1b. Kernelization

COMP6741: Parameterized and Exact Computation

Serge Gaspers

Semester 2, 2018

Contents

1 Vertex Cover
   1.1 Simplification rules ................................................................. 2
   1.2 Preprocessing algorithm ................................................................. 3

2 Kernelization algorithms ................................................................. 3

3 A smaller kernel for Vertex Cover ..................................................... 4

4 More on Crown Decompositions .......................................................... 6

5 Kernels and Fixed-parameter tractability ............................................. 6

6 Further Reading ................................................................................. 7

1 Vertex Cover

A vertex cover of a graph $G = (V, E)$ is a subset of vertices $S \subseteq V$ such that for each edge $\{u, v\} \in E$, we have $u \in S$ or $v \in S$.

Exercise 1

![Graph Image]
Is this a Yes-instance for Vertex Cover? (Is there $S \subseteq V$ with $|S| \leq 4$, such that $\forall uv \in E, u \in S$ or $v \in S$?)

Exercise 2

1.1 Simplification rules

(Degree-0)
If $\exists v \in V$ such that $d_G(v) = 0$, then set $G \leftarrow G - v$.

Proving correctness. A simplification rule is sound if for every instance, it produces an equivalent instance. Two instances $I, I'$ are equivalent if they are both Yes-instances or they are both No-instances.

Lemma 1. (Degree-0) is sound.

Proof. First, suppose $(G - v, k)$ is a Yes-instance. Let $S$ be a vertex cover for $G - v$ of size at most $k$. Then, $S$ is also a vertex cover for $G$ since no edge of $G$ is incident to $v$. Thus, $(G, k)$ is a Yes-instance.

Now, suppose $(G - v, k)$ is a No-instance. For the sake of contradiction, assume $(G, k)$ is a Yes-instance. Let $S$ be a vertex cover for $G$ of size at most $k$. But then, $S \setminus \{v\}$ is a vertex cover of $G$ of size at most $k$, a contradiction.

(Degree-1)
If $\exists v \in V$ such that $d_G(v) = 1$, then set $G \leftarrow G - N_G[v]$ and $k \leftarrow k - 1$.

Lemma 2. (Degree-1) is sound.

Proof. Let $u$ be the neighbor of $v$ in $G$. Thus, $N_G[v] = \{u, v\}$.

If $S$ is a vertex cover of $G$ of size at most $k$, then $S \setminus \{u, v\}$ is a vertex cover of $G - N_G[v]$ of size at most $k - 1$, because $u \in S$ or $v \in S$. If $S'$ is a vertex cover of $G - N_G[v]$ of size at most $k - 1$, then $S' \cup \{u\}$ is a vertex cover of $G$ of size at most $k$, since all edges that are in $G$ but not in $G - N_G[v]$ are incident to $v$.

(Large Degree)
If $\exists v \in V$ such that $d_G(v) > k$, then set $G \leftarrow G - v$ and $k \leftarrow k - 1$.

Lemma 3. (Large Degree) is sound.

Proof. Let $S$ be a vertex cover of $G$ of size at most $k$. If $v \notin S$, then $N_G(v) \subseteq S$, contradicting that $|S| \leq k$.

(Number of Edges)
If $d_G(v) \leq k$ for each $v \in V$ and $|E| > k^2$ then return No
Lemma 4. (Number of Edges) is sound.

Proof. Assume \( d_G(v) \leq k \) for each \( v \in V \) and \( |E| > k^2 \). Suppose \( S \subseteq V \), \( |S| \leq k \), is a vertex cover of \( G \). We have that \( S \) covers at most \( k^2 \) edges. However, \( |E| \geq k^2 + 1 \). Thus, \( S \) is not a vertex cover of \( G \). \( \square \)

1.2 Preprocessing algorithm

\textbf{VC-preprocess}

\textbf{Input}: A graph \( G \) and an integer \( k \).

\textbf{Output}: A graph \( G' \) and an integer \( k' \) such that \( G \) has a vertex cover of size at most \( k \) if and only if \( G' \) has a vertex cover of size at most \( k' \).

\[ G' \leftarrow G \]
\[ k' \leftarrow k \]
\[ \text{repeat} \]
\[ \text{execute simplification rules (Degree-0), (Degree-1), (Large Degree), and (Number of Edges) for } (G', k') \]
\[ \text{until no simplification rule applies} \]
\[ \text{return } (G', k') \]

Effectiveness of preprocessing algorithms

- How effective is VC-preprocess?
- We would like to study preprocessing algorithms mathematically and quantify their effectiveness.

First try

- Say that a preprocessing algorithm for a problem \( \Pi \) is \textit{nice} if it runs in polynomial time and for each instance for \( \Pi \), it returns an instance for \( \Pi \) that is strictly smaller.
- \( \rightarrow \) executing it a linear number of times reduces the instance to a single bit
- \( \rightarrow \) such an algorithm would solve \( \Pi \) in polynomial time
- For NP-hard problems this is not possible unless \( P = \text{NP} \)
- We need a different measure of effectiveness

Measuring the effectiveness of preprocessing algorithms

- We will measure the effectiveness in terms of the \textit{parameter}
- How large is the resulting instance in terms of the parameter?

Effectiveness of \textbf{VC-preprocess}

Lemma 5. \textit{For any instance } \((G, k)\text{ for Vertex Cover, VC-preprocess produces an equivalent instance } (G', k')\text{ of size } O(k^2)\).

Proof. Since all simplification rules are sound, \((G = (V, E), k)\) and \((G' = (V', E'), k')\) are equivalent. By (Number of Edges), \( |E'| \leq (k')^2 \leq k^2 \). By (Degree-0) and (Degree-1), each vertex in \( V' \) has degree at least 2 in \( G' \). Since \( \sum_{v \in V'} d_{G'}(v) = 2 |E'| \leq 2k^2 \), this implies that \( |V'| \leq k^2 \). Thus, \( |V'| + |E'| \leq O(k^2) \). \( \square \)

2 Kernelization algorithms

Kernelization: definition

Definition 6. A \textit{kernelization} for a parameterized problem \( \Pi \) is a \textbf{polynomial time} algorithm, which, for any instance \( I \) of \( \Pi \) with parameter \( k \), produces an \textit{equivalent} instance \( I' \) of \( \Pi \) with parameter \( k' \) such that \( |I'| \leq f(k) \) and \( k' \leq f(k) \) for a computable function \( f \). We refer to the function \( f \) as the \textit{size} of the kernel.

Note: We do not formally require that \( k' \leq k \), but this will be the case for many kernelizations.
**VC-preprocess** is a quadratic kernelization

**Theorem 7.** *VC-preprocess* is an $O(k^2)$ kernelization for Vertex Cover.

Can we obtain a kernel with fewer vertices?

## 3 A smaller kernel for Vertex Cover

### Integer Linear Program for Vertex Cover

The Vertex Cover problem can be written as an Integer Linear Program (ILP). For an instance $(G = (V, E), k)$ for Vertex Cover with $V = \{v_1, \ldots, v_n\}$, create a variable $x_i$ for each vertex $v_i$, $1 \leq i \leq n$. Let $X = \{x_1, \ldots, x_n\}$.

**ILP** $\text{VC}(G) = $

\[
\begin{align*}
\text{Minimize} & \sum_{i=1}^{n} x_i \\
\text{subject to} & \quad x_i + x_j \geq 1 \quad \text{for each } \{v_i, v_j\} \in E \\
& \quad x_i \in \{0, 1\} \quad \text{for each } i \in \{1, \ldots, n\}
\end{align*}
\]

Then, $(G, k)$ is a Yes-instance iff the objective value of ILP$_{\text{VC}}(G)$ is at most $k$.

### LP relaxation for Vertex Cover

**LP** $\text{VC}(G) = $

\[
\begin{align*}
\text{Minimize} & \sum_{i=1}^{n} x_i \\
\text{subject to} & \quad x_i + x_j \geq 1 \quad \text{for each } \{v_i, v_j\} \in E \\
& \quad x_i \geq 0 \quad \text{for each } i \in \{1, \ldots, n\}
\end{align*}
\]

**Note:** the value of an optimal solution for the Linear Program LP$_{\text{VC}}(G)$ is at most the value of an optimal solution for ILP$_{\text{VC}}(G)$

### Properties of LP optimal solution

- Let $\alpha : X \rightarrow \mathbb{R}_{\geq 0}$ be an optimal solution for LP$_{\text{VC}}(G)$. Let

\[
\begin{align*}
V_- &= \{v_i : \alpha(x_i) < 1/2\} \\
V_{1/2} &= \{v_i : \alpha(x_i) = 1/2\} \\
V_+ &= \{v_i : \alpha(x_i) > 1/2\}
\end{align*}
\]

**Lemma 8.** For each $i, 1 \leq i \leq n$, we have that $\alpha(x_i) \leq 1$.

**Lemma 9.** $V_-$ is an independent set.

**Lemma 10.** $N_G(V_-) = V_+$.

**Lemma 11.** For each $S \subseteq V_+$ we have that $|S| \leq |N_G(S) \cap V_-|$.

**Proof.** For the sake of contradiction, suppose there is a set $S \subseteq V_+$ such that $|S| > |N_G(S) \cap V_-|$. Let $\epsilon = \min_{v_i \in S} \{\alpha(x_i) - 1/2\}$ and $\alpha' : X \rightarrow \mathbb{R}_{\geq 0}$ s.t.

\[
\alpha'(x_i) = \begin{cases} 
\alpha(x_i) & \text{if } v_i \notin S \cup (N_G(S) \cap V_-) \\
\alpha(x_i) - \epsilon & \text{if } v_i \in S \\
\alpha(x_i) + \epsilon & \text{if } v_i \in N_G(S) \cap V_-
\end{cases}
\]

Note that $\alpha'$ is an improved solution for LP$_{\text{VC}}(G)$, contradicting that $\alpha$ is optimal. \qed
Theorem 12 (Hall’s marriage theorem). A bipartite graph $G = (V \uplus U, E)$ has a matching saturating $S \subseteq V$ if and only if for every subset $W \subseteq S$ we have $|W| \leq |N_G(W)|$.

Consider the bipartite graph $B = (V_-, V_+, \{\{u, v\} \in E : u \in V_-, v \in V_+\})$.

Lemma 13. There exists a matching $M$ in $B$ of size $|V_+|$.

Proof. The lemma follows from the previous lemma and Hall’s marriage theorem.

Crown Decomposition: Definition

Definition 14 (Crown Decomposition). A crown decomposition $(C, H, B)$ of a graph $G = (V, E)$ is a partition of $V$ into sets $C, H, \text{ and } B$ such that

- the crown $C$ is a non-empty independent set,
- the head $H = N_G(C)$,
- the body $B = V \setminus (C \cup H)$, and
- there is a matching of size $|H|$ in $G[H \cup C]$.

By the previous lemmas, we obtain a crown decomposition $(V_-, V_+, V_{1/2})$ of $G$ if $V_- \neq \emptyset$.

Crown Decomposition: Examples

\[
\begin{array}{c}
\text{crown decomposition} \\
(\{a, e, g\}, \{b, d, f\}, \{c\})
\end{array}
\hspace{1cm}
\begin{array}{c}
\text{has no crown decomposition}
\end{array}
\]

Using the crown decomposition

Lemma 15. Suppose that $G = (V, E)$ has a crown decomposition $(C, H, B)$. Then,

$$vc(G) \leq k \iff vc(G[B]) \leq k - |H|,$$

where $vc(G)$ denotes the size of the smallest vertex cover of $G$.

Proof. ($\Rightarrow$): Let $S$ be a vertex cover of $G$ with $|S| \leq k$. Since $S$ contains at least one vertex for each edge of a matching, $|S \cap (C \cup H)| \geq |H|$. Therefore, $S \cap B$ is a vertex cover for $G[B]$ of size at most $k - |H|$.

($\Leftarrow$): Let $S$ be a vertex cover of $G[B]$ with $|S| \leq k - |H|$. Then, $S \cup H$ is a vertex cover of $G$ of size at most $k$, since each edge that is in $G$ but not in $G'$ is incident to a vertex in $H$.

Nemhauser-Trotter

Corollary 16 ([Nemhauser, Trotter, 1974]). There exists a smallest vertex cover $S$ of $G$ such that $S \cap V_- = \emptyset$ and $V_+ \subseteq S$.

\footnote{A matching $M$ in a graph $G$ is a set of edges such that no two edges in $M$ have a common endpoint. A matching saturates a set of vertices $S$ if each vertex in $S$ is an end point of an edge in $M$.}
Crown reduction

(Crown Reduction)
If solving LP$_{VC}(G)$ gives an optimal solution with $V_\neg \neq \emptyset$, then return $(G - (V_\neg \cup V_\neg), k - |V_\neg|)$.

(Number of Vertices)
If solving LP$_{VC}(G)$ gives an optimal solution with $V_\neg = \emptyset$ and $|V| > 2k$, then return No.

Lemma 17. (Crown Reduction) and (Number of Vertices) are sound.

Proof. (Crown Reduction) is sound by previous Lemmas. Let $\alpha$ be an optimal solution for LP$_{VC}(G)$ and suppose $V_\neg = \emptyset$. The value of this solution is at least $|V|/2$. Thus, the value of an optimal solution for ILP$_{VC}(G)$ is at least $|V|/2$. Since $G$ has no vertex cover of size less than $|V|/2$, we have a No-instance if $k < |V|/2$. \qed

Linear vertex-kernel for Vertex Cover

Theorem 18. VERTEX COVER has a kernel with $2k$ vertices and $O(k^2)$ edges.

This is the smallest known kernel for VERTEX COVER. See [http://fpt.wikidot.com/fpt-races](http://fpt.wikidot.com/fpt-races) for the current smallest kernels for various problems.

4 More on Crown Decompositions

Crown Lemma

Lemma 19 (Crown Lemma). Let $G = (V, E)$ be a graph without isolated vertices and with $|V| \geq 3k + 1$. There is a polynomial time algorithm that either

- finds a matching of size $k + 1$ in $G$, or
- finds a crown decomposition of $G$.

To prove the lemma, we need König’s Theorem

Theorem 20 ([König, 1916]). In every bipartite graph the size of a maximum matching is equal to the size of a minimum vertex cover.

Proof of the Crown Lemma. Compute a maximum matching $M$ of $G$. If $|M| \geq k + 1$, we are done. Note that $I := V \setminus V(M)$ is an independent set with $|V| - |V(M)| \geq k + 1$ vertices. Consider the bipartite graph $B$ formed by edges with one endpoint in $V(M)$ and the other in $I$. Compute a minimum vertex cover $X$ and a maximum matching $M'$ of $B$. We know: $|X| = |M'| \leq |M| \leq k$. Hence, $X \cap V(M) \neq \emptyset$. Let $M^* = \{e \in M' : e \cap (X \cap V(M)) \neq \emptyset\}$. We obtain a crown decomposition with

- crown $C = V(M^*) \cap I$
- head $H = X \cap V(M) = X \cap V(M^*)$, and
- body $B = V \setminus (C \cup H)$.

As an exercise, verify that $(C, H, B)$ is indeed a crown decomposition. \qed

5 Kernels and Fixed-parameter tractability

Theorem 21. Let $\Pi$ be a decidable parameterized problem. $\Pi$ has a kernelization algorithm $\iff \Pi$ is FPT.

Proof. ($\Rightarrow$): An FPT algorithm is obtained by first running the kernelization, and then any brute-force algorithm on the resulting instance.

(\$\Leftarrow\$): Let $A$ be an FPT algorithm for $\Pi$ with running time $O(f(k)n^c)$. If $f(k) < n$, then $A$ has running time $O(n^{c+1})$. In this case, the kernelization algorithm runs $A$ and returns a trivial Yes- or No-instance depending on the answer of $A$. Otherwise, $f(k) \geq n$. In this case, the kernelization algorithm outputs the input instance. \qed
After computing a kernel ...

- ... we can use any algorithm to compute an actual solution.
- Brute-force, faster exponential-time algorithms, parameterized algorithms, often also approximation algorithms

Kernels

- A parameterized problem may not have a kernelization algorithm
  - Example, COLORING\(\) parameterized by \(k\) has no kernelization algorithm unless \(P = NP\).
  - A kernelization would lead to a polynomial time algorithm for the NP-complete 3-COLORING problem
- Kernelization algorithms lead to FPT algorithms ...
- ... FPT algorithms lead to kernels

6 Further Reading


\(^2\)Can one color the vertices of an input graph \(G\) with \(k\) colors such that no two adjacent vertices receive the same color?