13. Review
COMP6741: Parameterized and Exact Computation
Serge Gaspers
Semester 2, 2015

Contents

1 Review
1.1 Upper Bounds .................................................. 1
1.2 Lower Bounds .................................................. 3
2 Research in Parameterized Complexity .......................... 3

1 Review
1.1 Upper Bounds
Dynamic Programming across Subsets

- very general technique
- uses solutions of subproblems
- typically stored in a table of exponential size

Analysis of Branching Algorithm

Lemma 1 (Measure Analysis Lemma). Let

- A be a branching algorithm
- c ≥ 0 be a constant, and
- μ(·), η(·) be two measures for the instances of A,

such that on input I, A calls itself recursively on instances I₁, ..., Iₖ, but, besides the recursive calls, uses time \(O(|I|^c)\), such that

\[
(\forall i) \quad \eta(I_i) \leq \eta(I) - 1, \text{ and } \quad 2^{\mu(I_1)} \cdot \ldots \cdot 2^{\mu(I_k)} \leq 2^{\mu(I)}.
\]

Then A solves any instance I in time \(O(\eta(I)^{c+1}) \cdot 2^{\mu(I)}\).

Inclusion-Exclusion

Theorem 2 (IE-theorem – intersection version). Let \(U = A_0\) be a finite set, and let \(A_1, \ldots, A_k \subseteq U\).

\[
\left| \bigcap_{i \in \{1, \ldots, k\}} A_i \right| = \sum_{J \subseteq \{1, \ldots, k\}} (-1)^{|J|} \left| \bigcap_{i \in J} A_i \right|,
\]

where \(\overline{A_i} = U \setminus A_i\) and \(\bigcap_{i \in \emptyset} = U\).
**Theorem 3.** The number of covers with \(k\) sets and the number of ordered partitions with \(k\) sets of a set system \((V,H)\) can be computed in polynomial space and

1. \(O^*(2^n|H|)\) time if \(H\) can be enumerated in \(O^*(|H|)\) time and poly space,
2. \(O^*(3^n)\) time if membership in \(H\) can be decided in polynomial time, and
3. \(\sum_{j=0}^{n} \binom{n}{j} T_H(j)\) time if there is a \(T_H(j)\) time poly space algorithm to count for any \(W \subseteq V\) with \(|W| = j\) the number of sets \(S \in H\) s.t. \(S \cap W = \emptyset\).

**Main Complexity Classes**

- **P:** class of problems that can be solved in time \(n^{O(1)}\)
- **FPT:** class of problems that can be solved in time \(f(k) \cdot n^{O(1)}\)
- **W[\cdot]:** parameterized intractability classes
- **XP:** class of problems that can be solved in time \(f(k) \cdot n^{\Omega(k)}\)

\[P \subseteq FPT \subseteq W[1] \subseteq W[2] \cdots \subseteq W[P] \subseteq XP\]

Known: If FPT = W[1], then the Exponential Time Hypothesis fails, i.e. 3-Sat can be solved in time \(2^{o(n)}\).

**Kernelization: definition**

**Definition 4.** A kernelization for a parameterized problem \(\Pi\) is a **polynomial time** algorithm, which, for any instance \(I\) of \(\Pi\) with parameter \(k\), produces an equivalent instance \(I'\) of \(\Pi\) with parameter \(k'\) such that \(|I'| \leq f(k)\) and \(k' \leq f(k)\) for a computable function \(f\). We refer to the function \(f\) as the **size** of the kernel.

**Search trees**

Recall: A **search tree** models the recursive calls of an algorithm. For a \(b\)-way branching where the parameter \(k\) decreases by \(a\) at each recursive call, the number of nodes is at most \(b^{k/a} \cdot (k/a + 1)\).

If \(k/a\) and \(b\) are upper bounded by a function of \(k\), and the time spent at each node is FPT (typically, polynomial), then we get an FPT running time.

**Tree decompositions (by example)**

- A graph \(G\)

  ![Graph Image]

- A tree decomposition of \(G\)
Iterative Compression

For a minimization problem:

- **Compression step**: Given a solution of size \( k + 1 \), compress it to a solution of size \( k \) or prove that there is no solution of size \( k \).

- **Iteration step**: Incrementally build a solution to the given instance by deriving solutions for larger and larger subinstances.

- Often, we can get a solution of size \( k + 1 \) with only a polynomial overhead.

### 1.2 Lower Bounds

**Reductions**

We have seen several reductions, which, for an instance \((I, k)\) of a problem \(\Pi\), produce an equivalent instance \(I'\) of a problem \(\Pi'\).

<table>
<thead>
<tr>
<th>Reduction Type</th>
<th>Time</th>
<th>Parameter</th>
<th>Special Features</th>
<th>Used for</th>
</tr>
</thead>
<tbody>
<tr>
<td>kernelization</td>
<td>poly</td>
<td>(k' \leq g(k))</td>
<td>(</td>
<td>I'</td>
</tr>
<tr>
<td>parameterized reduction</td>
<td>FPT</td>
<td>(k' \leq g(k))</td>
<td>(\Pi = \Pi')</td>
<td>W[*]-hardness</td>
</tr>
<tr>
<td>OR-composition</td>
<td>poly</td>
<td>(k' \leq poly(k))</td>
<td>(\Pi = OR(\Pi'))</td>
<td>Kernel LBs</td>
</tr>
<tr>
<td>AND-composition</td>
<td>poly</td>
<td>(k' \leq poly(k))</td>
<td>(\Pi = AND(\Pi'))</td>
<td>Kernel LBs</td>
</tr>
<tr>
<td>polynomial transformation</td>
<td>poly</td>
<td>(k' \leq poly(k))</td>
<td>(\Pi = AND(\Pi'))</td>
<td>Kernel LBs</td>
</tr>
<tr>
<td>SubExponential Reduction</td>
<td>subexp((k))</td>
<td>(k' \in O(k))</td>
<td>Turing reduction</td>
<td>ETH LBs</td>
</tr>
<tr>
<td>Family</td>
<td></td>
<td></td>
<td>(</td>
<td>I'</td>
</tr>
</tbody>
</table>

### 2 Research in Parameterized and Exact Computation

**News**

- **BICLIQUE** has been solved (the first Open problem among “The Most Infamous” in [Downey Fellows, 2013]): it is W[1]-hard [Lin, SODA 2015]

- research focii
  - enumeration algorithms and combinatorial bounds
  - randomized algorithms
  - backdoors
  - treewidth: computation, bounds on the treewidth of grid or planar subgraphs / minors
  - bidimensionality
  - bottom-up: improving the quality of subroutines of heuristics
  - (S)ETH widely used now, also for poly-time lower bounds
  - quests for multivariate algorithms, lower bounds for Turing kernels
  - FPT-approximation algorithms
Resources

- FPT wiki: [http://fpt.wikidot.com](http://fpt.wikidot.com)
- Blog: [http://fptnews.org](http://fptnews.org)
- cstheory stackexchange: [http://cstheory.stackexchange.com](http://cstheory.stackexchange.com)
- FPT school 2014: [http://fptschool.mimuw.edu.pl](http://fptschool.mimuw.edu.pl)