Lower bounds based on ETH

Part 2

Michał Pilipczuk

Institutt for Informatikk, Universitetet i Bergen

August 21$^{\mathrm{st}}$, 2014

Previous lecture

- **ETH**: $\mathrm{3SAT}$ requires $\mathcal{O}^\star(2^{cn})$ time for some $c > 0$.

Previous lecture

- **ETH**: $\mathrm{3SAT}$ requires $\mathcal{O}^\star(2^{cn})$ time for some $c > 0$.
- **Sparsification Lemma**: $\mathrm{3SAT}$ requires $\mathcal{O}^\star(2^{c(n+m)})$ time for some $c > 0$. No $2^{o(n+m)}$ algorithm.

Previous lecture

- **ETH**: $3\mathrm{SAT}$ requires $\mathcal{O}^\star(2^{cn})$ time for some $c > 0$.
- **Sparsification Lemma**: $3\mathrm{SAT}$ requires $\mathcal{O}^\star(2^{c(n+m)})$ time for some $c > 0$. No $2^{o(n+m)}$ algorithm.
- **Corollary**: For a number of problems, exact and parameterized algorithms cannot achieve subexponential time.

Previous lecture

- **ETH**: 3SAT requires $\mathcal{O}^\star(2^{cn})$ time for some $c > 0$.
- **Sparsification Lemma**: 3SAT requires $\mathcal{O}^\star(2^{c(n+m)})$ time for some $c > 0$. No $2^{o(n+m)}$ algorithm.
- **Corollary**: For a number of problems, exact and parameterized algorithms cannot achieve subexponential time.
- **Corollary**: No $f(k) \cdot n^{o(k)}$ algorithm for CLIQUE under ETH, for any computable f.

This lecture

- Show more exotic lower bounds under ETH.

This lecture

- Show more exotic lower bounds under ETH.
- **Slightly super-exponential**:
  lower bounds excluding $\mathcal{O}^\star(2^{o(k \log k)})$ algorithms.

This lecture

- Show more exotic lower bounds under ETH.
- **Slightly super-exponential**:
  lower bounds excluding $\mathcal{O}^\star(2^{o(k \log k)})$ algorithms.
- **Hardness for planar problems**:

This lecture

- Show more exotic lower bounds under ETH.
- **Slightly super-exponential**:
  lower bounds excluding $\mathcal{O}^\star(2^{o(k \log k)})$ algorithms.
- **Hardness for planar problems**:
  - Last lecture: $\mathcal{O}^\star(2^{\sqrt{k}})$ lower bounds for FPT problems.

This lecture

- Show more exotic lower bounds under ETH.
- **Slightly super-exponential**:
  lower bounds excluding $\mathcal{O}^\star(2^{o(k \log k)})$ algorithms.
- **Hardness for planar problems**:
  - Last lecture: $\mathcal{O}^\star(2^{\sqrt{k}})$ lower bounds for FPT problems.
  - This lecture: $f(k) \cdot n^{o(\sqrt{k})}$ lower bounds for $\mathrm{W}[1]$-hard problems.

This lecture

- Show more exotic lower bounds under ETH.
- **Slightly super-exponential**:
  lower bounds excluding $\mathcal{O}^\star(2^{o(k \log k)})$ algorithms.
- **Hardness for planar problems**:
  - Last lecture: $\mathcal{O}^\star(2^{\sqrt{k}})$ lower bounds for FPT problems.
  - This lecture: $f(k) \cdot n^{o(\sqrt{k})}$ lower bounds for $\mathrm{W}[1]$-hard problems.
  - Also methodology for proving $\mathrm{W}[1]$-hardness of planar problems.

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$
- Appears naturally:

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$
- Appears naturally:
  (a) A branching procedure branches $\mathcal{O}(k)$ times, each time choosing one of $\mathrm{poly}(k)$ options.

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$
- Appears naturally:
  (a) A branching procedure branches $\mathcal{O}(k)$ times, each time choosing one of $\mathrm{poly}(k)$ options.
  (b) A treewidth DP has partitions of the bag as the states.

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$
- Appears naturally:
  (a) A branching procedure branches $\mathcal{O}(k)$ times, each time choosing one of $\mathrm{poly}(k)$ options.
  (b) A treewidth DP has partitions of the bag as the states.
- We focus on (a), but lower bounds for (b) are also possible.

Slightly super-exponential time

- Slightly super-exponential $= \mathcal{O}^\star(2^{\mathcal{O}(k \log k)}) = \mathcal{O}^\star(k^{\mathcal{O}(k)})$
- Appears naturally:
  (a) A branching procedure branches $\mathcal{O}(k)$ times, each time choosing one of $\mathrm{poly}(k)$ options.
  (b) A treewidth DP has partitions of the bag as the states.
- We focus on (a), but lower bounds for (b) are also possible.
- **Goal**: construct a methodology for showing that $\mathcal{O}^\star(2^{\mathcal{O}(k \log k)})$ cannot be improved.

Archetypical problem

- We need an archetypical problem for this running time.

Archetypical problem

- We need an archetypical problem for this running time.
- **Intuition**: k independent choices out of k options.

Archetypical problem

- We need an archetypical problem for this running time.
- **Intuition**: k independent choices out of k options.

$k \times k$-CLIQUE

Input: A graph H on vertex set $[k] \times [k]$

Question: Is there a k-clique in H that contains exactly one vertex from each row?

Archetypical problem

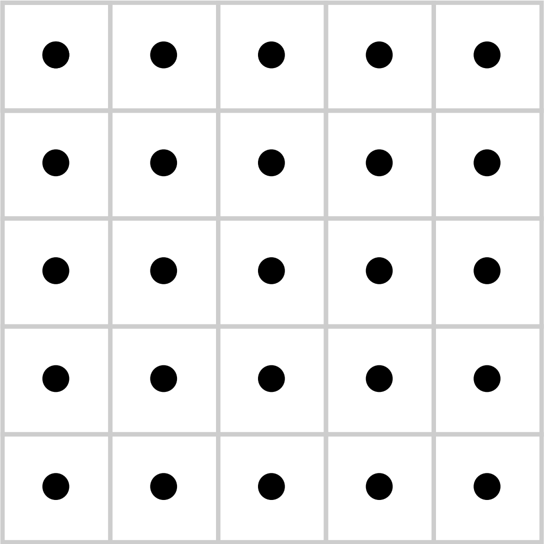
- We need an archetypical problem for this running time.
- **Intuition**: k independent choices out of k options.

$k \times k$-Clique

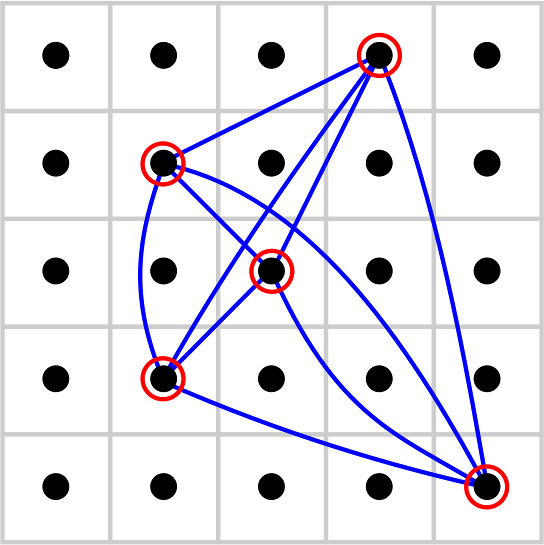
Input:	A graph H on vertex set $[k] \times [k]$
Question:	Is there a k-clique in H that contains exactly one vertex from each row?

- $[k] = \{1, 2, \ldots, k\}$.

On a picture



On a picture



On $k \times k$-Clique

- **Note**: the input to the problem is of size $\mathcal{O}(k^4)$.

On $k \times k$-Clique

- **Note**: the input to the problem is of size $\mathcal{O}(k^4)$.
- Trivial $\mathcal{O}^\star(k^k)$ algorithm: verify all the choices.

On $k \times k$-Clique

- **Note**: the input to the problem is of size $\mathcal{O}(k^4)$.
- Trivial $\mathcal{O}^\star(k^k)$ algorithm: verify all the choices.
- **Intuition**: $k \times k$-Clique models k independent 1-in-k choices in the same manner as (Multicolored) Clique models k independent 1-in-n choices.

On $k \times k$-Clique

- **Note**: the input to the problem is of size $\mathcal{O}(k^4)$.
- Trivial $\mathcal{O}^\star(k^k)$ algorithm: verify all the choices.
- **Intuition**: $k \times k$-Clique models k independent 1-in-k choices in the same manner as (Multicolored) Clique models k independent 1-in-n choices.
- Hence, we should imitate the lower bound for Clique from the previous lecture.

On $k \times k$-Clique

- **Note**: the input to the problem is of size $\mathcal{O}(k^4)$.
- Trivial $\mathcal{O}^\star(k^k)$ algorithm: verify all the choices.
- **Intuition**: $k \times k$-Clique models k independent 1-in-k choices in the same manner as (Multicolored) Clique models k independent 1-in-n choices.
- Hence, we should imitate the lower bound for Clique from the previous lecture.
- **Now**: $k \times k$-Clique does not admit an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm unless ETH fails.

Lower bound for $k \times k$-Clique

- **Starting point**: 3-Coloring on an N-vertex graph does not admit a $2^{o(N)}$ algorithm.

Lower bound for $k \times k$-Clique

- **Starting point**: 3-Coloring on an N-vertex graph does not admit a $2^{o(N)}$ algorithm.
- Take an instance G of 3-Coloring.

Lower bound for $k \times k$-Clique

- **Starting point**: 3-Coloring on an N-vertex graph does not admit a $2^{o(N)}$ algorithm.
- Take an instance G of 3-Coloring.
- Divide the vertices into $k := \frac{2N}{\log_3 N}$ groups, each of size $\frac{\log_3 N}{2}$.

Lower bound for $k \times k$-Clique

- **Starting point**: 3-Coloring on an N-vertex graph does not admit a $2^{o(N)}$ algorithm.
- Take an instance G of 3-Coloring.
- Divide the vertices into $k := \frac{2N}{\log_3 N}$ groups, each of size $\frac{\log_3 N}{2}$.
- For each of the groups list all the 3-colorings.

Lower bound for $k \times k$-Clique

- **Starting point**: 3-Coloring on an N-vertex graph does not admit a $2^{o(N)}$ algorithm.
- Take an instance G of 3-Coloring.
- Divide the vertices into $k := \frac{2N}{\log_3 N}$ groups, each of size $\frac{\log_3 N}{2}$.
- For each of the groups list all the 3-colorings.
  - There is $3^{\frac{\log_3 N}{2}} = \sqrt{N} \leq k$ of them.

Lower bound for $k \times k$-Clique

- For $i \in [k]$ and $j \in [\sqrt{N}]$, vertex (i, j) represents the j-th coloring of the i-th group.

Lower bound for $k \times k$-Clique

- For $i \in [k]$ and $j \in [\sqrt{N}]$, vertex (i,j) represents the j-th coloring of the i-th group.
- For $i \neq i'$, put an edge between (i,j) and (i',j') if respective colorings together form a proper coloring of the union of the groups i and i'.

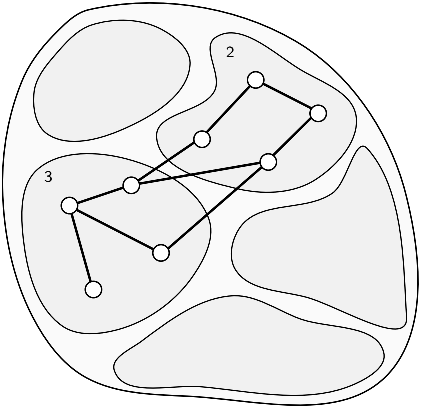
Lower bound for $k \times k$-Clique

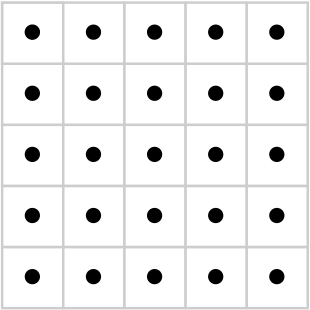
- For $i \in [k]$ and $j \in [\sqrt{N}]$, vertex (i, j) represents the j-th coloring of the i-th group.
- For $i \neq i'$, put an edge between (i, j) and (i', j') if respective colorings together form a proper coloring of the union of the groups i and i'.
- **Note**: colorings that are not proper already on their own groups will become isolated vertices.

Lower bound for $k \times k$-Clique

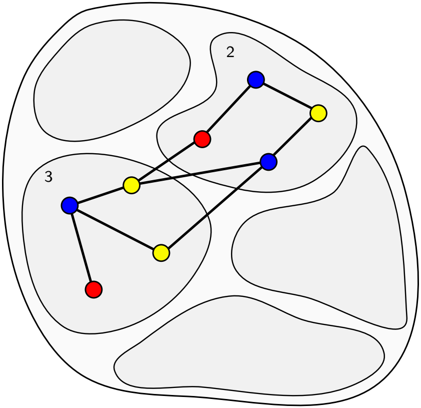
- For $i \in [k]$ and $j \in [\sqrt{N}]$, vertex (i, j) represents the j-th coloring of the i-th group.
- For $i \neq i'$, put an edge between (i, j) and (i', j') if respective colorings together form a proper coloring of the union of the groups i and i'.
- **Note**: colorings that are not proper already on their own groups will become isolated vertices.
- Finally, fill the rows with isolated vertices up to size k.

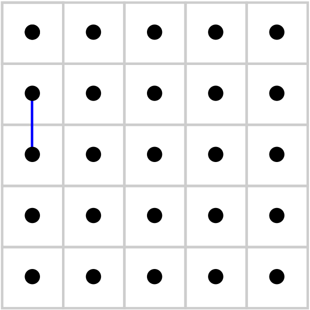
On a picture



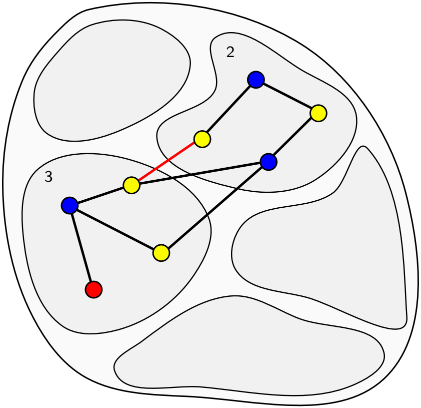


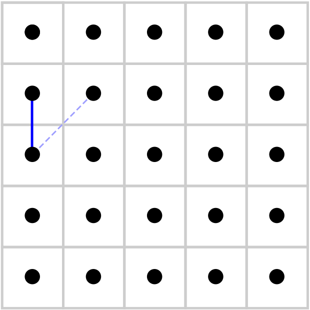
On a picture



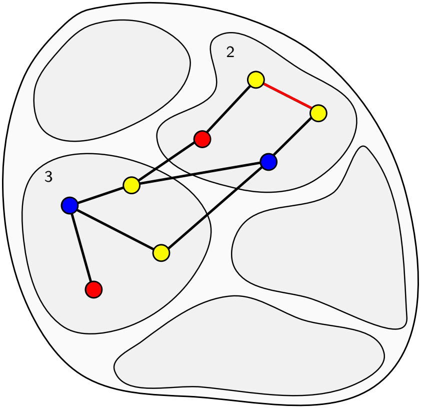


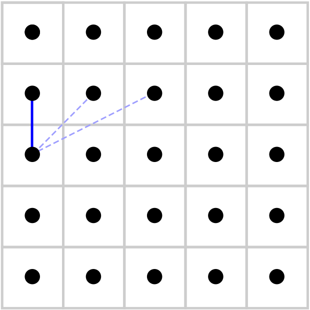
On a picture





On a picture





Equivalence

- If there is a coloring, then there is a clique: trivial.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.
- Suppose there is an edge with endpoints of the same color.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.
- Suppose there is an edge with endpoints of the same color.
  - Within a group: the coloring of this group would yield an isolated vertex.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.
- Suppose there is an edge with endpoints of the same color.
  - Within a group: the coloring of this group would yield an isolated vertex.
  - Between two groups: the corresponding colorings of the groups wouldn't be connected by an edge.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.
- Suppose there is an edge with endpoints of the same color.
  - Within a group: the coloring of this group would yield an isolated vertex.
  - Between two groups: the corresponding colorings of the groups wouldn't be connected by an edge.
- Since $k = \mathcal{O}(N/\log N)$, an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm for $k \times k$-Clique implies a $2^{o\left(\frac{N}{\log N}\cdot \log N\right)} = 2^{o(N)}$ algorithm for 3-Coloring.

Equivalence

- If there is a coloring, then there is a clique: trivial.
- If there is a clique, then consider the coloring imposed by it.
- Suppose there is an edge with endpoints of the same color.
  - Within a group: the coloring of this group would yield an isolated vertex.
  - Between two groups: the corresponding colorings of the groups wouldn't be connected by an edge.
- Since $k = \mathcal{O}(N/\log N)$, an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm for $k \times k$-CLIQUE implies a $2^{o\left(\frac{N}{\log N} \cdot \log N\right)} = 2^{o(N)}$ algorithm for 3-COLORING.
- And we are done.

Permutation $k \times k$-Clique

Permutation $k \times k$-Clique

Input: A graph H on vertex set $[k] \times [k]$

Question: Is there a k-clique in H that contains exactly one vertex from each row and each column?

PERMUTATION $k \times k$-CLIQUE

PERMUTATION $k \times k$-CLIQUE

Input: A graph H on vertex set $[k] \times [k]$

Question: Is there a k-clique in H that contains exactly one vertex from each row and each column?

- We would like to get the same lower bound also for this problem.

Lower bound

- Suppose we have an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm $\mathcal{A}$ for PERM $k \times k$-CLIQUE.

Lower bound

- Suppose we have an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm $\mathcal{A}$ for $\textsc{Perm}$ $k \times k$-$\textsc{Clique}$.
- Consider the following algorithm for $k \times k$-$\textsc{Clique}$.

Lower bound

- Suppose we have an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm $\mathcal{A}$ for $\textsc{Perm}$ $k \times k$-$\textsc{Clique}$.
- Consider the following algorithm for $k \times k$-$\textsc{Clique}$.
- Shuffle each row uniformly and independently at random.

Lower bound

- Suppose we have an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm $\mathcal{A}$ for $\textsc{Perm}$ $k \times k$-$\textsc{Clique}$.
- Consider the following algorithm for $k \times k$-$\textsc{Clique}$.
- Shuffle each row uniformly and independently at random.
- Apply algorithm $\mathcal{A}$.

Lower bound

- Suppose we have an $\mathcal{O}^\star(2^{o(k \log k)})$ algorithm $\mathcal{A}$ for PERM $k \times k$-CLIQUE.
- Consider the following algorithm for $k \times k$-CLIQUE.
- Shuffle each row uniformly and independently at random.
- Apply algorithm $\mathcal{A}$.
- Probability that a solution becomes a permutation is $\frac{k!}{k^k} \approx e^{-k}$.

Lower bound

- We need to repeat the experiment roughly e^k times to get error probability $< \frac{1}{2}$.

Lower bound

- We need to repeat the experiment roughly e^k times to get error probability $< \frac{1}{2}$.
- But $\mathcal{O}^\star(e^k \cdot 2^{o(k \log k)}) = \mathcal{O}^\star(2^{o(k \log k)})$.

Lower bound

- We need to repeat the experiment roughly e^k times to get error probability $< \frac{1}{2}$.
- But $\mathcal{O}^\star(e^k \cdot 2^{o(k \log k)}) = \mathcal{O}^\star(2^{o(k \log k)})$.
- This gives hardness of $\textsc{Perm}$ $k \times k$-$\textsc{Clique}$ under randomized ETH.

Lower bound

- We need to repeat the experiment roughly e^k times to get error probability $< \frac{1}{2}$.
- But $\mathcal{O}^\star(e^k \cdot 2^{o(k \log k)}) = \mathcal{O}^\star(2^{o(k \log k)})$.
- This gives hardness of $\textsc{Perm}$ $k \times k$-$\textsc{Clique}$ under randomized ETH.
- **Note**: This can be derandomized, so hardness holds under deterministic ETH as well.

$k \times k$-Hitting Set

$k \times k$-Hitting Set

Input: A family $\mathcal{F}$ of subsets of $[k] \times [k]$

Question: Is there a set X that contains exactly one vertex from each row and has a nonempty intersection with every set of $\mathcal{F}$?

$k \times k$-Hitting Set

$k \times k$-Hitting Set

Input: A family $\mathcal{F}$ of subsets of $[k] \times [k]$

Question: Is there a set X that contains exactly one vertex from each row and has a nonempty intersection with every set of $\mathcal{F}$?

- $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound: easy reduction from $k \times k$-Clique.

$k \times k$-Hitting Set

$k \times k$-Hitting Set

Input: A family $\mathcal{F}$ of subsets of $[k] \times [k]$
Question: Is there a set X that contains exactly one vertex from each row and has a nonempty intersection with every set of $\mathcal{F}$?

- $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound: easy reduction from $k \times k$-Clique.
  - For every non-edge $(i,j) - (i',j')$ of H, introduce a set containing the whole rows i and i' apart from (i,j) and (i',j').

$k \times k$-Hitting Set

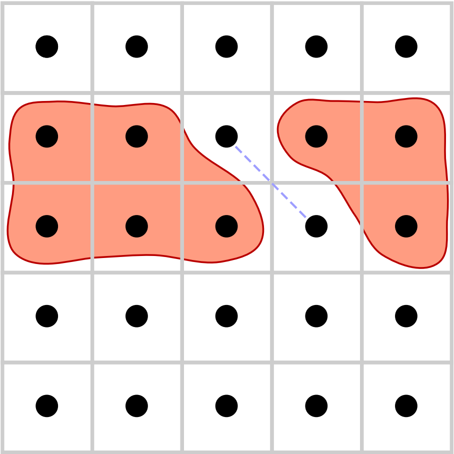
$k \times k$-Hitting Set

Input: A family $\mathcal{F}$ of subsets of $[k] \times [k]$

Question: Is there a set X that contains exactly one vertex from each row and has a nonempty intersection with every set of $\mathcal{F}$?

- $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound: easy reduction from $k \times k$-Clique.
  - For every non-edge $(i,j) - (i',j')$ of H, introduce a set containing the whole rows i and i' apart from (i,j) and (i',j').
- Same works for Perm $k \times k$-Hitting Set and Perm $k \times k$-Clique.

On a picture



... WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.

… WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.
- $k \times k$-Hitting Set with thin sets

... WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.
- $k \times k$-Hitting Set with thin sets
  - Every set of $\mathcal{F}$ is required to contain at most one vertex from each row.

... WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.
- $k \times k$-Hitting Set with thin sets
  - Every set of $\mathcal{F}$ is required to contain at most one vertex from each row.
- One can show an $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound also for $k \times k$-Hitting Set with thin sets.

... WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.
- $k \times k$-Hitting Set with thin sets
  - Every set of $\mathcal{F}$ is required to contain at most one vertex from each row.
- One can show an $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound also for $k \times k$-Hitting Set with thin sets.
- Technical reduction with a pivot problem: exercises in the book.

... WITH THIN SETS

- Seems that $k \times k$-Hitting Set in full generality is not that useful.
- $k \times k$-Hitting Set with thin sets
  - Every set of $\mathcal{F}$ is required to contain at most one vertex from each row.
- One can show an $\mathcal{O}^\star(2^{o(k \log k)})$ lower bound also for $k \times k$-Hitting Set with thin sets.
- Technical reduction with a pivot problem: exercises in the book.
- Same holds for Permutation....

Application: Closest String

Closest String

Input: An alphabet Σ, strings $x_1, x_2, \ldots, x_n$ over Σ, each of length L, and an integer d

Question: Is there a string $y \in \Sigma^L$ that has Hamming distance at most d to each x_i?

Application: Closest String

Closest String

Input: An alphabet Σ, strings $x_1, x_2, \ldots, x_n$ over Σ, each of length L, and an integer d

Question: Is there a string $y \in \Sigma^L$ that has Hamming distance at most d to each x_i?

- There are algorithms with running time $\mathcal{O}^\star(d^d)$ and $\mathcal{O}^\star(|\Sigma|^d)$.

Application: Closest String

Closest String

Input: An alphabet Σ, strings $x_1, x_2, \ldots, x_n$ over Σ, each of length L, and an integer d

Question: Is there a string $y \in \Sigma^L$ that has Hamming distance at most d to each x_i?

- There are algorithms with running time $\mathcal{O}^\star(d^d)$ and $\mathcal{O}^\star(|\Sigma|^d)$.
- We now show that this is tight: Under ETH, there is neither $\mathcal{O}^\star(2^{o(d \log d)})$ nor $\mathcal{O}^\star(2^{o(d \log |\Sigma|)})$ algorithm.

Application: Closest String

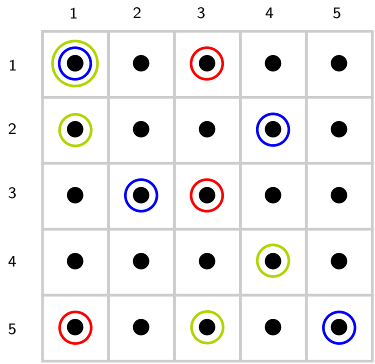
Closest String

Input: An alphabet Σ, strings $x_1, x_2, \ldots, x_n$ over Σ, each of length L, and an integer d

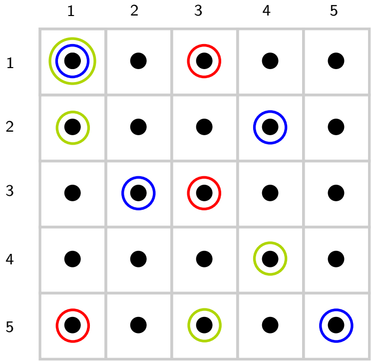
Question: Is there a string $y \in \Sigma^L$ that has Hamming distance at most d to each x_i?

- There are algorithms with running time $\mathcal{O}^\star(d^d)$ and $\mathcal{O}^\star(|\Sigma|^d)$.
- We now show that this is tight: Under ETH, there is neither $\mathcal{O}^\star(2^{o(d \log d)})$ nor $\mathcal{O}^\star(2^{o(d \log |\Sigma|)})$ algorithm.
- Reduction from Perm $k \times k$-Hitting Set wts that gives an instance with $L = k$, $d = k - 1$ and $|\Sigma| = k + 1$.

Reduction

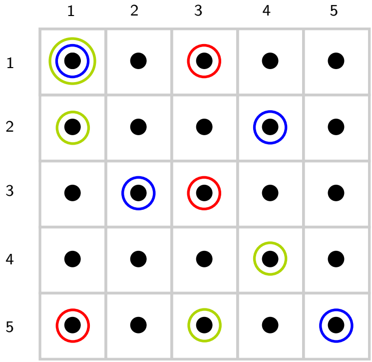


Reduction



$$\Sigma = [k] \cup \{\bigstar\}$$

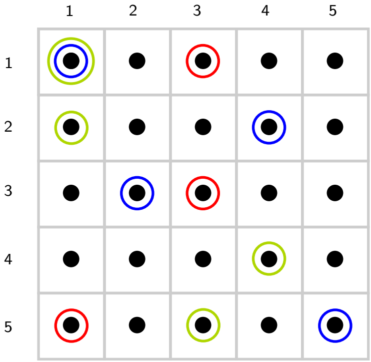
Reduction



$\Sigma = [k] \cup \{\bigstar\}$

Create strings:

Reduction

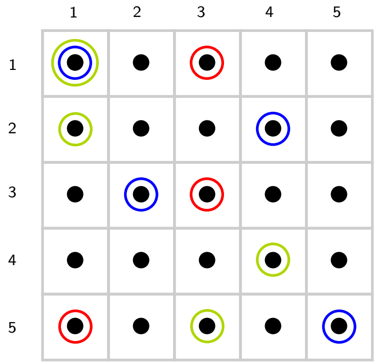


$$\Sigma = [k] \cup \{\bigstar\}$$

Create strings:

11111
22222
33333
44444
55555

Reduction



$\Sigma = [k] \cup \{\bigstar\}$

Create strings:

11111
22222
33333
44444
55555

142★5
3★3★1
11★43

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.
- Sharing with strings of the form $iii \ldots ii$
  $\Rightarrow$ y uses every symbol from $[k]$ exactly once.

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.
- Sharing with strings of the form $iii \ldots ii$
  $\Rightarrow$ y uses every symbol from $[k]$ exactly once.
- In particular, y does not use $\bigstar$, so it encodes a solution to Perm $k \times k$-Hitting Set wts.

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.
- Sharing with strings of the form $iii \dots ii$
  $\Rightarrow y$ uses every symbol from $[k]$ exactly once.
- In particular, y does not use $\bigstar$, so it encodes a solution to Perm $k \times k$-Hitting Set wts.
- y shares a symbol with x created for $X \Leftrightarrow$ the solution hits X

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.
- Sharing with strings of the form $iii \dots ii$
  $\Rightarrow$ y uses every symbol from $[k]$ exactly once.
- In particular, y does not use $\bigstar$, so it encodes a solution to $\textsc{Perm}\ k \times k\textsc{-Hitting Set wts}$.
- y shares a symbol with x created for X $\Leftrightarrow$ the solution hits X
- The second implication works the same.

Equivalence

- Suppose there is some string y at distance $\leq d = k - 1$ from all the constructed strings.
- Eq: y shares at least one symbol with each constructed string.
- Sharing with strings of the form $iii \ldots ii$
  $\Rightarrow y$ uses every symbol from $[k]$ exactly once.
- In particular, y does not use $\bigstar$, so it encodes a solution to $\textsc{Perm}$ $k \times k$-$\textsc{Hitting Set wts}$.
- y shares a symbol with x created for $X \Leftrightarrow$ the solution hits X
- The second implication works the same.
- **Ex**: What breaks if we start from $k \times k$-$\textsc{Hitting Set wts}$?

Other lower bounds

- Time $\mathcal{O}^\star(2^{\mathcal{O}(t \log t)})$ appears naturally when DP on treewidth keeps partitions of the bag as states.

Other lower bounds

- Time $\mathcal{O}^\star(2^{\mathcal{O}(t \log t)})$ appears naturally when DP on treewidth keeps partitions of the bag as states.
  - Cycle Packing: pack at least r cycles into the graph.

Other lower bounds

- Time $\mathcal{O}^\star(2^{\mathcal{O}(t \log t)})$ appears naturally when DP on treewidth keeps partitions of the bag as states.
    - Cycle Packing: pack at least r cycles into the graph.
    - Vertex Disjoint Paths: find k vertex-disjoint paths between given pairs of terminals $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$

Other lower bounds

- Time $\mathcal{O}^\star(2^{\mathcal{O}(t \log t)})$ appears naturally when DP on treewidth keeps partitions of the bag as states.
  - Cycle Packing: pack at least r cycles into the graph.
  - Vertex Disjoint Paths: find k vertex-disjoint paths between given pairs of terminals $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$
- Both these problems do not admit $\mathcal{O}^\star(2^{o(t \log t)})$ algorithms when parameterized by treewidth, unless ETH fails.

Other lower bounds

- Time $\mathcal{O}^\star(2^{\mathcal{O}(t \log t)})$ appears naturally when DP on treewidth keeps partitions of the bag as states.
  - Cycle Packing: pack at least r cycles into the graph.
  - Vertex Disjoint Paths: find k vertex-disjoint paths between given pairs of terminals $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$
- Both these problems do not admit $\mathcal{O}^\star(2^{o(t \log t)})$ algorithms when parameterized by treewidth, unless ETH fails.
- Methodology similar to what Daniel will talk about during the lecture on SETH.

W[1]-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.

W[1]-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also W[1]-hard ones.

$\mathrm{W}[1]$-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also $\mathrm{W}[1]$-hard ones.
- Typical behaviour:

W[1]-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also W[1]-hard ones.
- Typical behaviour:
  - **Upper bound**: an $n^{\mathcal{O}(\sqrt{k})}$ algorithm

W[1]-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also W[1]-hard ones.
- Typical behaviour:
  - **Upper bound**: an $n^{\mathcal{O}(\sqrt{k})}$ algorithm
  - **Lower bound**: no $f(k) \cdot n^{o(\sqrt{k})}$ algorithm for any computable f, unless ETH fails.

$\mathrm{W}[1]$-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also $\mathrm{W}[1]$-hard ones.
- Typical behaviour:
  - **Upper bound**: an $n^{\mathcal{O}(\sqrt{k})}$ algorithm
  - **Lower bound**: no $f(k) \cdot n^{o(\sqrt{k})}$ algorithm for any computable f, unless ETH fails.
- **Now**: a framework for proving such results.

W[1]-hardness on planar graphs

- **Previous lecture**: typical lower bounds of the form $\mathcal{O}^\star(2^{o(\sqrt{k})})$ for problems on planar graphs.
- On planar graphs, many more problems are FPT, but there are also $\mathrm{W}[1]$-hard ones.
- Typical behaviour:
  - **Upper bound**: an $n^{\mathcal{O}(\sqrt{k})}$ algorithm
  - **Lower bound**: no $f(k) \cdot n^{o(\sqrt{k})}$ algorithm for any computable f, unless ETH fails.
- **Now**: a framework for proving such results.
- **Recall**: under ETH, Clique does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f.

Grid Tiling

Grid Tiling

Input: Integers k, n and sets $S_{i,j} \subseteq [n] \times [n]$ for $(i, j) \in [k] \times [k]$.

Question: Can one pick $s_{i,j} \in S_{i,j}$ for each $(i, j) \in [k] \times [k]$ s.t.

(a) If $s_{i,j} = (a, b)$ and $s_{i+1,j} = (a', b')$, then $a = a'$.

(b) If $s_{i,j} = (a, b)$ and $s_{i,j+1} = (a', b')$, then $b = b'$.

On a picture

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

On a picture

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

Lower bound for Grid Tiling

- **Claim**: Grid Tiling does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.

Lower bound for GRID TILING

- **Claim**: GRID TILING does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of CLIQUE, where $V(G) = [n]$.

Lower bound for Grid Tiling

- **Claim**: Grid Tiling does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of Clique, where $V(G) = [n]$.
  - For $1 \le i \le k$, put $S_{i,i} = \{(a, a) \colon 1 \le a \le n\}$.

Lower bound for Grid Tiling

- **Claim**: Grid Tiling does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of Clique, where $V(G) = [n]$.
  - For $1 \leq i \leq k$, put $S_{i,i} = \{(a, a) \colon 1 \leq a \leq n\}$.
  - For $1 \leq i, j \leq k$, $i \neq j$, put $S_{i,j} = \{(a, b) \colon a \neq b \text{ and } ab \in E(G)\}$.

Lower bound for GRID TILING

- **Claim**: GRID TILING does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of CLIQUE, where $V(G) = [n]$.
  - For $1 \leq i \leq k$, put $S_{i,i} = \{(a, a) \colon 1 \leq a \leq n\}$.
  - For $1 \leq i, j \leq k$, $i \neq j$, put $S_{i,j} = \{(a, b) \colon a \neq b \text{ and } ab \in E(G)\}$.
- Cells on diagonal ensure that vertices chosen on rows are the same as vertices chosen on columns.

Lower bound for Grid Tiling

- **Claim**: Grid Tiling does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of Clique, where $V(G) = [n]$.
  - For $1 \leq i \leq k$, put $S_{i,i} = \{(a, a) \colon 1 \leq a \leq n\}$.
  - For $1 \leq i, j \leq k$, $i \neq j$, put $S_{i,j} = \{(a, b) \colon a \neq b \text{ and } ab \in E(G)\}$.
- Cells on diagonal ensure that vertices chosen on rows are the same as vertices chosen on columns.
- Cell (i, j) for $i \neq j$ ensures that the i-th and the j-th chosen vertex are distinct and adjacent.

Lower bound for Grid Tiling

- **Claim**: Grid Tiling does not have an $f(k) \cdot n^{o(k)}$ algorithm for any computable f, unless ETH fails.
- Take an instance (G, k) of Clique, where $V(G) = [n]$.
  - For $1 \leq i \leq k$, put $S_{i,i} = \{(a, a)\colon 1 \leq a \leq n\}$.
  - For $1 \leq i, j \leq k$, $i \neq j$, put $S_{i,j} = \{(a, b)\colon a \neq b \text{ and } ab \in E(G)\}$.
- Cells on diagonal ensure that vertices chosen on rows are the same as vertices chosen on columns.
- Cell (i, j) for $i \neq j$ ensures that the i-th and the j-th chosen vertex are distinct and adjacent.
- Hence, choosing vertices on the rows/columns models choosing a k-clique in G.

Ideas for applications

- A grid has a planar structure that a clique is missing.

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from GRID TILING:

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from Grid Tiling:
  - Model each cell with a planar 1-in-n gadget.

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from Grid Tiling:
  - Model each cell with a planar 1-in-n gadget.
  - Wire the neighboring cells to encode the Grid Tiling behaviour.

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from GRID TILING:
  - Model each cell with a planar 1-in-n gadget.
  - Wire the neighboring cells to encode the GRID TILING behaviour.
  - If each cell contributes $\mathcal{O}(1)$ to the parameter, then the final parameter is $\mathcal{O}(k^2)$.

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from Grid Tiling:
  - Model each cell with a planar 1-in-n gadget.
  - Wire the neighboring cells to encode the Grid Tiling behaviour.
  - If each cell contributes $\mathcal{O}(1)$ to the parameter, then the final parameter is $\mathcal{O}(k^2)$.
  - This gives $f(k) \cdot n^{o(\sqrt{k})}$ lower bound under ETH.

Ideas for applications

- A grid has a planar structure that a clique is missing.
- Generic reduction from Grid Tiling:
  - Model each cell with a planar 1-in-n gadget.
  - Wire the neighboring cells to encode the Grid Tiling behaviour.
  - If each cell contributes $\mathcal{O}(1)$ to the parameter, then the final parameter is $\mathcal{O}(k^2)$.
  - This gives $f(k) \cdot n^{o(\sqrt{k})}$ lower bound under ETH.
- Equality in Grid Tiling is not always convenient. For packing/domination, an inequality would be nicer.

Grid Tiling with $\leq$

Grid Tiling with $\leq$

Input: Integers k, n and sets $S_{i,j} \subseteq [n] \times [n]$ for $(i, j) \in [k] \times [k]$.

Question: Can one pick $s_{i,j} \in S_{i,j}$ for each $(i, j) \in [k] \times [k]$ s.t.

(a) If $s_{i,j} = (a, b)$ and $s_{i+1,j} = (a', b')$, then $a \leq a'$.

(b) If $s_{i,j} = (a, b)$ and $s_{i,j+1} = (a', b')$, then $b \leq b'$.

On a picture

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

Grid Tiling with $\leq$: lower bound

- Grid Tiling with $\leq$: also no $f(k) \cdot n^{o(k)}$ algorithm for any computable f under ETH.

Grid Tiling with $\leq$: lower bound

- Grid Tiling with $\leq$: also no $f(k) \cdot n^{o(k)}$ algorithm for any computable f under ETH.
- Technical reduction from standard Grid Tiling.

Grid Tiling with $\leq$: lower bound

- Grid Tiling with $\leq$: also no $f(k) \cdot n^{o(k)}$ algorithm for any computable f under ETH.
- Technical reduction from standard Grid Tiling.
- **Idea**: replace each row/column with 2 rows/columns. Whenever in the first row there is some a, on the second there is $M - a$. Hence the rows work against each other, implying equality.

Grid Tiling with $\leq$: lower bound

- Grid Tiling with $\leq$: also no $f(k) \cdot n^{o(k)}$ algorithm for any computable f under ETH.
- Technical reduction from standard Grid Tiling.
- **Idea**: replace each row/column with 2 rows/columns. Whenever in the first row there is some a, on the second there is $M - a$. Hence the rows work against each other, implying equality.
- Actually we need 4 rows/columns for synchronization, so each cell is replaced with 16 cells. Quite some details, see the book.

Application 1: d-Scattered Set

d-Scattered Set

Input: Graph G, integers k and d
Question: Does there exist a set of k vertices in G that are pairwise at distance at least d from each other?

On d-Scattered Set

- 2-Scattered Set=Independent Set

On d-Scattered Set

- 2-Scattered Set=Independent Set
- We focus on Planar d-Scattered Set.

On d-Scattered Set

- 2-Scattered Set=Independent Set
- We focus on Planar d-Scattered Set.
- When d is constant, then $\mathcal{O}^\star(2^{\mathcal{O}(\sqrt{k})})$ algorithm.

On d-Scattered Set

- 2-Scattered Set=Independent Set
- We focus on Planar d-Scattered Set.
- When d is constant, then $\mathcal{O}^\star(2^{\mathcal{O}(\sqrt{k})})$ algorithm.
- When d is a parameter, then FPT par. by $k + d$.

On d-Scattered Set

- 2-Scattered Set=Independent Set
- We focus on Planar d-Scattered Set.
- When d is constant, then $\mathcal{O}^\star(2^{\mathcal{O}(\sqrt{k})})$ algorithm.
- When d is a parameter, then FPT par. by $k + d$.
- When d is unbounded, then there is an $n^{\mathcal{O}(\sqrt{k})}$-time algorithm.

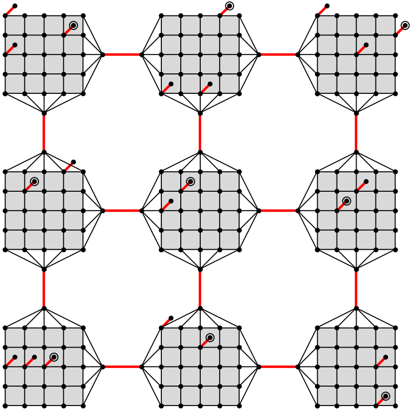
On d-Scattered Set

- 2-Scattered Set=Independent Set
- We focus on Planar d-Scattered Set.
- When d is constant, then $\mathcal{O}^\star(2^{\mathcal{O}(\sqrt{k})})$ algorithm.
- When d is a parameter, then FPT par. by $k + d$.
- When d is unbounded, then there is an $n^{\mathcal{O}(\sqrt{k})}$-time algorithm.
- **Now**: By a reduction from Grid Tiling with $\leq$, we show that the problem is $\mathrm{W}[1]$-hard and does not admit an $f(k) \cdot n^{o(\sqrt{k})}$ algorithm under ETH.

Reduction for PLANAR d-SCATTERED SET

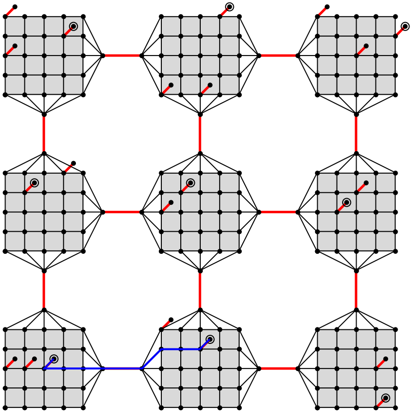
$S_{1,1}$: $(1,1)$ $(3,1)$ $(2,4)$	$S_{1,2}$: $(5,1)$ $(1,4)$ $(5,3)$	$S_{1,3}$: $(1,1)$ $(2,5)$ $(3,3)$
$S_{2,1}$: $(2,2)$ $(1,4)$	$S_{2,2}$: $(3,1)$ $(2,2)$	$S_{2,3}$: $(3,2)$ $(2,3)$
$S_{3,1}$: $(1,3)$ $(2,3)$ $(3,3)$	$S_{3,2}$: $(1,1)$ $(2,3)$	$S_{3,3}$: $(5,4)$ $(3,4)$

Reduction for PLANAR d-SCATTERED SET



$M = 10 \cdot n^{10}$

Reduction for Planar d-Scattered Set



$M = 10 \cdot n^{10}$ $d = 3M + n + 1$

Wrapping up

- We asked for a scattered set of size k^2, so the parameter blow-up is quadratic.

Wrapping up

- We asked for a scattered set of size k^2, so the parameter blow-up is quadratic.
- This gives an $f(k) \cdot n^{o(\sqrt{k})}$ lower bound for Planar d-Scattered Set.

Wrapping up

- We asked for a scattered set of size k^2, so the parameter blow-up is quadratic.
- This gives an $f(k) \cdot n^{o(\sqrt{k})}$ lower bound for Planar d-Scattered Set.
- **Upper bound**: DP on possible separators of the graph of interaction between vertices of the solution.

Wrapping up

- We asked for a scattered set of size k^2, so the parameter blow-up is quadratic.
- This gives an $f(k) \cdot n^{o(\sqrt{k})}$ lower bound for PLANAR d-SCATTERED SET.
- **Upper bound**: DP on possible separators of the graph of interaction between vertices of the solution.
- This graph is planar and hence has treewidth $\mathcal{O}(\sqrt{k})$.

Wrapping up

- We asked for a scattered set of size k^2, so the parameter blow-up is quadratic.
- This gives an $f(k) \cdot n^{o(\sqrt{k})}$ lower bound for $\textsc{Planar}$ d-$\textsc{Scattered Set}$.
- **Upper bound**: DP on possible separators of the graph of interaction between vertices of the solution.
- This graph is planar and hence has treewidth $\mathcal{O}(\sqrt{k})$.
- Plays well with the lower bound: the grid has asymptotically the worst possible treewidth.

Application 2: Unit Disk Independent Set

Unit Disk Independent Set

Input: A set of open disks of diameter 1 on the plane, integer k

Question: Can one select k pairwise disjoint disks?

Application 2: Unit Disk Independent Set

Unit Disk Independent Set

Input: A set of open disks of diameter 1 on the plane, integer k

Question: Can one select k pairwise disjoint disks?

- (Alber, Fiala) Unit Disk Independent Set can be solved in time $n^{\mathcal{O}(\sqrt{k})}$.

Application 2: Unit Disk Independent Set

Unit Disk Independent Set

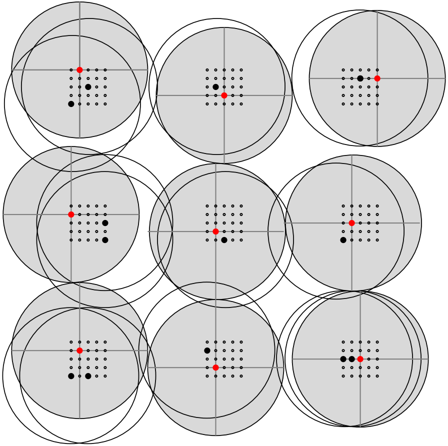
Input: A set of open disks of diameter 1 on the plane, integer k

Question: Can one select k pairwise disjoint disks?

- (Alber, Fiala) Unit Disk Independent Set can be solved in time $n^{\mathcal{O}(\sqrt{k})}$.
- **Now**: Again, by a reduction from Grid Tiling with $\leq$, we show that the problem is $\mathrm{W}[1]$-hard and does not admit an $f(k) \cdot n^{o(\sqrt{k})}$ algorithm under ETH.

Reduction for Unit Disk Independent Set

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



Wrap-up

- Pick $\varepsilon = 1/n^{10}$.

Wrap-up

- Pick $\varepsilon = 1/n^{10}$.
- For $(a, b) \in S_{i,j}$, put disk with centre in $(i + a\varepsilon, j + b\varepsilon)$.

Wrap-up

- Pick $\varepsilon = 1/n^{10}$.
- For $(a, b) \in S_{i,j}$, put disk with centre in $(i + a\varepsilon, j + b\varepsilon)$.
- It is easy to see that disks for (i, j) and $(i + 1, j)$ are disjoint iff $a \leq a'$.

Wrap-up

- Pick $\varepsilon = 1/n^{10}$.
- For $(a, b) \in S_{i,j}$, put disk with centre in $(i + a\varepsilon, j + b\varepsilon)$.
- It is easy to see that disks for (i, j) and $(i + 1, j)$ are disjoint iff $a \leq a'$.
  - Change of length from shifting by $n\varepsilon$ vertically will not compensate for change of length from shifting by ε horizontally.

Wrap-up

- Pick $\varepsilon = 1/n^{10}$.
- For $(a, b) \in S_{i,j}$, put disk with centre in $(i + a\varepsilon, j + b\varepsilon)$.
- It is easy to see that disks for (i, j) and $(i + 1, j)$ are disjoint iff $a \leq a'$.
  - Change of length from shifting by $n\varepsilon$ vertically will not compensate for change of length from shifting by ε horizontally.
- Hence the choice of disks models the Grid Tiling with $\leq$ instance.

Wrap-up

- Pick $\varepsilon = 1/n^{10}$.
- For $(a, b) \in S_{i,j}$, put disk with centre in $(i + a\varepsilon, j + b\varepsilon)$.
- It is easy to see that disks for (i, j) and $(i + 1, j)$ are disjoint iff $a \leq a'$.
  - Change of length from shifting by $n\varepsilon$ vertically will not compensate for change of length from shifting by ε horizontally.
- Hence the choice of disks models the GRID TILING WITH $\leq$ instance.
- As we ask for k^2 disks, the lower bound follows.

Conclusions

- Hardness of $3\mathrm{SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.

Conclusions

- Hardness of $3\mathrm{SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:

Conclusions

- Hardness of $3\mathrm{SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:
  - $\mathcal{O}^\star(2^{o(n)})$, $\mathcal{O}^\star(2^{o(\sqrt{n})})$ for classical complexity;

Conclusions

- Hardness of $3\mathrm{SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:
  - $\mathcal{O}^\star(2^{o(n)})$, $\mathcal{O}^\star(2^{o(\sqrt{n})})$ for classical complexity;
  - $\mathcal{O}^\star(2^{o(k)})$, $\mathcal{O}^\star(2^{o(\sqrt{k})})$, $\mathcal{O}^\star(2^{o(k \log k)})$, $\mathcal{O}^\star(2^{2^{o(k)}})$ for FPT problems;

Conclusions

- Hardness of $\mathrm{3SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:
  - $\mathcal{O}^\star(2^{o(n)})$, $\mathcal{O}^\star(2^{o(\sqrt{n})})$ for classical complexity;
  - $\mathcal{O}^\star(2^{o(k)})$, $\mathcal{O}^\star(2^{o(\sqrt{k})})$, $\mathcal{O}^\star(2^{o(k \log k)})$, $\mathcal{O}^\star(2^{2^{o(k)}})$ for FPT problems;
  - $f(k) \cdot n^{o(k)}$ and $f(k) \cdot n^{o(\sqrt{k})}$ for $\mathrm{W}[1]$-hard problems;

Conclusions

- Hardness of $\mathrm{3SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:
  - $\mathcal{O}^\star(2^{o(n)})$, $\mathcal{O}^\star(2^{o(\sqrt{n})})$ for classical complexity;
  - $\mathcal{O}^\star(2^{o(k)})$, $\mathcal{O}^\star(2^{o(\sqrt{k})})$, $\mathcal{O}^\star(2^{o(k \log k)})$, $\mathcal{O}^\star(2^{2^{o(k)}})$ for FPT problems;
  - $f(k) \cdot n^{o(k)}$ and $f(k) \cdot n^{o(\sqrt{k})}$ for $\mathrm{W}[1]$-hard problems;
  - many others that we did not mention.

Conclusions

- Hardness of $3\mathrm{SAT}$ turned out to be a very robust assumption for proving lower bounds on time complexity.
- Very different lower bounds:
  - $\mathcal{O}^\star(2^{o(n)})$, $\mathcal{O}^\star(2^{o(\sqrt{n})})$ for classical complexity;
  - $\mathcal{O}^\star(2^{o(k)})$, $\mathcal{O}^\star(2^{o(\sqrt{k})})$, $\mathcal{O}^\star(2^{o(k \log k)})$, $\mathcal{O}^\star(2^{2^{o(k)}})$ for FPT problems;
  - $f(k) \cdot n^{o(k)}$ and $f(k) \cdot n^{o(\sqrt{k})}$ for $\mathrm{W}[1]$-hard problems;
  - many others that we did not mention.
- **Optimality program**: understand the precise complexity of the problem by providing matching upper and lower bounds.

Exercises

Exercises 14.5–14.9, 14.13