ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.11, November 2008, pp.2879−2886
DOI: 10.3724/SP.J.1001.2008.02879

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

P_2-Packing问题参数算法的改进*

王建新+, 宁 丹, 冯启龙, 陈建二

(中南大学 信息科学与工程学院,湖南 长沙 410083)

Improved Parameterized Algorithm for P_2-Packing Problem

WANG Jian-Xin+, NING Dan, FENG Qi-Long, CHEN Jian-Er

(School of Information Science and Engineering, Central South University, Changsha 410083, China)

+ Corresponding author: E-mail: jxwang@mail.csu.edu.cn

Wang JX, Ning D, Feng QL, Chen JE. Improved parameterized algorithm for P_2-packing problem. *Journal of Software*, 2008,19(11):2879−2886. http://www.jos.org.cn/1000-9825/19/2879.htm

Abstract: P_2-Packing is a typical NP-hard problem. This paper provides a further study on the structures of the P_2-packing problem, and proposes a kernelization algorithm that can obtain a kernel of size at most $7k$, which greatly reduces the current best kernel $15k$. Based on the kernelization algorithm, an improved parameterized algorithm with running time $O^*(2^{4.142k})$ is also presented, which greatly improves the current best result $O^*(2^{5.301k})$.
Key words: P_2-packing; kernelization; parameterized algorithm

摘 要: P_2-Packing 问题是一个典型的 NP 难问题.目前这个问题的最好结果是时间复杂度为 $O^*(2^{5.301k})$的参数算法,其核的大小为 $15k$.通过对 P_2-packing 问题的结构作进一步分析,提出了改进的核心化算法,得到大小为 $7k$ 的核,并在此基础上提出了一种时间复杂度为 $O^*(2^{4.142k})$的参数算法,大幅度改进了目前文献中的最好结果.
关键词: P_2-packing;核心化;参数算法
中图法分类号: TP301 **文献标识码**: A

Packing 问题是一类重要的 NP 难问题.图的 H-Packing 问题是 Packing 问题的一个重要分支,被广泛应用于调度、代码优化和生物信息学等领域.为便于进一步准确描述和讨论该问题,这里先给出图的 H-Packing 问题的定义[1]:

定义 1(图的 H-Packing 问题). 给定一个图 G 和一个图 H,G 和 H 都是连通图.图 G 的一个 H-Packing 是 G 的一个互不相交的子图的集合,每个子图都与图 H 同构.

人们从传统复杂性理论角度出发主要是在近似算法方面研究图的 H-Packing 问题(即 MAXIMUM H-Packing 问题):找出图的一个 H-Packing,使得该 H-Packing 集合的元素数目最多).依据 H 的大小,人们对图的 MAXIMUM H-Packing 问题有了不同的结果.若图 H 是完全图 K_2,即图 H 是由两个点组成的连通图,MAXIMUM

* Supported by the National Natural Science Foundation of China under Grant Nos.60433020, 60773111 (国家自然科学基金); the Program for New Century Excellent Talents in University of China under Grant No.NCET-05-0683 (新世纪优秀人才支持计划); the Program for Changjiang Scholars and Innovative Research Team in University of China under Grant No.IRT0661 (长江学者和创新团队发展计划)

Received 2007-09-02; Accepted 2008-01-08

H-Packing 问题也就是我们熟悉的二分图的最大匹配问题,它已被证明是多项式可解的.当图 H 是一个至少有 3 个点的连通图时,人们证明了该问题是 MAX-SNP(maximum strict non-deterministic polynomial)完全的[2].

在参数计算领域,目前现有的许多技术,如 Color-Coding[3]、局部贪婪[4]等,可被用来求解 H-Packing 问题.当 H 为一般图时,文献[5]利用局部贪婪技术给出了时间复杂度为 $2O^{(|H|k\log k+k|H|\log|H|)}$ 的参数算法.文献[6]研究了边不相交的三角形 packing 参数问题,即在图中找出 k 个边不相交的三角形,通过核心化技术得到一种固定参数可解算法,提出了一个 $4k$ 大小的核,算法的时间复杂度为 $O\left(2^{\frac{9k}{2}\log k+\frac{9k}{2}}\right)$.文献[7]中研究了 $K_{1,s}$-Packing 的参数问题(H 为完全二分图 $K_{1,s}$),即 k-$K_{1,s}$-Packing 问题,其具体定义如下:

定义 2(k-$K_{1,s}$-Packing 问题). 给定一个连通图 $G=(V,E)$和一个正整数 k,在图 G 中是否存在 k 个点互不相交的完全二分图 $K_{1,s}$.

文献[7]在 k-$K_{1,s}$-Packing 参数问题上给出了一个 $O(k^3)$的核,从而证明了该问题是固定参数可解的.同时重点研究了 k-$K_{1,s}$-Packing 问题的一个特殊实例,即 k-P_2-Packing 问题.在图论中,P_i 表示由 i+1 个点和 i 条边组成的一条路径,于是,P_2 就是由 3 个点和 2 条边构成的一条路径.对于一条路径 P_2 来说,其中间的点被称为中心点,其他两个点被称为端点.k-P_2-Packing 问题的具体定义如下:

定义 3(k-P_2-Packing 问题). 给定一个连通图 $G=(V,E)$和一个正整数 k,在图 G 中是否存在 k 个点互不相交的 P_2.

文献[7]提出了基于皇冠分解思想的核心化预处理算法,得到一个大小最多为 $15k$ 个点的核,并由此针对 k-P_2-Packing 问题提出了时间复杂度为 $O^*(2^{5.301k})$(对于一个参数 k 的函数 $f(k)$,用 $O^*(f(k))$来表示界 $O(f(k)n^{o(1)})$)的参数算法.

降低求解 k-P_2-Packing 问题的算法时间复杂度的关键技术之一是如何减小问题的核,本文主要研究 k-P_2-Packing 问题的核心化.通过对问题结构的深入分析,在文献[7]的核心化算法基础上提出了一种可以获得大小最多为 $7k$ 个点的核的核心化算法,从而使该问题的参数算法时间复杂度降低为 $O^*(2^{4.142k})$.

本文第 1 节给出相关定义及引理.第 2 节描述核心化算法及计算核的大小.第 3 节分析算法运行时间.第 4 节给出结论.

1 相关定义及引理

下面先给出相关的图的概念和术语[8].

假设 $G=(V,E)$表示一个无向连通图,其中 V 是图 G 中点的集合,且$|V|=n$;E 是图 G 中边的集合.一个点 v 的邻接点表示为 $N(v)$.对于图 G 的任意一个子图 H,用 $N(H)$表示不在 H 中但至少与 H 中的 1 个点相连的点的集合.如果点 $u\in N(H)$,就说子图 H 与点 u 相连.特别地,如果点 u 不是边 e 的一个端点但至少与边 e 的 1 个端点相连,则说边 e 与点 u 相连.从图 G 中除去点 v 表示为 $G\backslash v$,即 $G=(V\backslash\{v\},E)$;从图 G 中除去边 e 表示为 $G\backslash e$,即 $G=(V, E\backslash\{e\})$.类似地,V'是一个点的集合,从图 G 中除去点的集合 V'表示为 $G\backslash V'$,即 $G=(V\backslash V',E)$;E'是一个边的集合,从图 G 中除去边的集合 E'表示为 $G\backslash E'$,即 $G=(V,E\backslash E')$.

为了描述方便,下面先给出“double 皇冠”分解和“fat 皇冠”分解的基本概念[7].

定义 4(double 皇冠分解). 图 $G=(V,E)$的一个“double 皇冠”分解(H,C,R)是图 G 的 3 个点集 H,C 和 R 的划分,并具备以下属性:

1. H(头)是图 G 的一个分割点集,使得 G 中属于 C 集的点和属于 R 集的点之间不存在边.

2. 皇冠 C 是图 G 的一个独立集,且 $C=C_u\cup C_m\cup C_{m2}$,其中,C_m 和 C_{m2} 是被皇冠 C 与头 H 形成的匹配浸润的点(即与匹配中的边相关联的点),C_u 是未被浸润的点(即与匹配中的边无关联的点).

3. 在 C_m 和 H,C_{m2} 和 H 之间分别存在一个完美匹配.C_m 和 H 的大小相同,即$|C_m|=|H|$.C_{m2} 和 H 的大小也相同,即$|C_{m2}|=|H|$.

定义 5(fat 皇冠分解). 图 $G=(V,E)$中的一个“fat 皇冠”分解(H,C,R)是图的 3 个点集 H,C 和 R 的划分,并具备

以下属性:

1. H(头)是图 G 中的一个分割点集,使得 G 中属于 C 集的点和属于 R 集的点之间不存在边.

2. 皇冠 C 的导出子图 $G[C]$是一个森林,每个组成部分都同构于完全图 K_2.

3. $|C|\ge|H|$,头 H 和皇冠 C 的一个大小为$|H|$的子集之间存在一个完美匹配 M,M 的每条边的两个端点分别是 H 中的一个点和 C 中的一个 K_2,而该 K_2 中只有一个端点是被 M 浸润的.

本文的核心化算法中应用了"double 皇冠"分解和"fat 皇冠"分解.下面给出与"double 皇冠"分解和"fat 皇冠"分解相关的几个引理[7].

引理 1. 当且仅当图 $G\backslash(H\cup C)$有一个$(k-|H|)$-P_2-Packing 时,存在一个"double 皇冠"分解(H,C,R)的图 $G=(V,E)$一定有一个 k-P_2-Packing.

引理 2. 当且仅当图 $G\backslash(H\cup C)$有一个$(k-|H|)$-P_2-Packing 时,存在一个"fat 皇冠"分解(H,C,R)的图 G 有一个 k-P_2-Packing.

引理 3. 如果图 G 中有一个独立集 I,并且$|I|\ge2|N(I)|$,则图 G 能够在线性时间内构建一个"double 皇冠"分解(H, C,R),且 $H\subseteq N(I)$.

引理 4. 如果图 G 中有一个由独立的完全图 K_2 构成的集合 J,并且$|J|\ge|N(J)|$,则图 G 能够在线性时间内构建一个"fat 皇冠"分解(H,C,R),且 $H\subseteq N(J)$.

2 k-P_2-Packing问题的核心化算法

假设 $W=\{L_1,...,L_t\}$,$t\le k-1$,其中每个 $L_i(1\le i\le t)$是图 G 中同构于 P_2 的一个子图.W 表示图 G 的一个极大 P_2-packing(一个 P_2-packing 是由点互不相交的 P_2 构成的一个集合,极大 P_2-packing 是指在图 G 中任意选择一个 P_2 添加到该集合中都不可能再形成一个新的 P_2-packing);$Q=V\backslash V(W)$,其中 V 是图 G 中点的集合,$V(W)$是 W 中点的集合.Q_i 表示 Q 的导出子图中度为 i 的点的集合,Q 中不存在度数大于等于 2 的点,只存在度为 0 的点和度为 1 的点[7],即 Q_0 点和 Q_1 点.假设 Q_0 中的每个点用 Q_0-vertex 表示;Q_1 中的每个点用 Q_1-vertex 表示,每条边用 Q_1-edge 表示,一条 Q_1-edge 也就是两个 Q_1-vertex 形成的一条边.令$|W|$表示 W 中所有点的个数,$|Q|$表示 Q 中所有点的个数.其中,$|Q_0|$表示所有 Q_0-vertex 的个数,即$|Q_0|=|Q_0\text{-vertex}|$;$|Q_1|$表示所有 Q_1-vertex 的个数,即$|Q_1|=|Q_1\text{-vertex}|$.$|Q_1\text{-edge}|$表示所有 Q_1-edge 的个数,因为每条 Q_1-edge 包含两个 Q_1-vertex,所以$|Q_1|=|Q_1\text{-vertex}|=2|Q_1\text{-edge}|$.因此,$|Q|=|Q_0|+|Q_1|=|Q_0\text{-vertex}|+|Q_1\text{-vertex}|=|Q_0\text{-vertex}|+2|Q_1\text{-edge}|$.

2.1 算法RPLW

本文提出的能够得到 $7k$ 个点的核的核心化是在完成文献[7]的核心化之后通过对算法 RPLW 的反复调用来实现的.我们知道,由文献[7]的核心化可以得到一个由极大 P_2-packing W 及其对应的 Q 组成的图 G,算法 RPLW 则是对 W 和 Q 中的点作进一步的优化处理.为了描述如何得到 $7k$ 大小的核,本文将着重讨论算法 RPLW.

算法 RPLW 分别处理 Q 中的 Q_0-vertex 和 Q_1-edge.当 W 的大小没有发生变化时,算法 RPLW 的目的是减少 Q 中的 Q_0-vertex 的个数.而当减少 Q_1-edge 的个数时,会使得 W 中包含的 P_2 的个数增加,也就是 W 增大,算法就返回该增大的 W 并将该 W 作为输入参数,继续不断地调用算法 RPLW.如果在图中找到"double 皇冠"分解或者"fat 皇冠"分解,就可以找到$|H|$个($|H|$是皇冠头的大小)点互不相交的 P_2,因此参数 k 将减少($k=k-|H|$[7]),算法返回该减少的参数 k 并将该参数 k 作为输入参数,继续不断地调用算法 RPLW.

为了更好地分析算法 RPLW,首先讨论以下两种操作结构,如图 1 和图 2 所示.在图中,用实心点和加粗的线表示 W 中的点和边,用空心点和不加粗的线表示 Q 中的点和边.特别地,由一条不加粗的线连接的两个空心点表示一条 Q_1-edge.

图 1 描述的是,为了减少 Q 中 Q_0-vertex 的个数,对 W 中的 L_i 进行替换.具体操作如下:

在图 1(a)中,假设该 P_2 为 L_i,其中心点为 c,两端点分别为 t_1,t_2,且 Q_0-vertex q_2 与中心点 c 相连,Q_0-vertex q_1 与端点 t_1 相连.如果用 q_2,c 和 t_2 形成的新 P_2 L'_i 替换 L_i,则点 q_1 和点 t_1 形成一条 Q_1-edge,Q_0-vertex 减少 2 个.

在图 1(b)中,假设该 P_2 为 L_i,其中心点为 c,两端点分别为 t_1,t_2,且 Q_0-vertex q_2 与端点 t_2 相连,Q_0-vertex q_1 与

端点 t_1 相连.如果用 q_1,t_1 和 c 形成的新 P_2 L'_i 替换 L_i,则点 q_2 和点 t_2 形成一条新 Q_1-edge,Q_0-vertex 减少 2 个.

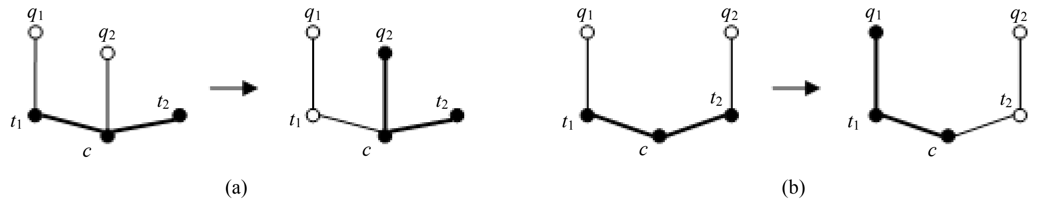


Fig.1 Reduce the number of Q_0-vertex

图 1 减少 Q_0-vertex 的个数

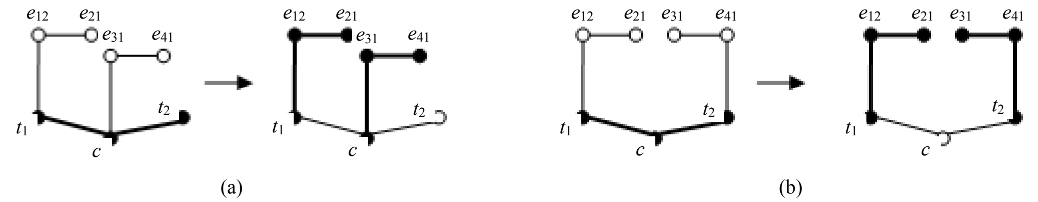


Fig.2 Reduce the number of Q_1-edge

图 2 减少 Q_1-edge 的个数

图 2 描述的是,为了减少 Q 中 Q_1-edge 的个数和增加 W 的大小,对 W 中的 L_i 进行替换.具体操作如下:

在图 2(a)中,假设该 P_2 为 L_i,其中心点为 c,两端点分别为 t_1,t_2,且 Q_1-edge(e_1,e_2)与端点 t_1 相连,Q_1-edge(e_3,e_4)与中心点 c 相连.如果用 e_2,e_1,t_1 形成的新 $P_2L'_i$ 和 e_4,e_3,c 形成的新 $P_2L''_i$ 替换 L_i,则 W 的大小增 1 且 Q_1-edge 减少 1 个.

在图 2(b)中,假设该 P_2 为 L_i,其中心点为 c,两端点分别为 t_1,t_2,且 Q_1-edge(e_1,e_2)与端点 t_1 相连,Q_1-edge(e_3,e_4)与端点 t_2 相连.如果用 e_2,e_1,t_1 形成的新 P_2 L'_i 和 e_3,e_4,t_2 形成的新 P_2 L''_i 替换 L_i,则 W 的大小增 1 且 Q_1-edge 减少 1 个.

图 1 和图 2 形象地描述了如何对 W 中的 L_i 进行替换从而使 Q 中的 Q_0-vertex 和 Q_1-edge 的个数减少.根据图 1 和图 2 可以得到以下两条操作规则.

规则 1. 如果 W 中的一个 L_i 中有两个点分别连接到一个 Q_0-vertex(这两个 Q_0-vertex 是不相同的),则应用图 1 所描述的操作过程,从而减少 2 个 Q_0-vertex,但增加 1 条 Q_1-edge.

规则 2. 如果 W 中的一个 L_i 中有两个点分别连接到一条 Q_1-edge(这两个 Q_1-edge 是不相同的),则应用图 2 所描述的操作过程,从而减少 1 条 Q_1-edge,但使 W 大小加 1.

上述规则在算法 RPLW 中得到了充分应用,具体见算法 1.

算法 1 中 Step 1 的目的是在 W 的大小没有发生变化时,减少 Q 中的 Q_0-vertex 的个数.因为图 G 中的点是有限的(最多为 $15k$ 个点[7]),所以 while 循环不会总是执行下去.在替换 Q_0-verttex 的过程中可能在 Q 中找到新的 P_2,将在 Q 中找到的新的 P_2 添加到 W 中,使 W 中的点互不相交的 P_2 个数增加,从而使 W 增大.在应用规则 1 和规则 2 的过程中,所产生的 W 可能不再是极大的.这种情况下,在进一步应用规则 1 和规则 2 之前需利用任意正确的贪婪算法先使 W 又成为极大 P_2-packing.因此,在 Step 2 中构建一个极大 W'.为了在这个极大 W'中找到 k 个点互不相交的 P_2,将该 W'作为输入参数返回,继续调用算法 RPLW.在 Step 3 中的 Q_1-edge 替换过程中,W 也会增大,为了在这个更大的 W'中找到 k 个点互不相交的 P_2,也将该 W'作为输入参数返回继续调用算法 RPLW.算法的 Step 4 和 Step 5 是要在图中找到由 Q_0-vertex 和 Q_1-edge 的替换而产生的"double 皇冠"分解和"fat 皇冠"分解.一旦找到"double 皇冠"分解或者"fat 皇冠"分解,则参数 k 必将减少,参数 k 越小,接下来需要找出的点互不相交的 P_2 的个数也将越少.因此,算法返回该减小的参数 k'并将其作为输入参数继续调用算法 RPLW,从而找到

k 个点互不相交的 P_2.在调用算法 RPLW 的过程中,如果找到 k 个点互不相交的 P_2,则停止对算法的调用,算法结束.

算法 1. 算法 RPLW.

RPLW(G,W,k)

输入:G,W,k.

输出:一个极大 P_2-packing W',$|W'|>|W|$或者参数 k',$|k'|<|k|$或者图 G',$|G'|<|G|$.

Step 1. while W 是一个极大 P_2-packing 并且 W 中的一个 L_i 上有两个点分别连接到一个 Q_0-vertex do 应用规则 1; // W 中的 L_i 被替换,但 W 的大小不变,Q 中 Q_0-vertex 的个数减少

Step 2. if W 不是极大的 then
  用贪婪算法构造一个极大 P_2-packing W';
  return (G,W',k).

Step 3. if W 中的一个 L_i 上有两个点分别连接到一条 Q_1-edge then
  应用规则 2 得到一个更大的 P_2-packing W';
  return (G,W',k).

Step 4. if $|Q_0|\ge 2|W|$ then
  图 G 产生一个“double 皇冠”分解;
  return (G,W,k').

Step 5. if $|Q_1|\ge|W|$ then
  图 G 产生一个“fat 皇冠”分解;
  return (G,W,k').

Step 6. return (G',W,k).

下面我们将证明算法的正确性,并分析反复调用算法所需的时间.

引理 5. 在 $O(k^3)$的时间内反复调用算法 RPLW 要么能找到一个 k-P_2-packing,要么将减少图 G 的规模.

证明:从算法 RPLW 的 Step 2 和 Step 3 可以看出,Q_0-vertiex 和 Q_1-edge 的替换都会使 W 中 P_2 的个数增加,从而使 W 增大.通过对算法的反复调用,当 W 中的 P_2 的个数增加到 k 时,则说明找到了一个 k-P_2-packing,问题得到求解.同时,在算法的 Step 4 和 Step 5 中,由于替换使得图 G 产生“double 皇冠”分解或者“fat 皇冠”分解,因此参数 k 将减少,接下来需要找出的点互不相交的 P_2 也将减少.通过对算法的反复调用,当参数 k 减少到 0 时,则说明找到了一个 k-P_2-packing,问题得到求解.另一方面,由于对 Q_0-vertiex 和 Q_1-edge 的替换限制了 Q 中 Q_0-vertiex 和 Q_1-edge 的大小,并且由于“double 皇冠”分解和“fat 皇冠”分解,一部分点将从图 G 中移走,因此图 G 的规模也就减小了.所以,如果算法没有找到一个 k-P_2-packing,则能够减小图 G 的规模.

最后,分析算法 RPLW 的时间复杂度.对算法 RWPL 的反复调用,也就是反复应用规则 1 和规则 2,这个过程能够在多项式时间内完成.算法 RPLW 并不可能无限制地应用规则 1 和规则 2,因为每一次应用规则 1 都会使 Q_0-vertex 的个数减少 2 个,而图 G 中最多有 $15k$ 个点[7],所以规则 1 最多可被应用 $7.5k$ 次.又因为每一次应用规则 2 都会使 W 中 P_2 的个数增加 1 个,所以规则 2 最多可被应用 k 次.也就是说,由于 W 是最多 $k-1$ 个点互不相交的 P_2 构成的一个集合,所以 W 中 P_2 的个数最多增加 k 次就能找到一个 k-P_2-packing,从而使问题得到求解.由于每次产生“double 皇冠”分解或者“fat 皇冠”分解,参数 k 都必将减少,因此参数 k 最多减少 k 次,否则就能找到一个 k-P_2-packing,问题得到求解.而当 W 增大或者参数 k 减少时,算法都会退出然后再被重新调用.也就是说,算法最多可被调用 $9.5k$ 次,“double 皇冠”分解或者“fat 皇冠”分解可以在 $O(k^2)$时间内完成[7],因此整个调用过程所需的时间是 $O(k^3)$. □

该核心化算法的执行过程是反复地调用算法 RPLW,也就是以一个极大 P_2-packing W 开始来反复地应用规则 1 和规则 2(W 保持为极大 P_2-packing),直到规则 1 和规则 2 都不能再被应用的过程.这个过程是可以在多项式时间内完成的.算法最终要么能够找到一个 k-P_2-packing,要么在规则 1 和规则 2 都不能再被应用时得到一个

小于 k 的极大 P_2-packing,此时图 G 的规模已减小,而规模减小的图 G 则可以作为 k-P_2-packing 问题的核来继续对问题求解.当算法中规则 1-2 都不可再被应用时,可分析出 W 具有以下属性:

属性 1. 如果有多个 Q_0-vertex 与 W 中的任意一个 L_i 相连,则与该 L_i 相连的所有这些 Q_0-vertex 都必须连到 L_i 中的同一个点上.

属性 2. 如果 W 中的任意一个 L_i 中有多个点都与 Q_0-vertex 相连,则该 L_i 中所有与 Q_0-vertex 相连的这些点都必须连到同一个 Q_0-vertex 上.

属性 3. 如果有多条 Q_1-edge 与 W 中的任意一个 L_i 相连,则与该 L_i 相连的所有这些 Q_1-edge 都必须连到 L_i 中的同一个点上.

属性 4. 如果 W 中的任意一个 L_i 中有多个点都与 Q_1-edge 相连,则该 L_i 中所有与 Q_1-edge 相连的这些点都必须连到同一个 Q_1-edge 上.

2.2 一个更小的核

为了描述如何得到 $7k$ 个点的核,下面首先分析核心化之后整个 Q_0-vertex 和 Q_1-edge 的大小,然后计算如何由 Q_0-vertex 和 Q_1-edge 的大小得到一个最多为 $7k$ 个点的核.

定理 1. $|Q_0\text{-vertex}|\le 2(k-1)$,否则在多项式时间内可以得到图 G 中的一个"double 皇冠"分解.

证明:当算法中规则 1 和规则 2 都不能再被应用时,设 $W=\{L_1,...,L_t\},t\le k-1$,其中 $L_1,...,L_t$ 是点互不相交的 P_2 的一个集合,且满足属性 1~属性 4.把 W 中的 P_2 分成两部分$(L_1,...,L_d),(L_{d+1},…,L_t)$,使它们分别满足以下性质:对于任意一个 $L_i(1\le i\le d)$,每个与 L_i 相连的 Q_0-vertex 可以连接到 L_i 的多个点,假设这些 Q_0-vertex 点用 Q_{0i} 表示,$1\le i\le d$;对于任意一个 $L_j(j>d)$,每个与 L_j 相连的 Q_0-vertex 只连接到 L_j 的同一个点.

令集合 Q'_0-vertex$=Q_0$-vertex$-\{Q_{01},...,Q_{0d}\}$.假设集合 $W'=\{v_1,...,v_s\}$表示 $L_{d+1}\cup…\cup L_t$ 中与集合 Q_0-vertex 中的点为邻居的点的集合.根据对 L_j 的划分,对于任意一个 $L_j(j>d),L_j$ 中最多只有 1 个点出现在集合 W'中.因此,可以得出 $s\le t-d$.而根据属性 2 可知,Q'_0-vertex 中没有一个点会与 L_i 中的任意一个点相连,所以每个 Q'_0-vertex 点的邻居都在 W'中,即 $W'=N(Q'_0\text{-vertex})$.

假设 Q'_0-vertex 的个数为 p,如果 $p>2s$,则由引理 3 可知,图 G 中必定存在一个"double 皇冠"分解,从而又可以继续调用算法 RPLW,这与规则 1 和规则 2 都不可再被应用相矛盾.如果 $p\le 2s$,则$|Q_0\text{-vertex}|=|Q'_0\text{-vertex}|+d=p+d\le 2s+d\le 2(s+d)\le 2t\le 2(k-1)$. □

定理 2. $|Q_1\text{-edge}|\le k-1$,否则在多项式时间内可得到图 G 中的一个"fat 皇冠"分解.

证明:当算法中规则 1 和规则 2 都不可再被应用时,设 $W=\{L_1,...,L_t\},t\le k-1$,其中 $L_1,...,L_t$ 是点互不相交的 P_2 的一个集合,且满足属性 1~属性 4.把 W 中的 P_2 分成两部分$(L_1,...,L_d),(L_{d+1},…,L_t)$,使它们分别满足以下性质:对于任意一个 $L_i,1\le i\le d$,每个与 L_i 相连的 Q_1-edge 可以连到 L_i 的多个点,假设这些 Q_1-edge 中的点用 $Q_{1i}(1\le i\le d)$来表示;对于任意一个 $L_j(j>d)$,每个与 L_j 相连的 Q_1-edge 只连接到 L_j 的同一个点.

令集合 Q'_1-edge$=Q_1$-edge-$\{Q_{11},...,Q_{1d}\}$.假设集合 $W'=\{v1,...,vs\}$表示 $L_{d+1}\cup…\cup L_t$ 中与集合 Q_1-edge 中的点为邻居的点的集合.根据对 L_j 的划分,对于任意一个 $L_j(j>d),L_j$ 中最多只有 1 个点出现在集合 W'中.因此,可以得出 $s\le t-d$.而根据属性 4 可知,Q'_1-edge 中没有一个点会与 L_i 中的任意一个点相连,所以每个 Q'_1-edge 点的邻居都在集合 W'中,即 $W'=N(Q'_1\text{-edge})$.

假设 Q'_1-edge 的大小为 p,如果 $p>s$,则由引理 4 可知,图 G 中必定存在一个"fat 皇冠"分解,从而可以继续调用算法 RPLW,这与规则 1 和规则 2 都不可再被应用相矛盾.如果 $p\le s$,则$|Q_1\text{-edge}|=|Q'_1\text{-edge}|+d=p+d\le s+d\le t\le k-1$. □

定理 3. k-P_2-Packing 问题的核的大小最多为 $7k$ 个点.

证明:通过反复调用算法 RPLW 来完成问题的核心化之后,图 G 中的点由 W 中的点和 Q 中的点构成,而 Q 中只存在度为 1 和度为 0 的点.因为$|V(W)|\le 3(k-1)$,由定理 1 可知,$|Q_0\text{-vertex}|\le 2(k-1)$,即$|Q_0|\le 2(k-1)$.由定理 2 可知,$|Q_1\text{-edge}|\le k-1$,而每条 Q_1-edge 都有两个 Q_1-vertex,所以$|Q_1|=|Q_1\text{-vertex}|=2|Q_1\text{-edge}|\le 2(k-1)$.用 $V(G)$表示图 G 中的点,可以知道$|V(G)|=|V(W)|+|Q|=|V(W)|+|Q_1|+|Q_0|\le 3(k-1)+2(k-1)+2(k-1)=7k$,而如果$|V(G)|>7k$,则图 G 中一定

存在一个大小为 k 的 P_2-packing,即 k-P_2-packing,所以 k-P_2-Packing 问题的核的大小最多为 $7k$ 个点. □

3 改进的k-P_2-packing问题参数算法

针对 k-P_2-Packing 问题,本文基于大小最多为 $7k$ 个点的核提出一种参数算法 KPPW.在算法 KPPW 中,首先通过执行核心化算法得到一个大小最多为 $7k$ 个点的核.在得到问题的核以后,再利用穷举法来找到问题的解.k-P_2-Packing 问题要找到一个包含 k 个点不相交的 P_2 的集合,而每个 P_2 都有一个中心点,k 个 P_2 就有 k 个中心点,所以要在 $7k$ 个点中穷举出所有可能的 k 个点的组合,从而找出 k 个点不相交的 P_2 的 k 个中心点.具体见算法 2.

算法 2. 算法 KPPW.

KPPW(G,k)

输入:图 G=(V,E),参数 k.

输出:k 个点互不相交的 P_2,否则返回不存在.

Step 1. 运用贪婪算法构造一个极大 P_2-packing W;

Step 2. 调用算法 *RPLW*(G,W,k); //得到一个大小最多为 $7k$ 个点的核

Step 3. if $|V(G)|>7k$ then

  输出“在图 G 中存在 k 个点不相交的 P_2”,算法结束;

Step 4 else 在所得的大小最多为 $7k$ 个点的核上穷举出所有可能的 k 个点的组合;

Step 5. for 每个组合 C do

Step 6. 用组合 C 中的 k 个点及其邻接点构建一个辅助二分图.C 中的 k 个点及其 k 个复制点构成二分图左集的点,k 个点的所有邻接点构成二分图右集的点.

Step 7. 求二分图的最大匹配;

Step 8. if 最大匹配浸润了二分图左集的所有点 then

  输出“在图 G 中存在 k 个点不相交的 P_2”,算法结束.

Step 9. 输出“在图 G 中不存在 k 个点不相交的 P_2”,算法结束.

定理 4. 如果图 G 中存在 k 个点不相交的 P_2,则算法 *KPPW*(G,k)将返回该 k-P_2-Packing,否则返回不存在,且算法的时间复杂度为 $O^*(2^{4.142k})$.

证明:算法 KPPW 的 Step 2 的核心化是在文献[7]的核心化之后通过对算法 RPLW 的反复调用来实现的,从而找到一个大小最多为 $7k$ 个点的核.Step 3 是判断核心化后图 G 中点的大小,如果图 G 中的点超过核的大小,则说明核心化算法 RPLW 在还没有把图 G 的规模减小到 $7k$ 个点时,就已经找到一个 k-P_2-packing,所以算法无须再执行下去.Step 4~Step 8 通过对 $7k$ 个点的核的穷举来找出问题的最优解.穷举出核中所有可能的 k 个点的组合,并判断每个组合中的 k 个点是否是 k 个点不相交的 P_2 的 k 个中心点,因此,利用每个组合中的 k 个点及其邻接点构建一个辅助二分图,C 中的 k 个点及其 k 个复制点构成二分图左集的点,k 个点的所有邻接点构成二分图右集的点.若二分图的最大匹配浸润了二分图左集的所有点,则说明构成该二分图的 k 个点是 k 个中心点,故输出“在图 G 中存在 k 个点不相交的 P_2”,算法结束.若在所有组合中都不能找到这样的 k 个中心点,则说明在图 G 中不存在 k 个点不相交的 P_2,因此输出“在图 G 中不存在 k 个点不相交的 P_2”,算法结束.

假设算法 KPPW 的时间复杂度为 $Tp(k)$,下面是整个算法时间复杂度的分析.

Step 1 中用贪婪算法构造一个极大 P_2-packing W 可在多项式时间内完成.文献[7]的核心化可在 $O(n^3)$时间内完成,由以上分析可知,反复调用算法 RPLW 所需时间为 $O(n^3)$,所以,Step 2 的运行时间为 $O(n^3+k^3)$.Step 3 是判断核心化后图 G 中的点的大小,所以其运行时间为 $O(k)$.Step 4~Step 8 是在 $7k$ 个点的核中穷举出所有可能的 k 个点的组合,并对每一个组合 C,判断 C 中所包含的 k 个点是否为 k 个中心点.在 $7k$ 个点中穷举 k 个点共有 $\binom{7k}{k}$

种组合,由 Stirling 近似公式,这个值近似于 $2^{4.142k}$.由于最大二分图匹配能够在 $O\left(\sqrt{|V|}|E|\right)$ 内完成,因此由组合 C 中的 k 个点及其邻接点构成的二分图的最大匹配问题可以在时间 $O(k^{2.5})$内解决,故 Step 4~Step 8 的运行时间为 $O(2^{4.142k}k^{2.5})$.Step 9 可以在常数时间内执行,所以算法 KPPW 的时间复杂度 $Tp(k)=O(n^3+k^3+k+2^{4.142k}k^{2.5})$,由于多项式时间相对于指数时间来说是可以忽略的,所以算法 KPPW 的时间复杂度可以被表示成 $O^*(2^{4.142k})$,即 k-P_2-Packing 问题的参数算法时间复杂度为 $O^*(2^{4.142k})$. □

4 结束语

本文主要研究 k-P_2-Packing 问题的核心化算法.通过对问题结构的深入分析,提出了一种可以获得大小最多为 $7k$ 个点的核的核心化算法.与文献[7]的核心化算法不同之处在于:本文提出的核心化算法是在文献[7]的核心化算法之后对 W 中的 P_2 及其相连的 Q_0 点和 Q_1 点作进一步的优化处理,目的是为了减小 Q 中 Q_0 点和 Q_1 点的大小,从而获得了一个大小最多为 $7k$ 个点的核,在 $7k$ 个点的核的基础上给出了一种时间复杂度为 $O^*(2^{4.142k})$的参数算法.很明显,本文得到的结果比文献[7]提出的时间复杂度为 $O^*(2^{5.301k})$的参数算法要快很多.即使在 k 为 10 的时候,本文的算法也将比 Elena Prieto 的算法提高上千倍,这是目前关于这个问题的最好结果.

References:

[1] Hell P. Graph packings. Electronic Notes in Discrete Mathematics, 2000,5:170−173.

[2] Kann V. Maximum bounded *H*-matching is MAX-SNP-complete. Information Processing Letters, 1994,49(6):309−318.

[3] Alon N, Yuster R, Zwick U. Color-Coding. Journal of the ACM, 1995,42(4):844−856.

[4] Chen JE, Friesen DK, Jia WJ, Kanj IA. Using nondeterminism to design efficient deterministic algorithms. Algorithmica, 2004,40(2): 83−97.

[5] Fellows M, Heggernes P, Rosamond F, Sloper C, Telle JA. Finding *k* disjoint triangles in an arbitrary graph. In: In: Hromkovic J, Nagl M, Westfechtel B, eds. Proc. of the 30th Workshop on Graph Theoretic Concepts in Computer Science. Berlin: Springer-Verlag, 2004. 235−244.

[6] Mathieson L, Prieto E, Shaw P. Packing edge disjoint triangles: A parameterized view. In: Downey R, Fellows M, Dehne F, eds. Proc. of the 1st Workshop on Parameterized and Exact Computation. LNCS 3162, Berlin: Springer-Verlag, 2004. 127−137.

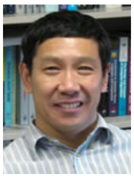
[7] Prieto E, Sloper C. Looking at the stars. Theoretical Computer Science, 2006,351(3):437−445.

[8] Diestel R. Graph Theory. 3rd ed., Berlin: Springer-Verlag, 2005. 2−28.

王建新(1969－),男,湖南邵东人,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算机优化算法,网络优化理论,生物信息学.

冯启龙(1982－),男,博士生,主要研究领域为参数计算.

宁丹(1979－),女,硕士生,主要研究领域为参数计算.



陈建二(1954－),男,博士,教授,博士生导师,主要研究领域为生物信息学,计算机理论,计算复杂性及优化,计算机网络优化算法,计算机图形理论与算法.