CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Anti-Factor is FPT Parameterized by Treewidth and List Size (but Counting is Hard)

IPEC 2022

Dániel Marx[1] Govind S. Sankar[2] **Philipp Schepper**[1]

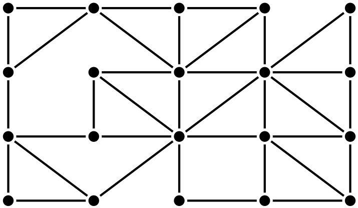
[1] CISPA Helmholtz Center for Information Security, Germany
[2] Duke University, Durham, USA

September 8, 2022

The Classical Matching Problem

Definition (Perfect Matching)

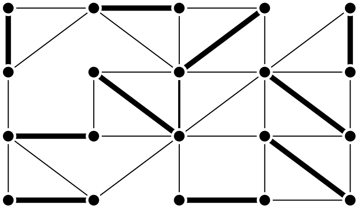
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) = 1$, for all $v \in V$.



The Classical Matching Problem

Definition (Perfect Matching)

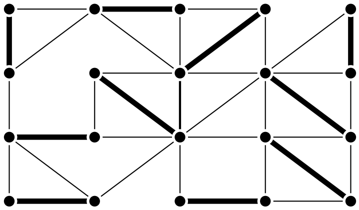
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) = 1$, for all $v \in V$.



The Classical Matching Problem

Definition (Perfect Matching)

Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) = 1$, for all $v \in V$.

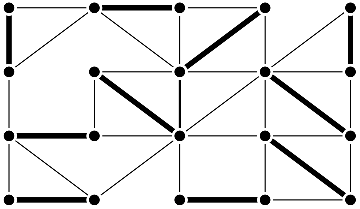


Well known to be poly-time solvable (Edmonds 1965).

The Classical Matching Problem

Definition (Perfect Matching)

Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) = 1$, for all $v \in V$.



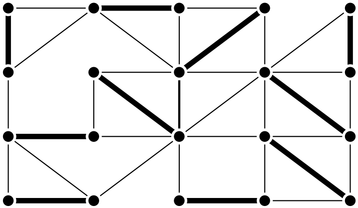
Well known to be poly-time solvable (Edmonds 1965).

Several generalizations studied (b-matching, even-degree subgraph, ...).
Usually reducible to Perfect Matching.

The Classical Matching Problem

Definition (Perfect Matching)

Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) = 1$, for all $v \in V$.



Well known to be poly-time solvable (Edmonds 1965).

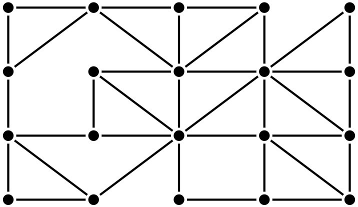
Several generalizations studied (b-matching, even-degree subgraph, ...).
Usually reducible to Perfect Matching.

Lovász introduced 1972 a general version called B-FACTOR.

The B-FACTOR Problem

Definition (B-FACTOR (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.



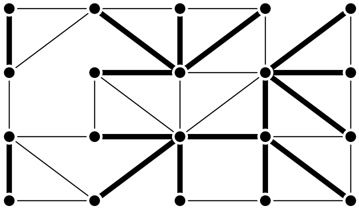
$B = \{1, 4\}$

The *B*-Factor Problem

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (*B*-Factor (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.



$B = \{1, 4\}$

The B-Factor Problem

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (B-Factor (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.

Complexity of B-Factor is understood.

Theorem (Cornuéjols 1988, Marx, Sankar, S. 2021)

Depending on B (conditions are known), B-Factor can be solved

The B-Factor Problem

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (B-Factor (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.

Complexity of B-Factor is understood.

Theorem (Cornuéjols 1988, Marx, Sankar, S. 2021)

Depending on B (conditions are known), B-Factor can be solved

- in polynomial time or

The B-Factor Problem

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (B-Factor (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.

Complexity of B-Factor is understood.

Theorem (Cornuéjols 1988, Marx, Sankar, S. 2021)

Depending on B (conditions are known), B-Factor can be solved

- in polynomial time or
- in time $(\max B + 1)^{\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given which is essentially optimal unter SETH.

The B-Factor Problem

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (B-Factor (Lovász 1972))

Fixed: A finite, non-empty set $B \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \in B$, for all $v \in V$.

Complexity of B-Factor is understood.

Theorem (Cornuéjols 1988, Marx, Sankar, S. 2021)

Depending on B (conditions are known), B-Factor can be solved

- in polynomial time or
- in time $(\max B + 1)^{\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given which is essentially optimal unter SETH.

Until now we *allowed* degrees. What changes if we *exclude* degrees?

The X-AntiFactor Problem

Definition (X-AntiFactor)

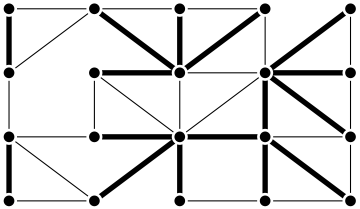
Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

The X-AntiFactor Problem

Definition (X-AntiFactor)

Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

$B = \{1, 4\}$:



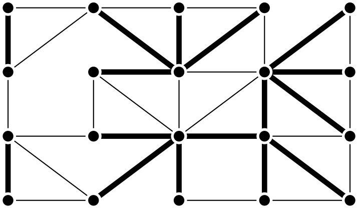
Solution for B-Factor

The X-ANTIFACTOR Problem

Definition (X-ANTIFACTOR)

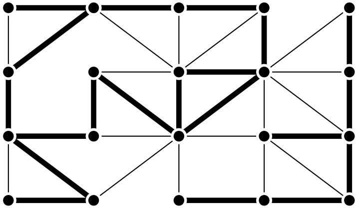
Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

$B = \{1, 4\}$:



Solution for B-FACTOR

$X = \{1, 4\}$:



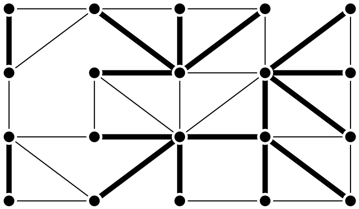
Complement of the solution for B-FACTOR

The X-AntiFactor Problem

Definition (X-AntiFactor)

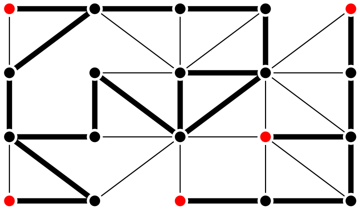
Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

$B = \{1, 4\}$:



Solution for B-Factor

$X = \{1, 4\}$:

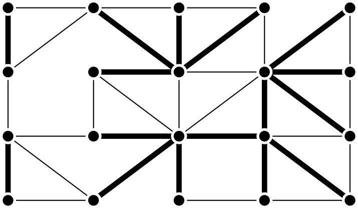


Complement of the solution for B-Factor does not work.

Definition (X-AntiFactor)

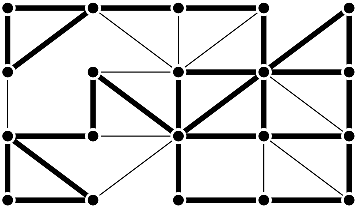
Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

$B = \{1, 4\}$:



Solution for B-Factor

$X = \{1, 4\}$:



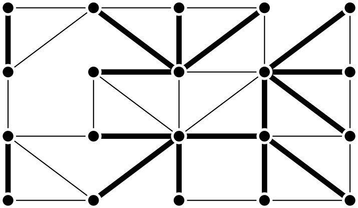
A correct solution for X-AntiFactor

The X-ANTIFACTOR Problem

Definition (X-ANTIFACTOR)

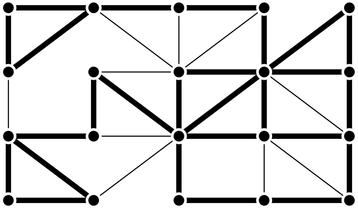
Fixed: A finite, non-empty set $X \subseteq \mathbb{N}$.
Input: A simple graph $G = (V, E)$.
Task: Check if there is a set $S \subseteq E$ such that $\deg_S(v) \notin X$, for all $v \in V$.

$B = \{1, 4\}$:



Solution for B-FACTOR

$X = \{1, 4\}$:



A correct solution for X-ANTIFACTOR

Which running time is needed to solve X-ANTIFACTOR?

Solving X-ANTIFACTOR

We extend the algorithm for B-FACTOR to solve X-ANTIFACTOR.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-ANTIFACTOR can be solved in time $(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Solving X-AntiFactor

We extend the algorithm for B-Factor to solve X-AntiFactor.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\mathrm{tw} + 1$ (the treewidth).

Solving X-AntiFactor

We extend the algorithm for B-Factor to solve X-AntiFactor.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\text{tw} + 1$ (the treewidth).
- For each vertex: Store the number of selected incident edges.

Solving X-AntiFactor

We extend the algorithm for B-Factor to solve X-AntiFactor.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\mathrm{tw} + 1$ (the treewidth).
- For each vertex: Store the number of selected incident edges.
- If a vertex has at least $\max X + 1$ selected incident edges, we do not care about the precise number.

Solving X-AntiFactor

We extend the algorithm for B-Factor to solve X-AntiFactor.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\mathrm{tw} + 1$ (the treewidth).
- For each vertex: Store the number of selected incident edges.
- If a vertex has at least $\max X + 1$ selected incident edges, we do not care about the precise number.
- Need $\max B + 2$ many states to encode this: 0, 1, ..., $\max B$, and "$\geq \max X + 1$" edges.

Solving X-AntiFactor

We extend the algorithm for B-Factor to solve X-AntiFactor.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\text{tw} + 1$ (the treewidth).
- For each vertex: Store the number of selected incident edges.
- If a vertex has at least $\max X + 1$ selected incident edges, we do not care about the precise number.
- Need $\max B + 2$ many states to encode this: 0, 1, . . . , $\max B$, and "$\geq \max X + 1$" edges.
- At most $(\max B + 2)^{\text{tw}+1}$ states needed.

Solving X-ANTIFACTOR

We extend the algorithm for B-FACTOR to solve X-ANTIFACTOR.

Theorem (Parameterize by $\max X$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-ANTIFACTOR can be solved in time $(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Intuition behind the base:

- Consider a vertex set of size at most $\text{tw} + 1$ (the treewidth).
- For each vertex: Store the number of selected incident edges.
- If a vertex has at least $\max X + 1$ selected incident edges, we do not care about the precise number.
- Need $\max B + 2$ many states to encode this: $0, 1, \ldots, \max B$, and "$\geq \max X + 1$" edges.
- At most $(\max B + 2)^{\text{tw}+1}$ states needed.

Is this algorithm optimal?

For example:
Is the $1002^{\text{tw}} n^{\mathcal{O}(1)}$ algorithm for $\{0, 999, 1000\}$-ANTIFACTOR optimal?

Excluding Only Few Numbers

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Excluding Only Few Numbers

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\mathrm{tw}} n^{\mathcal{O}(1)}$ to $4^{4\mathrm{tw}} n^{\mathcal{O}(1)} = 256^{\mathrm{tw}} n^{\mathcal{O}(1)}$.

Excluding Only Few Numbers

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\text{tw}} n^{\mathcal{O}(1)}$ to $4^{4\text{tw}} n^{\mathcal{O}(1)} = 256^{\text{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.

Excluding Only Few Numbers

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\mathrm{tw}} n^{\mathcal{O}(1)}$ to $4^{4\mathrm{tw}} n^{\mathcal{O}(1)} = 256^{\mathrm{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.
- Fix four solutions which give degree 0, 300, 500, and 999 to v, respectively.

Excluding Only Few Numbers

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\text{tw}} n^{\mathcal{O}(1)}$ to $4^{4\text{tw}} n^{\mathcal{O}(1)} = 256^{\text{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.
- Fix four solutions which give degree 0, 300, 500, and 999 to v, respectively.
- A future extension increases the degree of V by f.

Excluding Only Few Numbers

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\text{tw}} n^{\mathcal{O}(1)}$ to $4^{4\text{tw}} n^{\mathcal{O}(1)} = 256^{\text{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.
- Fix four solutions which give degree 0, 300, 500, and 999 to v, respectively.
- A future extension increases the degree of V by f.
- One of these four solutions can be combined with the new solution as $\{0+f, 300+f, 500+f, 999+f\} \neq \{0, 999, 1000\} = X$.

Excluding Only Few Numbers

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\mathrm{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\mathrm{tw}} n^{\mathcal{O}(1)}$ to $4^{4\mathrm{tw}} n^{\mathcal{O}(1)} = 256^{\mathrm{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.
- Fix four solutions which give degree 0, 300, 500, and 999 to v, respectively.
- A future extension increases the degree of V by f.
- One of these four solutions can be combined with the new solution as $\{0+f, 300+f, 500+f, 999+f\} \neq \{0, 999, 1000\} = X$.

Formal proof is based on an efficient computation of representative sets.

Excluding Only Few Numbers

Theorem (Parameterize by $|X|$)

Let $X \subseteq \mathbb{N}$ be finite and fixed. X-AntiFactor can be solved in time $(|X|+1)^{4\text{tw}} n^{\mathcal{O}(1)}$ assuming a tree decomposition of width tw is given.

Consider $X = \{0, 999, 1000\}$.

Running time:
Improves from $1002^{\text{tw}} n^{\mathcal{O}(1)}$ to $4^{4\text{tw}} n^{\mathcal{O}(1)} = 256^{\text{tw}} n^{\mathcal{O}(1)}$.

Intuition:

- Fix a vertex v.
- Fix four solutions which give degree 0, 300, 500, and 999 to v, respectively.
- A future extension increases the degree of V by f.
- One of these four solutions can be combined with the new solution as $\{0+f, 300+f, 500+f, 999+f\} \neq \{0, 999, 1000\} = X$.

Formal proof is based on an efficient computation of representative sets.

Once more: Is this algorithm optimal (up to constants in the exponent)?

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0 Construct a grid-like B-Factor instance.

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0 Construct a grid-like B-Factor instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0 Construct a grid-like B-Factor instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0 Construct a grid-like B-Factor instance.
1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.
2 Have one row for each group of variables.
3 Have one column for each clause of the formula.

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0 Construct a grid-like B-Factor instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

3 Have one column for each clause of the formula.

4 At the crossing points check if the assignment satisfies the clause.

Lower Bounds for B-FACTOR [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-FACTOR:

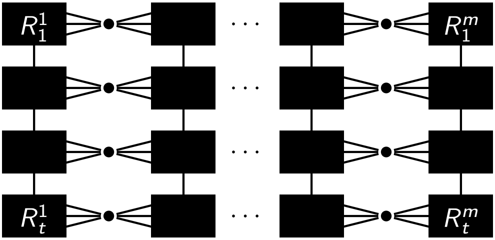
0 Construct a grid-like B-FACTOR instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

3 Have one column for each clause of the formula.

4 At the crossing points check if the assignment satisfies the clause.



Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

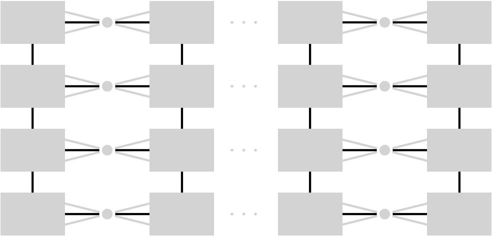
0 Construct a grid-like B-Factor instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

3 Have one column for each clause of the formula.

4 At the crossing points check if the assignment satisfies the clause.



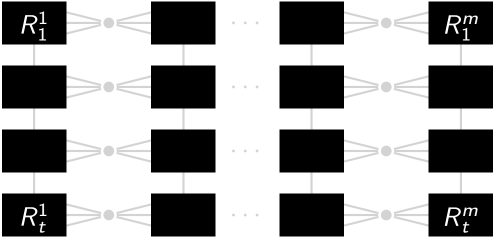
Main parts of the construction:

1 Grid-like structure:
Gives us a low treewidth

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

0. Construct a grid-like B-Factor instance.
1. Group $\approx \log(\max B)$ variables together.
   Encode the assignments by the number of selected incident edges of a vertex.
2. Have one row for each group of variables.
3. Have one column for each clause of the formula.
4. At the crossing points check if the assignment satisfies the clause.



Main parts of the construction:

1. Grid-like structure:
   Gives us a low treewidth
2. Relations:
   Check if the clauses are satisfied

Lower Bounds for B-FACTOR [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-FACTOR:

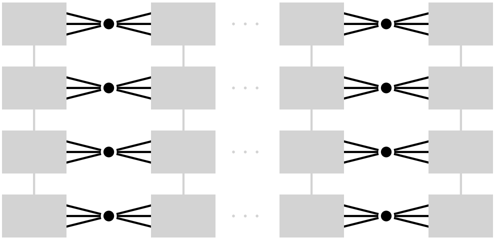
0 Construct a grid-like B-FACTOR instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

3 Have one column for each clause of the formula.

4 At the crossing points check if the assignment satisfies the clause.



Main parts of the construction:

1 Grid-like structure:
Gives us a low treewidth

2 Relations:
Check if the clauses are satisfied

3 Information vertices:
Transfer information/assignment between columns/clauses

Lower Bounds for B-Factor [Marx et al. '21]

Reduction from SAT with n variables and m clauses to B-Factor:

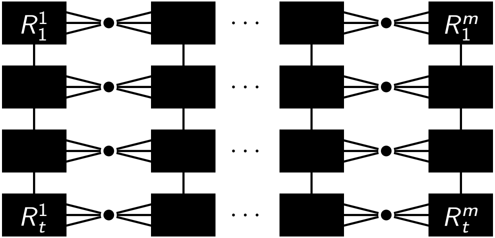
0 Construct a grid-like B-Factor instance.

1 Group $\approx \log(\max B)$ variables together.
Encode the assignments by the number of selected incident edges of a vertex.

2 Have one row for each group of variables.

3 Have one column for each clause of the formula.

4 At the crossing points check if the assignment satisfies the clause.



Main parts of the construction:

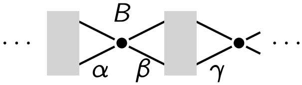
1 Grid-like structure:
Gives us a low treewidth

2 Relations:
Check if the clauses are satisfied

3 Information vertices:
Transfer information/assignment between columns/clauses

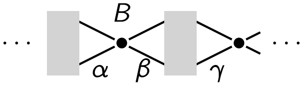
Parts 1 and 2: They can be rather easily adopted for X-AntiFactor.
Part 3: Behavior of information vertices changes dramatically.

Understanding Information Vertices



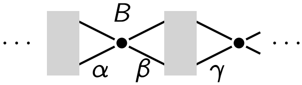
- The number of selected edges to the left, i.e., α encodes the assignment.

Understanding Information Vertices



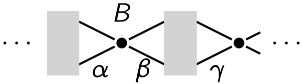
- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.

Understanding Information Vertices



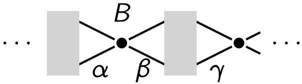
- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.

Understanding Information Vertices



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

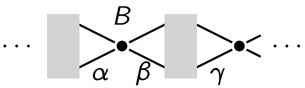
Understanding Information Vertices



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$B = \{0, 2\}$:

Understanding Information Vertices

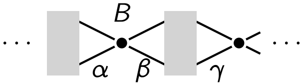


- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$$B = \{0, 2\}:$$

$$\begin{aligned} \alpha &\mapsto \beta &&\mapsto \gamma = 2 - \beta \\ 0 &\mapsto 0, 2 &&\mapsto 2, 0 \\ 1 &\mapsto 1 &&\mapsto 1 \\ 2 &\mapsto 0 &&\mapsto 2 \end{aligned}$$

Understanding Information Vertices



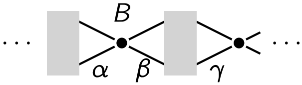
- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$$B = \{0, 2\}:$$

$$\begin{aligned} \alpha &\mapsto \beta &&\mapsto \gamma = 2 - \beta \\ 0 &\mapsto 0, 2 &&\mapsto 2, 0 \\ 1 &\mapsto 1 &&\mapsto 1 \\ 2 &\mapsto 0 &&\mapsto 2 \end{aligned}$$

Does not decrease.

Understanding Information Vertices



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$B = \{0, 2\}$:

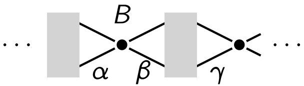
$\alpha \mapsto \beta$	$\mapsto \gamma = 2 - \beta$
$0 \mapsto 0, 2$	$\mapsto 2, 0$
$1 \mapsto 1$	$\mapsto 1$
$2 \mapsto 0$	$\mapsto 2$

Does not decrease.

$X = \{1\}$ ($B = \{0, 2, 3, 4, \dots\}$):

Understanding Information Vertices

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$B = \{0, 2\}$:

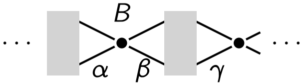
$$\begin{aligned} \alpha &\mapsto \beta &&\mapsto \gamma = 2 - \beta \\ 0 &\mapsto 0, 2 &&\mapsto 2, 0 \\ 1 &\mapsto 1 &&\mapsto 1 \\ 2 &\mapsto 0 &&\mapsto 2 \end{aligned}$$

Does not decrease.

$X = \{1\}$ $(B = \{0, 2, 3, 4, \dots\})$:

$$\begin{aligned} \alpha &\mapsto \beta &&\mapsto \gamma = 2 - \beta \\ 0 &\mapsto 0, 2 &&\mapsto 2, 0 \\ 1 &\mapsto 1, 2 &&\mapsto 1, 0 \\ 2 &\mapsto 0, 1, 2 &&\mapsto 2, 1, 0 \end{aligned}$$

Understanding Information Vertices



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$$B = \{0, 2\}:$$

$$\begin{array}{lll} \alpha \mapsto \beta & \mapsto \gamma = 2 - \beta \\ 0 \mapsto 0, 2 & \mapsto 2, 0 \\ 1 \mapsto 1 & \mapsto 1 \\ 2 \mapsto 0 & \mapsto 2 \end{array}$$

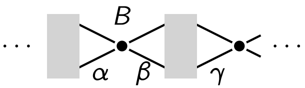
Does not decrease.

$$X = \{1\} \ (B = \{0, 2, 3, 4, \ldots\}):$$

$$\begin{array}{lll} \alpha \mapsto \beta & \mapsto \gamma = 2 - \beta \\ 0 \mapsto 0, 2 & \mapsto 2, 0 \\ 1 \mapsto 1, 2 & \mapsto 1, 0 \\ 2 \mapsto 0, 1, 2 & \mapsto 2, 1, 0 \end{array}$$

Can decrease: $0 \rightsquigarrow 2 \rightsquigarrow 1 \rightsquigarrow 0$

Understanding Information Vertices



- The number of selected edges to the left, i.e., α encodes the assignment.
- We need the same assignment for all clauses to be correct.
- Hence, the number of selected edges must not change.
  We allow increases in the number but no decreases.
- The relations compute γ based on β such that $\gamma = \max B - \beta$.

$B = \{0, 2\}$:

$$\begin{array}{llll} \alpha \mapsto & \beta & \mapsto & \gamma = 2 - \beta \\ 0 \mapsto & 0, 2 & \mapsto & 2, 0 \\ 1 \mapsto & 1 & \mapsto & 1 \\ 2 \mapsto & 0 & \mapsto & 2 \end{array}$$

Does not decrease.

$X = \{1\}$ $(B = \{0, 2, 3, 4, \dots\})$:

$$\begin{array}{llll} \alpha \mapsto & \beta & \mapsto & \gamma = 2 - \beta \\ 0 \mapsto & 0, 2 & \mapsto & 2, 0 \\ 1 \mapsto & 1, 2 & \mapsto & 1, 0 \\ 2 \mapsto & 0, 1, 2 & \mapsto & 2, 1, 0 \end{array}$$

Can decrease: $0 \rightsquigarrow 2 \rightsquigarrow 1 \rightsquigarrow 0$

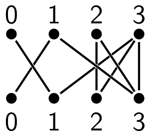
Have to use some other property of B, respective X!

Half-Induced Matchings

Fix a bipartite graph $G = (U \dot\cup V, E)$.

Definition (Induced Matching)

G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



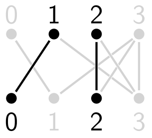
Half-Induced Matchings

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Fix a bipartite graph $G = (U \dot\cup V, E)$.

Definition (Induced Matching)

G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.

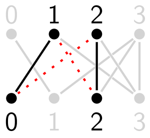


Half-Induced Matchings

Fix a bipartite graph $G = (U \dot\cup V, E)$.

Definition (Induced Matching)

G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.

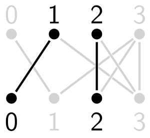


Half-Induced Matchings

Fix a bipartite graph $G = (U \dot{\cup} V, E)$.

Definition (Induced Matching)

G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



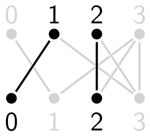
We relax this notation via *half-induced matchings*.

Half-Induced Matchings

Fix a bipartite graph $G = (U \,\dot\cup\, V, E)$.

Definition (Induced Matching)

G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



We relax this notation via *half-induced matchings*.

Definition (Half-Induced Matching)

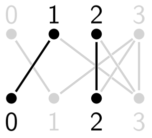
G has a *half-induced matching* of size ℓ if there are pairwise different $a_0, \ldots, a_{\ell-1} \in U$ and pairwise different $b_0, \ldots, b_{\ell-1} \in V$ such that (1) $(a_i, b_i) \in E$ for all i but (2) $(a_i, b_j) \notin E$ for all $j < i$.

Half-Induced Matchings

Fix a bipartite graph $G = (U \dot\cup V, E)$.

Definition (Induced Matching)

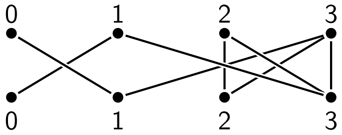
G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



We relax this notation via *half-induced matchings*.

Definition (Half-Induced Matching)

G has a *half-induced matching* of size ℓ if there are pairwise different $a_0, \ldots, a_{\ell-1} \in U$ and pairwise different $b_0, \ldots, b_{\ell-1} \in V$ such that (1) $(a_i, b_i) \in E$ for all i but (2) $(a_i, b_j) \notin E$ for all $j < i$.

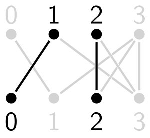


Half-Induced Matchings

Fix a bipartite graph $G = (U \dot\cup V, E)$.

Definition (Induced Matching)

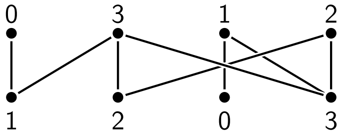
G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



We relax this notation via *half-induced matchings*.

Definition (Half-Induced Matching)

G has a *half-induced matching* of size ℓ if there are pairwise different $a_0, \ldots, a_{\ell-1} \in U$ and pairwise different $b_0, \ldots, b_{\ell-1} \in V$ such that (1) $(a_i, b_i) \in E$ for all i but (2) $(a_i, b_j) \notin E$ for all $j < i$.

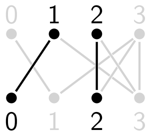


Half-Induced Matchings

Fix a bipartite graph $G = (U \dot{\cup} V, E)$.

Definition (Induced Matching)

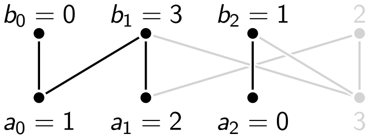
G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



We relax this notation via *half-induced matchings*.

Definition (Half-Induced Matching)

G has a *half-induced matching* of size ℓ if there are pairwise different $a_0, \ldots, a_{\ell-1} \in U$ and pairwise different $b_0, \ldots, b_{\ell-1} \in V$ such that (1) $(a_i, b_i) \in E$ for all i but (2) $(a_i, b_j) \notin E$ for all $j < i$.

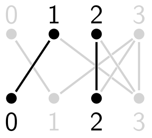


Half-Induced Matchings

Fix a bipartite graph $G = (U \dot{\cup} V, E)$.

Definition (Induced Matching)

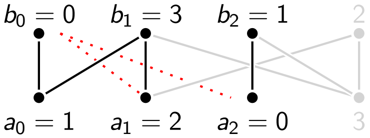
G has an induced matching of size ℓ, if there are two sets $A \subseteq U$ and $B \subseteq V$ of size ℓ such that the edges between A and B are a perfect matching of size ℓ.



We relax this notation via *half-induced matchings*.

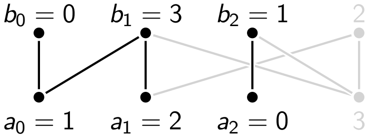
Definition (Half-Induced Matching)

G has a *half-induced matching* of size ℓ if there are pairwise different $a_0, \ldots, a_{\ell-1} \in U$ and pairwise different $b_0, \ldots, b_{\ell-1} \in V$ such that (1) $(a_i, b_i) \in E$ for all i but (2) $(a_i, b_j) \notin E$ for all $j < i$.



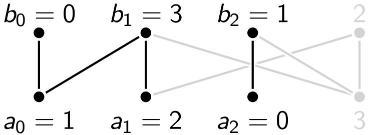
Fixing the Behavior of Information Vertices

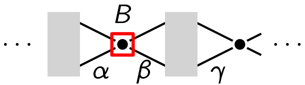
We use half-induced matchings to modify the information vertices.



Fixing the Behavior of Information Vertices

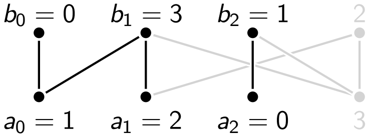
We use half-induced matchings to modify the information vertices.

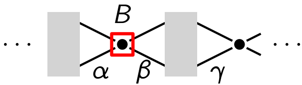




Before the modification.

We use half-induced matchings to modify the information vertices.





Before the modification.

$B = \{2\}$:

$\alpha \mapsto \beta \mapsto \gamma$

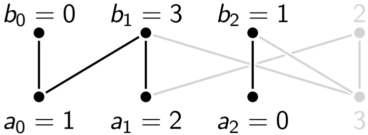
$0 \mapsto 2 \mapsto 0$

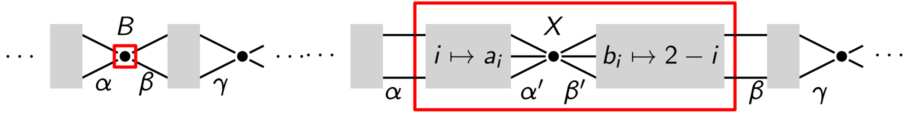
$1 \mapsto 1 \mapsto 1$

$2 \mapsto 0 \mapsto 2$

Fixing the Behavior of Information Vertices

We use half-induced matchings to modify the information vertices.





Before the modification.

After the modification.

$B = \{2\}$:

$\alpha \mapsto \beta \mapsto \gamma$

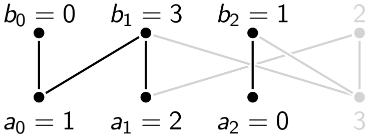
$0 \mapsto 2 \mapsto 0$

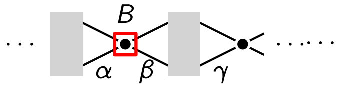
$1 \mapsto 1 \mapsto 1$

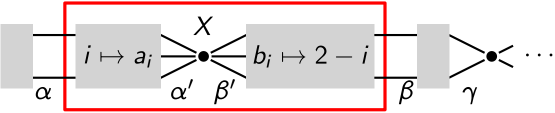
$2 \mapsto 0 \mapsto 2$

Fixing the Behavior of Information Vertices

We use half-induced matchings to modify the information vertices.







Before the modification.

After the modification.

$B = \{2\}$:

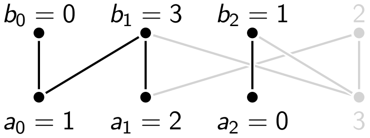
α	$\mapsto \beta$	$\mapsto \gamma$
0	$\mapsto 2$	$\mapsto 0$
1	$\mapsto 1$	$\mapsto 1$
2	$\mapsto 0$	$\mapsto 2$

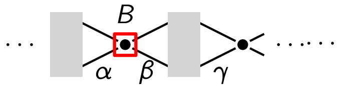
$X = \{0, 2, 3\}$ ($B = \{1, 4, 5, \ldots\}$):

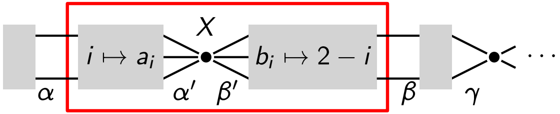
α	$\mapsto \alpha'$	$\mapsto \beta'$	$\mapsto \beta$	$\mapsto \gamma$
0	$\mapsto 1$	$\mapsto 0, 3$	$\mapsto 2, 1$	$\mapsto 0, 1$
1	$\mapsto 2$	$\mapsto 2, 3$	$\mapsto 1$	$\mapsto 1$
2	$\mapsto 0$	$\mapsto 1$	$\mapsto 0$	$\mapsto 0$

Fixing the Behavior of Information Vertices

We use half-induced matchings to modify the information vertices.







Before the modification.

$B = \{2\}$:

$\alpha \mapsto$	$\beta \mapsto$	γ
$0 \mapsto$	$2 \mapsto$	0
$1 \mapsto$	$1 \mapsto$	1
$2 \mapsto$	$0 \mapsto$	2

After the modification.

$X = \{0, 2, 3\}$ ($B = \{1, 4, 5, \dots\}$):

$\alpha \mapsto$	β	$\mapsto \gamma$
$0 \mapsto$	$2, 1$	$\mapsto 0, 1$
$1 \mapsto$	1	$\mapsto 1$
$2 \mapsto$	0	$\mapsto 0$

Lower Bounds

Theorem (Lower Bound)

Fix a finite set $X \subseteq \mathbb{N}$ such that X contains a half-induced matching of size h and $0 \in X$ and $\text{max-gap}(X) > 1$.
For any $\varepsilon > 0$, there is no $(h - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$ time algorithm for X-AntiFactor even if we are given a tree decomposition of width tw, unless SETH fails.

Lower Bounds

Theorem (Lower Bound)

Fix a finite set $X \subseteq \mathbb{N}$ such that X contains a half-induced matching of size h and $0 \in X$ and $\text{max-gap}(X) > 1$.
For any $\varepsilon > 0$, there is no $(h - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$ time algorithm for X-AntiFactor even if we are given a tree decomposition of width tw, unless SETH fails.

This bound does not match the improved upper bound based on $|X|$ but there are connections:

Lower Bounds

Theorem (Lower Bound)

Fix a finite set $X \subseteq \mathbb{N}$ such that X contains a half-induced matching of size h and $0 \in X$ and $\text{max-gap}(X) > 1$.
For any $\varepsilon > 0$, there is no $(h - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$ time algorithm for X-AntiFactor even if we are given a tree decomposition of width tw, unless SETH fails.

This bound does not match the improved upper bound based on $|X|$ but there are connections:

- Half-induced matchings imply lower bounds for the size of representative sets.

Lower Bounds

Theorem (Lower Bound)

Fix a finite set $X \subseteq \mathbb{N}$ such that X contains a half-induced matching of size h and $0 \in X$ and $\text{max-gap}(X) > 1$.
For any $\varepsilon > 0$, there is no $(h-\varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$ time algorithm for X-AntiFactor even if we are given a tree decomposition of width tw, unless SETH fails.

This bound does not match the improved upper bound based on $|X|$ but there are connections:

- Half-induced matchings imply lower bounds for the size of representative sets.
- We conjecture that the half-induced matchings are the actual base for the lower bound of the problem.

Counting Version

What changes if we *count* the number of solutions?

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$ algorithm.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:

Modify the information vertices in a better way:

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:

Modify the information vertices in a better way:

- Use interpolation tricks to simulate the original behavior.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:
Modify the information vertices in a better way:

- Use interpolation tricks to simulate the original behavior.
- Half-induced matchings can be avoided.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:

Modify the information vertices in a better way:

- Use interpolation tricks to simulate the original behavior.
- Half-induced matchings can be avoided.
- As a consequence a tight lower bound can be obtained.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:
Modify the information vertices in a better way:

- Use interpolation tricks to simulate the original behavior.
- Half-induced matchings can be avoided.
- As a consequence a tight lower bound can be obtained.

When parameterizing by the size of the set:

- The problem becomes #W[1]-hard.

Counting Version

What changes if we *count* the number of solutions?

Upper Bounds:

- $\max X + 2$ algorithm can be modified to include counting.
- Approach with representative sets is not applicable (as they forget solutions). Hence, no $|X|^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$ algorithm.

Lower Bounds:
Modify the information vertices in a better way:

- Use interpolation tricks to simulate the original behavior.
- Half-induced matchings can be avoided.
- As a consequence a tight lower bound can be obtained.

When parameterizing by the size of the set:

- The problem becomes #W[1]-hard.
- Result holds already for the case when only one number is excluded.

Conclusion

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum		
forbidden degree		
list size*		

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$	
list size*		

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum	$(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	
forbidden degree	no $(h + 1 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	
list size*		

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$	
	no $(h + 1 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$	
list size*	$(\lvert X\rvert + 1)^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$	

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$	
	no $(h + 1 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$	
list size*	$(\lvert X\rvert + 1)^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$	
	no $(\lvert X\rvert + 1 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$	

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$	$(\max X + 2)^{\text{tw}} n^{\mathcal{O}(1)}$
	no $(h + 1 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$	no $(\max X + 2 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$
list size*	$(\lvert X \rvert + 1)^{\mathcal{O}(\text{tw})} n^{\mathcal{O}(1)}$	
	no $(\lvert X \rvert + 1 - \varepsilon)^{\text{tw}} n^{\mathcal{O}(1)}$	

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	$(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$
	no $(h + 1 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	no $(\max X + 2 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$
list size*	$(\lvert X\rvert + 1)^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$	/
	no $(\lvert X\rvert+1-\varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	#W[1]-hard

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Conclusion

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Fix some finite, non-empty set $X \subseteq \mathbb{N}$.
Let h be the size of the largest half-induced matching in X.

Parameter	Decision Version	Counting Version
maximum forbidden degree	$(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	$(\max X + 2)^{\mathrm{tw}} n^{\mathcal{O}(1)}$
	no $(h + 1 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	no $(\max X + 2 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$
list size*	$(\lvert X\rvert + 1)^{\mathcal{O}(\mathrm{tw})} n^{\mathcal{O}(1)}$	/
	no $(\lvert X\rvert + 1 - \varepsilon)^{\mathrm{tw}} n^{\mathcal{O}(1)}$	#W[1]-hard

Always assuming

- the problem is not polynomial-time solvable (or trivial) because of X,
- a tree decomposition is given,
- and SETH or #SETH holds.

* We use a variation of X-AntiFactor where multiple set are allowed but the size of the sets is bounded by a constant.

Full version: `arXiv:2110.09369`

What Makes B-Factor Easy

Definition (Max-Gap)

For finite $\emptyset \neq B \subseteq \mathbb{N}$, max-gap B is the largest d such that at most d consecutive numbers are missing in B.
Formally, there is an a with $\{a, a+1, \ldots, a+d+1\} \cap B = \{a, a+d+1\}$.

What Makes B-Factor Easy

Definition (Max-Gap)

For finite $\emptyset \neq B \subseteq \mathbb{N}$, max-gap B is the largest d such that at most d consecutive numbers are missing in B.
Formally, there is an a with $\{a, a+1, \ldots, a+d+1\} \cap B = \{a, a+d+1\}$.

Example: $B = \{1, 3, 6\}$ has two gaps: one of size 1 and one of size $2 =$ max-gap B.

What Makes B-Factor Easy

Definition (Max-Gap)

For finite $\emptyset \neq B \subseteq \mathbb{N}$, max-gap B is the largest d such that at most d consecutive numbers are missing in B.
Formally, there is an a with $\{a, a+1, \ldots, a+d+1\} \cap B = \{a, a+d+1\}$.

Example: $B = \{1, 3, 6\}$ has two gaps: one of size 1 and one of size $2 =$ max-gap B.

Theorem (Cornuéjols 1988)

B-Factor is solvable in polynomial time if max-gap $B \leq 1$.

What Makes B-Factor Easy

Definition (Max-Gap)

For finite $\emptyset \neq B \subseteq \mathbb{N}$, max-gap B is the largest d such that at most d consecutive numbers are missing in B.
Formally, there is an a with $\{a, a+1, \ldots, a+d+1\} \cap B = \{a, a+d+1\}$.

Example: $B = \{1, 3, 6\}$ has two gaps: one of size 1 and one of size $2 =$ max-gap B.

Theorem (Cornuéjols 1988)

B-Factor is solvable in polynomial time if max-gap $B \leq 1$.

Cornuéjols implicitly showed NP-hardness when max-gap > 1 (using two different sets).
Stronger hardness than Cornuéjols was known implicitly in the literature (Dalmau and Ford 2003).

What Makes B-FACTOR Easy

CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Definition (Max-Gap)

For finite $\emptyset \neq B \subseteq \mathbb{N}$, max-gap B is the largest d such that at most d consecutive numbers are missing in B.
Formally, there is an a with $\{a, a+1, \ldots, a+d+1\} \cap B = \{a, a+d+1\}$.

Example: $B = \{1, 3, 6\}$ has two gaps: one of size 1 and one of size $2 =$ max-gap B.

Theorem (Cornuéjols 1988)

B-FACTOR is solvable in polynomial time if max-gap $B \leq 1$.

Cornuéjols implicitly showed NP-hardness when max-gap > 1 (using two different sets).
Stronger hardness than Cornuéjols was known implicitly in the literature (Dalmau and Ford 2003).

What do we know about the parameterized version, e.g. for treewidth?