NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA

PROJECT REPORT

on

FP-Growth Algorithm

Submitted in partial fulfilment of the requirement for the award of Degree of

Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

Shubham Prasad	1NT19CS183
Shreyas Agnihotri	1NT19CS182
Sathvik Reddy Chaganur	1NT19CS170

Department of Computer Science and Engineering

2021-22

Nitte Meenakshi Institute of Technology

(AN AUTONOMOUS INSTITUTION AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM)
PB No. 6429, Yelahanka, Bangalore 560-064, Karnataka
Telephone: 080- 22167800, 22167860
Fax: 080 - 22167805

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the case study project in Data Mining (18CS54) as LA component titled **FP-Growth Algorithm** is an authentic work carried out by **Shreyas Agnihotri (1NT19CS182), Shubham Prasad (1NT19CS183) and Sathvik Reddy Chaganur (1NT19CS170)**, bonafide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfilment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2021-2022.***

Signature of the Guide | **Signature of the HOD**

Dr. SUJATA JOSHI | **Dr. SAROJA DEVI H.**

DECLARATION

I/We hereby declare that

(i) This Presentation/report does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the report and in the References sections.

(ii) All corrections and suggestions indicated during the internal presentation have been incorporated in the report.

(iii) Content of the report has been checked for the plagiarism requirement

NAME	USN	Signature
Shubham Prasad	1NT19CS183	Shubham Prasad
Shreyas Agnihotri	1NT19CS182	[illegible]
Sathvik Reddy Chaganur	1NT19CS170	[illegible]

Date: 20/01/2022

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. I/we express my/our sincere gratitude to our Principal **Dr. H. C. Nagaraj**, Nitte Meenakshi Institute of Technology for providing facilities.

I/we wish to thank our HoD**, Dr. Saroja Devi H.** for the support and encouragement for the project work. I/We also thank him for the invaluable guidance provided which has helped in the creation of a better technical report.

I/We thank **Dr. Sujata Joshi** for the guidance and support given while doing this project work and preparing the report & presentation. I/We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the presentation.

Name & USN:	Shreyas Agnihotri:	1NT19CS182
	Sathvik Reddy Chaganur:	1NT19CS170
	Shubham Prasad:	1NT19CS183

Date: 20/01/2022

ABSTRACT

Data mining is passed down to arrange with the data stored in the backend to extract the required information and expertise. It has a number of ways for finding data, association rule mining is the very robust data mining approach. Its main work to find out required hidden patterns from bulk storage. Data miners go through with many techniques, out of them frequent pattern growth is an effective algorithm to extract required association rules. It examines the directory two times for handling. FP growth algorithm has some concern to generate enormous conditional FP trees. Data miners introduce a new technique which extracts all the frequent item sets without the generation of the conditional FP trees. It also catches the frequency of the usual item sets to extract the required association rules.

TABLE OF CONTENTS

INTRODUCTION

The FP-Growth Algorithm is an alternative way to find frequent itemsets without using candidate generations, thus improving performance.FP-Growth Algorithm uses a divide-and-conquer strategy. The core of this method is the usage of a special data structure named frequent-pattern tree (FP-tree), which retains the itemset association information. In simple words, this algorithm works as follows: first it compresses the input database creating an FP-tree instance to represent frequent items. After this first step it divides the compressed database into a set of conditional databases, each one associated with one frequent pattern. Finally, each such database is mined separately. Using this strategy, the FP-Growth reduces the search costs looking for short patterns recursively and then concatenating them in the long frequent patterns, offering good selectivity.

In large databases, it's not possible to hold the FP-tree in the main memory. A strategy to cope with this problem is to firstly partition the database into a set of smaller databases, and then construct an FP-tree from each of these smaller databases.

Motivation

To implement a faster alternative to Apriori algorithm which requires lesser database scans.

Problem Statement

Implementation and demonstration of FP-Growth Algorithm using Python.

Aim and Objective

Find all frequent itemsets using minimum support.

DATA SOURCE AND DATA QUALITY

Datasets Used

Dataset of supermarkets in which we are having details of 5 transactions of items:

```
['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs',
'Yogurt'],
['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs',
'Yogurt'],
['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
['Milk',   'Unicorn',   'Corn',   'Kidney   Beans',
'Yogurt'],
['Corn', 'Onion', 'Onion', 'Ice cream', 'Eggs']
```

METHODS AND MODELS

Data Mining Model

Association rule mining, one of the most important and well researched techniques of data mining. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories.

Data Mining Task

In data science, association rules are used to find correlations and co-occurrences between data sets. They are ideally used to explain patterns in data from seemingly independent information repositories, such as relational databases and transactional databases. The act of using association rules is sometimes referred to as “association rule mining” or “mining associations”. We use FP-Growth Algorithm to find the association of different items in the given dataset.

Data Mining Algorithm

Algorithm for construction of FP-Tree :

Input: A transaction database DB and a minimum support threshold.

Output: FP-tree, the frequent-pattern tree of DB.

Method: The FP-tree is constructed as follows.

1. Scan the transaction database DB once. Collect F, the set of frequent items, and the support of each frequent item. Sort F in support-descending order as FList, the list of frequent items.
2. Create the root of an FP-tree, T, and label it as “null”. For each transaction Trans in DB do the following:

• Select the frequent items in Trans and sort them according to the order of FList. Let the sorted frequent-item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert tree([p | P], T).

• The function insert tree([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N 's count by 1; else create a new node N , with its count initialized to 1, its parent link linked to T , and its node-link linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert tree(P, N) recursively.

<u>Algorithm for FP-Growth:</u>

Input: A database DB, represented by FP-tree constructed according to Algorithm 1, and a minimum support
threshold ?.

Output: The complete set of frequent patterns.

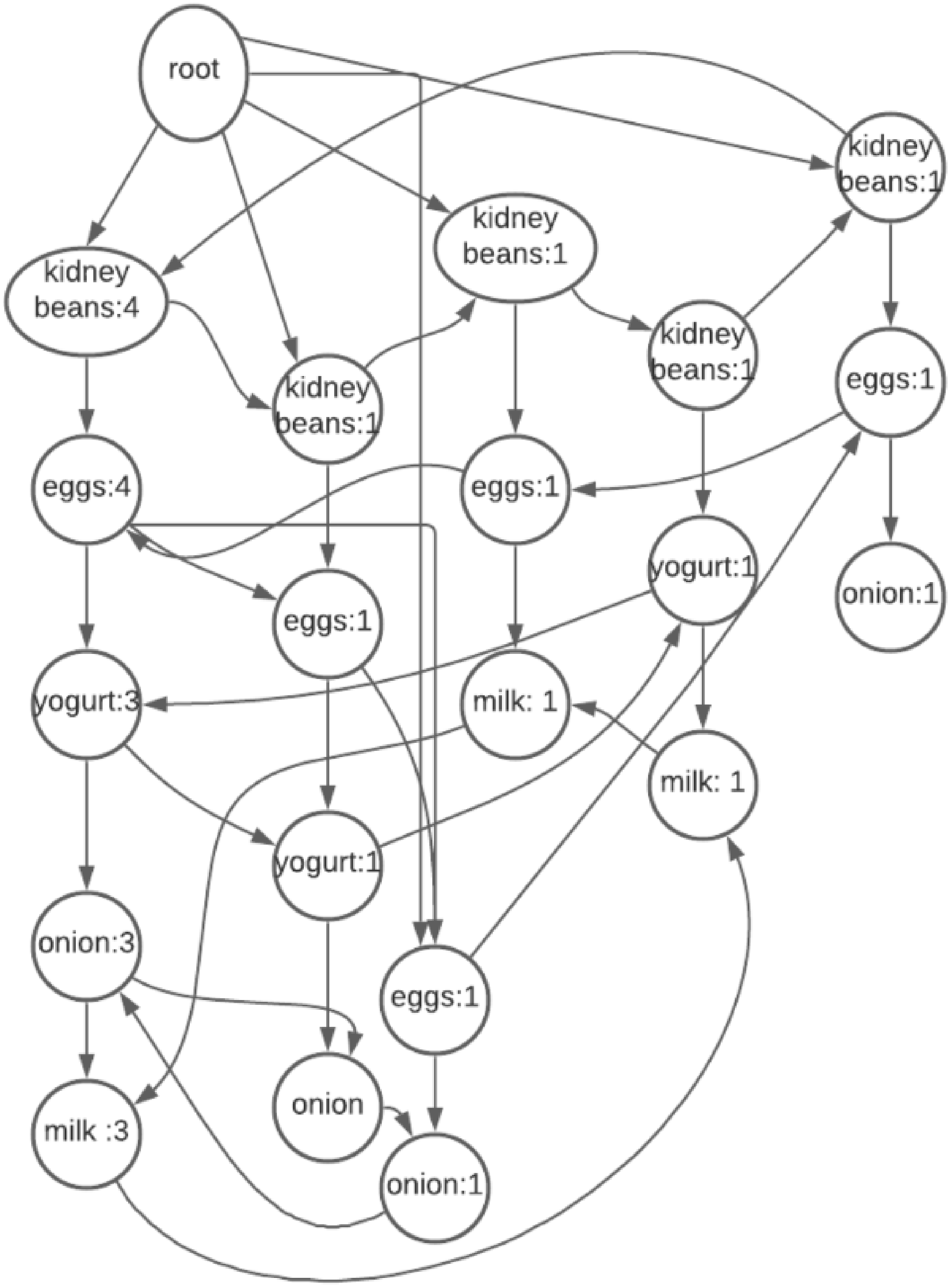
Method: call FP-growth(FP-tree, null).

```
Procedure FP-growth(Tree, a) {
(01) if Tree contains a single prefix path then // Mining single prefix-path FP-tree {
        (02) let P be the single prefix-path part of Tree;
        (03) let Q be the multipath part with the top branching node replaced by a null root;
        (04) for each combination (denoted as ß) of the nodes in the path P do
        (05) generate pattern ß ∪ a with support = minimum support of nodes in ß;
        (06) let freq pattern set(P) be the set of patterns so generated;
}
(07) else let Q be Tree;
(08) for each item ai in Q do { // Mining multipath FP-tree
        (09) generate pattern ß = ai ∪ a with support = ai .support;
        (10) construct ß's conditional pattern-base and then ß's conditional FP-tree Tree ß;
        (11) if Tree ß ≠ Ø then
                (12) call FP-growth(Tree ß , ß);
        (13) let freq pattern set(Q) be the set of patterns so generated;
}
(14) return(freq pattern set(P) ∪ freq pattern set(Q) ∪ (freq pattern set(P) × freq pattern set(Q)))
}
```

When the FP-tree contains a single prefix-path, the complete set of frequent patterns can be generated in three parts: the single prefix-path P, the multipath Q, and their combinations (lines 01 to 03 and 14). The resulting patterns for a single prefix path are the enumerations of its subpaths that have the minimum support (lines 04 to 06). Thereafter, the multipath Q is defined (line 03 or 07) and the resulting patterns from it are processed (lines 08 to 13). Finally, in line 14 the combined results are returned as the frequent patterns found.

MODEL EVALUATION AND DISCUSSION

The FP-Tree generated for the used dataset:



CONCLUSION

The FP growth algorithm largely outperforms both varieties of Apriori algorithm in the matter of time complexity in case of large data. This is easily explained, as in this algorithm the database scan has been minimized drastically. The space complexity achieved is also a plus-point of this algorithm, as the whole database is being compressed into a FP-tree and this leads to further reduction in data-handling time as efficient and time-tested tree-handling routines can be utilized.

REFERENCES

[1] Data Mining Concepts & Techniques, Jiawei Han, Micheline Kamber, Jian Pei-3rd Edition 2012

[2] https://en.wikipedia.org/wiki/Frequent_pattern_discovery

[3] https://www.geeksforgeeks.org/ml-frequent-pattern-growth-algorithm/

[4] https://www.geeksforgeeks.org/frozenset-in-python/

[5] https://pypi.org/project/tabulate/