魏硕, 杜明, 周军锋. 一种高效的 (p,q) 二分团计数算法[J]. 智能计算机与应用,2026,16(3):1-9. DOI:10.20169/j.issn.2095-2163.24050702

一种高效的(p,q)二分团计数算法

魏 硕, 杜 明, 周军锋
(东华大学 计算机科学与技术学院, 上海 201620)

摘 要: 二分图可以对 2 种不同类型实体之间的关系进行建模。二分图中的团称为二分团,是其基本的稠密子结构。计算给定二分图中 (p,q) 二分团的个数具有十分重要的意义。针对现有二分团计数方法因输入图规模较大所导致的低效性问题,本文提出针对 (p,q) 二分团计数的优化方法,该方法通过构建二分图中所有不相交最大二分团的索引 MBC_I,可以显著减少二分团计数方法输入图的规模,提升 (p,q) 二分团计数的效率。之后,提出基于度的整体删减策略,并基于此提出优化计数算法 CNBC_I*,来进一步提高计数效率。最后,在多个数据集上进行了实验,实验结果验证了本文方法的高效性。

关键词: 二分图; (p,q) 二分团; 索引; 最大二分团
中图分类号: O157.5 **文献标志码**: A **文章编号**: 2095-2163(2026)03-0001-09

An efficient algorithm for counting (p,q)-bicliques

WEI Shuo, DU Ming, ZHOU Junfeng
(School of Computer Science and Technology, Donghua University, Shanghai 201620, China)

Abstract: The bipartite graph can model the relationship between two different types of entities. The clique in a bipartite graph is called a biclique and is its basic dense substructure. Calculating the number of (p,q) - bicliques in a given bipartite graph is of great significance. In response to the inefficiency caused by the large input graph size of existing bicliques counting methods, this paper proposes an optimization method for (p,q) - bicliques counting. This method significantly reduces the input graph size of bicliques counting methods and improves the efficiency of (p,q) - bicliques counting by constructing the index MBC_I for all non-intersecting maximum bicliques in the bipartite graph. Afterwards, a degree based overall pruning strategy is proposed, and based on this, an optimized counting algorithm CNBC_I* is proposed to further improve counting efficiency. Finally, experiments are conducted on multiple datasets, and the experimental results validate the efficiency of the proposed method.

Key words: bipartite graph; (p,q) - biclique; index; maximum biclique

0 引 言

二分图可以对 2 种不同类型的实体之间的关系进行建模,二分图中的团(clique)称作二分团(biclique)。(p,q) 二分团是指 2 层顶点数分别为 p 和 q 的二分团。顾客-商品关系的二分图如图 1 所示,其中 U 层顶点代表顾客, V 层顶点代表商品,边代表购买关系。图 1 中存在一个(2,3)二分团 $B_{2,3}$, $U(B_{2,3})=\{u_1,u_2\}$, $V(B_{2,3})=\{v_1,v_2,v_3\}$, 表示 Sam 和 Frank 均购买了 cake、book 和 milk。

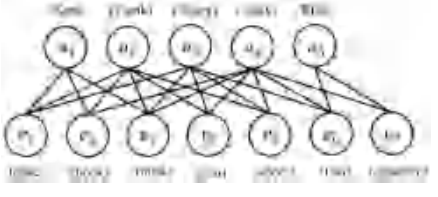

图 1 顾客-商品构成的二分图 G

Fig. 1 Bipartite graph G of customer-commodity

本文旨在解决二分图中的 (p,q) 二分团计数问题,其主要应用有:检测最密集子图,例如 (p,q) 二分团个数可以用来计算二分图的 (p,q) 二分团密

基金项目: 国家自然科学基金(62372101, 61873337, 62272097)。

作者简介: 魏 硕(2000—),女,硕士研究生,主要研究方向:密集子图挖掘; 周军锋(1977—),男,博士,教授,主要研究方向:数据挖掘,大数据与人工智能,大图数据的查询处理技术。
通信作者: 杜 明(1975—),男,博士,教授,主要研究方向:图数据管理,自然语言处理,数据分析与数据挖掘。Email: duming@dhu.edu.cn。

收稿日期: 2024-05-07

度((p,q) - biclique density),进而查找二分图中(p,q)二分团密度最大的二分子图((p,q) - biclique densest subgraph)[1]。此外,(p,q)二分团还可以分析二分图中的内聚子组(cohesive subgroup)[2]。

在(p,q)二分团中,结构最简单的是(2,2)二分团[3]。对于给定的二分图,文献[4-6]研究了(2,2)二分团计数问题,这些解决方法本质上都是通过枚举其基础的楔形子结构 wedge 实现的。楔形子结构指的是(1,2)二分团或(2,1)二分团。对于一般(p,q)二分团计数问题,因其中存在大量的 wedge,以上方法并不适用。此外,文献[7]针对(p,q)二分团计数问题,提出了一种计数方法,该方法以原始二分图作为输入,没有考虑输入图规模较大,导致计数算法效率低。

为了提升(p,q)二分团计数的效率,本文提出了基于索引 MBC_I 的(p,q)二分团计数算法 CNBC_I。该算法通过构建存储二分图中所有不相交最大二分团的索引 MBC_I,可以在计数时以索引中最大二分团的扩展子图作为输入,一方面降低了输入图的规模,另一方面可以根据最大二分团的特性来实现高效的(p,q)二分团计数。为了进一步提高计数算法的效率,提出基于度的整体删减策略,并基于此提出优化计数算法 CNBC_I*,对于给定的二分扩展子图,该策略只检查一次所有顶点,将不满足阈值条件的顶点放入待删除顶点队列中,在顶点删除之后只需要检查与其相关联的顶点,避免了冗余操作。

本文的主要贡献如下:

(1)提出存储二分图中所有不相交最大二分团的索引 MBC_I。

(2)提出基于索引 MBC_I 的(p,q)二分团计数算法 CNBC_I,在规模较小的二分扩展子图进行求解能够提升计数算法的运行效率。

(3)提出采用基于度的整体删减策略的优化计数算法 CNBC_I*,提升二分团的计数效率。

(4)在 6 个真实数据集上进行实验,实验结果表明本文提出的优化算法 CNBC_I* 可以高效地在大规模二分图上计算(p,q)二分团的个数。

1 背景知识和相关工作

1.1 相关概念和问题定义

本文将二分图建模为三元组 $G=(U,V,E)$,其中 U 和 V 表示 2 个不相交的顶点集合,$U \cap V = \varnothing$,$E=\{(u,v) \mid u \in U, v \in V\}$ 表示边的集合。

定义 1 二分团 给定二分图 G 及其子图 B,如果对 B 中任意一对顶点 u,v,其中 $u \in U(B)$,$v \in V(B)$,都存在边 $(u,v) \in E(B)$,则称 B 为二分团。

定义 2 最大二分团 给定二分图 G,其中二分团 B 的大小利用边的数量来衡量,由于二分团 B 是一个完全二分子图,因此其大小为 $|U(B)| \times |V(B)|$。将二分图中边数最大的二分团称为最大二分团,表示为 $B_{\max}$。

定义 3 (p,q)二分团 给定二分团 B,如果满足 $|U(B)|=p$ 且 $|V(B)|=q$,则称此二分团是一个(p,q)二分团,记作 $B_{p,q}$。

例如,图 1 展示了一个二分图 G,其中存在一个(3,3)二分团 $B_{3,3}$,$U(B_{3,3})=\{u_2,u_3,u_4\}$,$V(B_{3,3})=\{v_3,v_4,v_5\}$。同时,$B_{3,3}$ 也是最大二分团,其大小为 $|U(B_{3,3})| \times |V(B_{3,3})| = 3 \times 3 = 9$。此外,存在一个(2,2)二分团 $B_{2,2}$,$U(B_{2,2})=\{u_4,u_5\}$,$V(B_{2,2})=\{v_6,v_7\}$。该 $B_{2,2}$ 是一个极大二分团,原因是并不包含在大小比其更大的二分团中。

问题定义 给定二分图 $G=(U,V,E)$ 和 2 个阈值 p 和 q,返回二分图 G 中(p,q)二分团的个数。

本文所用到的符号及意义见表 1。

表 1 本文常用符号及其意义

Table 1 Common symbols in this paper and their meanings

符号	意义
G	二分图
U、V	G 的 U、V 层顶点集合
$\deg(u)$	顶点 u 的度数
$B_{p,q}$	(p,q)二分团
$B_{\max}$	最大二分团
g	二分扩展子图
$N(u,G)$	顶点 u 在 G 中的邻居集合
$N_2(u,G)$	顶点 u 在 G 中的 τ 强度两跳邻居集合
u_{2hop}	顶点 u 的一个 q 强度两跳邻居
v_{2hop}	顶点 v 的一个 p 强度两跳邻居

1.2 相关工作

1.2.1 (p,q)二分团计数问题

(2,2)二分团是结构最简单的(p,q)二分团,由 2 个被称为 wedge (v_s,u_m,v_e) 的基础子结构组成,其中 wedge 由起始顶点(v_s)、中间顶点(u_m)、终止顶点(v_e)以及连接这 3 个顶点的 2 条边构成[6]。Wang 等学者[4]和 Sanei-Mehri 等学者[5]通过枚举二分图中 wedge 的个数计算出(2,2)二分团的个数。二分图中(2,2)二分团计数问题是(p,q)

二分团计数问题的一个特例，本文旨在解决一般的 (p,q) 二分团计数问题。由于 (p,q) 二分团中包含大量的 wedge，结构比(2,2)二分团复杂，因此上述求解(2,2)二分团计数问题的方法无法直接用来解决 (p,q) 二分团计数问题。

针对一般的 (p,q) 二分团计数问题，Yang 等学者[7]提出 BCList 算法，该算法首先基于 (α,β) - core 对二分图 G 进行缩减，再利用度数估计选择二分图中开销成本较小的一层作为搜索层。处理搜索层顶点的过程中，先计算所有顶点的两跳邻居集合并构造一个两跳图 H，然后根据顶点度数进行顶点排序[8-10]，最后使用预分配数组和顶点标记技术对 (p,q) 二分团枚举和计数。在枚举计数过程中，先迭代地在两跳图 H 上枚举 p - 团，同时使用向量 S 来存储 p - 团中所有顶点在二分图 G 中的共同邻居，最后组合 2 部分顶点集合便得到 (p,q) 二分团。

由于真实二分图数据集的规模较大，而 BCList 算法[7]没有考虑到这一点，只基于 (α,β) - core 对二分图 G 进行缩减，这种不充分的图缩减导致图规模仍比较大，造成 (p,q) 二分团计数的低效性问题，因此就需要进一步缩小图规模，提升计数算法的效率。

1.2.2　最大二分团搜索问题

对于二分图中的最大二分团而言，其有 2 种形式：一种是最大顶点数二分团[11]；另一种是最大边数二分团[12]。本文和文献[12]保持一致，根据边数衡量二分团的大小。首先枚举出二分图中的所有极大二分团，取其中边数最多的二分团就可以得到最大二分团。

对于极大二分团枚举问题，文献[9,13]提出了膨胀法，即通过在二分图中的 2 层顶点内部增加边将二分图转化为普通图，从而将问题转化为枚举极大团问题。文献[14]指出对于给定的二分图 $G=(U,V,E)$，将二分图转化为普通图需要增加的边数为：$C_{|U|}^2+C_{|V|}^2$，因此膨胀法将二分图的边密度从 $\frac{|E|}{|U|\times|V|}$ 增加至 $\frac{|E|+C_{|U|}^2+C_{|V|}^2}{C_{|U|+|V|}^2}$。如果采用膨胀法，即使二分图是稀疏的，可能仍会导致生成的普通图十分稠密。Zhang 等学者[14]提出了 iMBEA 算法来枚举二分图中的极大二分团，该算法直接在二分图上进行搜索，利用二分图固有的结构，采用分支定界框架，通过在搜索空间中删除非极大候选顶点，枚举出二分图中所有极大二分团。此外，还进一步提出了 2 种优化策略：

(1)提前从候选集中移除顶点。

(2)根据共同邻居大小的非递减顺序选择候选顶点。

这 2 种优化都可以避免生成大量非极大候选顶点，从而提升算法效率[15-20]。但是，该算法只能处理小规模的二分图，当二分图规模变大时将面临严重的效率问题。

2　基于索引的(p,q)二分团计数

2.1　BCList 算法分析

给定二分图 G(见图 2)，计算图中 (p,q) 二分团个数时，BCList 算法[7]的基本思路为：首先基于 (α,β) - core 对图进行缩减。在此过程中，将遍历二分图 G 的 U 层顶点，如果存在一个顶点 $u\in U(G)$ 满足 $\deg(u)<q$，则将顶点 u 及其所有边从二分图 G 中删除。其次，遍历二分图 G 的 V 层顶点，如果存在一个顶点 $v\in V(G)$ 满足 $\deg(v)<p$，则可以将顶点 v 及其所有边从二分图 G 中删除。接着反复迭代地遍历二分图 G 的 U 层和 V 层中所有顶点，直到二分图 G 中不存在可以删除的顶点和边时算法终止。

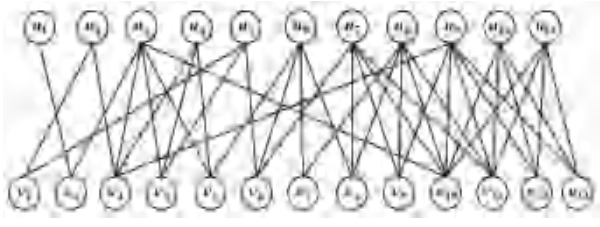

图 2　二分图 G

Fig. 2　Bipartite graph G

例如，对于图 2 中的二分图 G，给定阈值 $p=2$、$q=3$，缩减后的二分图如图 3 所示。

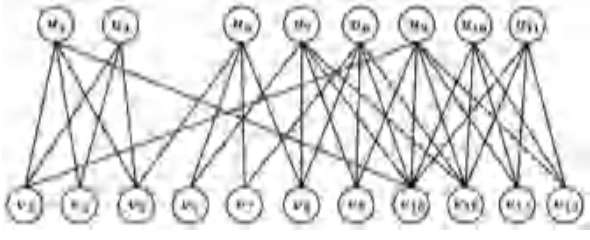

图 3　缩减后的二分图 G

Fig. 3　Reduced bipartite graph G

实现图缩减之后，算法 BCList 在缩减后的图上对 (p,q) 二分团计数。虽然 BCList 算法为 (p,q) 二分团的计数问题提供了一个有用的计算框架，但是 BCList 算法存在 2 个问题：

(1)图约减效果差。BCList 算法中只对二分图做核约减，没有考虑输入图规模较大，不充分的图缩

减可能导致约减后图规模仍比较大,影响计数算法的效率。

(2)图约减效率低。算法 BCList 中的图缩减方法在检查每层中不满足阈值条件的顶点时都需要检查所有顶点,满足阈值条件的顶点每次也都会被检查到,产生大量冗余操作,从而影响算法的约减效率。

基于以上分析,2.2.1 节、2.2.2 节针对问题(1)做出改进,提出存储二分图中所有不相交最大二分团的索引 MBC_I,实现图规模的缩减;2.2.3 节针对问题(2)做出改进,提出基于度的整体删减策略,无需反复检查每层满足阈值条件的顶点,避免了冗余操作。

2.2 算法优化

2.2.1 MBC_I 索引构建

针对算法 BCList 中图约减不充分导致图约减的效果差这一问题,本文提出一个索引 MBC_I 存储二分图中所有不相交最大二分团,实现图规模的缩减。索引 MBC_I 的构建思路为:首先调用分支定界技术求二分图上的 $B_{\max}$,之后将挖掘到的 $B_{\max}$ 插入到索引 $I_B[L][R]$ 的位置处并从二分图中删除,其中 L 和 R 分别表示 $B_{\max}$ 的 U 层和 V 层顶点个数。循环以上操作直到查找不到 $B_{\max}$ 为止,索引 MBC_I 构建完成。

例如,给定图 2 中的二分图 G,首先在 G 中挖掘到一个 $B_{\max}=\{u_9,u_{10},u_{11},v_{10},v_{11},v_{12},v_{13}\}$,其中 $B_{\max}$ 的 U 层和 V 层顶点个数分别为 3 和 4,则将该 $B_{\max}$ 存储到 $I_B[3][4]$ 位置的节点块中并从 G 中删去。接着继续挖掘剩余图 G 中的 $B_{\max}$,得到 $B_{\max}=\{u_3,u_4,v_3,v_4,v_5\}$,同理将此 $B_{\max}$ 存储到 $I_B[2][3]$ 位置的节点块中。继续迭代以上过程直到找不到 G 中的 $B_{\max}$,得到索引 MBC_I,如图 4 所示。

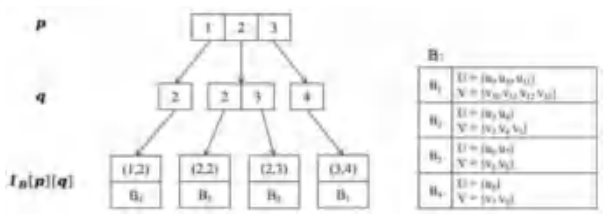

图 4 图 2 中二分图 G 的 MBC_I 索引

Fig. 4 The MBC_I index of G in Fig. 2

2.2.2 基于索引的 (p,q) 二分团计数算法 CNBC_I

以索引中的 $B_{\max}$ 作为 (p,q) 二分团计数的输入时会出现结果遗漏的情况,原因是 (p,q) 二分团的所有顶点并不一定都存在于 $B_{\max}$ 中。例如,对于图 2 中的二分图 G,给定阈值 $p=2$、$q=3$,将 B_1 存储到索引 MBC_I 中并将其从二分图 G 中删去之后,二分图 G 中的(2,3)二分团 $B_{2,3}=\{u_8,u_9,v_9,v_{10},v_{11}\}$ 在计数时会被遗漏掉。

针对这一现象,需要将二分图 G 中可能与 $B_{\max}$ 中顶点组成 $B_{p,q}$ 的所有顶点扩展到一个二分扩展子图 g 中。对 $B_{\max}$ 进行扩展时引入 τ 强度两跳邻居的概念,具体定义如下。

定义 4 τ 强度两跳邻居 给定一个二分图 $G=(U,V,E)$ 和一个整数 τ,对于二分图 G 中的一个顶点 w,如果图中存在顶点与顶点 w 拥有至少 τ 个共同邻居,则称该顶点为顶点 w 的 τ 强度两跳邻居,并加入到顶点 w 的 τ 强度两跳邻居集合 $N_2^{\tau}(w,G)$ 中,集合 $N_2^{\tau}(w,G)$ 的定义公式为:

$$N_2^{\tau}(w,G)=\{w' \mid w'\in U\cup V \wedge \mid N(w,G)\cap N(w',G)\mid \geqslant \tau\} \quad (1)$$

给定一个二分图 G 和 2 个整数 p、q,$B_{p,q}$ 的 U 层(V 层)顶点之间至少有 $q(p)$ 个共同邻居。集合 $N_2^q(u,G)$ 以及集合 $N_2^p(v,G)$ 中的顶点可扩展到 g 中,为了方便展示,在后文中将 $N_2^q(u,G)$、$N_2^p(v,G)$ 分别表示为 $N_2(u,G)$、$N_2(v,G)$。

利用 τ 强度两跳邻居对 $B_{\max}$ 进行扩展的实现细节具体见算法 1。

算法 1 Get_Subgraph($B_{\max}$,G)

输入 最大二分团 $B_{\max}$,二分图 G

输出 二分扩展子图 g

```
1. g ← B_max;
2. for each u ∈ U(B_max) do
3.     for each u_2hop ∈ N_2(u,G) do
4.         U(g) . push( u_2hop);
5. for each v ∈ V(B_max) do
6.     for each v_2hop ∈ N_2(v,G) do
7.         V(g) . push (v_2hop);
8. return g;
```

算法 1 中,首先初始化 g 为需要进行扩展的 $B_{\max}$(第 1 行)。第 2~4 行表示将 $B_{\max}$ 的 U 层顶点的 q 强度两跳邻居集合 $N_2(u,G)$ 中的顶点扩展到 $U(g)$ 中。第 5~7 行表示将 $B_{\max}$ 的 V 层顶点的 p 强度两跳邻居集合 $N_2(v,G)$ 中的顶点扩展到 $V(g)$ 中。第 8 行表示返回二分扩展子图 g。

对于图 3 中缩减后的二分图 G,给定阈值 $p=2$ 和 $q=3$,索引 MBC_I 中 $I_B[3][4]$ 位置处的最大二分团 B_1 满足阈值条件 p、q 的约束。对 B_1 进行子图扩展得到的二分扩展子图 g 如图 5 所示。

图 5　二分扩展子图 g

Fig. 5　Bipartite extension subgraph g

图 5 中的二分扩展子图 g 与图 3 中的核约减后的二分图相比，其图规模更小，在该规模较小的二分扩展子图上计数 (p,q) 二分团能够缩小搜索空间，进而提升计数算法的效率。算法 Get_Subgraph 通过遍历最大二分团的 U 层和 V 层顶点的 τ 强度两跳邻居来得到扩展二分子图，因此算法的时间复杂度为 $O(|B_{\max}|\cdot|E|)$。

基于 BCList 算法的计数思想，本文提出一个在二分扩展子图 g 上计数 (p,q) 二分团的算法 BCList*，实现细节具体见算法 2。

算法 2　BCList*(g,p,q)

输入　二分扩展子图 g，阈值 p 和 q

输出　二分扩展子图 g 中 (p,q) 二分团的个数 res

1. res←0;
2. if Cost ($U(g)$,p) > Cost($V(g)$,q) then
3. 　　Swap ($U(B_{\max})$,$V(B_{\max})$);
4. 　　Swap ($U(g)$,$V(g)$);
5. 　　Swap (p,q);
6. Collect2HopNeighbors (g,p,q);
7. Compute the rank $r(u)$ for each $u \in U(g)$;
8. Construct 2-hop graph H on $U(g)$;
9. $S \leftarrow p$ arrays initialized as empty;
10. LayerBasedListing(0, H,ϕ);
11. return res;
12. Proceduce LayerBasedListing (l,H,L)
13. if $l = p$ then
14. 　for each $R \subseteq S[l-1]$: $|R| = q$ do
15. 　　if $L \cap U(B_{\max}) \neq \phi$ or $R \cap V(B_{\max}) \neq \phi$ then res←res+1;
16. 　　else continue;
17. for each $u \in U(g)$ do
18. 　if $l = 0$ then $S[l] \leftarrow N(u,g)$;
19. 　else $S[l] \leftarrow S[l-1] \cap N(u,g)$;
20. 　if $|S[l]| < q$ or $|N(u,H)| < p-l-1$ then continue;
21. 　Construct subgraph H' of H induced by $N(u,H)$;
22. 　LayerBasedListing ($l+1$,H',$L \cup \{u\}$);

算法 2 中，首先将 g 中 (p,q) 二分团的总个数 res 初始化为 0(第 1 行)。之后利用成本估计选择时间开销较小的一层作为锚定层(2~5 行)。随后对锚定层中的顶点进行处理(6~8 行)，包括计算两跳邻居、顶点排序以及构建两跳图 H。接下来，初始化存储 $U(H)$ 中顶点共同邻居的向量 S 为一组大小为 p 的空数组(第 9 行)。然后，使用 LayerBasedListing 计数所有的 (p,q) 二分团(第 10 行)。最后，返回 g 中的所有 (p,q) 二分团个数 res(第 11 行)。在 LayerBasedListing 的计数过程中(12~22 行)，迭代地在两跳图 H 上枚举 p -团，当枚举到 p -团时，只对 p -团以及其共同邻居中有顶点存在于 $B_{\max}$ 中的 (p,q) 二分团计数。综合前述分析，基于索引 MBC_I 的 (p,q) 二分团计数算法 CNBC_I 的整体流程见算法 3。

算法 3　CNBC_I(G,p,q)

输入　二分图 G，索引 I_B，阈值 p 和 q

输出　二分图 G 中 (p,q) 二分团个数

1. res←0;
2. Reduce graph based on (α,β) - core;
3. for i from p to $|I_B|$ do
4. 　for j from q to $|I_B[i]|$ do
5. 　　for each $B_{\max} \in I_B[i][j]$ do
6. 　　$g \leftarrow$ Get_Subgraph($B_{\max}$,G);
7. 　　res←res+BCList*(g,p,q);
8. return res;

算法 3 中，首先初始化二分图 G 中 (p,q) 二分团的总个数 res 为 0(第 1 行)。之后对 G 进行基于 (α,β) - core 的图约减(第 2 行)。第 3~7 行表示取出满足条件的所有 $B_{\max}$，并分别进行子图扩展和 (p,q) 二分团计数。第 8 行返回 G 中的 (p,q) 二分团总个数。

例如，想要计算约减后的二分图 G 中的 $B_{2,3}$ 个数。由图 4 中的 MBC_I 索引结构可知，最大二分团 B_1 满足阈值条件约束。首先，计算与 B_1 相关的 $B_{2,3}$ 个数，基于 B_1 扩展得到的二分扩展子图 g 见图 5。计数过程中，对 g 中的顶点进行排序，得到 $U(g)=\{u_9,u_{11},u_{10},u_8,u_7\}$，研究中则先处理顶点 u_9。因此向量 S 中先存储顶点 u_9 在 g 中的邻居，即 $S[0]=N(u_9,g)=\{v_9,v_{10},v_{11},v_{12},v_{13}\}$。对于 $N(u_9,H)=\{u_{11},u_{10},u_8,u_7\}$，计算顶点 u_9 和顶点 u_{11} 在 g 中的

共同邻居,得到 $S[1]=\{v_{10},v_{11},v_{12},v_{13}\}$。此时 $l=p=2$,并且集合 L 和 $S[1]$ 中有顶点存在于 $B_{\max}$ 中,计算得到的 $B_{2,3}$ 个数为 $C_4^3=4$。继续上述处理,最终得到 g 中的 $B_{2,3}$ 个数为 18。同理,对于满足阈值条件约束的最大二分团 B_2,与其相关的 $B_{2,3}$ 个数为 1。至此,二分图 G 中的所有 $B_{2,3}$ 个数计算结束,总数为 18+1=19。

给定图 2 的二分图 G,基于索引 MBC_I 的 (p,q) 二分团计数算法 CNBC_I 求得的 $B_{2,3}$ 个数与算法 BCList 求得的结果相同。此外与算法 BCList 相比,算法 CNBC_I 是在规模更小的扩展二分子图上计数,具有更高的计数效率。

2.2.3 基于度的整体删减策略

针对算法 BCList 中图约减方法反复检查每层满足阈值条件顶点这一问题,本文提出基于度的整体删减策略来提升性能,并基于该策略提出优化的计数算法 CNBC_I*。删减策略的基本思想是:首先遍历二分图中的所有顶点,将不满足阈值条件的顶点放入待删除顶点队列 Del 中,然后删除这些顶点。在删除 Del 中每个顶点的过程中,先将顶点的邻居顶点度数减一,并将与该顶点相关联的度数不满足阈值条件的邻居顶点加入到 Del 中。重复执行以上操作,直到 Del 中不存在顶点。基于度的整体删减策略的具体描述见算法 4。

算法 4 Del_deg(G,p,q)

输入 二分图 $G=(U,V,E)$,阈值 p 和 q

输出 约减后的二分图 G

1. Del←ϕ;
2. for each vertex ∈ $U(G)$ ∪ $V(G)$ do
3. if vertex ∈ $U(G)$ and deg(vertex) < q or vertex ∈ $V(G)$ and deg(vertex) < p then
4. Del. push(vertex);
5. while Del ≠ ϕ do
6. vertex←Del. pop();
7. for each neighbor ∈ N(vertex,G) do
8. deg(neighbor)←deg(neighbor)−1;
9. if neighbor ∈ $U(G)$ and deg(neighbor) < q or neighbor ∈ $V(G)$ and deg(neighbor) < p then
10. Del. push(neighbor);
11. remove vertex and its edges from G;
12. return G;

算法 4 中,第 1 行首先初始化待删除顶点队列 Del 为空集,队列 Del 用于存储二分图 G 中需要被删除的顶点。第 2~4 行表示遍历二分图 G 中的所有顶点,如果顶点 vertex 是二分图 G 的 U 层顶点且该顶点的度数小于阈值 q 时,或者顶点 vertex 是二分图 G 的 V 层顶点且该顶点的度数小于阈值 p 时,则将顶点 vertex 加入到队列 Del 中。第 5~11 行表示当队列 Del 中还有需要被删除的顶点时,则继续循环地移除顶点。首先从 Del 中取出一个顶点 vertex,将该顶点及其相关联边从二分图 G 中移除。由于移除边会影响邻居顶点的度数,因此先将顶点 vertex 的所有邻居顶点 neighbor 的度数减一,如果此时受影响顶点 neighbor 的度数小于阈值条件,说明 neighbor 及其相关联边需要被移除,则将这些 neighbor 放到队列 Del 中。第 12 行表示返回基于度的整体删减策略约减后的二分图 G。

对于图 2 中的二分图 G,当阈值 $p=2$、$q=3$,待删除顶点队列 Del 的更新过程如图 6 所示。

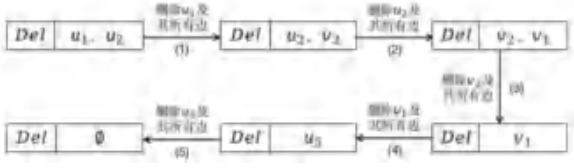

图 6 队列 Del 的更新过程

Fig. 6 The process of updating the queue Del

综上所述,Del_deg 算法利用一个队列整体地检查不满足阈值条件的顶点,当删除一个顶点时只检查与这个顶点相关联的顶点是否仍满足条件,不需要反复检查每层满足阈值条件顶点的度数,避免了 BCList 算法中核约减部分存在的大量冗余操作,提高了图约减的效率。

Del_deg 算法的时间复杂度为 $O(|U(G)|+|V(G)|+|E(G)|)$,空间复杂度为 $O(|U(G)|+|V(G)|)$。

3 实验

3.1 实验环境

实验所使用的硬件配置为 8 核的 Intel Core 17 CPU,操作频率为 2.1 GHz,运行内存为 64 GB,操作系统为 Ubuntu 22.04。实验比较的算法有:CNBC_I、CNBC_I* 和 BCList[7]。其中,算法 BCList 是目前求解 (p,q) 二分团计数的最优算法;算法 CNBC_I 为本文提出的基于 MBC_I 索引的 (p,q) 二分团计数算法;算法 CNBC_I* 是采用基于度的整体删减策略的优化算法。以上算法均采用 C++进行实现。

3.2 数据集

本文实验部分所采用的 6 个数据集为:

Youtube，Actor-movie，Twitter，IMDB，Amazon，DBLP。其中，Youtube 描述的是 Youtube 网站用户和群组之间的关系。Actor-movie 描述的是演员和参演电影之间的关系。Twitter 描述的是 Twitter 网站中用户和话题标签之间的关系。IMDB 描述的是人物和作品之间的关系。Amazon 描述的是 Amazon 网站用户和产品之间的关系。DBLP 描述的是论文作者之间的关系。表 2 中记录以上 6 个真实二分图数据集的相关信息，其中 $|U|$ 表示二分图中 U 层顶点个数，$|V|$ 表示二分图中 V 层顶点个数，$|E|$ 表示二分图中的边数，E 类型表示不同层顶点之间的关系。

表 2　数据集

Table 2　Datasets

数据集	类别	$\|U\|$	U 类型	$\|V\|$	V 类型	$\|E\|$	E 类型
Youtube	Affiliation	94 238	User	30 087	Group	293 360	Membership
Actor-movie	Affiliation	127 823	Actor	383 640	Movie	1 470 404	Appearance
Twitter	Interaction	175 214	User	530 418	Hashtag	1 890 661	Usage
IMDB	Affiliation	685 568	Actor	186 414	Movie	2 715 604	Appearance
Amazon	Rating	2 146 057	User	1 230 915	Product	5 743 258	Rate
DBLP	Authorship	1 953 085	Author	5 624 219	Publication	12 282 059	Authorship

3.3　性能比较分析

本文实验采用的评价标准为：CNBC_I、CNBC_I* 和 BCList 三种算法计数 (p,q) 二分团所消耗的时间。为了更好地对算法进行评估和挖掘到更有意义的 (p,q) 二分团，挖掘满足阈值 $p \geqslant 5$ 和 $q \geqslant 5$ 的所有不相交最大二分团，分析索引规模以及构建时间。为了测试算法的扩展性，改变阈值 p、q 的取值，对算法的运行效率进行评估。

3.3.1　运行时间比较

表 3 展示了 3 种算法在 6 个数据集上计数 $(8,6)$ 二分团个数所消耗的时间。分析可知，CNBC_I 算法在规模更小的二分扩展子图上求解 (p,q) 二分团个数，比 BCList 算法更高效。CNBC_I* 算法要优于 CNBC_I 算法，原因是算法 CNBC_I* 采用了基于度的整体删减策略，避免了冗余操作，从整体上提升运行效率。

表 3　运行时间

Table 3　Running time　　s

数据集	BCList	CNBC_I	CNBC_I*
Youtube	29.442	21.056	**20.975**
Actor-movie	2.593	1.628	**1.602**
Twitter	99.439	82.024	**80.796**
IMDB	23.451	18.244	**18.153**
Amazon	20.322	19.431	**17.476**
DBLP	5.408	5.382	**3.540**

3.3.2　索引规模和构建效率

表 4 中展示了在不同数据集上索引 MBC_I 的规模和构建时间。根据表 4 中数据发现，索引的规模随着数据集的规模存在递增的趋势，但是数据集 IMDB 的索引规模大于数据集 Amazon，造成递增趋势发生偏差的原因是数据集 IMDB 的稠密度大于数据集 Amazon。

表 4　索引规模和构建时间

Table 4　Index size and index construction time

数据集	索引规模	$\max_{\|B_{\max}\|}$	$\|\bar{B}_{\max}\|$	构建时间
Youtube	70	355	65.157	4 270
Actor-movie	103	270	48.029	23 926
Twitter	177	2 465	111.401	27 310
IMDB	478	585	44.073	137 236
Amazon	210	408	50.185	100 967
DBLP	2 228	336	35.975	61 015

从表 4 中也可以观察到：

(1) 在数据集 Youtube 上索引 MBC_I 的构建时间最短，原因是该数据集的图规模最小。

(2) 在数据集 DBLP 上索引 MBC_I 的构建时间低于在数据集 IMDB 和 Amazon 上的构建时间，其主要原因是虽然数据集 DBLP 相较于这 2 个数据集来说图规模较大，但是 DBLP 比 IMDB 和 Amazon 更稀疏。

(3) 在数据集 IMDB 上索引 MBC_I 的构建时间最长，是因为与 Youtube、Actor-movie 和 Twitter 相比，IMDB 的图规模较大，与 Amazon 和 DBLP 相比，IMDB 更稠密。

3.3.3　算法的扩展性

图 7 中展示 3 种算法在阈值对 (p,q) 取不同值时的性能，通过固定 $p + q = 14$ 来评估。

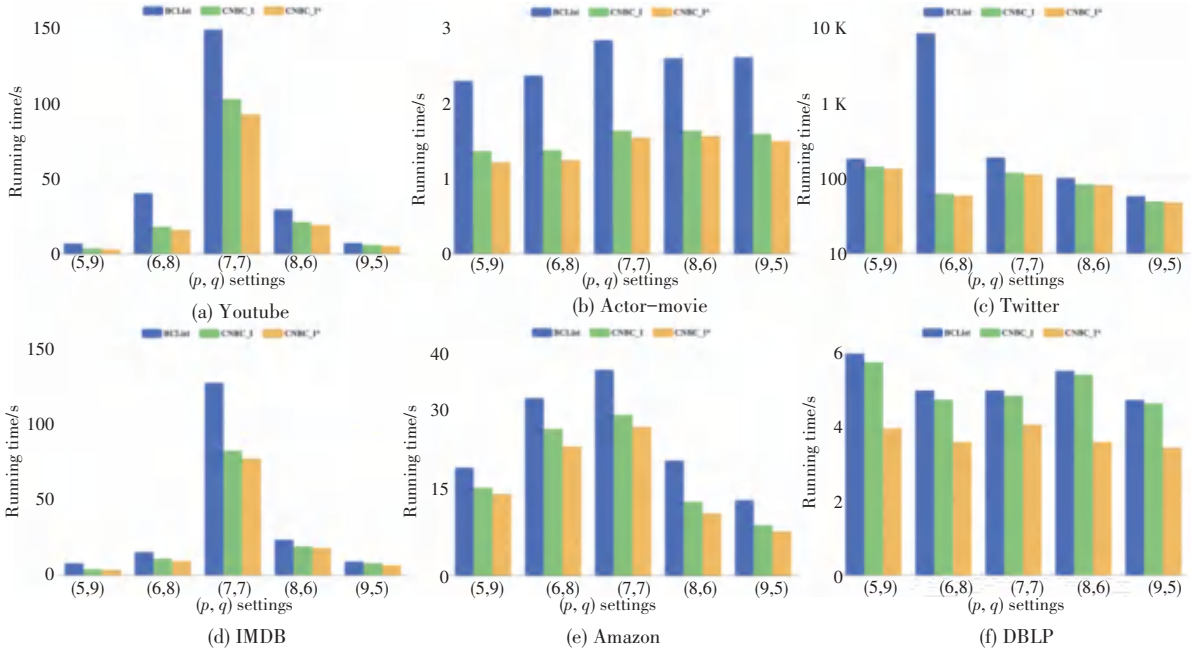

图 7 改变 p 和 q 的值

Fig. 7 Varying values of p and q

根据图 7 中数据,可以得到以下结论:

(1)不论 p、q 之间的比率很大时还是 p、q 取相同值(即 $p = q = 7$)时,CNBC_I* 算法都是最优的。

(2)CNBC_I 算法和 CNBC_I* 算法优于 BCList 算法,原因在于 CNBC_I 算法和 CNBC_I* 算法在规模更小的二分扩展子图上计数 (p,q) 二分团。

(3)CNBC_I* 算法采用了基于度的整体删减策略,避免了冗余操作,在图约减上表现出比 CNBC_I 算法更高的性能。

为了进一步分析以上 3 种算法的扩展性,将 p、q 设置为更大的值,并在大规模二分图数据集 DBLP 上进行 (p,q) 二分团计数,3 种算法的时间消耗情况见表 5。

表 5 DBLP 上改变 p 和 q 的值

Table 5 Varying values of p and q in DBLP

(p,q) 取值	BCList	CNBC_I	CNBC_I*
(8,11)	6.124	6.016	**4.661**
(9,12)	5.568	5.362	**4.736**
(10,13)	6.059	5.208	**4.288**
(11,14)	5.272	4.714	**4.217**
(12,15)	4.687	4.621	**4.082**

从表 5 中可以看出,对于阈值对 (p,q) 取值的任一种情况,CNBC_I* 算法表现出的效果都要优于其他 3 种算法,CNBC_I* 算法与 BCList 算法相比,(p,q) 二分团的计数效率提升了 15%~30%。

利用以上取值较大的阈值对 (p,q),计算 DBLP 中所包含的 (p,q) 二分团个数,计算结果见表 6。不难发现,随着阈值 p 和 q 的增大,DBLP 中所包含的 (p,q) 二分团个数呈现出递减的趋势,原因是阈值 p 和 q 的增大导致目标二分团中顶点的约束条件更加严格,所以包含在二分图中的 (p,q) 二分团的数量越来越少。

表 6 DBLP 中 (p,q) 二分团的个数

Table 6 The number of (p,q)-biclique in DBLP

(p,q) 取值	(p,q) 二分团个数
(8,11)	436 498 434
(9,12)	160 865 892
(10,13)	32 153 745
(11,14)	2 621 279
(12,15)	152

4 结束语

针对现有 (p,q) 二分团计数方法因输入图规模较大所导致的低效性问题,本文提出一种存储二分图中所有不相交最大二分团的索引。基于该索引可以获取规模较小的二分团计数输入子图,从而提升 (p,q) 二分团计数效率,同时,提出基于度的整体删减策略,避免了冗余操作,并基于此提出优化计数算法 CNBC_I*。最后,通过实验验证了本文提出的优化算法 CNBC_I* 的高效性,实验结果表明本文方法比现有方法在二分团计数时效率提高了 20%~50%。

参考文献

[1] MITZENMACHER M, PACHOCKI J, PENG R, et al. Scalable large near-clique detection in large-scale networks via sampling [C]//Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York:ACM, 2015: 815-824.

[2] BORGATTI S P, EVERETT M G. Network analysis of 2-mode data[J]. Social Networks, 1997, 19(3): 243-269.

[3] SARIYÜCE A E, PINAR A. Peeling bipartite networks for dense subgraph discovery [C]//Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining. New York:ACM, 2018: 504-512.

[4] WANG Jia, FU A W C, CHENG J. Rectangle counting in large bipartite graphs [C]//Proceedings of 2014 IEEE International Congress on Big Data. Piscataway,NJ:IEEE, 2014: 17-24.

[5] SANEI-MEHRI S V, SARIYUCE A E, TIRTHAPURA S. Butterfly counting in bipartite networks[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. New York: ACM, 2018: 2150-2159.

[6] WANG Kai, LIN Xuemin, QIN Lu, et al. Vertex priority based butterfly counting for large-scale bipartite networks [J]. Proceedings of the VLDB Endowment, 2019, 12(10): 1139-1152.

[7] YANG Jianye, PENG Yun, ZHANG Wenjie, et al. (p,q)-biclique counting and enumeration for large sparse bipartite graphs [J]. The VLDB Journal, 2023, 32: 1137-1161.

[8] SCHWEIGER R, LINIAL M, LINIAL N. Generative probabilistic models for protein-protein interaction networks: The biclique perspective[J]. Bioinformatics, 2011, 27(13): 142-148.

[9] LI Jinyan, LIU Guimei, LI Haiquan, et al. Maximal biclique subgraphs and closed pattern pairs of the adjacency matrix: A one-to-one correspondence and mining algorithms [J]. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(12): 1625-1637.

[10] DAS A. Incremental and parallel algorithms for dense subgraph mining[D]. Ames: Iowa State University, 2019.

[11] GAREY M R, JOHNSON D S. Computers and intractability [M]. San Francisco: Freeman, 1979.

[12] PEETERS R. The maximum edge biclique problem is NP-complete[J]. Discrete Applied Mathematics, 2003, 131(3): 651-654.

[13] LIU G, SIM K, LI J. Efficient mining of large maximal bicliques [C]//Proceedings of 8th International Conference on Data Warehousing and Knowledge Discovery. Cham: Springer, 2006: 437-448.

[14] ZHANG Yun, PHILLIPS C A, ROGERS G L, et al. On finding bicliques in bipartite graphs: A novel algorithm and its application to the integration of diverse biological data types [J]. BMC Bioinformatics, 2014, 15: 1-18.

[15] LARADJI I H, ROSTAMZADEH N, PINHEIRO P O, et al. Where are the blobs: Counting by localization with point supervision[C]// Proceedings of the 15th European Conference on Computer Vision (ECCV). Cham: Springer, 2018: 560-576.

[16] TOPKAYA I S, ERDOGAN H, PORIKLI F. Counting people by clustering person detector outputs[C]// Proceedings of the 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). Piscataway,NJ: IEEE, 2014: 313-318.

[17] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks [C]// Proceedings of the 25th International Conference on Neural Information Processing Systems. New York: ACM, 2012: 1097-1105.

[18] CAO Xinkun, WANG Zhipeng, ZHAO Yanyun, et al. Scale aggregation network for accurate and efficient crowd counting [C]// Proceedings of the 15th European Conference on Computer Vision (ECCV). Cham: Springer, 2018: 757-773.

[19] YU F, KOLTUN V. Multi-scale context aggregation by dilated convolutions[J]. arXiv preprint arXiv,1511.07122, 2015.

[20] BABU S D, SURYA S, VENKATESH B R. Switching convolutional neural network for crowd counting [C]// Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway,NJ: IEEE, 2017: 4031-4039.