International Journal of Hybrid Information Technology
Vol.7, No.3 (2014), pp.155-166
http://dx.doi.org/10.14257/ijhit.2014.7.3.16

MS-FP-Growth: A multi-support Vrsion of FP-Growth Agorithm

Wiem Taktak[1] and Yahya Slimani[2]

[1]*Computer Sc. Dept, Higher Institute of Arts MultiMedia (ISAMM), University Manouba 2010, Manouba, Tunisia.*
[2]*Computer Sc. Dept, Higher Institute of Arts MultiMedia (ISAMM), University Manouba 2010, Manouba, Tunisia.*
[1]*Wayma2007@gmail.com*, [2]*Yahya.slimani@gmail.com*

Abstract

In this paper, we propose a new version of FP-Growth algorithm to find association rules. In this version, we vary the minsup value from one level to another. This variation is made in two cases: increasing and decreasing the minsup value. We performed a set of experiments to validate the usefulness of our proposition in the generation of association rules process.

Keywords: *Multi-support, FP-Growth, by level*

1. Introduction

Association rule mining represents one of the most important and well techniques in the data mining field, was first introduced by Agrawal et al. [1]. Its main objective is to extract interesting correlations or associations among sets of items in transaction databases. Due to their importance in information systems, association rules are widely used in various areas such as market, finance, telecommunication networks, etc.

Apriori algorithm [1] is one of the most significant approaches to find association rules from datasets. This algorithm generates association rules from so-called frequent itemsets that are itemsets that satisfy a statistical measure, called minimum support (minsup). This minsup is the core of any association rule algorithm, and it is used to distinguish between frequent and unfrequent itemsets, essentially to limit the number of generated association rules. While the minsup allows to reach this objective, using only one minsup value for all items of a given database is problematic. In fact, with only one minsup value, an association rule mining algorithm assumes that all items (or attributes) of a dataset have the same frequency. This assumption is not totally true in real applications, where datasets contain some items of a high frequency, while other ones are of a low frequency. Both items with high and low frequencies are important to distinguish between frequent and rare itemsets. To overcome the constraint of one minsup, data mining algorithms have been adapted to mine association rules with multiple minimum supports.

The main objective of these algorithms is to produce interesting knowledge from low level to high level. In this context, we propose to introduce the multi-support approach in the FP-Growth algorithm [2]. The remainder of this paper is organized as follows: Section 2 reviews related works. Section 3 reminds key notions used in this paper. Our proposed multi-support version of FP-Growth algorithm is described in Section 4. Section 5 discusses the efficiency of our approach through some experimentation. Section 6 concludes the paper and lists some perspectives for future researches.

2. Related works

In the literature, Apriori algorithm and its variants have been proposed for generating association rules using frequent itemsets. This generation is based on two categories of

ISSN: 1738-9968 IJHIT

itemsets: the candidate and the frequent itemsets. A candidate itemset is a potentially frequent itemset (its support was not still computed), while a frequent itemset is an itemset whose support is greater than the minimum support minsup. All variants of Apriori algorithm use only one single minsup value for all items to find frequent itemsets. But, they do not allow finding the so-called rare association rules [3] that are likely to be of great interest in some data mining applications. As examples of these applications, we can mention identifying relatively rare diseases, predicting telecommunication equipment failure, finding associations between infrequently purchased supermarket items, and so on. Though the variants of Apriori algorithm are theoretically expected to be capable of finding rare association rules, they actually become intractable if the minimum support is set low. Rare association rules have very low support and high confidence in contrast to general association rules which have high support and high confidence. Mining of rare association rules with a uniform minimal support for all itemsets of a dataset may cause the well know rare item problem dilemma. This problem can be illustrated as follows: if the minsup is high, frequent itemsets involving rare items are missed because their support is less than the minsup. To find frequent itemsets involving rare items, the minsup value should be fixed at low value. As a result, the number of frequent itemsets explodes. To improve the performance of generating association rule algorithms involving rare items, efforts are being made to propose new approaches to mine rare association rules [3, 4, 5, 6, and 7]. The first approach, known as Multiple Support Apriori (MSApriori), has been proposed in [8]. In this approach, the minimum support of an association rule is defined in terms of minimum item supports (MIS) of the items that appear in the rule. Hence, each item in a dataset can have a minimum item support defined by the user. By providing different minimum item supports, the user can express different support requirements for different rules.

In the rest of this section, we focus on works based on FP-Growth algorithm and multiple minimum supports. In the field of data mining, FP-Growth algorithm is widely recognized more efficient in term of performance than Apriori algorithm, because it scans the database only twice compared with Apriori algorithm that scans the database N twice, where N is the number of transactions in the database.

Multiple minimum support approaches can be classified into two categories. The first one varies the value of minsup from one item to another [8, 9]. For example, in [8], authors consider that one itemset is frequent, if each item belonging to this itemset satisfies its minsup. This approach is called MIS (Minimum Itemset Support). Subsequently, in [9] authors give another definition of frequent itemsets to reduce their number without losing information. The second category varies the value of minsup when moving from one level to another [10]. Approaches for mining association rules at multiple concept levels, allow not only discover rules at different levels but also have high potential to find non trivial informative association rules because of its flexibility at putting the attention to different datasets and applying different supports at different levels.

3. Background

In this section, we briefly remind some notions that we use in the rest of the paper.

Definition 1: A context of extraction is a triplet K = (O; I; R), such that O and I are two finite sets of, respectively, objects and items (or attributes), and $R \subseteq O \times I$ is a binary relation between objects and items. A couple $(o; i) \in R$ means that an object $o \in O$ contains the item $i \in I$, oRi.

Example 1: We consider the following context of extraction.

Table 1. Context of Extraction

	A	B	C	D	E	F
T1	X	X				
T2	X		X	X		
T3			X	X	X	
T4				X	X	X
T5	X	X	X	X	X	
T6	X	X	X			

With this context, we study two cases: the case where values of minsup are assigned in increasing order, and the case where the values are assigned in a decreasing order. For the first case (increasing), we obtain the following results:

1. For level 1, we x the minsup1 value at 0.5; we obtain the following frequent 1-itemsets with their respective supports: A(4), B(3), C(4), D(4) and E(3).
2. For the second level, we x the minsup2 at 0.3; with this new value, we obtain the following results: AB(3), AC(3), AD(2), BC(2), CD(3), CE(2) and DE(3).
3. For level 3, we obtain the following frequent itemsets, when we assign the value 0.2 to minsup3: ABC(2), ACD(2) and CDE(2).
4. Finally, we x minsup4 to 0.1 for level 4 and we find that the set of frequent 4-itemsets is empty.

Now, we consider the second case (decreasing order). We have:

1. For level 1 and minsup1=0.1, we obtain the following frequent 1-itemsets and their respective supports: A(4), B(3), C(4), D(4), E(3) and F(1).
2. For level 2 and minsup2=0.2, we have: AB(3),AC(3), AD(2), BC(2), CD(3), CE(2) and DE(3).
3. For level 3 and minsup3=0.3, we obtain the following frequent 3-itemsets with their respective supports: ABC(2), ACD(2) and CDE(2).
4. For level 4, the set of frequent 4-itemsets is empty.

Table 2. Comparative Table between Mono and Multi Supports

XP number	Level	Mono support		Multi support	
		Minsup	Itemsets	Minsup	Itemsets
1	1	0.50	A(4), B(3), C(4), D(4) and E(3)	0.50	A(4), B(3), C(4), D(4) and E(3)
	2	0.50	AB(3), AC(3), AD(2), CD(3), and DE(3)	0.30	AB(3), AC(3), AD(2), BC(2), CD(3), CE(2) and DE(3)
	3	0.50	Empty set	0.20	ABC(2), ACD(2) and CDE(2)
	4	0.50	Empty set	0.10	Empty set

2	1	0.10	A(4), B(3), C(4), D(4), E(3) and F(1)	0.10	A(4), B(3), C(4), D(4), E(3) and F(1)
	2	0.10	AB(3), AC(3), AD(2), BC(2), BD(1), BE(1), CD(3), CE(2), DE(3), DF(1) and EF(1)	0.20	AB(3), AC(3), AD(2), BC(2), CD(3), CE(2) and DE(3)
	3	0.10	ABC(2), ABD(1), ACD(2), BCD(1), BCE(1), BDE(1), CDE(2) and DEF(1)	0.30	ABC(2), ACD(2) and CDE(2)
	4	0.10	ABCD(1) and BCDE(1)	0.50	Empty set

4. Proposed Approach

In this section, we propose a new variant of FP-Growth algorithm in which we introduce multi-supports at different levels. This variant is applied in three cases:

1. In the first case, we increase the minsup value from one level to another.
2. In the second case, we decrease the minsup value between levels.
3. In the last and third case, we vary the minsup value randomly (increasing and decreasing).

For each case, the minsup value will be fixed according to three methods: (i) by the user (U); (ii) by the user assisted by a function (UF); (iii) automatically by a function (A). The overall structure of our proposed variant is defined as follows:

Algorithm 1: Algorithm for defining multi-supports

Data: A transaction database DB and the choice of the user (U for User only, UF for User assisted by a Function, and A for automatically).

Result: MS: Set of minimum supports.

MS=∅;

If Choice=='U' **then**

for i:=1 to | largest − itemset | step 1 **do**

MS = MS ∪ minsupi ;

Else

If Choice=='UF' **then**

//*Uniform variation of the minsup from one level to another*//

MS = MS ∪ minsup1;

α = (100 − minsup1)=(| largest − itemset | −1) mod 10;

for i:=1 to | largest − itemset | step 1 **do**

minsupi = minsupi 1 + α;

MS = MS ∪ minsup i;

Else

//*Non-uniform variation of the minsup*//

MS = MS ∪ minsup1;

for i:=1 to | largest − itemset | step 1 **do**

αi = Random((100 − minsupi 1)=(| largest − itemset | −1)) mod 10;

minsupi = minsupi 1 + αi ;

End

We vary the value of the minsup from one level to another according to three cases:

1. In the first case, the variation is done according to the length of the largest itemset in the database; hence, the size of the MS set is equal to the length of the largest itemset in the database.
2. In the second case, the user fixes the first value of the MS set; then, the subsequent values are determined by dividing or multiplying the first value on or by the length of the current itemset.
3. The third case is identical to the second, except that the first value of MS set (first minsup) is computed randomly.

Algorithm 2 describes in a pseudocode form the steps you would use to find frequent itemsets with modified version of FP-Growth algorithm.

Algorithm 2: FP-Growth with multi-supports

Data: FP-tree constructed, using DB and a set minimum support threshold MS for each item.

Result: The complete set of frequent patterns.

Method: Call FP-Growth (FP-tree; null), which is implemented as follows.

Procedure FP-Growth (Tree; α)

If Tree contains a single path P **then**

Foreach combination (denoted as β) of the nodes in the path P **do**

support = minimum support of nodes in β;

generate pattern β ∪ α with and support ≥ $MS_{length(\beta)}$;

End

Else

Foreach α_i in the header of Tree **do**

β= α_i ∪ α;

support = α_i:support ;

generate pattern β with and support ≥ $MS_{length(\beta)}$;

construct β′ s conditional pattern base and then β′ s conditional FP-tree $Tree_\beta$;

If $Tree_\beta \neq \emptyset$ **then**

call FP-Growth ($Tree_\beta$, β, MS);

End

End

End

5. Experimental results

The aim of this section is to study the impact of multi-supports on number of Association Rules (AR) generated. We conducted our experiments on Nursery benchmark database from http://archive.ics.uci.edu/ml/datasets.html . This database is composed of 8 attributes and 12960 records. In the rest of this section, we present and discuss the results obtained for different variations of the minsup.

5.1 Increasing minsup variant

In this variant, the user is helped by a function to determine the minsup values. The user fixes the first minsup value, and then a function computes the subsequent minsup values. This function computes a given minsup as follows: minsupi = $minsup_{i-1}$ + increment. The term increment is either constant (uniform for all levels) or random (non-uniform for all levels). For the uniform case, the increment is defined by the following formula: $(100 - \epsilon - minsup_1) \div (|\text{ largestitemset }| - 1) \bmod 10$. Here, we used ϵ in order to avoid cases with minsup values equal to 0 and 100. For the non-uniform case, the increment is fixed as follows: Random $(100 - \epsilon - minsup_1) \div (|\text{ largestitemset }| - 1)$ mod 10. Using these assumptions and fixing ϵ to 0.5, we obtain the following results (see Tables 3 and 4).

Table 3. Number of Association Rules Generated when Increasing the Minsup Value with Constant Value

Minsup 1	Minsup 2	Minsup 3	Minsup 4	Minsup 5	Minsup 6	Minsup 7	Minsup 8	AR
10	12.78	15.57	18.36	21.14	23.93	26.71	29.50	48
9	11.93	14.86	17.78	20.71	23.64	26.57	29.50	80
8	11.07	14.14	17.21	20.28	23.36	26.43	29.50	188
7	10.21	13.42	16.64	19.86	23.07	26.14	29.50	188
6	9.35	12.71	16.07	19.43	22.78	26.14	29.50	208
5	8.50	12.00	15.50	19.00	22.50	26.00	29.50	208
4	7.64	11.28	14.93	18.57	22.21	25.86	29.50	400
3	6.78	10.57	14.36	18.14	21.93	25.71	29.50	40

								0
2	5.93	9.86	13.78	17.71	21.64	25.57	29.50	552
1	5.07	9.14	13.21	17.28	21.36	25.43	29.50	632

Table 4. Number of Association Rules Generated when Increasing the Minsup Value with Random Value

Minsup 1	Minsup 2	Minsup 3	Minsup 4	Minsup 5	Minsup 6	Minsup 7	Minsup 8	AR
15	16.04	21.47	27.42	32.46	37.32	41.82	45.89	46
14	15.21	20.87	26.30	31.37	36.28	40.73	45.09	48
13	14.17	20.52	26.42	31.63	36.19	40.48	44.55	48
12	13.30	19.88	25.94	30.78	35.54	40.29	44.76	48
11	12.38	18.55	24.31	29.79	34.91	39.41	43.59	69
10	11.41	18.19	24.40	29.63	35.02	39.56	43.84	143
9	10.44	17.01	22.72	28.60	33.79	38.43	42.64	199
8	9.53	15.25	20.74	26.36	31.70	36.29	40.82	207
7	8.58	14.76	20.69	26.01	31.04	35.81	40.40	308
6	7.71	13.25	19.40	25.46	30.21	35.59	40.42	379
5	6.85	13.27	19.58	25.30	29.98	34.89	39.54	509
4	5.54	11.96	18.03	23.78	28.91	33.83	38.45	542
3	5.08	11.59	17.92	23.42	28.38	33.72	38.17	687
2	3.67	10.83	17.17	23.07	28.64	33.60	38.44	696
1	2.52	9.82	16.09	21.23	26.77	31.80	36.54	903

Tables presented above show different minsup with the corresponding number of association rules. The minsup1 values vary from 10 to 1 in the uniform case and from 15 to 1 in the non-uniform case. We remark that if we exceed the value 10 for minsup1 in the uniform case and 15 in the non-uniform case, we cannot obtain association rules.

During the generation process of rule associations, the minsup value is increasing at each level and the itemset supports are decreasing mutually. The number of association rules is inversely reciprocal to the value of minsup. Also, we must mention that the number of generated association rules is sensitive to the value update between minsupi and minsupi+1 (i ∈ {1::7}). So, according to results of Table 3, we have generated 48 valid association rules for minsup1 = 10. Conversly, we we have generated 143 valid association rules for the same value of minsup1 (see Table 4). These two simple examples show the importance of the update value of the minsup. Hence, if the value update is small, then we can extract association rules, but if the update value is high, we risk do not generate namely association rules.

The number of association rules is sensitive to the first value of minsup, minsup1. According to our database, if the value of minsup1 is greater or equal to 10 in the uniform case and greater or equal to 15 in the non-uniform case, so we will not generate association rules. Now, if the value of minsup1 is very small, we must see the importance of others minsup values, i.e., in levels 2 and 3, etc. We remark that the value of the first path (i.e., MS2) is the most that affect the number of valid generated association rules.

5.2 Decreasing minsup variant

In the present variant, the user is helped by a function to fix the minsup value. The user fixes the first value of the minsup, and then the subsequent values are determined by the function. This function consists on the following methods: $MS_i = MS_{i-1} - step$. The step is determined as the increasing variant according to two methods namely: constant and random. In the first method, the value step is calculating by the following formula: (MS1 − ϵ) ÷ (| sizeofthelargestitemset | −1) mod10. In the second method, the value of step is calculating as follows: random (MS1 − ϵ) ÷ (size of thelargest itemset -1) mod 10. Using these assumptions and fixing ϵ equal to 0:5, we obtain respectively the following results (see Tables 5 and 6).

Table 5. Experimental Results when Decreasing the Minsup Value with Constant Value

Minsup 1	Minsup 2	Minsup 3	Minsup 4	Minsup 5	Minsup 6	Minsup 7	Minsup 8	AR
19	16.35	13.71	11.07	8.43	5.78	3.14	0.50	48
18	15.50	13.00	10.50	8.00	5.50	3.00	0.50	48
17	14.64	12.28	9.93	7.57	5.21	2.86	0.50	48
16	13.78	11.57	9.36	7.14	4.93	2.71	0.50	48
15	12.92	10.86	8.78	6.71	4.64	2.57	0.50	48
14	12.07	10.14	8.21	6.28	4.36	2.43	0.50	80
13	11.21	9.43	7.64	5.86	4.07	2.28	0.50	80
12	10.36	8.71	7.07	5.43	3.78	2.14	0.50	188
11	9.50	8.00	6.50	5.00	3.21	1.86	0.50	208
10	8.64	7.28	5.93	4.57	3.21	1.86	0.50	208
9	7.78	6.57	5.36	4.14	2.93	1.71	0.50	400
8	6.93	5.86	4.78	3.71	2.64	1.57	0.50	400
7	6.07	5.14	4.21	3.28	2.36	1.43	0.50	1200
6	5.21	4.43	3.64	2.86	2.07	1.28	0.50	1200
5	4.36	3.71	3.07	2.43	1.78	1.14	0.50	2240

4	3.50	3.00	2.50	2.00	1.50	1.00	0.50	3648
3	2.64	2.28	1.93	1.57	1.21	0.86	0.50	5776
2	1.78	1.57	1.36	1.14	0.93	0.71	0.50	17944
1	0.93	0.86	0.78	0.71	0.64	0.57	0.50	38770

Table 6. Experimental Results when Decreasing the Minsup Value with Random Value

Minsup 1	Minsup 2	Minsup 3	Minsup 4	Minsup 5	Minsup 6	Minsup 7	Minsup 8	AR
20	17.54	15.36	13.46	11.60	9.54	8.19	6.59	8
19	16.70	14.56	12.55	20.68	8.98	7.36	5.88	20
18	15.79	13.49	11.91	10.07	8.40	14.19	5.36	46
17	14.75	12.55	10.64	8.91	7.43	5.93	4.64	48
16	13.88	11.89	21.78	8.25	6.72	5.29	3.91	49
15	12.98	11.00	9.24	7.63	6.14	4.75	3.14	57
14	12.03	10.16	8.49	6.91	5.38	4.09	2.80	76
13	11.17	9.40	7.73	6.19	4.77	3.47	4.21	128
12	10.17	8.46	6.88	5.39	4.03	2.79	1.51	196
11	9.21	7.56	6.08	4.52	3.32	2.09	0.67	208
10	8.33	6.83	5.41	4.11	2.85	1.67	1.51	307
9	7.38	5.78	4.45	3.16	1.98	0.45	0.01	554
8	6.46	5.06	3.78	2.52	120	0.03	0.01	1073
7	5.54	4.23	2.94	1.76	0.19	0.01	0.01	1676
6	4.62	3.34	2.14	0.55	0.01	0.01	0.01	3426
5	3.65	2.43	1.15	0.01	0.01	0.01	0.01	37356

In the decreasing variant of PF-Growth, update value are randomly determined, i.e. minsup1 values are decreased from 19 to 1 in the uniform case and from 20 to 5 in the non-uniform case. In this variant, we observed that we can use high values of minsup, while this is not possible in the increasing variant.

In Table 6, we study the case where we decrease the minsup value. From results obtained with this case, we remark that in high levels, the algorithm finds more frequent itemsets, and by consequence more valid association rules than the increasing case. Also, we mention that the number of association rules is sensitive to the update value of the minsup between minsupi and minsupi+1, for i varying from 1 to 7, and the inital value of minsup in each level. Finally, we have remarked that if the minsup1 is less than 5 in the random case, we obtained an explosion in the number of valid association rules, which is not the case in the uniform case.

6. Conclusion

We proposed a multi-support version of FP-Growth algorithm for generating association rules from databases. Contrary to the original version of FP-Growth that uses only one minimum support at all items of a database, the new version that we propose in this paper uses a minimum support of each level (for each k-itemsets). The use of multi-support allows to generate rare association rules that are very important in some data mining applications. The experiments that we have carried out have shown the interest of our proposal. In the short time, we plan to propose another variant of the same algorithm, in which a minimum support value will be associated with each item, and an another version in which we combine the variation of the minimum support between levels and items.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my advisor Prof. Yahia slimani for the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this article. Besides , I would like to thank my husband, my friends and my family: my parents, my brother, my sister for giving birth to me at the first place and supporting me spiritually throughout my life.

References

[1] R. Agrawal, T. Imielinski and A. N. Swami, "Mining association rules between sets of items in large databases", In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, May 26-28, pp. 207-216. Washington, D.C.

[2] H. Li, Y. Wang, D. Zhang, M. Zhang and E. Y. Chang, "Pfp: parallel fp-growth for query recommendation", Proceedings of the 2008 ACM conference on Recommender systems, **(2008)** October 23-25, pp. 107-114, Lausanne, Switzerland.

[3] L. Troiano, G. Scibelli and C. Birtolo, "A Fast Algorithm for Mining Rare Itemsets", 9h International Conference on Intelligent Systems Design and Applications (ISDA'09), November 30 - December 2, pp. 1149-1155, Pisa, Italy.

[4] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. I. Verkamo, "Fast discovery of association rules. In Advances in knowledge discovery and data mining", American Association for Artificial Intelligence, vol. 12, **(1996),** pp. 307-328.

[5] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal, "Efficient mining of association rules using closed itemset lattices", Information Systems, vol. 24, no. 1, **(1999),** pp. 25-46.

[6] Y. Bastide, R. Taouil, N. Pasquier, G. Stumme and L. Lakhal, "Mining minimal non redundant association rules using frequent closed itemsets", In Computational Logic, Lecture Notes in Artificial Intelligence, vol. 1861, **(2000)**, pp. 972-986, Springer.

[7] K. Gouda and M. J. Zaki, "Efficiently mining maximal frequent itemsets", In IEEE International Conference on Data Mining, IEEE Computer Society, **(2001),** pp. 163-170, Washington, DC, USA.

[8] B. Liu, W. Hsu and Y. Ma, "Mining association rules with multiple minimum supports", Proceedings of 1999 ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining, vol. 2009, **(1999)** June 28 - July 1, pp. 337-341, Paris, France.

[9] T. Wan and K. Zeitouni, "Mining Association rules with Multiple Min-supports Application to Symbolic Data", Student Journal (ISSN 1420-1011), vol. 5, no. 1, **(2004),** pp. 59-74.

[10] J. Han and Y. Fu, "Discovery of multiple-level association rules from large databases", Proceedings of the 21st International Conference on Very Large Data Bases, **(1995),** pp. 420-431, Zurich, Switzerland.

[11] K. Wang, Y. He and J. Han, "Mining frequent item sets using support constraints", Proceedings of the 26th International Conference on Very Large Data Bases, **(2000)** September 10 -14, pp. 43-52, Cairo, Egypt.

[12] C. S. Kanimozhi Selvi and A. Tamilarasi, "Association Rule Mining with Dynamic Adaptive Support Thresholds for Associative Classification", Proceedings of the International Conference on Computational Intelligence and Multimedia Applications, ICCIMA, **(2007)** December 13-15, vol. 2, pp. 76-80, Sivakasi, Tamilnadu, India.

Authors

Wiem Taktak
Master in Computer Management.
Professor JAVA, C/C++, ISG Tunis.

Yahia Slimani
Professor in higher education in computer Science.
Professor Databases, Data Mining, ISAMM Manouba.