带权无向图中反馈顶点集的固定参数枚举算法

王建新 江国红 陈建二
(中南大学信息科学与工程学院 长沙 410083)

摘 要 反馈顶点集(FVS)问题是一个经典的 NP-完全问题,在很多领域有重要的应用.人们对该问题进行了大量的研究,但目前还没有有效的算法枚举带权无向图的反馈顶点集.文中通过对带权无向图中反馈顶点集问题的结构的深入分析,给出了一个有效的基于分支搜索技术的固定参数枚举算法.算法将反馈顶点集问题转化为反馈边集问题,通过枚举 z 个权值最大的森林来枚举 z 个权值最小的含 k 条边的反馈边集,从而得到 z 个权值最小的含 k 个顶点的反馈顶点集,算法时间复杂度为 $O(5^k n^2(\log n+k)+3^k z(n^2\log n+z))$.

关键词 反馈顶点集;无向图;带权;参数;固定参数枚举
中图法分类号 TP301 **DOI 号**: 10.3724/SP.J.1016.2010.01140

Fixed Parameter Enumeration Algorithm for Feedback Vertex Set in Weighted Undirected Graphs

WANG Jian-Xin JIANG Guo-Hong CHEN Jian-Er
(*School of Information Science and Engineering, Central South University, Changsha* 410083)

Abstract Feedback vertex set (FVS) problem is a classical NP-complete problem. There are important applications of the problem in many areas. Many studies have been done on the FVS problem, but there is no effective algorithm to enumerate the FVS in weighted undirected graphs. In this paper, a fixed-parameter enumeration algorithm based on the branch-and-search technique is given by analyzing the structure of the FVS problem in weighted undirected graphs. In the algorithm, the feedback vertex set problem is transformed to feedback edge set (FES) problem and the z minimum-weight FES of size k are enumerated by enumerating z maximum-weight forests, then the z minimum-weight FVS of size k can be found. The fixed-parameter enumeration algorithm for FVS runs in time $O(5^k n^2(\log n+k)+3^k z(n^2\log n+z))$.

Keywords feedback vertex set; undirected graphs; weighted; parameter; fixed-parameter enumeration

1 引 言

图 G 的一个反馈顶点集(Feedback Vertex Set, FVS)是一个由 G 中一些顶点构成的集合,从图 G 中删除该集合中的所有点后,图中不含圈,即图 G 中的每个圈至少有一个点在 FVS 中.如果图 G 为带权图,即给每个点分配一个非负实数的权值,那么 FVS 的权值是 FVS 中所有顶点的权值之和.找一个最小的 FVS(对带权图为找权值最小的 FVS,不

收稿日期:2008-12-17;最终修改稿收到日期:2010-06-21.本课题得到国家自然科学基金(60773111,60873265)、国家"九七三"重点基础研究发展规划项目基金前期研究专项(2008CB317107)和国家教育部创新团队资助计划(IRT0661)资助.**王建新**,男,1969 年生,博士,教授,博士生导师,主要研究领域为计算机算法、网络优化理论、生物信息学. E-mail: jxwang@mail.csu.edu.cn. **江国红**,女,1985 年生,硕士研究生,主要研究方向为参数计算、计算机理论. **陈建二**,男,1954 年生,博士,长江学者特聘教授,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.

带权图为找顶点数最少的 FVS)问题是一个经典的 NP-完全问题[1]. FVS 问题在电路测试、操作系统的解死锁、分析工艺流程和生物计算等领域都有应用[2]. FVS 问题最初在组合电路设计中被阐述. 在组合电路中,为了避免紊乱情况的发生,在电路中的每条环路上放置一个寄存器. 需要放置寄存器的节点数越少,总延迟就越短,寻求延迟最短的放置寄存器方案就等价于在对应的电路图中找含定点数最少的 FVS 问题. 另外,FVS 问题在生物信息计算领域基因组装问题中有着重要的应用.

由于 FVS 问题是 NP 难的,除非 NP=P,否则 FVS 问题是不可能有多项式时间算法的,因此人们首先从近似算法的角度展开了对 FVS 问题的研究. Bar-Yehuda 等提出了对不带权 FVS 近似度为 4 和对带权 FVS 近似度为 $\min(2\Delta^2, 4\log n)$ 的多项式时间可解的近似算法,其中 Δ 为图 G 中顶点的最大度数[3]. 在此基础上,Bafna 等提出了对不带权 FVS 近似度为 2 的近似算法[4]和带权 FVS 近似度为 $2-2/(2\Delta-2)$ 的基于"局部近似"的近似算法[5].

由于近似算法的结果不够精确,往往不能满足需求,人们开展了 FVS 的精确算法的研究. 对不带权 FVS,Razgon 等用分枝-剪枝策略(Branch-Prune)提出了时间复杂度为 $O(1.8899^n)$ 的精确算法[6],其中 n 是图 G 中顶点的数目. 在此基础上,Fomin 等利用加权分支技术提出了时间复杂度为 $O(1.7548^n)$ 的精确算法[7]. 在实际应用中,n 的数目往往比较大,这些算法无法应用于实际. 但人们发现在很多实际应用中,FVS 的大小 k(FVS 中所含顶点的个数)比较小,于是人们将 FVS 问题参数化. 不带权 FVS 问题的参数化定义[8]如下.

定义 1.　给定含有 n 个顶点的图 G 和非负整数 k,能否在图 G 中找到一个不大于 k 的 FVS.

人们对不带权 FVS 的参数化问题进行了大量的研究. Downey 与 Fellows[8]、Bodlaender[9]最早证明了不带权无向图中 FVS 问题是 FPT(Fixed-Parameter Tractable,即在时间 $f(k)n^{O(1)}$ 内可解,其中 $f(k)$是仅关于 k 的一个函数[8])的. 对于不带权无向图中 FVS 问题:文献[10]给出了时间复杂度为 $O((2k+1)^k n^2)$ 的确定性算法;Raman 等对短圈(长度短的圈)上的点进行分支,给出了时间复杂度为 $O(\max\{12^k, (4\log k)^k\}n^\omega)$($n^\omega$ 是求两个 $n\times n$ 矩阵的乘积的时间)的算法[11];文献[12]应用极值图理论得到时间复杂度为 $O((2\log k+2\log\log k+18)^k n^2)$ 的算法;基于结论:当图中顶点度数至少为 3 且存在一个大小不超过 $n^{1-\varepsilon}/3$ 的 FVS,那么图中最短圈的长度最大为 $6/\varepsilon(\varepsilon\geq 1/2)$,文献[13]利用分支技术提出了时间复杂度为 $O((12\log k/\log\log k+6)^k n^\omega)$ 的确定性算法;文献[14]和文献[15]利用迭代压缩技术,通过对 FVS 问题的结构进行分析,分别提出了时间复杂度为 $O((37.7)^k n^2)$ 和 $O((10.567)^k n^3)$ 的确定性算法;Chen 等利用分支搜索和迭代压缩技术,通过分析问题分支后的隐含特性,提出了时间复杂度为 $O(5^k kn^2)$ 的确定性算法[16],这是目前无向图中 FVS 问题的 FPT 算法的最好结果. 对于有向图,文献[17]证明了不带权有向图的 FVS 问题是 FPT 的,通过将有向图 FVS 问题转化为受限制的 Multicut 问题提出一个时间复杂度为 $O(4^k k!n^{O(1)})$ 的 FPT 算法.

Chen 等在文献[16]中给出了带权 FVS 问题的参数化定义,具体如下.

定义 2.　给定含有 n 个顶点的带权图 G 和整数 k,能否在图 G 中找到一个权值最小的不大于 k 的 FVS.

Becker 等对带权无向图中 FVS 问题提出了时间复杂度为 $O(6^k kn^2)$ 且概率为 $1-(1-1/6^k)^{c6^k}$ 的随机算法[18]. 对带权无向图中 FVS 问题,Chen 等首次证明了该问题是 FPT 的,并提出了一个时间复杂度为 $O(5^k kn^2)$ 的确定性 FPT 算法[16]. 带权有向图中 FVS 问题是否存在 FPT 算法仍然是一个开放性研究课题.

在实际应用中,我们经常遇到枚举与计数两类问题,在参数计算领域人们提出了参数化的枚举和计数问题[19-20]. 文献[20]给出了一个时间复杂度为 $O(2^k k^2+kn)$ 的算法枚举所有大小为 k 的点覆盖,其中 k 等于最小点覆盖的大小. 很多实际应用并不需要找问题所有的解,而只需要一定数目的最优解. 固定参数枚举算法的研究已经引起参数计算研究领域学者的关注. 文献[21]首次给出了固定参数可枚举的概念,即对于一个 NP 问题,若存在一个算法能够在 $f(k)n^{O(1)}z^{O(1)}$ 的时间内(其中 $f(k)$是关于 k 的一个函数)找出该问题的 z 个大小为 k 的最优解,则称该问题是固定参数可枚举的. 文献[21]给出了 Vertex Cover、Path 以及平面图中 Domnating Set 的固定参数枚举算法,所用技术分别为分支搜索、彩色编码和树分解.

枚举多个最优的 FVS 在实际应用中很有意义,尤其是在操作系统的解死锁的应用中. 下面给出无向图中 FVS 的固定参数枚举问题的定义.

定义 3(FVS 的固定参数枚举).　给定一个带权无向图 G 和两个非负整数 k,z,要求找出图 G 中

z 个权值最小的 k-FVS，即假设所有的 k-FVS 按权值非降序排列，取排在前面的 z 个 k-FVS，其中 k-FVS 为含有 k 个点的 FVS.

目前还没有 FVS 的固定参数枚举算法，本文对带权无向图的 FVS 问题提出一种有效的基于分支搜索技术的固定参数枚举算法. 本文首先求出给定图 G 的一个 k-FVS-F'，然后穷举 F' 的所有子集 F_1，由 F_1 构造图 G 的一种结构划分，对结构划分利用分支搜索技术得到一系列元组，各元组为分支搜索树上各叶子节点所表示的状态结构. 再对元组表示的状态结构进行分析，将 FVS 问题转化为反馈边集(Feedback Edge Set，FES，由 G 中一些边构成的集合，从图 G 中删除 FES 中的边将使得图 G 不含圈)问题，最后通过枚举 z 个权值最大森林来枚举 z 个权值最小的 FES. 整个算法的时间复杂度为 $O(5^k n^2(\log n+k)+3^k z(n^2\log n+z))$.

本文第 2 节给出相关定义和引理；第 3 节给出 FVS 的固定参数枚举子过程；第 4 节给出 FVS 的固定参数枚举算法；第 5 节是本文的结论.

2 相关定义和引理

$G=(V,E)$ 为一个含有 n 个顶点的图，V 是 G 的顶点集，E 是边集. 设 $W\subseteq V$，设 $G[W]=(W,E_W)$ 为由顶点子集 W 导出的图 G 的子图，其中边集合 $E_W=\{(u,v)\mid u,v\in W$ 且 $(u,v)\in E\}$. 设 v 是图 G 中一个顶点，$G\backslash v$ 表示从图 G 中删除 v 点以及以 v 为端点的边之后得到的新图，$V\backslash v$ 表示从顶点集 V 中删除 v.

为了更好地描述 FVS 的固定参数枚举算法，首先给出以下定义.

定义 4(IF-划分)[16]. 给定一个图 $G=(V,E)$，称三元组 (V_0,V_1,V_2) 是图 G 的独立集-森林划分(Independent- Forest Partition，简称 IF-划分)，其中 (V_0,V_1,V_2) 满足如下条件：

(1) $V_0\cup V_1\cup V_2=V$，并且 V_0，V_1 和 V_2 互不相交；

(2) $G[V_1]$ 和 $G[V_2]$ 都是森林；

(3) V_0 是独立集，V_0 中的每个点在图 G 中都是 2 度点，且其邻居都在 V_2 中.

定义 5(IF-划分 FVS). 给定图 G、图 G 的 IF-划分 (V_0,V_1,V_2) 和非负整数 k，IF-划分 FVS F 是图 G 的 FVS，并且满足 $F\subseteq V_0\cup V_1$，$|F|\leq k$.

定义 6. 给定一个图 G 和图 G 的 IF-划分 (V_0,V_1,V_2)，IF-划分 (V_0,V_1,V_2) 的压缩图是一个新图 $H=(V_H,E_H)$，其中 V_H 中每个点对应图 $G[V_2]$ 中的一个连通块，E_H 中每条边 (u,v) 对应 V_0 中的一个点 w，且 w 的两个邻居分别在点 u 和 v 对应的 $G[V_2]$ 的连通块中.

IF-划分 (V_0,V_1,V_2) 的压缩图 H 可由以下方法得出：首先将 $G[V_2]$ 中的每个连通块压缩为一个点作为压缩图 H 的顶点. 对于 V_0 中任意一个点 w，若 w 的两个邻居分别在图 H 的顶点 u 和 v 对应的 $G[V_2]$ 的连通块中，则在图 H 的顶点 u 和 v 之间连接一条边. 因为 V_0 中的点都是 2 度点，所以 V_0 中的点和 H 中的边一一对应. 图 H 中的连通块和图 $G[V_0\cup V_2]$ 中的连通块也一一对应. 如果图 G 是顶点带权图，给 H 中每条边赋予一个权值，边的权值等于边所对应的 V_0 中点的权值，这样图 H 是一个边带权的图.

设 $\tau(k,V_0,V_1,V_2)=k-(|V_0|-\sharp c(V_2)+1)$，$\varphi(k,V_0,V_1,V_2)=k-(|V_0|-\sharp c(V_2)+\sharp c(V_0\cup V_2))$，其中 $\sharp c(V_2)$ 和 $\sharp c(V_0\cup V_2)$ 分别是图 $G[V_2]$ 和图 $G[V_0\cup V_2]$ 中连通块的数目. 设 IF-划分 (V_0,V_1,V_2) 的压缩图为 H，由压缩图定义可知，压缩图 H 的任意一个不大于 k 的 FES 中的边所对应的 V_0 中的点集合都是图 $G[V_0\cup V_2]$ 的 FVS. 设 H 的最大生成树或最大生成森林为 T(由压缩图的每个连通块的生成树构成的森林为生成森林)，不在 T 中的边集为 H 的权值最小的 FES，也是所含边数最少的 FES，FES 中边对应的 V_0 中点的集合就是图 $G[V_0\cup V_2]$ 的权值最小的含顶点数最少的 FVS. 由于 H 中有 $\sharp c(V_2)$ 个点，H 的连通块数等于 $G[V_0\cup V_2]$ 中的连通块数 $\sharp c(V_0\cup V_2)$，则 T 中含有 $\sharp c(V_2)-\sharp c(V_0\cup V_2)$ 条边，那么 H 的 FES 至少含 $|V_0|-\sharp c(V_2)+\sharp c(V_0\cup V_2)$ 条边，即至少要删除 $|V_0|-\sharp c(V_2)+\sharp c(V_0\cup V_2)$ 个 V_0 中的点才能使得图 $G[V_0\cup V_2]$ 不含圈，同样说明图 G 的 IF-划分 FVS 中至少包含 $|V_0|-\sharp c(V_2)+\sharp c(V_0\cup V_2)$ 个 V_0 中的点，则 IF-划分 FVS 中最多包含 $\varphi(k,V_0,V_1,V_2)$ 个 V_1 中的点. 可见，$\tau(k,V_0,V_1,V_2)$ 和 $\varphi(k,V_0,V_1,V_2)$ 是 IF-划分 FVS 中包含 V_1 中点数目的上界.

令 $C(G)$ 表示图 G 是否含有圈，若图 G 含有圈，则 $C(G)=1$，否则 $C(G)=0$. 结合以上分析，得出以下引理.

引理 1. 给定图 G 的一个 IF-划分 (V_0,V_1,V_2) 和参数 k. 如果 $(V_1=\varnothing$ and $\varphi(k,V_0,V_1,V_2)>0)$ or $V_2=\varnothing$ or $(\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=0)$，则求图 G 中 IF-划分 (V_0,V_1,V_2) FVS 问题可以转化为求 IF-划分的压缩图 H 中不大于 k

的 FES 问题；如果（$\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=1$) or $\varphi(k,V_0,V_1,V_2)<0$，则找不到图 G 中 IF-划分(V_0,V_1,V_2) FVS.

证明. 如果 $V_1=\varnothing$，则 $G=G[V_0\cup V_2]$，则求 G 的 IF-划分 FVS 问题转化为求 IF-划分的压缩图 H 的 FES 问题. 压缩图 H 的任意一个不大于 k 的 FES 中的边所对应的 V_0 中的点集合都是 IF-划分 FVS.

如果 $V_2=\varnothing$，V_0 也必然等于 $\varnothing$，那么图 $G=G[V_1]$，其压缩图为空图，V_1 的任意一个子集都是图 G 的 FVS.

当 $\varphi(k,V_0,V_1,V_2)=0$ 时，图 G 的 IF-划分 FVS 中的 k 个点必须属于 V_0. 此时，如果 $C(G[V_1\cup V_2])=0$，且 V_1 和 V_2 不为空集，则把 V_1 中的点移到 V_2 后构成的元组 $(V_0,\varnothing,V_2\cup V_1)$ 是一个有效的 IF-划分，基于此划分同样求得原划分的 FVS. 如果 $C(G[V_1\cup V_2])=1$，则找不到 IF-划分 FVS.

如果 $\varphi(k,V_0,V_1,V_2)<0$，要删除多于 k 个点使得图 $G[V_0\cup V_2]$ 不含圈，那么图 G 不存在 IF-划分 FVS. 由此证明了此引理. 证毕.

下面给出文献[16]中关于 FVS 问题的重要引理.

引理 2[16]. 给定含 n 个点的图 G，可以在 $O(5^k kn^2)$ 时间内找到一个不大于 k 的 FVS 或者证明图 G 不存在不大于 k 的 FVS.

3 FVS 的固定参数枚举子过程

带权无向图的固定参数枚举过程简述如下：首先在图中找一个大小为 k 的 FVS-F'，接着找出图 G 中所有基于 F' 的 IF-划分. 然后对每个 IF-划分，枚举 z 个权值最小的 k-FVS. 最后从所有求得的 k-FVS 中选取 z 个权值最小的 k-FVS，即为问题的解. IF-划分 FVS 的枚举过程为：首先利用分支搜索技术得到一组表示分支搜索树中叶子节点状态的元组，然后枚举各元组对应的 z 个权值最小 k-FVS. 本节主要介绍枚举 IF-划分 FVS 中两个主要步骤：元组构造和基于元组的局部枚举.

3.1 元组构造

元组构造过程是为了求得满足（$V_1=\varnothing$ and $\varphi(k,V_0,V_1,V_2)>0$) or $V_2=\varnothing$ or（$\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=0$）的 IF-划分 (V_0,V_1,V_2)，k 表示对当前的 IF-划分所要找的 FVS 大小. 用元组 $r=(G,V_0,V_1,V_2,F,A,S)$ 表示找到满足条件时图的状态，其中 G 是元组对应的图，A 是元组构造算法中产生的度数不大于 1 的点集合，F 为必须放入 FVS 的点集合，S 中的元素是顶点集，每个顶点集中的点是待处理的点，且同一个集合中的所有点在原图中构成一条路径.

对一个 IF-划分 (V_0,V_1,V_2)，元组构造算法 *ConstructGroup*(G,V_0,V_1,V_2,k,F,A,S) 依次考虑 V_1 中满足不同条件的点，然后对各类点进行相应操作，得到一组元组，且元组中划分 (V_0,V_1,V_2) 满足 $V_1=\varnothing$ 且 $\varphi(k,V_0,V_1,V_2)>0$ 或 $V_2=\varnothing$ 或 $\varphi(k,V_0,V_1,V_2)=0$ 且图 $G[V_1\cup V_2]$ 不含圈，算法见图 1. 以下分析元组构造算法的各个步骤.

ConstructGroup (G,V_0,V_1,V_2,k,F,A,S)
输入：图 $G=(V,E)$ 和图 G 的一个 IF-划分 (V_0,V_1,V_2)，整数 k，$F=A=S=\varnothing$
输出：一个集合 R，R 收集了一系列元组 $r=(G,V_0,V_1,V_2,F,A,S)$，其中，A 是算法中产生的度数不大于 1 的点的集合，F 是必须放入 FVS 中的点的集合，S 中的元素是顶点集，每个顶点集中点是待处理的 2 度点且所有这些点能构成一条路径
1. $R=\varnothing$；
2. if ($V_1=\varnothing$ and $\varphi(k,V_0,V_1,V_2)>0$) or ($V_2=\varnothing$) or ($\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=0$)，将元组 $r=(G,V_0,V_1,V_2,F,A,S)$ 加到 R 中，返回 R；
3. if ($\varphi(k,V_0,V_1,V_2)<0$) or ($\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=1$)，返回 R；
4. if V_1 中存在点 w 在图 G 中的度数不大于 1，则返回 ***ConstructGroup*** $(G\backslash w,V_0,V_1\backslash w,V_2,k,F,A\cup\{w\},S)$；
5. else if V_1 中存在点 w 有 2 个邻居在 V_2 中，且在图 G 中的度数为 2，则返回 ***ConstructGroup*** $(G,V_0\cup\{w\},V_1\backslash w,V_2,k,F,A,S)$；
6. else if V_1 中存在点 w 至少有 2 个邻居在 V_2 中而且在图 G 中的度数大于 2
6.1. {if w 至少有 2 个 V_2 中的邻居在图 $G[V_2]$ 的同一块中，则返回 ***ConstructGroup*** $(G\backslash w,V_0,V_1\backslash w,V_2,k-1,F\cup\{w\},A,S)$；
6.2. else ***ConstructGroup*** $(G\backslash w,V_0,V_1\backslash w,V_2,k-1,F\cup\{w\},A,S)\cup$ ***ConstructGroup*** $(G,V_0,V_1\backslash w,V_2\cup\{w\},k,F,A,S)$；}
7. else 从图 $G[V_1]$ 中选择任意一棵树中的最深的叶子节点 w_1，w 是 w_1 的父亲节点，w 的孩子节点为 $w_1,\cdots,w_t$
7.1. {if w 有一个邻居在 V_2 中或者 w 有多个孩子，则返回 ***ConstructGroup*** $(G\backslash w,V_0,V_1\backslash w,V_2,k-1,F\cup\{w\},A,S)\cup$ ***ConstructGroup*** $(G,V_0\cup\{w_1,\cdots,w_t\},V_1-\{w,w_1,\cdots,w_t\},V_2\cup\{w\},k,F,A,S)$；
7.2. else 将 V_1 中点 w_1 着成红色，且从图 G 中压缩 w，并将 w 放入 S 中包含 w_1 的集合 S_{w_1}，设压缩 w 后的图为 G_c，返回 ***ConstructGroup*** $(G_c,V_0,V_1\backslash w,V_2,k,F,A,S\cup\{S_{w_1}\})$；}

图 1　元组构造算法

(1) 当($V_1=\varnothing$ and $\varphi(k,V_0,V_1,V_2)>0$) or $V_2=\varnothing$ or ($\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=0$) 成立时,由引理 1,可以将 FVS 问题转化为 FES 问题,此时把元组 $r=(G,V_0,V_1,V_2,F,A,S)$ 放入集合 R.

(2) 如果 $\varphi(k,V_0,V_1,V_2)<0$ 或 ($\varphi(k,V_0,V_1,V_2)=0$ 且 $C(G[V_1\cup V_2])=1$),由引理 1,在图中找不到 IF-划分(V_0,V_1,V_2)的 k-FVS. 因此当算法步 2 条件满足,结束递归分支.

(3) 如果 V_1 中存在一个点 w 在图 G 中的度数不大于 1,则 w 不在图 G 的任何圈中. 由于需要枚举 k-FVS,可能将 w 作为冗余点加入小于 k 的 FVS 使得 FVS 中含 k 个点,于是将 w 保存于 A.

(4) 如果 V_1 中存在一个点 w 在图 G 中是 2 度点,且它的两个邻居都在 V_2 中,根据 IF-划分的定义,将 w 从 V_1 移入 V_0 后同样构成有效的 IF-划分.

(5) 如果 V_1 中存在一个点 w 至少有两个邻居在 V_2 中,且 w 在图 G 中的度数大于 2,若 w 在 V_2 中的 2 个邻居属于 $G[V_2]$ 的同一棵树,则图 $G[V_2\cup\{w\}]$ 中含有圈,需将 w 加入 FVS,即将 w 加入 F 且将 w 从 G 中删除,k 减 1;若 w 在 V_2 中的邻居属于 $G[V_2]$ 的不同树,则对 w 进行分支:分支 1,w 属于 FVS,即将 w 加入 F 且将 w 从 G 中删除,k 减 1;分支 2,w 不属于 FVS,即将 w 从 V_1 移入 V_2.

(6) 如果 V_1 中的点在图 G 中的度数均大于 1,且每个点最多有一个邻居在 V_2 中时,设 T 为 $G[V_1]$ 中一棵树,任意设一个点作为 T 的根节点,则总能找到 T 中最深的(与根节点距离最远)叶子节点,设此叶子节点为 w_1,其父亲节点为 w,且 $w_1,\cdots,w_t$ 均为 w 的孩子. 容易分析出 $w_1,\cdots,w_t$ 度数均为 2 且只有一个邻居在 V_2 中. 下面分两种情况讨论.

情况 1. 若 w 有一个邻居或者多个孩子在 V_2 中,则对 w 进行分支:分支 1,w 属于 FVS,即将 w 加入 F 且将 w 从 G 中删除,k 减 1. 分支 2,w 不属于 FVS,由于 $w_1,\cdots,w_t$ 在图中的度数均为 2,且均有 1 个邻居在 V_2 中,如果将 w 移入 V_2,则 $w_1,\cdots,w_t$ 有 2 个邻居在 V_2 中,故将 w 移到 V_2 中的同时将 $w_1,\cdots,w_t$ 移入 V_0.

情况 2. 若 w 没有邻居在 V_2 且只有一个孩子,由于 w 的度数大于 1,故 w 不可能是根节点,则 w 的度数为 2,w_1 和 w 是两个相邻的 2 度点. 对 w 点进行压缩(设 w' 是 w 的父亲节点,压缩 w 点是从图 G 中删除 w 并连接 w_1 和 w')并将 w 放入包含 w_1 的集合 S_{w_1},可见 S_{w_1} 中的点在压缩前均为 2 度点,且 S_{w_1} 中所有点在原图中构成一条路径. 如果 S_{w_1} 已经在 S 中,则更新 S 中的 S_{w_1},如果 S_{w_1} 不在 S 中,则新建一个包含 w_1 和 w 的集合 S_{w_1},并将 S_{w_1} 加入 S,算法中统一用 $S\cup\{S_{w_1}\}$ 表示. 为了将 w_1 和其它点区分开,将 V_1 中点 w_1 着成红色. w_1 移到其它集合后颜色状态保持不变,则 V_2 中不可能有红点(如果 w_1 的父亲节点 w 不被压缩,则 w 有邻居在 V_2 中或者 w_1 的父亲节点有多个孩子,根据前面的步骤,w_1 的父亲节点或者属于 F,或者属于 V_2. 如果将 w_1 的父亲节点放入 F,则 w_1 的度数变为 1,将 w_1 放入 A_0 中;如果将 w_1 的父亲节点放入 V_2,则 w_1 的两个邻居都在 V_2 中,将 w_1 放入 V_0).

引理 3. 算法 *ConstructGroup* (G,V_0,V_1,V_2,k,F,A,S) 返回一个元组集合 R,且对图 G 中 IF-划分(V_0,V_1,V_2)的任意一个 k-FVS,在元组集合 R 中一定存在一个元组 r 与该 k-FVS 相对应.

证明. 设 F_1 是图 G 中 IF-划分(V_0,V_1,V_2)的一个 k-FVS,则 F_1 是 $V_0\cup V_1$ 的子集. 如果元组 $r=(G,V_0,V_1,V_2,F,A,S)$ 满足 $F\subseteq F_1$ 且 F_1 与 R 中的 V_2 的交集为空集,那么称 F_1 与元组 r 对应. 元组构造算法的递归过程对应一棵分支搜索树,且算法只在步 6.2 和步 7.1 产生两个分支,假设 v 是步 6.2 或步 7.1 步所考虑的点,对 v 点进行分支后,v 或者属于 F,或者属于 V_2,可见在这两个分支下的叶子节点代表的元组是对立的. 如果 F_1 包含 v,则与 F_1 对应的元组 r 在 v 属于 F 的分支中,如果 F_1 不含 v,则 r 在 v 属于 V_2 的分支中. 因此,要找到 F_1 所对应的元组 r,可从上往下在分支搜索树中搜索,当只有一个节点只有一个分支时直接考虑此节点的孩子节点(根据前面的算法分析,对点 v 只有一个分支且分支中将 v 加到 F,则 v 也一定属于 F_1,否则 F_1 不是 IF-划分(V_0,V_1,V_2)的 k-FVS);当点 v 有两个分支时,如果 v 属于 F_1,则在 v 属于 F 的分支中找,否则在 v 属于 V_2 的分支找,最后总能找到 F_1 所唯一对应的叶子节点. 分支搜索树中一个叶子节点对应一个元组,由元组构造算法可知,对于不属于 R 的元组,不存在与其对应的 IF-划分 FVS,而 F_1 是 IF-划分的 k-FVS,因此 F_1 和 R 中唯一一个元组对应.

证毕.

下面分析算法产生元组的数目和时间复杂度.

引理 4. 算法 *ConstructGroup* (G,V_0,V_1,V_2,k,F,A,S) 在 $O(2^{\tau(k,V_0,V_1,V_2)}n^2)$ 时间内返回元组集合 R,且 R 中最多包含 $2^{\tau(k,V_0,V_1,V_2)}$ 个元组,其中 $\tau(k,V_0,V_1,V_2)=k-(|V_0|-\sharp c(V_2)+1)$,$\sharp c(V_2)$ 是

$G[V_2]$中连通块的数目.

证明.　R中元组的个数不大于算法对应的分支搜索树中叶子节点的数目.用$T(k,V_0,V_1,V_2)$表示分支搜索树中叶子节点的数目.下面用归纳法证明$T(k,V_0,V_1,V_2)\leq\max\{1,2^{\tau(k,V_0,V_1,V_2)}\}$.

如果$\tau(k,V_0,V_1,V_2)<0$,则$\varphi(k,V_0,V_1,V_2)<0$,根据算法*ConstructGroup*步3可知此时$T(k,V_0,V_1,V_2)=1\leq\max\{1,2^{\tau(k,V_0,V_1,V_2)}\}$.对于其它情况,假设某节点的孩子节点的子树中叶子节点数目$T(k',V_0',V_1',V_2')\leq 2^{\tau(k',V_0',V_1',V_2')}$,只要证明该节点的子树中叶子节点数目$T(k,V_0,V_1,V_2)\leq 2^{\tau(k,V_0,V_1,V_2)}$就可以证明该引理.

在算法*ConstructGroup*的步2返回一个元组,可知此时$T(k,V_0,V_1,V_2)=1\leq\max\{1,2^{\tau(k,V_0,V_1,V_2)}\}$.

算法步4删除G中一个点w,k、V_0和V_2不变,则$T(k,V_0,V_1,V_2)=T(k,V_0,V_1\backslash w,V_2)$.由于$\tau(k,V_0,V_1\backslash w,V_2)=\tau(k,V_0,V_1,V_2)$,故可得$T(k,V_0,V_1,V_2)=T(k,V_0,V_1\backslash w,V_2)\leq 2^{\tau(k,V_0,V_1,V_2)}$.

算法步5把w从V_1移到V_0,则$T(k,V_0,V_1,V_2)=T(k,V_0\cup\{w\},V_1\backslash w,V_2)$.由于$\tau(k,V_0\cup\{w\},V_1\backslash w,V_2)=\tau(k,V_0,V_1,V_2)-1$,故$T(k,V_0,V_1,V_2)=T(k,V_0\cup\{w\},V_1\backslash w,V_2)\leq 2^{\tau(k,V_0\cup\{w\},V_1\backslash w,V_2)}<2^{\tau(k,V_0,V_1,V_2)}$.

步6.1 V_0和V_2不变,k减1,$T(k,V_0,V_1,V_2)=T(k-1,V_0,V_1\backslash w,V_2)$.由于$\tau(k-1,V_0,V_1\backslash w,V_2)=\tau(k,V_0,V_1,V_2)-1$,故$T(k,V_0,V_1,V_2)=T(k-1,V_0,V_1\backslash w,V_2)\leq 2^{\tau(k-1,V_0,V_1\backslash w,V_2)}<2^{\tau(k,V_0,V_1,V_2)}$.

步6.2对w进行分支,$T(k,V_0,V_1,V_2)=T(k-1,V_0,V_1\backslash w,V_2)+T(k,V_0,V_1\backslash w,V_2\cup\{w\})$.分支1和步6.1相同,即$\tau(k-1,V_0,V_1\backslash w,V_2)=\tau(k,V_0,V_1,V_2)-1$.步6.2中,$w$在$V_2$中的邻居分别在$G[V_2]$的不同树中,故将$w$加入$V_2$中至少连接$G[V_2]$中的两个连通块,连通块的数目至少减1,因此$\sharp c(V_2\cup\{w\})\leq\sharp c(V_2)-1$,于是有$\tau(k,V_0,V_1\backslash w,V_2\cup\{w\})=k-(|V_0|-\sharp c(V_2\cup\{w\})+1)\leq\tau(k,V_0,V_1,V_2)-1$.故

$$\begin{aligned}T(k,V_0,V_1,V_2)&=T(k-1,V_0,V_1\backslash w,V_2)+\\&\quad T(k,V_0,V_1\backslash w,V_2\cup\{w\})\\&\leq 2^{\tau(k,V_0,V_1,V_2)-1}+2^{\tau(k,V_0,V_1,V_2)-1}\\&=2^{\tau(k,V_0,V_1,V_2)}.\end{aligned}$$

下面考虑步7.1.设$V_0'=V_0\cup\{w_1,\cdots,w_t\}$,$V_1'=V_1\backslash\{w,w_1,\cdots,w_t,\}$,$V_2'=V_2\cup\{w\}$,则

$$T(k,V_0,V_1,V_2)=T(k-1,V_0,V_1\backslash w,V_2)+T(k,V_0',V_1',V_2')\quad(1)$$

步7.1的分支1和步6.1相同,因此

$$T(k-1,V_0,V_1\backslash w,V_2)\leq 2^{\tau(k-1,V_0,V_1\backslash w,V_2)}<2^{\tau(k,V_0,V_1,V_2)}\quad(2)$$

步7.1的分支2中,$|V_0'|=|V_0|+t$,若w有一个邻居在V_2中,则$t\geq 1$,若w没有邻居在V_2中,且w有多个孩子,则$t\geq 2$,参照对步5和步6.2分支2的分析,同样可以得出$\tau(k,V_0',V_1',V_2')\leq\tau(k,V_0,V_1,V_2)-1$.故

$$T(k,V_0',V_1',V_2')\leq 2^{\tau(k,V_0',V_1',V_2')}\leq 2^{\tau(k,V_0,V_1,V_2)-1}\quad(3)$$

由不等式(1)、(2)和(3)可得

$$T(k,V_0,V_1,V_2)\leq 2^{\tau(k,V_0,V_1,V_2)}.$$

步7.2删除V_1中一个点w,k、V_0和V_2不变,则$T(k,V_0,V_1,V_2)=T(k,V_0,V_1\backslash w,V_2)\leq 2^{\tau(k,V_0,V_1,V_2)}$.

综上所述,元组构造算法的递归过程对应的分支搜索树中叶子节点个数$T(k,V_0,V_1,V_2)\leq 2^{\tau(k,V_0,V_1,V_2)}$,即$R$中的元组个数最多为$2^{\tau(k,V_0,V_1,V_2)}$.

分支搜索树中由根节点到叶子节点的路径上,除步1~3外,每执行一步,V_1中的顶点数至少减1,则元组构造算法中各步的执行次数不超过n.步1时间复杂度为$O(1)$,容易分析出其它各步的时间复杂度均为$O(n)$.分支搜索树中由根节点到叶子节点的路径不超过$2^{\tau(k,V_0,V_1,V_2)}$条(路径条数等于叶子节点数),因此元组构造算法的运行时间为$O(2^{\tau(k,V_0,V_1,V_2)}n^2)$.

证毕.

3.2　基于元组的局部枚举

设G'表示原图,k为要找的图G'中FVS的大小,$r=(G,V_0,V_1,V_2,F,A,S)$是对图G'经元组构造算法得出的一个元组,简称为图G'的元组.F中的点必须放入FVS,S中的元素为顶点集,设S中所有顶点集的并集为V_S,那么A,V_0,V_1,V_S中的点均可能是FVS中的点.设$k_1=k-|F|$,图$G'[V-F]$中基于元组r的k_1-FVS只包含$A\cup V_0\cup V_1\cup V_S$中点.图$G'[V-F]$中基于元组$r$的$k_1$-FVS加上$F$中的点就可以构成图$G'$的基于元组$r$的$k$-FVS.基于元组的局部枚举主要过程是对图$G'[V-F]$中基于元组$r$的$z$个权值最小的$k_1$-FVS进行枚举.本文首先将FVS问题转化为FES问题,然后通过枚举z个权值最大的森林来枚举z个权值最小的k_1-FES,从而找到z个权值最小的k_1-FVS.

3.2.1　将FVS问题转化为FES问题

由引理1,如果($V_1=\varnothing$ and $\varphi(k,V_0,V_1,V_2)>0$) or $V_2=\varnothing$ or ($\varphi(k,V_0,V_1,V_2)=0$ and $C(G[V_1\cup V_2])=0$)成立,可将FVS转化为FES问题.由于IF-划分k-FVS最多有$\varphi(k,V_0,V_1,V_2)$个点属于

V_1,如果 $\varphi(k_1,V_0,V_1,V_2)=0$,则 k-FVS 都是 V_0 中的点,将 V_1 中的点移入 V_2. 如果 $V_2=\varnothing$,由 IF-划分的定义,$V_0=\varnothing$,那么 $G=G[V_1]$,由于 $G[V_1]$是森林,这样 V_1 中点不在任何圈中,可把 V_1 中的点全部移入 A.

根据以上分析,给定一个元组构造算法得出的元组,要将 FVS 转化为 FES 问题,需将 $G[V_2]$中连通块转化为点,然后将 $A\cup V_0\cup V_S$ 中的点全部转化为边,具体转化方法如下.

(1) 对一个元组 $r=(G,V_0,V_1,V_2,F,A,S)$,如果 $\varphi(k_1,V_0,V_1,V_2)=0$,令 $V_0'=V_0$,$V_1'=\varnothing$,$V_2'=V_1\cup V_2$,$A'=\varnothing$;如果 $V_2=\varnothing$,则令 $V_0'=V_0$,$V_2'=V_2$,$V_1'=\varnothing$,$A'=A\cup V_1$. 然后令元组 $r=(G,V_0',V_1',V_2',F,A',S)$.

(2) 对元组 r 中的 IF-划分(V_0,V_1,V_2)构造压缩图 H.

(3) 对 V_0 中的红点 w(此处及后文提到的红点均为元组构造算法中产生的红点),S 中的元素$S_w=\{w,w_1,\cdots,w_{l-1}\}$是包含 w 的集合,在图 $G'[V_0\cup V_2\cup S_w]$中存在一条由 S_w 中所有点构成的路径 $P=\{(w,w_1),(w_1,w_2),\cdots,(w_{l-2},w_{l-1})\}$. 因为压缩图 H 中每条边都是由 V_0 中点转化而成的,即 H 中每条边对应 V_0 中一个点,可以找到图 H 中与 w 对应的边(u,v). 删除图 H 中边(u,v),然后在图 H 中添加 $l-1$ 个点 $u_1,\cdots,u_{l-1}$ 以及添加边(u,u_1),(u_1,u_2),$\cdots$,(u_{l-2},u_{l-1}),(u_{l-1},v),这 l 条边的权值分别等于 P 中点 $w,w_1,\cdots,w_{l-1}$的权值. 这样 H 中边(u,u_1)对应 P 中点 w;边(u_i,u_{i+1})对应 P 中点 w_i,对 $1\leq i\leq l$;边(u_{l-1},v)对应 P 中的点 w_{l-1},即 H 中路径 $P_H=\{(u,u_1),(u_1,u_2),\cdots,(u_{l-2},u_{l-1}),(u_{l-1},v)\}$与图 $G'[V_0\cup V_2\cup S_w]$中路经 P 对应.

(4) 对 V_0 中每个红点执行步 3 后,得到图 H_r.

(5) 对 A 中每个红点 w,S_w是包含 w 的集合,将 S_w 中的点并到 A 中,设得到的 $A=\{w_1,w_2,\cdots,w_{|A|}\}$. 然后在 H_r 中加 $2|A|$ 个新点 $u_1,v_1,\cdots,u_{|A|},v_{|A|}$ 以及 $|A|$ 条边$(u_1,v_1),\cdots,(u_{|A|},v_{|A|})$,这 $|A|$ 条边的权值分别等于 A 中点 $w_1,w_2,\cdots,w_{|A|}$ 的权值. 这样 H_r 中新添加的 $|A|$ 条边与 A 中点一一对应.

由 H_r 的构造方法可知 H_r 中边与 $A\cup V_0\cup V_S$ 中点一一对应. 设 A 中有 d 个红点,且包含 A 中各红点的集合的并集为 V_S',每添加一条孤立边到 H_r 将增加两个点;每将一条边扩展为一条长为 l 的路径增加 $l-1$ 个点,l 等于包含某红点集合的大小,而且需扩展的路径条数为 $|S|-d$. 因此图 H_r 中边的数目 $|E|=|A\cup V_0\cup V_S|\leq n$,点的数目 $|V|=\#c(V_2)+(|V_S-V_S'|-(|S|-d))+2|A\cup V_S'|\leq 2n$,其中 n 为图 $G'[V-F]$中顶点数.

引理 5. 给定图 $G'=(V,E)$、整数 k 和图 G'的一个元组 r,设 $k_1=k-|F|$,枚举图 $G'[V-F]$的基于此元组的 z 个权值最小的 k_1-FVS 可转化为枚举图 H_r 中 z 个权值最小的 k_1-FES,其中 H_r 是由元组 r 转化得到的.

证明. 设元组 $r=(G,V_0,V_1,V_2,F,A,S)$,S 中包含 A 中各红点的集合的并集为 V_S'. A 中的点是在元组算法中产生的度数不大于 1 的点,这些点不在任何圈中,因此可以把 $A\cup V_S'$中的点都当作 0 度点处理. 在图 $G'[V-F]$中删除以 $A\cup V_S'$中的点为端点的边,得到的新图为 G_0,则 G_0 比图 $G'[V_0\cup V_2\cup(V_S-V_S')]$多 $|A\cup V_S'|$ 个孤立点,显然求图 $G'[V-F]$基于元组 r 的 FVS 等价于求图 G_0 中不含 V_2 中的点的 FVS.

如果 $A=\varnothing$且 V_0 中不含红点,图 G 的压缩图为 H,那么 $G_0=G[V_0\cup V_2]=G$,$H_r=H$,根据引理 1,求图 G_0 中所有基于元组 r 的 k-FVS 可转化为求图 H_r 中所有 k-FES 问题.

如果 A 不为空,根据图 H_r 的构造方法,在图 G_0 中,每个 $A\cup V_S'$中的点在图 H_r 中对应一条边. 如果 V_0 中含红点,对 V_0 中的红点 w_1,由元组构造算法步 7.2 可知,包含 w_1 的集合 S_{w_1} 中的点在压缩前都是 2 度点,那么在 G_0 也是 2 度点,并且 S_{w_1} 中的点在图 G_0 中构成一条路径 P,根据图 H_r 的构造方法,路径 P 对应于 H_r 中一条路径 P_H. 每个 V_0 中的非红点对应 H_r 中一条边. 因此,对于 A 不为空且 V_0 中含红点,A 不为空且 V_0 中不含红点以及 A 为空且 V_0 中含红点这 3 种情况,都满足 H_r 的边与 $A\cup V_0\cup V_S$ 中的点一一对应,H_r 的连通块与 G_0 的连通块一一对应. 因此,求图 G_0 中所有基于元组 r 的 k-FVS 可转化为求图 H_r 中所有 k-FES 问题.

综上所述,枚举图 $G'[V-F]$基于元组 r 的 z 个权值最小的 k_1-FVS 可以转化为枚举元组 r 转化的图 H_r 中 z 个权值最小的 k_1-FES. 证毕.

3.2.2 枚举 z 个权值最大的森林

根据引理 5,枚举 z 个权值最小的 k_1-FVS 可转化为枚举图 $H_r=(V_r,E_r)$ 的 z 个权值最小的 k_1-FES,而枚举 z 个权值最小的 k_1-FES 可通过找 H_r 的 z 个权值最大的、含 $|E|-k_1$ 条边的森林来实现. 对于每个森林,不在森林中的边集合就是 H_r 的 k_1-FES,z 个权值最大的且含 $|E|-k_1$ 条边的森林对应

z 个权值最小的 k_1-FES.

枚举图 H_r 的 z 个权值最大、含 t 条边的森林的算法思想为：假设图中所有含 t 条边的森林包含于一个森林集合 P，首先找出 P 中权值最大的森林，然后用此森林将 P 划分为互不相交的子集，接着求取各子集中权值最大的森林，比较得出其中权值最大的森林为 P 中权值第 2 大的森林，再用权值第 2 大的森林对包含它的子集做进一步划分. 依此下去，直到找出权值第 z 大的森林. 由此可见，枚举森林的核心过程为找一个森林集合中权值最大的森林和对森林集合 P 的划分，下面对这两个过程进行介绍.

(1) 找一个森林集合中权值最大的森林

设 P 是图 H_r 中含 t 条边的森林集合，如果 P 表示为以下形式

$$P=\{T\mid(i_1,j_1)\in T,\cdots,(i_r,j_r)\in T,(m_1,p_1)\notin T,\cdots,(m_h,p_h)\notin T,|T|=t\}.$$

$X_P=\{(i_1,j_1),\cdots,(i_r,j_r)\}$，$Y_P=\{(m_1,p_1),\cdots,(m_h,p_h)\}$，设 $T(X_p,Y_p)$ 为包含 X_P 中边但不包含 Y_P 中边、含 t 条边的森林，那么 P 为所有 $T(X_p,Y_p)$ 的集合. 容易看出二元组 (X_P,Y_P) 能唯一确定一个森林集合 P，把集合 P 中森林 $T(X_p,Y_p)$ 也称为基于 (X_P,Y_P) 的森林.

引理 6.　给定一个图 $H_r(V_r,E_r)$ 和二元组 (X,Y)，在 $O(|E_r|\log|V_r|)$ 时间内能够找出权值最大的 $T(X,Y)$，$T(X,Y)$ 为包含 X 中边但不包含 Y 中边且含 t 条边的森林.

证明.　求一个权值最大的 $T(X,Y)$ 的算法可参照求解最小生成树算法[22]来实现，即首先将 X 中的边放入 T，然后从图 H_r 中删除 X 和 Y 中的边，接着根据权值由大到小的顺序依次对 H_r 的剩余边进行考虑，如果一条边加到当前找到的森林 T 中后不构成圈，则将此边加到 T 中并从图中删除，否则从图中直接删除此边，再考虑下一条边，直到 T 中含有 t 条边. 根据 Kruskal 算法的分析，容易得出求一个权值最大的 $T(X,Y)$ 的时间复杂度为 $O(|E_r|\log|V_r|)$.

证毕.

如果森林集合 P 由二元组 (X_P,Y_P) 确定，那么可通过引理 6 求取集合 P 中权值最大的含 t 条边的森林.

(2) 对森林集合 P 进行划分

设森林集合 P 由二元组 (X_P,Y_P) 确定，P 中权值最大的森林为 $T(P)$，$T(P)$ 中不属于 X_P 的边为 $e_1,e_2,\cdots,e_h$. 用 $T(P)$ 将集合 P 划分成互不相交的子集等价于将二元组 (X_P,Y_P) 进行划分，划分 (X_P,Y_P) 的步骤为：对整数 i，$1\le i\le h$，将 $e_1,e_2,\cdots,e_{i-1}$ 加到 X_P，并将 e_i 加到 Y_P 构成新的二元组 (X_i,Y_i)，即 $X_i=X_P\cup\{e_1,\cdots,e_{i-1}\}$，$Y_i=Y_P\cup\{e_i\}$. 所构成的 h 个二元组为以下形式：$(X_1,Y_1)=(X_P,Y_1=Y_P\cup\{e_1\})$，$(X_2,Y_2)=(X_P\cup\{e_1\},Y_2=Y_P\cup\{e_2\})$，$\cdots$，$(X_h,Y_h)=(X_P\cup\{e_1,\cdots,e_{h-1}\},Y_h=Y_P\cup\{e_h\})$. 很明显，基于 (X_P,Y_P) 的森林除去 $T(P)$ 后等于基于各二元组 $(X_1,Y_1),\cdots,(X_h,Y_h)$ 的森林的并集.

根据以上分析，H_r 中所有含 t 条边的森林集合 P 由二元组 $(X=\varnothing,Y=\varnothing)$ 确定，枚举森林算法设计如下：首先找出图 H_r 中基于 (X,Y) 的权值最大的森林 T_1，为图 H_r 中权值最大的森林. 然后用 T_1 将 (X,Y) 划分为互不相交的二元组，将所有二元组放入 L，再找出基于 L 中各二元组的权值最大的森林，从中选出权值最大的一个为 T_2，T_2 为权值第 2 大的森林，用 T_2 将 T_2 所对应的二元组进行划分，依此下去直到找出第 z 大的森林. 具体算法过程见图 2.

Enumerate-Forest (H_r,z,t)
输入：图 H_r，整数 z,t
输出：z 个权值最大的含 t 条边的森林
1. 二元组集合 $L=\{(X=\varnothing,Y=\varnothing)\}$；
2. $U_T=\varnothing$；　//U_T 保存找到的含 t 条边的森林
3. for($i=1$；$i\le z$；$i++$)
4. {在图 H_r 上分别找出与序列 L 中各元组对应的权值最大的含 t 条边的森林，设其中权值最大的含 t 条边的森林为 T，T 对应的二元组为 (X_1,Y_1)；
5.　　$T_i=T$，将 T_i 放入 U_T；
6.　　用 T_i 对 (X_1,Y_1) 进行划分，并将新产生的二元组插入 L，从 L 中删除 (X_1,Y_1)；}
7. 返回 U_T；

图 2　枚举 z 个权值最大的含 t 条边的森林

引理 7.　给定图 $H_r(V_r,E_r)$，算法 *Enumerate-Forest*(H_r,z,t) 可以在 $O(zt|E_r|\log|V_r|+z^2)$ 时间内找出 z 个权值最大的、含 t 条边的森林.

证明.　算法 *Enumerate-Forest*(H_r,z,t) 中每执行完一次步 3 的循环将生成一个森林.

第 1 次循环：根据引理 6 找出 H_r 中权值最大的含 t 条边的森林 T_1，$T_1=\{(i_1,j_1),\cdots,(i_t,j_t)\}$，并用 T_1 生成一组二元组，其中：$(X_1,Y_1)=(\varnothing,\{e_1\})$，$(X_2,Y_2)=(\{e_1\},\{e_2\})$，$\cdots$，$(X_t,Y_t)=(\{e_1,\cdots,e_{t-1}\},\{e_t\})$. 将二元组 $(X_1,Y_1),\cdots,(X_t,Y_t)$ 放入 L.

第 i 次循环：设第 $i-1$ 次循环后 $L=\{(X_1,Y_1),\cdots,(X_h,Y_h)\}$，首先找出基于各二元组的权值最大的含 t 条边的森林 $T(P_1),\cdots,T(P_h)$，并选出其中权值最大的森林赋给 T_i，则 T_i 为 H_r 中权值第 i 大、含 t 条边的森林. 设 T_i 是基于 (X_k,Y_k) 的森林，用 T_i 将 (X_k,Y_k) 进行划分，将所有二元组加到 L

中，并从 L 中删除(X_k,Y_k).

将算法进行到第 z 次循环，能找到 z 个权值最大含 t 条边的森林. 根据划分二元组的方法，每次循环新产生的二元组数目最多为 t 个. 划分一个二元组的时间复杂度为 $O(t)$，根据引理 6，基于二元组找一个权值最大含 t 条边的森林的时间复杂度为 $O(|E_r|\log|V_r|)$. 如果每次都将新划分的二元组分为一组，那么 L 中的元组在第 i 次循环中就有 $i-1$ 个分组，选出各组内二元组对应的权值最大含 t 条边的森林，然后组间进行比较，进而选出与 L 中二元组对应的权值最大含 t 条边的森林. 这样，第 i 次循环所需时间为 $O(t|E_r|\log|V_r|+t+i)$，那么，整个算法即枚举 z 个权值最大含 t 条边的森林算法的时间复杂度为 $O(zt|E_r|\log|V_r|+z^2)$. 证毕.

3.2.3 基于元组的局部枚举算法

前面已经介绍了将 FVS 问题转化为 FES 问题和枚举 z 个权值最大的森林的过程，则枚举图 G' 中基于元组 $r=(G,V_0,V_1,V_2,F,A,S)$ 的 z 个权值最小的 k-FVS 的算法设计如下：首先将图 $G'[V-F]$ 的 FVS 问题转化为求元组 r 对应图的 FES 问题，即构造元组 r 的转化图 H_r；设 $k_1=k-|F|$，然后枚举图 H_r 的 z 个权值最大的含有 $|E_r|-k_1$ 条边的森林，找到 z 个森林对应的图 H_r 的 z 个 FES，由各 FES 找到对应的图 $G'[V-F]$ 的 FVS，图 $G'[V-F]$ 的 z 个 FVS 分别加上 F 中点构成图 G' 基于元组 r 的 z 个权值最小的 k-FVS. 具体算法见图 3.

```
LocalEnumeration (G', r, z, k)
输入：图 G'，一个元组 r=(G,V0,V1,V2,F,A,S)，整数 z 和 k
输出：图 G' 中基于元组 r 的 z 个权值最小的 k-FVS
1. k1=k-|F|;
2. U=∅; //U 保存找到的 k-FVS
3. 构造元组 r 的图 Hr(Vr,Er);
4. UT=Enumerate-Forest (Hr,z,|Er|-k1);
5. for UT 中每个含 |Er|-k1 条边森林 T
6.   {Hr 中不属于 T 中的边所构成的集合为 Hr 的一个 FES Fe，找到与 Fe 中边所对应 的 V0∪VS∪A 中的点加入 F，再将 F 加到集合 U;}
7. 返回 U;
```

图 3 基于元组的局部枚举算法

引理 8. 给定图 G'、参数 k 和元组 $r=(G,V_0,V_1,V_2,F,A,S)$，算法 *LocalEnumeration* 可以在时间 $O(z(n^2\log n+z))$ 内枚举出图 G' 中基于元组的 z 个权值最小的 k-FVS.

证明. 算法 *LocalEnumeration* 首先将图 $G'[V-F]$ 的 FVS 问题转化为求元组对应的图 $H_r(V,E)$ 的 FES 问题；设 $k_1=k-|F|$，然后利用算法 *Enumerate-Forest* 枚举图 H_r 的 z 个权值最大的含有 $|E_r|-k_1$ 条边的森林，对某个森林 T，图 H_r 中不在 T 中的边就构成了 H_r 的一个 FES-F_e，找到与 F_e 对应的 $V_0\cup V_S\cup A$ 中的点，再把这些点加入 F，此时 F 就是图 G' 中一个 k-FVS. 完成对图 H_r 的 z 个权值最大的含有 $|E_r|-k_1$ 条边的森林的处理后，就得到了图 G' 中基于元组 r 的 z 个权值最小的 k-FVS. 根据 H_r 的构造方法，H_r 中的边数目 $|E_r|\leq n$，顶点数目 $|V_r|\leq 2n$，其中 n 为图 $G'[V-F]$ 中顶点数. 由引理 7 可知，枚举 z 个权值最大的含 $|E|-k_1$ 条边的森林的时间复杂度为

$$O(z(|E_r|-k_1)|E_r|\log|V_r|+z^2)$$
$$=O(z(n-k_1)n\log n+z^2)=O(z(n^2\log n+z)).$$

算法步 1、步 2 和步 7 的时间复杂度为 $O(1)$，步 3 构造图 H_r 以及步 5 的时间复杂度为 $O(zn)$. 因此，枚举图 G' 中基于给定元组 r 的 z 个权值最小的 k-FVS 的时间复杂度为 $O(z(n^2\log n+z))$. 证毕.

4 FVS 的固定参数枚举算法

在第 3 节已经描述了 FVS 的固定参数枚举问题的两个主要过程：元组构造和基于元组的局部枚举算法. 本节将介绍 FVS 的固定参数枚举算法并分析其时间复杂度. 算法首先构造一组 IF-划分，然后调用元组构造算法和基于元组的局部枚举算法枚举各 IF-划分的 z 个权值最小的 k-FVS，并保留其中 z 个权值最小的 k-FVS，算法具体过程如图 4 所示.

```
Enumeration-FVS(G,k,z)
输入：G=(V,E),k,z
输出：图 G 的 z 个权值最小的 k-FVS
1. 利用文献[16]中的算法求 G 一个大小为 k 的 FVS F';
2. Y=∅;  //Y 保存 G 的 z 个权值最小 k-FVS;
3. for (j=0; j≤k; j++)
4. {for F' 的每个 j 大小的子集 F1，如果 G[F'-F1] 是森林，则
5.   { G0=G[V-F1],V0=∅,V1=V-F',V2=F'-F1,
     k'=k-j,F=A=S=R=U=U1=U2=∅;
6.   R=ConstructGroup(G0,V0,V1,V2,k',F,A,S);
7.   if z≤|R|，则 z1=1;
8.   else z1=z/√|R|;
9.   for R 中的每个元组 r
10.    U1=U1∪取 LocalEnumeration(G0,r,z1,k');
11.  保留 U1 中 z 个权值最小的 k'-FVS，设所对应的 z1 个 k'-FVS 都在 U1 中的元组的集合为 R';
12.  for R' 中的每个元组 r'
13.    U2=U2∪LocalEnumeration(G0,r',z,k');
14.  将 U1∪U2 中 z 个权值最小的 k'-FVS 放入 U;
15.  将 U 中 z 个 k'- 都加上 F1 中点构成 k-FVS，放入 Y;}}
                                        //endfor
16. 输出 Y 中 z 个权值最小且互不相同的 k-FVS;
```

图 4 FVS 的固定参数枚举算法

算法步 4～步 15 为枚举一个 IF-划分的 z 个权

值最小 FVS，根据图 4 中算法，得到以下引理.

引理 9.　给定图 G 和图 G 的一个 IF-划分 (V_0, V_1, V_2)，能够在 $O(2^{\tau(k,V_0,V_1,V_2)}n^2\log n + 1.414^{\tau(k,V_0,V_1,V_2)}z(n^2\log n+z))$ 时间内枚举 IF-划分 (V_0,V_1,V_2) 的 z 个权值最小 k-FVS.

证明.　R 是 IF-划分 (V_0,V_1,V_2) 经元组构造算法得到的元组集合，设 N 是 R 中元组数目. 由引理 3，一个 k-FVS 只对应一个元组，设集合 Y 为图 G 的 z 个权值最小 k-FVS 的集合，下面分两种情况讨论.

如果 $z\leq\sqrt{N}$，设 $z_1=1$，通过基于元组的局部枚举算法，分别构造 R 中各元组对应的 1 个权值最小的 k-FVS 并放于集合 U_1，那么 U_1 中有 N 个 k-FVS，根据引理 8，此过程的时间复杂度为 $O(N(n^2\log n+1)$. 对任意一个元组 r，如果基于 r 的 k-FVS 都不在 U_1 中，那么基于 r 的 k-FVS 不可能属于 Y，则从元组集合 R 中删除该元组. 因此，R 中剩余的元组个数 $N_1=z$. 对 R 中剩下的元组，通过基于元组的局部枚举算法，分别求基于各元组的 z 个权值最小的 k-FVS 并放入集合 U_2，则 U_2 中有 $N_1z=z^2$ 个 k-FVS，根据引理 8，此过程的时间复杂度为 $O(N_1z(n^2\log n+z))=O(\sqrt{N}z(n^2\log n+z))$. $U_1\cup U_2=U_2$ 中 z 个权值最小的 k-FVS 为 IF-划分的 z 个权值最小的 k-FVS. 因此，在 $z\leq\sqrt{N}$ 的情况下，枚举 IF-划分 (V_0,V_1,V_2) 的 z 个权值最小的 k-FVS 的时间复杂度为 $O(Nn^2\log n+\sqrt{N}z(n^2\log n+z))$.

如果 $z>\sqrt{N}$，设 $z_1=z/\sqrt{N}$，通过基于元组的局部枚举算法，分别对 N 个元组构造与其对应的 z_1 个权值最小的 k-FVS 放入 U_1，那么 U_1 中有 Nz_1 个 k-FVS，根据引理 8，此过程的时间复杂度为 $O(Nz_1(n^2\log n+z_1))=O(\sqrt{N}z(n^2\log n+z_1))$. 对任意一个元组 r，若与之对应的 z_1 个权值最小的 k-FVS不全在 U_1 中，则基于元组 r 但不属于 U_1 的 k-FVS不可能属于 Y，从元组集合 R 中删除该元组. 这样 R 中剩下的元组为对应的 z_1 个权值最小的 k-FVS 都属于 U_2 的元组. 根据引理 3，一个 k-FVS 只对应一个元组，所以，R 中剩下的元组数 N_1 不超过 $\sqrt{N}$. 对这 N_1 个元组，分别找各元组对应的 z 个权值最小 k-FVS 并保存于 U_2，则 U_2 中有 N_1z 个 k-FVS，根据引理 8，此过程的时间复杂度为 $O(N_1z(n^2\log n+z))=O(\sqrt{N}z(n^2\log n+z))$. $U_1\cup U_2$ 中 z 个权值最小的 k-FVS 为 IF-划分的 z 个权值最小的 k-FVS. 因此，当 $z>\sqrt{N}$ 时，枚举 IF-划分 (V_0,V_1,V_2) 的 z 个权值最小的 k-FVS 的时间复杂度为 $O(\sqrt{N}z(n^2\log n+z))$.

综上所述，给定一个 IF-划分 (V_0,V_1,V_2)，找图 G 关于 IF-划分 (V_0,V_1,V_2) 的 z 个权值最小的 k-FVS 的时间复杂度为 $O(Nn^2\log n+\sqrt{N}z(n^2\log n+z))$. 由引理 4，$R$ 中元组个数不超过 $2^{\tau(k,V_0,V_1,V_2)}$，那么 $N\leq 2^{\tau(k,V_0,V_1,V_2)}$，$\sqrt{N}\leq 1.414^{\tau(k,V_0,V_1,V_2)}$，由此证明此引理.　证毕.

定理 1.　给定图 $G=(V,E)$，枚举图 G 中 z 个权值最小的 k-FVS 的时间复杂度为 $O(5^kn^2(\log n+k)+3^kz(n^2\log n+z))$.

证明.　给定图 $G=(V,E)$，用文献[16]中求取不带权图的 FVS 的算法，可得一个大小为 k 的 FVS F'. 图 G 的每个 k-FVS 可看作由 F' 的大小为 j 的子集 F_1 和 $V-F'$ 的一个大小为 $k-j$ 的子集 F_2 构成，其中 $0\leq j\leq k$，即找图 G 的 k-FVS 问题可以转化为找图 $G_0=G[V-F_1]$ 的不含 $F'-F_1$ 中点的 $(k-j)$-FVS，图 G_0 的每个 $(k-j)$-FVS 加上 F_1 都是图 G 的 k-FVS. 明显可知图 $G[F'-F_1]$ 和图 $G[V-F']$ 都是森林且都是图 G_0 的子图，图 G_0 的 $(k-j)$-FVS 不含 $F'-F_1$ 中的点，若设 $V_0=\varnothing$，$V_1=V-F'$，$V_2=F'-F_1$，则 (V_0,V_1,V_2) 是图 G_0 有效的 IF-划分. 对所有整数 j，$0\leq j\leq k$，穷举 F' 中所有大小为 j 且满足图 $G[F-F_1]$ 是森林的子集 F_1，用 F_1 能形成一组 *IF*-划分. 枚举各 IF-划分对应的图 G 的 z 个权值最小的 k-FVS，然后取其中 z 个权值最小的 k-FVS，即为图 G 的 z 个权值最小的 k-FVS.

由引理 9，对一个 IF-划分 (V_0,V_1,V_2) 的 z 个权值最小的 $k-j$ 大小 FVS 可以在以下时间内完成：

$$\begin{aligned}&O(2^{\tau(k-j,V_0,V_1,V_2)}n^2\log n+\\&1.414^{\tau(k-j,V_0,V_1,V_2)}z(n^2\log n+z))\\&=O(2^{(k-j)-(0-\sharp c(F'-F_1)+1)}n^2\log n+\\&\quad 1.414^{(k-j)-(0-\sharp c(F'-F_1)+1)}z(n^2\log n+z)\\&=O(4^{k-j}n^2\log n+2^{k-j}z(n^2\log n+z)),\end{aligned}$$

其中，$\sharp c(F'-F_1)\leq|F'-F_1|=k-j$.

那么对所有整数 j，$0\leq j\leq k$，穷举 F' 的所有大小为 j 的子集 F_1 并生成 IF-划分，然后再枚举各 IF-划分的 z 个权值最小、大小为 $k-j$ 的 FVS，可以在以下时间内完成：

$$\sum_{j=0}^{k}\binom{k}{j}O(4^{k-j}n^2\log n+2^{k-j}z(n^2\log n+z))$$
$$=O(5^k n^2\log n+3^k z(n^2\log n+z)).$$

IF-划分的总个数不超过 2^k 个，那么总共枚举出了 $2^k z$ 个 k-FVS，从中取 z 个权值最小的时间为 $O(2^k z)$. 根据引理 2，求取图 G 的一个 k-FVS 的时间复杂度为 $O(5^k kn^2)$. 因此，对带权图 G，枚举 z 个权值最小的 k-FVS 的时间复杂度为 $O(5^k n^2(\log n+k)+3^k z(n^2\log n+z))$. 证毕.

5 结 论

本文讨论了枚举带权无向图中 z 个权值最小的 k-FVS 的问题，提出了一个 FVS 的固定参数枚举算法，其时间复杂度为 $O(5^k n^2(\log n+k)+3^k z(n^2\log n+z))$. 算法基于分支搜索技术，将 FVS 问题转化为 FES 问题，通过枚举转化图 H_r 的 z 个权值最大的森林来枚举 z 个权值最小的 FES，从而枚举出 z 个权值最小的 k-FVS.

如果将算法中枚举图 H_r 的 z 个权值最大的森林的过程改为枚举图 H_r 的 z 个权值最大的生成森林(枚举生成森林的算法见文献[23-24])，那么各生成森林对应图 H_r 的一个极小 FES(删除 FES 中任何点将不再是图 H_r 的 FES)，通过极小 FES 找到的 FVS 将是大小不超过 k 的 FVS，这样，改变后的算法可用于枚举 z 个权值最小且大小不超过 k 的 FVS. 在本文的算法中，将分支过程中产生的 0 度点和 1 度点都转化为图 H_r 的边，然后在图 H_r 中枚举森林时一块考虑了这些点，但是当参数 k 恰好等于图 G 中含顶点数最少的 FVS 的大小时，这些 0 度点和 1 度点不可能在图 G 的 k-FVS 中；当要枚举 z 个权值最小且大小不超过 k 的 FVS 时，这些 0 度点和 1 度点也不可能在所要枚举的 FVS 中，因此，当参数 k 恰好等于图中含顶点数最少的 FVS 的大小以及当要枚举 z 个权值最小且大小不超过 k 的 FVS 时，无需将 0 度点和 1 度点都转化为 H_r 的边，这样 H_r 的规模将减小很多，枚举的时间也将减小. 当参数 k 接近于图中含顶点数最少的 FVS 的大小 d 时，也可不将 0 度点和 1 度点转化为图 H_r 的边，可参照文献[21]中枚举 k-Vertex Cover 时处理 0 度点和 1 度点的方法对这些点单独处理，但是这样对每个整数 j，$d\leq j\leq k$，需枚举图 H_r 的 z 个权值最小的 j-FES，即枚举的森林个数将相对增多.

FVS 问题是经典的 NP-完全问题，有很多重要应用，人们对此问题进行了大量的研究. 在参数计算理论领域，对无向图中参数化 FVS 问题有一系列 FPT 算法，而且对无向图 FVS 的核心化也进行了研究[25-27]，最近得出 $O(k^2)$ 的核[27]. 目前已经证明不带权有向图 FVS 是 FPT 的，但是有向图有无多项式的核以及带权有向图的 FVS 有无 FPT 算法仍然是开放性课题，有待于进一步研究.

参 考 文 献

[1] Karp R. Complexity of computer computations: Reducibility among combinatorial problems. New York: Plenum Press, 1972

[2] Festa P, Pardalos P M, Mauricio and Resende G C. Feedback set problems//Handbook of Combinatorial Optimization. Kluwer: Kluwer Academic Publishers, 2000: 209-258

[3] Bar-Yehuda R, Geiger D. Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and bayesian inference. SIAM Journal on Computing, 1998, 27(4): 942-959

[4] Bafna V, Berman P, and Fujito T. A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM Journal on Discrete Mathematics, 1999, 12(3): 289-297

[5] Bafna V, Berman P, and Fujito T. Constant ratio approximations of feedback vertex sets in weighted undirected graphs//Staples J, Eades P, Katoh N, Moffat A eds. ISAAC 1995. Cairns, Australia. LNCS 1004. Springer-Verlag, 1995: 142-151

[6] Razgon I. Exact computation of maximum induced forest//Arge L and Freivalds R eds. SWAT 2006. Riga, Latvia. LNCS 4059. Springer-Verlag, 2006: 160-171

[7] Fomin F, Gaspers S, and Pyatkin A. Finding a minimum feedback vertex set in time $O(1.7548^n)$//Bodlaender H L, Langston M A eds. IWPEC 2006. Zürich, Switzerland. LNCS 4169. Springer-Verlag, 2006: 184-191

[8] Downey R, Fellows M. Complexity Theory: Current Research: Fixed Parameter Tractability and Completeness. New York, USA: Cambridge University Press, 1992

[9] Bodlaender H L. On disjoint cycle. International Journal of Foundations of Computer Science, 1994, 5(1): 59-68

[10] Downey R, Fellows M. Parameterized Complexity. Springer-Verlag, 1999

[11] Raman V, Saurabh S, and Subramanian C R. Faster fixed parameter tractable algorithms for undirected feedback vertex set//Bose P, Morin P eds. ISAAC 2002. Vancouver(BC), Canada. LNCS 2518. Springer-Verlag, 2002: 241-248

[12] Kanj I, Pelsmajer M, Schaefer M. Parameterized algorithms for feedback vertex set//Downey R, Fellows M, Dehne F eds. IWPEC 2004. Bergen, Norway. LNCS 3162. Springer-Verlag, 2004: 235-247

[13] Raman V, Saurabh S, Subramanian C R. Faster fixed parameter tractable algorithms for finding feedback vertex sets. ACM Transactions on Algorithms, 2006, 2(3): 403-415

[14] Guo J, Gramm J, Hüffner F, Niedermeier R, Wernicke S. Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. Journal of Computer and System Sciences, 2006, 72(8): 1386-1396

[15] Dehne F, Fellows M, Langston M, Rosamond F, Stevens K. An $o(2^{o(k)}n^3)$ fpt algorithm for the undirected feedback vertex set problem. Theory Computing Systems, 2007, 41(3): 479-492

[16] Chen J, Fomin F, Liu Y, Lu S, Villanger Y. Improved algorithms for the feedback vertex set problems//Dehne F, Sack J, Zeh N eds. WADS 2007. Halifax, Canada. LNCS 4619. Springer-Verlag, 2007: 422-433

[17] Chen J, Liu Y, Lu S, O'Sullivan B, Razgon I. A fixed-parameter algorithm for the directed feedback vertex set problem//Proceedings of the STOC 2008. Victoria (BC), Canada, 2008: 177-186

[18] Becker A, Bar-Yehuda R, Geiger D. Randomized algorithms for the loop cutset problem. Journal of Artificial Intelligence Research, 2000, 12: 219-234

[19] Flum J, Grohe M. The parameterized complexity of counting problems. SIAM Journal on Computing, 2002, 33(4): 892-922

[20] Fernau F. On parameterized enumeration//Ibarra O H, Zhang L eds. COCOON 2002. Singapore. LNCS 2387. Springer-Verlag, 2002: 564-573

[21] Chen J, Kanj I, Meng J, Xia G, Zhang F. On the effective enumerability of NP problems//Bodlaender H L, Langston M A eds. IWPEC 2006. Zürich, Switzerland. LNCS 4169. Springer-Verlag, 2006: 215-226

[22] Cormen T, Leiserson C, Rivest R, Stein C. Introduction to Algorithms. 2nd Edition. Boston, MA: McGraw-Hill Book Company, 2001

[23] Kapoor S, Ramesh H. Algorithms for enumerating all spanning trees of undirected and weighted graphs. SIAM Journal on Computing, 1995, 24(2): 247-265

[24] Sorensen K, Janssens G. An algorithm to generate all spanning trees of a graph in order of increasing cost. Pesquisa Operacional, 2005, 25(2): 219-229

[25] Burrage K, Estivill-Castro V, Fellows M, Langston M, Mac S, Rosamond F. The undirected feedback vertex set problem has a poly(k) kernel//Bodlaender H L, Langston M A eds. IWPEC 2006. Zürich, Switzerland. LNCS 4169. Springer-Verlag, 2006: 192-202

[26] Bodlaender H L. A cubic kernel for feedback vertex set//Thomas W, Weil P eds. STACS 2007. Aachen, Germany. LNCS 4393. Springer-Verlag, 2007: 320-331

[27] Thomassé S. A quadratic kernel for feedback vertex set//SODA 2009, New York, 2008: 115-119

WANG Jian-Xin, born in 1960, Ph. D., professor. His research interests include computer algorithm, network optimization, computational biology.

JIANG Guo-Hong, born in 1985, M. S.. Her research interests include parameterized computation, computer theory.

CHEN Jian-Er, born in 1954, Ph. D., professor. His research interests include computational biology, computer theory, computational complexity and optimization.

Background

This work is supported by the National Basic Research Program (973 Program) of China (2008CB317107) which focuses on the design of parameterized algorithm in various information processing fields, the National Natural Science Foundation of China under Grants (60773111, 60873265) which focus on the development of parameterized algorithm technique and the application of parameterized algorithm in network, and the Program for Changjiang Scholars and Innovative Research Team in University of China under Grant (IRT0661).

The theory of parameterized computation and complexity is a recently developed subarea in theoretical computer science. The theory is amied at practically solving a large number of computational problems that are theoretically intractable. The theory is based on the observation that many intractable computational problems in practice are associated with a parameter that varies within a small or moderate range. Therefore, by taking the advantages of the small parameters, many theoretically intractable problems can be solved effectively and practically. On the other hand, the theory of pa-

rameterized computation and complexity has also offered powerful techniques that explain why certain theoretically tractable problems cannot be solved effectively and practically. The theory of parameterized computation and complexity has found wide applications in areas such as database systems, programming languages, networks, VLSI design, parallel and distributed computing, computational biology, and robotics.

Feedback set (including FVS and FAS) problems are classical NP-hard problems. There are numerous applications of the feedback set problem in areas such as circuit testing, deadlock resolution, analyzing manufacturing processes and computational biology. Many different algorithmic approaches were tried on these problems, including approximation algorithms, linear programming, local search, polyhedral combinatorics and probabilistic algorithms. The feedback set problem has also been extensively studied from the parameterized view. It is known that the parameterized FVS problems both in directed and undirected unweighted graphs are FPT problems. For FVS problem, there is an algorithm to find all the minimal-FVS, but the running time of it is practically intractable. In this paper, a fixed-parameter enumeration algorithm with running time $O(5^k n^2(\log n + k) + 3^k z(n^2 \log n + z))$ is given for the weighted FVS problem in undirected graphs. The algorithm is the first fixed-parameter enumeration algorithm for the FVS problem and the running time is next to that of the best previous algorithm for finding a FVS of size at most k. The algorithm can also be used to find the z minimum-weight minimal-FVS of at most k by some transformation.