Introduction to Parameterized Complexity

M. Pouly

Department of Informatics
University of Fribourg, Switzerland

Internal Seminar

June 2006
Introduction & Motivation
Parameterized Tractability
Parameterized Intractability
Conclusions

Outline

1. Introduction & Motivation
2. Parameterized Tractability
3. Parameterized Intractability
4. Conclusions
The scientist Dr. $\Theta$ has collected a number of data points that support a new theory. But some of these observations are in conflict. Naturally, Dr. $\Theta$ wants to know the minimum set of points that ”explain” the inconsistencies.
We model data points by dots and conflicts by lines.
We model data points by dots and conflicts by lines.

This is VERTEX COVER!
The Perspective of Complexity Theories

The traditional View:

Forget it!
VERTEX COVER is NPC.

"But my data set is rather small – 40 or 50 data points ..."

The parameterized View:

The best known algorithm has $O(kn + (4/3)^k k^2)$. This is a good practical algorithm for $k \leq 70$. 
The Perspective of Complexity Theories

The traditional View:

Forget it!
VERTEX COVER is NPC.

"But my data set is rather small – 40 or 50 data points ..."

The parameterized View:

The best known algorithm has $O(kn + (4/3)^k k^2)$. This is a good practical algorithm for $k \leq 70$. 

M. Pouly
Introduction to Parameterized Complexity
The Perspective of Complexity Theories

The traditional View:

Forget it!
VERTEX COVER is NPC.

"But my data set is rather small – 40 or 50 data points ..."

The parameterized View:

The best known algorithm has $O(kn + (4/3)^k k^2)$. This is a good practical algorithm for $k \leq 70$. 
NP-complete Problems:

- NPC suggests that exhaustive search is the only approach.
- NPC discourages the investment of effort.
- In practice, the size of most instances is naturally bounded.

Polynomial Problems:

- Is $O(x^{100})$ really practical?
- Are there algorithms with $O(x^n)$ for $n > 4$?
NP-complete Problems:

- NPC suggests that exhaustive search is the only approach.
- NPC discourages the investment of effort.
- In practice, the size of most instances is naturally bounded.

Polynomial Problems:

- Is $O(x^{100})$ really practical?
- Are there algorithms with $O(x^n)$ for $n > 4$?
Outline

1. Introduction & Motivation
2. Parameterized Tractability
3. Parameterized Intractability
4. Conclusions
Problem input: \((x, k) \in \Sigma^* \times \mathbb{N}\)

- \(k\) is called *parameter*
- *Parameterized language*: Yes instances \(L \subseteq \Sigma^* \times \mathbb{N}\)

For a fixed \(k\), we denote \(L_k\) the \(k\)-th slice of \(L\):

\[
L_k = \{(x, k) : (x, k) \in L\}
\]
Problem input: \((x, k) \in \Sigma^* \times \mathbb{N}\)

- \(k\) is called *parameter*
- *Parameterized language*: Yes instances \(L \subseteq \Sigma^* \times \mathbb{N}\)

For a fixed \(k\), we denote \(L_k\) the \(k\)-th slice of \(L\):

\[
L_k = \{(x, k) : (x, k) \in L\}
\]
Examples of Parameterized Problems

- Some problems belong to both theories.
  ~ Example: VERTEX COVER

- Some problems have a parameterized counterpart:
  ~ Example: WEIGHTED CNF SAT

  - Instance: Propositional formula $X$ in CNF
  - Parameter: $k$
  - Question: Does $X$ have a satisfying weight $k$ assignment?

- Other problems only exist in one of the two worlds.
Examples of Parameterized Problems

- Some problems belong to both theories.
  ~ Example: VERTEX COVER

- Some problems have a parameterized counterpart:
  ~ Example: WEIGHTED CNF SAT

  - **Instance**: Propositional formula $X$ in CNF
  - **Parameter**: $k$
  - **Question**: Does $X$ have a satifying weight $k$ assignment?

- Other problems only exist in one of the two worlds.
Examples of Parameterized Problems

- Some problems belong to both theories.
  - Example: VERTEX COVER

- Some problems have a parameterized counterpart:
  - Example: WEIGHTED CNF SAT
    - Instance: Propositional formula $X$ in CNF
    - Parameter: $k$
    - Question: Does $X$ have a satisfying weight $k$ assignment?

- Other problems only exist in one of the two worlds.
Fixed Parameter Tractability I

We are interested in languages that are tractable by slice:

Definition (FPT = Fixed Parameter Tractability)

A parameterized problem \( L \subseteq \Sigma^* \times \mathbb{N} \) is fixed parameter tractable if there is an algorithm that correctly decides, for input \((x, k) \in \Sigma^* \times \mathbb{N}\) whether \((x, k) \in L\) in time \(f(k)|x|^\alpha\), where \(\alpha\) is constant and \(f\) is an arbitrary function.

- The relation between good and bad part is defined to be multiplicative. A fundamental property of FPT is that the definition is unchanged if we replace it by \(f(k) + |x|^{\alpha'}\).
- \(\text{VERTEX COVER} \in \text{FPT}\)
Fixed Parameter Tractability I

We are interested in languages that are tractable by slice:

---

**Definition (FPT = Fixed Parameter Tractability)**

A parameterized problem \( L \subseteq \Sigma^* \times \mathbb{N} \) is fixed parameter tractable if there is an algorithm that correctly decides, for input \((x, k) \in \Sigma^* \times \mathbb{N}\) whether \((x, k) \in L\) in time \(f(k)|x|^\alpha\), where \(\alpha\) is constant and \(f\) is an arbitrary function.

- The relation between good and bad part is defined to be multiplicative. A fundamental property of FPT is that the definition is unchanged if we replace it by \(f(k) + |x|^\alpha'\).

- VERTEX COVER \(\in\) FPT
**Definition**

Let $A$ be a parameterized problem.

a) We say that $A$ is **uniformly fixed-parameter tractable** if there is an algorithm $\Phi$, a constant $c$, and an arbitrary function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that
- the running time of $\Phi(x, k)$ is at most $f(k)|x|^c$,
- $(x, k) \in A$ iff $\Phi(x, k) = 1$.

b) We say that $A$ is **strongly uniformly fixed-parameter tractable** if $A$ is uniformly fixed-parameter tractable via some $\Phi$ and $f$ such that $f$ is recursive.

c) We say that $A$ is **nonuniformly fixed-parameter tractable** if there is a constant $c$, a function $f : \mathbb{N} \rightarrow \mathbb{N}$ and a collection of procedures $\{\Phi_k : k \in \mathbb{N}\}$ such that for each $k$ the running time of $\Phi_k$ is at most $f(k)|x|^c$ and $(x, k) \in A$ iff $\Phi_k(x, k) = 1$. 

M. Pouly

Introduction to Parameterized Complexity
Definition

Let $A$ be a parameterized problem.

a) We say that $A$ is *uniformly fixed-parameter tractable* if there is an algorithm $\Phi$, a constant $c$, and an arbitrary function $f : \mathbb{N} \to \mathbb{N}$ such that

1. the running time of $\Phi(x, k)$ is at most $f(k)|x|^c$,
2. $(x, k) \in A$ iff $\Phi(x, k) = 1$.

b) We say that $A$ is *strongly uniformly fixed-parameter tractable* if $A$ is uniformly fixed-parameter tractable via some $\Phi$ and $f$ such that $f$ is recursive.

c) We say that $A$ is *nonuniformly fixed-parameter tractable* if there is a constant $c$, a function $f : \mathbb{N} \to \mathbb{N}$ and a collection of procedures $\{\Phi_k : k \in \mathbb{N}\}$ such that for each $k$ the running time of $\Phi_k$ is at most $f(k)|x|^c$ and $(x, k) \in A$ iff $\Phi_k(x, k) = 1$. 

M. Pouly
Introduction to Parameterized Complexity
Fixed Parameter Tractability II

Definition

Let $A$ be a parameterized problem.

a) We say that $A$ is *uniformly fixed-parameter tractable* if there is an algorithm $\Phi$, a constant $c$, and an arbitrary function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that
   - the running time of $\Phi(x, k)$ is at most $f(k)|x|^c$,
   - $(x, k) \in A$ iff $\Phi(x, k) = 1$.

b) We say that $A$ is *strongly uniformly fixed-parameter tractable* if $A$ is uniformly fixed-parameter tractable via some $\Phi$ and $f$ such that $f$ is recursive.

c) We say that $A$ is *nonuniformly fixed-parameter tractable* if there is a constant $c$, a function $f : \mathbb{N} \rightarrow \mathbb{N}$ and a collection of procedures $\{\Phi_k : k \in \mathbb{N}\}$ such that for each $k$ the running time of $\Phi_k$ is at most $f(k)|x|^c$ and $(x, k) \in A$ iff $\Phi_k(x, k) = 1$. 
How Parameters arise in Practice

- **Natural Parameters**: Frequent in graph theory:
  - *Instance*: A graph $G$ and a positive integer $k$
  - *Parameter*: $k$
  - *Question*: Is $\omega(G) \leq k$ ?

$\omega(G)$ is a width metric, i.e. Treewidth, Cutwidth, . . .

- **Implicit Parameters**: Frequent in formula problems:
  - *Instance*: A database $d$ and a query $\phi$.
  - *Parameter*: $|\phi|$  
  - *Question*: Is the query relation defined by $\phi$ nonempty ?

- **Engineering Parameters**: Key length in cryptology.

- **Approximation Parameters**: Related to the $\epsilon$. 

M. Pouly

Introduction to Parameterized Complexity
How Parameters arise in Practice

- **Natural Parameters**: Frequent in graph theory:
  - *Instance*: A graph $G$ and a positive integer $k$
  - *Parameter*: $k$
  - *Question*: Is $\omega(G) \leq k$?
  
  $\omega(G)$ is a width metric, i.e. Treewidth, Cutwidth, . . .

- **Implicit Parameters**: Frequent in formula problems:
  - *Instance*: A database $d$ and a query $\phi$.
  - *Parameter*: $|\phi|$
  - *Question*: Is the query relation defined by $\phi$ nonempty?

- **Engineering Parameters**: Key length in cryptology.

- **Approximation Parameters**: Related to the $\epsilon$. 

M. Pouly

Introduction to Parameterized Complexity
How Parameters arise in Practice

- **Natural Parameters**: Frequent in graph theory:
  - *Instance*: A graph $G$ and a positive integer $k$
  - *Parameter*: $k$
  - *Question*: Is $\omega(G) \leq k$?
  
  $\omega(G)$ is a width metric, i.e. Treewidth, Cutwidth, . . .

- **Implicit Parameters**: Frequent in formula problems:
  - *Instance*: A database $d$ and a query $\phi$.
  - *Parameter*: $|\phi|$
  - *Question*: Is the query relation defined by $\phi$ nonempty?

- **Engineering Parameters**: Key length in cryptology.

- **Approximation Parameters**: Related to the $\epsilon$. 

M. Pouly
Introduction to Parameterized Complexity
How Parameters arise in Practice

- **Natural Parameters**: Frequent in graph theory:
  - *Instance*: A graph $G$ and a positive integer $k$
  - *Parameter*: $k$
  - *Question*: Is $\omega(G) \leq k$?

  $\omega(G)$ is a width metric, i.e. Treewidth, Cutwidth, . . .

- **Implicit Parameters**: Frequent in formula problems:
  - *Instance*: A database $d$ and a query $\phi$.
  - *Parameter*: $|\phi|$
  - *Question*: Is the query relation defined by $\phi$ nonempty?

- **Engineering Parameters**: Key length in cryptology.

- **Approximation Parameters**: Related to the $\epsilon$. 

M. Pouly

Introduction to Parameterized Complexity
Which $k$ are reasonable?

The are currently no theorems that indicate limits on what we might reasonably expect to achieve in terms of practical parameter ranges for FPT problems.

However, the goodness of a FPT algorithm can be measured by the largest value of $k$ for which $f(k)$ is less than some universal speed constant $U$.

For $U = 10^{20}$, we call this value number of Klams that the algorithm is worth.
The are currently no theorems that indicate limits on what we might reasonably expect to achieve in terms of practical parameter ranges for FPT problems.

However, the goodness of a FPT algorithm can be measured by the largest value of $k$ for which $f(k)$ is less than some universal speed constant $U$.

For $U = 10^{20}$, we call this value *number of Klams* that the algorithm is worth.
Outline

1. Introduction & Motivation
2. Parameterized Tractability
3. Parameterized Intractability
4. Conclusions
Barometer of Intractability:

\[ FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \]

- The W-Hierarchy replaces NPC of classical theory.
- The higher a problem is located in the W-hierarchy, the more unlike it is to be in FPT.

How can we find such classes?
- We need a complete problem for all W[i]!
Improving Classical Intractability

Barometer of Intractability:

\[ FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \]

- The W-Hierarchy replaces NPC of classical theory.
- The higher a problem is located in the W-hierarchy, the more unlike it is to be in FPT.

How can we find such classes?
- We need a complete problem for all W[i]!
Definition

A parameterized problem $L$ reduces to a parameterized problem $L'$ if we can transform an instance $(x, k)$ of $L$ into an instance $(x', g(k))$ of $L'$ in time $f(k) \cdot |x|^c$ ($f$ and $g$ are arbitrary computable functions), such that $(x, k)$ is a yes-instance of $L$ if and only if $(x', g(k))$ is a yes-instance of $L'$. 
Classical Reductions vs. Parameterized Reductions

Are classical poly-reductions also parameterized reductions?

- $\text{CNF} \propto \text{IP}_2 \Rightarrow \text{WEIGHTED CNF} \propto \text{WEIGHTED IP}_2$.

- $\text{CNF} \propto \text{3SAT} \not\Rightarrow \text{WEIGHTED CNF} \propto \text{WEIGHTED 3SAT}$. (Moreover, it is unknown if such a reduction exists.)

- In general, classical reductions do not have enough structure to induce parameterized reductions.
In Search of Complete Problems

Classical Completeness: SAT ∈ NPC
We are looking for a similar theorem for every W[i].

Question: What makes satisfiability difficult?

Definition
A propositional formula X is called \( t\)-normalized if X is of the form conjunction-of-disjunction-... of literals with \( t \) alternations.

- 2-normalized formula corresponds to CNF
- We believe that SAT for \( t \)-normalized formulae is strictly easier than for \((t+1)\)-normalized formulae.
Classical Completeness: SAT ∈ NPC
We are looking for a similar theorem for every W[i].

Question: What makes satisfiability difficult?

Definition
A propositional formula X is called \textit{t-normalized} if X is of the form conjunction-of-disjunction-... of literals with t alternations.

- 2-normalized formula corresponds to CNF
- We believe that SAT for t-normalized formulae is strictly easier than for \((t+1)\)-normalized formulae.
Some Completeness Results

**Proposition***

\[ \forall t \geq 1 \text{ WEIGHTED } t\text{-NORMALIZED SAT is complete for W}[t]. \]

* In reality, the W-classes are defined by use of boolean decision circuits and this proposition rises to a theorem.

**Theorem (Complexity Barometer)**

\[ FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \]

- VERTEX COVER \( \in \) FPT
- INDEPENDENT SET & CLIQUE are complete for W[1].

Amazing! They are equivalent in classical complexity theory.
Some Completeness Results

**Proposition**
\[ \forall t \geq 1 \text{ WEIGHTED } t\text{-NORMALIZED SAT is complete for } W[t]. \]

* In reality, the W-classes are defined by use of boolean decision circuits and this proposition rises to a theorem.

**Theorem (Complexity Barometer)**
\[ FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \]

- VERTEX COVER \( \in \) FPT
- INDEPENDENT SET & CLIQUE are complete for \( W[1] \).

Amazing! They are equivalent in classical complexity theory.
Beyond $W[t]$-Completeness

If we do not impose any bound on the depth of the propositional formula, we arrive at classes hard for $\cup_t W[t]$.

WEIGHTED SAT

- **Instance:** A boolean formula $X$.
- **Parameter:** $k$
- **Question:** Does $X$ have a weight $k$ satisfying assignment?

WEIGHTED CIRCUIT SAT

- **Instance:** A decision circuit $C$.
- **Parameter:** $k$
- **Question:** Does $C$ have a weight $k$ satisfying assignment?
Beyond $W[t]$-Completeness

If we do not impose any bound on the depth of the propositional formula, we arrive at classes hard for $\bigcup_t W[t]$.

**WEIGHTED SAT**
- *Instance*: A boolean formula $X$.
- *Parameter*: $k$
- *Question*: Does $X$ have a weight $k$ satisfying assignment?

**WEIGHTED CIRCUIT SAT**
- *Instance*: A decision circuit $C$.
- *Parameter*: $k$
- *Question*: Does $C$ have a weight $k$ satisfying assignment?
Beyond $W[t]$-Completeness

$W[SAT]$: problems reducible to WEIGHTED SAT.

$W[P]$: problems reducible to WEIGHTED CIRCUIT SAT.

Theorem

\[
FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \subseteq W[SAT] \subseteq W[P].
\]

The containment $W[SAT] \subseteq W[P]$ seems to be proper but there is no proof for this assumption.
Beyond $W[t]$-Completeness

$W[SAT]$: problems reducible to WEIGHTED SAT.

$W[P]$: problems reducible to WEIGHTED CIRCUIT SAT.

**Theorem**

\[
FPT \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \ldots \subseteq W[SAT] \subseteq W[P].
\]

The containment $W[SAT] \subseteq W[P]$ seems to be proper but there is no proof for this assumption.
Beyond $W[t]$-Completeness

$W[SAT]$: problems reducible to WEIGHTED SAT.

$W[P]$: problems reducible to WEIGHTED CIRCUIT SAT.

**Theorem**


The containment $W[SAT] \subseteq W[P]$ seems to be proper but there is no proof for this assumption.
**Definition (X Classes)**

Let $C$ be a classical complexity class. We say that a parameterized language $L$ is in $XC$ iff $L_k \in C$ for all $k$.

For instance: the class $XP$ consists of those languages that are slicewise polynomial.

**Theorem (Complexity Barometer)**

- $FPT \subset XP$
- $FPT \subseteq W[1] \subseteq W[2] \subseteq \ldots \subseteq W[SAT] \subseteq W[P] \subseteq XP$

The connection of the two last theorems shows that at least one inclusion in the $W$-hierarchy must be strict.
Definition (X Classes)

Let $C$ be a classical complexity class. We say that a parameterized language $L$ is in $XC$ iff $L_k \in C$ for all $k$.

For instance: the class $XP$ consists of those languages that are slicewise polynomial.

Theorem (Complexity Barometer)

- $FPT \subset XP$
- $FPT \subset W[1] \subset W[2] \subset \ldots \subset W[SAT] \subset W[P] \subset XP$

The connection of the two last theorems shows that at least one inclusion in the $W$-hierarchy must be strict.
Outline

1. Introduction & Motivation
2. Parameterized Tractability
3. Parameterized Intractability
4. Conclusions
"Parameterized complexity is surprisingly orthogonal to classical complexity theory. There are no systematic or trivial correlations between the two complexities of a given problem. The W-hierarchy can be regarded as a complexity barometer that measures how close a problem comes to a successful deal with the devil."

(Downey & Fellows)