doi:10.3969/j.issn.1005-3026.2017.05.004

一种面向图集合的相似性搜索技术

庞 俊,谷 峪,于 戈
(东北大学 计算机科学与工程学院,辽宁 沈阳 110819)

摘 要:目前图相似性的研究工作主要集中在子图的匹配,而没有充分关注图集合之间的匹配.针对这一问题,提出了一种基于过滤-求精框架的GSSS算法;提出了一种图集合距离定义,设计了Number,Size,Complete edge和Lower bound过滤器减小搜索空间,优化了图集合距离的计算;设计并优化了一种增量式的多层倒排索引,提高了查询效率,适应数据集的动态变化.真实数据集上的大量实验验证了GSSS算法的有效性和高效性.

关 键 词:图集合;相似性;搜索;索引;过滤
中图分类号:TP 311 **文献标志码**:A **文章编号**:1005-3026(2017)05-0625-05

A Similarity Search Technique for Graph Set

PANG Jun, *GU Yu*, *YU Ge*
(School of Computer Science & Engineering, Northeastern University, Shenyang 110819, China. Corresponding author: YU Ge, professor, E-mail: yuge@ise.neu.edu.cn)

Abstract: Existing studies of graph similarity search mainly focus on the subgraph matching instead of the graph set matching. To tackle this issue, GSSS algorithm was proposed based on filtering-and-verify framework. A graph set distance was defined. In order to reduce the search space, Number filter, Size filter, Complete edge filter and Lower bound filter were proposed. Then, the computation of the graph set distance was optimized. An incremental multi-layer inverted index was designed to further improve the query efficiency. Extensive experiments on a real-world dataset show that GSSS algorithm is effective and efficient.

Key words: graph set; similarity; search; index; filter

图集合是一个包含若干个图的集合.图集合相似性搜索指从一个由诸多图集合构成的数据集中查找出与给定查询图集合相似的所有图集合.图集合相似性搜索有重要的实际应用.例如:查询药效相似的药物.药物是一种由多种成分组成的混合物.每种成分由一种化学分子组成.一个化学分子可以表示为一个图.于是,一种药物可以表示为一个图的集合.因此,查找药效相似的药物,也就是搜索相似的图集合.虽然图集合相似性搜索具有重要的实际意义,但目前在图数据管理领域还没有得到充分的关注.现有图相似性搜索的相关研究主要分为两类:①从一个图数据集中找出所有与给定查询图相似的图[1-6];②从一个大图中找出所有满足条件的子结构[7-8].与图集合的相似性搜索问题不同,图相似性搜索的查询对象是图而不是图集合,计算过程中一般采用定义在图数据之间的编辑距离而不是图集合之间的距离.显然,因为图集合是一种新的数据表示形式,所以需要一种新的距离定义来表示两个图集合之间的相似性;此外,因为数据类型不同、相似性定义不同,现有的相似性搜索算法[1-11]也不能直接用于解决图集合的相似性搜索问题.

针对以上问题,本文提出一种图集合距离定义,并提出一个新的方法来解决图集合相似性搜索问题.该方法采用过滤-求精框架:在过滤阶段,基于本文提出的多个过滤器获取较小的候选

收稿日期:2015-12-10
基金项目:国家重点基础研究发展计划项目(2012CB316201);国家自然科学基金资助项目(61272179, 61472071).
作者简介:庞 俊(1984-),男,湖北咸宁人,东北大学博士研究生;谷 峪(1981-),男,辽宁鞍山人,东北大学教授,博士生导师;于 戈(1962-),男,辽宁大连人,东北大学教授,博士生导师.

结果集;在求精阶段,采用剪枝策略加速对候选结果集的求精,得到精确查询结果.

1 图集合距离和问题定义

定义 1(图集合) 图集合指一个无向有权简单连通图的集合,表示为 $G=\{g_1, g_2,\cdots,g_n\}$.

定义 2(二分图最小完美匹配) 定义为满足如下条件的二分图的边集合的一个子集:①任意两条边不相交;②覆盖了二分图的所有顶点;③边的权重之和最小. 二分图是一种满足如下条件的图,它的顶点集 V 可以划分为两个子集 X 和 Y,它的每条边的两个端点只能分别属于 X 和 Y 集合,不能同时属于 X 或者 Y 集合.

定义 3(图集合对的完全二分图表示) 已知两个图集合 d_1 和 d_2, d_1 和 d_2 可以构成这样一个完全二分图. d_1 和 d_2 分别对应二分图的 X 和 Y 集合. d_1 或 d_2 中的每个图对应 X 或 Y 中的一个顶点. 该完全二分图任意一条边所连接的两个顶点所对应的图之间的编辑距离是该边的权重. 该完全二分图称之为 d_1 和 d_2 的完全二分图,记作 $\mathrm{CBG}(d_1, d_2)$. 如果 d_1 和 d_2 的基数不相等,那么在基数较小的图集合中补充虚拟图,使得两个图集合的基数相等.

定义 4(图集合距离) 图集合 d_1 和 d_2 的距离 $\mathrm{GSD}(d_1, d_2)$ 是 d_1 和 d_2 的完全二分图的最小完美匹配.

定义 5(图集合相似性搜索) 已知图集合数据集 D 和查询图集合 q,图集合相似性搜索的目的是从 D 中找出所有与 q 的距离小于或等于一个给定阈值 t 的图集合 $R=\{d \mid \mathrm{GSD}(q,d)\leqslant t, d\in D\}$.

因为图的编辑距离计算是 NP - Hard 问题,所以本文提出的图集合距离计算是很复杂的. 为了提高查询效率,本文提出的算法采用了过滤 - 求精框架.

2 过滤和求精

2.1 算法框架

本文提出一个新的算法(GSSS)来解决图集合相似性搜索问题. 该算法采用过滤 - 求精策略,先对图集合数据集进行过滤,消除一些一定不是最终结果的(true negative)数据,得到一个包含了所有最终结果的候选结果集. 因为该候选结果集中包含了一些非最终(false positive)结果,所以计算候选结果集中每个图集合 d 与查询图集合 q 之间的图集合距离. 如果该距离不超过给定阈值 t,则 d 属于结果集;否则,d 不属于结果集.

2.2 过滤

为了减少搜索空间,本文提出了 Number filter, Size filter, Complete edge filter 和 Lower bound filter. 下文依次介绍这些过滤器,并证明它们的正确性.

已知两个图集合 d_1 和 d_2, $|d_1|=n$, $|d_2|=m$, $n\geqslant m$, d_1 和 d_2 构成的完全二分图 $\mathrm{CBG}(d_1,d_2)$ 和给定阈值 t.

定理 1(Number filter) 如果 d_1 和 d_2 相似,那么它们的基数之差的绝对值一定要小于或等于 t,即 $n-m\leqslant t$.

证明 根据已知条件和图集合距离定义可知,d_1 比 d_2 多的 $n-m$ 个图对它们的图集合距离的贡献至少为 $n-m$,也就是 $\mathrm{GSD}(d_1,d_2)\geqslant n-m$. 显然,如果 $\mathrm{GSD}(d_1,d_2)\leqslant t$,则 $n-m\leqslant t$.

定理 2(Size filter) 如果 d_1 和 d_2 相似,那么 d_1 中的 $n-m$ 个最小图的大小之和一定小于或等于 t.

证明 根据已知条件、图集合距离定义和 Number counting bound[1] 可知, d_1 比 d_2 多的 $n-m$ 个图对它们的图集合距离的贡献的下界是 d_1 中的 $n-m$ 个最小图的大小之和. 同理,定理 2 成立.

本文将 $g\in d_1$(或 d_2)在 $\mathrm{CBG}(d_1,d_2)$ 中的最小边距离定义为 g 对应顶点的所有邻接边的最小的边权,记作 $\mathrm{MED}(g)$,此外,d_1 和 d_2 的完全边距离定义为 d_1 中所有的图对应顶点的最小边距离之和. 记作:$\mathrm{CED}(d_1,d_2)=\sum(\mathrm{MED}(g)), g\in d_1$. 同理 d_2 和 d_1 的完全边距离 $\mathrm{CED}(d_2,d_1)=\sum(\mathrm{MED}(g)), g\in d_2$.

定理 3(Complete edge filter) 如果 d_1 和 d_2 相似,那么 $\max\{\mathrm{CED}(d_1,d_2), \mathrm{CED}(d_2,d_1)\}\leqslant t$.

证明 根据已知条件、图集合距离定义和完全边距离定义,不难推导出 $\mathrm{CED}(d_1,d_2)$ 和 $\mathrm{CED}(d_2,d_1)$ 是 d_1 和 d_2 的图集合距离的下界. 因此,如果 d_1 和 d_2 相似,那么 $\max\{\mathrm{CED}(d_1,d_2), \mathrm{CED}(d_2,d_1)\}\leqslant t$.

引理 1 假设 $\mathrm{CBG}(d_1,d_2)$ 的最小完美匹配的匹配数为 PM_1. 用 $\mathrm{CBG}(d_1,d_2)$ 中每条边的权重下界取代真实边权重得到一个新的完全二分图 $\mathrm{CBG}'(d_1,d_2)$,假设 $\mathrm{CBG}'(d_1,d_2)$ 最小完美匹配的匹配数为 PM_2,则 $\mathrm{PM}_2\leqslant\mathrm{PM}_1$.

证明 下面分两种情况进行证明. ①假设替

换后，最小完美匹配不变. 由于匹配中每条边的权重不变或者变小了，所以 $PM_2 \leqslant PM_1$. ②假设替换后，最小完美匹配发生改变. 换言之，$PM_2 < PM_1$. 综上所述，$PM_2 \leqslant PM_1$.

GSSS 算法采用如下图编辑距离下界的最大值作为图集合对构成二分图的边的权重下界：Number counting bound[1]，Label multiset bound[3] 和 Compact branch bound[6].

定理 4（Lower bound filter）：如果 d_1 和 d_2 相似，那么 $PM_2 \leqslant t$.

证明　由引理 1，可知 $PM_2 \leqslant PM_1$. 如果 d_1 和 d_2 相似，那么 $PM_1 \leqslant t$. 因此，$PM_2 \leqslant t$.

根据计算代价小优先使用的原则，按如下次序使用过滤器：Number filter，Size filter，Complete edge filter 和 Lower bound filter.

2.3　求精

求精阶段使用的剪枝策略的基本思想是：在求解完全二分图的最小完美匹配过程中，如果可以判断最小完美匹配的上界小于或等于给定的阈值，就提前终止计算，判定对应的两个图集合一定相似.

目前计算二分图最小完美匹配的最优算法是扩展的 KM 算法. KM 算法是求解二分图的最大完美匹配问题的算法，进行如下扩展可用于求解二分图最小完美匹配问题：先将边的权重取负数，接着使用 KM 算法计算出最大完美匹配，最后取该最大完美匹配的匹配数的负数，即所求的最小完美匹配的匹配数. KM 算法的流程如下：

1）设置二分图的顶点标记，使得如下结论始终成立：边权重始终小于或等于两个邻接点的标记之和.

2）使用匈牙利算法查找同等子图中的完美匹配. 如果找到，则找到的完美匹配为所求；否则，修改顶点标记，扩展同等子图.

3）循环 1）和 2），直至找到完美匹配为止.

定理 5　已知两个图集合 d_1 和 d_2，以及给定的阈值 t. 在查找同等子图的完美匹配的过程中，如果当前匹配的边权之和 $CMS(d_1, d_2)$ 大于或等于 $-t$，那么 $GSD(d_1, d_2) \leqslant t$.

证明　根据扩展的 KM 算法，$GSD(d_1, d_2) \leqslant t$，也就是对应的最大完美匹配的匹配数 $MAXPM \geqslant -t$. 因此，证明定理 5 成立，即证明如下结论成立：如果 $CMS(d_1, d_2) \geqslant -t$，那么 $MAXPM \geqslant -t$. 下面分 2 种情况讨论：①假设当前匹配是完美匹配，那么 $MAXPM = CMS(d_1, d_2) = -t$. ②假设当前匹配不是完美匹配，那么完美匹配将在后继步骤中找到. 根据匈牙利算法可知，后继匹配的边权之和大于当前匹配的边权之和. 因此，$MAXPM > CMS(d_1, d_2)$. 根据假设可知，$CMS(d_1, d_2) \geqslant -t$. 显然，$MAXPM > -t$. 综上所述，定理 5 成立. 定理 5 不仅可以帮助 GSSS 算法在求精阶段快速地找到最终结果，还可以加快 Lower bound filter 的计算.

2.4　索引

最基本的查询策略是将查询图集合与每个数据图集合依次进行过滤和求精，从而得到最终查询结果. 这种策略的查询效率很低. 当数据图集合数量非常多的时候，低效问题尤为突出. 为了提高查询效率，设计了一个多层倒排索引. 下文先介绍与多层倒排索引相关的基本概念，然后分别讨论该索引的组织结构、存储优化、构造方法、动态维护和查询方法.

本文将图信息单元定义为一个三元组 $GIU(g) = <$ graphId，size，branches $>$. 其中，graphId 指 g 的唯一标识，size 指 g 的顶点数和边数之和，branches 指 g 的所有 branch 结构. 一个 branch 由 g 中的一个顶点 v 和 v 的邻接边组成[6]. 此外，一个图集合 d 的信息单元由该集合的唯一标识和 d 中所有的图的信息单元组成，记作 $GSIU(d)$. 多层倒排索引的键是图集合的基数 $|d|$，列表中包括的是所有基数为 $|d|$ 的图集合的信息单元，如图 1 所示.

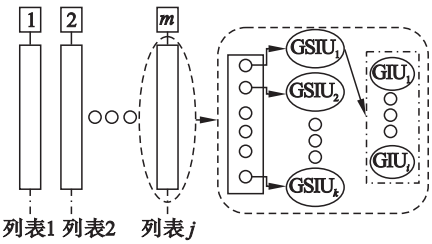

图 1　多层倒排索引示意图

Fig. 1　Illustration diagram of multi-layer inverted index

进一步，为了减少索引的存储空间，本文采用了将图的信息单元单独存储的策略. 因为不同的图集合可以包含相同的图，所以不同图集合的信息单元可能包含相同的图信息单元. 因此，构造一个哈希表来专门存储所有图的信息单元. 这样就避免索引中多次存储相同的图信息单元，从而优化了存储空间. 为了提高查询效率，离线建立索引. 该索引的构造方法与倒排索引的构造方法相类似. 限于篇幅，不对其构造方法进行详细的介绍.

多层倒排索引可以通过增量式的维护来适应数据集合的动态变化,从而避免了索引的重新建立. 当增加了一个新的数据集合 d_1,如果 d_1 中图的信息单元在哈希表中不存在,则添加进哈希表,并在倒排索引中添加 d_1 的集合信息单元. 当删除一个数据集合 d_2,则只需要删除多层倒排索引中 d_2 的集合信息单元即可. 类似地,当数据集合 d_1 中增加(或减少)了图 g,则相应地更新 d_1 对应的集合信息单元和哈希表.

GSSS 的查询算法如算法 1 所示. 给定一个查询图集合 q 和一个距离阈值 t,先根据 Number filter 计算出与 q 相似的数据图集合 d 的基数的范围. 然后,与多层倒排索引中的键相匹配,找到 Number filter 过滤后的候选结果集 1. 接着,对候选结果集 1 中的每个图集合,依次使用 Size filter,Complete edge filter 和 Lower bound filter 得到候选结果集 2. 最后,采用优化过的 KM 算法验证候选结果集 2,返回最终的查询结果. GSSS 算法流程描述如图 2 所示.

算法 1:GSSS 查询算法

输入:图集合数据集 D,查询图集合 q 和距离阈值 t.

输出:$R = \{d \mid \mathrm{GSD}(q,d) \leqslant t, d \in D\}$.

1) 建立 D 的多层倒排索引;

2) 根据 Number filter,q 和 t,查询第一步建立的多层倒排索引,得到候选结果集 1;

3) 依次使用 Size filter,Complete edge filter 和 Lower bound filter 对候选结果集 1 进行过滤,得到候选结果集 2;

4) 使用优化的 KM 算法验证候选结果集 2,返回最终的查询结果.

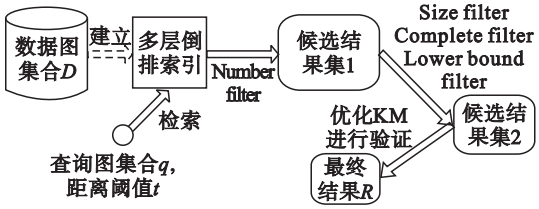

图 2 GSSS 算法流程示意图

Fig. 2 Flow diagram of GSSS algorithm

3 性能评价

在真实数据集上比较 GSSS,Naive 算法和扩展的 M 算法[6]的在线查询时间,并研究 GSSS 算法的索引构造时间和空间. 每组进行了 3 次实验,取均值作为实验结果. 下面分别报告使用的真实数据集、实验配置和实验结果.

NCI 化合物数据集:一个分子化合物可以表示成一个图. 原子表示为图的顶点,化学键表示为图的边. ID 为 1 的 NCI 化合物数据集共包含 42 490个分子化合物. 使用文献[12]实验提出的方法构造了 50 000 个分子组.

本文所有实验均在单台计算机上执行. 计算机配置如下所示:Linux 操作系统,2.93 GHz CPU,4GB RAM.

3.1 离线索引构造时间和空间

本节测试了多层倒排索引的构造时间和空间. 试验结果如图 3 和图 4 所示. 图 3 和图 4 分别显示:随着数据图集合的增大,多层倒排索引的构造时间和空间呈线性增长.

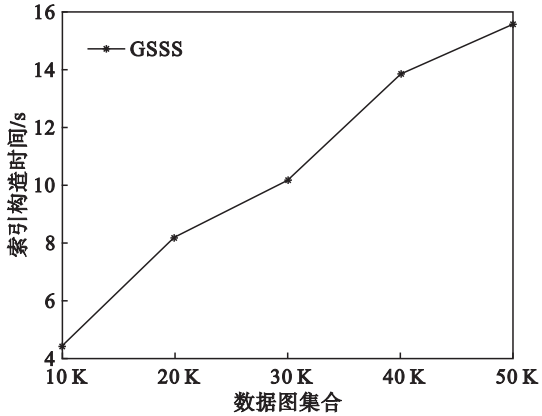

图 3 多层倒排索引构造时间

Fig. 3 Building time of multi-layer inverted index

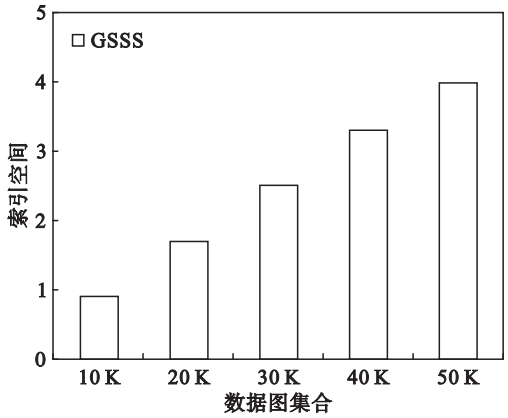

图 4 多层倒排索引空间

Fig. 4 Size of multi-layer inverted index

3.2 在线查询时间

本节比较了 GSSS 算法、Naive 算法和扩展的 M 算法(M^*)的在线查询时间. 结果如表 1 所示($|D| = 30$ K). N,S,C 和 L 分别表示采用 Number filer,Size filter,Complete edge filter 和 Lower bound filter 的 GSSS 算法. N + S + C + L 表示采用所有过滤器的 GSSS 算法. Naive 算法、M^* 算法、N 和 N + S 均未在 8 h 之内完成. 当距离

阈值等于 5，N + S + C + L 至少比 Naive 算法和 M* 算法快 150 倍；当距离阈值等于 20，N + S + C + L 至少比 Naive 算法和 M* 算法快 4 倍. 显然，GSSS 算法比 Naive 算法和 M* 算法要快. 因为 GSSS 算法采用了多个过滤器，并且采用了优化的 KM 算法. 同时，GSSS 算法采用的过滤器越多，在线查询时间越少. 因为过滤器越多，过滤掉的非最终结果越多.

表 1　采用不同过滤器的在线查询时间

Table 1　Online response time by using different filters　　s

阈值	5	10	15	20
Naive	> 28 800	> 28 800	> 28 800	> 28 800
M*	> 28 800	> 28 800	> 28 800	> 28 800
N	> 28 800	> 28 800	> 28 800	> 28 800
N + S	> 28 800	> 28 800	> 28 800	> 28 800
N + S + C	288. 7	1 921. 3	5 089. 7	9 410. 3
N + S + C + L	192. 7	1 729. 3	4 321. 8	7 488

3.3　不同阈值的性能

不同距离阈值对 GSSS 算法过滤性能的影响（$|D| = 30$ K），如图 5 所示. 随着距离阈值的增大，各组合过滤器的过滤效果变弱. 因为随着距离阈值的增大，Number filter，Size filter，Complete edge filter 和 Lower bound filter 的过滤效果均变弱. 根据定理 1，被 Number filter 过滤掉的图集合在距离阈值增大的情况下可能通过 Number filter. 因此，随着距离阈值的增大，Number filter 的过滤效果变弱. 同理，随着距离阈值的增大，Size filter，Complete edge filter 和 Lower bound filter 的过滤效果也变弱. 同时，对于同一个阈值，采用更多过滤器的 GSSS 算法的过滤效果更好. 因为对于同一个阈值，这些过滤器的过滤效果呈递增趋势.

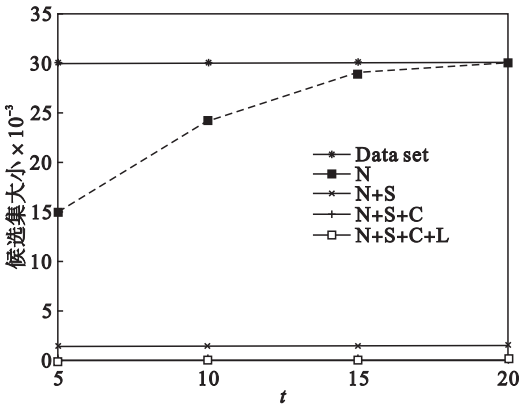

图 5　不同阈值的过滤效果

Fig. 5　Filtering effect of different distance thresholds

4　结　　论

本文采用一种新的数据类型（图集合）来表示对象，给出一种图集合间的距离定义，并提出了一个基于过滤 - 求精策略的 GSSS 算法来解决图集合相似性搜索问题；为了提高搜索效率，提出了一个增量式的多层倒排索引和多个下界剪枝策略，并优化了 KM 算法. 实际数据集上的大量实验证明了该算法的有效性和高效性.

参考文献：

[1] Zheng Z, Tung A K H, Wang J, et al. Comparing stars: on approximating graph edit distance [C]//Proceedings of the VLDB Endowment. Lyon, 2009: 25 - 36.

[2] Wang G R, Wang B, Yang X C, et al. Efficiently indexing large sparse graphs for similarity search [J]. *IEEE Transactions on Knowledge and Data Engineering*, 2012, 24(3): 440 - 451.

[3] Zhao X, Xiao C, Lin X M, et al. Efficient graph similarity joins with edit distance constraints [C]//Proceedings of the 28th International Conference on Data Engineering. Washington D C, 2012: 210 - 221.

[4] Zhao X, Xiao C, Lin X M, et al. A partition-based approach to structure similarity search [C]//Proceedings of the VLDB Endowment. Hangzhou, 2013, 7(3): 169 - 180.

[5] Pang J, Gu Y, Xu J, et al. Efficient graph similarity join with scalable prefix-filtering using MapReduce [C]//Proceedings of the 15th International Conference on Web-Age Information Management. Macau, 2014: 415 - 418.

[6] Zheng W G, Zou L, Lian X, et al. Efficient graph similarity search over large graph databases [J]. *IEEE Transactions on Knowledge and Data Engineering*, 2015, 27(4): 964 - 978.

[7] Li R H, Qin L, Yu X J, et al. Influential community search in large networks [C]// Proceedings of the VLDB Endowment. Hawaii, 2015, 8(5): 509 - 520.

[8] Huang X, Cheng H, Li R H, et al. Top-*k* structural diversity search in large networks [J]. *International Journal on Very Large Data Bases*, 2015, 24(3): 319 - 343.

[9] Xu J, Zhang Z J, Anthony K H, et al. Efficient and effective similarity search over probabilistic data based on earth mover's distance [C]//Proceedings of the VLDB Endowment. Singapore, 2010, 3(1): 758 - 769.

[10] Li G L, Deng D, Feng J H. A partition-based method for string similarity joins with edit-distance constraints [J]. *ACM Transactions on Database Systems*, 2013, 38(2): 721 - 741.

[11] Yang X C, Wang Y S, Wang B, et al. Local filtering: improving the performance of approximate queries on string collections [C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. Melbourne, 2015: 377 - 392.

[12] Wu J, Zhu X Q, Zhang C Q, et al. Bag constrained structure pattern mining for multi-graph classification [J]. *IEEE Transactions on Knowledge and Data Engineering*, 2014, 26(10): 2382 - 2396.