

# Subexponential Parameterized Algorithms

Geevarghese Philip

*The Institute of Mathematical Sciences, Chennai, India*

IMPECS School on Parameterized and Exact Computation  
Chennai, India  
December 12, 2010

# Outline

## Parameterized Complexity and Bidimensionality

- Parameterized Complexity

- The Planar Separator Theorem

- Treewidth and Grid Minors

- Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

- Partial Cover Problems

## Directed Graphs

- Sparse Directed Graphs

- Feedback Arc Set in Tournaments

# Outline

## Parameterized Complexity and Bidimensionality

### Parameterized Complexity

The Planar Separator Theorem

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

## Introduction

- Two dimensional complexity analysis; the running time as a function of the input size  $n$ , and a parameter  $k$ .
- Pioneered by Rodney Downey and Michael Fellows.
- The parameter is typically (but not necessarily) the solution size. Other natural parameters include maximum degree, size of the alphabet, treewidth, ....
- The aim : design algorithms which run in  $f(k)n^{O(1)}$  time, in contrast to (a typically easy)  $n^{O(k)}$ -time algorithm.
  - Fixed-Parameter Tractable (FPT) algorithms.
  - There is also a hardness theory.
- Why? Practical when the parameter  $k$  is moderately large. For example, can we find
  - an  $O(\log n)$  sized solution in polynomial time? ( $c^k$ )
  - $O(\log^2 n)$  sized solution? ( $c^{\sqrt{k}}$ )



# Parameterized Complexity

Some lower and upper bounds

Many algorithms that run in  $f(k)n^{O(1)}$  time

- $1.3^k$  – Vertex Cover (VC)
- $3.83^k$  – Undirected Feedback Vertex Set (UFVS)
- $k^k$  – Directed feedback vertex set (DFVS)
- .....
  
- For many problems (*Dominating Set, Independent Set,...*) such  $(f(k)n^{O(1)})$  algorithms are **unlikely** (because these problems are  $W$ -hard).
- For some problems like VC, UFVS, DFVS, it is known that  $2^{o(k)}$  algorithms are unlikely under the ETH.

# Parameterized Complexity

Some lower and upper bounds

Many algorithms that run in  $f(k)n^{O(1)}$  time

- $1.3^k$  – Vertex Cover (VC)
- $3.83^k$  – Undirected Feedback Vertex Set (UFVS)
- $k^k$  – Directed feedback vertex set (DFVS)
- .....
  
- For many problems (*Dominating Set, Independent Set,...*) such  $(f(k)n^{O(1)})$  algorithms are **unlikely** (because these problems are *W*-hard).
- For some problems like VC, UFVS, DFVS, it is known that  $2^{o(k)}$  algorithms are unlikely under the ETH.

# Parameterized Complexity

Some lower and upper bounds

Many algorithms that run in  $f(k)n^{O(1)}$  time

- $1.3^k$  – Vertex Cover (VC)
- $3.83^k$  – Undirected Feedback Vertex Set (UFVS)
- $k^k$  – Directed feedback vertex set (DFVS)
- .....
  
- For many problems (*Dominating Set, Independent Set,...*) such  $(f(k)n^{O(1)})$  algorithms are **unlikely** (because these problems are  $W$ -hard).
- For some problems like VC, UFVS, DFVS, it is known that  $2^{o(k)}$  algorithms are unlikely under the ETH.

# This talk is about ...

- There are  $c^{\sqrt{k}}$  time algorithms for vertex cover, dominating set, independent set .... when the input is restricted to **planar graphs** [Alber et al., 2002].
- These are *subexponential* FPT algorithms. For what problems and in what graph classes do such subexponential FPT algorithms exist?
- There is now a 'meta theory' which yields a number of such results. We will take a look at this (and more) in this talk ...

# This talk is about ...

- There are  $c^{\sqrt{k}}$  time algorithms for vertex cover, dominating set, independent set .... when the input is restricted to **planar graphs** [Alber et al., 2002].
- These are *subexponential* FPT algorithms. For what problems and in what graph classes do such subexponential FPT algorithms exist?
- There is now a 'meta theory' which yields a number of such results. We will take a look at this (and more) in this talk ...

# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

**The Planar Separator Theorem**

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# The Planar Separator Theorem

The Theorem (Lipton, Tarjan, 1979)

## Theorem

*Given a planar graph  $G = (V, E)$  on  $n$  vertices, one can, in linear time, find three vertex sets  $A, B$  and  $C$  such that*

- $A \cup B \cup C = V$ ,
- $|B|$  is  $O(\sqrt{n})$ ,
- $|A| \leq 2n/3, |C| \leq 2n/3$ , and
- *there is no edge with one end point in  $A$  and the other end point in  $C$ .*

*$B$  is called a separator of  $G$ .*

# The Planar Separator Theorem

The Theorem (Lipton, Tarjan, 1979)

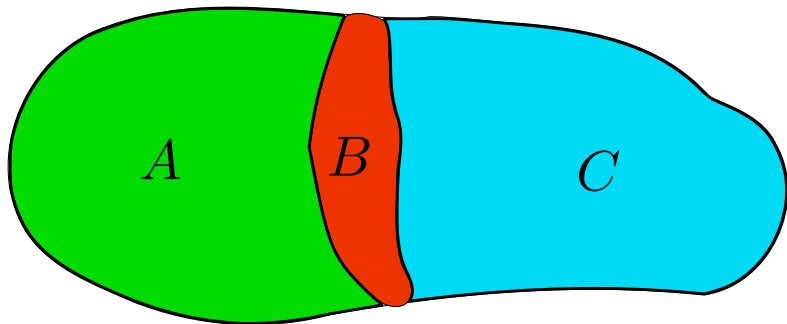


Figure:  $|A|, |C| \leq 2n/3, |B| = O(\sqrt{n})$

# The Planar Separator Theorem

The Theorem (Lipton, Tarjan, 1979)

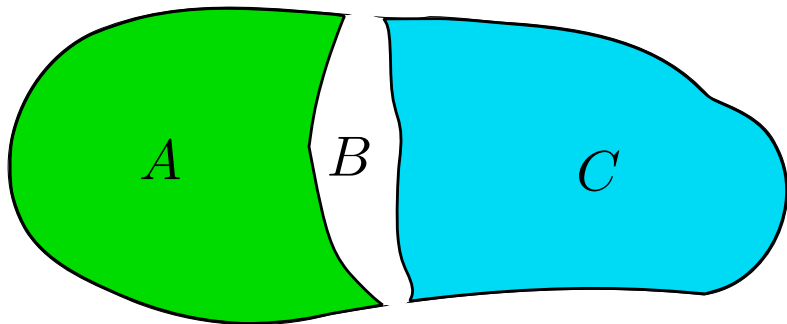


Figure:  $|A|, |C| \leq 2n/3, |B| = O(\sqrt{n})$

# The Planar Separator Theorem

An Application : Planar Maximum Independent Set

- Given a planar graph  $G$  on  $n$  vertices, find an independent set of the maximum size in  $G$ .
- This is NP-hard.
- The theorem yields a  $2^{O(\sqrt{n})}$ -time algorithm.
  - Subexponential in the input size.

# The Planar Separator Theorem

An Application : Planar Maximum Independent Set

Let  $S$  be an (unknown) maximum-size independent set of  $G$ .

- 1 Find an  $O(\sqrt{n})$ -size separator  $B$  of  $G$ .

# The Planar Separator Theorem

An Application : Planar Maximum Independent Set

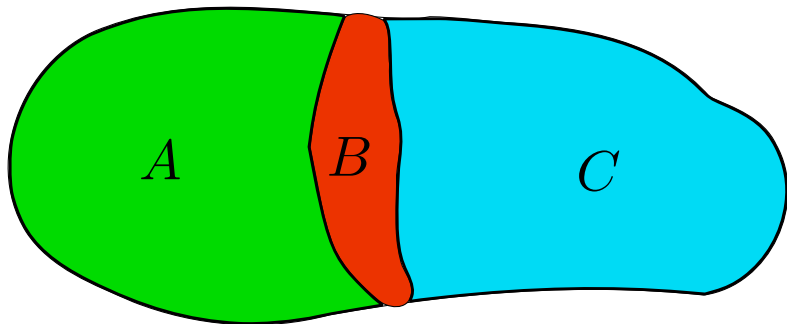


Figure:  $|A|, |C| \leq 2n/3, |B| = O(\sqrt{n})$

# The Planar Separator Theorem

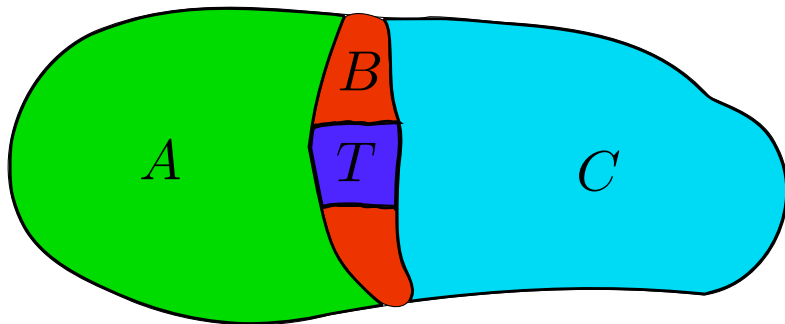
## An Application : Planar Maximum Independent Set

Let  $S$  be an (unknown) maximum-size independent set of  $G$ .

- 1 Find an  $O(\sqrt{n})$ -size separator  $B$  of  $G$ .
- 2 "Guess" the subset  $T = B \cap S$ 
  - Try every subset of  $B$  :  $2^{|B|} = 2^{O(\sqrt{n})}$  subsets,  $2^{O(\sqrt{n})}$  tries.

# The Planar Separator Theorem

An Application : Planar Maximum Independent Set



# The Planar Separator Theorem

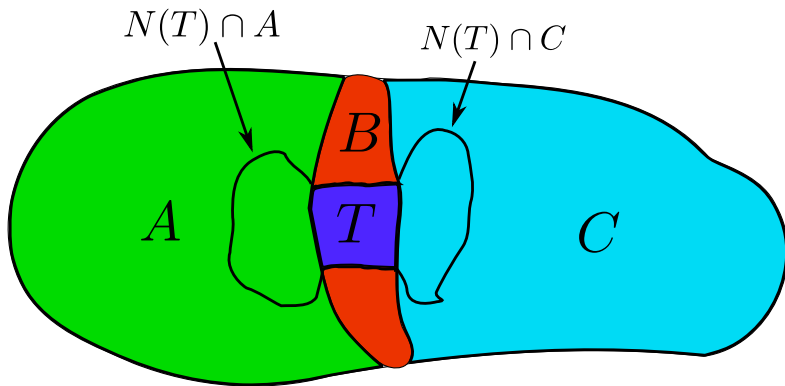
## An Application : Planar Maximum Independent Set

Let  $S$  be an (unknown) maximum-size independent set of  $G$ .

- 1 Find an  $O(\sqrt{n})$ -size separator  $B$  of  $G$ .
- 2 "Guess" the subset  $T = B \cap S$ 
  - Try every subset of  $B$  :  $2^{|B|} = 2^{O(\sqrt{n})}$  time.
- 3 Remove  $B$  from  $G$ 
  - We have correctly guessed the part  $T$  of  $B$  which is in the solution.
- 4 Remove all neighbours of  $T$  from  $G$ 
  - Since  $T$  is in the solution, none of its neighbours can be in the solution.

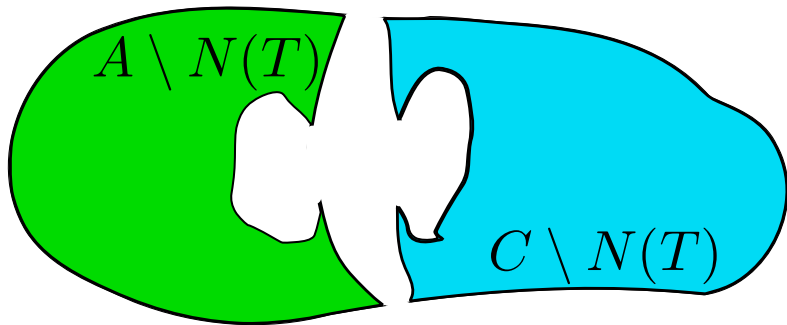
# The Planar Separator Theorem

An Application : Planar Maximum Independent Set



# The Planar Separator Theorem

An Application : Planar Maximum Independent Set



# The Planar Separator Theorem

## An Application : Planar Maximum Independent Set

Let  $S$  be an (unknown) maximum-size independent set of  $G$ .

- 1 Find an  $O(\sqrt{n})$ -size separator  $B$  of  $G$ .
- 2 "Guess" the subset  $T = B \cap S$ 
  - Try every subset of  $B$  :  $2^{|B|} = 2^{O(\sqrt{n})}$  time.
- 3 Remove  $T \cup N(T)$  from  $G$
- 4 Find (recursively) the maximum independent sets  $X$  in  $G[A - N(T)]$  and  $Y$  in  $G[B - N(T)]$ .
- 5  $T \cup X \cup Y$  is a maximum-size independent set in  $G$  for this guess of  $T$ .

# The Planar Separator Theorem

## An Application : Planar Maximum Independent Set

- The running time recurrence is  $T(n) = O(n) + 2^{O(\sqrt{n})} \cdot 2T(2n/3)$ .
  - This evaluates to  $2^{O(\sqrt{n})}$ .
- The algorithm runs in  $2^{O(\sqrt{n})}$  time.
- A similar strategy works also for Planar Dominating Set, Planar Vertex Cover, ...
- Can we solve *parameterized* Planar Independent Set (Given planar  $G$ ,  $k$ , does  $G$  have an independent set of size at least  $k$ ?) in  $2^{o(k)}$  time?

# Subexponential FPT Algorithms

## For Planar Graph Problems

- A  $2^{O(\sqrt{k})}$ -time FPT algorithm for Planar Independent Set comes “for free”.
  - Here  $k$  is the size of the independent set that we are after.
  - Essentially the same as the  $2^{O(\sqrt{n})}$  algorithm, once we notice something ...

# Subexponential FPT Algorithms

For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- How?

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- How?
- If  $k \leq n/4$ , then answer YES
  - Any planar graph on  $n$  vertices has an independent set of size at least  $n/4$ : The Four Colour Theorem.
  - A four-colouring can be found in quadratic time [Robertson, Sanders, Thomas].
- Otherwise,  $n < 4k$ , and we have the linear kernel.

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- How?
- If  $k \leq n/4$ , then answer YES
  - Any planar graph on  $n$  vertices has an independent set of size at least  $n/4$ : The Four Colour Theorem.
- Otherwise,  $n \leq 4k$ .
- So we can assume, without loss of generality, that  $n$  is  $O(k)$ .

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- The  $2^{O(\sqrt{n})}$ -time algorithm thus implies a  $2^{O(\sqrt{k})}$ -time FPT algorithm.



# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- The  $2^{O(\sqrt{n})}$ -time algorithm thus implies a  $2^{O(\sqrt{k})}$ -time FPT algorithm.
- Planar Dominating Set, Vertex Cover, Feedback Vertex Set, all have **linear** kernels.
- Quite non-trivial. The Dominating Set result, for example, is a JACM paper [Alber, Fellows, Niedermeier, 2004].

# Subexponential FPT Algorithms

## For Planar Graph Problems

- Planar Independent Set has a linear **kernel**.
- Given  $(G, k)$ , one can obtain, in polynomial time, an 'equivalent' instance  $(G', k')$  such that  $|G'|$  and  $k'$  are linear functions of  $k$ .
- The  $2^{O(\sqrt{n})}$ -time algorithm thus implies a  $2^{O(\sqrt{k})}$ -time FPT algorithm.
- Planar Dominating Set, Vertex Cover, Feedback Vertex Set, all have **linear** kernels ( $k$  is the solution size).
- These problems (and others) thus have  $2^{O(\sqrt{k})}$ -time FPT algorithms.

# Subexponential FPT Algorithms

What if there is no linear kernel?

- Two ingredients of the  $2^{O(\sqrt{k})}$ -time algorithm for Planar Independent Set:
  - The Planar Separator Theorem: yields a  $2^{O(\sqrt{n})}$ -time algorithm
  - A linear *kernel* in planar graphs: helps us replace  $n$  by  $k$ .
- What if a problem has *no* linear kernel in planar graphs?
  - How do we get a  $2^{o(k)}$ -time algorithm?
  - Clearly, we need a different technique . . .



# Subexponential FPT Algorithms

What if there is no linear kernel?

- Two ingredients of the  $2^{O(\sqrt{k})}$ -time algorithm for Planar Independent Set:
  - The Planar Separator Theorem: yields a  $2^{O(\sqrt{n})}$ -time algorithm
  - A linear *kernel* in planar graphs: helps us replace  $n$  by  $k$ .
- What if a problem has *no* linear kernel in planar graphs?
  - How do we get a  $2^{o(k)}$ -time algorithm?
  - Clearly, we need a different technique . . .

# Subexponential FPT Algorithms

A problem with no linear kernel

## Example (Planar $k$ -path)

Does a planar graph  $G$  have a simple path of length at least  $k$ ?

- Can be solved in  $O(c^k \cdot n^{O(1)})$  time.
  - Colour Coding — [Alon, Yuster, Zwick; JACM 1995].
- Does not have a polynomial (let alone linear) kernel in planar graphs unless  $\text{CoNP} \subseteq \text{NP/Poly}$  [Bodlaender, Downey, Fellows, Hermelin; JCSS 2009].
- We need a different technique to get a  $2^{o(k)}$  algorithm for this problem.



# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

The Planar Separator Theorem

**Treewidth and Grid Minors**

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



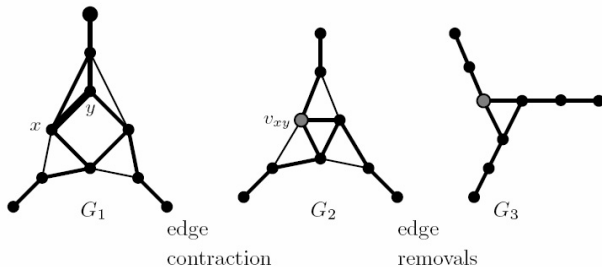
# Treewidth

## Approximation in FPT time

- A measure of how “tree-like” a graph is.
- Given  $G$  and  $k$ , the problem of finding if  $tw(G) \leq k$  is NP-hard.
- Constant-factor approximation in FPT time [Amir, Algorithmica 2010]:
  - Input: Graph  $G$ ,  $k \in \mathbb{N}$ .
  - Output: A tree decomposition of  $G$  of width at most  $3\frac{2}{3}k$ , or:  $G$  has treewidth more than  $k$ .
  - Running time:  $O(2^{3.7k} k^3 \cdot n^3 \lg^4 n)$

# Graph Contractions and Graph Minors

Edge contraction, edge removal



# Graph Contractions and Graph Minors

## Definitions

- A graph  $H$  is a *contraction* of graph  $G$  ( $H \leq_c G$ ) if  $H$  can be obtained from  $G$  by applying a series of edge contractions.
- $H$  is a *minor* of  $G$  ( $H \leq_m G$ ) if  $H$  is a contraction of some subgraph of  $G$ .

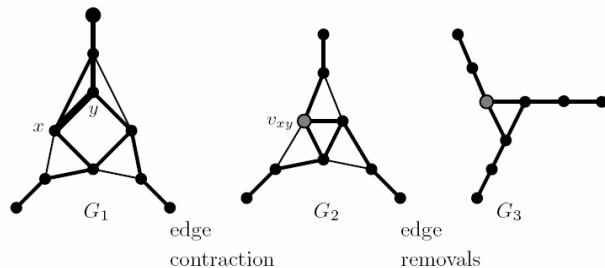
# Graph Contractions and Graph Minors

## Definitions

- A graph  $H$  is a *contraction* of graph  $G$  ( $H \leq_c G$ ) if  $H$  can be obtained from  $G$  by applying a series of edge contractions.
- $H$  is a *minor* of  $G$  ( $H \leq_m G$ ) if  $H$  is a contraction of some subgraph of  $G$ .

# Graph Contractions and Graph Minors

## Examples



$G_3 \leq_m G_2 \leq_m G_1$ ,  $G_2 \leq_c G_1$ , but  $G_3 \not\leq_c G_2$  and  $G_3 \not\leq_c G_1$

# Graph Contractions and Graph Minors

Minor-closed (Contraction-closed) graph classes

- A graph class  $\mathcal{G}$  is minor (contraction) closed if any minor (contraction) of a graph in  $\mathcal{G}$  is again in  $\mathcal{G}$ .
- A graph  $G$  is  $H$ -minor-free when it does not contain  $H$  as a minor.
- A graph class  $\mathcal{G}$  is  $H$ -minor-free (or excludes  $H$  as a minor) when *all* its members are  $H$ -minor-free.

# Graph Contractions and Graph Minors

Minor-closed (Contraction-closed) graph classes

- A graph class  $\mathcal{G}$  is minor (contraction) closed if any minor (contraction) of a graph in  $\mathcal{G}$  is again in  $\mathcal{G}$ .
- A graph  $G$  is  $H$ -minor-free when it does not contain  $H$  as a minor.
- A graph class  $\mathcal{G}$  is  $H$ -minor-free (or excludes  $H$  as a minor) when *all* its members are  $H$ -minor-free.

# Graph Contractions and Graph Minors

Minor-closed (Contraction-closed) graph classes

- A graph class  $\mathcal{G}$  is minor (contraction) closed if any minor (contraction) of a graph in  $\mathcal{G}$  is again in  $\mathcal{G}$ .
- A graph  $G$  is  $H$ -minor-free when it does not contain  $H$  as a minor.
- A graph class  $\mathcal{G}$  is  $H$ -minor-free (or excludes  $H$  as a minor) when *all* its members are  $H$ -minor-free.

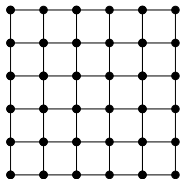
# $H$ -minor-free Graph Classes

## Examples

- ▶ Forests:  $K_3$
- ▶ Outerplanar Graphs:  $K_{2,3}$ ,  $K_4$
- ▶ Planar Graphs:  $K_{3,3}$ ,  $K_5$  (Kuratowski, Wagner)
- ▶ Graphs of treewidth at most  $w$ : Depends on  $w$

# Treewidth and the Grid as a Minor

## The Grid



**Figure:** The  $6 \times 6$  grid. The  $\ell \times \ell$  grid has treewidth exactly  $\ell$ .

# Treewidth and the Grid as a Minor

- Let  $\ell \geq 1$  be an integer. Every planar graph of treewidth  $\geq 6\ell$  contains  $\begin{array}{|c|c|c|c|} \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \square & \square & \square & \square \\ \hline \end{array}_\ell$  as a minor [Robertson, Seymour, Thomas; 1994].
- Holds for *any* class of graphs that excludes a fixed graph as a minor ( $H$ -minor free graphs) [Demaine, Hajighayi; Combinatorica, 2008].
  - The constant is not 6, but depends on  $H$ .

# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

The Planar Separator Theorem

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# A $2^{O(\sqrt{k})}$ -time algorithm for $k$ -path

In  $H$ -minor-free graphs

- *Observation:* If  $G$  has no  $k$ -path, then no minor of  $G$  has a  $k$ -path. Equivalently: if a minor of  $G$  has a  $k$ -path, then so does  $G$ .
- **Algorithm:**
  - If  $\text{treewidth}(G) > c\sqrt{k}$ , then answer YES — by the Grid Theorem,  $G$  has a  $\sqrt{k} \times \sqrt{k}$  grid minor, which in turn has a  $k$ -path.
  - Else: solve the problem using dynamic programming over a tree decomposition of  $G$  of width at most  $c\sqrt{k}$ .
- Runs in  $2^{O(\sqrt{k})}$  time.

# A $2^{O(\sqrt{k})}$ -time algorithm for $k$ -path

In  $H$ -minor-free graphs

- *Observation:* If  $G$  has no  $k$ -path, then no minor of  $G$  has a  $k$ -path. Equivalently: if a minor of  $G$  has a  $k$ -path, then so does  $G$ .
- **Algorithm:**
  - If  $\text{treewidth}(G) > c\sqrt{k}$ , then answer YES — by the Grid Theorem,  $G$  has a  $\sqrt{k} \times \sqrt{k}$  grid minor, which in turn has a  $k$ -path.
  - Else: solve the problem using dynamic programming over a tree decomposition of  $G$  of width at most  $c\sqrt{k}$ .
- Runs in  $2^{O(\sqrt{k})}$  time.

# A $2^{O(\sqrt{k})}$ -time algorithm for $k$ -path

In  $H$ -minor-free graphs

- *Observation:* If  $G$  has no  $k$ -path, then no minor of  $G$  has a  $k$ -path. Equivalently: if a minor of  $G$  has a  $k$ -path, then so does  $G$ .
- **Algorithm:**
  - If  $\text{treewidth}(G) > c\sqrt{k}$ , then answer YES — by the Grid Theorem,  $G$  has a  $\sqrt{k} \times \sqrt{k}$  grid minor, which in turn has a  $k$ -path.
  - Else: solve the problem using dynamic programming over a tree decomposition of  $G$  of width at most  $c\sqrt{k}$ .
- Runs in  $2^{O(\sqrt{k})}$  time.

# A $2^{O(\sqrt{k})}$ -time algorithm for $k$ -path

In  $H$ -minor-free graphs

- *Observation:* If  $G$  has no  $k$ -path, then no minor of  $G$  has a  $k$ -path. Equivalently: if a minor of  $G$  has a  $k$ -path, then so does  $G$ .
- **Algorithm:**
  - If treewidth  $(G) > c\sqrt{k}$ , then answer YES — by the Grid Theorem,  $G$  has a  $\sqrt{k} \times \sqrt{k}$  grid minor, which in turn has a  $k$ -path.
  - Else: solve the problem using dynamic programming over a tree decomposition of  $G$  of width at most  $c\sqrt{k}$ .
- Runs in  $2^{O(\sqrt{k})}$  time.

# Bidimensional Graph Parameters

- Notion introduced by Demaine and Hajiaghayi.
- These are parameters on graphs that
  - are  $\Omega(k^2)$  on  $k \times k$  grids and
  - do not increase when taking minors (edge deletions and contractions)
- Examples:
  - Longest Path
  - Minimum Vertex cover,
  - Minimum Feedback Vertex Set ...
- All these problems have subexponential parameterized algorithms in  $H$ -minor-free graphs.

# Bidimensionality: Subexponential Algorithms

In  $H$ -minor-free graphs

## Algorithm (Demaine and Hajiaghayi):

- 1 If  $\text{treewidth}(G) \leq c\sqrt{k}$ , then find the answer using dynamic programming over a tree decomposition of small width.
- 2 Else: answer YES (or NO) because of existence of a large grid minor

**Key Requirements:** The parameter:

- is  $\Omega(k^2)$  on  $k \times k$  grids,
- does not increase on taking minors
- can be computed optimally in  $2^{o(k^2)}$  time.



# Contraction Bidimensionality

## The Dominating Set problem

What about the parameter Domination Number?

- While it is large on a grid, it *does increase* on subgraphs.
- It is not minor closed, but is *contraction*-closed.

# Contraction Bidimensionality

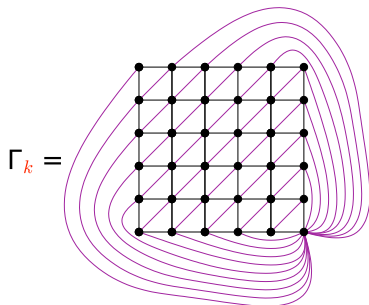
## The Dominating Set problem

What about the parameter Domination Number?

- While it is large on a grid, it *does increase* on subgraphs.
- It is not minor closed, but is *contraction*-closed.

# A Grid Theorem for Contraction Bidimensionality

The graph  $\Gamma_k$  is:



Theorem (Fomin, Golovach, Thilikos ESA 2009)

There is a function  $\phi : \mathbb{N} \rightarrow \mathbb{N}$  such that for every graph  $G$  excluding a fixed  $h$ -vertex apex graph  $H$  as contraction the following holds:

► if  $tw(G) \geq \phi(h) \cdot k$  then  $\Gamma_k \leq_c G$ .

# Contraction Bidimensionality: Subexponential Parameterized Algorithms

- A contraction-closed parameter is *bidimensional* if it is  $\Omega(k^2)$  on  $\Gamma_k$ .
- Thus we get subexponential FPT algorithms for contraction closed bidimensional parameters (like dominating set,  $r$ -dominating set, connected dominating set...) in apex-minor free graphs.

# Contraction Bidimensionality: Subexponential Parameterized Algorithms

- A contraction-closed parameter is *bidimensional* if it is  $\Omega(k^2)$  on  $\Gamma_k$ .
- Thus we get subexponential FPT algorithms for contraction closed bidimensional parameters (like dominating set,  $r$ -dominating set, connected dominating set...) in apex-minor free graphs.

# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

The Planar Separator Theorem

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not* bidimensional : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not* bidimensional : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not bidimensional* : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not* bidimensional : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not* bidimensional : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Beyond Bidimensional Parameters

What about parameters that are not bidimensional? (For example, Partial Dominating Set, Partial Vertex Cover)?

- Find the maximum number of vertices that can be dominated or the edges to be covered using at most  $k$  vertices? ( $k$  is the parameter)
- W-hard in general graphs.
- *Not* bidimensional : not large on a grid.
- Subexponential parameterized algorithms using the “irrelevant vertex” technique (Fomin, Lokshtanov, Raman, Saurabh; FSTTCS 2009).
- Keep deleting irrelevant vertices. When not possible, argue that the resultant graph has bounded treewidth; works for apex minor free graphs.

# Planar Partial Vertex Cover

## The Key Lemma

*Lemma:* Let  $G = (V, E)$  be an undirected graph with vertices  $(v_1, v_2, \dots, v_n)$  ordered in non-increasing order of their degrees. Let  $C \subseteq V$  be the *lexicographically smallest* solution with the last vertex in  $C$  being vertex  $v_s$ . Then  $C$  is a dominating set of size at most  $k$  for  $G[\{v_1, v_2, \dots, v_s\}]$ .



*Proof:* Suppose not: there is some vertex  $v_i$  with  $i < s$  not dominated by  $C$  (i.e. has no neighbours in  $C$ ). Replacing  $v_s$  with  $v_i$  gives a lexicographically smaller solution.

# Planar Partial Vertex Cover

## The Key Lemma

*Lemma:* Let  $G = (V, E)$  be an undirected graph with vertices  $(v_1, v_2, \dots, v_n)$  ordered in non-increasing order of their degrees. Let  $C \subseteq V$  be the *lexicographically smallest* solution with the last vertex in  $C$  being vertex  $v_s$ . Then  $C$  is a dominating set of size at most  $k$  for  $G[\{v_1, v_2, \dots, v_s\}]$ .



*Proof:* Suppose not: there is some vertex  $v_i$  with  $i < s$  not dominated by  $C$  (i.e. has no neighbours in  $C$ ). Replacing  $v_s$  with  $v_i$  gives a lexicographically smaller solution.

# Consequence of the Key Lemma

*Corollary:* If  $G[\{v_1, v_2, \dots, v_n\}]$  has no dominating set of size at most  $k$ , then  $v_n$  is not part of the lexicographically smallest solution.

It's a WIN-WIN situation: If  $G$  has a dominating set of size at most  $k$ , then it has treewidth  $O(\sqrt{k})$ .

We test for a dominating set of size at most  $k$  using the  $2^{O(\sqrt{k})} \text{poly}(n)$  algorithm (from contraction bidimensionality).

# Consequence of the Key Lemma

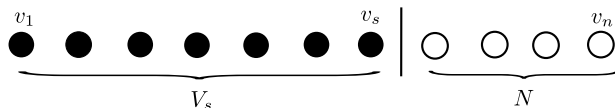
*Corollary:* If  $G[\{v_1, v_2, \dots, v_n\}]$  has no dominating set of size at most  $k$ , then  $v_n$  is not part of the lexicographically smallest solution.

It's a WIN-WIN situation: If  $G$  has a dominating set of size at most  $k$ , then it has treewidth  $O(\sqrt{k})$ .

We test for a dominating set of size at most  $k$  using the  $2^{O(\sqrt{k})} \text{poly}(n)$  algorithm (from contraction bidimensionality).

# The Algorithm

## For Planar Partial Vertex Cover



- 1 Starting from the last vertex of the current set of vertices, say  $v_s$ ,
  - 1 check whether  $G[V_s]$  has a dominating set of size at most  $k$ , if so  $G[V_s]$  has treewidth  $O(\sqrt{k})$ ,
  - 2 if not,  $v_s$  is irrelevant, set  $s = s - 1$  and repeat.
- 2 Remove edges within  $N$ , the set of irrelevant vertices.
- 3 Now  $G$  has a 2-dominating set of size  $k$ , and hence has  $O(\sqrt{k})$  treewidth.
- 4 Complete using dynamic programming.

# Other Planar Partial Cover Problems

- Also works for partial dominating set and partial  $r$ -dominating set (for constant  $r$ ).

# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

The Planar Separator Theorem

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# On Sparse Directed Graphs

- The lack of grid theorems and clear notions of treewidth make subexponential algorithms rare in directed graphs.
- Dorn et al.(STACS 2010) derived subexponential algorithms for two variations of spanning tree (max leaf, and max internal nodes) in directed graphs whose underlying undirected graphs are planar (or apex minor free).
- Two techniques used:
  - 1 Find a solution or find  $o(k^2)$  vertices whose removal makes the underlying undirected graph of constant treewidth. So the original underlying undirected graph has  $o(k)$  treewidth (as we can't destroy a  $k \times k$  grid with  $o(k^2)$  vertices). Apply DP.
  - 2 Reduce the instance into  $2^{o(k)}$  instances (graphs) each of treewidth  $o(k)$  such that the original instance is an YES instance if and only if one of these instances is.

# Outline

## Parameterized Complexity and Bidimensionality

Parameterized Complexity

The Planar Separator Theorem

Treewidth and Grid Minors

Subexponential algorithms for  $k$ -path

## Beyond Bidimensionality

Partial Cover Problems

## Directed Graphs

Sparse Directed Graphs

Feedback Arc Set in Tournaments



# A Subexponential Algorithm for FAST

Fast FAST [Alon, Lokshantov, Saurabh; ICALP 2009]

*Problem:* Given a tournament  $T$  on  $n$  vertices, does it have at most  $k$  arcs whose reversal makes it *acyclic*?

- Is NP-complete
- The previous best fixed parameter algorithm:  
 $O(2.5^k + n^{O(1)})$
- Has an  $O(k^2)$  kernel.

This paper gives an  $2^{O(\sqrt{k} \log^2 k)} + n^{O(1)}$  algorithm using a variation of Colour Coding.

# A Subexponential Algorithm for FAST

## Outline

- 1 Kernelize the tournament  $T$  to have  $O(k^2)$  vertices.
- 2 Randomly colour the vertices of  $T$  using colours  $1, 2, \dots, \sqrt{8k}$ .
- 3 Find a feedback arc set of  $T$  of size at most  $k$ , if exists, such that no arc is *monochromatic*.
- 4 If not found, repeat

**Theorem:** If  $T$  has a FAS with  $k$  arcs, then the probability that all the  $k$  arcs are non-monochromatic is at least  $2^{-c\sqrt{k}}$ .

*Proof:* Consider the subgraph  $H$  of  $T$  on the  $k$  arcs of the FAS.  $H$  is  $\sqrt{2k}$ -degenerate. Arrange the vertices of  $H$  in the degeneracy order (all  $d_i \leq \sqrt{2k}$ ).

Then the required probability is

$$\prod_{i=1}^n \left(1 - \frac{d_i}{\sqrt{8k}}\right) \geq (2e)^{-\sqrt{k/8}}$$

# A Subexponential Algorithm for FAST

## Outline

- 1 Kernelize the tournament  $T$  to have  $O(k^2)$  vertices.
- 2 Randomly colour the vertices of  $T$  using colours  $1, 2, \dots, \sqrt{8k}$ .
- 3 Find a feedback arc set of  $T$  of size at most  $k$ , if exists, such that no arc is *monochromatic*.
- 4 If not found, repeat

**Theorem:** If  $T$  has a FAS with  $k$  arcs, then the probability that all the  $k$  arcs are non-monochromatic is at least  $2^{-c\sqrt{k}}$ .

*Proof:* Consider the subgraph  $H$  of  $T$  on the  $k$  arcs of the FAS.  $H$  is  $\sqrt{2k}$ -degenerate. Arrange the vertices of  $H$  in the degeneracy order (all  $d_i \leq \sqrt{2k}$ ).

Then the required probability is

$$\prod_{i=1}^n \left(1 - \frac{d_i}{\sqrt{8k}}\right) \geq (2e)^{-\sqrt{k/8}}$$

# A Subexponential Algorithm for FAST

## Outline

- 1 Kernelize the tournament  $T$  to have  $O(k^2)$  vertices.
- 2 Randomly colour the vertices of  $T$  using colours  $1, 2, \dots, \sqrt{8k}$ .
- 3 Find a feedback arc set of  $T$  of size at most  $k$ , if exists, such that no arc is *monochromatic*.
- 4 If not found, repeat

**Theorem:** If  $T$  has a FAS with  $k$  arcs, then the probability that all the  $k$  arcs are non-monochromatic is at least  $2^{-c\sqrt{k}}$ .

*Proof:* Consider the subgraph  $H$  of  $T$  on the  $k$  arcs of the FAS.  $H$  is  $\sqrt{2k}$ -degenerate. Arrange the vertices of  $H$  in the degeneracy order (all  $d_i \leq \sqrt{2k}$ ).

Then the required probability is

$$\prod_{i=1}^n \left(1 - \frac{d_i}{\sqrt{8k}}\right) \geq (2e)^{-\sqrt{k/8}}$$

# A Subexponential Algorithm for FAST

## Outline

- 1 Kernelize the tournament  $T$  to have  $O(k^2)$  vertices.
- 2 Randomly colour the vertices of  $T$  using colours  $1, 2, \dots, \sqrt{8k}$ .
- 3 Find a feedback arc set of  $T$  of size at most  $k$ , if exists, such that no arc is *monochromatic*.
- 4 If not found, repeat

**Theorem:** If  $T$  has a FAS with  $k$  arcs, then the probability that all the  $k$  arcs are non-monochromatic is at least  $2^{-c\sqrt{k}}$ .

*Proof:* Consider the subgraph  $H$  of  $T$  on the  $k$  arcs of the FAS.  $H$  is  $\sqrt{2k}$ -degenerate. Arrange the vertices of  $H$  in the degeneracy order (all  $d_i \leq \sqrt{2k}$ ).

Then the required probability is

$$\prod_{i=1}^n \left(1 - \frac{d_i}{\sqrt{8k}}\right) \geq (2e)^{-\sqrt{k/8}}$$

# Finding a Colourful FAS in a Coloured Tournament

Given a  $t = \sqrt{8k}$ -coloured tournament, find a minimum FAS where all the arcs are non mono-chromatic.

*Observation:* Minimum FAS can be characterized by an ordering of the vertices that minimize the number of 'backward' arcs.

*Divide and Conquer:*

- Guess the vertices in first half
- Find the best ordering among them
- Find the best ordering among the second half

$$T(n) \leq \binom{n}{n/2} 2T(n/2) \text{ which solves to about } 2^{2n}$$

# Finding a Colourful FAS in a Coloured Tournament

But note that in a “properly” coloured tournament,

- The subtournament induced on each colour class is acyclic; hence has a unique topological ordering.
- If a vertex  $v$  of a certain colour appears in the first half, then all vertices of the same colour that appear before  $v$  in the topological order appear in the first half.
- So the first half is characterized by the ‘last vertex’ of each colour class.
- Guess these “last” vertices . . .
- The running time recurrence is  $T(n) \leq n^t 2T(n/2)$ , which solves to  $n^{t \log n}$ .
- Since  $n = O(k^2)$ ,  $t = O(\sqrt{k})$ , the algorithm runs in  $k^{O(\sqrt{k} \log k)} = 2^{O(\sqrt{k} \log^2 k)}$  time.



# Derandomization

- We need a family of functions  $F = \{f : [k^2] \rightarrow [\sqrt{8k}]\}$  such that
  - For any graph on  $k^2$  vertices and  $k$  edges, at least one of the functions colour the graph properly.
- Such a family is called a *universal* colouring family.
- Alon et al. devised a universal colouring family of this form of size  $2^{O(\sqrt{k} \log k)}$
- Using this universal colouring family, the derandomized algorithm runs in  $2^{O(\sqrt{k} \log^2 k)} n^{O(1)}$  time.
- Later improved by Feige to  $2^{O(\sqrt{k})} n^{O(1)}$  .

# Summarizing

- Bidimensionality (and its variations) have proved useful in obtaining subexponential algorithms for a number of graph problems in sparse graphs.
- Subexponential algorithms in directed graphs are in a nascent stage.
- Some open problems:
  - Subexponential FPT algorithm for Steiner tree in planar graphs ( $k$  is the number of terminals).
  - Subexponential FPT algorithm for directed  $k$ -path in planar digraphs.

Thank You!