

引用格式:唐德权,黄金贵. 基于图数据的极大频繁子树挖掘算法研究[J]. 微电子学与计算机, 2020, 37(10): 54-58. [TANG D Q, HUANG J G. Algorithm of maximum frequent sub-tree mining based on graph data[J]. Microelectronics & Computer, 2020, 37(10): 54-58.]

基于图数据的极大频繁子树挖掘算法研究

唐德权^{1,2}, 黄金贵³

(1 湖南师范大学 数学与统计学院, 湖南 长沙 410081; 2 湖南警察学院 信息技术系, 湖南 长沙 410138;

3 湖南师范大学 信息科学与工程学院, 湖南 长沙 410081)

摘 要: 由于极大频繁子树中已经隐含了所有频繁子树信息, 尤其处理大型图数据集时候, 挖掘极大频繁子树对提高频繁子树挖掘算法效率具有重要意义. 首先在有效编码的基础上提出连接和扩展操作算法, 通过两个算法产生所有极大候选子树; 其次引入嵌入集计算解决子树同构问题, 对子树同构问题进行了优化, 进一步提出了一种新的极大频繁子树挖掘算法(MFST); 最后证明了算法的正确性和分析了算法在最坏情况下的时间性能, 并与其它基于半结构化数据集的频繁子树挖掘算法进行了比较. 实验结果表明, MFST 算法具有更好的时间性能和空间性能, 可以在图数据集中有效挖掘频繁子树.

关键词: 极大频繁子树; 图数据集; 候选子树; 子树同构; 半结构化数据集

中图分类号: TP311.2

文献标识码: A

文章编号: 1000—7180(2020)10—0054—05

Algorithm of maximum frequent sub-tree mining based on graph data

TANG De-quan^{1,2}, HUANG Jin-gui³

(1 School of Mathematics and Statistics, Hunan Normal University, Changsha 410081, China;

2 Department of Information Technology, Hunan Police Academy, Changsha 410138, China;

3 College of Information Science and Engineering, Hunan Normal University, Changsha 410081, China)

Abstract: In view of the the current frequent subtree mining problems, the maximal frequent subtree mining algorithm can improve the efficiency of the frequent subtree mining algorithm. Firstly, propose two operations of the join and extension on the basis of effective coding, which are generate all candidate subtrees. Secondly, the embedded set is used to solve the problem of subtree isomorphism, and a new maximum frequent subtree mining algorithm(MFST) is proposed. Finally, the correctness of the algorithm is proved and the time performance of the algorithm in the worst case is analyzed. And compared with other frequent subtree mining algorithms based on semi-structured data sets, prove this algorithm is superior to the current frequent subtree mining algorithms. The experimental results show that the MFST algorithm has better time and spatial performance, and can effectively mining frequent subtree on the graph data sets.

Key words: maximal frequent subtree; graph data sets; candidate subtree; subtree isomorphism; semi-structured data sets

1 引言

频繁子树挖掘技术应用广泛^[1],但由于树作为特殊图数据结构的复杂性^[2],在现有的很多频繁子树挖掘算法采用技术通常以大量时间和空间作为代价^[3].随着数据集规模不断扩大,现有的频繁子树挖掘算法已经不能满足需求^[4],频繁的子树的数量与树的大小呈指数级增长.因此,设计更适合大型数据集或复杂数据集的频繁子树挖掘算法势在必行.由于极大频繁子树中已经隐含了所有频繁子树,挖掘极大频繁子树不会丢失频繁子树信息,而且能显著减少挖掘的子树总数^[5],对处理大型图数据库也同样具有优势^[6].

本文提出了一种新的基于图数据集的极大频繁子树挖掘算法 MFST (Maximal Frequent SubTree mining algorithm, 简称 MFST),算法总体的时间复杂度为 $O(2^{n \log n})$,其中 n 是顶点数,在面向图数据集时,本文算法明显优于基于半结构化(树结构)数据集的算法^[7].

2 基本概念

定义 1 标号图 标号图 G 是一个无向简单图,定义为四元组 $G = (V, E, \Sigma, l)$,其中 V 是图 G 的顶点集, $E \subseteq V \times V$ 是图 G 的边集, Σ 是由数字或字母组成的具有偏序关系的标号集, l 是顶点和边到标号集的映射函数,即 $l: V \cup E \rightarrow \Sigma$.

定义 2 支持度 给定图数据集 GD ,其中与图 G 同构的子图的个数与图数据集 GD 中所有图的数量的比值称为图 G 的支持度,记为 σ_G ,表示为

$$\sigma_G = \frac{|\{G' \in GD \mid G \subseteq G'\}|}{|GD|}.$$

定义 3 频繁子图和频繁子树 给定图数据集 GD ,假设其最小支持度阈值为 $\text{Min_}\sigma$,如果图 G 的支持度大于或等于最小支持度 ($\sigma_G \geq \text{Min_}\sigma$),那么图 G 称为频繁图;频繁图 G 的子图 G' 称为频繁子图.特别地,若频繁图 G 是树则称为频繁树;频繁树的子树称为频繁子树.

定义 4 极大频繁子图和极大频繁子树 对于图数据集 GD 中的一个频繁子图 G ,如果任意真包含 G 的图 G' ($G \subset G'$) 都不是频繁图,则称图 G 为极大频繁子图.特别地,若极大频繁子图是树,则称为极大频繁子树.

3 图的编码与规范化矩阵

因为邻接矩阵是对称矩阵,为了提高效率,对邻

接矩阵进行简化,取邻接矩阵 A 的下三角矩阵,称为图 G 的下三角邻接矩阵,仍记为 A .

定义 5 给定任意连通图 G ,其下三角邻接矩阵 A 的下三角元素依次按行列顺序排列给出一个编码:

$$a_{1,1} \ a_{2,1} \ a_{2,2} \ \cdots \ a_{n,1} \ a_{n,2} \ \cdots \ a_{n,n-1} \ a_{n,n}$$

记作 $\text{code}(A)$. 称 $\text{code}(A)$ 是图 G 的编码.

定义 6 设图 G 的下三角邻接矩阵 A ,若其编码 $\text{code}(A)$ 在所有编码中最大,则称为图 G 的规范化编码(Canonical code),记为 $\text{code}(G)$. 对应的下三角邻接矩阵 A 称为图 G 的规范化邻接矩阵(Canonical Adjacency Matrix),记为 $\text{CAM}(G)$ 或 CAM .

图 G 的 CAM 矩阵具有很好的性质.

定理 1 若 A 是 n -阶连通标号图 G 的 CAM 矩阵,则

- (1) $a_{1,1}$ 是图 G 中标号最大的顶点;
- (2) 每一行至少有一个非零边元素;
- (3) 对任意的 $1 \leq k < n$, A 的前 k 行组成的下三角矩阵也是 CAM ,且对应 G 的一个连通 k -子图.

证明:(1) 因为 $\text{code}(A)$ 是最大编码,所以 $a_{1,1}$ 必然是图 G 中标号最大的顶点.

(2) 用反证法可以证明,若有得到更大的编码,将与 $\text{code}(A)$ 是最大编码矛盾.

(3) 对任意的 $1 \leq k < n$, A 的前 k 行组成的下三角矩阵在图 G 的 k -子图中也是编码最大的,即是某个 k -子图的 CAM . 如若不然,则有 G 的一个更大编码从而得出矛盾.

定理 2 连通标号图的规范化编码是唯一的,且不同构的标号图其规范化编码也不同.

证明:显然图 G 的编码取决于它的邻接矩阵,而邻接矩阵又取决于图顶点的一个排列. 设图 G 有 n 个顶点,则顶点有 $n!$ 种不同的排列,每种排列对应一个邻接矩阵,从而对应 $n!$ 个不同的编码,这些编码中必定存在一个最大的,且是惟一的. 对任意两个标号图 G_1 和 G_2 ,设其最大编码分别为 $\text{code}(G_1)$ 和 $\text{code}(G_2)$. 因为 $\text{code}(G_1) = \text{code}(G_2)$,所以两者的规范化矩阵也相等,两个图 G_1 和 G_2 同构.

4 候选子树的生成

极大频繁子树的挖掘算法关键在于候选子树的生成方法. 为了寻求所有的候选子树,定义相关的基本操作:连接(Join)和扩展(Extension).

4.1 连接

在频繁子树挖掘中,采用连接操作由两个已知的频繁子树生成新的候选子树。

定义 7 设 k 阶树 T 的 CAM 矩阵为 $\mathbf{A}$, 称它的前 $k-1$ 行为 $\mathbf{A}$ 的核, 它所对应的 $(k-1)$ -子树, 称为 T 的极大真子树。

设 $\mathbf{A}$ 和 $\mathbf{B}$ 分别是两个 k -树的 CAM 矩阵, $\mathbf{A}$ 和 $\mathbf{B}$ 的前 $k-1$ 行相同, 则可以进行连接操作, 生成一个 $k+1$ 阶下三角邻接矩阵 $\mathbf{D}$, 即得一个候选 $(k+1)$ -树。连接算法描述如下。

算法 1. $\mathbf{D} = \text{Join}(\mathbf{A}, \mathbf{B})$

输入: 两个共核的 k 阶 CAM 矩阵 $\mathbf{A}$ 和 $\mathbf{B}$,

输出: $k+1$ 阶下三角邻接矩阵 $\mathbf{D}$ 。

1. $\mathbf{D}(1:k-1, 1:k-1) \leftarrow \mathbf{A}(1:k-1, 1:k-1)$; // $\mathbf{D}$ 的前 $k-1$ 行为两者相同的核。

2. $\text{IFcode}(\mathbf{A}) \text{code}(\mathbf{B})$ Then

3. $\mathbf{D}(k, :) \leftarrow \mathbf{A}(k, :)$; // $\mathbf{D}$ 的第 k 行取 $\mathbf{A}$ 的第 k 行。

4. $\mathbf{D}(k+1, :) \leftarrow [\mathbf{B}(k, 1:k-1), 0, \mathbf{B}(k, k)]$; // $\mathbf{D}$ 的第 $k+1$ 行是 $\mathbf{B}$ 的第 k 行。

5. Else

6. $\mathbf{D}(k, :) \leftarrow \mathbf{B}(k, :)$; // $\mathbf{D}$ 的第 k 行取 $\mathbf{B}$ 的第 k 行。

7. $\mathbf{D}(k+1, :) \leftarrow [\mathbf{A}(k, 1:k-1), 0, \mathbf{A}(k, k)]$; // $\mathbf{D}$ 的第 $k+1$ 行是 $\mathbf{A}$ 的第 k 行。

8. End.

显然, 算法 1 的时间复杂度为 $O(k^2)$, 且输出的 $k+1$ 阶下三角邻接矩阵 $\mathbf{D}$ 是 CAM 矩阵。

4.2 扩展

除了连接操作外, 对于给定的频繁子图, 还可以通过边的扩展生成更大的候选子图。

定义 8 设 G 是图数据集 GD 的子图, e 是图数据集 GD 的边, 若 e 不是 G 的边但至少有一个端点是 G 的顶点, 则称 e 是 G 的外部关联边。两个端点都是 G 的顶点的外部关联边称为 G 的内边。仅有一个端点是 G 的顶点的外部关联边称为 G 的外边。

算法 2. $\mathbf{B} = \text{Expansion}(\mathbf{A}, e)$

输入: k 阶 CAM 矩阵 $\mathbf{A}$, 外边 e 且与 $\mathbf{A}$ 的最后一个顶点关联。

输出: $k+1$ 阶下三角邻接矩阵 $\mathbf{B}$ 。

1. $\mathbf{B}(1:k, 1:k) \leftarrow \mathbf{A}$; // $\mathbf{B}$ 的前 k 行与 $\mathbf{A}$ 相同。

2. $\mathbf{B}(k+1, :) \leftarrow 0$; // 求 $\mathbf{B}$ 的第 $k+1$ 行。

3. $\mathbf{B}(k+1, k:k+1) \leftarrow [l(e), l(v)]$;

算法 2 的外边扩展操作是 $k+1$ 阶矩阵的赋值操作, 其时间复杂度不超过 $O(k^2)$ 。

5 极大频繁子树挖掘算法

对于已完成规范化编码的图数据集 GD , 其频繁边集为 C , 且 C 中的频繁边按规范化编码降序排列。我们给出极大频繁子树挖掘算法 (Maximal

Frequent SubTree mining algorithm, MFST)。

算法 3 极大频繁子树挖掘算法 MFST

$\text{MFTSet } T = \text{MFST}(GD, C)$

输入: 图数据集 GD , 频繁边集 C ;

输出: 极大频繁子树集 T 。

1. $T \leftarrow \emptyset$; $S \leftarrow C$;

2. While $S \neq \emptyset$ Do

3. $R \leftarrow \{ \text{Join}(s_1, s_2) \mid s_1, s_2 \in S \text{ 且 } s_1 \text{ 与 } s_2 \text{ 共核} \}$;

4. $R \leftarrow R \cup \{ \text{Extension}(s, e) \mid s \in S \wedge e \in C \wedge e \text{ 是与 } s \text{ 最后一个顶点关联的外边} \}$;

5. $R \leftarrow$ 去掉 R 中的冗余模式;

6. $S \leftarrow$ 去掉 S 中已被更新的频繁子树;

7. $T \leftarrow T \cup S$;

8. $S \leftarrow \{ r \mid r \in R \text{ 且在 } GD \text{ 中是频繁子树} \}$;

9. Return T

定理 3 对于含有 n 个顶点的图数据集 GD , MFST 算法能在 $O(2^{n \log n})$ 时间内挖掘出它的所有极大频繁子树。

证明: 首先证明 MFST 算法的正确性。因为子树最大不会超过 n 个顶点 $n-1$ 条边, 而算法进入下一轮循环后的候选子树都会增加一个顶点一条边, 所以循环一定会终止。

记 S_k 为 GD 中所有的频繁 k -子树的集合, 初始时 $k=2, S_2=C$, 即所有的频繁边 (频繁 2-子树)。又记 R_{k+1} 为从 S_k 中的频繁 k -子树通过连接和扩展 (步骤 3 和步骤 4) 得到的候选 $(k+1)$ -子树。设 t 是 GD 的任意一个频繁 $(k+1)$ -子树, 为了能够挖掘到它, 在 R_{k+1} 中必须要有候选子树 t 。设 t 的 CAM 矩阵为 $\mathbf{M}$, 它的最后两行 (第 $k, k+1$ 行) 分别对应顶点 u, v , 即 $m_{k, k} = l(u), m_{k+1, k+1} = l(v)$, 第 $k+1$ 行边元素 $m_{k+1, k}$ 表示 t 是否含有顶点 u, v 之间的连边。根据 $m_{k+1, k}$ 的取值分两种情况讨论。

(1) 若 $m_{k+1, k} = 0$, 即没有顶点 u, v 之间的连边。由 t 的连通性知, 这时一定存在两个连通频繁 k -子树 t_A, t_B , 且 $t_A = t - \{v\}, t_B = t - \{u\}$ 。 t_A 的下三角邻接矩阵 $\mathbf{A}$ 是 $\mathbf{M}$ 的前 k 行, 根据定理 3.1(3) 知它是一个 CAM 矩阵。 t_B 的下三角邻接矩阵 $\mathbf{B}$ 则由 $\mathbf{M}$ 的前 $k-1$ 行和第 $k+1$ 行组成。如果在 t_B 的 CAM 矩阵中顶点 v 不在最后一行, 则在 t_A 的 CAM 矩阵中顶点 u 也不在最后一行, 从而与 $\mathbf{A}$ 是 CAM 矛盾。因此, t 的 CAM 矩阵能够由连接操作得到, 即

$$\mathbf{M} = \text{Join}(\mathbf{A}, \mathbf{B}).$$

(2) $m_{k+1, k} = l(u, v) > 0$ 时, 即在 t 中顶点 u, v 之间有连边, 则 v 与其它顶点无连边。这时一定存在频繁 k -子树 t_A 且以 $e = (u, v)$ 为外边, 子树 t_A 以 $\mathbf{M}$ 的前 k 行为一个下三角邻接矩阵, 记为 $\mathbf{A}$, 根据定理 3.1(3) 知 $\mathbf{A}$ 是 CAM 矩阵。因此 t 的 CAM 矩阵可由

外边扩展操作得到,即 $M = Expansion(A, e)$.

下面分析算法的时间复杂度,并证明算法时间复杂度为 $O(2^{n \log n})$,其中 n 是图数据集 GD 的顶点个数. 记 S_k 为算法中第 k 轮循环得到的所有频繁 k -子树的集合(算法中步骤 8); R_{k+1} 为从 S_k 中的频繁 k -子树通过连接和扩展(算法中的步骤 3 和步骤 4)得到的所有候选 $(k+1)$ -子树.

算法第 3 步是针对 S_k 中所有共核的 CAM 对 (s_1, s_2) 作连接. 因为核是 $(k-1)$ 阶 CAM,所以不超过 C_n^{k-1} 个,每个核外剩余 $n-k+1$ 个顶点,所以同一个核最多不超过 C_{n-k+1}^2 次连接操作.

算法第 4 步是针对 S_k 中所有的 k 阶 CAM 作外边扩展. 已经有文献证明^[8], k 个顶点的简单无向图最多有 k^{k-2} 个生成树. 因此 S_k 中的频繁 k -子树(CAM)个数最多不超过 $C_n^k \cdot k^{k-2}$.

算法第 5 步是针对 R_{k+1} 中所有的 $k+1$ 阶候选模式找出冗余模式并清除. 对于同一个 $k+1$ 顶点集,通过连接和扩展所得到的下三角矩阵最多不超过 $(k+1)!$ 个,比较它们的编码,留下最大编码其余删除. 所以花费时间不超过 $(k+1)^2 \cdot C_n^{k+1} \cdot (k+1)!$.

算法第 6 步是去掉 S_k 中已被更新的频繁子树,通过作标记来实现. 花费时间不超过 S_k 的大小. 第 7 步类似,两步花费时间不超过 $2C_n^k \cdot k^{k-2}$.

算法第 8 步是对 R_{k+1} 中的所有候选模式进行子图同构测试. 先求图数据库 GD 中的 $k+1$ 个顶点的嵌入集,然后进行图同构测试并统计支持度. 总共花费时间不超过 $(k+1)^2 n^2 \cdot C_n^{k+1} \cdot (k+1)^{k-1}$.

综合上述对第 k 轮循环各步骤的时间分析, k 从 2 到 $n-1$ 求和,即得算法总时间复杂度为 $O(2^{n \log n})$.

6 实验分析

本文实验采用化学分子数据集,将 MFST 与目前认为快速挖掘频繁子树算法 FSM_CGTG^[8] 进行了比较. 利用 FSM_CGTG 算法的结果对 MFST 在真实化学数据集中得到的极大频繁子树进行交叉验证.

实验数据来源致癌物质化学分子结构的 carcinogen 数据库,该数据库从牛津大学的网站上获得. carcinogen 包含的 337 个化合物的分子结构以 2 个文件形式存储. 文件 atoms.pl 定义了所有化合物中的原子的定义. 如: $atm(d1, d1_1, c, 22, -0.133)$ 表示一个化合物 $d1$ 有一个标记为 $d1_1$ 的 c 原子,原

子类型为 22, 偏电荷为 -0.133 . 另一个文件 bonds.pl 提供了化合物中原子间的键信息. 如: $bond(d1, d1_1, d1_2, 7)$ 表明在化合物 $d1$ 中,它的原子 $d1_1$ 和 $d1_2$ 通过键类型为 7 的键连接. 其中有 4 种键类型分别为 1、2、3 和 7, 66 种原子类型,共有 24 种原子. 对每一个化合物,本文没有考虑原子的类型和偏电荷信息,原子对应图中的一个顶点,键类型对应图中的一条边,这样得到了一个具有 337 个化合物分子的图数据库. 其中最小的图大小为 2 个顶点,最大图的大小为 214 个顶点.

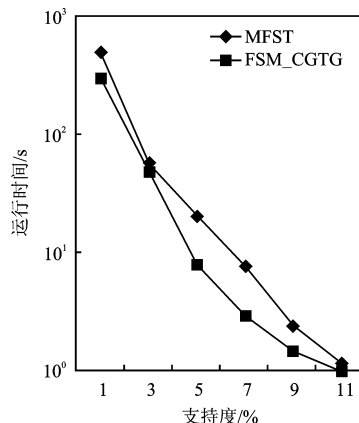


图 1 两个算法时间性能比较

图 1 使用 carcinogen 数据库对 MFST 和 FSM_CGTG 的实验性能进行比较. 图 1 中可以看出,在支持度从 1 到 3 变化过程中,时间性能方面 MFST 比 FSM_CGTG 算法提高了近 2 倍. 图 2 可以看出,与 FSM_CGTG 相比, MFST 挖掘极大频繁子图的总数减少 2 个数量级, FSM_CGTG 算法既能保证频繁子树的完整性,又能节省存储空间,提高了空间性能. 为了较好地显示数量与支持度的关系,图 1 和图 2 的纵坐标使用了以基数为 10 的对数刻度.

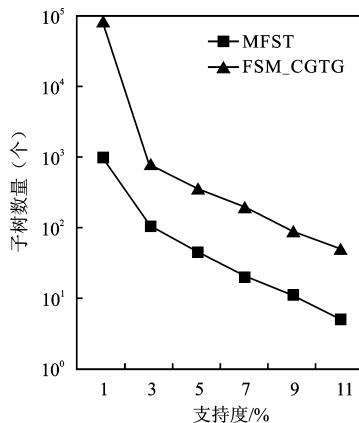


图 2 两个算法空间性能比较

7 结束语

本文提出了一种新的从图数据集中挖掘极大频繁子树 MFST 算法,从连接和扩展操作得到极大频繁子树. 与目前算法先比较, MFST 算法的优点在于将算法的整体效率提高了,整个算法的时间复杂度不超过 $O(2^{n \log n})$. 其原因主要来自于两个方面,一方面是利用连接和扩展操作,求极大频繁子树个数时间不超过 $O(2^{n \log n})$;另一方面是利用计算嵌入集解决子树同构问题,将时间复杂性由指数级转化为多项式 $O(n^2)$,这两个方面正是频繁子树挖掘算法影响复杂度的关键部分,最后的实验也证明了这一点.

参考文献:

[1] HAN K, LV W, YIN B, et al. Constrained frequent subtree mining method [C]// Luo X N, Luo Z X. Digital Home, Better Life: Proceedings of the 2014 5th International Conference on Digital Home. Washington, DC: IEEE Computer Society, 2014: 287-292.

[2] TANG D Q, TAN Y. Algorithm for frequent subgraph mining [J]. Computer Engineering, 2012, 38 (7): 31-33.

[3] AKSHAY D, DAVID F B, SRIKANTA T, et al. EvoMiner: frequent subtree mining in phylogenetic databases [J]. Knowledge and Information Systems,

2014, 41(3):559-590.

[4] ZHANG S, DU Z, WANG J T. New techniques for mining frequent patterns in unordered trees [J]. IEEE Transactions on Cybernetics, 2015, 45(6):1113-1125.

[5] WANG J, LIU Z, LI W, et al. Research on a frequent maximal induced subtrees mining method based on the compression tree sequence [J]. Expert Systems with Applications, 2015, 42(1):94-100.

[6] YANG P, TAN Q. Maximum frequent tree mining and its applications [J]. Computer Science, 2008, 35 (2):150-153.

[7] RON S, DEKEL T. Faster subtree isomorphism [J]. Journal of Algorithms, 1999, 33(2):267 - 280.

[8] PETER W S. A New Proof of Cayley's Formula for Counting Labeled Trees [J]. Journal of Combinatorial Theory, Series A. 1995, 71(1):154-158.

[9] TANG D Q, XIE W J. A new frequent subtree mining algorithm research and implementation [J]. Computer Applications and Software, 2012, 29 (4): 174-177

作者简介:

唐德权 男, (1979-), 博士研究生, 副教授. 研究方向为数据挖掘、信息安全.

黄金贵(通讯作者) 男, (1964-), 博士, 教授, 博士生导师. 研究方向为算法复杂性与 NP 优化、深度学习与大数据算法及应用. E-mail: hjpg@hunnu. edu. cn