

Kernelization

Serge Gaspers

UNSW

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Vertex cover

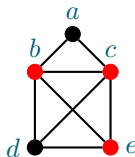
A **vertex cover** of a graph $G = (V, E)$ is a subset of vertices $S \subseteq V$ such that for each edge $\{u, v\} \in E$, we have $u \in S$ or $v \in S$.

VERTEX COVER

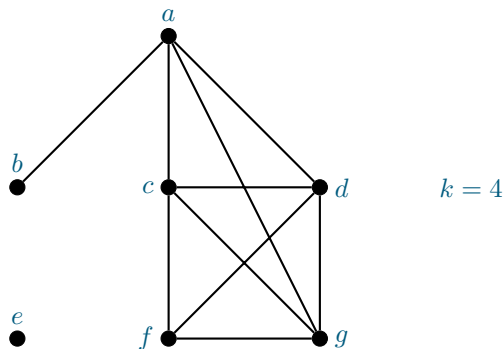
Input: A graph $G = (V, E)$ and an integer k

Parameter: k

Question: Does G have a vertex cover of size at most k ?



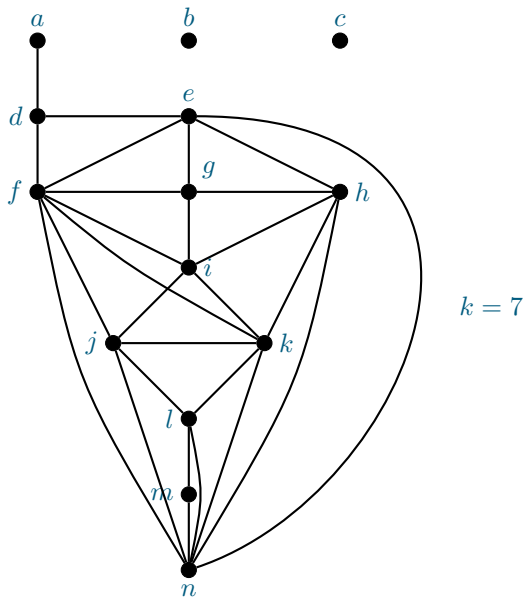
Exercise 1



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



Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Simplification rules for VERTEX COVER

(Degree-0)

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

Simplification rules for VERTEX COVER

(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.

Simplification rules for VERTEX COVER

(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. \square

Simplification rules for VERTEX COVER

(Degree-1)

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

Simplification rules for VERTEX COVER

(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 1

(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 u . □

Simplification rules for VERTEX COVER

(Large Degree)

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

Simplification rules for VERTEX COVER

(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 1

(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 $|S| \leq k$. □

Simplification rules for VERTEX COVER

(Number of Edges)

If $d_G(v) \leq k$ for each $v \in V$ and $|E| > k^2$ then return No

Simplification rules for VERTEX COVER

(Number of Edges)

If $d_G(v) \leq k$ for each $v \in V$ and $|E| > k^2$ then return **No**

Lemma 1

(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 . □

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Preprocessing algorithm for VERTEX COVER

VC-preprocess

Input: A graph G and an integer k .

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' .

$G' \leftarrow G$

$k' \leftarrow k$

repeat

 Execute simplification rules (Degree-0), (Degree-1), (Large Degree), and
 (Number of Edges) for (G', k')

until *no simplification rule applies*

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 Π is **nice** if it runs in polynomial time and for each instance for Π , it returns an instance for Π that is strictly smaller.

First try

- Say that a preprocessing algorithm for a problem Π is **nice** if it runs in polynomial time and for each instance for Π , it returns an instance for Π 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 Π 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 VC-preprocess

Lemma 2

For any instance (G, k) for VERTEX COVER, 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, $|V'| + |E'| \leq O(k^2)$. □

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Definition 3

A **kernelization** for a parameterized problem Π is a **polynomial time** algorithm, which, for any instance I of Π with parameter k , produces an **equivalent** instance I' of Π 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 **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 4

VC-preprocess is a $O(k^2)$ kernelization for VERTEX COVER.

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

HAMILTONIAN CYCLE I

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

vc-HAMILTONIAN CYCLE

Input: A graph $G = (V, E)$.

Parameter: $k = vc(G)$, the size of a smallest vertex cover of G .

Question: Does G have a Hamiltonian cycle?

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).

HAMILTONIAN CYCLE III

- 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**

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

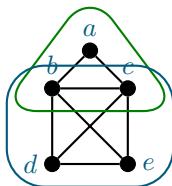
EDGE CLIQUE COVER

Definition 5

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 $\mathcal{C} \subseteq 2^V$ is an edge clique cover then each $S \in \mathcal{C}$ is a clique in G and for each $\{u, v\} \in E$ there exists an $S \in \mathcal{C}$ such that $u, v \in S$.

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



EDGE CLIQUE COVER

EDGE CLIQUE COVER

Input: A graph $G = (V, E)$ and an integer k

Parameter: k

Question: Does G have an edge clique cover of size at most k ?

The **size** of an edge clique cover \mathcal{C} is the number of cliques contained in \mathcal{C} and is denoted $|\mathcal{C}|$.

Helpful properties

Definition 5

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

Lemma 6

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

Proof sketch.

(\Rightarrow): Replace each clique $S \in \mathcal{C}$ by a maximal clique S' with $S \subseteq S'$.

(\Leftarrow): Trivial, since \mathcal{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?

Simplification rules for EDGE CLIQUE COVER II

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 7

(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. \square

Simplification rules for EDGE CLIQUE COVER II

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 7

(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. \square

(Isolated-Edge)

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

Simplification rules for EDGE CLIQUE COVER III

(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 8

(Twins) is sound.

Simplification rules for EDGE CLIQUE COVER III

(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 8

(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 .

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

(\Leftarrow): Let \mathcal{C}' be an edge clique cover of $G - v$ of size at most k . Partition \mathcal{C}' into $\mathcal{C}'_u = \{S \in \mathcal{C}' : u \in S\}$ and $\mathcal{C}'_{\neg u} = \mathcal{C}' \setminus \mathcal{C}'_u$. Note that each set in $\mathcal{C}_u = \{S \cup \{v\} : S \in \mathcal{C}'_u\}$ 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, $\mathcal{C}_u \cup \mathcal{C}'_{\neg u}$ is an edge clique cover of G of size at most k . \square

Simplification rules for EDGE CLIQUE COVER IV

(Size-V)

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

Lemma 9

(Size-V) is sound.

Simplification rules for EDGE CLIQUE COVER IV

(Size-V)

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

Lemma 9

(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 \mathcal{C} of size at most k . Since $2^{\mathcal{C}}$ (the set of all subsets of \mathcal{C}) has size at most 2^k , and every vertex belongs to at least one clique in \mathcal{C} by (Isolated), we have that there exists two vertices $u, v \in V$ such that $\{S \in \mathcal{C} : u \in S\} = \{S \in \mathcal{C} : v \in S\}$. But then, $N_G[u] = \bigcup_{S \in \mathcal{C}: u \in S} S = \bigcup_{S \in \mathcal{C}: v \in S} S = N_G[v]$, contradicting that (Twin) is not applicable. \square

Theorem 10 ((Gramm et al., 2008))

EDGE CLIQUE COVER *has a kernel with $O(2^k)$ vertices and $O(4^k)$ edges.*

Corollary 11

EDGE CLIQUE COVER *is FPT.*

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Theorem 12

Let Π be a decidable parameterized problem.

Π has a kernelization algorithm $\Leftrightarrow \Pi$ is FPT.

Kernels and Fixed-parameter tractability

Theorem 12

Let Π be a decidable parameterized problem.

Π has a kernelization algorithm $\Leftrightarrow \Pi$ 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 Π 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. □

Outline

- 1 Vertex Cover
 - Simplification rules
 - Preprocessing algorithm
- 2 Kernelization algorithms
- 3 Kernel for HAMILTONIAN CYCLE
- 4 Kernel for EDGE CLIQUE COVER
- 5 Kernels and Fixed-parameter tractability
- 6 Further Reading

Further Reading

- Chapter 2, *Kernelization* in (Cygan et al., 2015)
- Chapter 4, *Kernelization* in (Downey and Fellows, 2013)
- Chapter 7, *Data Reduction and Problem Kernels* in (Niedermeier, 2006)
- Chapter 9, *Kernelization and Linear Programming Techniques* in (Flum and Grohe, 2006)
- the kernelization book (Fomin et al., 2019)

References

- Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh (2015). *Parameterized Algorithms*. Springer. DOI: [10.1007/978-3-319-21275-3](https://doi.org/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](https://doi.org/10.1007/978-1-4471-5559-1).
- Jörg Flum and Martin Grohe (2006). *Parameterized Complexity Theory*. Springer. DOI: [10.1007/3-540-29953-X](https://doi.org/10.1007/3-540-29953-X).
- Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi (2019). *Kernelization. Theory of Parameterized Preprocessing*. Cambridge University Press.
- Jens Gramm, Jiong Guo, Falk Huffner, and Rolf Niedermeier (2008). “Data reduction and exact algorithms for clique cover”. In: *ACM J. Exp. Algorithmics* 13. DOI: [10.1145/1412228.1412236](https://doi.org/10.1145/1412228.1412236).
- Rolf Niedermeier (2006). *Invitation to Fixed Parameter Algorithms*. Oxford University Press. DOI: [10.1093/ACPROF:OS0/9780198566076.001.0001](https://doi.org/10.1093/ACPROF:OS0/9780198566076.001.0001).