Survey of connections between approximation algorithms and parameterized complexity

Dániel Marx

Humboldt-Universität zu Berlin, Germany

Dagstuhl Seminar 11091:

Packing and Scheduling Algorithms for Information and Communication Services

February 28, 2011

Parameterized complexity

Main idea: Instead of expressing the running time as a function $T(n)$ of n, we express it as a function $T(n, k)$ of the input size n and some parameter k of the input.

We do not want to be efficient on all inputs of size n, only for those where k is small.

What can be the parameter k?

- The size k of the solution we are looking for.
- The maximum degree of the input graph.
- The diameter of the input graph.
- The length of clauses in the input Boolean formula.
- ...

Fixed-parameter tractability

Definition: A **parameterization** of a decision problem is a function that assigns an integer parameter k to each input instance x.

The parameter can be

- explicit in the input (for example, if the parameter is the integer k appearing in the input (G, k) of VERTEX COVER), or
- implicit in the input (for example, if the parameter is the diameter d of the input graph G).

Main definition:

A parameterized problem is **fixed-parameter tractable (FPT)** if there is an $f(k)n^c$ time algorithm for some constant c.

Example: VERTEX COVER is FPT: can be solved in time $O(1.2832^k k + k|V|)$ [Niedermeier, Rossmanith, 2003]

FPT problems

Examples of NP-hard problems that are FPT:

- Finding a vertex cover of size k.
- Finding a path of length k.
- Finding k disjoint triangles.
- Drawing the graph in the plane with at most k edge crossings.
- Finding disjoint paths that connect k given pairs of points.
- ...

W[1]-hardness

Negative evidence similar to NP-completeness. If a problem is **W[1]-hard,** then the problem is not FPT unless FPT=W[1].

Some W[1]-hard problems:

- Finding a clique/independent set of size k.
- Finding a dominating set of size k.
- Finding k pairwise disjoint sets.
- …

Standard parameterization

Given an **optimization** problem we can turn it into a **decision** problem: the input is a pair (x, k) and we have to decide if there is a solution for x with cost at least/at most k.

The **standard parameterization** of an optimization problem is the associated decision problem, with the value k appearing in the input being the parameter.

Example:

VERTEX COVER

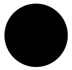
Input:	(G, k)
Parameter:	k
Question:	Is there a vertex cover of size at most k?

If the standard parameterization of an optimization problem is FPT, then (intuitively) it means that we can solve it efficiently if the optimum is small.

Bounded search tree method

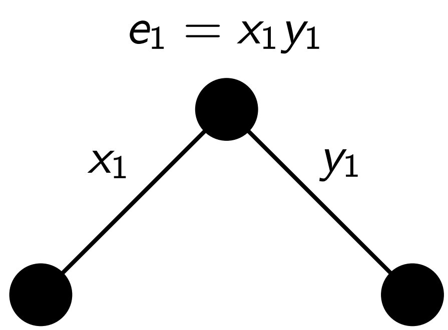
Algorithm for MINIMUM VERTEX COVER:

$e_1 = x_1 y_1$



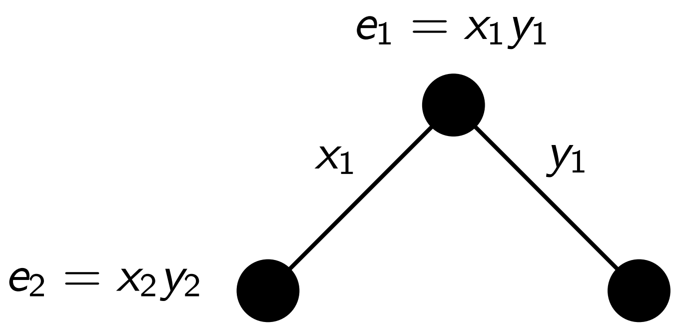
Bounded search tree method

Algorithm for MINIMUM VERTEX COVER:



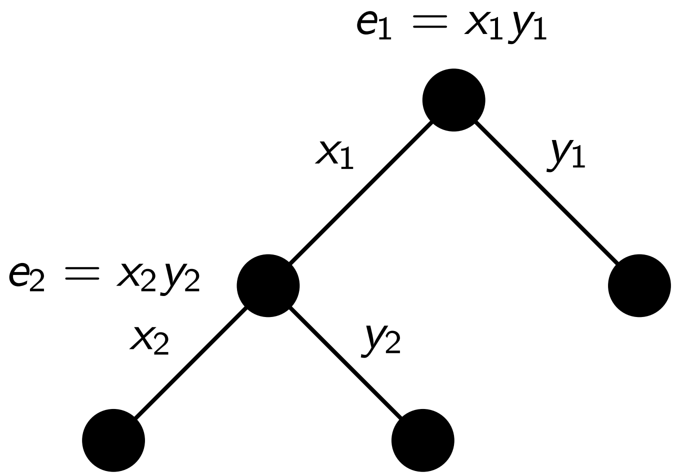
Bounded search tree method

Algorithm for MINIMUM VERTEX COVER:



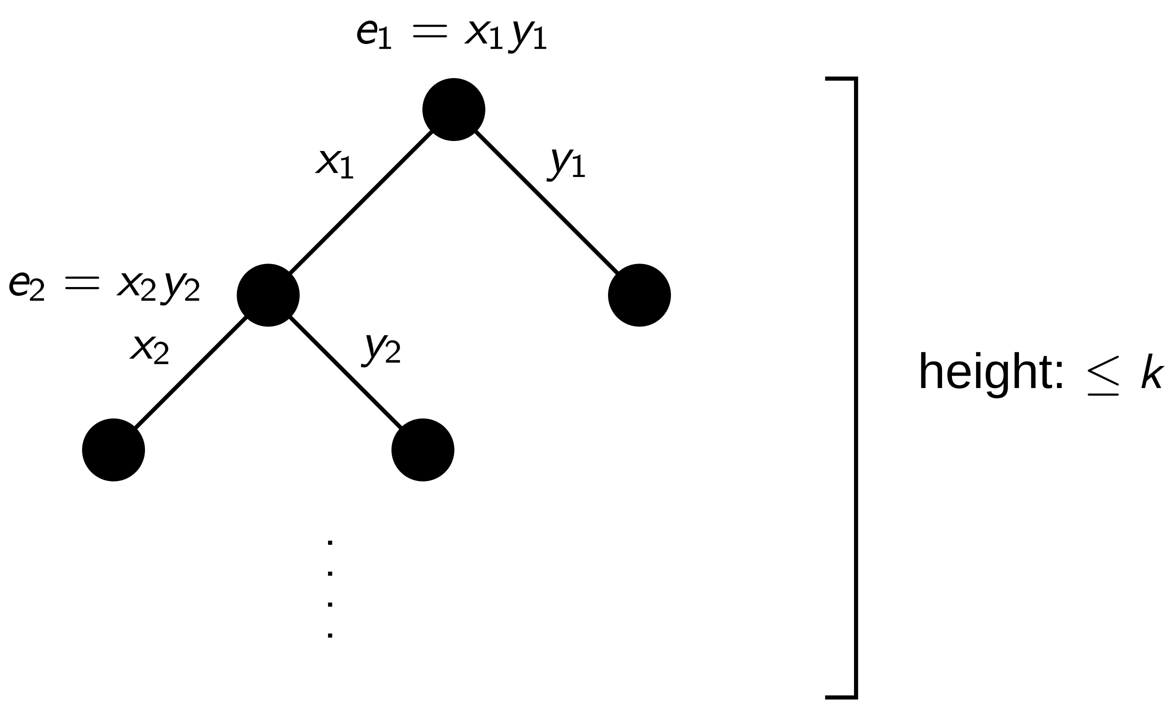
Bounded search tree method

Algorithm for MINIMUM VERTEX COVER:



Bounded search tree method

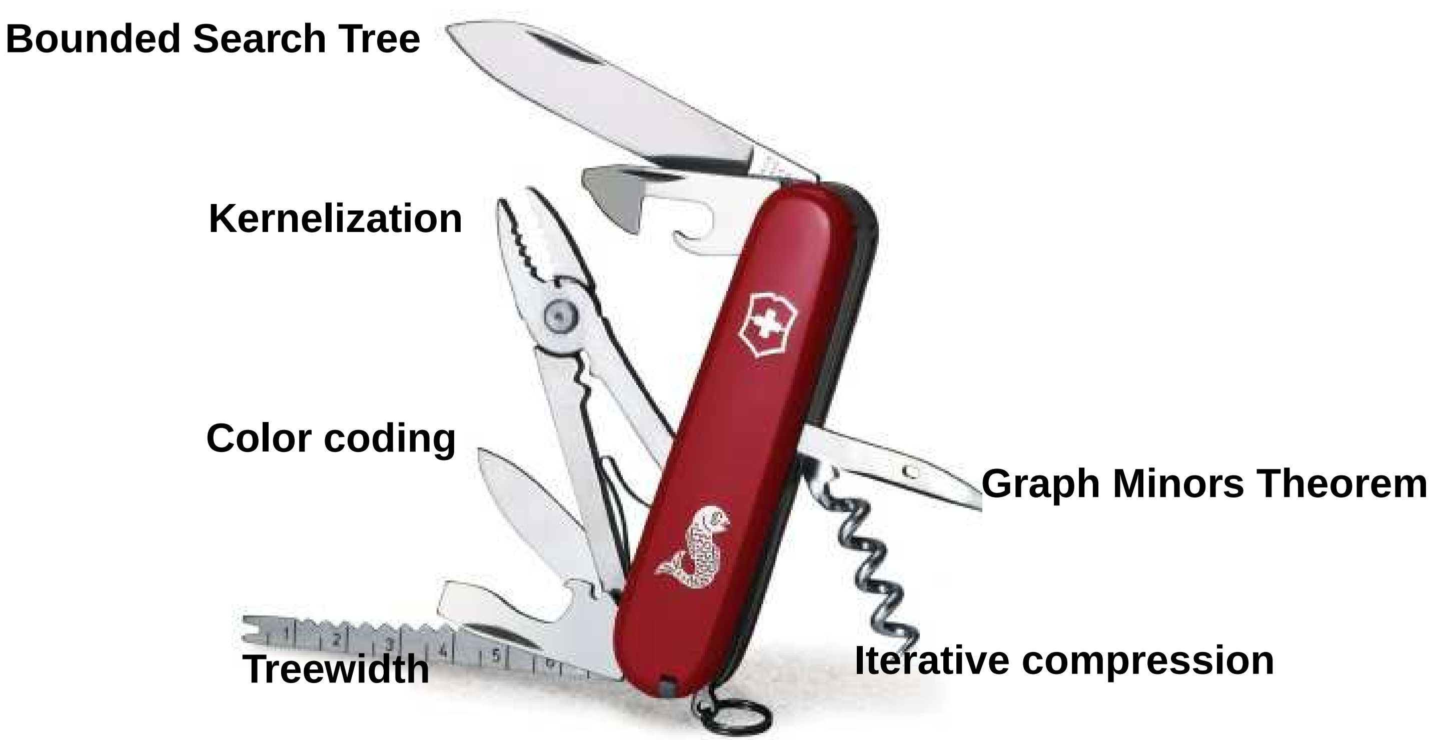
Algorithm for MINIMUM VERTEX COVER:



Height of the search tree is $\leq k \Rightarrow$ number of leaves is $\leq 2^k \Rightarrow$ complete search requires $2^k \cdot$ poly steps.

FPT algorithmic techniques

- Significant advances in the past 20 years or so (especially in recent years).
- Powerful toolbox for designing FPT algorithms:



Overview

- Approximation schemes and parameterized complexity.
- Approximation algorithms parameterized by “something.”
- Approximation algorithms parameterized by the cost.

Approximation schemes

Polynomial-time approximation scheme (PTAS):

Input:	Instance x, $\epsilon > 0$
Output:	$(1+\epsilon)$-approximate solution
Running time:	polynomial in $|x|$ for every fixed ϵ

- **PTAS:** running time is $|x|^{f(1/\epsilon)}$
- **EPTAS:** running time is $f(1/\epsilon) \cdot |x|^{O(1)}$
- **FPTAS:** running time is $(1/\epsilon)^{O(1)} \cdot |x|^{O(1)}$

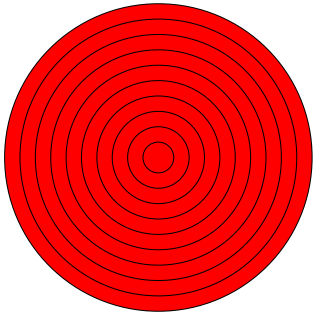
Connections with parameterized complexity:

- Methodological similarities between EPTAS and FPT design.
- Lower bounds on the efficiency of approximation schemes.

Baker's shifting strategy for EPTAS

Theorem: There is a $2^{O(1/\epsilon)} \cdot n$ time EPTAS for INDEPENDENT SET.

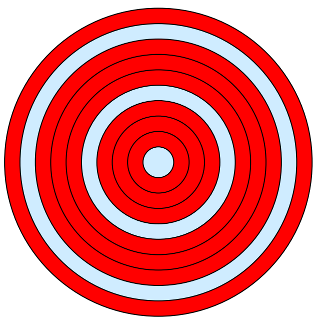
A planar graph can be decomposed into a series of "layers."



Baker's shifting strategy for EPTAS

Theorem: There is a $2^{O(1/\epsilon)} \cdot n$ time EPTAS for INDEPENDENT SET.

A planar graph can be decomposed into a series of "layers."

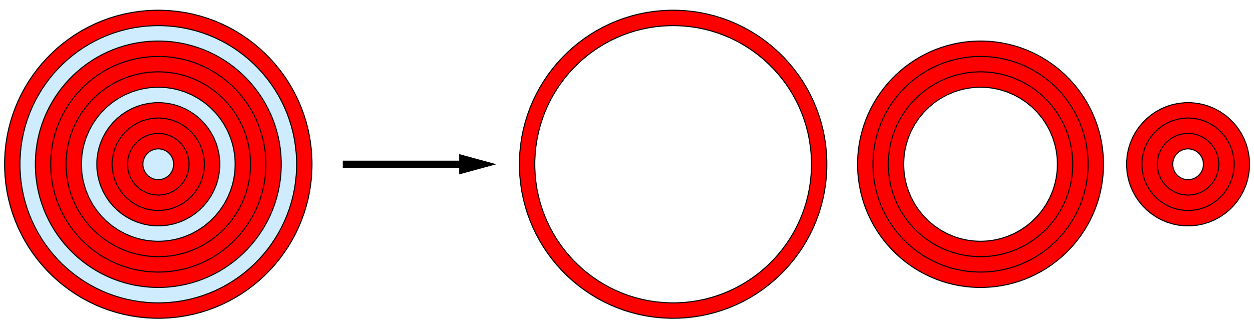


Let $D := 1/\epsilon$. For a fixed $0 \le s < D$, delete layers L_s, L_{s+D}, L_{s+2D}, ...

Baker's shifting strategy for EPTAS

Theorem: There is a $2^{O(1/\epsilon)} \cdot n$ time EPTAS for INDEPENDENT SET.

A planar graph can be decomposed into a series of "layers."



Let $D := 1/\epsilon$. For a fixed $0 \le s < D$, delete layers L_s, L_{s+D}, L_{s+2D}, ...

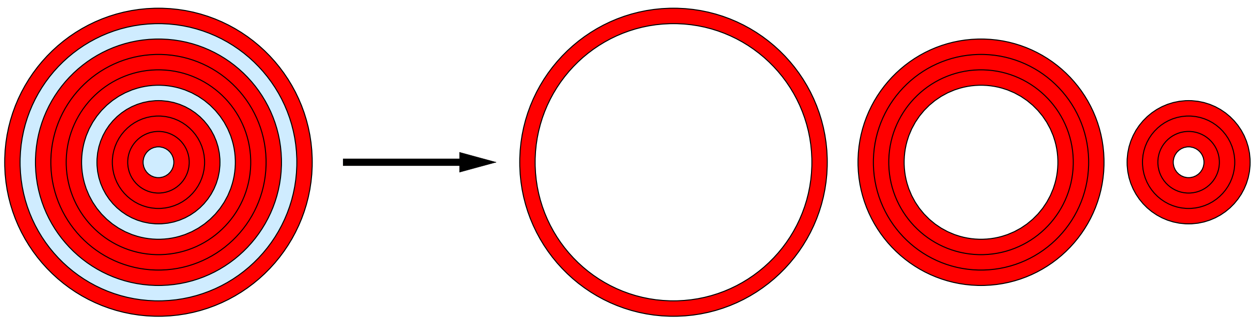
Lemma: [Bodlaender] The treewidth of a k-layer graph is at most $3k + 1$.

Thus after the deletion, we can solve the problem in time $O(2^{3D+1} \cdot n)$ using treewidth techniques.

We do this for every $0 \le s < D$: for at least one value of s, only ϵ-fraction of the optimum solution is deleted $\Rightarrow$ we get a $(1 + \epsilon)$-approximation.

Baker's shifting strategy for FPT

Theorem: SUBGRAPH ISOMORPHISM for planar graphs (given planar graphs H and G, is H a subgraph of G?) is FPT parameterized by $k := |V(H)|$.



Let $D := k + 1$. For a fixed $0 \leq s < D$, delete layers L_s, L_{s+D}, L_{s+2D}, ... $\Rightarrow$ the resulting graph has treewidth $3k + 1 \Rightarrow$ SUBGRAPH ISOMORPHISM can be solved in time $k^{O(k)} \cdot n$ using treewidth techniques.

We do this for every $0 \leq s < D$: for at least one value of s, we do not delete any of the k vertices of the solution $\Rightarrow$ we find a copy of H in G if there is one.

Lower bounds

Observation: [Bazgan 1995] [Cesati, Trevisan 1997] If the standard parameterization of an optimization problem is W[1]-hard, then it does not have an EPTAS, unless FPT $=$ W[1].

Proof: Suppose an $f(1/\epsilon) \cdot n^{O(1)}$ time EPTAS exists. Running this EPTAS with $\epsilon := 1/(k+1)$ decides if the optimum is at most k.

Lower bounds

Observation: [Bazgan 1995] [Cesati, Trevisan 1997] If the standard parameterization of an optimization problem is W[1]-hard, then it does not have an EPTAS, unless FPT $=$ W[1].

Proof: Suppose an $f(1/\epsilon) \cdot n^{O(1)}$ time EPTAS exists. Running this EPTAS with $\epsilon := 1/(k+1)$ decides if the optimum is at most k.

Thus W[1]-hardness results immediately show that (assuming W[1] $\neq$ FPT)

- No EPTAS for INDEPENDENT SET for unit disks/squares [M. 2005]
- No EPTAS for DOMINATING SET for unit disks/squares [M. 2005]
- No EPTAS for planar TMIN, TMAX, MPSAT [Cai et al. 2007]

Note: All these problems have $n^{O(1/\epsilon)}$ time approximation schemes.

Tighter bounds

We have seen that there are no EPTAS for some problems (unless FPT $= W[1]$). But is there a PTAS with running time say $n^{O(\log\log(1/\epsilon))}$?

Tighter bounds

We have seen that there are no EPTAS for some problems (unless FPT $= W[1]$). But is there a PTAS with running time say $n^{O(\log\log(1/\epsilon))}$?

The following hypothesis can be used to obtain lower bounds on the exponent:

Exponential-time hypothesis (ETH): n-variable 3SAT cannot be solved in time $2^{o(n)}$.

Theorem: Assuming ETH, there is no $f(1/\epsilon)n^{o(\sqrt{1/\epsilon})}$ time PTAS for

- INDEPENDENT SET for unit disks/squares
- DOMINATING SET for unit disks/squares
- planar TMIN, TMAX, MPSAT

Tighter bounds

We have seen that there are no EPTAS for some problems (unless FPT $= W[1]$). But is there a PTAS with running time say $n^{O(\log\log(1/\epsilon))}$?

The following hypothesis can be used to obtain lower bounds on the exponent:

Exponential-time hypothesis (ETH): n-variable 3SAT cannot be solved in time $2^{o(n)}$.

Theorem: Assuming ETH, there is no $f(1/\epsilon)n^{o(\sqrt{1/\epsilon})}$ time PTAS for

- Independent Set for unit disks/squares
- Dominating Set for unit disks/squares
- planar TMIN, TMAX, MPSAT

Theorem: [M. 2007] Assuming ETH, there is no PTAS with running time $2^{(1/\epsilon)^{O(1)}} \cdot n^{O((1/\epsilon)^{(1-\delta)})}$ for any $\delta > 0$ for these problems.

FPT & PTAS – Summary

- Methodological similarities on planar graphs
  - Baker's shifting strategy, reduction to bounded treewidth
  - Framework of bidimensionalty
- Lower bounds on the quality of approximation schemes: it is possible to prove (almost) tight results.

Approximation parameterized by “something”

Idea: Instead of finding an approximation algorithm with running time $n^{O(1)}$, we try to find an approximation algorithm with running time $f(k) \cdot n^{O(1)}$, where k is some parameter of the optimization problem instance.

Approximation parameterized by “something”

Idea: Instead of finding an approximation algorithm with running time $n^{O(1)}$, we try to find an approximation algorithm with running time $f(k) \cdot n^{O(1)}$, where k is some parameter of the optimization problem instance.

Example: [Böckenhauer et al. 2007] METRIC TSP WITH DEADLINE is the standard metric TSP problem, extended with a set D of deadline nodes. The salesperson must reach $v \in D$ within time at most $d(v)$.

Let $|D|$ be the parameter.

Approximation parameterized by "something"

Idea: Instead of finding an approximation algorithm with running time $n^{O(1)}$, we try to find an approximation algorithm with running time $f(k) \cdot n^{O(1)}$, where k is some parameter of the optimization problem instance.

Example: [Böckenhauer et al. 2007] METRIC TSP WITH DEADLINE is the standard metric TSP problem, extended with a set D of deadline nodes. The salesperson must reach $v \in D$ within time at most $d(v)$.

Let $|D|$ be the parameter.

- Approximation: The problem has no constant factor approximation (unless P $=$ NP).
- Parameterization: The problem is NP-hard even for $|D| = 1$, thus it is not FPT (unless P $=$ NP).
- Approximation + parameterization: A 2.5-approximation can be found in time $O(n^3 + |D|! \cdot |D|)$

Partial Vertex Cover

PARTIAL VERTEX COVER: Select k vertices, maximizing the number of edges covered.

- Approximation: The problem has a constant factor approximation, but has no PTAS (unless P $=$ NP).
- Parameterization: The problem is W[1]-hard, thus it is not FPT (unless FPT $=$ W[1]).
- Approximation + parameterization: A $(1+\epsilon)$-approximation can be found in time $f(k,\epsilon) \cdot n^{O(1)}$.

Genus

Genus: A graph has genus at most k if it can be drawn on the sphere with k handles attached to it.

- $g = 0 \Leftrightarrow$ graph is planar.
- Vertex Coloring and Independent Set are NP-hard on planar graphs, thus these problems are not FPT parameterized by genus (unless P $=$ NP).
- A 2-approximation of Vertex Coloring can be found in time $f(g) \cdot n^{O(1)}$ [Demaine et al. 2005].
- A $(1+\epsilon)$-approximation for Independent Set can be found in time $f(g,\epsilon) \cdot n^{O(1)}$ [Demaine and Hajiaghayi 2004], [Grohe 2003].

k-CENTER *and* k-MEDIAN

k-CENTER

Input:	Set $\mathbb{R}^2$ of points, integer k
Find:	Subset $C \subseteq S$ of size k
Goal:	Minimize $\max_{s \in S} \min_{c \in C} d(s, c)$.

k-MEDIAN

Input:	Set $\mathbb{R}^2$ of points, integer k
Find:	Subset $C \subseteq S$ of size k
Goal:	Minimize $\sum_{s \in S} \min_{c \in C} d(s, c)$.

Theorem: [Har-Peled, Mazumdar 2004] A $(1+\epsilon)$-approximation for k-MEDIAN can be found in time $f(\epsilon) \cdot n^{O(1)}$.

Theorem: [Gonzalez 1985] There is a polynomial 2-approximation for k-CENTER, but there is no PTAS, unless $\mathrm{P} = \mathrm{NP}$.

Theorem: [Agarwal, Procopiuc 2002] A $(1+\epsilon)$-approximation for k-CENTER can be found in time $f(k, \epsilon) \cdot n^{O(1)}$.

Approximation parameterized by "something" – summary

- A straightforward combination of approximation and FPT.
- $f(k) \cdot n^{O(1)}$ or $f(k, \epsilon) \cdot n^{O(1)}$ time approximation algorithms, where k is some parameter of the optimization problem.
- Can give constant factor approximation or PTAS for problems where polynomial-time algorithms cannot.
- Some relevant parameters: dimension, number of centers, maximum degree, ...

Approximation parameterized by the cost

Idea: Approximation algorithms that are efficient if the optimum is small.

Intuitively, we would like to parameterize by the optimum value, but that is problematic since usually we expect that the parameter is known.

More or less equivalent definitions by [Chen, Grohe, Grüber 2006], [Downey, Fellows, McCartin 2006], and [Cai, Huang 2006].

Approximation parameterized by the cost

Definition: An **fpt-approximation algorithm** with ratio ϱ for a minimization problem is an algorithm that, given an input (x, k) with $\text{opt}(x) \le k$, outputs in time $f(k) \cdot n^{O(1)}$ a solution with cost $\le k \cdot \varrho(k)$.

We require that $k \cdot \varrho(k)$ is nondecreasing.

Definition: An **fpt-approximation algorithm** with ratio ϱ for a maximization problem is an algorithm that, given an input (x, k) with $\text{opt}(x) \ge k$, outputs in time $f(k) \cdot n^{O(1)}$ a solution with cost $\ge k/\varrho(k)$.

We require that $k/\varrho(k)$ is unbounded and nondecreasing.

Approximation parameterized by the cost

Definition: An **fpt-approximation algorithm** with ratio ϱ for a minimization problem is an algorithm that, given an input (x, k) with $\text{opt}(x) \leq k$, outputs in time $f(k) \cdot n^{O(1)}$ a solution with cost $\leq k \cdot \varrho(k)$.

We require that $k \cdot \varrho(k)$ is nondecreasing.

Definition: An **fpt-approximation algorithm** with ratio ϱ for a maximization problem is an algorithm that, given an input (x, k) with $\text{opt}(x) \geq k$, outputs in time $f(k) \cdot n^{O(1)}$ a solution with cost $\geq k/\varrho(k)$.

We require that $k/\varrho(k)$ is unbounded and nondecreasing.

Two differences from polynomial-time approximation:

- $f(k) \cdot n^{O(1)}$ time instead of $n^{O(1)}$
- ratio $\varrho(k)$ depends on k ($\approx$ optimum) and not on the input size.

Edge multicut

EDGE MULTICUT: Given pairs of vertices $(s_1, t_1), \dots, (s_\ell, t_\ell)$, delete at most k edges such that there is no $s_i - t_i$ path for any i.

Theorem: [M. 2004] EDGE MULTICUT can be solved in time $f(k,\ell)n^{O(1)}$.

Theorem: [M., Razgon 2009] EDGE MULTICUT has an FPT 2-approximation in time $f(k)n^{O(1)}$.

Theorem: [M., Razgon 2011] [Bousquet, Daligault, Thomassé 2011] EDGE MULTICUT can be solved in time $f(k)n^{O(1)}$.

Topological bandwidth

Definitions:

- **Linear layout** of a graph $G(V, E)$ is a bijection between V and $\{1, \dots, |V|\}$.
- **Bandwidth** of a layout: the maximum "length" of an edge.
- **Cutwidth** of a layout: the maximum no. of edges crossing some $(i, i+1)$.
- Bandwidth bw(G) and cutwidth cw(G) of a graph is the minimum possible bandwidth/cutwidth of a linear layout.
- **Topological bandwidth** tbw(G) is the minimum bandwidth of a subdivision of G.

Fact: Cutwidth is FPT [Thilikos et al. 2000], but (topological) bandwidth is W[1]-hard [Bodlaender et al. 1994].

Topological bandwidth

FPT approximation for topological bandwidth based on the following observation:

Observation: [Fellows] $\text{tbw}(G) \leq \text{cw}(G) + 1 \leq \text{tbw}(G)^2$

If $\text{tbw}(G) \leq k$, then $\text{cw}(G) \leq k^2 - 1$ and we can find such a layout in FPT time.

The first inequality is algorithmic: a layout with cutwidth at most $k^2 - 1$ can be used to obtain a subdivision of G and a layout for it having bandwidth $\leq k^2$.

$\Rightarrow$ FPT-approximation for topological bandwidth with ratio k.

Disjoint directed cycles

DISJOINT DIRECTED CYCLES: Find a maximum number of disjoint cycles in a directed graph.

Theorem: [Slivkins 2003] DISJOINT DIRECTED CYCLES is W[1]-hard.

Theorem: [Grohe, Grüber 2007] DISJOINT DIRECTED CYCLES has an FPT ϱ-approximation for some function ϱ.

Disjoint directed cycles

DISJOINT DIRECTED CYCLES: Find a maximum number of disjoint cycles in a directed graph.

Theorem: [Slivkins 2003] DISJOINT DIRECTED CYCLES is W[1]-hard.

Theorem: [Grohe, Grüber 2007] DISJOINT DIRECTED CYCLES has an FPT ϱ-approximation for some function ϱ.

It turns out that something stronger is true:

Theorem: [Grohe, Grüber 2007] There is a polynomial-time algorithm that finds a solution of DISJOINT DIRECTED CYCLES with OPT$/\varrho$(OPT) cycles for some nontrivial function ϱ.

Disjoint directed cycles

DISJOINT DIRECTED CYCLES: Find a maximum number of disjoint cycles in a directed graph.

Theorem: [Slivkins 2003] DISJOINT DIRECTED CYCLES is W[1]-hard.

Theorem: [Grohe, Grüber 2007] DISJOINT DIRECTED CYCLES has an FPT ϱ-approximation for some function ϱ.

It turns out that something stronger is true:

Theorem: [Grohe, Grüber 2007] There is a polynomial-time algorithm that finds a solution of DISJOINT DIRECTED CYCLES with $\mathrm{OPT}/\varrho(\mathrm{OPT})$ cycles for some nontrivial function ϱ.

Surprisingly, it is true for every optimization problem (where a trivial solution is easy to find) that an FPT ϱ-approximation implies a polynomial-time ϱ' approximation for some other function ϱ'.

From FPT time to polynomial time

Theorem: Suppose that a minimization problem has an FPT time ϱ-approximation algorithm $\mathbb{A}$ and a trivial solution can be found in polynomial time. Then there is a polynomial-time algorithm that finds a solution with cost OPT $\cdot\ \varrho'$(OPT) for some nontrivial function ϱ'.

From FPT time to polynomial time

Theorem: Suppose that a minimization problem has an FPT time ϱ-approximation algorithm $\mathbb{A}$ and a trivial solution can be found in polynomial time. Then there is a polynomial-time algorithm that finds a solution with cost $\text{OPT} \cdot \varrho'(\text{OPT})$ for some nontrivial function ϱ'.

Proof: Suppose that the running time of $\mathbb{A}$ is $f(k)|x|^c$.

We do the following on instance x:

- Find a trivial solution.
- For $i = 1, 2, \dots, |x|$, simulate $\mathbb{A}$ on (x, i) for $|x|^{c+1}$ steps.
- Output: the best of these at most $|x| + 1$ solutions.

From FPT time to polynomial time

Theorem: Suppose that a minimization problem has an FPT time ϱ-approximation algorithm $\mathbb{A}$ and a trivial solution can be found in polynomial time. Then there is a polynomial-time algorithm that finds a solution with cost $\text{OPT} \cdot \varrho'(\text{OPT})$ for some nontrivial function ϱ'.

Proof: Suppose that the running time of $\mathbb{A}$ is $f(k)|x|^c$.

We do the following on instance x:

- Find a trivial solution.
- For $i = 1, 2, \ldots, |x|$, simulate $\mathbb{A}$ on (x, i) for $|x|^{c+1}$ steps.
- Output: the best of these at most $|x| + 1$ solutions.

Approximation ratio: Let $k := \text{opt}(x)$.

If $|x| \geq \max\{k, f(k)\}$, then the simulation of $\mathbb{A}$ on (x, k) terminates in $f(k) \cdot |x|^c \leq |x|^{c+1}$ steps and we get a solution with ratio at most $\varrho(k)$.

From FPT time to polynomial time

Theorem: Suppose that a minimization problem has an FPT time ϱ-approximation algorithm $\mathbb{A}$ and a trivial solution can be found in polynomial time. Then there is a polynomial-time algorithm that finds a solution with cost $\mathrm{OPT} \cdot \varrho'(\mathrm{OPT})$ for some nontrivial function ϱ'.

Proof: Suppose that the running time of $\mathbb{A}$ is $f(k)|x|^c$.

We do the following on instance x:

- Find a trivial solution.
- For $i = 1, 2, \dots, |x|$, simulate $\mathbb{A}$ on (x, i) for $|x|^{c+1}$ steps.
- Output: the best of these at most $|x| + 1$ solutions.

Approximation ratio: Let $k := \mathrm{opt}(x)$.

The number of instances with $|x| < \max\{k, f(k)\}$ is bounded by a function of k, thus the ratio of the trivial solution is at most $\tau(k)$ for such instances.

From FPT time to polynomial time

Theorem: Suppose that a minimization problem has an FPT time ϱ-approximation algorithm $\mathbb{A}$ and a trivial solution can be found in polynomial time. Then there is a polynomial-time algorithm that finds a solution with cost OPT $\cdot\, \varrho'$(OPT) for some nontrivial function ϱ'.

Proof: Suppose that the running time of $\mathbb{A}$ is $f(k)|x|^c$.

We do the following on instance x:

- Find a trivial solution.
- For $i = 1, 2, \ldots, |x|$, simulate $\mathbb{A}$ on (x, i) for $|x|^{c+1}$ steps.
- Output: the best of these at most $|x| + 1$ solutions.

Approximation ratio is at most $\max(\varrho(\text{opt}(x)), \tau(\text{opt}(x)))$.

Open questions

Can we do anything nontrivial for CLIQUE or for HITTING SET?

- Is there an FPT ϱ-approximation for CLIQUE with any ratio function ϱ?
- Is there a polynomial-time algorithm for CLIQUE that finds a clique of size, say, $O(\log\log\log \mathrm{OPT})$?

Because of the previous result, these two questions are equivalent!

In case of a negative answer, very deep techniques are required: the only known way to show (assuming P $\neq$ NP) that CLIQUE and HITTING SET have no constant-factor polynomial time approximation is by using the PCP theorem.

Inapproximability

An optimization problem is **not FPT-approximable** if it has no FPT-approximation algorithm for any function ϱ.

Theorem: [Downey et al. 2008] INDEPENDENT DOMINATING SET is not FPT-approximable, unless FPT $= W[1]$.

Theorem: [Chen, Grohe, Grüber 2006] WEIGHTED CIRCUIT SATISFIABILITY is not FPT-approximable, unless FPT $= W[P]$.

WEIGHTED CIRCUIT SATISFIABILITY: Given a Boolean circuit, find a satisfying assignment with minimum number of 1's.

These two problems are not monotone, so the results are not very surprising.

Monotone inapproximability results

MONOTONE WEIGHTED CIRCUIT SATISFIABILITY

Input:	Boolean cirucuit C **without negations**
Find:	A satisfying assignment a of C
Goal:	Minimize the number of 1's in a

Theorem: [Alekhnovich, Razborov 2001] There is no FPT 2-approximation for MONOTONE WEIGHTED CIRCUIT SATISFIABILITY, unless Randomized FPT $= W[P]$.

Theorem: [Eickmeyer, Grohe, Grüber 2008] There is no FPT ϱ-approximation for MONOTONE WEIGHTED CIRCUIT SATISFIABILITY with polylogarithmic ϱ, unless FPT $= W[P]$.

Theorem: [M. 2010] MONOTONE WEIGHTED CIRCUIT SATISFIABILITY is not FPT-approximable, unless FPT $= W[P]$.

Parameterization by cost – summary

- Idea: efficient approximation when the solution is small.
- Not too many convincing examples so far.
- Inapproximation results could be very deep and challenging.

Conclusions

- Several possible connections to look at between approximation and fixed-parameter tractability.
- There are lots of possibilities for finding new algorithmic results.
- Inapproximability results probably require new approaches.