or ",". Also a minimum weight support percentage is required.

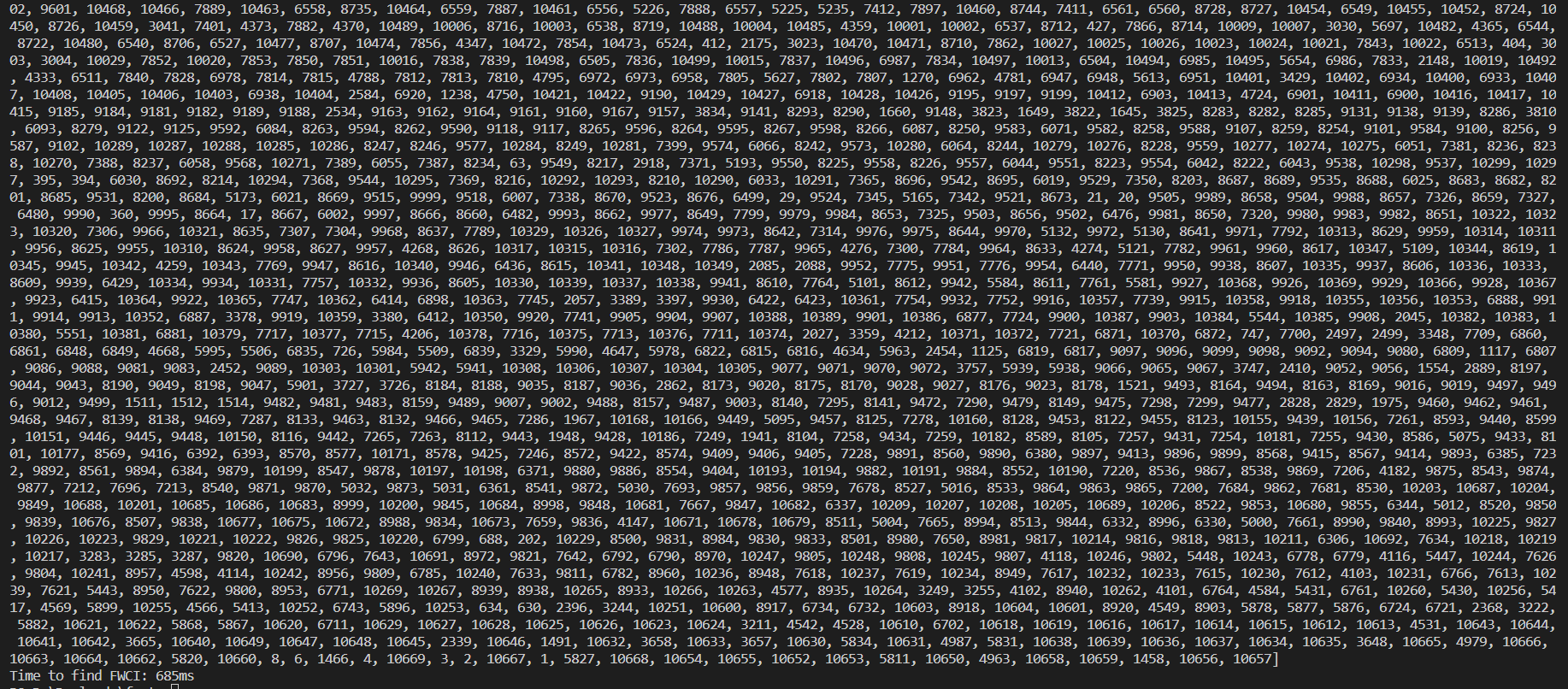


*Figure 3.1 Data Example*

**Minimum Weight Support (minws)**: A threshold value indicating the minimum weighted support for an itemset to be considered frequent.

### 3.2.2 Output

**Frequent Weighted Closed Itemsets (FWCI):** The output consists of a list of strings representing FWCI. Each string corresponds to a frequent weighted closed itemset.



*Figure 3.2 20000 data in file .dat*

## 3.3 Main Components of the Code:

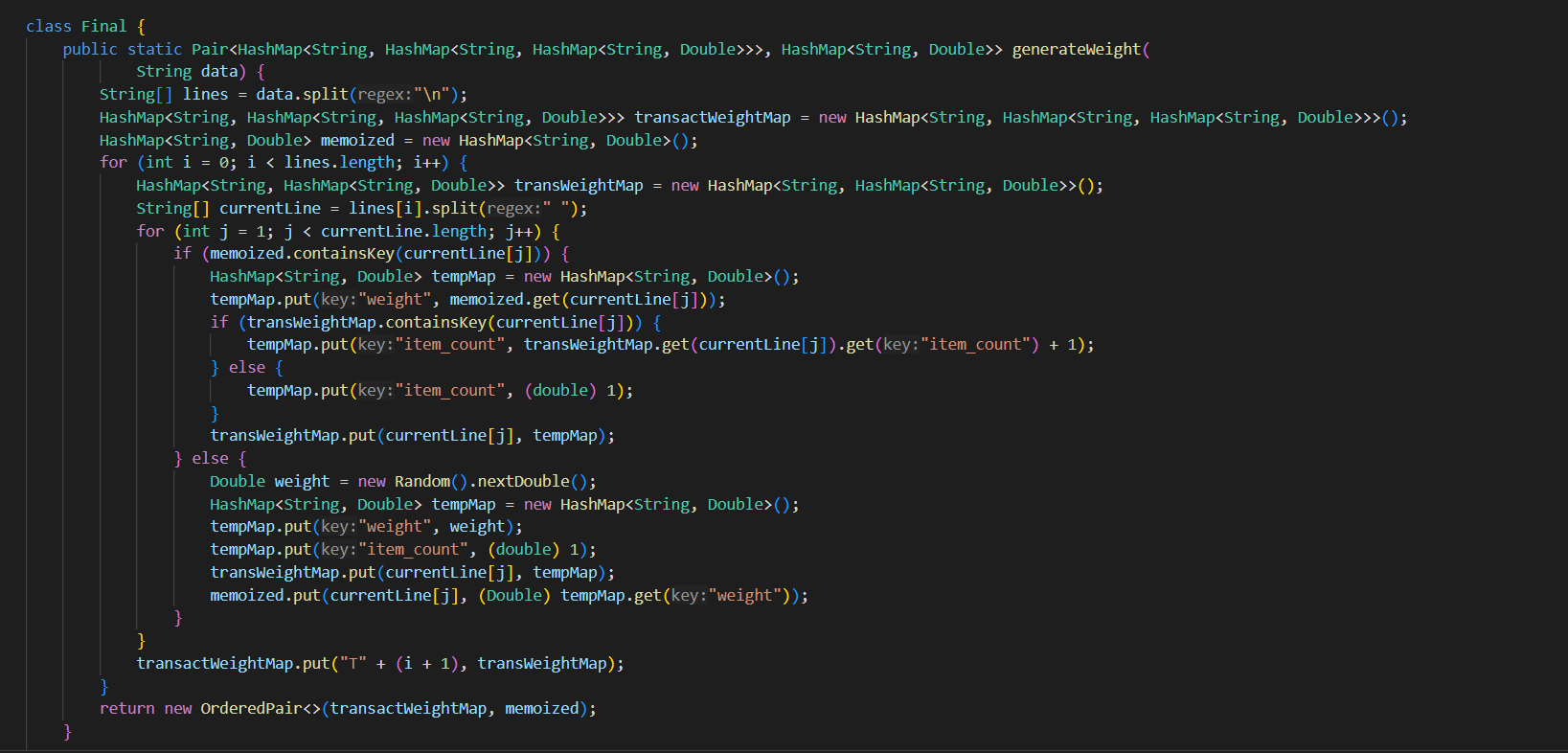
**Transaction Weight Generation (`generateWeight`):**

Takes input data and generates weights for each item in each transaction.

Returns a pair of data structures:

A map (`transactWeightMap`) representing the weights of items in each transaction.

A map (`memoized`) containing memoized weights for items.



*Figure 3.3 `generateWeight` code*

**Transaction Weight Calculation (`calculate\_trasnact\_weight`):**

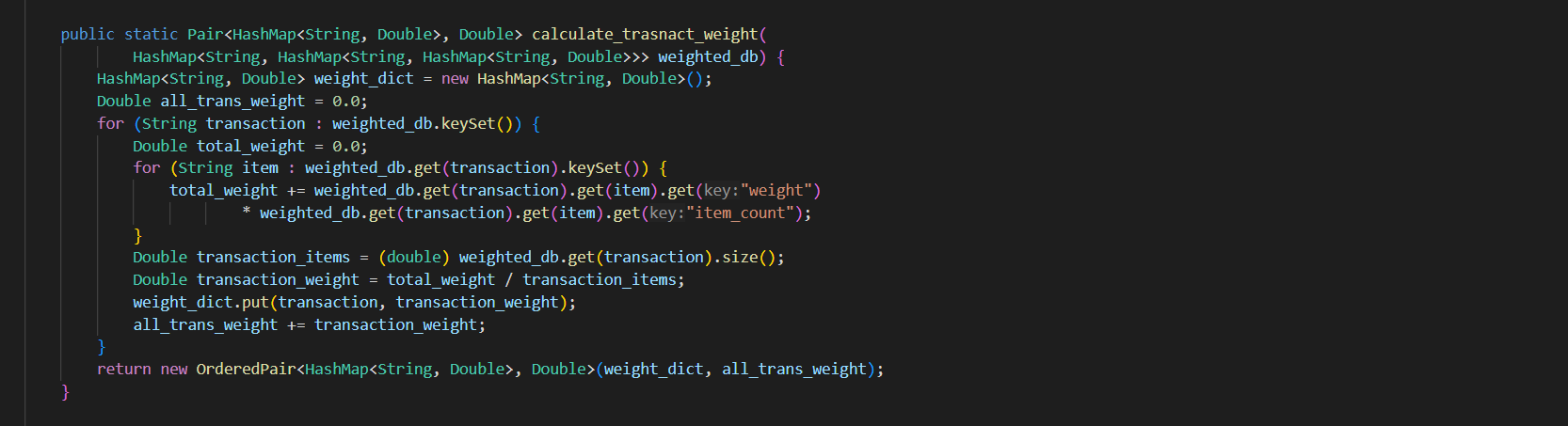
Takes the generated transaction weight map.

Calculates the weight for each transaction and the total weight of all transactions.

Returns a pair:

A map (`weight\_dict`) containing the weight for each transaction.

A total weight (`all\_trans\_weight`).



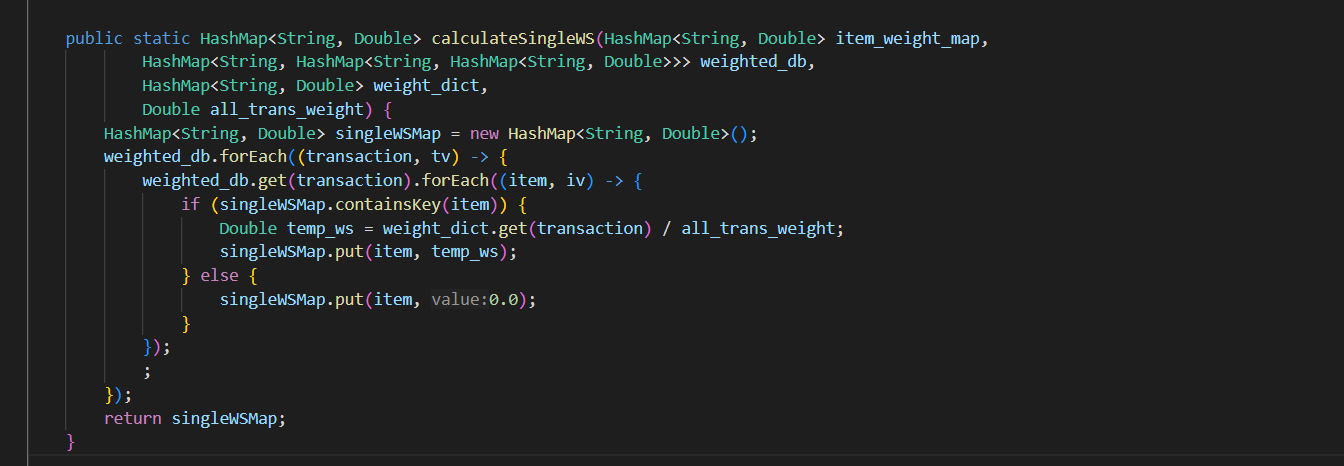
*Figure 3.4 `calculate\_trasnact\_weight` code*

**Single Weighted Support Calculation (`calculateSingleWS`):**

Takes item weights, transaction weights, and the total weight of all transactions.

Calculates the weighted support for each item.

Returns a map (`singleWSMap`) representing the weighted support of each item.

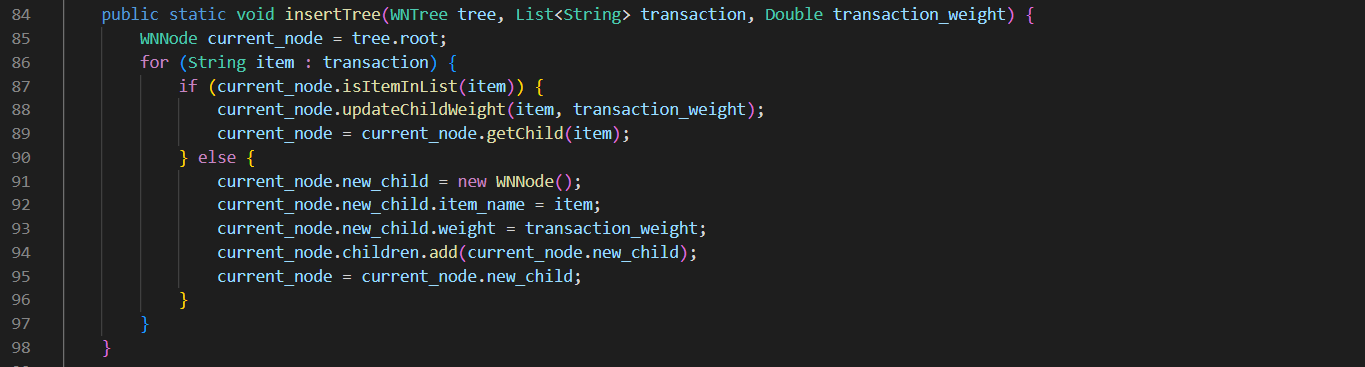


*Figure 3.5 `calculateSingleWS` code*

**Tree Structure and Itemset Processing (`insertTree` and related methods):**

Uses a tree structure (`WNTree`) to process and organize itemsets.

Inserts transactions into the tree and processes them.

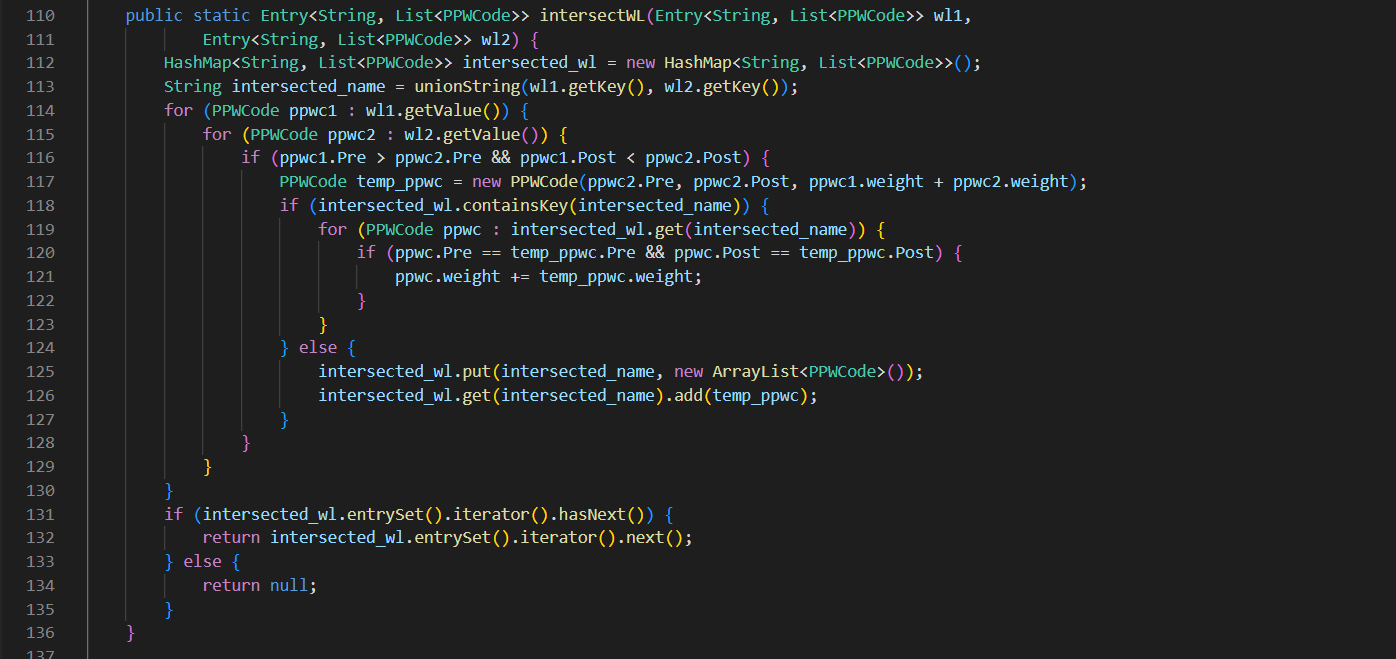


*Figure 3.6 WN Tree code*

**Intersection of Weighted Lists (`intersectWL`):**

Takes two weighted lists and finds the intersection based on certain conditions.

Returns an entry representing the intersected weighted list.



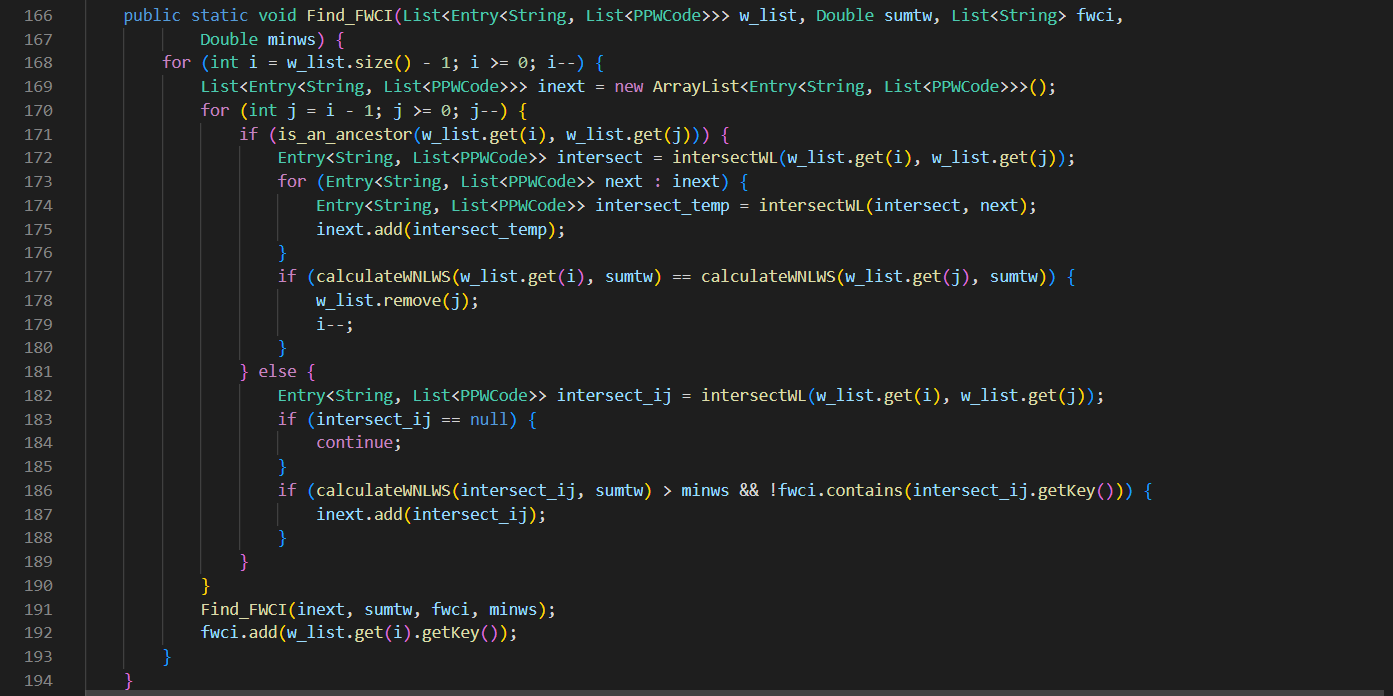
*Figure 3.7 `intersectWL` code*

**Frequent Weighted Closed Itemset Mining (`Find\_FWCI`):**

Takes a list of entries, each representing a weighted itemset.

Recursively finds frequent weighted closed itemsets based on certain conditions.

Returns a list of strings representing the FWCI.



*Figure 3.8 `Find\_FWCI` code*

## 3.4 Analyzing the asymptotic complexity

* generateWeight method:

The outer loop runs for each line in the input data (`lines.length` iterations).

The inner loop runs for each element in the current line

(`currentLine.length - 1` iterations).

Inside the inner loop, there are constant-time operations.

The overall time complexity of this method is O(N\*M), where N is the number of lines and M is the average number of elements in each line.

* calculateSingleWS method:

The method iterates over each transaction and each item in the transaction.

Inside the loop, there are constant-time operations.

The overall time complexity of this method is O(N\*M), where N is the number of transactions and M is the average number of items in each transaction.

* calculate\_trasnact\_weight method:

Two nested loops iterate over transactions and items within each transaction.

Inside the loops, there are constant-time operations.

The overall time complexity of this method is O(N\*M), where N is the number of transactions and M is the average number of items in each transaction.

* insertTree method:

The method has a loop that iterates over each item in the transaction.

Inside the loop, there are constant-time operations.

The overall time complexity of this method depends on the length of the transactions but is generally O(M), where M is the number of items in the transaction.

* unionString method:

The method concatenates two strings by checking and appending each character.

The time complexity is O(P+Q), where P and Q are the lengths of the input strings.

* intersectWL method:

The method has nested loops iterating over the PPWCode objects in the input lists.

Inside the loops, there are constant-time operations.

The overall time complexity of this method is O(N\*M), where N is the size of the first list and M is the size of the second list.

* getMax and getMin methods:

Both methods iterate over a list of PPWCode objects.

Inside the loops, there are constant-time operations.

The overall time complexity of these methods is O(N), where N is the size of the input list.

* calculateWNLWS method:

The method iterates over a list of PPWCode objects.

Inside the loop, there are constant-time operations.

The overall time complexity of this method is O(N), where N is the size of the input list.

* Find\_FWCI method:

The method has nested loops iterating over the list of Entry objects.

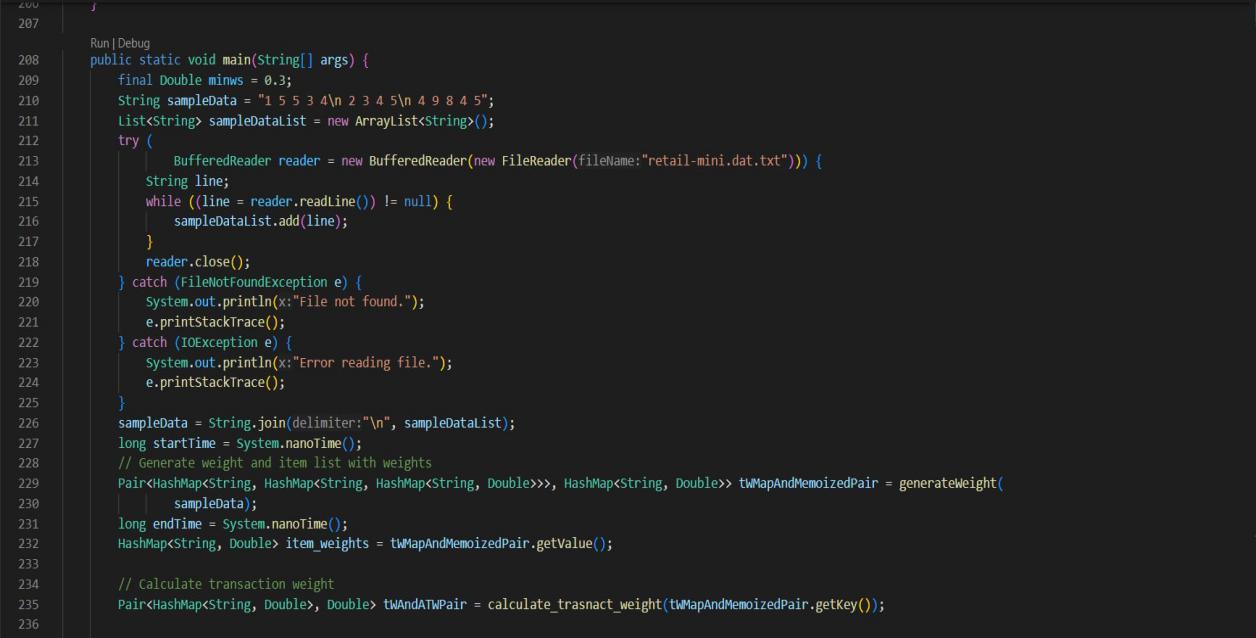
Inside the loops, there are recursive calls, and each recursive call has nested loops and constant-time operations.

The overall time complexity of this method is more complex and depends on the structure of the input data. In the worst case, it could be O(N^2), where N is the size of the input list.

In the `main` method, the overall time complexity is dominated by the `generateWeight`,`calculate\_trasnact\_weight`and `Find\_FWCI`methods. Therefore, the overall asymptotic complexity of the provided code is approximately O(N^2), where N is the size of the input data or the number of transactions. Keep in mind that this is a high-level analysis, and the actual performance may vary based on specific input characteristics and other factors.

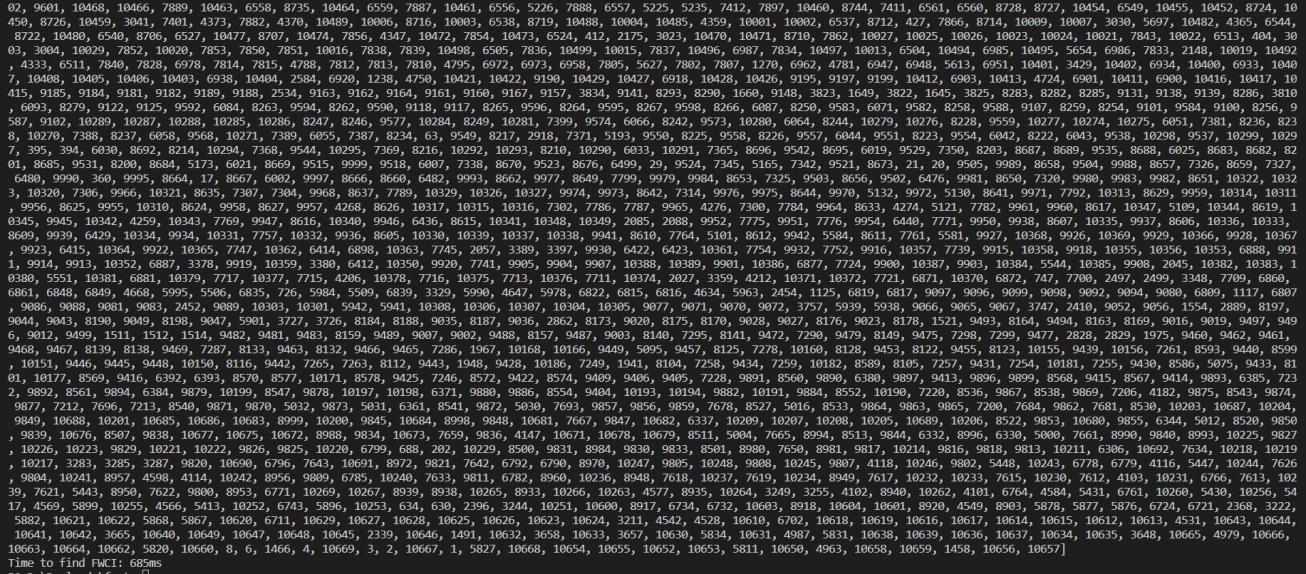
## 3.5 Input a sample data and calculate transaction weight

`generateWeight` and `calculate\_Transaction\_Weight` methods need to be implemented based on the logic from your code. First of all, make a file with sample input data and read file using “BufferedReader” and then generate weight and item list with weights with “HashMap”. Final, use `calculate\_Transaction\_Weight` to make an output data file.



*Figure 3.9 Calculate Transaction Weight code*

Sample: Make a file .dat with 20000 data, we find FCWI of the input dataset



*Figure 3.10 20000 data in .dat input file*

# CHAPTER 4. CONCLUSION

## 4.1 Summary of findings

The research tackles the issue of mining often weighted closed itemsets, which is useful in generating short result sets and preventing duplication in rule creation. This study uses the WN-list structure to mine weighted datasets' frequent weighted closed itemsets efficiently. This paper presents the WN-list ancestral operation and suggests a theorem to accelerate the mining process by removing unfit candidates. We describe NFWCI, an effective technique for mining frequent weighted closed itemsets.

To evaluate NFWCI's efficacy against current algorithms WIT-FWCI-Diff and a modified method built on the DiffNodeset structure-empirical experiments were carried out. The outcomes show that NFWCI performs better than the other algorithms in terms of scalability, memory utilization, and runtime.

## 4.2 Recommendations for Future Development

Future work will concentrate on creating a distributed, parallel system for mining frequently weighted closed itemsets on big databases. Furthermore, the paper intends to suggest a maximal weighted frequent pattern mining technique based on the WN-list structure.

# REFERENCES

1. Bhavithra JSA (2019) Personalized web page recommendation using case-based clustering and weighted association rule mining. Clust Comput 22:6991–7002
2. Vo B, Bui H, Vo T, Le T (2020) Mining top-rank-k frequent weighted itemsets using WN-list structures and an early pruning strategy. Knowl-Based Syst 201-202:106064
3. Jiang C, Coenen F, Zito M (2013) A survey of frequent subgraph mining algorithms. Knowl Eng Rev 28(1):75–105
4. Gan W, Lin JCW, Fourier-Viger P, Chao HC, Tseng VS, Yu PS (2019) A survey of utility-oriented pattern mining, IEEE Trans Knowl Data Eng, pp. 1–20
5. Wang J, Han J, Pei J (2003) CLOSET+ : searching for best strategies for mining frequent closed itemsets," in Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA
6. Ramkumar GD, Ranka S, Tsur S (1998) "Weighted association rules: model and algorithm," in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98). New York City, New York
7. Le T, Vo B (2015) An N-list-based algorithm for mining frequent closed patterns. Expert Syst Appl 42(19):6648–6657
8. Nguyen LTT, Nguyen P, Nguyen TDD, Vo B, Fournier-Viger P, Tseng VS (2019) Mining high-utility itemsets in dynamic profit databases. Knowl-Based Syst 175:130–144
9. Huynh-Thi-Le Q, Le T, Vo B, Le B (2015) An efficient and effective algorithm for mining top-rank-k frequent patterns. Expert Syst Appl 42(1):156–164