Brief Introduction to Parameterized Algorithms

Nicolas Nisse

Université Côte d’Azur, Inria, CNRS, I3S, France

TADaaM seminar, October 12th, 2021
How to face NP-hard problems? (but not only)

- Exact exponential algorithms \( O(c^n), c > 1 \)
- Approximation algorithms, heuristics not always possible/desired.
- Particular instances e.g., bipartite graphs, planar graphs...
- ...
- Parameterized Complexity: “Refined” analysis of the complexity depending not only on the size of the instance but on some “parameter” describing the “structure” of the instance.

Two very nice books:

Brief introduction to parameterized algorithms
How to face NP-hard problems? (but not only)

- Exact exponential algorithms \( O(c^n), c > 1 \)
- Approximation algorithms, heuristics not always possible/desired.
- Particular instances e.g., bipartite graphs, planar graphs...
- ...

**Parameterized Complexity:** “Refined” analysis of the complexity depending not only on the size of the instance but on some “parameter” describing the “structure” of the instance.

Two very nice books:

M. Cygan, F.V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh:


F.V. Fomin, D. Lokshtanov, S. Saurabh, M. Zehavi:

Parameterized Complexity in a nutshell
Some well known NP-hard problems

**Vertex Cover:**
- **Inputs:** a graph $G = (V, E)$, and $k \in \mathbb{N}$;
- **Output:** does there exist $Q \subseteq V$, $|Q| \leq k$, $\forall e \in E$, $Q \cap e \neq \emptyset$?

  best known $O(1.2114^n)$ [Bourgeois et al., 12], 2-approximation (maximal matching)

**Clique:**
- **Inputs:** a graph $G = (V, E)$, and $k \in \mathbb{N}$;
- **Output:** does there exist $Q \subseteq V$, $|Q| \geq k$, $\forall u, v \in Q$, $\{u, v\} \in E$?

  best known $O(1.1888^n)$ [Robson et al., 01], $O(n(\log\log n)^2/\log^3 n)$-approximation [Feige 04]

**Proper Coloring:**
- **Inputs:** a graph $G = (V, E)$, and $k \in \mathbb{N}$;
- **Output:** $\chi(G) \leq k$?, i.e., is there a proper coloring $c : V \rightarrow \{1, \ldots, k\}$?
Some well known NP-hard problems

What if the size $k$ of the solution is a fixed parameter?

**k-VERTEX COVER:**

- **Input:** a graph $G = (V, E)$;
- **Output:** does there exist $Q \subseteq V$, $|Q| \leq k$, $\forall e \in E$, $Q \cap e \neq \emptyset$?

**k-CLIQUE:**

- **Input:** a graph $G = (V, E)$;
- **Output:** does there exist $Q \subseteq V$, $|Q| \geq k$, $\forall u, v \in Q$, $\{u, v\} \in E$?

Polynomial-time solvable!: try $n^{O(k)}$ possibilities.

**k-PROPER COLORING:**

- **Input:** a graph $G = (V, E)$;
- **Output:** $\chi(G) \leq k$?, i.e., is there a proper coloring $c : V \rightarrow \{1, \ldots, k\}$?

Still NP-hard for any fixed $k \geq 3$! (constant-time in planar graphs if $k \geq 4$)
Parameter

A parameter is given by a polynomial-time computable function, which maps instances of our problem to natural numbers.

e.g., size of the solution, genus, treewidth, number of vertices to be removed to obtain a bipartite graph, etc.

Class XP

A problem is in \textit{XP} parameterized by \( k \) if there exists an algorithm which solves the problem in time \( O(n^{f(k)}) \) for some function \( f \).

If the parameter \( k \) is the size of the solution:

- \textit{k-Vertex-Cover} and \textit{k-Clique} are in \textit{XP}.
- \textit{k-Coloring} \( \not\in \text{XP} \) for any \( k \geq 3 \).
Let’s go further: example of $k$-Vertex-Cover

**Vertex Cover** of $G = (V, E)$: set $Q \subseteq V$ s.t., $\forall e \in E$, $e \cap Q \neq \emptyset$.

**Trivial lemmas:** let $Q$ be a Vertex Cover of $G$, then
- for all $\{u, v\} \in E$, $u \in Q$ or $v \in Q$ or both.
- if $|Q| \leq k$, then $|E| \leq k|V|$.
Let’s go further: example of $k$-Vertex-Cover

**Vertex Cover** of $G = (V, E)$: set $Q \subseteq V$ s.t., $\forall e \in E$, $e \cap Q \neq \emptyset$.

**Trivial lemmas:** let $Q$ be a Vertex Cover of $G$, then
- for all $\{u, v\} \in E$, $u \in Q$ or $v \in Q$ or both.
- if $|Q| \leq k$, then $|E| \leq k|V|$.

**Branch & Bound Algorithm (BB) for deciding if $vc(G) \leq k$**

If $|E| > 0$ and $k = 0$, Return $\infty$. Else if $|E| = 0$, Return 0. Else if $|E| = 1$, Return 1. Else Let $\{u, v\} \in E$, Return $\min\{BB(G \setminus u, k - 1), BB(G \setminus v, k - 1)\} + 1$

**Limited recursion depth + bounded # of edges ⇒ Complexity:** $O(2^k \cdot |V|)$

linear in $|V|$, $|V|$ and $k$ are “separated”
A problem is fixed parameter tractable (FPT) parameterized by a parameter $k$ if there exists an algorithm which solves the problem in time $f(k) \cdot n^{O(1)}$. 

**Vertex-Cover** is FPT parameterized by the size of the solution.

Which problems are FPT?
A problem is fixed parameter tractable (FPT) parameterized by a parameter $k$ if there exists an algorithm which solves the problem in time $f(k) \cdot n^{O(1)}$.

**Vertex-Cover** is FPT parameterized by the size of the solution.

Which problems are FPT?

### The W-Hierarchy

$\mathsf{FPT} \subseteq \mathsf{W[1]} \subseteq \mathsf{W[2]} \subseteq \cdots$

$\mathsf{W[\ldots]}$ defined using \textit{weft} of Boolean circuits

It is strongly believed that $\mathsf{FPT} \neq \mathsf{W[1]}$.

- $k$-\textsc{Clique} is $\mathsf{W[1]}$-complete
- $k$-\textsc{Dominating Set}, $k$-\textsc{Set Cover}, $k$-\textsc{Hitting Set} are $\mathsf{W[2]}$-complete.
  
  but... $k$-\textsc{Clique} FPT in planar graphs
Let’s go even further: example of $k$-\textsc{Vertex-Cover}

**Simple lemmas:** let $Q$ be a Vertex Cover of $G = (V, E)$ of size $\leq k$, then
- if $v \in V$ has degree $> k$, then $v \in Q$;
- if no vertex with degree $> k$, then $|E| \leq k^2$. 

**Kernelization Algorithm (Ker) for deciding if $vc(G) \leq k$**
- Remove isolated vertices
- If $|E| = 0$, Return TRUE.
- Else if $k = 0$, Return FALSE.
- Else if no vertex of degree $> k$ and $|E| > k^2$, Return FALSE.
- Else if $v$ is a vertex of degree $> k$. Return Ker ($G \setminus v$, $k - 1$).
- Else Return BB ($G$, $k$).

**Complexity:** $O(2^k k^3 + |V|^2)$ polynomial-time “data-reduction” to an instance of order $k^2$. 

N. Nisse

Brief introduction to parameterized algorithms
Let’s go even further: example of $k$-Vertex-Cover

**Simple lemmas:** let $Q$ be a Vertex Cover of $G = (V, E)$ of size $\leq k$, then
- if $v \in V$ has degree $> k$, then $v \in Q$;
- if no vertex with degree $> k$, then $|E| \leq k^2$.

**Kernelization Algorithm (Ker) for deciding if $vc(G) \leq k$**

Remove isolated vertices
If $|E| = 0$, Return $TRUE$.
Else if $k = 0$, Return $FALSE$
Else if no vertex of degree $> k$ and $|E| > k^2$, Return $FALSE$
Else if $v$ is a vertex of degree $> k$. Return $Ker(G \setminus v, k - 1)$.
Else Return $BB(G, k)$

Complexity: $O(2^k k^3 + |V|^2)$

polynomial-time “data-reduction” to an instance of order $k^2$. 
Kernelization

A kernelization for a parameterized problem \( \Pi \) is an algorithm that takes an instance \((x, k)\) and maps it in time polynomial in \(|x|\) and \(k\) to an instance \((x', k')\) s.t.

- \((x, k) \in \Pi \iff (x', k') \in \Pi\),
- \(k' + |x'| \leq g(k)\) where \(g\) is a function called the size of the kernel.

“Preprocessing” algorithm that reduces to an instance of size \(\leq\) a function only of \(k\).
Kernelization

A kernelization for a parameterized problem $\Pi$ is an algorithm that takes an instance $(x, k)$ and maps it in time polynomial in $|x|$ and $k$ to an instance $(x', k')$ s.t.

- $(x, k) \in \Pi \iff (x', k') \in \Pi$,
- $k' + |x'| \leq g(k)$ where $g$ is a function called the size of the kernel.

“Preprocessing” algorithm that reduces to an instance of size $\leq$ a function only of $k$.

Kernelization $\iff$ FPT

$\Rightarrow n^{O(1)}$ “data-reduction” + brute force on instance of size dependent only on $k$

$\Leftarrow$. Assume there exists an algorithm solving the problem in time $f(k)n^{O(1)}$.

- if $n \leq f(k)$, nothing to be done;
- else, $f(k)n^c \leq n^{c+1}$ and the algorithm is polynomial in $n$. Apply it and return a trivial Yes or No instance.
Kernelization vs. FPT

Kernelization

A **kernelization** for a parameterized problem $\Pi$ is an algorithm that takes an instance $(x, k)$ and maps it in time polynomial in $|x|$ and $k$ to an instance $(x', k')$ s.t.

- $(x, k) \in \Pi \iff (x', k') \in \Pi$,
- $k' + |x'| \leq g(k)$ where $g$ is a function called the size of the **kernel**.

"Preprocessing" algorithm that reduces to an instance of size $\leq$ a function only of $k$.

Kernelization $\iff$ FPT

$\Rightarrow n^{O(1)}$ “data-reduction” + brute force on instance of size dependent only on $k$

$\Leftarrow$. Assume there exists an algorithm solving the problem in time $f(k)n^{O(1)}$.

- if $n \leq f(k)$, nothing to be done;
- else, $f(k)n^c \leq n^{c+1}$ and the algorithm is polynomial in $n$. Apply it and return a trivial Yes or No instance.

Can we always get a kernel of polynomial size?

- **OK** for $k$-Vertex-Cover, $k$-Feedback Vertex-Set, $k$-Planar Dominating Set...
- **NOK** for $k$-Path...

(method of compositionability)
Fractional relaxation (LP) for Vertex Cover:

\[
\begin{align*}
\text{Min.} & \quad \sum_{v \in V} x_v \\
\text{s.t.:} & \quad x_v + x_u \geq 1 \quad \forall \{u, v\} \in E \\
& \quad x_v \geq 0 \quad \forall v \in V
\end{align*}
\]

Theorem: Optimal solution: \((x_v)_{v \in V}\)

Let \(V_1 = \{v \in V \mid x_v > 1/2\}\) and \(V_{1/2} = \{v \in V \mid x_v = 1/2\}\).
Then, \(\exists\) optimal (Integral) Vertex-Cover \(Q\) such that \(V_1 \subseteq Q \subseteq V_1 \cup V_{1/2}\)
Linear Kernel for \textit{k-Vertex-Cover}

Fractional relaxation (\textit{LP}) for Vertex Cover:

\begin{align*}
\text{Min.} \quad & \sum_{v \in V} x_v \\
\text{s.t.:} \quad & x_v + x_u \geq 1 \quad \forall \{u, v\} \in E \\
& x_v \geq 0 \quad \forall v \in V
\end{align*}

Theorem: Optimal solution: \((x_v)_{v \in V}\)

Let \(V_1 = \{v \in V \mid x_v > 1/2\}\) and \(V_{1/2} = \{v \in V \mid x_v = 1/2\}\).
Then, \(\exists\) optimal (Integral) Vertex-Cover \(Q\) such that \(V_1 \subseteq Q \subseteq V_1 \cup V_{1/2}\)

Linear Kernel (\textit{LK}) for deciding if \(vc(G) \leq k\)

\textbf{If} \(|E| = 0\), \textit{Return TRUE}
\textbf{Remove isolated vertices}
\textbf{Let} \((x_v)_{v \in V}\) \textbf{be an optimal solution obtained by LP}
\textbf{If} optimal fractional solution \(> k\), \textit{Return FALSE}
\textbf{Else} let \(V_1 = \{v \in V \mid x_v > 1/2\}\).
\textbf{If} \(V_1 \neq \emptyset\) then \textit{Return LK}(\(G \setminus V_1\), \(k - |V_1|\)).
\textbf{Else} \textit{Return BB}(\(G\), \(k\))

When \(V_1 = \emptyset\), \(x_v = 1/2\) for all \(v \in V\), and so \(|V| \leq 2k\) \hspace{1cm} \text{kernel of linear size.}
A bit on scheduling
A bit of Scheduling from parameterized point of view

“Surprisingly”, scheduling problems have received “few” attention in the context of parameterized complexity until the last 5 years.

“Reason”: “Many numerical input data, which alone render many problems NP-hard” [Mnich, Wiese 15]

Some known results on Makespan minimization... according to different parameters

- on identical machines: FPT parameterized by the size of the solution [Alon et al. 98]
- without preemption: FPT parameterized by the maximum processing time of a job [Mnich, Wiese 15]
- unrelated machines: FPT parameterized by the # of distinct processing times and the # of machines [Mnich, Wiese 15]
- unrelated machines: FPT parameterized by the treewidth of the primal graph [Jansen, Maack, Solis-Oba 20]
- ...

N. Nisse

Brief introduction to parameterized algorithms
Efficient Polynomial Time Approximation Scheme (EPTAS)

For all $\epsilon > 1$, $\epsilon$-approximation algorithm running in time $f\left(\frac{1}{\epsilon}\right)n^{O(1)}$.

Theorem: EPTAS $\Rightarrow$ FPT parameterized by the size of the solution [Bazgan 95]

Proof for minimization problem:

Apply the EPTAS for $\epsilon = \frac{1}{k+1}$.

If the obtained solution has value $\leq k$, then $TRUE$.

Else, $OPT \geq Value(Sol)/(1 + \frac{1}{k+1}) \geq (k + 1)/(1 + \frac{1}{k+1}) > k$, then $FALSE$. 
A simple example: width of dependency DAG

**Inputs:** set $T$ of $n$ tasks, unit length, with individual deadlines $d(t)$ and precedence constraints (partial order $P$).

**Output:** Is there a schedule of $T$ with at most $k$ late task?

**Theorem:** above problem is $W[1]$-hard parameterized by $k$ [Fellows, McCartin 03]

Compute (in time $O(n^2, 5)$) a partition of $P$ into $w(P)$ chains [Brightwell 94];

If a maximal element $e$ has deadline $\geq n$, put it in last position and recurse on $T\{e\};$

Else branch on the at most $w(P)$ maximal elements (any maximal element put in last position will be late).
A simple example: width of dependency DAG

Inputs: set $T$ of $n$ tasks, unit length, with individual deadlines $d(t)$ and precedence constraints (partial order $P$).

Output: Is there a schedule of $T$ with at most $k$ late task?

Theorem: above problem is $W[1]$-hard parameterized by $k$  
[Fellows,McCartin 03]

width of $P$: $w(P) = \min$. # of chains forming a partition of $P$

Theorem: above problem is FPT in $w(P)$ and $k$  
[Fellows,McCartin 03]

Compute (in time $O(n^{2.5})$) a partition of $P$ into $w(P)$ chains [Brightwell 94];

- If a maximal element $e$ has deadline $\geq n$, put it in last position and recurse on $T \setminus \{e\}$;
- Else branch on the at most $w(P)$ maximal elements
  (any maximal element put in last position will be late).

Limited recursion depth $\Rightarrow$ Complexity $O(w(P)^{k+1}n + n^{2.5})$. 
Powerfull Meta-Theorems for FPT algorithms
**Minor Graph Theorem**  
*Robertson, Seymour 04*

Every minor-closed property is recognizable in time $O(n^3)$ time.

(Finite number of minimal obstructions and $O(f(|V(H)|)n^3)$ algorithm for deciding if an $n$-node graph admits $H$ as minor.)

*(but very, very,... huge constants and non-constructive algorithm)*

**Courcelle’s Theorem**  
*Courcelle 90*

Every graph property definable in the monadic second-order logic of graphs can be decided in linear time on graphs of bounded treewidth.

**Bi-dimensionnality theory**  
*Demaine,Hajiaghayi 08*

FPT and EPTAS algorithms in bounded genus graphs for many problems.

*(Based on the Grid-Minor Theorem and on Courcelle’s Theorem.)*

**Meta-Kernelization**  
*Bodlaender et al. 16*

Linear Kernels for huge family of graph’s problems.
Conclusion

FPT in P: Not only NP-hard problems are concerned, e.g.:

- Radius and Diameter can be solved in $2^{O(k \log k)} n^{1+O(1)}$-time, where $k$ is treewidth. [Abboud, Vassilevska Williams, Wang 16] (sub-cubic algorithms are unexpected in general graphs)

- Maximum Matching can be solved in $O(k^4 n + m)$-time where $k$ is either the modular-width or the $P_4$-sparseness. [Coudert, Ducoffe, Popa 19]

Scheduling, many open problems:


Bring Parameterized algorithms to practice

E.g., last three Meta-theorems of previous slide are based on the computation of “good” tree-decompositions.

Computing treewidth: NP-hard, not-approximable, FPT (but not practical), $O(2^{O(k)} n)$ algorithm that decides if a graph has treewidth at most $5k + 4$.

Practical exact or approximation algorithms?
FPT in P: Not only NP-hard problems are concerned, e.g.:

- Radius and Diameter can be solved in $2^{O(k \log k)} n^{1+O(1)}$-time, where $k$ is treewidth.  
  [Abboud, Vassilevska Williams, Wang 16]  
  (sub-cubic algorithms are unexpected in general graphs)

- Maximum Matching can be solved in $O(k^4 n + m)$-time where $k$ is either the modular-width or the $P_4$-sparseness.  
  [Coudert, Ducoffe, Popa 19]

Scheduling, many open problems:


Bring Parameterized algorithms to practice

E.g., last three Meta-theorems of previous slide are based on the computation of “good” tree-decompositions.

Computing treewidth: NP-hard, not-approximable, FPT (but not practical), $O(2^{O(k)} n)$ algorithm that decides if a graph has treewidth at most $5k + 4$.

Practical exact or approximation algorithms?

Merci!