2b. Kernel Lower Bounds

COMP6741: Parameterized and Exact Computation

Serge Gaspers$^{12}$

$^1$ School of Computer Science and Engineering, UNSW Sydney, Australia
$^2$ Decision Sciences, Data61, CSIRO, Australia

Semester 2, 2018
Outline

1. Introduction
2. Compositions
3. Polynomial Parameter Transformations
4. Further Reading
For some FPT problems, only exponential kernels are known. Could it be that all FPT problems have polynomial kernels? We will see that polynomial kernels for some fixed-parameter tractable parameterized problems would contradict complexity-theoretic assumptions.
Intuition by example

**Long Path**

**Input:** A graph $G = (V, E)$, and an integer $k \leq |V|$.

**Parameter:** $k$

**Question:** Does $G$ have a path of length at least $k$ (as a subgraph)?

*Long Path* is **NP-complete** but **FPT**.
Intuition by example

- Assume **Long Path** has a \( k^c \) kernel, where \( c = O(1) \).
- Set \( q = k^c + 1 \) and consider \( q \) instances with the same parameter \( k \):
  \[
  (G_1, k), (G_2, k), \ldots, (G_q, k).
  \]
- Let \( G = G_1 \oplus G_2 \oplus \cdots \oplus G_q \) be the disjoint union of all these graphs.
- Note that \( (G, k) \) is a **Yes**-instance if and only if at least one of \( (G_i, k), 1 \leq i \leq q \), is a **Yes**-instance.
- Kernelizing \( (G, k) \) gives an instance of size \( k^c \), i.e., on average less than one bit per original instance.
- “The kernelization must have solved at least one of the original NP-hard instances in polynomial time”.
- Note that this is not a rigorous argument, and we will make this more formal now.
Outline

1. Introduction
2. Compositions
3. Polynomial Parameter Transformations
4. Further Reading
Distillation

Definition 1

Let $\Pi_1, \Pi_2$ be two problems. An OR-distillation (resp., AND-distillation) from $\Pi_1$ into $\Pi_2$ is a polynomial time algorithm $D$ whose input is a sequence $I_1, \ldots, I_q$ of instances for $\Pi_1$ and whose output is an instance $I'$ for $\Pi_2$ such that

- $|I'| \leq \text{poly}(\max_{1 \leq i \leq q} |I_i|)$, and
- $I'$ is a $\text{YES}$-instance for $\Pi_2$ if and only if for at least one (resp., for each) $i \in \{1, \ldots, q\}$ we have that $I_i$ is a $\text{YES}$-instance for $\Pi_1$. 

NP-complete problems don’t have distillations

**Theorem 2** ([Fortnow, Santhanam, 2008])

*If any NP-complete problem has an OR-distillation, then coNP ⊆ NP/poly.*  

**Note:** coNP ⊆ NP/poly is not believed to be true and it would imply that the polynomial hierarchy collapses to the third level: PH ⊆ Σ₃^P.

**Theorem 3** ([Drucker, 2012])

*If any NP-complete problem has an AND-distillation, then coNP ⊆ NP/poly.*

---

^1NP/poly is the class of all decision problems for which there exists a polynomial-time nondeterministic Turing Machine \( M \) with the following property: for every \( n \geq 0 \), there is an advice string \( A \) of length \( \text{poly}(n) \) such that, for every input \( I \) of length \( n \), the machine \( M \) correctly decides the problem with input \( I \), given \( I \) and \( A \).
Composition algorithms

Definition 4

Let $\Pi$ be a parameterized problem. An OR-composition (resp., AND-composition) of $\Pi$ is a polynomial time algorithm $A$ that receives as input a finite sequence $I_1, \ldots, I_q$ of $\Pi$ with parameters $k_1 = \cdots = k_q = k$ and outputs an instance $I'$ for $\Pi$ with parameter $k'$ such that

- $k' \leq \text{poly}(k)$, and
- $I'$ is a \textbf{YES}-instance for $\Pi$ if and only if for at least one (resp., for each) $i \in \{1, \ldots, q\}$, $I_i$ is a \textbf{YES}-instance for $\Pi$. 
Theorem 5 (Composition Theorem)

Let $\Pi$ be an $\text{NP}$-complete parameterized problem such that for each instance $I$ of $\Pi$ with parameter $k$, the value of the parameter $k$ can be computed in polynomial time and $k \leq |I|$. If $\Pi$ has an OR-composition or an AND-composition, then $\Pi$ has no polynomial kernel, unless $\text{coNP} \subseteq \text{NP}/\text{poly}$.
Theorem 5 (Composition Theorem)

Let $\Pi$ be an NP-complete parameterized problem such that for each instance $I$ of $\Pi$ with parameter $k$, the value of the parameter $k$ can be computed in polynomial time and $k \leq |I|$. If $\Pi$ has an OR-composition or an AND-composition, then $\Pi$ has no polynomial kernel, unless coNP $\subseteq$ NP/poly.

Proof sketch.

Suppose $\Pi$ has an OR/AND-composition and a polynomial kernel. Then, one can obtain an OR/AND-distillation from $\Pi$ into OR($\Pi$)/AND($\Pi$).
Tool for showing kernel lower bounds

**Theorem 5 (Composition Theorem)**

Let \( \Pi \) be an \( \text{NP} \)-complete parameterized problem such that for each instance \( I \) of \( \Pi \) with parameter \( k \), the value of the parameter \( k \) can be computed in polynomial time and \( k \leq |I| \). If \( \Pi \) has an OR-composition or an AND-composition, then \( \Pi \) has no polynomial kernel, unless \( \text{coNP} \subseteq \text{NP/poly} \).

**Proof sketch.**

Suppose \( \Pi \) has an OR/AND-composition and a polynomial kernel. Then, one can obtain an OR/AND-distillation from \( \Pi \) into \( \text{OR}(\Pi)/\text{AND}(\Pi) \).

\[
I_1 \quad I_2 \quad \ldots \quad I_q \quad q \text{ instances of size } \leq n = \max_{1 \leq i \leq q} |I_i|
\]

\[
\{I_i : k_i = 0\} \ldots \{I_i : k_i = n\} \quad \text{group by parameter}
\]

\[
I'_0 \quad I'_1 \quad \ldots \quad I'_n \quad \text{After OR-composition: } n + 1 \text{ instances with } k'_i \leq \text{poly}(n)
\]

\[
I''_0 \quad I''_1 \quad \ldots \quad I''_n \quad \text{After kernelization: } n + 1 \text{ instances of size } \text{poly}(n) \text{ each}
\]

This is an instance of \( \text{OR}(\Pi) \) of size \( \text{poly}(n) \).
**Theorem 6**

*Long Path* has no polynomial kernel unless \( \text{NP} \subseteq \text{coNP/poly} \).

**Proof.**

Clearly, \( k \) can be computed in polynomial time and \( k \leq |V| \).

We give an OR-composition for *Long Path*, which will prove the theorem by the previous lemma.

It receives as input a sequence of instances for *Long Path*: \((G_1, k), \ldots, (G_q, k)\), and it produces the instance \((G_1 \oplus \cdots \oplus G_q, k)\), which is a **Yes**-instance if and only if at least one of \((G_1, k), \ldots, (G_q, k)\) is a **Yes**-instance.
\textbf{var-SAT} has no poly kernel I

\begin{tabular}{|l|}
\hline
\textbf{var-SAT} \\
\hline
\textbf{Input:} \hspace{1cm} A propositional formula $F$ in conjunctive normal form (CNF) \\
\textbf{Parameter:} \hspace{0.5cm} $n = |\text{var}(F)|$, the number of variables in $F$ \\
\textbf{Question:} \hspace{0.5cm} Is there an assignment to $\text{var}(F)$ satisfying all clauses of $F$? \\
\hline
\end{tabular}

**Example:**

\[(x_1 \lor x_2) \land (\neg x_2 \lor x_3 \lor \neg x_4) \land (x_1 \lor x_4) \land (\neg x_1 \lor \neg x_3 \lor \neg x_4)\]

or

\[
\{\{x_1, x_2\}, \{\neg x_2, x_3, \neg x_4\}, \{x_1, x_4\}, \{\neg x_1, \neg x_3, \neg x_4\}\}
\]
Theorem 7

*var-SAT has no polynomial kernel unless* $\text{NP} \subseteq \text{coNP}/\text{poly}$.

Proof.

Clearly, $\text{var}(F)$ can be computed in polynomial time and $n = |\text{var}(F)| \leq |F|$. We give an OR-composition for var-SAT, which will prove the theorem by the previous lemma.

- Let $F_1, \ldots, F_q$ be CNF formulas, $|F_i| \leq m$, $|\text{var}(F_i)| = n$.
- We can decide whether one of the formulas is satisfiable in time $\text{poly}(mt2^n)$. Hence, if $q > 2^n$, the check is polynomial. If some formula is satisfiable, we output this formula, otherwise we output $F_1$. 

S. Gaspers (UNSW) 
Kernel Lower Bounds
Semester 2, 2018 13 / 23
Proof (continued).

- It remains the case $q \leq 2^n$. We assume $\text{var}(F_1) = \cdots = \text{var}(F_q)$, otherwise we change the names of variables.
- Let $s = \lceil \log_2 q \rceil$. Since $q \leq 2^n$, we have that $s \leq n$.
- We take a set $Y = \{y_1, \ldots, y_s\}$ of new variables. Let $C_1, \ldots, C_{2^s}$ be the sequence of all $2^s$ possible clauses containing exactly $s$ literals over the variables in $Y$.
- For $1 \leq i \leq q$ we let $F'_i = \{C \cup C_i : C \in F_i\}$.
- We define $F = \bigcup_{i=1}^{q} F'_i \cup \{C_i : q + 1 \leq i \leq 2^s\}$.
- Claim: $F$ is satisfiable if and only if $F_i$ is satisfiable for some $1 \leq i \leq q$.
- Hence we have an OR-composition.
Outline

1. Introduction

2. Compositions

3. Polynomial Parameter Transformations

4. Further Reading
Another tool for showing kernel lower bounds I

Definition 8

Let $\Pi_1, \Pi_2$ be parameterized problems. A polynomial parameter transformation from $\Pi_1$ to $\Pi_2$ is a polynomial time algorithm, which, for any instance $I_1$ of $\Pi_1$ with parameter $k_1$, produces an equivalent instance $I_2$ of $\Pi_2$ with parameter $k_2$ such that $k_2 \leq \text{poly}(k_1)$. 
Theorem 9

Let $\Pi_1, \Pi_2$ be parameterized problems such that $\Pi_1$ is $\text{NP}$-complete, $\Pi_2$ is in $\text{NP}$, and there is a polynomial parameter transformation from $\Pi_1$ to $\Pi_2$. If $\Pi_2$ has a polynomial kernel, then $\Pi_1$ has a polynomial kernel.

Remark: If we know that an $\text{NP}$-complete parameterized problem $\Pi_1$ has no polynomial kernel (unless $\text{NP} \subseteq \text{coNP/poly}$), we can use the theorem to show that some other $\text{NP}$-complete parameterized problem $\Pi_2$ has no polynomial kernel (unless $\text{NP} \subseteq \text{coNP/poly}$) by giving a polynomial parameter transformation from $\Pi_1$ to $\Pi_2$. 
Proof.

- We show that under the assumptions of the theorem $\Pi_1$ has a polynomial kernel.
- Let $I_1$ be an instance of $\Pi_1$ with parameter $k_1$.
- We obtain in polynomial time an equivalent instance $I_2$ of $\Pi_2$ with parameter $k_2 \leq \text{poly}(k_1)$.
- We apply $\Pi_2$’s kernelization and obtain $I_2'$ of size $\leq \text{poly}(k_1)$.
- Since $\Pi_2$ is in NP and $\Pi_1$ is NP-complete, there exists a polynomial time reduction that maps $I_2'$ to an equivalent instance $I_1'$ of $\Pi_1$.
- The size of $I_1'$ is polynomial in $k_1$.
Definition 10

A CNF formula $F$ is a 2CNF formula if each clause of $F$ has at most 2 literals.

Note: SAT is polynomial time solvable when the input is restricted to be a 2CNF formula.

Definition 11

A 2CNF-backdoor of a CNF formula $F$ is a set of variables $B \subseteq \text{var}(F)$ such that for each assignment $\alpha : B \rightarrow \{0, 1\}$, the formula $F[\alpha]$ is a 2CNF formula. Here, $F[\alpha]$ is obtained by removing all clauses containing a literal set to 1 by $\alpha$, and removing the literals set to 0 from all remaining clauses.
Input: A CNF formula $F$ and a 2CNF-backdoor $B$ of $F$
Parameter: $k = |B|$
Question: Is $F$ satisfiable?

**Note:** the problem is **FPT** by trying all assignments to $B$ and evaluating the resulting formulas.
Theorem 12

**Theorem**

2CNF-Backdoor Evaluation $\textit{has no polynomial kernel unless}$ $\text{NP} \subseteq \text{coNP/poly}$.

**Proof.**

We give a polynomial parameter transformation from var-SAT to 2CNF-Backdoor Evaluation. Let $F$ be an instance for var-SAT. Then, $(F, B = \text{var}(F))$ is an equivalent instance for 2CNF-Backdoor Evaluation with $|B| \leq |\text{var}(F)|$. 

\[\square\]
Outline

1. Introduction
2. Compositions
3. Polynomial Parameter Transformations
4. Further Reading
Further Reading