13. Review

COMP6741: Parameterized and Exact Computation

Serge Gaspers[1,2]

[1]School of Computer Science and Engineering, UNSW Australia
[2]Optimisation Resarch Group, NICTA

Semester 2, 2015

Outline

1. Review
   - Upper Bounds
   - Lower Bounds

2. Research in Parameterized and Exact Computation

Outline

1. Review
   - Upper Bounds
   - Lower Bounds

2. Research in Parameterized and Exact Computation

Outline

1. Review
   - Upper Bounds
   - Lower Bounds

2. Research in Parameterized and Exact Computation

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*
- $c \geq 0$ *be a constant, and*
- $\mu(\cdot), \eta(\cdot)$ *be two measures for the instances of* A,

such that on input I, A *calls itself recursively on instances* $I_1, \ldots, I_k$, *but, besides the recursive calls, uses time* $O(|I|^c)$, *such that*

$$(\forall i) \quad \eta(I_i) \leq \eta(I) - 1, \text{ and} \tag{1}$$

$$2^{\mu(I_1)} + \ldots + 2^{\mu(I_k)} \leq 2^{\mu(I)}. \tag{2}$$

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 $A_1, \dots, A_k \subseteq U$.

$$\left| \bigcap_{i \in \{1,\dots,k\}} A_i \right| = \sum_{J \subseteq \{1,\dots,k\}} (-1)^{|J|} \left| \bigcap_{i \in J} \overline{A_i} \right|,$$

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 $S \in H$ st. $S \cap W = \emptyset$.*

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[·]: parameterized intractability classes
XP: class of problems that can be solved in time $f(k) \cdot n^{g(k)}$

$$\text{P} \subseteq \text{FPT} \subseteq \text{W}[1] \subseteq \text{W}[2] \cdots \subseteq \text{W}[P] \subseteq \text{XP}$$

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

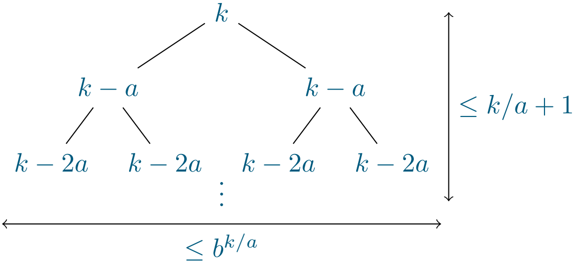
Definition 4

A kernelization for a parameterized problem Π is a **polynomial time** algorithm, which, for any instance I of Π with parameter k, produces an **equivalent** instance I' of Π 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

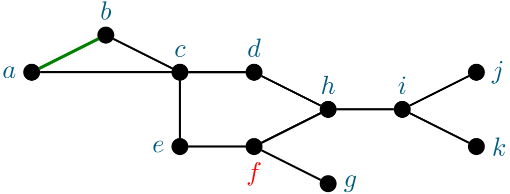
Recall: A search tree models the recursive calls of an algorithm.
For a b-way branching where the parameter k decreases by a at each recursive call, the number of nodes is at most $b^{k/a} \cdot (k/a+1)$.



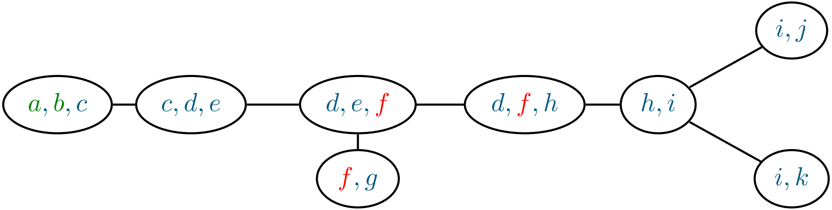
If k/a and b are upper bounded by a function of k, and the time spent at each node is FPT (typically, polynomial), then we get an FPT running time.

Tree decompositions (by example)

- A graph G



- A *tree decomposition* of G



Conditions: covering and connectedness.

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

Outline

1. Review
   - Upper Bounds
   - Lower Bounds

2. Research in Parameterized and Exact Computation

Reductions

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

	time	parameter	special features	used for
kernelization	poly	$k' \leq g(k)$	$\lvert I' \rvert \leq g(k)$ $\Pi = \Pi'$	$g(k)$-kernels
parameterized reduction	FPT	$k' \leq g(k)$		W[]-hardness
OR-composition	poly	$k' \leq \text{poly}(k)$	$\Pi = \text{OR}(\Pi')$	Kernel LBs
AND-composition	poly	$k' \leq \text{poly}(k)$	$\Pi = \text{AND}(\Pi')$	Kernel LBs
polynomial parameter transformation	poly	$k' \leq \text{poly}(k)$		Kernel LBs (S)ETH LBs
SubExponential Reduction Family	subexp(k)	$k' \in O(k)$	Turing reduction $\lvert I' \rvert = \lvert I \rvert^{O(1)}$	ETH LBs

Outline

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`
- FPT newsletter: `http://fpt.wikidot.com/fpt-news:the-parameterized-complexity-newsletter`
- Blog: `http://fptnews.org`
- cstheory stackexchange: `http://cstheory.stackexchange.com`
- FPT school 2014: `http://fptschool.mimuw.edu.pl`