Outline

1 Branching algorithms

2 Running time analysis

3 Feedback Vertex Set

4 Maximum Leaf Spanning Tree

5 Further Reading
Outline

1. Branching algorithms
2. Running time analysis
3. Feedback Vertex Set
4. Maximum Leaf Spanning Tree
5. Further Reading
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
Algorithm \text{vc1}(G, k);
Outline

1. Branching algorithms
2. Running time analysis
3. Feedback Vertex Set
4. Maximum Leaf Spanning Tree
5. Further Reading
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.
1 Branching algorithms
2 Running time analysis
3 Feedback Vertex Set
4 Maximum Leaf Spanning Tree
5 Further Reading
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\(^1\) simplification rule.

\[(\text{Finished})\]

If \( G \) is acyclic and \( k \geq 0 \), then return \textbf{Yes}.

\[(\text{Budget-exceeded})\]

If \( k < 0 \), then return \textbf{No}.

\(^1\)A simplification rule is \textit{applicable} if it modifies the instance.
We apply the first applicable simplification 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 $k \leftarrow k - 1$.

---

1A simplification rule is applicable if it modifies the instance.
We apply the first applicable simplification 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 $uv$.

---

1A simplification rule is applicable if it modifies the instance.
Simplification Rules II

(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 $N_G(v) = \{u, w\}$ and set $G \leftarrow G' = (V \setminus \{v\}, (E \setminus \{vu, vw\}) \cup \{uw\})$. 

Lemma 1 (Degree-2) is sound.

Proof.
Suppose $S$ is a feedback vertex set of $G$ of size at most $k$. Let $S' = S$ if $v \not\in S$, $S' = S \setminus \{v\} \cup \{u\}$ if $v \in S$.

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 path $(u, v, w)$. 

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

S. Gaspers (UNSW)
(Degree-2)

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

Lemma 1

(Degree-2) is sound.

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 path $(u, v, w)$.

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

\[ \square \]
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?
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?
Lemma 2

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

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

$$|E| = |E| - |E(F)| + |E(F)|$$

$$\leq \left( \sum_{v \in S} d_G(v) \right) + (|V| - |S| - 1)$$

$$= \left( \sum_{v \in S} (d_G(v) - 1) \right) + |V| - 1.$$
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,

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

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

$$2|E| - |V| = \sum_{v \in V} (d_G(v) - 1)$$

$$= \sum_{v \in H} (d_G(v) - 1) + \sum_{v \in V \setminus H} (d_G(v) - 1)$$

$$\geq 3 \cdot \left( \sum_{v \in S} (d_G(v) - 1) \right) + \sum_{v \in S} (d_G(v) - 1)$$

$$\geq 4 \cdot (|E| - |V| + 1)$$

$$\Leftrightarrow 3|V| \geq 2|E| + 4.$$ 

But this contradicts the fact that every vertex of $G$ has degree at least 3.
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 [IK19],
- $O^*(2.7^k)$ time randomized [LN19]
Outline

1 Branching algorithms
2 Running time analysis
3 Feedback Vertex Set
4 Maximum Leaf Spanning Tree
5 Further Reading
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?
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 $\iff$ $G$ 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: $x = 0 \checkmark$.

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. \qed
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 $r$. 
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 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.
Apply the first applicable simplification rule:

(Halt-Yes)
If $|L| + |B| \geq k$, then return \textbf{Yes}.

(Halt-No)
If $|B| = 0$, then return \textbf{No}.

(Non-extendable)
If $\exists v \in B$ with $N_G(v) \cap X = \emptyset$, then move $v$ to $L$. 
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 $v \not\in V(T')$, 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.
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. \qed
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.
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.
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.
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 \( \text{No} \).

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 \(\leq \mu \leq 2k\)
Theorem 8 ([KLR11])

**Maximum Leaf Spanning Tree** can be solved in $O^*(4^k)$ time.

Current best: $O(3.188^k)$ [Zeh18]
1. Branching algorithms
2. Running time analysis
3. Feedback Vertex Set
4. Maximum Leaf Spanning Tree
5. Further Reading
Further Reading

- Chapter 3, *Bounded Search Trees* in [Cyg+15]
- Chapter 3, *Bounded Search Trees* in [DF13]
- Chapter 8, *Depth-Bounded Search Trees* in [Nie06]
References


References II
