An Improved Frequent Itemset Mining with FP Tree using Soft Set Theory

Santhosh Kottam, Varghese Paul

***Abstract---** Currently, data mining experts are using various algorithms for retrieving frequent itemsets. One of the exciting methods is frequent pattern growth that retrieves the whole set of frequent items by avoiding candidate set production. It has certain limitations. To overcome these limitations, we propose a new algorithm called FP-softset, it implements the application of a soft computing tool- Soft set theory. Our research work discusses applications of soft set theory for extracting frequent itemset from a large data set. First two sections of this paper cover introduction and preliminaries. In the remaining sections, the authors explain soft set theory and its uses for improving FP-growth algorithm. Finally, the paper ends with a new algorithm, which the authors developed as software using python programming. The proposed work we can use for data mining. It has enormous use in the fields of market basket analysis, agricultural, health, education, and many more.*

***Key words:** Soft Computing, Soft set, FP- Growth, FP-soft set, Association rule, frequent itemset, Apriori algorithm.*

I. INTRODUCTION

Data mining is the process of retrieving useful patterns representing knowledge, which are stored in huge databases and other gigantic data depository. Interesting patterns, we describe in different formats, known as data mining functionalities. Data mining processes have included a sequence of functionalities. Among them, association analysis is very popular and is getting more attention from industry, academia, and research scholars. The mining of rules helps us to bring out interesting relationships and to make more accurate decisions in different areas like industry, health, crime investigation, and many more. Generally, we discover frequent itemsets from huge data repositories. A good method for extracting frequent itemsets from a huge data set is Apriori algorithm. Prior knowledge of frequent itemsets properties is essential for implementing this algorithm.

Even though Apriori is a classical algorithm, it has two drawbacks. Firstly, it has to generate a big collection of candidate sets. Secondly, it has to inspect the dataset continually and verify a large set of candidate sets by pattern matching. Researchers have added active modifications to apriori algorithm for improving its performance and flexibility and finally generated a divide and conquer method, which is known as FP growth.

FP algorithm brings out frequent itemsset without candidate itemsets generation. This method directly compresses the databases into a particular tree, which is known as frequent pattern tree and generates the associative rule from it. It scans the database only two times and reduces the I/O load. Eventually, this method saves execution time and through which it increases performance. The generation of FP-growth algorithm requires a high volume of memory space. Scanning the dataset twice reduces the full efficiency of FP- growth algorithm and produces a large number of FP-trees.

A new soft computing tool for handling uncertainties in different decision making problems is soft set theory invented by Russian mathematician D. Molodtsov. Soft set theory provides sufficient parameterization techniques for managing with data uncertainty. In the soft set theory, the beginning picture of the object has a rough nature, and we do not have to bring in the idea of an exact solution [1]. We propose a new alternative to FP algorithm with the help of soft set theory. Conditional FP tree and frequent pattern creation are simplified by soft set theory.

II. PRELIMINARIES

A. Association Rules

Association rule extraction is one of the distinguished approaches of knowledge mining. It intends to retrieve interesting patterns and associations from a large data set [2]. Let R be a set of items, $R = \{r1, rs2 \ldots\ldots rn\}$ and B be a subsets of R. For $P \subset R$ and $Q \subset R$, then $b \in B$ accommodate P if and only if $P \subseteq b$. An association rule has the format, $P \to Q$, where $(|Q| = 1)$ and $P \cap Q = \emptyset$. Then set P is called the antecedent of the rule and set Q is called the consequent. Usually, a rule $P \to Q$ means that if an operation includes P it very likely includes Q as well.

There are two parameters related to a rule: Support and confidence. To describe these parameters, we use Bs and Bc to indicate the subset of B that contains both P and Q, and the subset of B that contains P, respectively. It is obvious $Bs \subseteq Bc \subseteq B$.

Definition: The support value of the rule P→ Q derived from dataset B is the ratio of the cardinality of Bs to the cardinality of B. Hence, the support of the rule is

$$S=\frac{|Bs|}{|B|}$$

Definition: The confidence value of the rule P → Q obtained from dataset B is the ratio of the cardinality of Bs to the cardinality of Bc. Therefore, the confidence of the rule is

$$C=\frac{|Bs|}{|Bc|}$$

Manuscript published on January 30, 2020.
* Correspondence Author
Santhosh Kottam*, Research Scholar, Research and Development Centre, Bharathiyar University, Coimbatore, Email: sankottam@gmail.com.
Dr. Varghese Paul, Professor & HOD, Department of Information Technology, Cochin University of Science & Technology, Thrikkakara, Kochi, Pincode: 682022, Email: vp.itcusat@gmail.com

Definition: The *lift* of a rule is specified as $\text{lift}(P \Rightarrow Q) = \frac{\text{SUP}(P \cup Q)}{\text{SUP}(P) \times \text{SUP}(Q)}$ or the ratio of the examined support to that expected if P and Q were independent.

B. Apriori Rules

Apriori is a conventional association rule fetching algorithm invented by R. Agarwal and R. Srikant in 1994. This algorithm employs level-wise search to mining frequent itemsets. It is used J^{th} level itemsets for determining $(J+1)^{th}$ level itemsets. Initially, the set of frequent items is decided by browsing the whole dataset to find the count for each item and gather those items that assure minimum support. Initially, the resulting set is represented as M1, next stage M1 is used to discover M2, which is used to discover M3 and so on, until no more frequent itemsets can be found. For each stage, a whole examination of the dataset is necessary. This will reduce the performance of the algorithm execution. To increase the performance of level-wise examination, we use apriori property to reduce search area. According to this property- all nonempty subsets of a frequent itemset is frequent. This algorithm uses the following steps for its process - join and prune[3].

Join action says that to create frequent itemset Mk, a set of candidate K-itemsets (CK) is created by combining the frequent itemsets Mk-1 with itself.

Prune action says that from the candidate K-itemsets; filter the candidates keep a count not less than minimum support. For reducing the complexity of this action, here we use apriori property.

C. FP Algorithm

Frequent pattern growth algorithm is an enhancement of apriori algorithm. It use divide and conquer method. A FP tree is a compressed representation of a dataset that helps in finding of frequent itemset without the identification of candidate itemset creation. The root of the frequent pattern tree is denoted as 'NULL' value. Remaining nodes stand for Item Name, Node link and Count. Nodes correspond to items and have a count value[3].

Development of frequent pattern tree uses two stages over the data set. In the first stage, it finds count for each item and removes non-frequent items. The set of frequent item is arranged in descending order based on their support count.

Example 1. Consider an example, where the items are p, q, r, s, t.

Table I-Transaction Data Set

TID	ITEMS
1	{p,q}
2	{q,r,s}
3	{p,r,s,t}
4	{p,s,t}
5	{p,q,r}
6	{p,q,r,s}
7	{p}
8	{p,q,r}
9	{p,q,s}
10	{q,r,t}

These items are sorted on their support count. Sorted order is L= {p, q, r, s, t}. The first phase is the construction of FP tree. Select the first transaction T1= {p, q}, arrange it into L order {p, q}. Construct first branch of the tree and set the count values of p and q to 1. Read the transactions 2, 3 ….10, and add the links to FP tree. Continue this process until all transactions are arranged to a path in the frequent pattern tree. The final output of the process is shown in Fig. 1

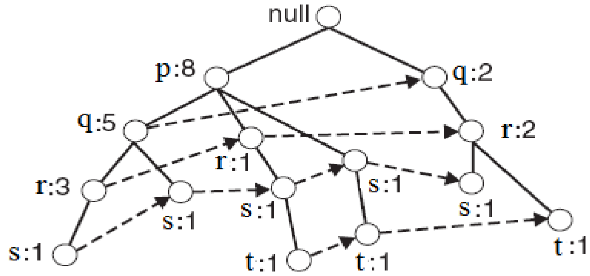


Fig. 1 Final constructed FP tree.

In second stage, frequent pattern growth fetches frequent itemsets from the frequent pattern tree. First extract prefix path sub tree ending in an item. Each prefix path sub tree processes repeatedly to extract the frequent itemsets. Results are then consolidated.

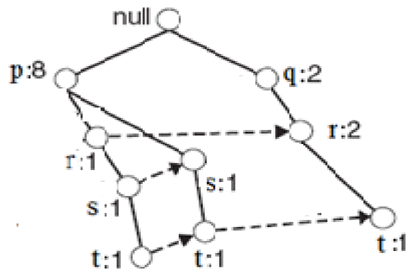


Fig. 2 Prefix path sub tree ending in t.

Using a divider and conquer approach, the prefix path sub tree for t will be used to extract frequent itemsets ending in 't', then in st, rt, qt and pt, then in rst, qst, pst etc. Let minSup=2 and extract all frequent itemsets containing't'. Count number of times't' repeat in the prefix path tree. If count is greater than or equal to 2, extract {t} as a frequent itemset. Here it is 3 and 't' is a frequent itemset. Next find frequent itemsets ending in't', ie st, rt, qt and pt. Decompose the problem recursively. To do this, we must first to obtain the conditional FP tree for't'. Revise the counts along the prefix paths from't' to reflect the number of transactions containing't'. q and r should be set to 1 and p to 2. Remove the nodes containing't', since information about node't' is no longer needed. Also remove infrequent items from the prefix path. Since q has a support of 1, remove it from conditional FP tree. FP tree conditional on t is given below.

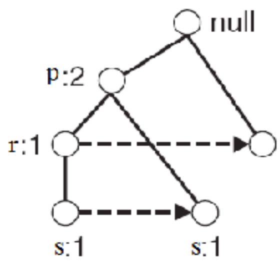


Fig. 3 FP tree conditional on't'.

Use the conditional FP-tree for't' to find frequent Itemsets ending in st, rt and pt.

For each of them (eg. st), find the prefix paths from the conditional tree for t, extract frequent itemsets, generate conditional FP-tree, etc... Recursively.
Example: t -> st -> pst ({s,t},{p,s,t}) are found to be frequent. Generate conditional FP tree for p, q, r, s and find all of its frequent itemsets. Final result is given in Table 2.

Table II- Frequent Itemsets Results

Suffix	Frequent Itemsets
t	{t},{s,t},{p,s,t},{r,t},{p,t}
s	{s},{r,s},{q,r,s},{p,r,s},{q,s},{p,q,s},{p,s}
r	{r},{q,r},{p,q,r},{p,r}
q	{q},{p,q}
p	{p}

III. SOFT SET

Soft computing is a group of techniques that works collectively to deal with real-world problems. It has reliable and flexible data execution capacity for handling problems having uncertainty, ambiguity, and approximate reasoning [4]. In the data mining industry following soft computing methodologies - genetic algorithms, neural networks, fuzzy sets, and rough sets - are widely used for retrieving relevant data from huge repositories. All these techniques have certain difficulties, as mentioned out in [1]. Most of the cases, causes for these limitations an insufficiency of the parameterization tool of these theories. Famous Russian mathematician, Molodtsov introduced a new mathematical concept called soft set theory. It has the capability to dealing uncertainties and free from the above difficulties.

A. Definition and Preliminaries

Definition (Soft Set)

A pair (G, X) is called a soft set over S, where G is a mapping given by

$$G : X \to P(S)$$

In other words, a soft set over S is a parameterized family of subsets of the universe S. Every set $G(\varepsilon), \varepsilon \in X$ may be treated as the set of ε - elements of the soft set (G, X). A soft set is not a set.
Soft set theory mainly differs from conventional mathematics on item managing. In conventional mathematics, we develop a mathematical model of an item and discover the idea of the precise solution of this model. In soft set theory, it allows using an alternate method – develops an idea of the approximate solution and determine that solution. This increases the popularity of soft theory. Soft set theory permits parameterization in different ways. We can use numbers, words, sentences, functions and many more as parameters in our real-world problems[5].

IV. SOFT SET THEORY FOR IMPROVING FP GROWTH ALGORITHM

Here, we discuss the role of soft set theory for increasing the performance of FP growth algorithm. The pre-requirement for applying soft set method for the proposed work is data that have to be converted into a soft set, where each element is treated as a parameter.

Example 2 The soft set describing Table 1 is given here. Soft set (F, X) illustrates the occurrence of each element in different dealings.

Let Y denote set of all transactions.
X is the set of elements. Each element is an item.
Y={Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,Y9,Y10} and
X= {x1, x2, x3, x4, x5}
Where
x_1 represents for the element ‘p’
x_2 represents for the element ‘q’
x_3 represents for the element ‘r’
x_4 represents for the element ‘s’
x_5 represents for the element ‘t’

Here
F(x_1) = { Y1,Y3,Y4,Y5,Y6,Y7,Y8,Y9}
F (x_2) = { Y1,Y2,Y5,Y6,Y8,Y9,Y10}
F (x_3) = { Y2,Y3,Y5,,Y6,Y8,Y10}
F (x_4) = { Y2,Y4,Y6,Y9}
F (x_5) = {Y3, Y4, Y10}}

The soft set (F, X) is a parameterized family {F (x_i), i=1, 2, 3……, 5} of subsets of the set Y and gives us a collection of rough information of an item. F (x_1) means “item (p)” whose functional value is the set {Y1, Y3, Y4, Y5, Y6, Y7, Y8, Y9}. The soft set (F, X) is a collection of approximation as below:

$$(F,X)=\left\{\begin{array}{l} p= \{Y1, Y3, Y4, Y5, Y6, Y7, Y8, Y9\}, \\ q= \{Y1, Y2, Y5, Y6, Y8, Y9, Y10\}, \\ r= \{Y2, Y3, Y5, Y6, Y8, Y10\}, \\ s= \{Y2, Y4, Y6, Y9\}, t= \{Y3, Y4, Y10\} \end{array}\right\}$$

Fig. 4. The soft set representing Table

A pictorial representation of soft set (F, X) is given in Table 4.

Table III-Tabular Representation Of The Soft Set

Item	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10
p	1	0	1	1	1	1	1	1	1	0
q	1	1	0	0	1	1	0	1	1	1
r	0	1	1	0	1	1	0	1	0	1
s	0	1	0	1	0	1	0	0	1	0
t	0	0	1	1	0	0	0	0	0	1

From the soft set (F, X), representation of different transactions is given below.
Y1 = {p,q}
Y2 = {q,r,s}
Y3 = {p,r,s,t}
Y4 = {p,s,t}
Y5 = {p,q,r}
Y6 = {p,q,r,s}
Y7 = {p}
Y8 = {p,q,r}
Y9 = {p,q,s}
Y10 = {q,r,t}
Fig. 5. The involvement of items in different transaction.

Definition
Let (F, X) be a soft set over the universal set Y and E $\subseteq$ *X. The support count of an element E is represented by sup(E). i.e* sup(E) is number of transactions Y containing the element E.

Sup(E)= |{u: E$\subseteq$u}|, here |E| support count of E.

From the above definition supported count collected for different frequent itemsets are given below.

sup{p}=|{Y1,Y3,Y4,Y5,Y6,Y7,Y8,Y9}|=8
sup{q}=|{ Y1, Y2, Y5, Y6, Y8, Y9, Y10}|=7
sup{r}=|{ Y2, Y3, Y5, Y6, Y8, Y10}|=6
sup{s}=|{ Y2, Y4, Y6, Y9}|=4
sup{t}=|{ Y3, Y4, Y10}|
sup{p, q}=|{Y1,Y5,Y6,Y8,Y9}|=5
sup{r, s}=|{Y2,Y3,Y6}|=3
sup{s, t}=|{Y3,Y4}|=2
sup{q, r}=|{Y2,Y5,Y6,Y8,Y10}|=5
sup{q, s}= |{Y2,Y6,Y9}|=3
sup{r, t}=|{Y3,Y10}|=2
sup{p, r}=|{Y3,Y5,Y6,Y8}|=4
sup{p, s}=|{Y3,Y4,Y6,Y9}|=4
sup{p, t}=|{Y3,Y4}|=2
sup{q, r, s}=|{Y2,Y6}|=2
sup{p, s, t}=|{Y3,Y4}|=2
sup{p, r, s}=|{Y3,Y6}|=2
sup{p, q, s}=|{Y6,Y9}|=2
sup{p, q, r}=|{Y5,Y6,Y8}|=3

Fig.6. The supported sets.

Next, we see, how this soft set theory is useful for improving the FP growth algorithm. The whole process starts with organizing items in the downward order of support count. The result will be *L*={p, q, r, s, t}. Assume minimum support count for frequent itemset as 2.

FP growth algorithm extracts association rules without generating the candidate set. It passes through the following steps

1) FP-Tree construction
2) Prefix subpath generation
3) Generate conditional FP Tree
4) Frequent Itemset Generation

In the proposed approach, the first two steps remain the same. For the remaining steps, we apply the application of soft set theory. The conditional FP tree and frequent itemset mining steps are carried out by the following method. This method is performed on all items of the FP-Tree.

We first consider prefix path sub tree for‘t’, which is the last item in *L*, rather than the first. ‘t’ occurs in three branches of the prefix path subtree.

F (t) = {Y3, Y4, Y10}

Where Y3 = {p, r, s, t}, Y4 = {p, s, t} and Y10 = {q, r, t}

If Y3$\subseteq$Y4 or Y3$\supseteq$Y4
Y3∩Y4= {p, r, s, t} ∩ {p, s, t} = {p, s, t}

Which indicates {p, s, t} is repeated two times in the transactions {Y3, Y4}.

ie, {p,s,t:2} is 3-frequent itemset

Next we use apriori property in the new process. According to this theory all nonempty subsets of a frequent itemset is frequent. Use this property to find all subset, which are included the item‘t’. Resultant subsets are also frequent.

{p, t:2},{s,t:2} are 2- frequent itemsets

Next Y3 and Y10, if Y3$\subseteq$Y10 or Y3$\supseteq$Y10
Y3∩Y10= {p, r, s, t} ∩{r, t} = {r, t}

{r, t:2} is a 2-frequent itemset

Next combination is Y4 and Y10
If Y4$\subseteq$Y10 or Y4$\supseteq$Y10
Y4∩Y10= {p, s, t} ∩ {q, r, t} = {t}

{t:2} is 1-frequent itemset, which is proved in Fig. 6.

Frequent itemsets ending with‘t’ are
{t},{p,t:2},{s,t:2},{r,t:2} and {p,s,t:2}

Apply this same method for remaining items- s, r and q. Final frequent itemset results are the same as Table 5.

Table IV-Frequent Itemsets Results Using Soft Set Theory

Item	Frequent Itemsets
t	{t},{s, t},{p, s, t},{r, t},{p, t}
s	{s},{r, s},{q, r, s},{p, r, s},{q, s},{p, q, s},{p, s}
r	{r},{q, r},{p, q, r},{p, r}
q	{q},{p, q}
p	{p}

The algorithm for mining frequent itemsets using soft set theory is given in Fig. 7.

Algorithm:Fp-softset. Extract frequent itemsets using a Frequent pattern tree by soft set approach.
Input:
D, be a dataset
M_sup, the minimum support count.
Output:
Complete set of frequent items
Method:
The Frequent Pattern tree FPT is produced in the following phases:
Phase I

- Examine the dataset *D* once and find support count of each element.
- Remove item which is not keeping minimum support count.
- Use support count and arrange frequent items F, in descending order.

Phase II

- For all transactions sort frequent items according to the order of L.
- Read transactions Y1, Y2, Y3….continue until all transactions are mapped to a path in the Frequent Pattern tree.
- Call Fp_soft(FPT)

Procedure Fp_soft(FPT)

Convert Fp-tree into a soft set representation.

for each frequent item $e_i \in F$, $i=1,2....n$

{

Extract prefix path sub tree PT and process it iteratively to generate the frequent itemsets.

From the prefix path sub tree PT, find F (*ei*), Result will be a set of transactions and subset of (F, E).

For each transaction *tk* in F (*ei*), *k=1,2...sup.count(F(ei))*

For each transaction *tl* in F (*ei*), *l=k+1,...sup.count(F(ei))*

{

If $tk \subseteq tl$ *or* $tl \supseteq tk$

$C=tk \cap tl$

If $C \neq$ NULL, *C* is a frequent item set.

}

Apply apriori property on *C* and generate all subsets, which are included the item *ei*.

}

Fig.7 Fp-soft set algorithm.

V. EXPERIMENT RESULT

In this part, we evaluate the new FP-Softset frequent itemset mining method with two conventional algorithms- Apriori and FP growth. The proposed approach is implemented in the data sets [6] and [7]. The proposed approach is coded in python programming language. All of the algorithms are implemented consecutively on a processor Intel Core i5-6200U CPU. The RAM is 4 GB and the OS is Windows 7 Professional.

A. Grocery store dataset

Grocery data set is collected from the data repository Kaggle. It contains 11 items and twenty transactions. Column headings are- jam, maggi, sugar, coffee, cheese, tea, bournvita, cornflakes, bread, biscuit and milk [6]. Here we performed frequent itemset mining using the Apriori, FP and proposed FP-softset algorithms. This contains searching for sequences of items that frequently happen in different transactions in order to understand a customer's purchasing behaviour. The procedure for generating useful information is given below.

Table V-Transaction Dataset For Grocery Basket Analysis

TID	ITEM	TID	ITEM
1	Milk, bread, biscuit	11	Biscuit, coffee, cock, cornflakes
2	Milk, bread, biscuit, cornflakes	12	Biscuit, coffee, cock, cornflakes
3	Bread, tea, bournvita	13	Coffee, suger, bournvita
4	Milk, bread, jam, maggi	14	Bread, coffee, cock
5	Biscuit, tea, maggi	15	Bread, biscuit, suger
6	Bread, tea, bournvita	16	Coffee, suger, cornflakes
7	Tea, magi, cornflakes	17	Bread, suger, bournvita
8	Bread, biscuit, tea, maggi	18	Bread, coffee, suger
9	Bread, tea, jam, maggi	19	Bread, coffee, suger
10	Milk, bread	20	Milk, tea, coffee, cornflakes

Assume minimum support count as two and Y denotes set of all transactions. The soft set representing supported data sets are

(F, E)= { milk= {Y1, Y2, Y4, Y10, Y20}, bread = {Y1, Y2, Y3, Y4, Y6, Y8, Y9, Y10, Y14, Y15, Y17, Y18, Y19}, biscuit= { Y1, Y2, Y5, Y8 ,Y11, Y12, Y15}, Cornflakes= {Y2, Y7, Y11, Y12, Y16,Y20},tea={Y3,Y5,Y6,Y7,Y8,Y9,Y20}, bournvita={Y3,Y6,Y13,Y17}, jam= {Y4, Y9}, cock= {Y11, Y12, Y14}, maggi= {Y4, Y5, Y7, Y8, Y9}, Coffee= {Y11, Y12, Y13, Y14, Y16, Y18, Y19, Y20}, Suger= {Y13, Y15, Y16, Y17, Y18, Y19} }

Fig. 8. The soft set representing Table

Construct FP tree and prefix sub path for all items present in the FP tree. Apply proposed approach on conditional FP tree and frequent itemset generation. We will get following frequent itemset.

Table VI-Frequent Itemset Results Using Soft Set Theory

Item	Frequent Itemsets
biscuit	{biscuit},{biscut,bread},{biscuit,coffee},…….{biscuit,maggi,tea},{biscut,cock,coffee,cornflakes}
coffee	{coffee},{coffee,bread},{coffee,cock}…….{coffee, bread, suger},{coffee, cock, cornflakes}
. . .	. . .
Cock	{cock}, {cock, cornflakes}
jam	{jam},{jam,bread},{jam,maggi}, {jam,magi,bread}

The same set of frequent items are derived from both apriori and FP algorithm methods. The execution time of our proposed approach-FP-softset- through this data set is .00399 s. The comparison of the response time of traditional apriori, FP growth , and proposed FP-soft set-based frequent itemset generation approaches is given in the following Fig 9.

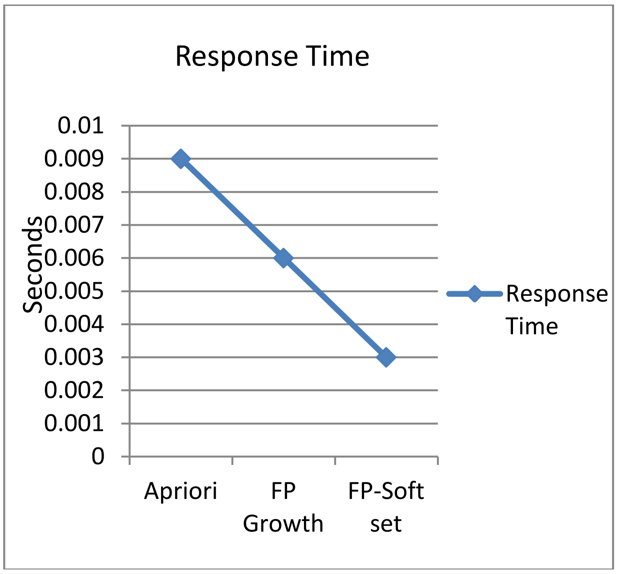


Fig. 9. The comparison of executing time.

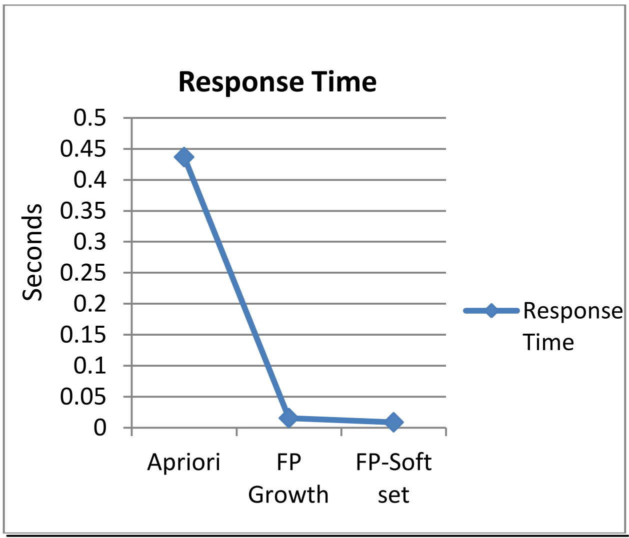
From the above figure 9, the response time improvement of FP-Softset with other two methods is given below.

Table VII-Response Time Improvement Of Fp-Softset

Method	Response time improvement of FP--Softset
Apriori	66.67%
FP-Method	50.00%

B. Student performance data set

This data is related to student achievement in secondary education of one Portuguese school and is collected from UCI repository. The data set contain relevant details of students for determining their performance. Dataset is contributed regarding the performance of the subject: Mathematics [7]. It consists of 33 attributes, 369 instances and used in the implementation of the above three frequent itemset generation algorithms. Response time of FP-soft set is better than the other two methods. The comparison result is given below in figure 10.



In the second dataset, the response time of FP-Softset is significantly improved in comparison with other two traditional methods. The execution time of FP-Softset is .00875s. Its response time improvement is given below.

Table VIII-Response Time Improvement Of FP-Softset

Method	Response time improvement of FP--Softset
Apriori	97.99%
FP-Method	43.58%

In both experiments, FP-Sofset method produced a good response time.

VI. CONCLUSION AND FUTURE WORK

Soft computing methods are playing a siginificant role in knowledge discovery process. Different soft computing methods have emerged with reliable performance and accuracy. Examples are – Fuzzy set, Rough set, and Soft set. Among them, soft set is a powerful one and it has wide range applications in data mining industry. We envisaged the possibilities of soft set theory in frequent itemset mining.
In our research work, firstly, we presented two well-known techniques for retrieving frequent itemset from data sets - Apriori and FP-growth. We brought out the demerits of these methods. Frequent pattern growth is a unique method for finding frequent itemset. It has to go to different steps for completing its process. These steps are increasing memory utilization FP- growth. This drawback we experienced in our implementation. To overcome this limitation, we have implemented the proposed algorithm FP--Softset, it could to reduce the overhead of FP-growth method. Apriori, FP-growth, and FP-Softset algorithms are implemented in Python language, using the data sets grocery and student performance. At the end of the implementation, we compared the result of FP-softset with results obtained from apriori and FP-Growth. In different experiments, FP-Softset produced a good response time.

In future, we would like to examine opportunities of soft set theory in different data mining functionalities like classification and clustering.

REFERENCES

1. D. Molodtsov," Soft set theory- First results", An International Journal of Computers & mathematics with applications", Elsevier, Volume 37, Issue 4-5, Page 19-31, February-March1999.
2. Qiankun Zhao, Sourav S. Bhowmick, "Association Rule Mining: A Survey", Technical Report, CAIS, Nanyang Technological University, Singapore, No. 2003116 , 2003.
3. Jiawei Han, Micheline Kamber, "Data Mining: Concept and Techniques", Elsevier, second edition, 2006.
4. Sushmita Mitra, Sankar K. Pal, Pabitra Mitra "Data Mining in Soft Computing Framework: A Survey", ieee transactions on neural networks, vol. 13, no. 1, january 2002.
5. Santhosh kottam, Varghese paul "uncertain data handling in data mining using soft set theory" international journal of applied engineering research, issn 0973-4562 vol. 10 no.73 2015
6. GROCERY STORE DATA SET, HTTPS:// WWW.KAGGLE.COM/ shazadudwadia/supermarket
7. Student Performance Data Set, https://archive.ics.uci.edu/ml/datasets/student+performance.

AUTHORS PROFILE

Mr. Santhosh Kottam, completed his Master of Computer Applications (MCA) from Madras University, Tamilnadu and B.Sc mathematics degree from M G University, Kerala. He is currently pursuing PhD in Data Mining at Bharatiyar University, Coimbatore. His research area is data mining and has more than 19 years of teaching experience, which includes UG and PG. He has been serving Federal Institute of Science and Technology (FISAT), Angamaly, Kerala as HOD & Assistant Professor (Senior Grade) in the Department of Computer Applications since May 2008. During this period of time, he has taught many subjects including Programming Languages, System Analysis and Design, Operating Systems, Object Oriented Modeling and design and Data Mining. He has published research papers in the International Journals, National and International Conferences.

Dr. Varghese Paul, is completed B.Sc (Engg) in Electrical Engineering from Kerala University, M.Tech in Electronics and Ph.D in Computer Science from Cochin University of Science and Technology. Research Supervisor of Cochin University of Science and Technology, M G University Kottayam, Anna Technical University Chennai, Bharathiar University Coimbatore, Bharathidasan University Trichy and Karpagam University Coimbatore. Under the guidance, 29 research scholars had already completed research studies and degree awarded. Research areas are Data Security using Cryptography, Data Compression, Data Mining, Image Processing and E_Governance. Developed TDMRC Coding System for character representation in computer systems and encryption system using this unique coding system. Published many research papers in international as well as national journals and a text book also. Earlier worked as Industrial Engineer with O/E/N India Ltd Cochin, Communication Engineer with KSE Board, SCADA Engineer in Saudi Electricity Department, Head of IT Department CUSAT and Dean (CS, IT and Research) in Toc H Institute of Science and Technology. Certified Software Test Manager, Ministry of Information Technology, Govt of India. Life Member, Information System Audit and Control Association, USA (ISACA), Indian Society for Technical Education, India (ISTE) amd National Geographic Society, USA.