

Parameterized Algorithms

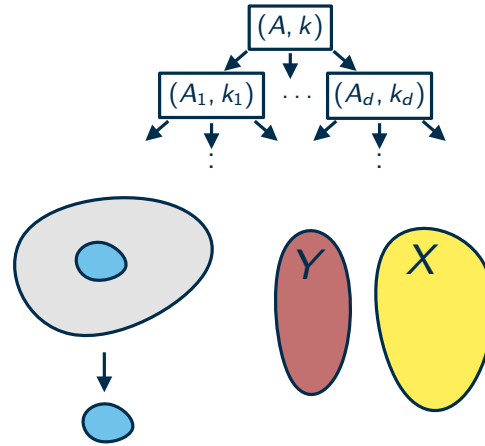
Lower Bounds: ETH and SETH

Thomas Bläsius

Content

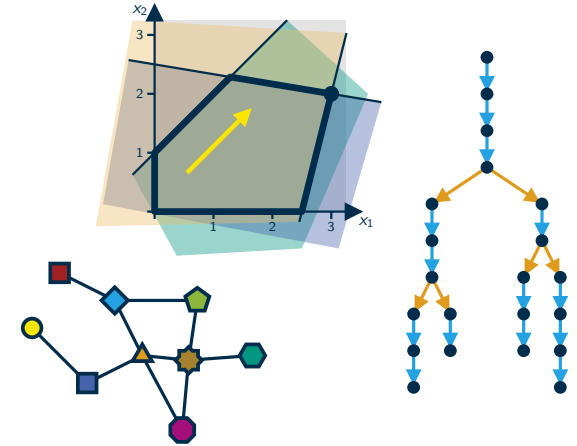
Basic toolbox

- bounded search trees
- kernelization
- iterative compression



Extended toolbox

- linear programs
- branch-and-reduce
- color coding



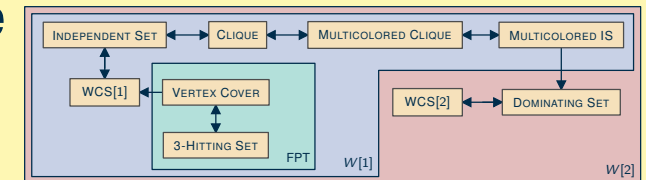
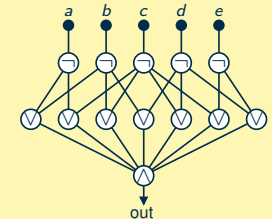
Tree width

- dynamic programming
- chordal and planar graphs
- Courcelle's theorem



Lower bounds

- kernel lower bounds
- parameterized reductions
- circuits and the W-hierarchy
- ETH and SETH



(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems ← requires stronger assumption than $FPT \neq W[1]$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $FPT \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $FPT \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$: $\delta_q \leq 1$, as there is an $2^{n+o(n)}$ -algorithm for every q

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$: $\delta_q \leq 1$, as there is an $2^{n+o(n)}$ -algorithm for every q

ETH: $\delta_3 > 0$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$: $\delta_q \leq 1$, as there is an $2^{n+o(n)}$ -algorithm for every q

ETH: $\delta_3 > 0$

meaning: no sub-exponential algo for 3-SAT

believe: the hypothesis is true

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $\text{FPT} \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in \text{P}$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$: $\delta_q \leq 1$, as there is an $2^{n+o(n)}$ -algorithm for every q

ETH: $\delta_3 > 0$

meaning: no sub-exponential algo for 3-SAT

believe: the hypothesis is true

SETH: $\lim_{q \rightarrow \infty} \delta_q = 1$

(Strong) exponential-time hypothesis

Goal

- $n^{\log \log k}$ is not FPT but still much better than n^k
- show: $f(k) \cdot n^{o(k)}$ impossible for some $W[1]$ -problems $\longleftarrow$ requires stronger assumption than $FPT \neq W[1]$

Definition: For $q \geq 2$, let δ_q be the infimum of all $c \in \mathbb{R}$, for which there is an $2^{cn+o(n)}$ algorithm for q -SAT with n variables.

What do we know?

- $q = 2$: $\delta_q = 0$, as $2\text{-SAT} \in P$ and $\text{poly}(n) \in 2^{cn+o(n)}$ for every $c > 0$
- $q > 2$: $\delta_q \leq 1$, as there is an $2^{n+o(n)}$ -algorithm for every q

ETH: $\delta_3 > 0$

meaning: no sub-exponential algo for 3-SAT

believe: the hypothesis is true

SETH: $\lim_{q \rightarrow \infty} \delta_q = 1$

meaning: impossible to beat brute-force for SAT

believe: unclear, but a better algorithm would be a major breakthrough

Some implications

Theorem
SETH implies ETH.

Proof idea

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Some implications

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Theorem
SETH implies ETH.

Proof idea

- reduce SAT to 3-SAT
- formula does not grow so much that something sub-exponential for 3-SAT is exponential for SAT

Some implications

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Theorem
SETH implies ETH.

Proof idea

- reduce SAT to 3-SAT
- formula does not grow so much that something sub-exponential for 3-SAT is exponential for SAT

Theorem
ETH implies $\text{FPT} \neq W[1]$.

Proof idea

Some implications

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Theorem
SETH implies ETH.

Proof idea

- reduce SAT to 3-SAT
- formula does not grow so much that something sub-exponential for 3-SAT is exponential for SAT

Theorem
ETH implies $\text{FPT} \neq W[1]$.

Proof idea

- later: ETH implies lower bound of $n^{\Omega(k)}$ for a $W[1]$ -problem
- directly implies $\text{FPT} \neq W[1]$

Some implications

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Theorem
SETH implies ETH.

Proof idea

- reduce SAT to 3-SAT
- formula does not grow so much that something sub-exponential for 3-SAT is exponential for SAT

Theorem
ETH implies $\text{FPT} \neq W[1]$.

Proof idea

- later: ETH implies lower bound of $n^{\Omega(k)}$ for a $W[1]$ -problem
- directly implies $\text{FPT} \neq W[1]$

Theorem
ETH implies, that there is no $2^{o(n+m)}$ -algorithm for VERTEX COVER, DOMINATING SET, FEED-BACK VERTEX SET, 3-COLORING, or HAMILTONIAN CYCLE.

Proof idea

Some implications

ETH: no sub-exponential algo for 3-SAT
SETH: 2^n is the best one can do for SAT

Theorem
SETH implies ETH.

Proof idea

- reduce SAT to 3-SAT
- formula does not grow so much that something sub-exponential for 3-SAT is exponential for SAT

Theorem
ETH implies $\text{FPT} \neq W[1]$.

Proof idea

- later: ETH implies lower bound of $n^{\Omega(k)}$ for a $W[1]$ -problem
- directly implies $\text{FPT} \neq W[1]$

Theorem
ETH implies, that there is no $2^{o(n+m)}$ -algorithm for VERTEX COVER, DOMINATING SET, FEED-BACK VERTEX SET, 3-COLORING, or HAMILTONIAN CYCLE.

Proof idea

- linear reduction from 3-SAT to these problems

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

What do we want?

- lower bound of the form $f(k) \cdot n^{\Omega(k)}$ for a parameterized problem

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

What do we want?

- lower bound of the form $f(k) \cdot n^{\Omega(k)}$ for a parameterized problem

Rough plan

- reduce a “normal” problem to a parameterized problem

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

What do we want?

- lower bound of the form $f(k) \cdot n^{\Omega(k)}$ for a parameterized problem

Rough plan

- reduce a “normal” problem to a parameterized problem
- let the reduction list possible partial solution and let the target problem “choose” k of them

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

What do we want?

- lower bound of the form $f(k) \cdot n^{\Omega(k)}$ for a parameterized problem

Rough plan

- reduce a “normal” problem to a parameterized problem
- let the reduction list possible partial solution and let the target problem “choose” k of them
- hides part of the difficulty in the input size and another part in the selection of k partial solutions

ETH and parameterization

What do we have?

- lower bound of the form $2^{\Omega(n+m)}$

What do we want?

- lower bound of the form $f(k) \cdot n^{\Omega(k)}$ for a parameterized problem

Rough plan

- reduce a “normal” problem to a parameterized problem
- let the reduction list possible partial solution and let the target problem “choose” k of them
- hides part of the difficulty in the input size and another part in the selection of k partial solutions

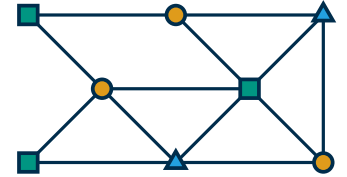
Problem we reduce from: 3-COLORING

- has no output size $\rightarrow$ no temptation to view it as parameterized problem
- local partial solutions can be combined nicely (we’ll see this in a moment)

Choosing compatible partial solutions

Problem: 3-COLORING

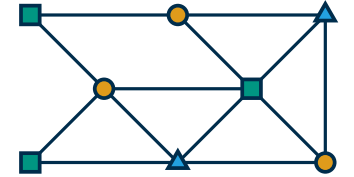
Color vertices of G with three colors such that neighbors have different colors.



Choosing compatible partial solutions

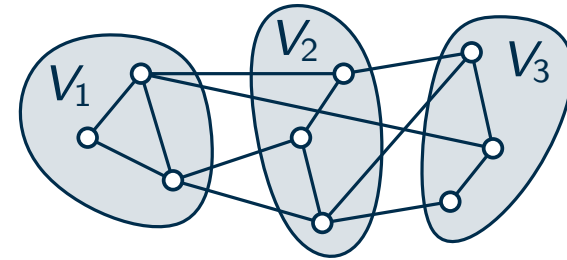
Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.



Partial solutions

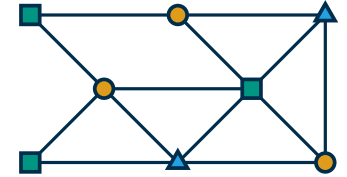
- partition $V = V_1 \cup \dots \cup V_k$



Choosing compatible partial solutions

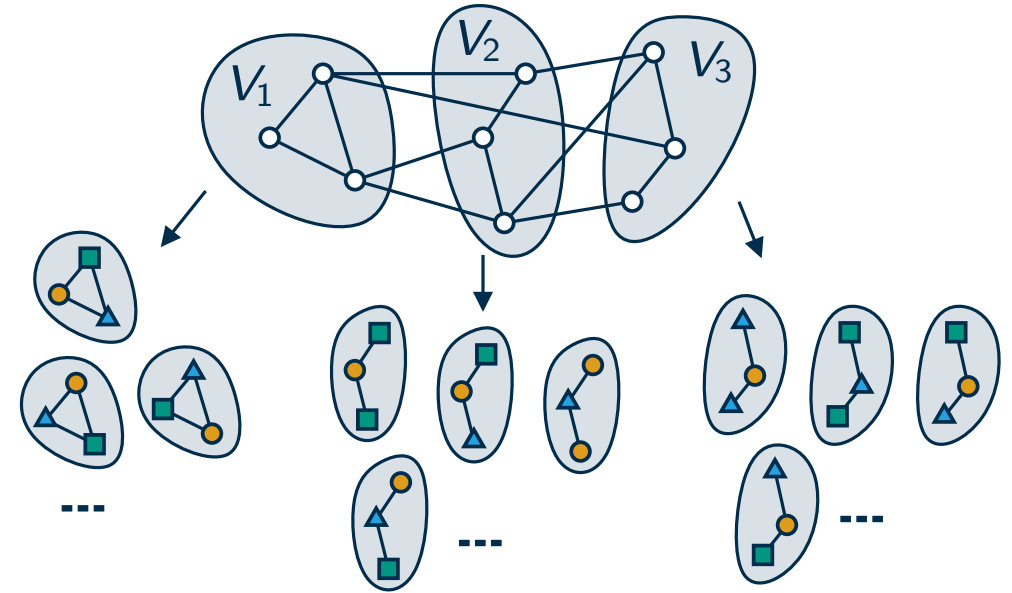
Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.



Partial solutions

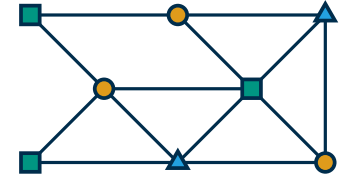
- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$



Choosing compatible partial solutions

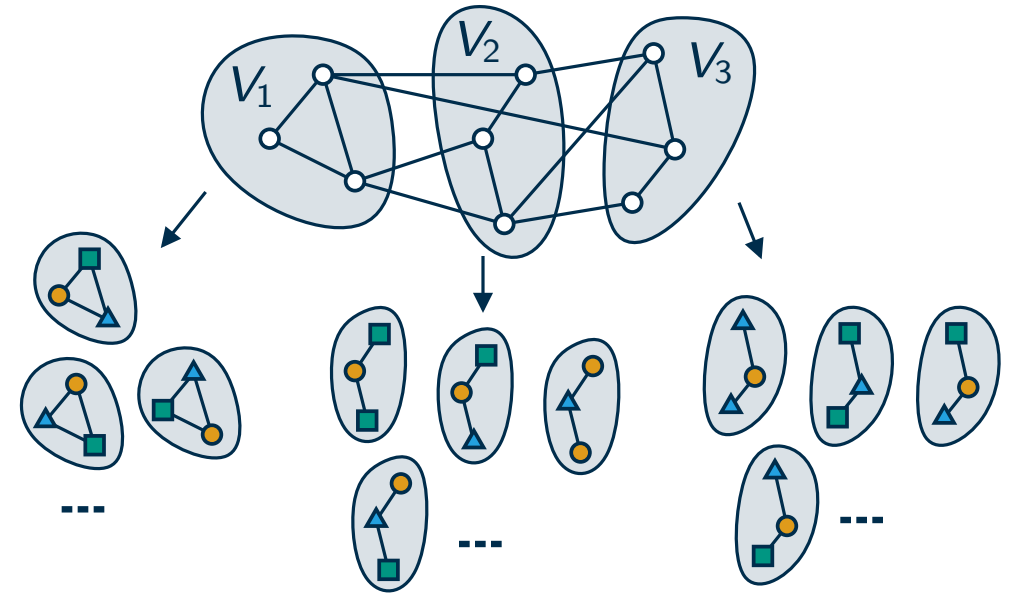
Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.



Partial solutions

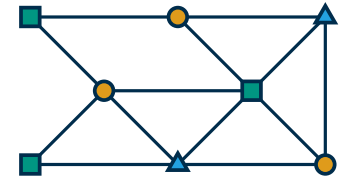
- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$
- choose a coloring $c_i \in C_i$ for every i such that choices are compatible



Choosing compatible partial solutions

Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.

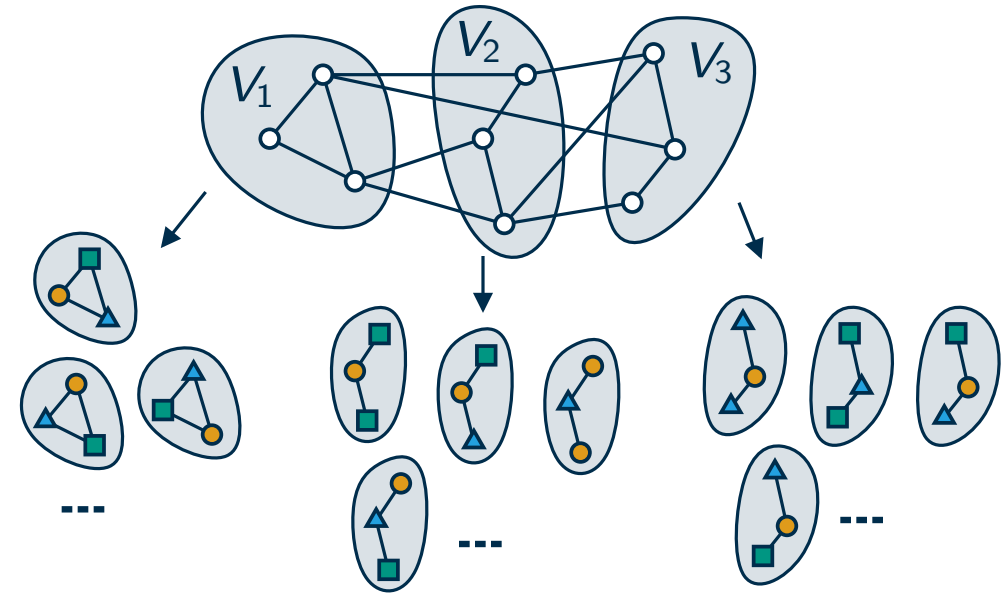


Partial solutions

- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$
- choose a coloring $c_i \in C_i$ for every i such that choices are compatible

Interpretation

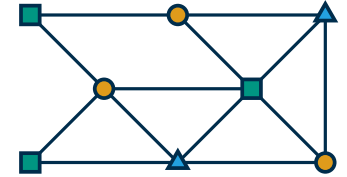
- part of the difficulty is hidden in the problem size (we list many colorings)
- another part lies in the choice for k fitting partial solutions



Choosing compatible partial solutions

Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.

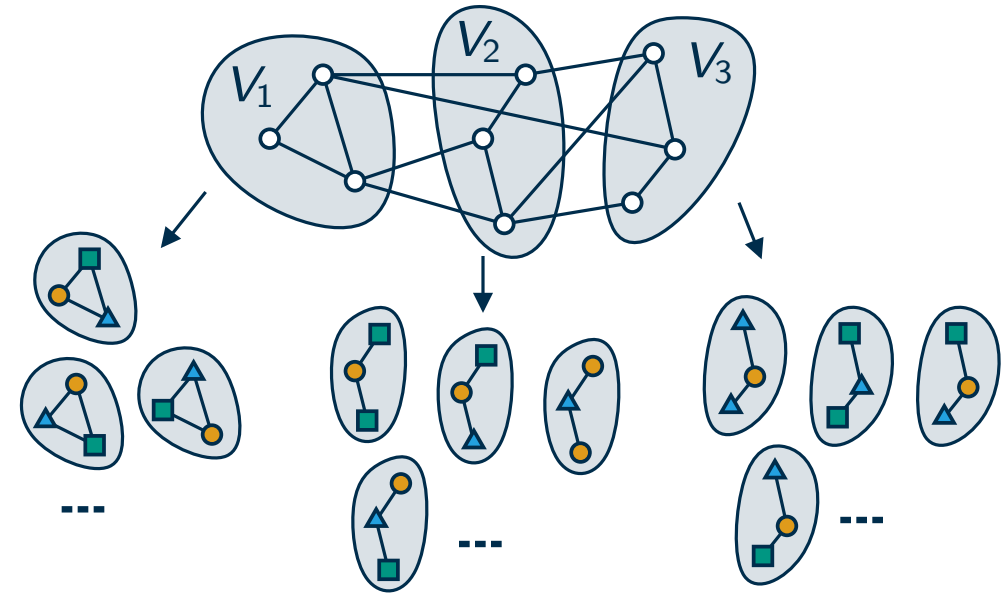


Partial solutions

- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$
- choose a coloring $c_i \in C_i$ for every i such that choices are compatible

Changing perspective

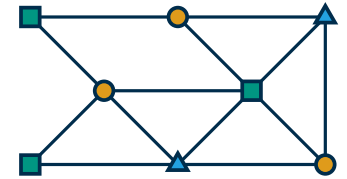
- view $C_1 \cup \dots \cup C_k$ as node set



Choosing compatible partial solutions

Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.

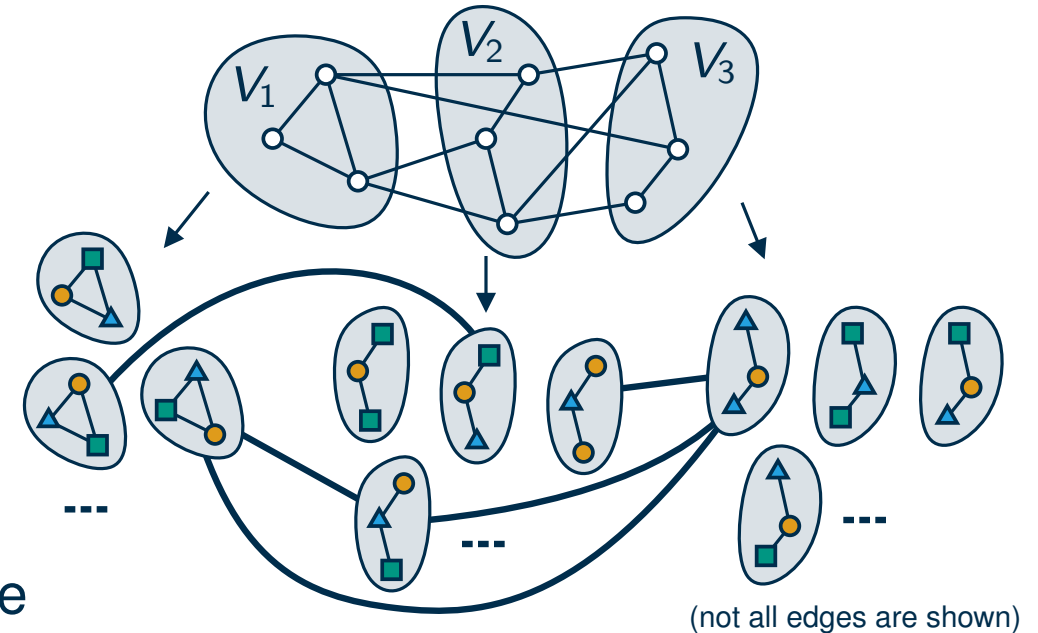


Partial solutions

- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$
- choose a coloring $c_i \in C_i$ for every i such that choices are compatible

Changing perspective

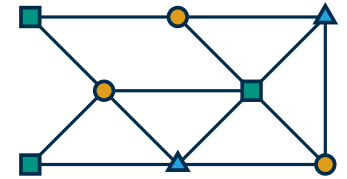
- view $C_1 \cup \dots \cup C_k$ as node set
- add edge $\{c_i, c_j\}$ if $c_i \in C_i$ and $c_j \in C_j$ are compatible



Choosing compatible partial solutions

Problem: 3-COLORING

Color vertices of G with three colors such that neighbors have different colors.

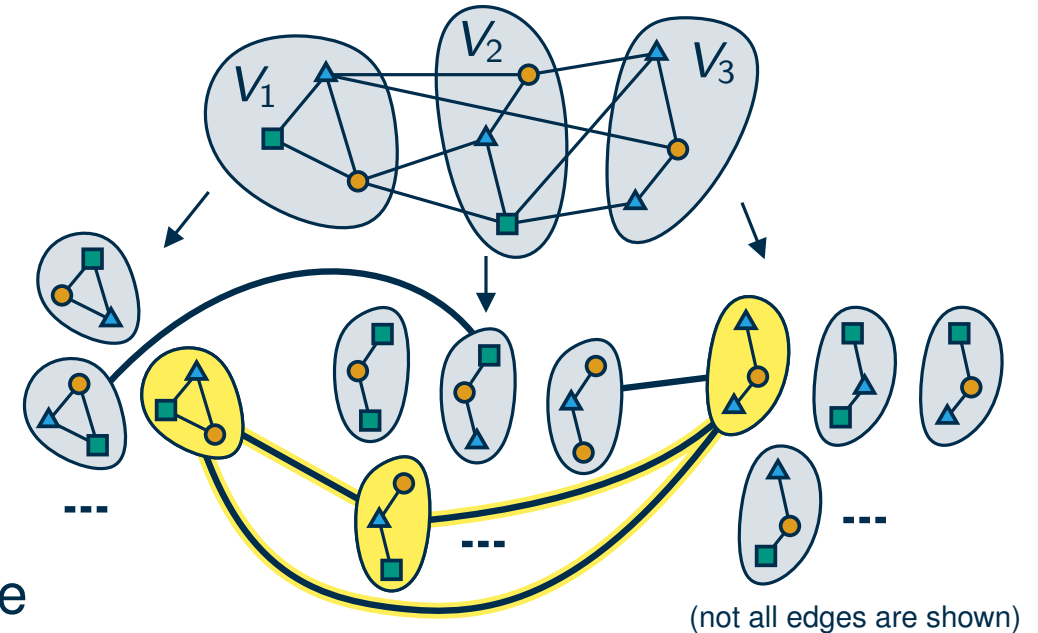


Partial solutions

- partition $V = V_1 \cup \dots \cup V_k$
- C_i : set of all 3-colorings of $G[V_i]$
- choose a coloring $c_i \in C_i$ for every i such that choices are compatible

Changing perspective

- view $C_1 \cup \dots \cup C_k$ as node set
- add edge $\{c_i, c_j\}$ if $c_i \in C_i$ and $c_j \in C_j$ are compatible



Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \text{\#vertices in the clique CLIQUE-instance}$)

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

- goal: choose k such that this is in $2^{o(n)}$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

- goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

■ goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

■ goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$
 $\uparrow$ $\uparrow$
may be too large ok

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

- goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$
↑ ↑
may be too large ok

second try: $k \in O(1)$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

■ goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$$

↑
may be too large

↑
ok

second try: $k \in O(1)$

careful:

$$o(k) \neq o(1) \text{ for } k \in O(1)$$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

- goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$$

↑
may be too large

↑
ok

second try: $k \in O(1)$

careful:

$$o(k) \neq o(1) \text{ for } k \in O(1)$$

($o(k)$ grows sublinear with growing k ; if k does not grow, then $o(k)$ is constant and not falling in n)

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

■ goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$$

↑
may be too large

↑
ok

second try: $k \in O(1)$

careful:

$$o(k) \neq o(1) \text{ for } k \in O(1)$$

($o(k)$ grows sublinear with growing k ; if k does not grow, then $o(k)$ is constant and not falling in n)

third try: $k = g(n) = f^{-1}(n)$

(we can assume that $g(n)$ is strictly growing)

$$n \cdot 3^{\frac{n}{g(n)} \cdot o(g(n))} = 2^{o(n)}$$

A lower bound for CLIQUE

Theorem

ETH $\Rightarrow$ no $2^{o(n+m)}$ -algorithm for 3-COLORING.

Lemma: The resulting graph with $3^{\frac{n}{k}} \cdot k$ vertices has a k -clique if and only if G is 3-colorable.

Assumption: there is an $f(k) \cdot n_c^{o(k)}$ algorithm for CLIQUE ($n_c = 3^{\frac{n}{k}} = \# \text{vertices in the clique CLIQUE-instance}$)

Resulting running time for 3-COLORING: $f(k) \cdot 3^{\frac{n}{k} \cdot o(k)}$

■ goal: choose k such that this is in $2^{o(n)}$

first try: $k = n$

$$f(n) \cdot 3^{\frac{n}{n} \cdot o(n)}$$

↑
may be too large

↑
ok

second try: $k \in O(1)$

careful:

$$o(k) \neq o(1) \text{ for } k \in O(1)$$

($o(k)$ grows sublinear with growing k ; if k does not grow, then $o(k)$ is constant and not falling in n)

third try: $k = g(n) = f^{-1}(n)$

(we can assume that $g(n)$ is strictly growing)

$$n \cdot 3^{\frac{n}{g(n)} \cdot o(g(n))} = 2^{o(n)}$$

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

Implications

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

More lower bounds

- parameterized reduction from CLIQUE yield lower bounds for other problems
- simple example: ETH $\Rightarrow$ INDEPENDENT SET has no $f(k)n^{o(k)}$ -algorithm

Implications

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

More lower bounds

- parameterized reduction from CLIQUE yield lower bounds for other problems
- simple example: ETH $\Rightarrow$ INDEPENDENT SET has no $f(k)n^{o(k)}$ -algorithm
- same for MULTICOLORED CLIQUE, MULTICOLORED INDEPENDENT SET, DOMINATING SET, CONNECTED DOMINATING SET, ...

Implications

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

More lower bounds

- parameterized reduction from CLIQUE yield lower bounds for other problems
- simple example: ETH $\Rightarrow$ INDEPENDENT SET has no $f(k)n^{o(k)}$ -algorithm
- same for MULTICOLORED CLIQUE, MULTICOLORED INDEPENDENT SET, DOMINATING SET, CONNECTED DOMINATING SET, ...
- **note:** you have to keep track of how the parameter changes in the reduction

Implications

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

More lower bounds

- parameterized reduction from CLIQUE yield lower bounds for other problems
- simple example: ETH $\Rightarrow$ INDEPENDENT SET has no $f(k)n^{o(k)}$ -algorithm
- same for MULTICOLORED CLIQUE, MULTICOLORED INDEPENDENT SET, DOMINATING SET, CONNECTED DOMINATING SET, ...
- **note:** you have to keep track of how the parameter changes in the reduction

Theorem: ETH impliziert $\text{FPT} \neq \text{W}[1]$.

Implications

Theorem: ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for CLIQUE.

More lower bounds

- parameterized reduction from CLIQUE yield lower bounds for other problems
- simple example: ETH $\Rightarrow$ INDEPENDENT SET has no $f(k)n^{o(k)}$ -algorithm
- same for MULTICOLORED CLIQUE, MULTICOLORED INDEPENDENT SET, DOMINATING SET, CONNECTED DOMINATING SET, ...
- **note:** you have to keep track of how the parameter changes in the reduction

Theorem: ETH impliziert $FPT \neq W[1]$.

Next

- additional somewhat different implications
- useful intermediate problem: GRID TILING

GRID TILING

Problem: GRID TILING

($[n] = [1, n]$)

Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

Example

- $k = 3$ and $n = 5$

$S_{1,1}$ (1, 1) (3, 1) (2, 4)	$S_{1,2}$ (5, 1) (1, 4) (5, 3)	$S_{1,3}$ (1, 1) (2, 4) (3, 3)
$S_{2,1}$ (2, 2) (1, 4)	$S_{2,2}$ (3, 1) (1, 2)	$S_{2,3}$ (2, 2) (2, 3)
$S_{3,1}$ (1, 3) (2, 3) (3, 3)	$S_{3,2}$ (1, 1) (1, 3)	$S_{3,3}$ (2, 3) (5, 3)

GRID TILING

Problem: GRID TILING

($[n] = [1, n]$)

Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

Example

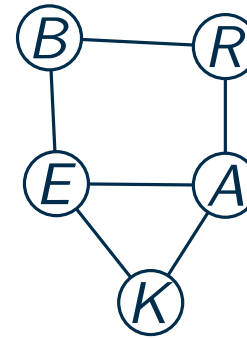
- $k = 3$ and $n = 5$

$S_{1,1}$ (1, 1) (3, 1) (2, 4)	$S_{1,2}$ (5, 1) (1, 4) (5, 3)	$S_{1,3}$ (1, 1) (2, 4) (3, 3)
$S_{2,1}$ (2, 2) (1, 4)	$S_{2,2}$ (3, 1) (1, 2)	$S_{2,3}$ (2, 2) (2, 3)
$S_{3,1}$ (1, 3) (2, 3) (3, 3)	$S_{3,2}$ (1, 1) (1, 3)	$S_{3,3}$ (2, 3) (5, 3)

What is going on?

Is this instance of GRID TILING solvable?
What does it mean for the graph?

$S_{1,1}$ $(1, 1)$ $(3, 1)$ $(2, 4)$	$S_{1,2}$ $(5, 1)$ $(1, 4)$ $(5, 3)$	$S_{1,3}$ $(1, 1)$ $(2, 4)$ $(3, 3)$
$S_{2,1}$ $(2, 2)$ $(1, 4)$	$S_{2,2}$ $(3, 1)$ $(1, 2)$	$S_{2,3}$ $(2, 2)$ $(2, 3)$
$S_{3,1}$ $(1, 3)$ $(2, 3)$ $(3, 3)$	$S_{3,2}$ $(1, 1)$ $(1, 3)$	$S_{3,3}$ $(2, 3)$ $(5, 3)$



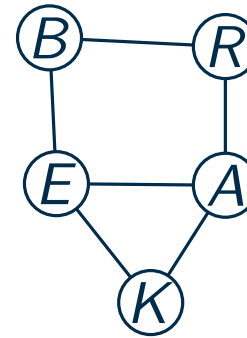
$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Problem: GRID TILING

Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

What is going on?

Is this instance of GRID TILING solvable?
What does it mean for the graph?



$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Problem: GRID TILING

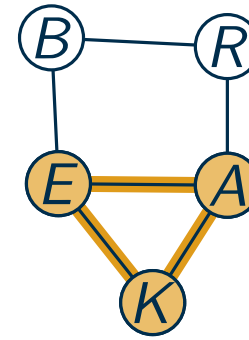
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

What is going on?

Is this instance of GRID TILING solvable?

What does it mean for the graph?

- a 3-clique in G yields a solution for GRID TILING



$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Problem: GRID TILING

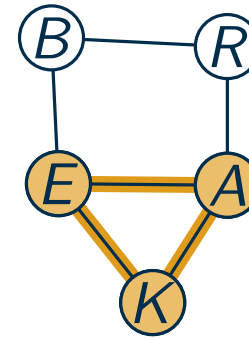
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

What is going on?

Is this instance of GRID TILING solvable?

What does it mean for the graph?

- a 3-clique in G yields a solution for GRID TILING
- inverse is also true



$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Problem: GRID TILING

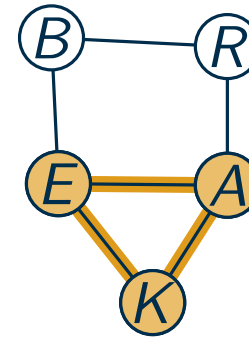
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

What is going on?

Is this instance of GRID TILING solvable?

What does it mean for the graph?

- a 3-clique in G yields a solution for GRID TILING
- inverse is also true



Note

- $n = \# \text{vertices}$, $k = \text{clique size}$
- this is a parameterized reduction from CLIQUE to GRID TILING
- parameter k translates linearly

$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Problem: GRID TILING

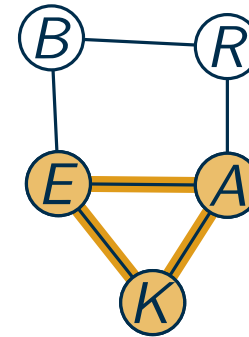
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. Choose for each cell one $s_{i,j} \in S_{i,j}$, such that $s_{i,j}$ and $s_{i+1,j}$ coincide in the first, and $s_{i,j}$ and $s_{i,j+1}$ in the second coordinate.

What is going on?

Is this instance of GRID TILING solvable?

What does it mean for the graph?

- a 3-clique in G yields a solution for GRID TILING
- inverse is also true



Note

- $n = \# \text{vertices}$, $k = \text{clique size}$
- this is a parameterized reduction from CLIQUE to GRID TILING
- parameter k translates linearly

$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)	$(R, B) (E, B)$ $(A, R) (A, E)$ $(K, E) (K, A)$
$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, R) (B, E)$ $(R, A) (E, A)$ $(E, K) (A, K)$	$(B, B) (R, R)$ $(E, E) (A, A)$ (K, K)

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Comments on GRID TILING

Reducing from GRID TILING

- the rough structure of GRID TILING is a grid
- all conditions are local: only between neighboring cells
- simple conditions: equality of a coordinate

Comments on GRID TILING

Reducing from GRID TILING

- the rough structure of GRID TILING is a grid
 - all conditions are local: only between neighboring cells
 - simple conditions: equality of a coordinate
- } well suited to prove hardness on planar graphs

Comments on GRID TILING

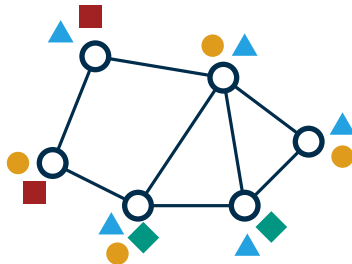
Reducing from GRID TILING

- the rough structure of GRID TILING is a grid
 - all conditions are local: only between neighboring cells
 - simple conditions: equality of a coordinate
- } well suited to prove hardness on planar graphs

Problem: PLANAR LIST COLORING

Given a planar graph and a list of colors for each vertex. For every vertex, choose a color from its list such that neighbors have different colors.

Example



Comments on GRID TILING

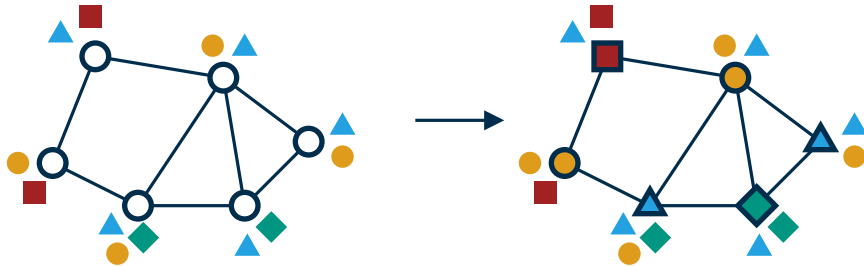
Reducing from GRID TILING

- the rough structure of GRID TILING is a grid
 - all conditions are local: only between neighboring cells
 - simple conditions: equality of a coordinate
- } well suited to prove hardness on planar graphs

Problem: PLANAR LIST COLORING

Given a planar graph and a list of colors for each vertex. For every vertex, choose a color from its list such that neighbors have different colors.

Example



Comments on GRID TILING

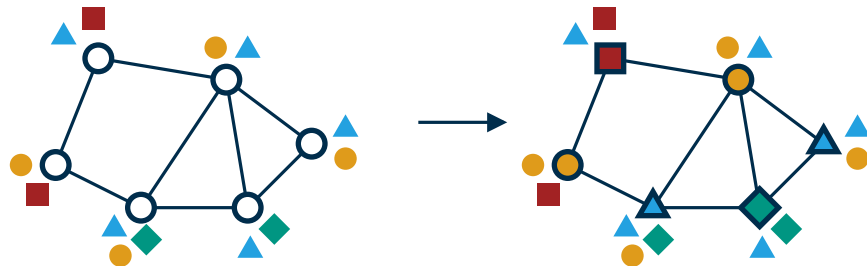
Reducing from GRID TILING

- the rough structure of GRID TILING is a grid
 - all conditions are local: only between neighboring cells
 - simple conditions: equality of a coordinate
- } well suited to prove hardness on planar graphs

Problem: PLANAR LIST COLORING

Given a planar graph and a list of colors for each vertex. For every vertex, choose a color from its list such that neighbors have different colors.

Example



Goal: reduce GRID TILING to PLANAR LIST COLORING

GRID TILING $\rightarrow$ LIST COLORING

Choice of vertices and colors

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

GRID TILING $\rightarrow$ LIST COLORING

Choice of vertices and colors

- obvious attempt: one vertex per cell, one color per pair

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

$\circ \begin{pmatrix} (1, 1) \\ (3, 1) \\ (2, 4) \end{pmatrix}$
 $\circ \begin{pmatrix} (5, 1) \\ (1, 4) \\ (5, 3) \end{pmatrix}$
 $\circ \begin{pmatrix} (1, 1) \\ (2, 4) \\ (3, 3) \end{pmatrix}$

$\circ \begin{pmatrix} (2, 2) \\ (1, 4) \end{pmatrix}$
 $\circ \begin{pmatrix} (3, 1) \\ (1, 2) \end{pmatrix}$
 $\circ \begin{pmatrix} (2, 2) \\ (2, 3) \end{pmatrix}$

$\circ \begin{pmatrix} (1, 3) \\ (2, 3) \\ (3, 3) \end{pmatrix}$
 $\circ \begin{pmatrix} (1, 1) \\ (1, 3) \end{pmatrix}$
 $\circ \begin{pmatrix} (2, 3) \\ (5, 3) \end{pmatrix}$

GRID TILING $\rightarrow$ LIST COLORING

Choice of vertices and colors

- obvious attempt: one vertex per cell, one color per pair

Forbid certain color choices

- forbid choice: $(1, 1)$ for $S_{1,1}$ and $(2, 2)$ for $S_{2,1}$

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

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

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

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

GRID TILING $\rightarrow$ LIST COLORING

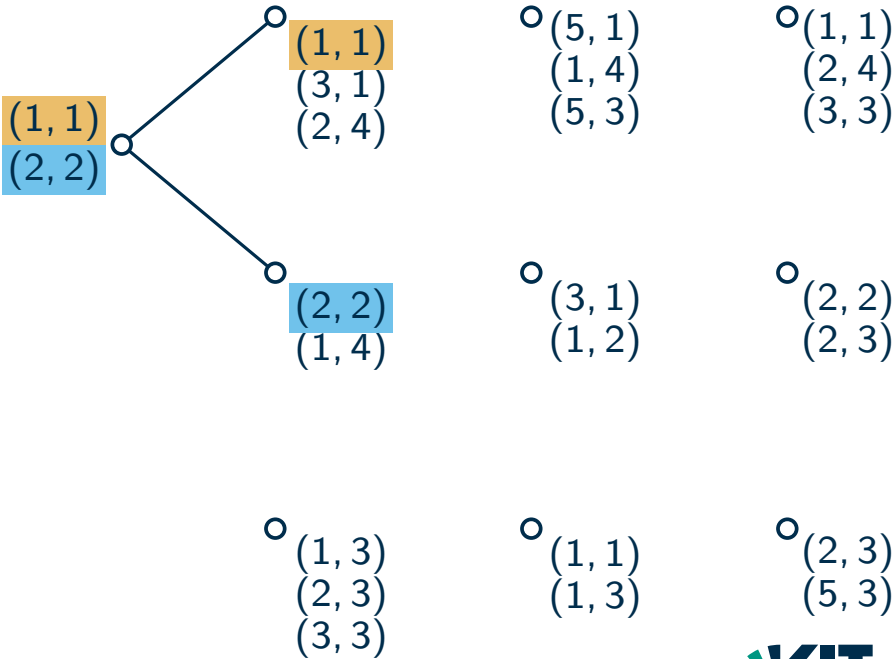
Choice of vertices and colors

- obvious attempt: one vertex per cell, one color per pair

Forbid certain color choices

- forbid choice: (1, 1) for $S_{1,1}$ and (2, 2) for $S_{2,1}$
- connect both with new vertex with colors (1, 1) and (2, 2)

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



GRID TILING $\rightarrow$ LIST COLORING

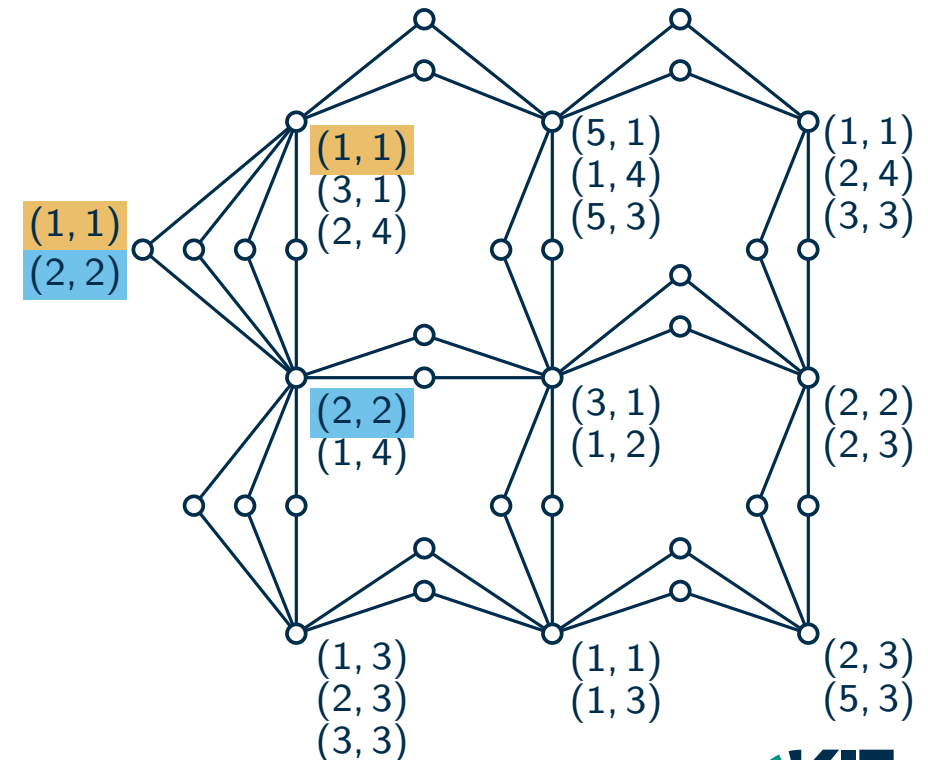
Choice of vertices and colors

- obvious attempt: one vertex per cell, one color per pair

Forbid certain color choices

- forbid choice: $(1, 1)$ for $S_{1,1}$ and $(2, 2)$ for $S_{2,1}$
- connect both with new vertex with colors $(1, 1)$ and $(2, 2)$
- same for all undesired pairs

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



GRID TILING $\rightarrow$ LIST COLORING

Choice of vertices and colors

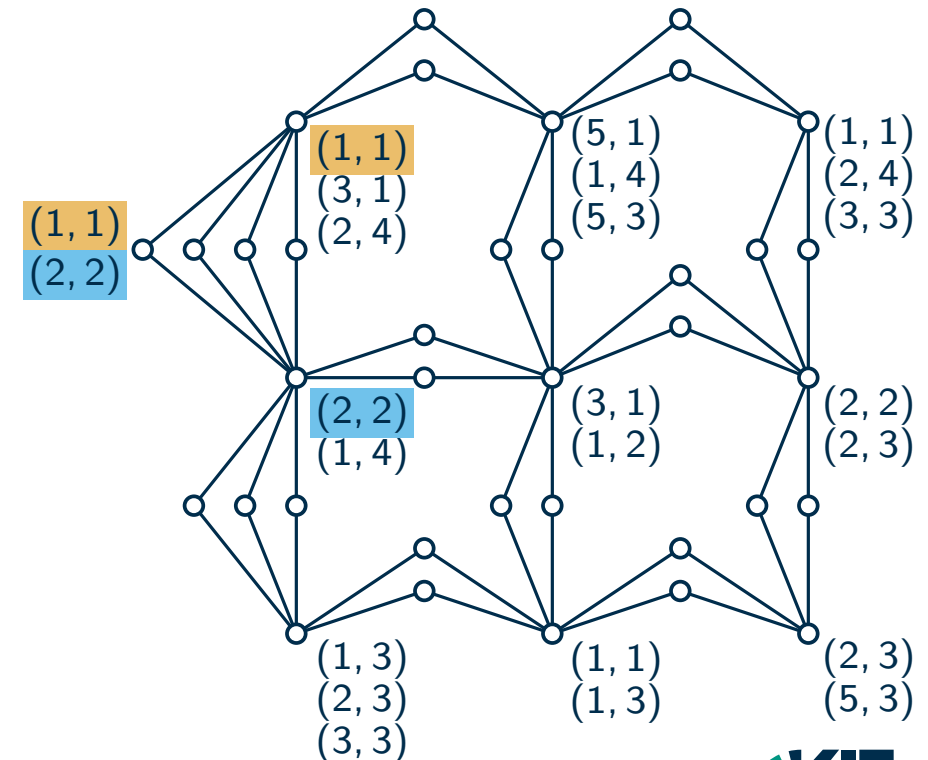
- obvious attempt: one vertex per cell, one color per pair

Forbid certain color choices

- forbid choice: $(1, 1)$ for $S_{1,1}$ and $(2, 2)$ for $S_{2,1}$
- connect both with new vertex with colors $(1, 1)$ and $(2, 2)$
- same for all undesired pairs

Note: graph is planar and instances are equivalent

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



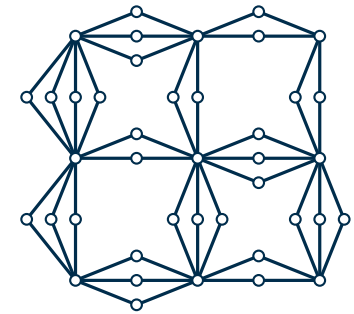
What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).



What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

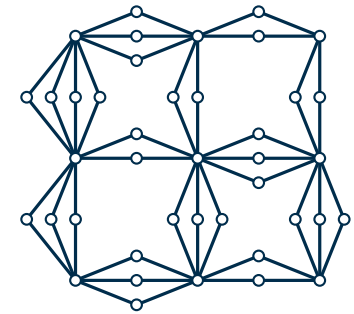
Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).

For which parameter k ?

Reminder

- k is the grid size



What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).

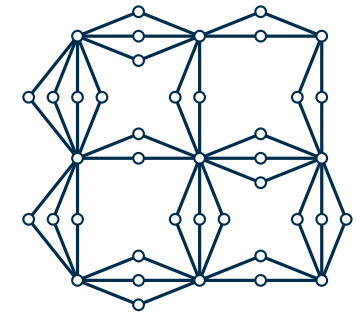
For which parameter k ?

- observation: the resulting graph has treewidth k

Why?

Reminder

- k is the grid size



What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).

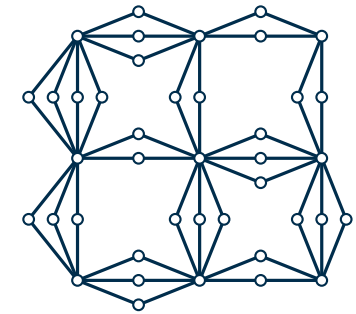
For which parameter k ?

- observation: the resulting graph has treewidth k
- thus: theorem holds when parameterized by treewidth

Why?

Reminder

- k is the grid size



What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).

For which parameter k ?

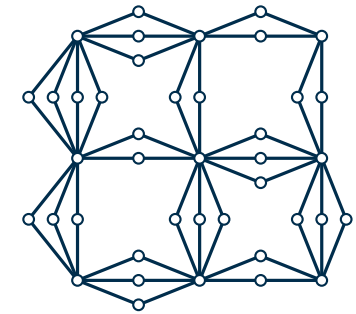
- observation: the resulting graph has treewidth k
- thus: theorem holds when parameterized by treewidth

Why?

Can we not solve LIST COLORING via DP on a tree decomposition?

Reminder

- k is the grid size



What did we show now?

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING (and it is $W[1]$ -hard).

Theorem

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for PLANAR LIST COLORING (and it is $W[1]$ -hard).

For which parameter k ?

- observation: the resulting graph has treewidth k
- thus: theorem holds when parameterized by treewidth

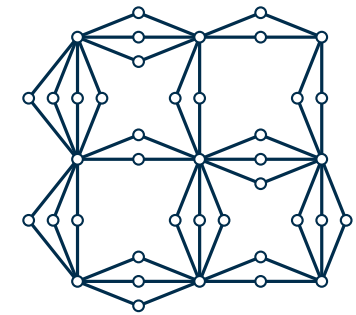
Why?

Can we not solve LIST COLORING via DP on a tree decomposition?

we can, but it is too slow

Reminder

- k is the grid size



Relaxed GRID TILING

Problem: GRID TILING WITH $\leq$ ($[n] = [1, n]$)
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. For each cell, choose one $s_{i,j} \in S_{i,j}$, such that the first and second coordinate is non-decreasing in every column and row, respectively.

Example: $k = 3, n = 5$

$S_{1,1}$ (1, 1) (3, 1) (2, 4)	$S_{1,2}$ (5, 1) (1, 4) (5, 3)	$S_{1,3}$ (1, 1) (2, 4) (3, 3)
$S_{2,1}$ (2, 2) (1, 4)	$S_{2,2}$ (3, 1) (1, 2)	$S_{2,3}$ (2, 2) (2, 3)
$S_{3,1}$ (1, 3) (2, 3) (3, 3)	$S_{3,2}$ (1, 1) (1, 3)	$S_{3,3}$ (2, 3) (5, 3)

Relaxed GRID TILING

Problem: GRID TILING WITH $\leq$ ($[n] = [1, n]$)
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. For each cell, choose one $s_{i,j} \in S_{i,j}$, such that the first and second coordinate is non-decreasing in every column and row, respectively.

Note

- GRID TILING and GRID TILING WITH $\leq$ have the same instances
- a solution for GRID TILING also solves GRID TILING WITH $\leq$ (the inverse is not true)

Example: $k = 3, n = 5$

$S_{1,1}$ (1, 1) (3, 1) (2, 4)	$S_{1,2}$ (5, 1) (1, 4) (5, 3)	$S_{1,3}$ (1, 1) (2, 4) (3, 3)
$S_{2,1}$ (2, 2) (1, 4)	$S_{2,2}$ (3, 1) (1, 2)	$S_{2,3}$ (2, 2) (2, 3)
$S_{3,1}$ (1, 3) (2, 3) (3, 3)	$S_{3,2}$ (1, 1) (1, 3)	$S_{3,3}$ (2, 3) (5, 3)

Relaxed GRID TILING

Problem: GRID TILING WITH $\leq$ ($[n] = [1, n]$)
Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. For each cell, choose one $s_{i,j} \in S_{i,j}$, such that the first and second coordinate is non-decreasing in every column and row, respectively.

Note

- GRID TILING and GRID TILING WITH $\leq$ have the same instances
- a solution for GRID TILING also solves GRID TILING WITH $\leq$ (the inverse is not true)
- that does not mean that GRID TILING WITH $\leq$ is an easier problem

Example: $k = 3, n = 5$

$S_{1,1}$ (1, 1) (3, 1) (2, 4)	$S_{1,2}$ (5, 1) (1, 4) (5, 3)	$S_{1,3}$ (1, 1) (2, 4) (3, 3)
$S_{2,1}$ (2, 2) (1, 4)	$S_{2,2}$ (3, 1) (1, 2)	$S_{2,3}$ (2, 2) (2, 3)
$S_{3,1}$ (1, 3) (2, 3) (3, 3)	$S_{3,2}$ (1, 1) (1, 3)	$S_{3,3}$ (2, 3) (5, 3)

Relaxed GRID TILING

Problem: GRID TILING WITH $\leq$ ($[n] = [1, n]$)
 Given a $k \times k$ matrix of sets $S_{i,j}$, with $S_{i,j} \subseteq [n] \times [n]$. For each cell, choose one $s_{i,j} \in S_{i,j}$, such that the first and second coordinate is non-decreasing in every column and row, respectively.

Note

- GRID TILING and GRID TILING WITH $\leq$ have the same instances
- a solution for GRID TILING also solves GRID TILING WITH $\leq$ (the inverse is not true)
- that does not mean that GRID TILING WITH $\leq$ is an easier problem

Example: $k = 3, n = 5$

$S_{1,1}$ $(1, 1)$ $(3, 1)$ $(2, 4)$	$S_{1,2}$ $(5, 1)$ $(1, 4)$ $(5, 3)$	$S_{1,3}$ $(1, 1)$ $(2, 4)$ $(3, 3)$
$S_{2,1}$ $(2, 2)$ $(1, 4)$	$S_{2,2}$ $(3, 1)$ $(1, 2)$	$S_{2,3}$ $(2, 2)$ $(2, 3)$
$S_{3,1}$ $(1, 3)$ $(2, 3)$ $(3, 3)$	$S_{3,2}$ $(1, 1)$ $(1, 3)$	$S_{3,3}$ $(2, 3)$ $(5, 3)$

Theorem (without proof)

ETH implies that there is no $f(k) \cdot n^{o(k)}$ -algorithm for GRID TILING WITH $\leq$ (and it is $W[1]$ -hard).

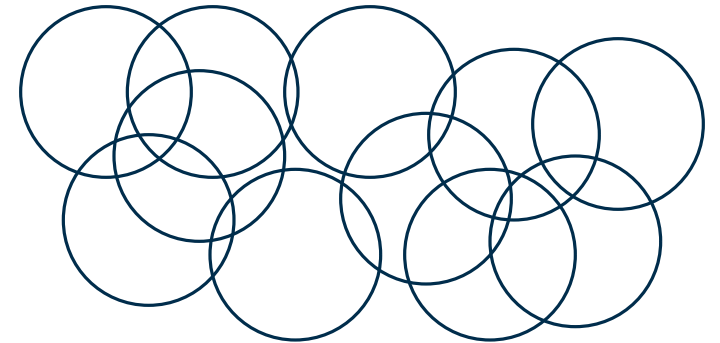
Independent Unit Disks

Problem: UNIT DISK INDEPENDENT SET

Given a set of disks with radius 1 and a parameter ℓ , find a subset of ℓ disjoint disks.

Example

- are there $\ell = 6$ disjoint disks?



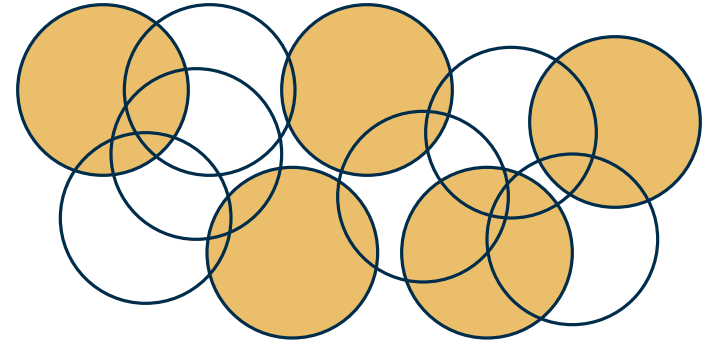
Independent Unit Disks

Problem: UNIT DISK INDEPENDENT SET

Given a set of disks with radius 1 and a parameter ℓ , find a subset of ℓ disjoint disks.

Example

- are there $\ell = 6$ disjoint disks?
- no, 5 is the maximum



Independent Unit Disks

Problem: UNIT DISK INDEPENDENT SET

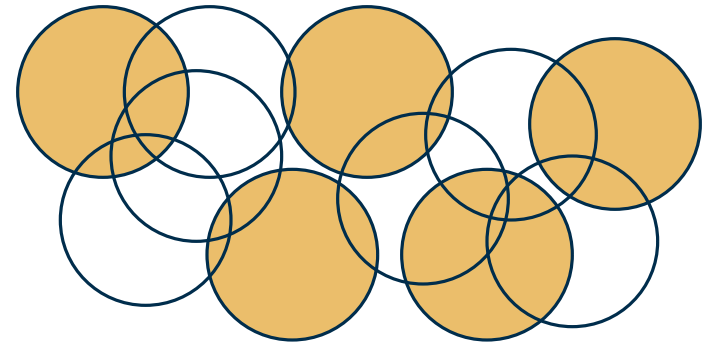
Given a set of disks with radius 1 and a parameter ℓ , find a subset of ℓ disjoint disks.

Note

- equivalent to INDEPENDENT SET on unit disk graphs
- potentially easier than INDEPENDENT SET

Example

- are there $\ell = 6$ disjoint disks?
- no, 5 is the maximum



Independent Unit Disks

Problem: UNIT DISK INDEPENDENT SET

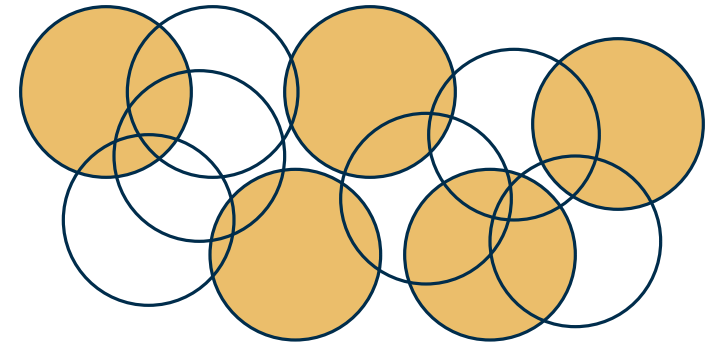
Given a set of disks with radius 1 and a parameter ℓ , find a subset of ℓ disjoint disks.

Note

- equivalent to INDEPENDENT SET on unit disk graphs
- potentially easier than INDEPENDENT SET
- ETH lower bound of INDEPENDENT SET: $n^{\Omega(\ell)}$
- UNIT DISK INDEPENDENT SET can be solved in $n^{O(\sqrt{\ell})}$

Example

- are there $\ell = 6$ disjoint disks?
- no, 5 is the maximum



Independent Unit Disks

Problem: UNIT DISK INDEPENDENT SET

Given a set of disks with radius 1 and a parameter ℓ , find a subset of ℓ disjoint disks.

Note

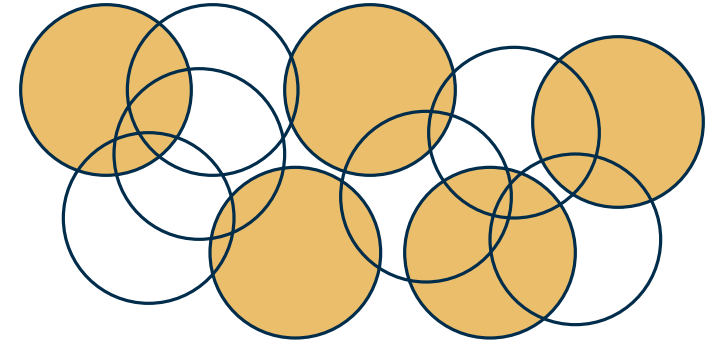
- equivalent to INDEPENDENT SET on unit disk graphs
- potentially easier than INDEPENDENT SET
- ETH lower bound of INDEPENDENT SET: $n^{\Omega(\ell)}$
- UNIT DISK INDEPENDENT SET can be solved in $n^{O(\sqrt{\ell})}$

Next slide

- this is tight: lower bound of $n^{\Omega(\sqrt{\ell})}$ UNIT DISK INDEPENDENT SET
- reduction from GRID TILING WITH $\leq$

Example

- are there $\ell = 6$ disjoint disks?
- no, 5 is the maximum



Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers

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

Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller

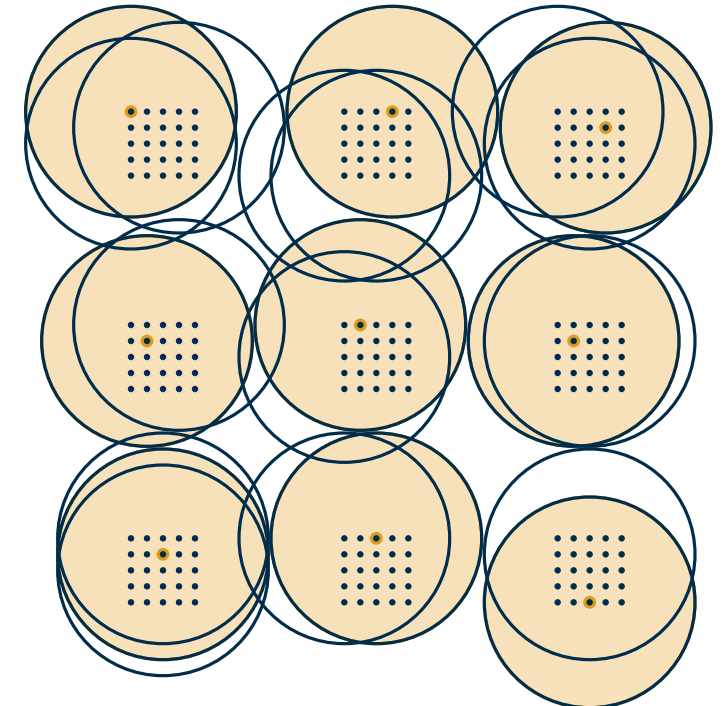
(1, 1) (3, 1) (2, 4)	(5, 1) (1, 4) (5, 3)	(1, 1) (2, 4) (3, 3)
(2, 2) (1, 4)	(3, 1) (1, 2)	(2, 2) (2, 3)
(1, 3) (2, 3) (3, 3)	(1, 1) (1, 3)	(2, 3) (5, 3)

Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller

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

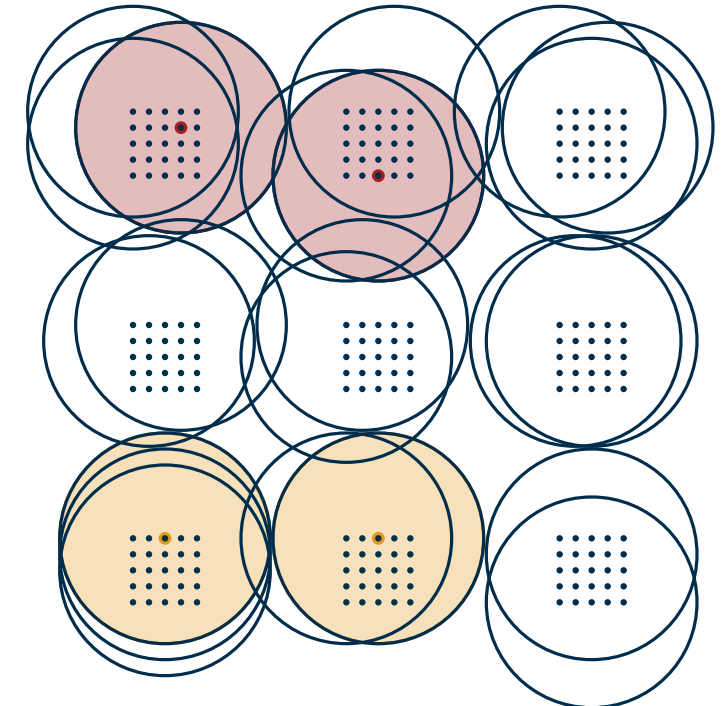


Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller

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

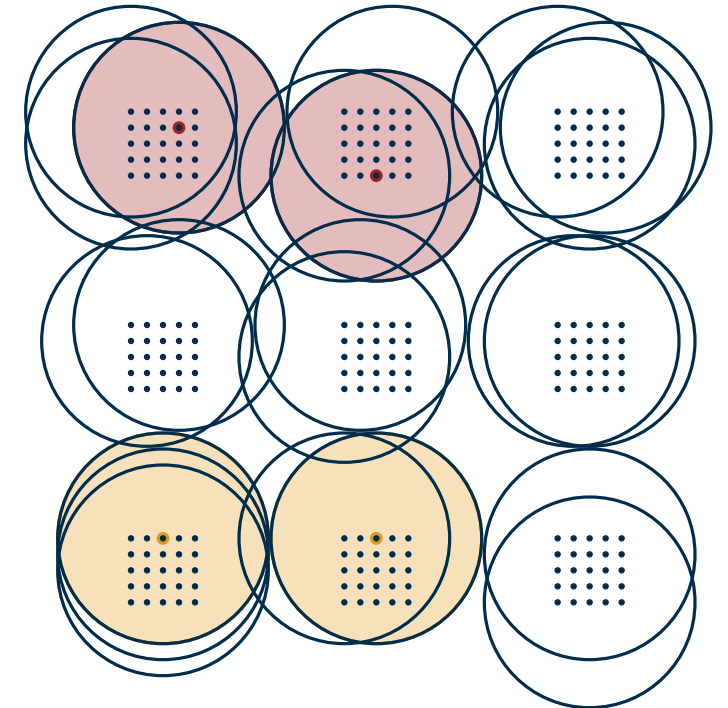


Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller
- grid tiling yes instance $\Leftrightarrow k^2$ independent disks

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



Lower bound for UNIT DISK IS

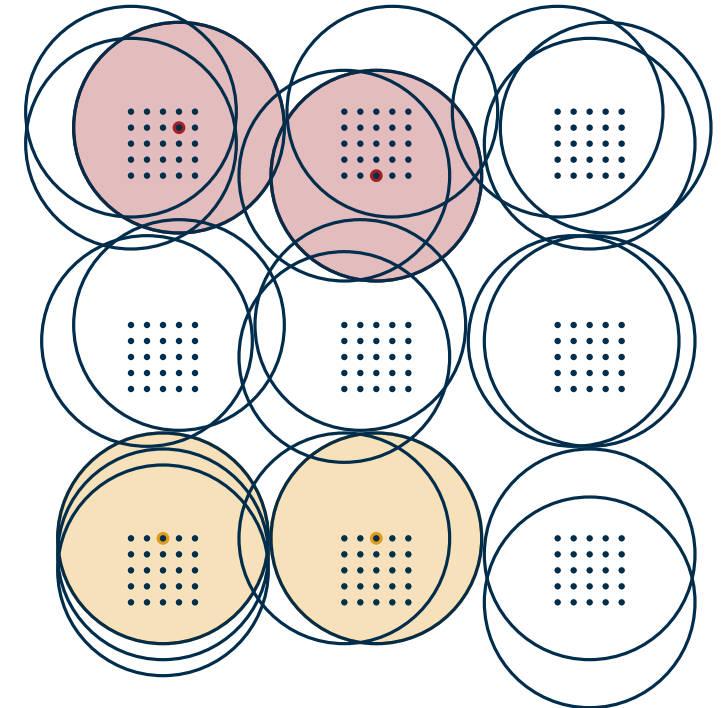
Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller
- grid tiling yes instance $\Leftrightarrow k^2$ independent disks

How does the parameter change?

- grid tiling: $k \times k$ grid $\rightarrow$ parameter k
- unit disks: parameter $\ell = k^2$

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



Lower bound for UNIT DISK IS

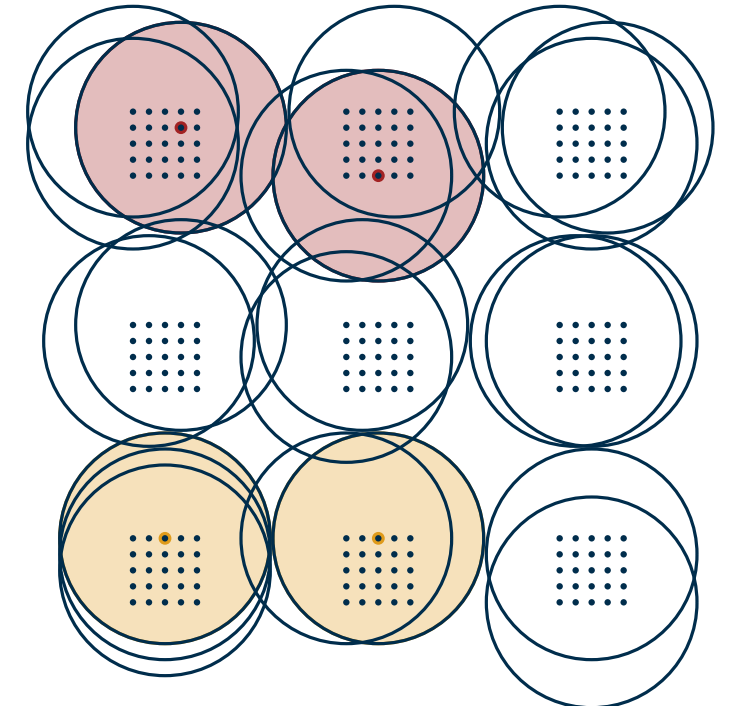
Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller
- grid tiling yes instance $\Leftrightarrow k^2$ independent disks

How does the parameter change?

- grid tiling: $k \times k$ grid $\rightarrow$ parameter k
 - unit disks: parameter $\ell = k^2$
- $$\left. \vphantom{\begin{matrix} \text{grid tiling: } k \times k \text{ grid} \rightarrow \text{parameter } k \\ \text{unit disks: parameter } \ell = k^2 \end{matrix}} \right\} n^{o(\sqrt{\ell})} \text{ would imply } n^{o(k)}$$

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



Lower bound for UNIT DISK IS

Core idea of the reduction

- one disk for every pair of numbers
- disks of pairs from the same cell always intersect
- disks from pairs of neighboring cells intersect, if the relevant coordinate gets smaller
- grid tiling yes instance $\Leftrightarrow k^2$ independent disks

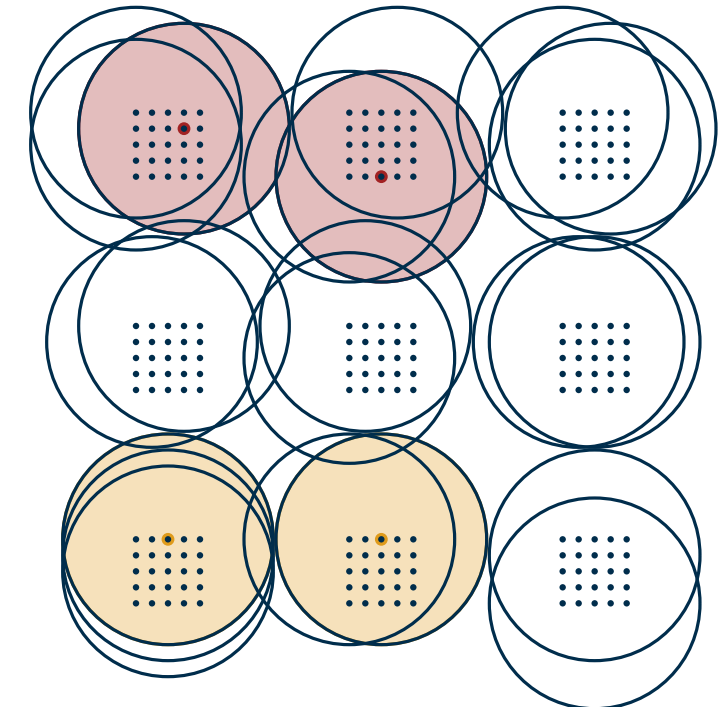
How does the parameter change?

- grid tiling: $k \times k$ grid $\rightarrow$ parameter k
 - unit disks: parameter $\ell = k^2$
- $$\left. \vphantom{\begin{matrix} \text{grid tiling: } k \times k \text{ grid} \rightarrow \text{parameter } k \\ \text{unit disks: parameter } \ell = k^2 \end{matrix}} \right\} n^{o(\sqrt{\ell})} \text{ would imply } n^{o(k)}$$

Theorem

ETH implies that there is no $f(\ell) \cdot n^{o(\sqrt{\ell})}$ -algorithm for UNIT DISK INDEPENDENT SET (and it is $W[1]$ -hard).

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



Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq W[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)

Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq \text{W}[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)
- translate bound $2^{\Omega(n)}$ (3-COLORING) to parameterized bound $f(k)n^{\Omega(k)}$ (CLIQUE)

Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq W[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)
- translate bound $2^{\Omega(n)}$ (3-COLORING) to parameterized bound $f(k)n^{\Omega(k)}$ (CLIQUE)
- yields finer subdivision of $W[1]$

Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq W[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)
- translate bound $2^{\Omega(n)}$ (3-COLORING) to parameterized bound $f(k)n^{\Omega(k)}$ (CLIQUE)
- yields finer subdivision of $W[1]$

GRID TILING

- nice problem to start reductions for geometric problems

Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq W[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)
- translate bound $2^{\Omega(n)}$ (3-COLORING) to parameterized bound $f(k)n^{\Omega(k)}$ (CLIQUE)
- yields finer subdivision of $W[1]$

GRID TILING

- nice problem to start reductions for geometric problems

Finer subdivision of FPT

- reminder: problem in FPT $\Leftrightarrow$ there is a kernelization algo
- reminder: lower bounds for kernel sizes

Wrap-Up

Lower bounds

- stronger assumptions ($\text{FPT} \neq W[1] \rightarrow \text{ETH}$) yield stronger lower bounds ($f(k)n^{\omega(1)} \rightarrow f(k)n^{\Omega(k)}$)
- translate bound $2^{\Omega(n)}$ (3-COLORING) to parameterized bound $f(k)n^{\Omega(k)}$ (CLIQUE)
- yields finer subdivision of $W[1]$

GRID TILING

- nice problem to start reductions for geometric problems

Finer subdivision of FPT

- reminder: problem in FPT $\Leftrightarrow$ there is a kernelization algo
- reminder: lower bounds for kernel sizes

Finer subdivision of P

- example: SETH $\Rightarrow$ no subquadratic algorithm for ORTHOGONAL VECTORS
- see: Fine-Grained Complexity Theory