KAIST, School of Computing, Spring 2024 | **Scribed By: Eunjung KIM**
Algorithms for NP-hard problems (CS492) | **Lecture #3**
Lecture: Eunjung KIM | **5 March 2024**

Contents

1 Kernelization 101

A *kernelization* of P is a polynomial-time (in $|x|$ and k) algorithm which transforms an instance (x, k) of P into another instance (x', k') of P satisfying

- equivalence: $(x, k) \in P$ if and only if $(x', k') \in P$
- size bound: $|x| \leq g(k)$ and $k' \leq g(k)$ for some computable function

An instance (x', k') obtained after applying kernelization is called a kernel. The function $g(k)$ is the size of a kernel.

Usually, a kernelization consists in applying a sequence of *reduction rules*. A reduction rule for P is a polynomial-time (in $|x|$ and k) algorithm which transforms an instance (x, k) of P into an equivalent instance (x', k') of P. The equivalence of (x, k) and (x', k') is also referred to as the *soundness* or *safeness* of the reduction rule. Notice that the size of the resulting instance of a reduction rule is not necessarily bounded. We say that an instance (x, k) is *irreducible with respect to a reduction rule* R if R cannot be applied to (x, k) anymore (or equivalently, applying R does not change the instance).

Lemma 1. *A decidable parameterized problem P is FPT if and only if it admits a kernelization.*

Proof: ($\Leftarrow$) If P admits a kernel of size $g(k)$, then run the kernelization algorithm (takes polynomial time in $|x| + k$) and then do an exhaustive search on the obtained kernel to decide whether $(x', k') \in P$. The whole procedure is an FPT-algorithm.
($\Rightarrow$) Let P has an FPT-algorithm $\mathcal{A}$ running in time $f(k) \cdot |x|^c$ for some constant c. Then run $\mathcal{A}$ on the given instance (x, k) for time $|x|^{c+1}$. If it ouputs YES/NO answer, then produce a constant-size instance of P accordingly. This would be the kernel. If $\mathcal{A}$ does not terminate in time $|x|^{c+1}$, this means $|x| < f(k)$. That is, the given instance (x, k) is already a kernel, and thus output (x, k). $\square$

Devising a kernelization with small size bound $g(k)$ (usually, polynomial g) is one of the most active research topic in parameterized complexity. Kernelization design involve the following steps.

- Devise reduction rules.
- Prove that the above reduction rules are safe.
- Prove that when an instance (x', k') of P is irreducible w.r.t the reduction rules, $|x'| \leq g(k)$ and $k' \leq g(k)$. The smaller the function g is, the better. Notice that the equivalence of kernelization is automatically guaranteed by the safeness of reduction rules.

2 Simple kernelization: VERTEX COVER

We look at a simple kernelization for VERTEX COVER yielding $O(k^2)$ vertices.

Reduction Rule 1: If a vertex v is isolated in G, then delete v. The new parameter is $k' := k$.

Reduction Rule 2: If a vertex v is incident with at least $k + 1$ edges in G, then delete v and set $k' := k - 1$.

It is trivial to see that Reduction Rule 1 is safe. To see that Reduction Rule 2 is safe, notice that any vertex cover of size at most k in G must contain v. Hence, if (G, k) is a yes-instance, $(G - v, k - 1)$ is also yes-instance. The opposite direction of equivalence is straightforward.

Consider an instance (G', k') for which none of Reduction Rules 1 and 2 can be applied, and let us analyze the size of G'. Since the parameter does not increase with the reduction rules, we know that $k' \leq k$.

Suppose G' is a yes-instance, and C is a vertex cover of G' with $|C| \leq k'$. Since (G', k') is irreducible with respect to Reduction Rule 2, every $v \in C$ is incident with at most k' edges of G. As each edge of G' is incident with C and $k' \leq k$, there are at most $|C| \cdot k' \leq k^2$ edges in G. Due to Reduction Rule 1, there's no isolated vertex in G' and thus every vertex in $V(G) \setminus C$ is adjacent with some vertex of C. For $deg(v) \leq k'$ for all $v \in C$, we can assign each vertex of $V(G') \setminus C$ to a vertex of C so that each vertex of C is assigned with at most k' vertices. Now, we have $|V(G')| = |V(G')) \setminus C| + |C| \leq |C| \cdot k' + k' = k(k+1)$. To bound $|E(G')|$, we similarly observe that every edge of G' is incident with (at least) one vertex of C, leading to $|E(G')| \leq |C| \cdot k' \leq k^2$.

Hence, if $|V(G')| > k(k+1)$ or $|E(G')| > k^2$, we know that (G', k') is a no-instance and output a constant-size no-instance as a kernel. Otherwise, (G', k') is a kernel with $|V(G')| \leq k(k+1)$ and $|E(G')| \leq k^2$.

3 LP-based kernelization for VERTEX COVER

Theorem 1 (Nemhauser-Trotter Theorem, NT Theorem in short)**.** *Given a graph G, a partition (R, H, C) satisfying the following can be computed in polynomial time.*

(a.) For any vertex cover S_r of $G[R]$, $S_r \cup H$ is a vertex cover of G.

(b.) There exists an optimal vertex cover containing H.

(c.) Any vertex cover of $G[R]$ is of size at least $\frac{1}{2}|R|$.

Before presenting an algorithm for computing such a partition (R, H, C), let's think how to use the above NT Theorem for computing a kernel. We propose the following reduction rules.

Reduction Rule 1: Let (R, H, C) be a partition such that (a)-(c) of NT Theorem is met and $H \cup C \neq \emptyset$. Then delete $H \cup C$ from G and set $k' := k - |H|$, i.e. the new instance is $(G[R], k - |H|)$.

Lemma 2. *Reduction Rule 1 is safe; (G, k) is a* YES*-instance if and only if $(G[R], k - |H|)$ is a* YES*-instance.*

Proof: Suppose (G, k) is a yes-instance and let S be an optimal vertex cover. Notice that $|S| \leq k$. By condition (b) of NT Theorem, we can assume that $H \subseteq S$. Take $S_r := S \cap R$ and observe that S_r is a vertex cover of $G[R]$. Due to condition (a), $S_r \cup H$ is a vertex cover of G. Since $S_r \cup H \subseteq S$ and is a vertex cover of G, the optimality of S implies that $S \cap C = \emptyset$. Hence, $|S_r| = |S| - |H| \leq k - |H|$ and $(G[R], k - |H|)$ is a yes-instance.

For the opposite direction, suppose $(G[R], k - |H|)$ is a yes-instance and let S_r be a vertex cover of $G[R]$ of size at most $k - |H|$. By condition (a), $S_r \cup H$ is a vertex cover of G and its size is at most k. That is, (G, k) is a yes-instance. $\square$

Lemma 3. VERTEX COVER *admits a kernel containing at most $2k$ vertices.*

Proof: Consider the following (kernelization) algorithm: find a partition (R, H, C) as in NT Theorem in polynomial time and apply Reduction Rule 1 if possible. Let $(G', k') = (G[R], k - |H|)$ be the resulting instance, which can be identical to (G, k) if $H \cup C = \emptyset$ and Reduction Rule 1 was not applied. The algorithm then performs the following.

- If $|R| > 2k$, output a constant-size no-instance.
- Otherwise, output (G', k').

In the first case, observe that any vertex cover of $G[R]$ contains more than k vertices by condition (c) of NT Theorem, and thus (G', k') is a no-instance. Therefore the output instance is equivalent to (G', k'), The equivalence of the output instance to (G, k) in both cases follows by Lemma . It is clear that the output instance has at most $2k$ vertices and that the algorithm runs in polynomial assuming the partition (R, H, C) of NT Theorem can be found in polynomial time. □

How can we find a partition (R, H, C) as in NT Theorem? There are several nice proofs of NT Theorem, and we have already seen a version using LP Relaxation of VERTEX COVER during the lecture in Week 01 (See the lecture note). Here is a reminder.

$$\begin{aligned}
\min \sum_{u \in V(G)} & x_u \\
& x_u + x_v \geq 1 && \forall (u, v) \in E(G) \\
& x_u \geq 0 && \forall u \in V(G)
\end{aligned}$$

Define

- $R_0 := \{u \in V(G) : x_u^* = 0.5\}$
- $H_0 := \{u \in V(G) : x_u^* > 0, 5\}$
- $C_0 := \{u \in V(G) : x_u^* < 0, 5\}$

LP can be solved in polynomial time, hence the partition (R_0, H_0, C_0) can be found in polynomial time. We already have seen that this partition satisfies the conditions (a)-(c) of NT Theorem in the previous lectures. Here is a recap.

Lemma 4. *The partition (R_0, H_0, C_0) meets the condition (a).*

Proof: Observe that C_0 is an independent set: indeed if there is an edge between $u, v \in C_0$, we have $x_u^* + x_v^* < 0.5 + 0.5 = 1$, violating the corresponding inequality in LP. For the same reason, there is no edge between C_0 and R_0. This means that $N(C_0) \subseteq H_0$, from which condition (a) holds. □

Lemma 5. *The partition (R_0, H_0, C_0) meets the condition (b).*

Proof: Lemma 8 of Week 01 Lecture Note. □

Lemma 6. *The partition (R_0, H_0, C_0) meets the condition (c).*

Proof: Recall that $y_v^* = 0.5$ for all $v \in R_0$ is an optimal fractional solution to LPVC of $G[R_0]$, see Lemma 7 of Week 01 Lecture Note. As any vertex cover of a graph has size at least the objective value of an optimal fractional solution to LPVC, we conclude that the size of an optimal vertex cover of $G[R_0]$ is at least $\sum_{v \in R_0} y^* = 0.5|R_0|$. □

4 $3k$-vertex kernel for VERTEX COVER using Crown Decomposition

A *crown decomposition* of a graph $G = (V, E)$ is a partition (R, H, C) of V satisfying the following conditions.

(i) $H \neq \emptyset, C \neq \emptyset$,

(ii) C is an independent set,

(iii) $N(C) \subseteq H$ ("H separates C from R")

(iv) For every $H' \subseteq H$, it holds that $|H'| \leq |N(H') \cap C|$ ("Hall's condition holds from H toward C").

Hall's condition is important in the context of maximum matching / vertex cover in a bipartite graph. We recall the celebrated Hall's theorem. We say that a vertex v of G is saturated by a matching M if v is incident with some edge of M. A vertex set is saturated by a matching M if every vertex in the set is saturated by M.

Theorem 2 (Hall's theorem)**.** *Let G be a bipartite graph on the bipartition X and Y. There is a matching saturating X if and only if for every $X' \subseteq X$, we have $|X'| \leq |N(X') \cap Y|$.*

Theorem 3 (König's theorem)**.** *The size of an optimal vertex cover equals the size of a maximum matching on a bipartite graph.*

The proof of the next lemma is omitted as it is essentially the same proof as in Lecture Note for Week 01.

Lemma 7. *If (R, H, C) is a crown decomposition of G, then there exists an optimal vertex cover of G containing all vertices of H.*

We can consider the following reduction rule.

Reduction Rule 1. *If there is a a crown decomposition (R, H, C) of G, output (G', k') where $G' := G - (H \cup C)$ and $k' := k - |H|$.*

Lemma 8. *Reduction Rule 1 is safe.*

Proof: Suppose that (G, k) is a YES-instance. By Lemma 7, there exists a vertex cover X of size at most k which contain H entirely. Now, $X - H$ is a vertex cover of $G[R] = G - (H \cup C)$ of size at most $k - |H|$.

Conversely, if X' is a vertex cover of $G[R]$ of size at most $k - |H|$, let us consider the vertex set $X' \cup H$. Clearly its size is at most k and it is routine to verify that $X' \cup H$ is indeed a vertex cover of G. □

Theorem 4. *Let G be a graph without isolated vertices on at least $4k + 1$ vertices. In polynomial time, one can*

- *either find a matching of size at least $k + 1$,*
- *or find a crown decomposition*

Proof: Consider the algorithm 1 **VC-CROWN**(G, k). It suffices to prove that (R, H, C) at line 13 is a crown decomposition, and that deciding if there is a set H' as in the while-condition at line 8 and finding one if exists can be done in polynomial time.

First, we prove that (R, H, C) at line 13 is a crown decomposition. To begin with, we show that each partitions (R_i, H_i, C_i) obtained when performing the while-loop maintains the invariant (i)-(iii) of the crown decomposition, and additionally the following invariant $(\star)$.

$$(\star) \qquad |H_i| < |C_i|.$$

Consider (R_0, H_0, C_0). Note that $|H_0| \leq |V(M)| \leq 2k$ since otherwise the algorithm would have terminated at line 6. From $|C_0| = |V| - |V(M)| \geq 2k + 1$ (see line 3), we know that $|H_0| < |C_0|$, the invariant $(\star)$. This implies $C_0 \neq \emptyset$, which in turn implies that $H_0 = N(C_0) \neq \emptyset$ because no vertex of G is isolated, (see line 2). The invariant (ii) is satisfied because M is a maximal matching (line 5). The invariant (iii) holds by construction at line 7.

We show that (R_i, H_i, C_i) maintains the invariant (i)-(iii) and $(\star)$ for every $i \geq 0$ counted during the while-loop. This is the case for $i = 0$ and it suffices to establish the following.

> $(\diamond)$ If (R_i, H_i, C_i) satisfies (i)-(iii) and $(\star)$, then $(R_{i+1}, H_{i+1}, C_{i+1})$ satisfies (i)-(iii) and $(\star)$.

Regarding $(\star)$, note that

- $|H_i| < |C_i| \qquad \because (R_i, H_i, C_i)$ satisfies $(\star)$ by induction hypothesis,
- $|H'| > |N(H') \cap C_i| \qquad \because$ the condition of the while-loop at line 8,

which, combined with the construction of C_{i+1} and H_{i+1}, leads to $(\star)$. This also establishes (i) as well by the same argument as in the case of $i = 0$. That C_{i+1} is independent follows from $C_{i+1} \subseteq C_i$. To see (iii), suppose the contrary, i.e. C_{i+1} is adjacent with R_{i+1}. By induction hypothesis, C_{i+1} is adjacent with $R_{i+1} - R_i$, that is, with $H' \cup (N(H') \cap C_i)$. This is impossible because C_i is independent by the invariant (ii) for i and if C_{i+1} has a vertex adjacent with H', then such a vertex is included in $N(H') \cap C_i$ and should be moved to R_{i+1}. This establishes the claim $(\diamond)$.

Now, the partition (R, H, C) at line 13 has no vertex set $H' \subseteq H$ with $|H'| > |N(H') \cap C|$. That is, the partition at hand satisfies (iv) on top of (i)-(iii) and it is a crown decomposition.

Secondly, let us see that deciding if there is a set H' as in the while-condition at line 8 and finding one if exists can be done in polynomial time. We use the known polynomial-time algorithm (e.g. Hopcroft-Karp, there are many) for the MINIMUM VERTEX COVER problem on the bipartite graph $G[H_0, C_0]$, namely the bipartite graph obtained from the subgraph of G induced on $H_0 \cup C_0$ by removing all edges within H_0. Let X be an optimal vertex cover of $G[H_0, C_0]$.

If $H_0 \subseteq X$ (implying $H_0 = X$), by Theorem 3 there is a matching saturating H_0. This means that there is no such H' as in line 8 by Theorem 2. If $H_0 - X \neq \emptyset$ (i.e. these are vertices of H_0 not in the vertex cover X), note that $N(H_0 - X) \cap C_0$ must be all contained in the vertex cover X. Take $H' := H_0 - X$. If $|H'| \leq |N(H') \cap C_0|$, we can alternatively take a new vertex cover $X' := (X - N(H') \cap C_0) \cup H'$; note that $|X'| \leq |X|$ and it is an easy exercise to verify that X' is indeed a vertex cover of $G[H_0, C_0]$. Now $H_0 \subseteq X'$, and we know that there is no such H'. If $|H'| > |N(H') \cap C_0|$, we have found a vertex set H' as required at line 8. $\square$

One can improve the bound of Theorem 4 to $3k + 1$ instead of $4k + 1$. The proof can be found on page 26 of the textbook by Cygan et al. "Parameterized Algorithms."

Algorithm 1 Kernelization for VERTEX COVER using crown decomposition

procedure VC-CROWN(G, k)
    Remove all isolated vertices.
    if $|V| \leq 4k+1$ **then return** (G, k).
    else
        Let M be a maximal matching.
        if $|M| \geq k+1$ **then return** **NO**
        $C_0 \leftarrow V - V(M)$, $H_0 \leftarrow N(C_0)$, $R_0 \leftarrow V - (C_0, H_0)$, and $i \leftarrow 0$
        while there exists $H' \subseteq H_i$ with $|H'| > |N(H') \cap C_i|$ **do**
            $C_{i+1} \leftarrow C_i - N(H') \cap C_i$
            $H_{i+1} \leftarrow H_i - H'$
            $R_{i+1} \leftarrow R_i \cup H' \cup (N(H') \cap C_i)$
            $i \leftarrow i+1$
        Let $(R, H, C) = (R_i, H_i, C_i)$.
        Return **VC-CROWN**$(G - (H \cup C), k - |H|)$