Algorithm Research of Top-Down Mining Maximal Frequent SubGraph Based on Tree Structure

Xiao Chen[1], Chunying Zhang[2,3], Fengchun Liu[1,2], and Jingfeng Guo[3]

[1] Qian'an College, Hebei United University,
5096 Yanshan Road, 064400 Qian'an, Hebei, China
chenxiao0604@163.com
[2] College of Science, Hebei United University,
46 Xinhua Road, 063009 Tang Shan, Hebei, China
{zchunying,lnobliu}@heut.edu.cn
[3] College of Information Science and Engineering, Yanshan University,
438 Hebei Ave, 066004 Qinhuangdao, Hebei, China
jfguo@ysu.edu.cn

Abstract. For the existence of problems with mining frequent subgraph by the traditional way, a new algorithm of top-down mining maximal frequent subgraph based on tree structure is proposed in this paper. In the mining process, the symmetry of graph is used to identify the symmetry vertex; determining graph isomorphism based on the attributed information of graph, the tree structure is top-down constructed and completed the calculation of support. Which is reduced the unnecessary operation and the redundant storage of graphs, and the efficiency of algorithm is improved. Experiments show that the algorithm is superior to the existing maximal frequent subgraph mining algorithms, without losing any patterns and useful information.

Keywords: Maximal Frequent Subgraph, Top-Down, Graph Isomorphism, Tree Structure.

1 Introduction

With the appearance of the large amounts of structured data, and the increasing demand on analysis, graph mining has become an active and important theme in data mining [1]. Among the various kinds of graph mining, the frequent subgraph mining is the most basic patterns and bottleneck; in order to improve the mining efficiency and reduce the number of the result set, we can convert mining frequent subgraph to mining maximal frequent subgraph. The existing methods of maximal frequent subgraph mining [2-4], which whether based on BFS or based on FP-growth are used the mining thinking of bottom-up, there are some problems, such as: (1) Need to through multiple iterations and the determine of subgraph isomorphism in the mining processing (to determine subgraph isomorphism is NP complete problems [5]). (2) Only reduced the number of results sets by pattern growth approach to mining maximal frequent subgraph sets, compared with the frequent subgraph sets, and does not reduced the mining difficult.

In addition to the bottom-up mining methods, the top-down mining strategy is referred in [6], which using this search order, the itemsets are traversed and checked from big to small. It's the outstanding advantages as follows. (1) Reduced the number of calculate subgraph support by the property that if the supersets is frequent then its subsets must be frequent; (2) The maximal frequent itemsets to facilitate fast discovery by the top-down mining approach, reduced the difficulty of mining and storage redundancy. Therefore, it is can be applied to graph mining areas that the top-down mining strategy, however, due to the complex graph structure, which can not be directly applied to mining maximal frequent subgraph, and need to improve their algorithm.

In this paper, we propose a new algorithm of maximal frequent subgraph based on top-down mining strategy. The apache is top-down construct tree structure, calculated the support using the determining of graph isomorphism tree structure, which to improve the algorithm efficiency through avoid to determine subgraph isomorphism. And considering the symmetry structure and the attribute information of graph reduced the unnecessary operation and the storage redundant of graphs in reduced graphs scale. Experiment shows that the algorithm is superior to the previous algorithms.

The rest of this paper is organized as follows. In Section 2 the preliminary concepts are introduced. Section 3 presents the related properties and algorithm description. An experimental study is presented in Section 4 and we conclude in Section 5.

2 Preliminaries

There are some preliminary concepts contributions in [1, 5], such as Labeled Graph, SubGraph, SupperGraph, Support, etc. In order to facilitate understanding, only a few of the more important concept are given in the following.

Definition 1. Given grahpsets D = {G_1, G_2,…, G_n} and the user-specified *min_Sup* ∈ (0,1], G_i is a maximal frequent subgraph iff there isn't existing Sup($G' = G_i \Diamond e$) ≥*min_Sup* in D [4].

Definition 2. Given a pair of labeled graphs G={V,E,ΣV,ΣE,L} and G'={V', E', ΣV', ΣE', L'}, G is isomorphic to G' *iff* there exists a mapping f : V(G) ↔ V(G') such that:

$\forall u \in V,\ L(u) = L'(f(u))$, $\forall u, v \in V,\ V((u,u) \in E) \Leftrightarrow$, $(f(u), f(v)) \in E)$ $\forall (u,v) \in V$, $L(u,v) = L'(f(u), f(v))$.

Property 1. If Graph G is isomorphic to G', then the vertices number of Graph G and G' is same, that is |V(G)| = |V(G')| [7].

Property 2. If Graph G is isomorphic to G', then the edges number of Graph G and G' is same, that is |E(G)| = |E(G')| [7].

Property 3. If Graph G is isomorphic to G', then the vertices number of same degrees in Graph G and G' is same [7].

The above three properties is the necessary conditions for graph isomorphism, but not sufficient. That is, the above three properties establish in the conditions of graph isomorphism, so graphs isomorphic is uncertain when only meeting the above three properties. For example, Given graph G and G', as shown in Figure 1, then, (1) $|V(G)| = |V(G')| = 7$; (2) $|E(G)| = |E(G')| = 12$; (3) The degrees of vertex a and 1 are 6, the other vertices degrees are 3. Obviously, the graph G and graph G' satisfies the above three properties, but the graph G and graph G' is non-isomorphic, this proved that the properties of the non-sufficiency.

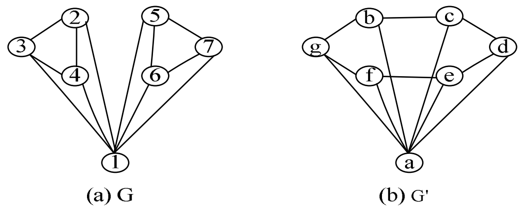


Fig. 1. An example of Non-isomorphic

Definition 3. A labeled graph G is subgraph isomorphic to a labeled graph G', denoted by $G \subseteq G'$, *iff* there exists a subgraph G″ of G' such that G is isomorphic to G″ [2].

However, in determining the subgraph isomorphism, because do not know in advance which the subgraph of G' isomorphism with G, need to match several times, reducing the algorithm efficiency. Therefore, it can be seen, subgraph isomorphism is more complex than isomorphism in computationally, so in the process of top-down mining, only determined graph isomorphism, to avoid the comparison of subgraph isomorphism, to improve the algorithm efficiency.

Definition 4. The graph G is Non-Unique Labeled Undirected Simple Graph iff the graph structure is described as following.

(1) A graph is simple graph, if the graph does not existing ring and the multiple edges between any two vertices;

(2) A graph is undirected, if the graph contains only undirected edges, such as (u, v);

(3) A graph is non-unique labeled graph, if exist the same vertex label or edge label in graph, which the label is the inherent information of the vertex or edge.

There are many non-unique identifier graphs in reality, such as benzene ring containing many carbon atoms of the same labels, the undirected graph of Non-unique identity are the study objects in this paper. In order to further simplify the determine graphs isomorphism, as the necessary conditions for determining isomorphic of non-unique identity graphs, attributed information of graph is defined as follows, which is storied by the form of single linked list.

Definition 5. An attributed information of labeled graph G is a four element-tuples $GAI = \{|V|; |E|; |C^{\Sigma V}{}_i|; C^{\Sigma V}{}_i[|V(C^{\Sigma V}{}_i)|\text{-MaxD}(C^{V}{}_i)]) \}$ (order by the lexicographic order of vertex labeling). Where $|V|$ is the number of vertices; $D(V_i)$ is degree of the vertices; $|E| = \sum D(V_i) / 2$ is the size of graph (that is the number of edges); $C^{\Sigma V}{}_i$ is the vertex labels types; $|C^{\Sigma V}{}_i|$ the number of vertex labels types; $|V(C^{\Sigma V}{}_i)|$ that the number of same labeling vertices; $\text{MaxD}(C^{\Sigma V}{}_i)$ that the maximum degree of vertex types.

Example 1. According to the method of reference [8] relabeled graph dataset D = {G_1, G_2, G_3}, as shown in Figure 2, the attributed information of graphs are as follows. GAI(G_1) = (4; 6; 2; X[2-3], Z[2-2]), that is, there are 4 vertices and 6 edges, which two types vertex labels are X and Z, the number of vertices with same labeled X is 2, the maximum degree of vertices X is 3, the other the same way available; GAI(G_2) = (4; 5; 2; X[3-3], Z[1-3]); GAI(G_3) = (4; 5; 1; X[4-3]).

Based on the traditional graph isomorphism property and the definition of attribute information of graph, the extended property as follows (the property of 1, 2 still valid).

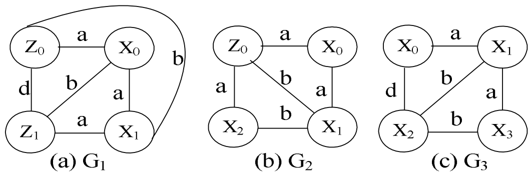


Fig. 2. Relabeled graph dataset

Property 4. If Graph G is isomorphic to G', then the number of vertex labels types is same in Graph G and G', that is $|C^{\Sigma V}{}_i(G)| = |C^{\Sigma V}{}_i(G')|$.

Proof: According to the definition graph isomorphism and labeled graph, when Graph G is isomorphic to G', $|V(G)| = |V(G')|$ and $|E(G)| = |E(G')|$, and $\forall$ u$\in$V(G), L(G(u)) = L(G'(f (u))); $\forall$ L(G(u,v))=L(G'(*f* (u), *f* (v)); then, then the number of vertex labels types is same in Graph G and G', that is $|C^{\Sigma V}{}_i(G)| = |C^{\Sigma V}{}_i(G')|$.

Property 5. If Graph G is isomorphic to G', then the number of same types vertex labeled and the maximum degree are same in Graph G and G', that is $C^{\Sigma V}{}_i[|V(C^{\Sigma V}{}_i)|\text{-MaxD}(C^V{}_i)](G) = C^{\Sigma V}{}_i[|V(C^{\Sigma V}{}_i)|\text{-MaxD}(C^V{}_i)](G')$.

Proof: According to the definition of graph isomorphism and the properties 1and 4, when graph G is isomorphic to G', then $|V(G)| = |V(G')|$, and $\forall$ u$\in$V(G), L(G(u)) = L(G' (*f* (u))) $\Rightarrow$ $|C^{\Sigma V}{}_i(G)|= |C^{\Sigma V}{}_i(G')|$, so $|V(C^{\Sigma V}{}_i(G))|=|V(C^{\Sigma V}{}_i(G'))|$; according to the properties 3, $\text{MaxD}(C^V{}_i(G)) = \text{MaxD}(C^V{}_i(G'))$; to sum up, when graph G is isomorphic to G', then $C^{\Sigma V}{}_i[|V(C^{\Sigma V}{}_i)|\text{-MaxD}(C^V{}_i)](G) = C^{\Sigma V}{}_i[|V(C^{\Sigma V}{}_i)|\text{-MaxD}(C^V{}_i)]$ (G').

Similarly, the properties 1, 2, 4 and 5 are necessary conditions for determine graph isomorphism, that is, these are established in the conditions of graph isomorphism; the graph must be non-isomorphic when not meet any one of the above four properties. Therefore, the above four properties can be as a precondition for determine isomorphism, based on graph attribute information, the method as follows.

```
Algorithm 1: To determine the graph G and G' whether isomorphic
  Input: Graph G and G', GAI(G) ={|V|; |E|; $|C^{\Sigma V}_i|$; $C^{\Sigma V}_i[|V(C^{\Sigma V}_i)|$-MaxD($C^V_i$)]},
         GAI(G') ={|V|; |E|; $|C^{\Sigma V}_i|$; $C^{\Sigma V}_i[|V(C^{\Sigma V}_i)|$-MaxD($C^V_i$)]};
  Output: True (isomorphic) or Flash (non-isomorphic).
  Begin
  (1)   If $|V(G)| \neq |V(G')|$, then the graph G and G' is non-isomorphic;
  (2)     Else if $|E(G)| \neq |E(G')|$, then the graph G and G' is non-isomorphic;
  (3)       Else if $|C^{\Sigma V}_i(G)| \neq |C^{\Sigma V}_i(G')|$ , then the graph G and G' is non-isomorphic;
  (4)         Else if $C^{\Sigma V}_i[|V(C^{\Sigma V}_i)|$-MaxD$(C^V_i)](G) \neq C^{\Sigma V}_i[|V(C^{\Sigma V}_i)|$-MaxD$(C^V_i)](G')$,
              then the graph G and G' is non-isomorphic;
  (5)           Else, using DFS Cord or determinant judgments graph isomorphic
  (6)   Return True or Flash
  End
```

Example 2. To determine isomorphism of graph G_2 and G_3, based on the attributed information of graph, firstly, compare to the number of vertices: $|V(G_2)| = |V(G_3)| = 4$, to downward compare to the sizes of graphs: $|E(G_2)| = |E(G_3)| = 5$, and continue to compare to the vertex labels types: $(|C^{\Sigma V}_i(G_2)| = 2) \neq (|C^{\Sigma V}_i(G_3)| = 1)$, then graph G_2 and G_3 have the different numbers of vertex labels types, that is graph G_2 is non-isomorphic to G_3, which reduces the number of comparisons graph isomorphism and improve the efficiency of the algorithm.

Definition 6. An isomorphism from a graph G to itself is called an automorphism.

Definition 7. A graph G is symmetric if, given any two pairs of linked vertices u_1-v_1 and u_2-v_2 of G, there is an automorphism f: $V(G)\rightarrow V(G)$ such that $f(u_1) = u_2$ and $f(v_1) = v_2$[7]. Given automorphism graph G and its subgraph G' ($G' \subset G$), if G' is symmetric, then G is locally symmetric graph.

Property 7. Symmetric graph G must be automorphism, automorphism graph G not necessarily symmetry [7].

Therefore, the symmetric vertex of symmetric graph must be a pair corresponding point in automorphism graph, as shown in Figure 2(a), (X_0, X_1) and (Z_0, Z_1) are symmetric vertices in G_1; the symmetric edge form by the symmetric vertices, that is a pair corresponding edge in automorphism graph, (Z_0, a, X_0) and (Z_1, a, X_1) are symmetric edges in G_1; graph G_2 is called a locally symmetric graph in Figure 2(b), because it contains symmetric subgraph (X_2, a, Z_0, a, X_0).

Definition 8. Given a labeled graphset D= $\{G_1, G_2,\ldots, Gn\}$, the frequency of g in D is the sum of the inner frequent of g in Gi, denoted by $f(g, D) = \sum_{i=1}^{n=|D|} in\_f(g, G_i)$ [8].

3 The Related Properties and Algorithm Description

3.1 The Unique Labeled Method Based on the Graph Symmetry

Usually, the unique labeled of graph is to distinguish different vertex with same label, which can simplify the problem of determining subgraph isomorphism, but this method can't reflect on the structure and other properties of graph. Therefore, we propose a new apache of unique labeled graph based on the graph symmetry and the storage table of frequent graph.

Example 3. Given graphset D= {G_1, G2, G3}, as shown in Figure 2, the frequency 1-edge sets is {(X, a, X), (X, a, Z), (X, b, Z)}. When the extension connected the subgraphs (X, a ,Z) and (X ,b, Z) of G1, because need to consider a variety of different connections between the same label vertices, it will produce 4 super-graph, respectively, (Z_1, b, X_0, a, Z_0), (X_0, a, Z_0, b, X_1), (X_1, a, Z_1, b, X_0) and (Z_0, b, X_1, a, Z_1), where (Z_1, b, X_0, a, Z_0) and (Z_0, b, X_1, a, Z_1), (X_0, a, Z_0, b, X_1) and (X_1, a, Z_1, b, X_0) are isomorphic, then appear to redundant storage. Although the algorithm can enhancethe space complexity to improve time efficiency, but still should pay attention to reduce the unnecessary waste of space, It is precisely because there is symmetry edge (X_0, a, Z_0) and (X_1, a, Z_1) in graph G_1 which leads to redundant storage. Therefore, to reduce redundancy, this paper relabeled graph G under assumes known the symmetry of graph (This paper only considering the rules of axial symmetry graph\locally symmetric graph, center symmetry graph and other symmetry graph for the further research). For example, the symmetric vertices (X_0, X_1) of graph G_1 is relabeled to (X_{01}, X_{10}) in Figure 2(a), it is example X_{01} that 0 represents the original vertex label and 1 represents its symmetrical label, Similarly, other vertices can be relabeled, as shown in Figure 3, that is, the new unique labeled graph based on the graph symmetry is proposed in this paper. Then, G_1's subgraph (X, b, Z, a, X) is stored as (X_{01}, b, Z_{10}, a, X_{10}) can be said a variety of forms, the storage table of G1 frequent 1-edge graph as shows in Table 1, each subgraph and it's in_frequency is stored within the table. Due to consider the graph symmetry, reducing the storage redundant of subgraph, and improves the efficiency of calculate the support.

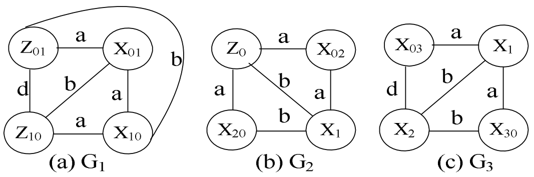


Fig. 3. Labeled GraphSets Based on Symmetrical Structure

Table 1. The new table of frequency graphs

Egde	*in_f*(g_i,G_1)		*in_f*(g_i,G_2)		*in_f*(g_i,G_3)		Sup
(g_i)	G_1	*in_f*	G_2	*in_f*	G_3	*in_f*	
(X,a,X)	(X_0,a,X_1)	1	(X_0,a,X_1)	1	(X_{03},a,X_1)	2	3
(X,a,Z)	(X_{01},a,Z_{01})	2	(X_{02},a,Z_0)	2		0	2
(X,b,Z)	(X_{01},b,Z_{10})	2	(X_0,b,Z_0)	1		0	2

3.2 Pruning Strategy

There are some properties in mining frequent subgraphs are as follows [1].

Property 8. Sup(g) ≥ Sup($g' = g \Diamond e$), which g' is a supergraph.

Property 9. If the subgraph g is infrequent, then any supergraph g' of g are infrequent.

In this paper, the top-down approach for mining maximal frequency subgraph sets, because includes some non-frequent edge in graphsets, so need to pre-operation graphsets by these two properties, which can delete all infrequent edge and the local frequent edge, when there is non-connected graph can be regarded as more graphs. Pretreated the graph set D to D' = {G_1', G_2', G_3'}, as shown in Figure 4.

Property 10. Give g and *min_Sup*, if g is a maximal frequency subgraph, so all subgrahp of g is frequency, then, can stop the operation to reduce the scale of g.

Proof: Due to using the top-down mining strategy, when g is a maximal frequency subgraph, Sup(g) $\geq$ *min_Sup*, then, Sup($g' = g\Diamond e$) $\leq$ *min_Sup*, and Sup(g-e) $\geq$Sup(g) $>$ Sup($g' = g\Diamond e$). According to the concept of maximal frequency subgraph, g-e is frequent and not maximal, so completed a task of mining maximum frequent son figure, it can stop the operation of reduced graph scale.

Example 4. Using the traditional top-down approach to reduce the scale of G_1' in Figure 4(a) as follows: First, remove the 1-edge of G_1', as the graph G scale is 5, you need to 5 times delete operation and obtain 5 sub-graph that its scale is 4, as shown in Figure 5, but (b) and (c), (d) and (e) are isomorphic in Figure 5, in order to reduce storage redundancy needs to be judged isomorphic. But, if we consider t G_1' is symmetric graph, that delete the symmetric edge (X_{01}, a, Z_{01}) or (X_{10}, a, Z_{10}) will be the same composition, thereby to reduce the unnecessary operation and to improve algorithm efficiency. At the same time, it is also necessary that identify the local symmetry vertices in the graph, such as the symmetric vertex of X_{02} and X_{20} in Figure 3(b), as graph G2' reduced to the symmetric graph (X_{02}, a, Z_0, a, X_{20}), and then delete the edge (X, a, Z) can be reflected on the value of the symmetry vertex. So we proposed a property based on the symmetry as follows.

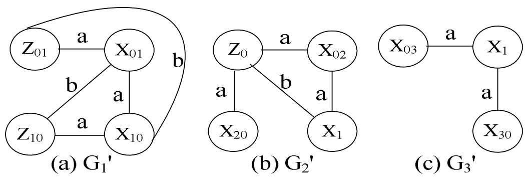


Fig. 4. Graph Dataset D'

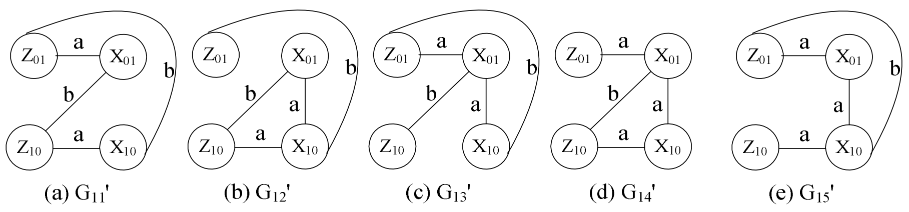


Fig. 5. K-1 Sub-graph Sets of Graph G_1'

Property 11. Given rule symmetric graph G(V,E), e_1, $e_2 \in$ E(G), if G' = G-e_1, G'' = G-e_2, and e_1 and e_2 is a pair symmetric edge, then G' $\cong$ G''.

Proof: According to the property 6 and the definition of graph isomorphism, we know that the symmetric graph must be auto-orphism, and exiting the corresponding symmetric edges. So, when remove the symmetry edge e_1, e_2 in rule symmetric graph G, can be obtained with the isomorphic graph G' and G''. That is the property 11 is right.

Thus, in the case of known symmetric graph, the property of 10 and 11 simplifies the operation, which remove edge and the number of determine isomorphism, and then improved the algorithm efficiency without losing any result set.

3.3 Top-Down Mining Maximal Frequency Subgraph Based on Tree Structure

This paper, graph sets will be constructed to a two-way tree, calculate the support through the tree structure, and broken down the larger graph into a small graph, which each child node are obtained by removed an edge of his father node in the tree.

Algorithm 2: Top-Down mining maximal frequency subgraph (TD-MG)

Input: Graphset D = {G_1, G_2, G_3} and *min_Sup* = 2;

Output: The Maximal Frequency Subgraph Sets.

Begin

(1) Scanning graphset D and labeled it, record the attributed information of graphs and calculate the support of 1-edge;

(2) Remove infrequent edges and local frequent edges by the property 8 and 9, obtain the storage table of frequent 1-edge, as show in Table 1, relabeled D into D' (Figure 3) using the symmetric, and the attribute information of graph as follows: GAI(G_1') = (4; 5; 2; X[2-3], Z[2-2]), GAI(G_2') = (4; 4; 2; X[3-2], Z[1-3]), GAI(G_3') = (3; 2; 1; X[3-2]);

(3) Removed frequent 1-edge using the top-down method based on the property 10 and 11, establish tree and calculate the support, as shown in Figure 6.

a) The root of tree is pretreatment graph set D'={G1', G2', G3'};
b) The first layer nodes of the tree is graph Gi' which is the largest scale graph in D', D' = D' - G_i', and record the support of G_i' (for the first time, Sup(G_i') = 1), if there is more than the same scale graph, order by the order in graph set. For example, there is only one largest graph G_1' in D , its scale is 5, and which is the first layer node in tree, *Sup*(G_1') = 1, and then D' = {G_2', G_3'};
c) In reducing scale of G1' by removing 1-edge, reduce the operator number of deleted edge, determine isomorphic and the storage redundancy according to the property 11, and exclude non-composition using Algorithm 1, obtain G1' three sub-graphs of G1' that the scale is 4, as the second layer nodes, and calculate the support;
d) Followed recursively, until finding all maximal frequent subgraphs of G1' so far, then established tree structure of graph G1';
e) Continue to find the largest graph Gi' in the remaining sets D', and find to the corresponding layer of the tree. First of all, to determine isomorphism with graphs of the same layer, if existing isomorphism , with the pointer point to the graph and add the attribute information of graph, support

cumulative; else, it as the last node in this layer, Sup(G1') = 1. For example, the largest graph is G2' in D' , which scale is 4 corresponds to the second layer node of the tree, first, determine to isomorphism comparison G2' with the second layer graphs, due to non-isomorphism, then it as the last node in this layer with other node together;

f)Due to Sup(Gi') ≤ min_Sup, reduce the graph Gi' scale and find to the maximal frequent subgraph of Gi', continue to construct the tree

g) Followed by recursion, Until D' = Φ.

Note: In order to express simple, link between the tree nodes is used double-way pointer, Node in the tree only shows he attribute information of graph, In the tree, the black point as a bifurcation point, that's it would be more than one sub-graphs when removed the edge of internal frequency is not 1, In the recursive process, labeled the layer and order of sub-graph, such as the G_{11}', G_{21}'', etc.. G_{11}'is the first K-1 sub-graph of G_1', G_{21}' is the first K-1 sub-graph of G_2'; the subscript of all sub-graphs of G_1' are beginning with 1, the subscript of all sub-graphs of G_2' are beginning with 2, similarly, can be get other label. So, we can calculation the number of super graph as long as find the different label of graph.

(4) Consture Tree T, as show in Figure 6, Shaded part of T for all frequently, to avoid output of non-maximal frequent subgraph, and need to verify it.

(5) Output the maximal frequent subgraphs sets.

End

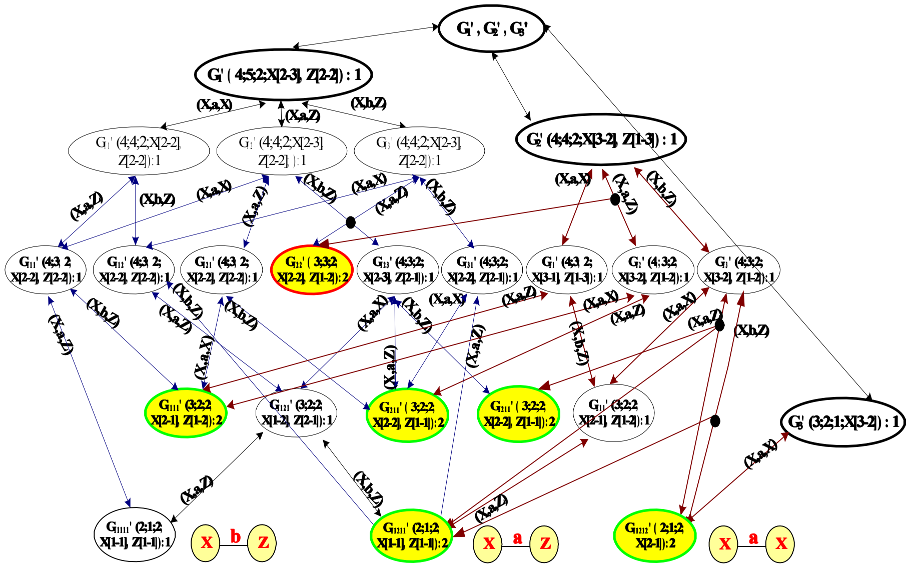
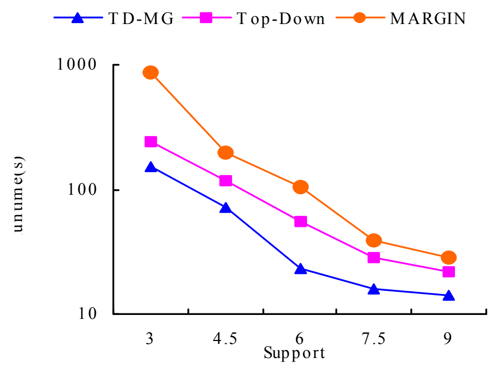

G1', G2', G3'
G1' (4;5;2;X[2-3], Z[2-2]) : 1
G2' (4;4;2;X[3-2], Z[1-3]) : 1
G3' (3;2;1;X[3-2]) : 1
(X,a,X)
(X,b,Z)
(X,a,Z)
G111' (3;2;2 X[2-1], Z[1-2]):2
G1111' (2;1;2 X[1-1], Z[1-1]):1
G1211' (2;1;2 X[1-1], Z[1-1]):2
G1212' (2;1;2 X[2-1]):2
X b Z
X a Z
X a X

Fig. 6. Tree T

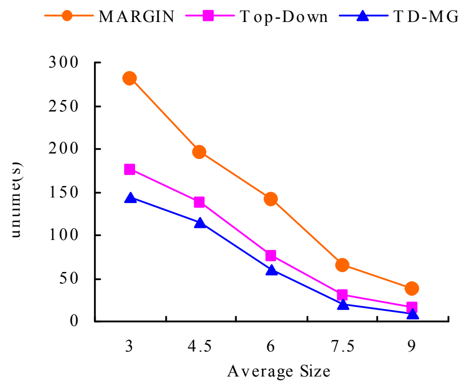
4 Experimental

Experimental environment: 2.26GHZ Intel Core i3 PC with 2G main memory, running Windows XP Professional SP3.

A comprehensive performance study has been conducted in our experiment on synthetic databases. We use a synthetic data *D*10*kT*16*V*5*E*5 generator provided by Kuramochi and Karypis. In order to facilitate comparison with previous experiments, select *min_Sup* from 3% to 9% of the running time, the run time and the effect of maximal frequent subgraph's average size are shown in Figure7. Figure 7(a) shows the runtime with *min_Sup* varying form 10% to 3% on the synthetic dataset. Effect ofmaximal frequent subgraph's average size on running time is shown in Figure 7(b). As we can see through the data curve of coordinate, (1) the difference of run time is small when the support is high; (2) with the increasing of the support, the running time of sharp decline, the efficiency of the DT-CM algorithm to only show a slight increase in the data sets; (3) the result set of mining is the same basically, meanwhile, which is also verified the correctness of the algorithm. Experiments show that the DT-M algorithm is superior to Top-Down and MARGIN algorithm.



(a) The effect of support on runtime



(b) Effect of maximal frequent subgraph's average size on running time

Fig. 7. The Experimental Result of Simulated Data Set

5 Conclusion

In this paper, we propose a new algorithm of top-down mining maximal frequent subgraph. We propose a new unique labeled apache by the graph symmetry, which is reduce the redundant storage and sacrificing space complexity; and reduced the number of determining graph isomorphism by the attribute information of graph. Our study shows that the algorithm outperforms other mining algorithms. However, we conduct the study on the premise of the symmetric of graph. Further researches are needed regarding how to judge the symmetric of graph, how to improve the accuracy and the efficiency of algorithm by the other structure and nature of graph.

Acknowledgment. This work is supported by the National Science Foundation of China, (No.60673136) Nation Science Program of Hebei Education Department (No.2009101) and Major Basic Research of Hebei Applied Basic Research Projects (No.10963527D). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, pp. 79–165. Mechanical Industry Publishing, Beijing (2007), Translate by Fan, M., Men, X.f.
2. Thomas, L.T., Valluri, S.R., Karlapalem, K.: MARGIN: Maximal Frequent Subgraph Mining. In: Proc of ICMD 2006, Hong Kong, pp. 1097–1101 (2006)
3. Wang, Y.-l., Yang, B.-r., Song, Z.-f., Chen, Z., Li, L.-n.: New Algorithm for Mining Maximal Frequent Subgraphs. Journal of System Simulation 20(18), 4872–4877 (2008)
4. Guo, J., Chai, R., Li, J.: Top-Down Algorithm for Mining Maximal Frequent Subgraph. Advanced Materials Research 204 - 210, 1472–1476 (2011)
5. Chakrabarti, D., Flaoutsos, C.: Graph mining: laws, generators, and algorithms. ACM Computing Surveys 38(1), 1–69 (2006)
6. Gouda, K., Zaki, M.J.: Efficiently Mining Maximal Frequent Itemsets, pp. 101–108. Springer Press, New York (2002)
7. Chen, X.: How to Confirm Isomorphic of Graph, `http://www.paper.edu.cn`
8. Zou, X., Chen, X., Guo, J., Zhao, L.: An improved algorithm for mining CloseGraph. ICIC Express Letters Journal of Research and Surveys 4(4), 1135–1140 (2010)