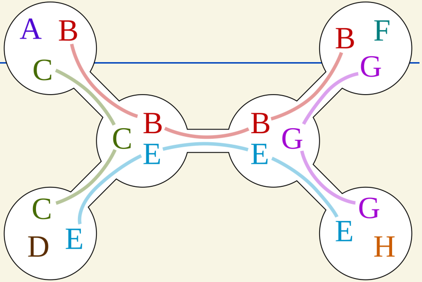
Preprocessing and Kernelization I



DM898: Parameterized Algorithms
Lars Rohwedder

Today's lecture

- Formalism for preprocessing (kernelization)
- Examples

Next lecture: advanced kernelization and preprocessing in practice (linear programming)

Theory of preprocessing

Definition (data reduction rule)

Algorithm that takes instance (I, k) of some parameterized problem Q and returns new instance (I', k') of the same problem Q. It satisfies

- Polynomial running time,
- (I', k') is smaller or simpler than (I, k) by some measure (or $(I', k') = (I, k)$ if no reduction possible),
- Instance (I, k) is a YES-instance if and only if instance (I', k') is a YES-instance. **(safeness)**

We allow it also to directly output YES/NO answer (can be treated as outputting a trivial YES or NO instance).

The guarantee on being smaller is intentionally vague and in this form not theoretically useful.

Definition (data reduction rule)

Algorithm that takes instance (I, k) of some parameterized problem Q and returns new instance (I', k') of the same problem Q. It satisfies

- Polynomial running time,
- (I', k') is smaller or simpler than (I, k) by some measure (or $(I', k') = (I, k)$ if no reduction possible),
- Instance (I, k) is a YES-instance if and only if instance (I', k') is a YES-instance. **(safeness)**

We allow it also to directly output YES/NO answer (can be treated as outputting a trivial YES or NO instance).

The guarantee on being smaller is intentionally vague and in this form not theoretically useful.

When we have a strong bound on the output size of the reduction rule, we use the term **kernelization**.

Definition (kernelization algorithm)

Data reduction rule, for which there exists a computable function $g(k)$ such that for every input instance (I, k) and corresponding output instance (I', k') it holds that

- $|I'| + k' \leq g(k)$.

We say, a problem Q has a **kernel of size** $g(k)$ if there exists such a kernelization algorithm.

We are interested in small kernels, for example, when $g(k)$ is linear in k or quadratic in k.

FPT algorithm from kernel

Theorem

Let Q be a decidable parameterized problem. If Q has a kernel, then it also has an FPT algorithm.

Proof. Since the problem is decidable, there is an algorithms that has a finite running time $f(|I|+k)$. Apply the kernelization algorithm to obtain an equivalent instance I', k' with $|I'|+k' \leq g(k)$ and then run the finite algorithm on it. The total running time is

$$\underbrace{|I|^{O(1)}}_{\text{kernelization}} + f(|I'|+k') \leq |I|^{O(1)} + (f \circ g)(k).$$

Kernel from FPT algorithm

Perhaps surprisingly, the other direction holds as well

Theorem

Let Q be a parameterized problem. If Q has an FPT algorithm, then it also has a kernel.

Kernel from FPT algorithm

Perhaps surprisingly, the other direction holds as well

Theorem

Let Q be a parameterized problem. If Q has an FPT algorithm, then it also has a kernel.

Proof. Let A be an algorithm deciding instances (I, k) of Q in time $|I|^c f(k)$. The following is a kernelization algorithm: run A for $|I|^{c+1}$ time. If it terminates by then, output its decision. Otherwise, return the same instance (I, k).

Note that if A did not terminate, then

$$|I|^{c+1} < |I|^c f(k).$$

Thus,

$$|I| + k < f(k) + k. \quad \square$$

Thus, kernelization is **equivalent** to FPT. This theorem, however, is mainly of theoretical interest. Kernels obtained in this way are usually impractically large.

Examples

Vertex Cover

In the first week we have seen the following reduction rules for Vertex Cover:

Reduction rule 1

If G contains an isolated vertex v, delete v from G. The new instance is $(G - v, k)$.

This reduction is safe because every solution for $(G - v, k)$ is a solution for (G, k) and for every solution U of (G, k), $U \setminus \{v\}$ is a solution for $(G - v, k)$.

Vertex Cover

In the first week we have seen the following reduction rules for Vertex Cover:

Reduction rule 1

If G contains an isolated vertex v, delete v from G. The new instance is $(G - v, k)$.

This reduction is safe because every solution for $(G - v, k)$ is a solution for (G, k) and for every solution U of (G, k), $U \setminus \{v\}$ is a solution for $(G - v, k)$.

Reduction rule 2

If G contains a vertex v of degree $> k$, delete v from G and decrement k by 1. The new instance is $(G - v, k - 1)$.

This reduction is safe because every solution U for (G, k) must contain v. Therefore $U \setminus \{v\}$ is of size $k - 1$ and a solution for $(G - v, k - 1)$. For every solution U of $(G - v, k - 1)$, $U \cup \{v\}$ is a solution for (G, k).

Vertex Cover

In the first week we have seen the following reduction rules for Vertex Cover:

Reduction rule 1

If G contains an isolated vertex v, delete v from G. The new instance is $(G - v, k)$.

This reduction is safe because every solution for $(G - v, k)$ is a solution for (G, k) and for every solution U of (G, k), $U \setminus \{v\}$ is a solution for $(G - v, k)$.

Reduction rule 2

If G contains a vertex v of degree $> k$, delete v from G and decrement k by 1. The new instance is $(G - v, k - 1)$.

This reduction is safe because every solution U for (G, k) must contain v. Therefore $U \setminus \{v\}$ is of size $k - 1$ and a solution for $(G - v, k - 1)$. For every solution U of $(G - v, k - 1)$, $U \cup \{v\}$ is a solution for (G, k).

Kernel with $O(k^2)$ vertices and $O(k^2)$ edges: Apply all reductions exhaustively. If remaining instance has more than k^2 edges, return NO-instance.

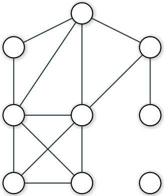
Proof of safeness. If G is YES-instance, then there exists $U \subseteq V$, $|U| \leq k$, that covers all edges E. Thus,

$$|E| \leq \sum_{u \in U} \deg(u) \leq k^2 \qquad \text{and} \qquad |V| \leq \sum_{v \in V} \deg(v) \leq 2|E| \leq 2k^2.$$

Edge Clique Cover

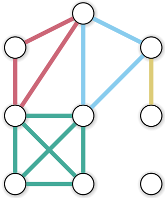
Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.



Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

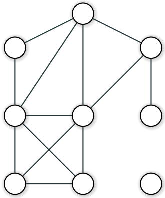


Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

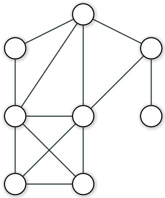


Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.



Edge Clique Cover

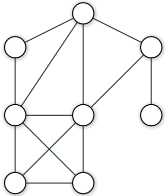
Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.



Edge Clique Cover

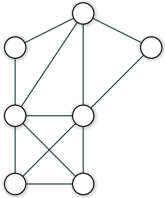
Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.



Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

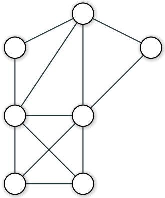
If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).



Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

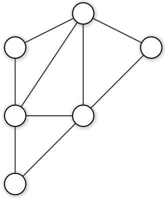
If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).



Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

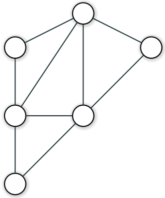
Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).

Safeness of Rule 3 (others omitted): solution for $(G - v, k)$ can be extended to (G, k) by adding v to all cliques that u is contained in.



Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

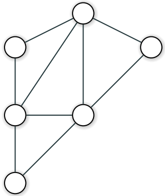
Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).

Safeness of Rule 3 (others omitted): solution for $(G - v, k)$ can be extended to (G, k) by adding v to all cliques that u is contained in.



Kernel with $\leq 2^k$ vertices: Apply previous rules exhaustively. If remaining instance has more than 2^k vertices, return NO-instance.

Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

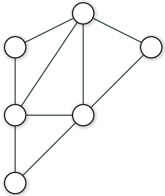
Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).

Safeness of Rule 3 (others omitted): solution for $(G - v, k)$ can be extended to (G, k) by adding v to all cliques that u is contained in.



Kernel with $\leq 2^k$ vertices: Apply previous rules exhaustively. If remaining instance has more than 2^k vertices, return NO-instance.

Proof of safeness. We show that in a YES-instance with more than 2^k vertices one of the rules cannot be exhausted. Let $C_1, \ldots, C_k$ be an edge clique cover. If $|V| > 2^k$, there must be vertices $u \neq v$ contained in exactly the same cliques.

Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

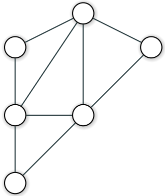
Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).

Safeness of Rule 3 (others omitted): solution for $(G - v, k)$ can be extended to (G, k) by adding v to all cliques that u is contained in.



Kernel with $\leq 2^k$ vertices: Apply previous rules exhaustively. If remaining instance has more than 2^k vertices, return NO-instance.

Proof of safeness. We show that in a YES-instance with more than 2^k vertices one of the rules cannot be exhausted. Let $C_1, \ldots, C_k$ be an edge clique cover. If $|V| > 2^k$, there must be vertices $u \neq v$ contained in exactly the same cliques.

If $N[u] = \{u\}$ or $N[v] = \{v\}$ then Rule 1 applies.

Otherwise, u and v are in some clique. Thus,

$$N[u] = \bigcup_{C_i \ni u} C_i = \bigcup_{C_i \ni v} C_i = N[v].$$

If $N[u] = N[v] = \{u, v\}$ then Rule 2 applies.

Edge Clique Cover

Given graph $G = (V, E)$ and number $k \in \mathbb{N}$, decide if the edges of G can be covered by k cliques.

Reduction Rule 1

If $G = (V, E)$ contains an isolated vertex v, return $(G - v, k)$.

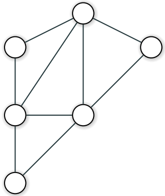
Reduction Rule 2

If $G = (V, E)$ contains a vertex v with $\deg(v) = 1$, return $(G - v, k - 1)$.

Reduction Rule 3

If $G = (V, E)$ contains vertices u, v with $N[u] = N[v] \supsetneq \{u, v\}$, return $(G - v, k)$. Here $N[\cdot]$ is the closed neighborhood (includes vertex itself).

Safeness of Rule 3 (others omitted): solution for $(G - v, k)$ can be extended to (G, k) by adding v to all cliques that u is contained in.



Kernel with $\leq 2^k$ vertices: Apply previous rules exhaustively. If remaining instance has more than 2^k vertices, return NO-instance.

Proof of safeness. We show that in a YES-instance with more than 2^k vertices one of the rules cannot be exhausted. Let $C_1, \ldots, C_k$ be an edge clique cover. If $|V| > 2^k$, there must be vertices $u \neq v$ contained in exactly the same cliques.

If $N[u] = \{u\}$ or $N[v] = \{v\}$ then Rule 1 applies.

Otherwise, u and v are in some clique. Thus,

$$N[u] = \bigcup_{C_i \ni u} C_i = \bigcup_{C_i \ni v} C_i = N[v].$$

If $N[u] = N[v] = \{u, v\}$ then Rule 2 applies.

If $N[u] = N[v] \supsetneq \{u, v\}$ then Rule 3 applies.