

# Branching algorithms

Serge Gaspers

## Contents

1	Branching algorithms	1
2	Running time analysis	1
3	Feedback Vertex Set	2
4	Maximum Leaf Spanning Tree	4
5	Further Reading	6

## 1 Branching algorithms

### Branching Algorithm

- *Selection*: Select a local configuration of the problem instance
- *Recursion*: Recursively solve subinstances
- *Combination*: Compute a solution of the instance based on the solutions of the subinstances
- *Halting* rule: 0 recursive calls
- *Simplification* rule: 1 recursive call
- *Branching* rule:  $\geq 2$  recursive calls

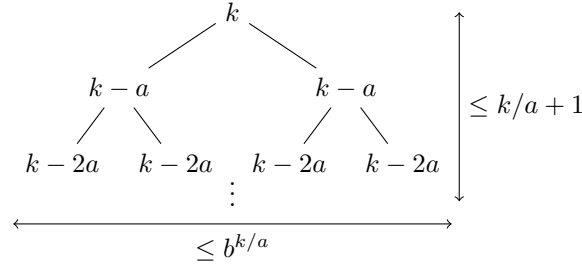
### Example: Our first Vertex Cover algorithm

```
Algorithm vc1( $G, k$ );  
1 if  $E = \emptyset$  then                                // all edges are covered  
2   | return Yes  
3 else if  $k \leq 0$  then                               // we cannot select any vertex  
4   | return No  
5 else  
6   | Select an edge  $uv \in E$ ;  
7   | return  $\text{vc1}(G - u, k - 1) \vee \text{vc1}(G - v, k - 1)$ 
```

## 2 Running time analysis

### Search trees

**Recall:** A *search tree* models the recursive calls of an algorithm. For a  $b$ -way branching where the parameter  $k$  decreases by  $a$  at each recursive call, the number of nodes is at most  $b^{k/a} \cdot (k/a + 1)$ .



If  $k/a$  and  $b$  are upper bounded by a function of  $k$ , and the time spent at each node is FPT (typically, polynomial), then we get an FPT running time.

### 3 Feedback Vertex Set

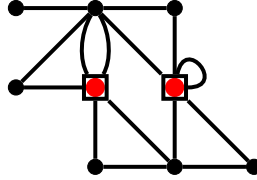
A *feedback vertex set* of a multigraph  $G = (V, E)$  is a set of vertices  $S \subseteq V$  such that  $G - S$  is acyclic.

#### FEEDBACK VERTEX SET

Input: Multigraph  $G = (V, E)$ , integer  $k$

Parameter:  $k$

Question: Does  $G$  have a feedback vertex set of size at most  $k$ ?



#### Simplification Rules

We apply the first *applicable*<sup>1</sup> simplification/halting rule.

##### (Finished)

If  $G$  is acyclic and  $k \geq 0$ , then return YES.

##### (Budget-exceeded)

If  $k < 0$ , then return NO.

##### (Loop)

If  $G$  has a loop  $vv \in E$ , then set  $G \leftarrow G - v$  and  $k \leftarrow k - 1$ .

##### (Multiedge)

If  $E$  contains an edge  $uv$  more than twice, remove all but two copies of  $uv$ .

##### (Degree-1)

If  $\exists v \in V$  with  $d_G(v) \leq 1$ , then set  $G \leftarrow G - v$ .

##### (Degree-2)

If  $\exists v \in V$  with  $d_G(v) = 2$ , then denote  $vu, vw$  its two incident edges and set  $G \leftarrow G' = (V \setminus \{v\}, (E \setminus \{vu, vw\}) \cup \{uw\})$ .

**Lemma 1.** (*Degree-2*) is sound.

<sup>1</sup>A rule is *applicable* if it modifies the instance.

*Proof.* Suppose  $S$  is a feedback vertex set of  $G$  of size at most  $k$ . Let

$$S' = \begin{cases} S & \text{if } v \notin S \\ (S \setminus \{v\}) \cup \{u\} & \text{if } v \in S. \end{cases}$$

Now,  $|S'| \leq k$  and  $S'$  is a feedback vertex set of  $G'$  since every cycle in  $G'$  corresponds to a cycle in  $G$ , with, possibly, the edge  $uw$  replaced by the walk  $(u, v, w)$ .

Suppose  $S'$  is a feedback vertex set of  $G'$  of size at most  $k$ . Then,  $S'$  is also a feedback vertex set of  $G$ .  $\square$

### Remaining issues

- A select–discard branching decreases  $k$  in only one branch
- One could branch on all the vertices of a cycle, but the length of a shortest cycle might not be bounded by any function of  $k$

Idea:

- An acyclic graph has average degree  $< 2$
- After applying simplification rules,  $G$  has average degree  $\geq 3$
- The selected feedback vertex set needs to be incident to many edges
- Does a feedback vertex set of size at most  $k$  contain at least one vertex among the  $f(k)$  vertices of highest degree?

### The fvs needs to be incident to many edges

**Lemma 2.** *If  $S$  is a feedback vertex set of  $G = (V, E)$ , then*

$$\sum_{v \in S} (d_G(v) - 1) \geq |E| - |V| + 1$$

*Proof.* Since  $F = G - S$  is acyclic,  $|E(F)| \leq |V| - |S| - 1$ . Since every edge in  $E \setminus E(F)$  is incident with a vertex of  $S$ , we have

$$\begin{aligned} |E| &= |E| - |E(F)| + |E(F)| \\ &\leq \left( \sum_{v \in S} d_G(v) \right) + (|V| - |S| - 1) \\ &= \left( \sum_{v \in S} (d_G(v) - 1) \right) + |V| - 1. \end{aligned}$$

$\square$

### The fvs needs to contain a high-degree vertex

**Lemma 3.** *Let  $G$  be a graph with minimum degree at least 3 and let  $H$  denote a set of  $3k$  vertices of highest degree in  $G$ . Every feedback vertex set of  $G$  of size at most  $k$  contains at least one vertex of  $H$ .*

*Proof.* Suppose not. Let  $S$  be a feedback vertex set with  $|S| \leq k$  and  $S \cap H = \emptyset$ . Then,

$$\begin{aligned} 2|E| - |V| &= \sum_{v \in V} (d_G(v) - 1) \\ &= \sum_{v \in H} (d_G(v) - 1) + \sum_{v \in V \setminus H} (d_G(v) - 1) \\ &\geq 3 \cdot \left( \sum_{v \in S} (d_G(v) - 1) \right) + \sum_{v \in S} (d_G(v) - 1) \\ &\geq 4 \cdot (|E| - |V| + 1) \\ \Leftrightarrow \quad 3|V| &\geq 2|E| + 4. \end{aligned}$$

But this contradicts the fact that every vertex of  $G$  has degree at least 3.  $\square$

## Algorithm for Feedback Vertex Set

**Theorem 4.** FEEDBACK VERTEX SET can be solved in  $O^*((3k)^k)$  time.

*Proof (sketch).* • Exhaustively apply the simplification rules.

- The branching rule computes  $H$  of size  $3k$ , and branches into subproblems  $(G - v, k - 1)$  for each  $v \in H$ . □

Current best:  $O^*(3.460^k)$  deterministic (Iwata and Kobayashi, 2019),  $O^*(2.7^k)$  time randomized (Li and Nederlof, 2019)

## 4 Maximum Leaf Spanning Tree

A *leaf* of a tree is a vertex with degree 1. A *spanning tree* in a graph  $G = (V, E)$  is a subgraph of  $G$  that is a tree and has  $|V|$  vertices.

### MAXIMUM LEAF SPANNING TREE

Input: connected graph  $G$ , integer  $k$   
 Parameter:  $k$   
 Question: Does  $G$  have a spanning tree with at least  $k$  leaves?

### Property

A *k-leaf tree* in  $G$  is a subgraph of  $G$  that is a tree with at least  $k$  leaves. A *k-leaf spanning tree* in  $G$  is a spanning tree in  $G$  with at least  $k$  leaves.

**Lemma 5.** Let  $G = (V, E)$  be a connected graph.  $G$  has a *k-leaf tree*  $\Leftrightarrow$   $G$  has a *k-leaf spanning tree*.

*Proof.* ( $\Leftarrow$ ): trivial

( $\Rightarrow$ ): Let  $T$  be a *k-leaf tree* in  $G$ . By induction on  $x := |V| - |V(T)|$ , we will show that  $T$  can be extended to a *k-leaf spanning tree* in  $G$ .

Base case:  $x = 0$  ✓.

Induction:  $x > 0$ , and assume the claim is true for all  $x' < x$ . Choose  $uv \in E$  such that  $u \in V(T)$  and  $v \notin V(T)$ . Since  $T' := (V(T) \cup \{v\}, E(T) \cup \{uv\})$  has  $\geq k$  leaves and  $< x$  external vertices, it can be extended to a *k-leaf spanning tree* in  $G$  by the induction hypothesis. □

### Strategy

- The branching algorithm will check whether  $G$  has a *k-leaf tree*.
- A tree with  $\geq 3$  vertices has at least one *internal* (= non-leaf) vertex.
- “Guess” an internal vertex  $r$ , i.e., do a  $|V|$ -way branching fixing an initial internal vertex  $r$ .
- In any branch, the algorithm has computed
  - $T$  – a tree in  $G$
  - $I$  – the internal vertices of  $T$ , with  $r \in I$
  - $B$  – a subset of the leaves of  $T$  where  $T$  may be extended: the boundary set
  - $L$  – the remaining leaves of  $T$
  - $X$  – the external vertices  $V \setminus V(T)$
- The question is whether  $T$  can be extended to a *k-leaf tree* where all the vertices in  $L$  are leaves.

### Simplification Rules

Apply the first applicable simplification rule:

#### (Halt-Yes)

If  $|L| + |B| \geq k$ , then return YES.

#### (Halt-No)

If  $|B| = 0$ , then return No.

#### (Non-extendable)

If  $\exists v \in B$  with  $N_G(v) \cap X = \emptyset$ , then move  $v$  to  $L$ .

### Branching Lemma

**Lemma 6** (Branching Lemma). *Suppose  $u \in B$  and there exists a  $k$ -leaf tree  $T'$  extending  $T$  where  $u$  is an internal vertex. Then, there exists a  $k$ -leaf tree  $T''$  extending  $(V(T) \cup N_G(u), E(T) \cup \{uv : v \in N_G(u) \cap X\})$ .*

*Proof.* Start from  $T'' \leftarrow T'$  and perform the following operation for each  $v \in N_G(u) \cap X$ .

If  $v \notin V(T')$ , then add the vertex  $v$  and the edge  $uv$ . Otherwise, add the edge  $uv$ , creating a cycle  $C$  in  $T$  and remove the other edge of  $C$  incident to  $v$ . This does not decrease the number of leaves, since it only increases the number of edges incident to  $u$ , and  $u$  was already internal.  $\square$

### Follow Path Lemma

**Lemma 7** (Follow Path Lemma). *Suppose  $u \in B$  and  $|N_G(u) \cap X| = 1$ . Let  $N_G(u) \cap X = \{v\}$ . If there exists a  $k$ -leaf tree extending  $T$  where  $u$  is internal, but no  $k$ -leaf tree extending  $T$  where  $u$  is a leaf, then there exists a  $k$ -leaf tree extending  $T$  where both  $u$  and  $v$  are internal.*

*Proof.* Suppose not, and let  $T'$  be a  $k$ -leaf tree extending  $T$  where  $u$  is internal and  $v$  is a leaf. But then,  $T - v$  is a  $k$ -leaf tree as well.  $\square$

### Algorithm

- Apply halting & simplification rules
- Select  $u \in B$ . Branch into
  - $u \in L$
  - $u \in I$ . In this case, add  $X \cap N_G(u)$  to  $B$  (Branching Lemma).
    - \* In the special case where  $|X \cap N_G(u)| = 1$ , denote  $\{v\} = X \cap N_G(u)$ , make  $v$  internal, and add  $N_G(v) \cap X$  to  $B$ , continuing the same way until reaching a vertex with at least 2 neighbors in  $X$  (Follow Path Lemma).
    - \* In the special case where  $|X \cap N_G(u)| = 0$ , return No.
- In one branch, a vertex moves from  $B$  to  $L$ ; in the other branch,  $|B|$  increases by at least 1.

### Running time analysis

- Consider the “measure”  $\mu := 2k - 2|L| - |B|$
- We have that  $0 \leq \mu \leq 2k$
- Branch where  $u \in L$ :
  - $|B|$  decreases by 1,  $|L|$  increases by 1
  - $\mu$  decreases by 1
- Branch where  $u \in I$ .
  - $u$  moves from  $B$  to  $I$
  - $\geq 2$  vertices move from  $X$  to  $B$
  - $\mu$  decreases by at least 1
- Binary search tree of height  $\leq \mu \leq 2k$

## Result for Maximum Leaf Spanning Tree

**Theorem 8** ((Kneis, Langer, and Rossmanith, 2011)). MAXIMUM LEAF SPANNING TREE can be solved in  $O^*(4^k)$  time.

Current best:  $O(3.188^k)$  (Zehavi, 2018)

## 5 Further Reading

- Chapter 3, *Bounded Search Trees* in (Cygan et al., 2015)
- Chapter 3, *Bounded Search Trees* in (Downey and Fellows, 2013)
- Chapter 8, *Depth-Bounded Search Trees* in (Niedermeier, 2006)

## References

- Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshantov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh (2015). *Parameterized Algorithms*. Springer. DOI: 10.1007/978-3-319-21275-3.
- Rodney G. Downey and Michael R. Fellows (2013). *Fundamentals of Parameterized Complexity*. Springer. DOI: 10.1007/978-1-4471-5559-1.
- Yoichi Iwata and Yusuke Kobayashi (2019). *Improved Analysis of Highest-Degree Branching for Feedback Vertex Set*. Tech. rep. abs/1905.12233. arXiv CoRR. URL: <http://arxiv.org/abs/1905.12233>.
- Joachim Kneis, Alexander Langer, and Peter Rossmanith (2011). “A New Algorithm for Finding Trees with Many Leaves”. In: *Algorithmica* 61.4, pp. 882–897.
- Jason Li and Jesper Nederlof (2019). *Detecting Feedback Vertex Sets of Size  $k$  in  $O^*(2.7^k)$  Time*. Tech. rep. abs/1906.12298. arXiv CoRR. URL: <http://arxiv.org/abs/1906.12298>.
- Rolf Niedermeier (2006). *Invitation to Fixed Parameter Algorithms*. Oxford University Press. DOI: 10.1093/ACPROF:OSO/9780198566076.001.0001.
- Meirav Zehavi (2018). “The  $k$ -leaf spanning tree problem admits a klam value of 39”. In: *Eur. J. Comb.* 68, pp. 175–203.