The Hardness Boundary for Neighborhood Diversity

Martin Koutecký
martin@koutecky.name

Faculty of Mathematics and Physics, Charles University in Prague
Overview

1. Introduction
   - The topic
   - The parameter

2. Where we are now
   - What piques our interest
   - The state of the art

3. Our questions
Introduction

The topic: The hardness boundary of neighborhood diversity
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**: 
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**: Do we really encounter the worst case instances in real life?
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**:
  - Do we really encounter the worst case instances in real life?
  - Perhaps our instances have some structure that we could exploit (bounded degree, planarity, ...)
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters \(\subseteq\) Parameterized complexity \(\subseteq\) Computational complexity
- **The motivation of parameterized complexity**: 
  - Do we really encounter the worst case instances in real life?
  - Perhaps our instances have some structure that we could exploit (bounded degree, planarity, …)
- **An archetype result:**
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**:
  - Do we really encounter the worst case instances in real life?
  - Perhaps our instances have some structure that we could exploit (bounded degree, planarity, ...)
- **An archetype result**:
  - An $O(2^k n)$ algorithm for VERTEX COVER
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**:
  - Do we really encounter the worst case instances in real life?
  - Perhaps our instances have some structure that we could exploit (bounded degree, planarity, ...)
- **An archetype result**:
  - An $O(2^k n)$ algorithm for VERTEX COVER
  - Courcelle’s Theorem (MSO model checking in $O(f(k)n)$)
Introduction

- **The topic**: The hardness boundary of neighborhood diversity
- **Taxonomy**: Graph structural parameters $\subseteq$ Parameterized complexity $\subseteq$ Computational complexity
- **The motivation of parameterized complexity**:  
  - Do we really encounter the worst case instances in real life?  
  - Perhaps our instances have some structure that we could exploit (bounded degree, planarity, ...)
- **An archetype result**:  
  - An $O(2^k n)$ algorithm for VERTEX COVER  
  - Courcelle’s Theorem (MSO model checking in $O(f(k)n)$)  
  - Generally: we look for an algorithm running in $O(f(k)n^c)$, demonstrating that the problem is *Fixed Parameter Tractable* (FPT).
Neighborhood diversity

\[ u \sim v \equiv (N(u) \setminus \{v\}) = (N(v) \setminus \{u\}) \]
Neighborhood diversity

- $u \sim v \equiv (N(u) \setminus \{v\}) = (N(v) \setminus \{u\})$
- Equivalence classes are bags or types
Neighborhood diversity

- \( u \sim v \equiv (N(u) \setminus \{v\}) = (N(v) \setminus \{u\}) \)
- Equivalence classes are bags or types
- The bags are either cliques or independent sets; between them are either no edges or all possible edges (a complete bipartite graph)
Neighborhood diversity

- \( u \sim v \equiv (N(u) \setminus \{v\}) = (N(v) \setminus \{u\}) \)
- Equivalence classes are bags or types
- The bags are either cliques or independent sets; between them are either no edges or all possible edges (a complete bipartite graph)
- Thus we can form a type graph:
The motivation for this parameter

- The most famous structural parameter: \textit{treewidth}
The motivation for this parameter

- The most famous structural parameter: treewidth
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
The motivation for this parameter

- The most famous structural parameter: treewidth
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)

Other parameters:
- The vertex cover size
- Cliquewidth
- Rankwidth
The motivation for this parameter

- The most famous structural parameter: **treewidth**
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)
  - A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO$_2$ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
The motivation for this parameter

- The most famous structural parameter: \textit{treewidth}
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (\textsc{Coloring}, \textsc{Independent Set}, \textsc{Hamiltonian Cycle}, \ldots)
  - A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every $\textit{MSO}_2$ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
  - A problem: the $f(k)$ above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]
The motivation for this parameter

- **The most famous structural parameter:** **treewidth**
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)
  - A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO₂ definable property is decidable on graphs with \( tw \leq k \) in time \( O(f(k)n) \))
  - **A problem:** the \( f(k) \) above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]

- **Other parameters:**
The most famous structural parameter: **treewidth**

- Algorithms (typically) work by dynamic programming on the tree-decomposition
- Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)
- A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO₂ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
- **A problem:** the $f(k)$ above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]

Other parameters:

- The vertex cover size
The most famous structural parameter: **treewidth**

- Algorithms (typically) work by dynamic programming on the tree-decomposition
- Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)
- A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO₂ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
- **A problem**: the $f(k)$ above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]

Other parameters:

- The vertex cover size
- Cliquewidth
The motivation for this parameter

- The most famous structural parameter: **treewidth**
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...) 
  - A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO₂ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
  - **A problem:** the $f(k)$ above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]

- Other parameters:
  - The vertex cover size
  - Cliquewidth
  - Rankwidth
The motivation for this parameter

- The most famous structural parameter: **treewidth**
  - Algorithms (typically) work by dynamic programming on the tree-decomposition
  - Many problems are FPT w.r.t. treewidth (COLORING, INDEPENDENT SET, HAMILTONIAN CYCLE, ...)
  - A generalization of these results is the metaalgorithmical Courcelle’s Theorem (every MSO$_2$ definable property is decidable on graphs with $tw \leq k$ in time $O(f(k)n)$)
  - **A problem:** the $f(k)$ above is an exponential tower and this cannot be improved (unless P=NP) [Frick, Grohe ’04]

- Other parameters:
  - The vertex cover size
  - Cliquewidth
  - Rankwidth
  - ...
Question: Is there a parameter, that allows $MSO_1$ model checking in time $O(f(k)n)$ for $f(k)$ a tower of constant height?
Faster MSO model checking?

- **Question**: Is there a parameter, that allows MSO₁ model checking in time $O(f(k)n)$ for $f(k)$ a tower of constant height?
- **Yes!** [Lampis ’11] On *neighborhood diversity* this is possible ($O(2^{2^k} n)$).
**Question**: Is there a parameter, that allows $MSO_1$ model checking in time $O(f(k)n)$ for $f(k)$ a tower of constant height?

**Yes**! [Lampis '11] On *neighborhood diversity* this is possible ($O(2^{2^k} n)$).

**ND’s relationships to other parameters:**
What piques our interest. . .

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
What piques our interest...

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - Capacitated Dominating Set / Capacitated Vertex Cover
What piques our interest...

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - Capacitated Dominating Set / Capacitated Vertex Cover
  - Precoloring Extension
ND is incomparable with TW, yet many problems hard for TW are easy for ND:

- **Capacitated Dominating Set / Capacitated Vertex Cover**
- **Precoloring Extension**
- **Equitable Coloring**
What piques our interest. . .

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - Capacitated Dominating Set / Capacitated Vertex Cover
  - Precoloring Extension
  - Equitable Coloring
  - Achromatic Number
What piques our interest. . .

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - **CAPACITATED DOMINATING SET / CAPACITATED VERTEX COVER**
  - **PRECOLORING EXTENSION**
  - **EQUITABLE COLORING**
  - **ACHROMATIC NUMBER**
  - **L(p,q)-LABELING**
ND is incomparable with TW, yet many problems hard for TW are easy for ND:

- Capacitated Dominating Set / Capacitated Vertex Cover
- Precoloring Extension
- Equitable Coloring
- Achromatic Number
- L(p,q)-Labeling

The information contained in an ND bounded graph is very small: $O(k^2 \log(n))$ – could this little information create a big solution space?
What piques our interest. . .

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - Capacitated Dominating Set / Capacitated Vertex Cover
  - Precoloring Extension
  - Equitable Coloring
  - Achromatic Number
  - L(p,q)-Labeling

- The information contained in an ND bounded graph is very small: $O(k^2 \log(n))$ – could this little information create a big solution space?

- The only hard problems for ND we know of have some extra information besides the graph on the input:
What piques our interest. . .

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - **CAPACITATED DOMINATING SET / CAPACITATED VERTEX COVER**
  - PRECOLORING EXTENSION
  - EQUITABLE COLORING
  - ACHROMATIC NUMBER
  - L(p,q)-LABELING

- The information contained in an ND bounded graph is very small: $O(k^2 \log(n))$ – could *this little* information create a *big* solution space?

- The only hard problems for ND we know of have **some extra information** besides the graph on the input:
  - LIST COLORING (already hard for VC)
What piques our interest... 

- ND is incomparable with TW, yet many problems hard for TW are easy for ND:
  - Capacitated Dominating Set / Capacitated Vertex Cover
  - Precoloring Extension
  - Equitable Coloring
  - Achromatic Number
  - L(p,q)-Labeling

- The information contained in an ND bounded graph is very small: $O(k^2 \log(n))$ – could this little information create a big solution space?

- The only hard problems for ND we know of have some extra information besides the graph on the input:
  - List Coloring (already hard for VC)
  - Channel Assignment (every instance can be reduced to a clique)
## The state of the art

<table>
<thead>
<tr>
<th>Problem</th>
<th>Treewidth</th>
<th>Neighborhood diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Precoloring extension</strong></td>
<td>W[1]-hard</td>
<td>FPT [Ganian12]</td>
</tr>
<tr>
<td><strong>L(p,q)-labeling</strong></td>
<td>NP-c for TW ≥ 2</td>
<td>FPT [FialaGKK13]</td>
</tr>
<tr>
<td><strong>List coloring</strong></td>
<td>W[1]-hard</td>
<td>W[1]-hard</td>
</tr>
<tr>
<td><strong>Channel assignment</strong></td>
<td>NP-c for TW ≥ 3</td>
<td>NP-h</td>
</tr>
</tbody>
</table>
### The state of the art (contd.)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Treewidth</th>
<th>Neighborhood diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equitable Coloring</td>
<td>W[1]-hard</td>
<td>FPT [*]</td>
</tr>
<tr>
<td>Achromatic Number</td>
<td>NP-c on trees</td>
<td>FPT [*]</td>
</tr>
<tr>
<td>CDS</td>
<td>W[1]-hard</td>
<td>FPT [*]</td>
</tr>
<tr>
<td>CVC</td>
<td>W[1]-hard</td>
<td>FPT [*]</td>
</tr>
<tr>
<td>$p$-Vertex-Disjoint Paths</td>
<td>W[1]-hard</td>
<td>FPT [Ganian12]</td>
</tr>
<tr>
<td>$p$-Edge-Disjoint Paths</td>
<td>NP-c for $\text{TW} \geq 2$</td>
<td>Open</td>
</tr>
<tr>
<td>Locally constrained homomorphism</td>
<td>W[1]-hard</td>
<td>Open</td>
</tr>
<tr>
<td>Edge Bipartization</td>
<td>FPT</td>
<td>Open</td>
</tr>
</tbody>
</table>
Questions

- Michael Lampis conjectures that all problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND
Michael Lampis conjectures that all problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND

If so, how does one prove that?
Questions

- Michael Lampis conjectures that all problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND
  - If so, how does one prove that?
  - If not, we need to find a problem that is hard (W[1]-hard) for ND
Questions

- Michael Lampis conjectures that all problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND
  - If so, how does one prove that?
  - If not, we need to find a problem that is hard (W[1]-hard) for ND

- A related question: since all positive results for ND were accomplished using FPT Integer Linear Programming [Lenstra ’83], is there a way to generalize this paradigm (think Courcelle’s Theorem)?
Michael Lampis conjectures that **all** problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND

- If so, how does one prove that?
- If not, we need to find a problem that is hard (W[1]-hard) for ND

A related question: since all positive results for ND were accomplished using FPT Integer Linear Programming [Lenstra ’83], is there a way to generalize this paradigm (think Courcelle’s Theorem)?

- We could exploit even stronger results than Lenstra’s: every Convex Integer Programming problem is FPT w.r.t. the dimension (the number of variables). [Khachiyan, Porkolab ’00]
Questions

- Michael Lampis conjectures that all problems whose input is only the graph (plus possibly some constant) are FPT w.r.t. ND
  - If so, how does one prove that?
  - If not, we need to find a problem that is hard (W[1]-hard) for ND

- A related question: since all positive results for ND were accomplished using FPT Integer Linear Programming [Lenstra ’83], is there a way to generalize this paradigm (think Courcelle’s Theorem)?
  - We could exploit even stronger results than Lenstra’s: every Convex Integer Programming problem is FPT w.r.t. the dimension (the number of variables). [Khachiyan, Porkolab ’00]

- What about shrub-depth?
Thank you for your attention!