The Hardness Boundary for Neighborhood Diversity

Martin Koutecký
martin@koutecky.name

Faculty of Mathematics and Physics, Charles University in Prague

Overview

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 $\subsetneq$ Parameterized complexity $\subsetneq$ 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

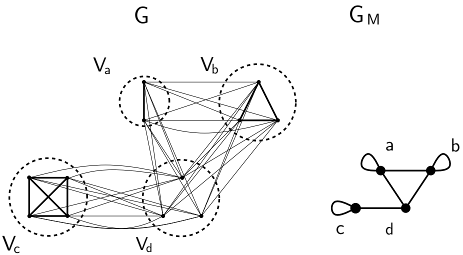
- $u \sim v \equiv (N(u) \setminus \{v\}) = (N(v) \setminus \{u\})$
- Equivalence classes are *bags* or *types*

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: **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, ...)

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: **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]

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 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

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

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

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
  - ...

Faster MSO model checking?

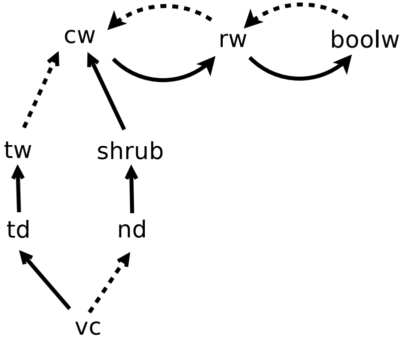
- **Question:** Is there a parameter, that allows MSO_1 model checking in time $O(f(k)n)$ for $f(k)$ a tower of constant height?

Faster MSO model checking?

- **Question:** Is there a parameter, that allows 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 ($O(2^{2^k} n)$).

Faster MSO model checking?

- **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

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

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

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?

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

Problem	**Treewidth**	**Neighborhood diversity**
PRECOLORING EXTENSION	W[1]-hard	FPT [Ganian12]
L(P,Q)-LABELING	NP-c for TW ≥ 2	FPT [FialaGKK13]
LIST COLORING	W[1]-hard	W[1]-hard
CHANNEL ASSIGNMENT	NP-c for TW ≥ 3	NP-h

The state of the art (contd.)

Problem	Treewidth	Neighborhood diversity
EQUITABLE COLORING	W[1]-hard	FPT [*]
ACHROMATIC NUMBER	NP-c on trees	FPT [*]
CDS	W[1]-hard	FPT [*]
CVC	W[1]-hard	FPT [*]
p-VERTEX-DISJOINT PATHS	W[1]-hard	FPT [Ganian12]
p-EDGE-DISJOINT PATHS	NP-c for TW ≥ 2	Open
Locally constrained homomorphism	W[1]-hard	Open
EDGE BIPARTIZATION	FPT	Open

Questions

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

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)?

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]

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*?

Conclusion

Thank you for your attention!