2a. Kernelization

COMP6741: Parameterized and Exact Computation

Serge Gaspers

19T3

Contents

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

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

3 Kernel for Hamiltonian Cycle ........ 4

4 Kernel for Edge Clique Cover ....... 4

5 Kernels and Fixed-parameter tractability 6

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

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 $v \in S$.

**Vertex Cover**

<table>
<thead>
<tr>
<th>Input:</th>
<th>A graph $G = (V, E)$ and an integer $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter:</td>
<td>$k$</td>
</tr>
<tr>
<td>Question:</td>
<td>Does $G$ have a vertex cover of size at most $k$?</td>
</tr>
</tbody>
</table>

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 size at most $k$ for $G - v$; 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. \((\text{Number of Edges})\) 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 \(k'\).

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

Effectiveness of preprocessing algorithms

- How effective is \textit{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 \(P = NP\)
- We need a different measure of effectiveness

Measuring the effectiveness of preprocessing algorithms

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

Effectiveness of \textit{VC-preprocess}

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

Can we obtain a kernel with fewer vertices?
We defer this question for now.

## 3 Kernel for Hamiltonian Cycle

A Hamiltonian cycle of $G$ is a subgraph of $G$ that is a cycle on $|V(G)|$ vertices.

<table>
<thead>
<tr>
<th>VC-HAMILTONIAN CYCLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: A graph $G = (V, E)$.</td>
</tr>
<tr>
<td>Parameter: $k = \text{vc}(G)$, the size of a smallest vertex cover of $G$.</td>
</tr>
<tr>
<td>Question: Does $G$ have a Hamiltonian cycle?</td>
</tr>
</tbody>
</table>

**Thought experiment:** Imagine a very large instance where the parameter is tiny. How can you simplify such an instance?

**Issue:** We do not actually know a vertex cover of size $k$. We do not even know the value of $k$ (it is not part of the input).

- Obtain a vertex cover using an approximation algorithm. We will use a 2-approximation algorithm, producing a vertex cover of size $\leq 2k$ in polynomial time.
- If $C$ is a vertex cover of size $\leq 2k$, then $I = V \setminus C$ is an independent set of size $\geq |V| - 2k$.
- No two consecutive vertices in the Hamiltonian Cycle can be in $I$.
- A kernel with $\leq 4k$ vertices can now be obtained with the following simplification rule.

(Too-large)
Compute a vertex cover $C$ of size $\leq 2k$ in polynomial time. If $2|C| < |V|$, then return No

## 4 Kernel for Edge Clique Cover

**Definition 8.** An edge clique cover of a graph $G = (V, E)$ is a set of cliques in $G$ covering all its edges. In other words, if $C \subseteq 2^V$ is an edge clique cover then each $S \in C$ is a clique in $G$ and for each $\{u, v\} \in E$ there exists an $S \in C$ such that $u, v \in S$.

Example: $\{\{a, b, c\}, \{b, c, d, e\}\}$ is an edge clique cover for this graph.

<table>
<thead>
<tr>
<th>EDGE CLIQUE COVER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: A graph $G = (V, E)$ and an integer $k$</td>
</tr>
<tr>
<td>Parameter: $k$</td>
</tr>
<tr>
<td>Question: Does $G$ have an edge clique cover of size at most $k$?</td>
</tr>
</tbody>
</table>

The size of an edge clique cover $C$ is the number of cliques contained in $C$ and is denoted $|C|$. 
Helpful properties

**Definition 9.** A clique $S$ in a graph $G$ is a maximal clique if there is no other clique $S'$ in $G$ with $S \subseteq S'$.

**Lemma 10.** A graph $G$ has an edge clique cover $C$ of size at most $k$ if and only if $G$ has an edge clique cover $C'$ of size at most $k$ such that each $S \in C'$ is a maximal clique.

*Proof sketch.* (⇒): Replace each clique $S \in C$ by a maximal clique $S'$ with $S \subseteq S'$.

(⇐): Trivial, since $C'$ is an edge clique cover of size at most $k$.

---

**Simplification rules for Edge Clique Cover**

**Thought experiment:** Imagine a very large instance where the parameter is tiny. How can you simplify such an instance?

The instance could have many degree-0 vertices.

**(Isolated)**

If there exists a vertex $v \in V$ with $d_G(v) = 0$, then set $G \leftarrow G - v$.

**Lemma 11.** *(Isolated) is sound.*

*Proof sketch.* Since no edge is incident to $v$, a smallest edge clique cover for $G - v$ is a smallest edge clique cover for $G$, and vice-versa.

**(Isolated-Edge)**

If $\exists uv \in E$ such that $d_G(u) = d_G(v) = 1$, then set $G \leftarrow G - \{u, v\}$ and $k \leftarrow k - 1$.

**(Twins)**

If $\exists u, v \in V$, $u \neq v$, such that $N_G[u] = N_G[v]$, then set $G \leftarrow G - v$.

**Lemma 12.** *(Twins) is sound.*

*Proof.* We need to show that $G$ has an edge clique cover of size at most $k$ if and only if $G - v$ has an edge clique cover of size at most $k$.

(⇒): If $C$ is an edge clique cover of $G$ of size at most $k$, then $\{S \setminus \{v\} : S \in C\}$ is an edge clique cover of $G - v$ of size at most $k$.

(⇐): Let $C'$ be an edge clique cover of $G - v$ of size at most $k$. Partition $C'$ into $C'_u = \{S \in C' : u \in S\}$ and $C'_{-u} = C' \setminus C'_u$. Note that each set in $C'_u = \{S \cup \{v\} : S \in C'_u\}$ is a clique in $G$ since $N_G[u] = N_G[v]$ and that each edge incident to $v$ is contained in at least one of these cliques. Now, $C'_u \cup C'_{-u}$ is an edge clique cover of $G$ of size at most $k$.

**(Size-V)**

If the previous simplification rules do not apply and $|V| > 2^k$, then return No.

**Lemma 13.** *(Size-V) is sound.*

*Proof.* For the sake of contradiction, assume neither (Isolated) nor (Twins) are applicable, $|V| > 2^k$, and $G$ has an edge clique cover $C$ of size at most $k$. Since $2^C$ (the set of all subsets of $C$) has size at most $2^k$, and every vertex belongs to at least one clique in $C$ by (Isolated), we have that there exists two vertices $u, v \in V$ such that $\{S \in C : u \in S\} = \{S \in C : v \in S\}$. But then, $N_G[u] = \bigcup_{S \in C : u \in S} S = \bigcup_{S \in C : v \in S} S = N_G[v]$, contradicting that (Twin) is not applicable.

---

**Kernel for Edge Clique Cover**

**Theorem 14.** Edge Clique Cover has a kernel with $O(2^k)$ vertices and $O(4^k)$ edges.

**Corollary 15.** Edge Clique Cover is FPT.
5 Kernels and Fixed-parameter tractability

Theorem 16. 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.

6 Further Reading

- Chapter 2, Kernelization in [Cyg+15]
- Chapter 4, Kernelization in [DF13]
- Chapter 7, Data Reduction and Problem Kernels in [Nie06]
- Chapter 9, Kernelization and Linear Programming Techniques in [FG06]
- the new book on kernelization [Fom+19]

References