Survey of connections between approximation algorithms and parameterized complexity

Dániel Marx

Tel Aviv University, Israel

Dagstuhl Seminar 09511

December 14, 2009

Overview

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

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.

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.

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.

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.

- The problem has no constant factor approximation (unless P $=$ NP).
- The problem is NP-hard even for $|D| = 1$, thus it is not FPT (unless P $=$ NP).
- 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.

- The problem has a constant factor approximation, but has not PTAS (unless P $=$ NP).
- The problem is W[1]-hard, thus it is not FPT (unless FPT $=$ W[1]).
- 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 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: [Gonzalez 1985] There is a polynomial 2-approximation for k-CENTER, but there is no PTAS, unless $P = NP$.

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

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

Approximation parameterized by "something"

- 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].

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.

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 $\mathrm{opt}(x) \leq k$, 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 $\mathrm{opt}(x) \geq k$, 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.

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 $\mathrm{opt}(x) \leq k$, 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 $\mathrm{opt}(x) \geq k$, 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.

Topological bandwidth

Linear layout of a graph $G(V, E)$ is a bijection between V and $\{1, \ldots, |V|\}$.

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 $\mathrm{bw}(G)$ and cutwidth $\mathrm{cw}(G)$ of a graph is the minimum possible bandwidth/cutwidth of a linear layout.

Topological bandwidth: $\mathrm{tbw}(G)$ minimum bandwidth of a subdivision of G. 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 cutwith at most $k^2 - 1$ can be used to obtain a subdivision of G and a layout for it having bandwidth at most k^2.

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

Treewidth

Theorem: [Bodlaender 1996] A tree decomposition of width k (if exists) can be computed in time $2^{O(k^2)} \cdot n$.

A much simpler approximation algorithm:

Theorem: If $\mathrm{tw}(G) \leq k$, then a tree decomposition of width $4k+1$ can be computed in time $2^{O(k)} \cdot n^2$.

Polynomial time approximation:

Theorem: [Feige, Hajiaghayi, Lee 2005] A tree decomposition of width $O(\mathrm{tw}(G)\sqrt{\log \mathrm{tw}(G)})$ can be computed in polynomial time.

Treewidth

Theorem: [Bodlaender 1996] A tree decomposition of width k (if exists) can be computed in time $2^{O(k^2)} \cdot n$.

A much simpler approximation algorithm:

Theorem: If $\mathrm{tw}(G) \leq k$, then a tree decomposition of width $4k+1$ can be computed in time $2^{O(k)} \cdot n^2$.

Polynomial time approximation:

Theorem: [Feige, Hajiaghayi, Lee 2005] A tree decomposition of width $O(\mathrm{tw}(G)\sqrt{\log \mathrm{tw}(G)})$ can be computed in polynomial time.

Situation is similar for other width measures (rank width, clique width, etc.): FPT-approximation algorithm is easier to find than exact FPT algorithm.

Theorem: [Oum, Seymour 2006] If clique width is at most k, then a $(2^{3k+2}-1)$-expression can be found in FPT time.

Edge multicut

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

Open: Is EDGE MULTICUT FPT parameterized by k?

Theorem: [M., Razgon 2009] EDGE MULTICUT has an FPT 2-approximation.

Edge multicut

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

Open: Is EDGE MULTICUT FPT parameterized by k?

Theorem: [M., Razgon 2009] EDGE MULTICUT has an FPT 2-approximation.

A key step is to use the following result:

ALMOST 2SAT: Delete at most k clauses from a 2SAT formula ϕ to make it satisfiable.

Theorem: [O'Sullivan, Razgon 2008] ALMOST 2SAT is FPT.

Edge multicut

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

Open: Is EDGE MULTICUT FPT parameterized by k?

Theorem: [M., Razgon 2009] EDGE MULTICUT has an FPT 2-approximation.

A key step is to use the following result:

ALMOST 2SAT: Delete at most k clauses from a 2SAT formula ϕ to make it satisfiable.

Theorem: [O'Sullivan, Razgon 2008] ALMOST 2SAT is FPT.

ALMOST 2SAT WITH PAIRS: Given a 2SAT formula ϕ where the clauses are partitioned into pairs, delete at most k pairs ($2k$ clauses) to make it satisfiable.

Observation: [M., Razgon 2009] ALMOST 2SAT WITH PAIRS is W[1]-hard, but has a (trivial) FPT 2-approximation.

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 $\mathrm{OPT}/\varrho(\mathrm{OPT})$ 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 $\mathrm{OPT} \cdot \varrho'(\mathrm{OPT})$ for some unbounded 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 $\mathrm{OPT} \cdot \varrho'(\mathrm{OPT})$ for some unbounded 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.

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 unbounded 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 $k := \mathrm{opt}(x)$.

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 unbounded 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 $k := \mathrm{opt}(x)$.

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 $\mathrm{OPT} \cdot \varrho'(\mathrm{OPT})$ for some unbounded 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 $\max(\varrho(\mathrm{opt}(x)), \tau(\mathrm{opt}(x))$.

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

Inapproximability proof

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

Proof: Suppose that there a FPT ϱ-approximation algorithm $\mathbb{A}$. Given a circuit C and an integer k, we construct a circuit C' such that

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not $k \cdot \varrho(k)$-satisfiable.

Now running $\mathbb{A}$ on C' decides if C is k-satisfiable or not: in the first case, it returns a solution with weight at most $k \cdot \varrho(k)$; in the second case, it cannot return such a solution.

If C has n inputs, then we can think of C as a Boolean function $C(S)$ on the subsets of $[n]$.

Inapproximability proof

Definition: A family $\mathcal{H}$ of functions $[n] \to [k]$ is a k**-perfect** family of hash functions if for every $S \subseteq [n]$ with $|S| = k$, there is a $h \in \mathcal{H}$ such that $h(x) \neq h(y)$ for any $x, y \in S$, $x \neq y$.

Theorem: [Alon, Yuster, Zwick] There is a k-perfect family of functions $[n] \to [k]$ having size $2^{O(k)} \log n$.

Inapproximability proof

Definition: A family $\mathcal{H}$ of functions $[n] \to [k]$ is a k**-perfect** family of hash functions if for every $S \subseteq [n]$ with $|S| = k$, there is a $h \in \mathcal{H}$ such that $h(x) \neq h(y)$ for any $x, y \in S$, $x \neq y$.

Theorem: [Alon, Yuster, Zwick] There is a k-perfect family of functions $[n] \to [k]$ having size $2^{O(k)} \log n$.

Let $k' := \lceil k \cdot \varrho(k) \rceil$ and let $\mathcal{H}$ be a k'-perfect family of hash functions $[n] \to [k']$.

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

$\binom{[k']}{\leq k}$: all at most k-element subsets of $[k']$

$h^{-1}(T)$: subset of $[n]$ mapped to T by h.

Inapproximability proof

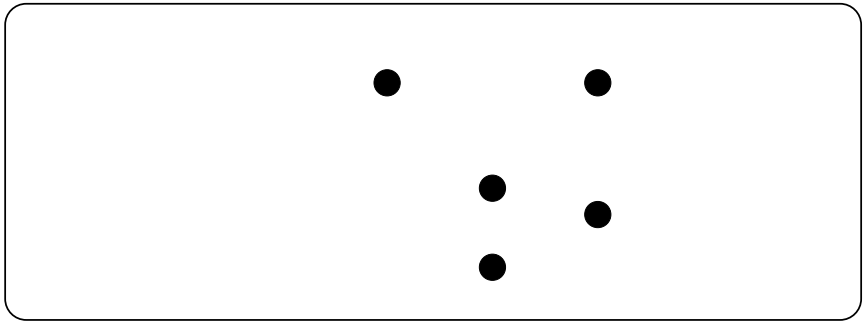
$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

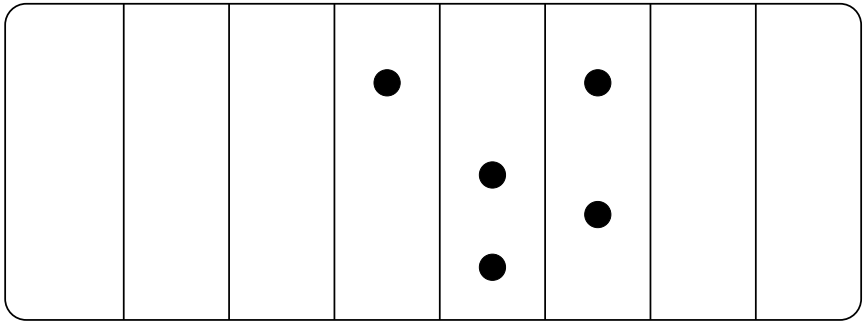


1. Suppose that S is a weight-k solution of C. For every $h \in H$, S gets at most k colors, thus there is a T for which $C(S \cap h^{-1}(T)) = C(S) = 1$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

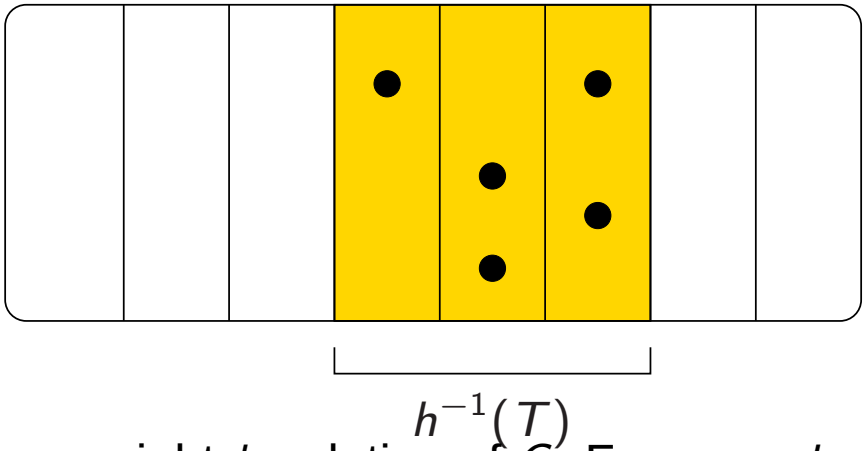


1. Suppose that S is a weight-k solution of C. For every $h \in H$, S gets at most k colors, thus there is a T for which $C(S \cap h^{-1}(T)) = C(S) = 1$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

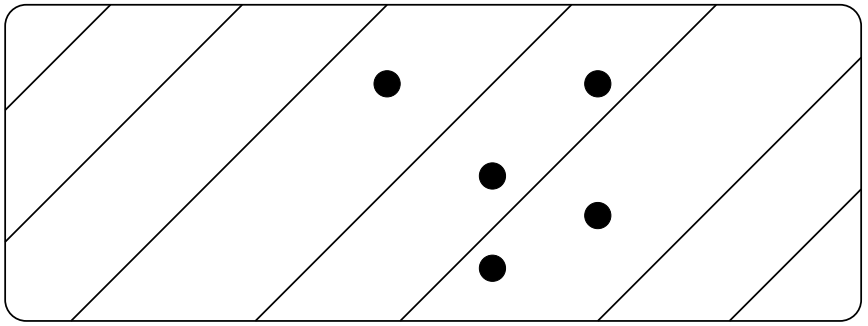


1. Suppose that S is a weight-k solution of C. For every $h \in H$, S gets at most k colors, thus there is a T for which $C(S \cap h^{-1}(T)) = C(S) = 1$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

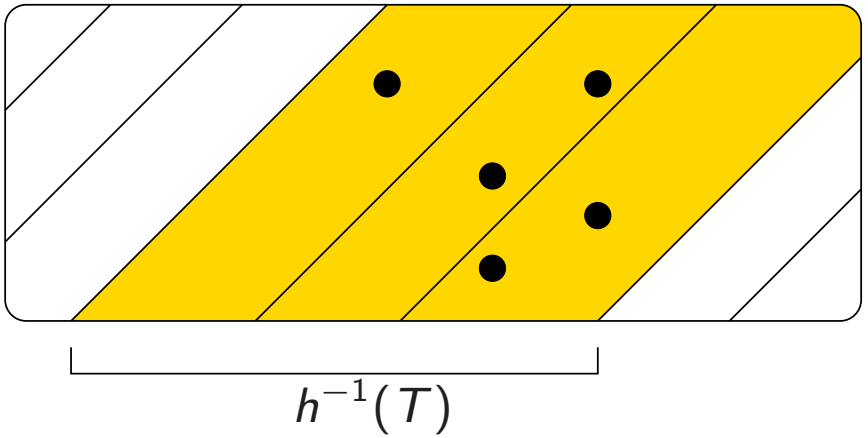


1. Suppose that S is a weight-k solution of C. For every $h \in H$, S gets at most k colors, thus there is a T for which $C(S \cap h^{-1}(T)) = C(S) = 1$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

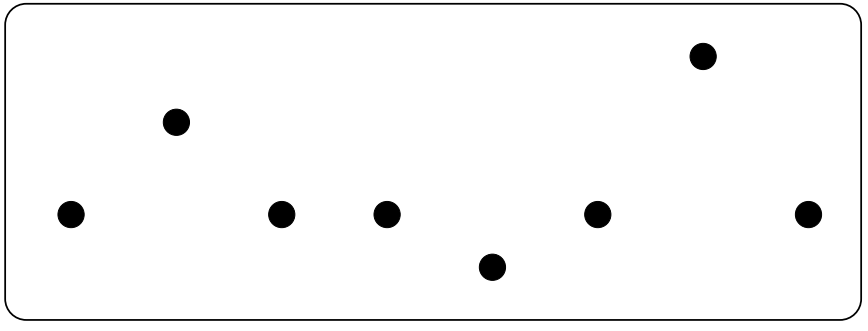


1. Suppose that S is a weight-k solution of C. For every $h \in H$, S gets at most k colors, thus there is a T for which $C(S \cap h^{-1}(T)) = C(S) = 1$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

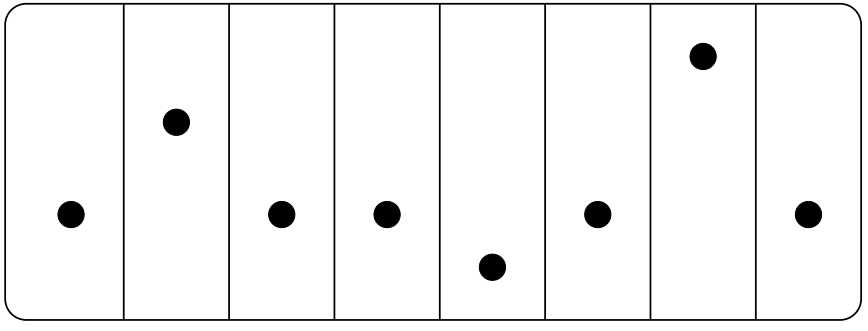


2. Suppose that S' is a weight k' solution of S'. There is a $h \in H$ that is one-to-one on S'. For every T, $|S' \cap h^{-1}(T)| \leq k$, hence $C(S' \cap h^{-1}(T)) = 0$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.

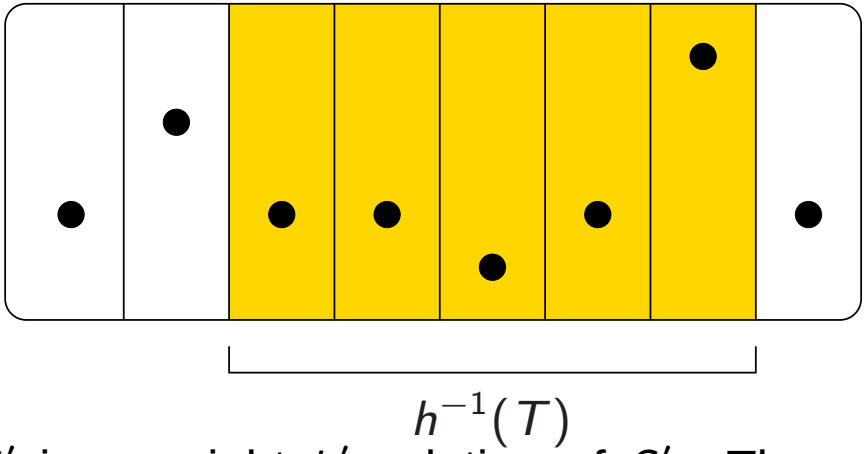


2. Suppose that S' is a weight k' solution of S'. There is a $h \in H$ that is one-to-one on S'. For every T, $|S' \cap h^{-1}(T)| \leq k$, hence $C(S' \cap h^{-1}(T)) = 0$.

Inapproximability proof

$$C'(S) := \bigwedge_{h \in \mathcal{H}} \bigvee_{T \in \binom{[k']}{\leq k}} C(S \cap h^{-1}(T))$$

- C is k-satisfiable $\Rightarrow$ C' is k-satisfiable.
- C is not k-satisfiable $\Rightarrow$ C' is not k'-satisfiable.



2. Suppose that S' is a weight k' solution of S'. There is a $h \in H$ that is one-to-one on S'. For every T, $|S' \cap h^{-1}(T)| \leq k$, hence $C(S' \cap h^{-1}(T)) = 0$.

Approximation schemes

Polynomial approximation scheme (PTAS):

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

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

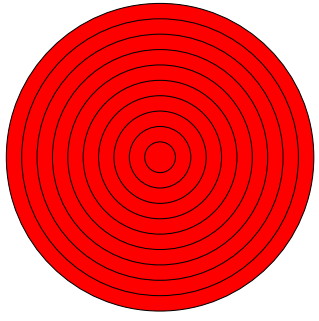
Connections with parameterized complexity:

- 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.

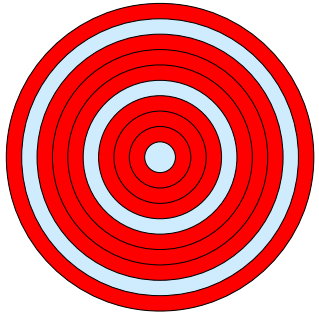
A planar graph can be decomposed into a series of "layers."



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."

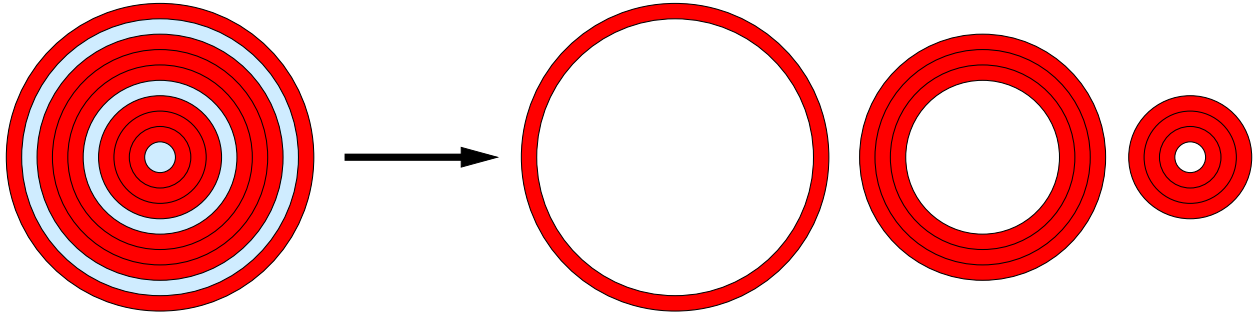


Let $L := 1/\epsilon$. For a fixed $0 \leq s < L$, delete every layer L_i with $i = s \pmod{L}$.

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 $L := 1/\epsilon$. For a fixed $0 \leq s < L$, delete every layer L_i with $i = s \pmod{L}$.

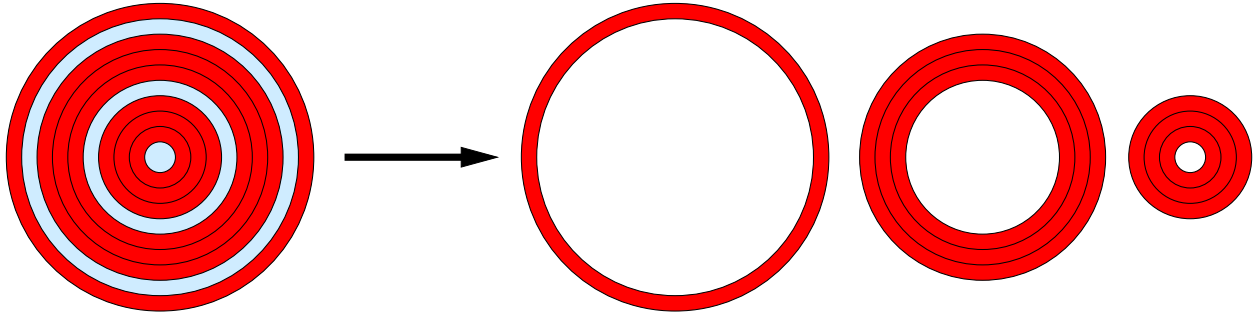
Lemma: [Bodlaender] The treewidth of a k-layer graph is at most $3k + 1$.

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

We do this for every $0 \leq s < L$: 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

SUBGRAPH ISOMORPHISM for planar graphs: given planar graphs H and G, find a copy of H in G as subgraph. Parameter $k := |V(H)|$.



Let $L := k + 1$. For a fixed $0 \leq s < k + 1$, delete every layer L_i with $i = s \pmod{L} \Rightarrow$ the resulting graph has treewidth $3k + 1 \Rightarrow$ INDUCED SUBGRAPH ISOMORPHISM can be solved in time $k^{O(k)} \cdot n$ using treewidth techniques.

We do this for every $0 \leq s < L$: 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/at least 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/at least 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]

Grid tiling

W[1]-hardness proofs for planar/geometric problems can be conveniently done by reduction from the following problem:

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Theorem: GRID TILING is W[1]-hard parameterized by k.

Grid tiling

W[1]-hardness proofs for planar/geometric problems can be conveniently done by reduction from the following problem:

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Theorem: GRID TILING is W[1]-hard parameterized by k.

$(1,1),(1,3),(4,2)$	$(1,5),(4,1),(3,5)$	$(1,1),(4,2),(3,3)$
$(2,2),(4,1)$	$(1,3),(2,1)$	$(2,2),(3,2)$
$(3,2),(3,1),(3,3)$	$(1,1),(3,1)$	$(3,2),(3,5)$

$k = 3$, $D = 5$

Grid tiling

W[1]-hardness proofs for planar/geometric problems can be conveniently done by reduction from the following problem:

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Theorem: GRID TILING is W[1]-hard parameterized by k.

$(1,1), (1,3), (4,2)$	$(1,5), (4,1), (3,5)$	$(1,1), (4,2), (3,3)$
$(2,2), (4,1)$	$(1,3), (2,1)$	$(2,2), (3,2)$
$(3,2), (3,1), (3,3)$	$(1,1), (3,1)$	$(3,2), (3,5)$

$k = 3$, $D = 5$

GRID TILING

Theorem: GRID TILING is W[1]-hard parameterized by k.

Proof: Reduction from k-CLIQUE.

Diagonal sets: $S_{i,i} := \{(d, d) \mid d \in D\}$

For $i \neq j$: $S_{i,j} = \{(u, v) \mid u \text{ and } v \text{ are adjacent vertices}\}$

(...)

Note: It follows in a simple way that MULTICOLORED GRID is W[1]-hard.

Reductions and gadgets

We reduce GRID TILING to various problems using problem-specific gadgets.

- Each gadget represents one of D^2 possible states: each state is a pair.
- Each gadget has a subset S of allowed states.
- Horizontally adjacent gadgets have to agree in the first component.
- Vertically adjacent gadgets have to agree in the second component.

Each gadget has a certain fixed cost, thus reduction from $k \times k$ GRID TILING produces an instance with optimum value $\Theta(k^2)$.

For some $\epsilon = \Theta(1/k^2)$, a $(1+\epsilon)$-approximate solution is actually optimal.

An $f(1/\epsilon) \cdot n^{O(1)}$ EPTAS would imply an $f(k^2) \cdot n^{O(1)}$ algorithm for GRID TILING.

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 of n:

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

Theorem: [Chen et al. 2004] Assuming ETH, there is no $f(k) \cdot n^{o(k)}$ algorithm for k-CLIQUE.

$\Rightarrow$ Assuming ETH, there is no $f(k) \cdot n^{o(k)}$ algorithm for $k \times k$ GRID TILING.

Tighter bounds

we get tighter bounds from ETH using the following argument:

- Assuming ETH, there is no $f(k) \cdot n^{o(k)}$ algorithm for $k \times k$ GRID TILING.
- If $k \times k$ GRID TILING can be reduced to a problem X (using $k \times k$ gadgets), then the $k \times k$ GRID TILING can be solved by running a PTAS for X with $\epsilon = \Theta(1/k^2)$.
- If there is a PTAS with running time $f(1/\epsilon) \cdot n^{o(\sqrt{1/\epsilon})}$ for X, then $k \times k$ GRID TILING can be solved in $f(k) \cdot n^{o(k)}$ time, contradicting ETH.

Theorem: Assuming ETH, there is no $f(1/\epsilon) \cdot n^{o(\sqrt{1/\epsilon})}$ time PTAS for INDEPENDENT SET for unit disks/squares, DOMINATING SET for unit disks/squares, and planar TMIN, TMAX, MPSAT.

Tighter bounds

we get tighter bounds from ETH using the following argument:

- Assuming ETH, there is no $f(k) \cdot n^{o(k)}$ algorithm for $k \times k$ GRID TILING.
- If $k \times k$ GRID TILING can be reduced to a problem X (using $k \times k$ gadgets), then the $k \times k$ GRID TILING can be solved by running a PTAS for X with $\epsilon = \Theta(1/k^2)$.
- If there is a PTAS with running time $f(1/\epsilon) \cdot n^{o(\sqrt{1/\epsilon})}$ for X, then $k \times k$ GRID TILING can be solved in $f(k) \cdot n^{o(k)}$ time, contradicting ETH.

Theorem: Assuming ETH, there is no $f(1/\epsilon) \cdot n^{o(\sqrt{1/\epsilon})}$ time PTAS for INDEPENDENT SET for unit disks/squares, DOMINATING SET for unit disks/squares, and planar TMIN, TMAX, MPSAT.

Note: The best known approximation schemes for this problem have running time $f(1/\epsilon) \cdot n^{O(1)}$.

Even tighter bounds

Can we show that there is no $n^{o(1/\epsilon)}$ PTAS for these problems?

The main problem is that k-CLIQUE is reduced to a problem with $k \times k$ gadgets.

- Can we reduce k-CLIQUE to a smaller GRID TILING?
  No: GRID TILING can be solved in time $n^{O(k)}$.
- The considered problems can be solved in time $n^{O(\sqrt{k})}$, thus **any** reduction from k-CLIQUE should have quadratic blowup.

Even tighter bounds

Can we show that there is no $n^{o(1/\epsilon)}$ PTAS for these problems?

The main problem is that k-CLIQUE is reduced to a problem with $k \times k$ gadgets.

- Can we reduce k-CLIQUE to a smaller GRID TILING?
  No: GRID TILING can be solved in time $n^{O(k)}$.
- The considered problems can be solved in time $n^{O(\sqrt{k})}$, thus **any** reduction from k-CLIQUE should have quadratic blowup.

So far, we only used the fact that the optimum is hard to find, and used this to conclude that for a certain ϵ, it is hard to find a $(1+\epsilon)$-approximation.

Optimization version of GRID TILING

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j} \cup \{\bigstar\}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Goal: Maximize the number of elements that are not $\bigstar$.

Optimization version of GRID TILING

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j} \cup \{\bigstar\}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Goal: Maximize the number of elements that are not $\bigstar$.

$(1,1),(1,3),(4,2)$	$(1,5),(4,1),(3,5)$	$(1,1),(4,2),(3,3)$
$(2,2),(4,1)$	$(1,3),(2,1)$	$(3,1),(3,2)$
$(3,2),(3,1),(3,3)$	$(1,1),(3,1)$	$(3,3),(3,5)$

$k = 3$, $D = 5$

Optimization version of GRID TILING

GRID TILING

Input: A $k \times k$ matrix and a set of pairs $S_{i,j} \subseteq [D] \times [D]$ for each cell.

Find: A value $s_{i,j} \in S_{i,j} \cup \{\bigstar\}$ for each cell such that
– horizontal neighbors agree in the first component,
– vertical neighbors agree in the second component.

Goal: Maximize the number of elements that are not $\bigstar$.

$(1,1),(1,3),(4,2)$	$(1,5),(4,1),(3,5)$	$(1,1),(4,2),(3,3)$
$(2,2),(4,1)$	$(1,3),(2,1)$	$\bigstar$
$(3,2),(3,1),(3,3)$	$(1,1),(3,1)$	$(3,3),(3,5)$

$k = 3$, $D = 5$, value=8

Optimization version of GRID TILING

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 GRID TILING (for any $\delta > 0$).

Optimization version of GRID TILING

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 GRID TILING (for any $\delta > 0$).

The proof uses the PCP theorem to obtain a lower bound on the running time for approximating 3SAT and a "gap-reducing" reduction from 3SAT to GRID TILING:

m vs. αm clauses are satisfiable in m-clause 3SAT

$\Downarrow$

k^2 vs. $k^2 - k + \alpha k$ is the optimum in GRID TILING (roughly)

Optimization version of GRID TILING

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 GRID TILING (for any $\delta > 0$).

The proof uses the PCP theorem to obtain a lower bound on the running time for approximating 3SAT and a "gap-reducing" reduction from 3SAT to GRID TILING:

m vs. αm clauses are satisfiable in m-clause 3SAT

$\Downarrow$

k^2 vs. $k^2 - k + \alpha k$ is the optimum in GRID TILING (roughly)

The result can be transferred to other problems:

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

Conclusions

- Several possible connections to look at.
- There are lots of possibilities for finding new algorithmic results.
- Inapproximability results probably require new approaches.