Acknowledgment

I am very grateful to

- Georg Gottlob and Stefan Szeider for providing me their slides of previous talks on related topics;
- Martin Lackner and Stefan Rümmele for their great help in creating this slide set.

Roadmap

I. High complexity everywhere
II. Parameterized Complexity
III. What is a Parameter?
IV. How to Make Use of Parameters?
V. Conclusion and Research Opportunities

Part I:
High complexity everywhere
A starting point

Theory of Computation: 1930s
– What can be computed?

Theory of Tractability: 1970s
– What can be computed efficiently?

Computational hardness

NP-completeness:
• small solutions
• easy to verify
• huge search space

thousands of NP-complete problems

NP-complete problems

Combinatorial auctions (e.g., London Transport)
Scheduling (e.g., work schedules)
Graph layout problems (e.g., circuit design)

Satisfiability

Is the following formula $\varphi$ satisfiable?

$\varphi = (a \lor b) \land (b \lor c) \land (\neg a \lor d) \land (\neg b \lor \neg c \lor \neg d)$
Satisfiability

Is the following formula $\phi$ satisfiable?

$$\phi = (a \lor b) \land (b \lor c) \land (\neg a \lor d) \land (\neg b \lor \neg c \lor \neg d)$$

very expressive NP-complete problem

Work(8-9) \land Work(9-10) \land Work(10-11) \rightarrow Break(11-12)

Work(8-9) \land Work(10-11) \rightarrow (Break(9-10) \lor Work(9-10))

The hierarchy of complexity

How to deal with intractability

Incomplete methods

- Approximation algorithms
- Heuristics
- Random algorithms

not exact

Exact methods

- Brute force
- “Islands of tractabilities”
- ...

How to deal with intractability

Incomplete methods

- Approximation algorithms
- Heuristics
- Random algorithms

not exact
Islands of tractability

Easy to solve special cases

For the Satisfiability problem

- Horn formulas:
  \[ \text{Work}(8-9) \land \text{Work}(9-10) \land \text{Work}(10-11) \rightarrow \text{Break}(11-12) \]

- Krom formulas:
  \[ \neg \text{Work}(8-9) \lor \neg \text{Break}(8-9) \]

However: Such “islands” are not robust!

Summary of Part I

- High complexity is ubiquitous
- There are many methods to tackle intractable problems, ...
- … but can we have a method that is
  - efficient,
  - exact and
  - robust?

Parameterized Complexity

Main idea:
Exploit structural properties of problem instances.

- efficient,
- exact and
- robust.

Part II:
Parameterized Complexity
Example: subway emergencies

Problem:
emergency teams for every line segment

For example:
There has to be a team either at Santa Ana or at Los Héroes to cover the line segment in between.

---

Vertex Cover

- Given a graph G and integer k, is there a set of vertices S of size at most k such that for every edge \{a,b\}, S contains a or b?
- NP-complete
- the best known exact algorithms require exponential time

---

Real-world versus random

108 stations, connected
108 nodes, connected

---

The classical point of view

Classical Complexity Theory
- measures complexity only in terms of size of an instance
- one-dimensional: \(O(f(n))\)
- tractable means polynomial time
The classical point of view

Classical Complexity Theory

- measures complexity only in terms of size of an instance
- one-dimensional: $O(f(n))$
- tractable means polynomial time

Criticism:

We (almost) always know more about the input than its number of bits.

The parameterized point of view

Parameterized Complexity Theory

- takes structural properties of problem instances into account (parameter $k$)
- multi-dimensional: $O(f(n,k))$
- tractability depends on parameter

Fixed-parameter tractability (FPT)

- ideal outcome of a parameterized complexity analysis
- $O(f(k) \cdot n^d)$
- example: $2^k \cdot n^2$
- consequence: tractable for parameter of bounded size.

Benefit:

Allows a more fine-grained notion of tractability.
Fixed-parameter tractability (FPT)

- ideal outcome of a parameterized complexity analysis
- $O(f(k) \cdot n^d)$
- example: $2^k \cdot n^2$
- consequence: tractable for parameter of bounded size.

Example: Vertex Cover with bounded treewidth

The class XP

- “second prize”
- $O(n^{f(k)})$
- example: $n^k$
- consequence: less favorable than FPT but still tractable for parameter of bounded size

Example: Conjunctive query evaluation with queries of bounded hypertree-width

Comparision: FPT and XP

<table>
<thead>
<tr>
<th></th>
<th>FPT</th>
<th>XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^k \cdot 100^2$</td>
<td>$100^k$</td>
<td>classical point of view: $2^{100^2} &gt; 10^{30}$</td>
</tr>
</tbody>
</table>
Summary of Part II

- Real-world problems often have structure
- **Classical** complexity only considers the size of a problem instance
- **Parameterized** complexity takes problem parameters into account
- Exploit inherent structure of problem instances

---

Boolean Satisfiability Revisited

\[ \varphi = (a \lor b) \land (b \lor c) \land (\neg a \lor d) \land (\neg b \lor \neg c \lor \neg d) \]

---

Boolean Satisfiability Revisited

\[ \varphi = (a \lor b) \land (b \lor c) \land (\neg a \lor d) \land (\neg b \lor \neg c \lor \neg d) \]

- Clause size = 3
- Positive clause size = 2
- Negative clause size = 3
- Number of non-Horn clauses = 2
- Number a variable occurs positively = 2
- Number of variables that occur as positive literals = 4
- … as negative literals … = 4
- etc.
Parameterized Complexity Analysis

- **Parameter “number of clauses”**
  - \(1.24^k \cdot \text{poly}(n)\)
- **Parameter “number of variables”**
  - \(1.49^k \cdot \text{poly}(n)\)
- **Parameter “clause size”**
  - NP-complete for \(k=3\)
  - \(O(2^{f(n)})\)

Generalization of Special Cases

- **SAT**
  - almost Horn / almost Krom ⇒ backdoors
- **Graph Problems**
  - almost trees ⇒ treewidth
- **CSPs / CQs**
  - almost acyclic ⇒ hypertree-width

Minimal Model Satisfiability (MMSAT)

**Instance:** CNF formulas \(\phi\) and \(\pi\).

**Question:** Is there a subset minimal model of \(\phi\) that also satisfies \(\pi\)?

- Important subtask in non-monotonic reasoning.
- MMSAT captures the complexity of computing a minimal model.

Minimal Model Satisfiability

- \(k\): maximum weight of the minimal model
- \(d\): clause size
- \(d^+,d^-\): positive/negative clause size
- \(h\): number of non-Horn clauses
- \(p\): number a variable occurs positively
- \(v^+,v^-\): number of variables that occur as positive/negative literals in \(\phi\) or in \(\pi\)
- \(d_x^+\): positive clause size in \(\pi\)
- \(||\pi||\): length of \(\pi\)
- \(b\): size of a strong Horn backdoor set

[Lackner, Pfandler 2012]
Summary of Part III

- **Any characteristics** of problem instances may serve as parameters.
- **Usefulness** of a parameter depends on:
  - FPT (or at least XP) result
  - Application context

Part IV:
How to Make Use of Parameters?

Toolbox of Parameterized Complexity

**Hardness Tools**
- W[i]-hardness
- Kernel lower bounds
- Exponential Time Hypothesis

**Algorithmic Tools**
- Bounded search trees
- Iterative compression
- Color coding
- Graph minors
- Logical meta-theorems
- Kernelization
Tree Decomposition

- Tree with a vertex set (= “bag”) associated with every node.
- For every edge (v,w): there is a bag containing both v and w.
- For every v: the nodes that contain v form a connected subtree.

Tree Decomposition

- Tree with a vertex set (= “bag”) associated with every node.
- For every edge (v,w): there is a bag containing both v and w.
- For every v: the nodes that contain v form a connected subtree.
Tree Decomposition

Graph
- Tree with a vertex set (= “bag”) associated with every node.
- For every edge (v,w): there is a bag containing both v and w.
- For every v: the nodes that contain v form a connected subtree.

Structure
- Tree with a set of domain elements (= “bag”) associated with every node.
- For every tuple \( a_1, \ldots, a_k \) in any relation \( R_i \): there is a bag containing \( \{a_1, \ldots, a_k\} \).
- For every \( a \): the nodes that contain \( a \) form a connected subtree.

Treewidth
- The width of a tree decomposition is the maximum bag size – 1.
- The treewidth of \( G \) is the minimum width over all tree decompositions of \( G \).
- For fixed \( k \), it is feasible in linear time to decide / compute tree decomposition of width \( \leq k \). [Bodlaender, 1996]

Treewidth of a CNF Formula
- Represent CNF formula \( F \) as finite structure \( A(F) \) over signature \( \tau = \{\text{cl}, \text{var}, \text{pos}, \text{neg}\} \)
  - \( \text{cl}(c), \text{var}(x) \): \( c \) is a clause (\( x \) is a variable) in \( F \)
  - \( \text{pos}(x,c), \text{neg}(x,c) \): \( x \) occurs unnegated (negated) in \( c \)
- Treewidth of \( F \) is defined as the treewidth of \( A(F) \)

Remark: this corresponds to the incidence graph of \( F \).
Treewidth of CNF: Example

Let $F = (x_1 \lor \neg x_2 \lor x_3) \land (x_2 \lor \neg x_3 \lor x_4)$

- Then $A(F)$ consists of:
  - $\text{cl}(c_1), \text{cl}(c_2)$
  - $\text{var}(x_1), \text{var}(x_2), \text{var}(x_3), \text{var}(x_4)$
  - $\text{pos}(x_1,c_1), \text{pos}(x_3,c_1), \text{pos}(x_2,c_2), \text{pos}(x_4,c_2)$
  - $\text{neg}(x_2,c_1), \text{neg}(x_3,c_2)$

$F$ has treewidth = 2.

Monadic Second-Order Logic (MSO)

- MSO extends first-order logic by the use of set variables.
- An MSO formula allows the following atoms:
  - Relational atoms, e.g. $R(x_1,\ldots,x_n)$.
  - Equational atoms, e.g. $x=y$.
  - Atoms based on set variables, e.g. $X(y)$.

Courcelle's Theorem: Any property of finite structures which is definable in MSO can be decided in time $O(f(k)\cdot n)$, where $n$ is the size of the structure and $k$ is its treewidth.
From MSO to Algorithms

Various (automatic) constructions of FPT-algorithms from MSO-encodings exist:
- Using correspondence between finite tree automata and MSO on trees [Courcelle 1990; Arnborg et al. 1991; Flum et al. 2001]
- Finite model theory (k-types) [Grohe 1999]
- Datalog [Gottlob, P., Wei 2007]
- Games [Kneis, Langer, Rossmanith, 2011]
- New automata models (Courcelle, Durand 2011)
- Answer-set programming [Bliem, P., Woltran 2013]

Applying Courcelle's Theorem

Theorem (folklore): SAT is FPT w.r.t. treewidth.

Proof: MSO-encoding.
Let F be a CNF formula and X a set of variables.
Encoding of X⊨F:
(∀c)cl(c) → (∃z)[(pos(z,c) ∧ X(z)) ∨ (neg(z,c) ∧ ¬X(z))]

Encoding of SAT:
(∃X) X|=F

Minimal Model Satisfiability

Theorem [Gottlob, P., Wei, 2006]:
Minimal Model SAT is FPT w.r.t. treewidth.

Instance: CNF formulas F and π
Question: Is there a subset minimal model of φ that also satisfies π?
Minimal Model Satisfiability

- Instance: CNF formulas $F$ and $\pi$
- Question: Is there a subset minimal model of $\phi$ that also satisfies $\pi$?

**Theorem [Gottlob, P., Wei, 2006]:** Minimal Model SAT is FPT w.r.t. treewidth.

Proof: MSO-encoding.

Encoding of $Y \subseteq X$:

\[ \forall x \ (Y(x) \rightarrow X(x)) \land \exists y \ (X(y) \land \neg Y(y)) \]

Encoding of Minimal Model SAT:

\[ \exists X, X \models F \land X \models \pi \land \forall Y \ (Y \subseteq X \rightarrow \neg Y \models F) \]

Treewidth as a Key to Tractable Reasoning

- Minimal models
- Various forms of closed-world reasoning
- Disjunctive logic programming
- Propositional abduction

[Gottlob, P., Wei, 2006]

Conclusion

Parameterized complexity is a viable way to tackle intractable problems:

- **Real-world** problems often have structure
- **Classical** complexity only considers the size of a problem instance
- **Parameterized** complexity takes problem parameters into account
The Gentle Revolution of Parameterized Complexity

- Bioinformatics, Operations Research, Optimization, Automated Reasoning, etc.
- STOC, FOCS, SODA, IJCAI, ...
- 4 Monographs
  - [Downey&Fellows 1999]
  - [Flum&Grohe 2006]
  - [Niedermeier 2006]
  - [Downey&Fellows 2013]
- The Computer Journal (BCS) two special issues in 2008

Research Opportunities

- Parameterized complexity theory:
  - New algorithmic methods
  - Kernelization (formal model of preprocessing)
  - Relationship with other approaches (approximation, heuristics)
- Further applications

Thank You

Gracias