Spanning Star Forest Problem

Adam Starak

May 16, 2019

Star definition

Definition

A **star** is a tree that:

Star definition

Definition

A **star** is a tree that:

(1) has at least 2 vertices.

Star definition

Definition

A **star** is a tree that:

(1) has at least 2 vertices.

(2) one vertex is adjacent to the rest.

Star definition

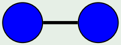
Definition

A **star** is a tree that:

(1) has at least 2 vertices.

(2) one vertex is adjacent to the rest.

Examples



Star definition

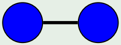
Definition

A **star** is a tree that:

(1) has at least 2 vertices.

(2) one vertex is adjacent to the rest.

Examples



Star definition

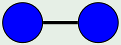
Definition

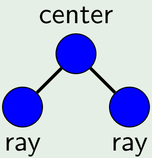
A **star** is a tree that:

(1) has at least 2 vertices.

(2) one vertex is adjacent to the rest.

Examples





Star definition

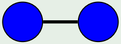
Definition

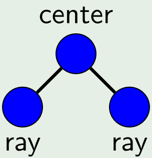
A **star** is a tree that:

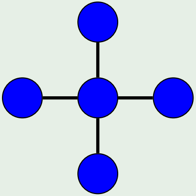
(1) has at least 2 vertices.

(2) one vertex is adjacent to the rest.

Examples





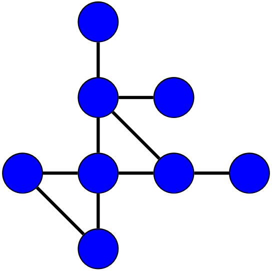


Spanning Star Forest Problem

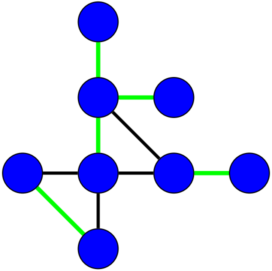
Definition

Given a graph G decide whether it has a *Spanning Star Forest*.

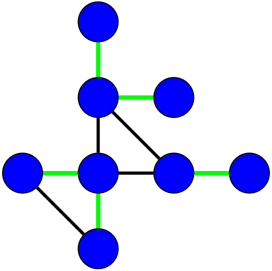
Example



Example



Example



Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Every star has at least 2 vertices.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Every star has at least 2 vertices.

G has no isolated vertices.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Every star has at least 2 vertices.

G has no isolated vertices.

Backward implication.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Every star has at least 2 vertices.

G has no isolated vertices.

Backward implication.

Proof on the board.

Spanning Star Forest Problem

Lemma

G contains a Spanning Star Forest if and only if it does not contain any isolated vertices.

Forward implication.

Let S - Spanning Star Forest of G.

Every vertex belongs to a star.

Every star has at least 2 vertices.

G has no isolated vertices.

Backward implication.

Proof on the board.

Corollary

Spanning Star Forest problem can be solved in **linear time**.

Minimal Spanning Star Forest

Minimal Spanning Star Forest

Given a pair (G, k) decide whether there exists a Spanning Star Forest of at most $\mathbf{k}$.

Minimal Spanning Star Forest

Minimal Spanning Star Forest

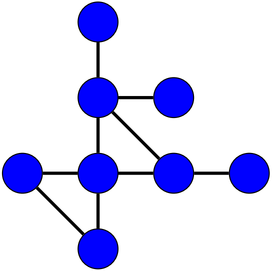
Given a pair (G, k) decide whether there exists a Spanning Star Forest of at most **k**.

Dominating Set

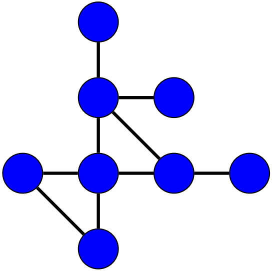
Given a pair (G, k) find a set $D \subseteq V(G)$ such that $|D| \leq k$ and every node is either in D or adjacent to D.

Examples

$(G, 3)$ for Dominating Set.

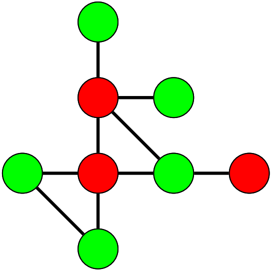


$(G, 3)$ for Minimal Spanning Star Forest.

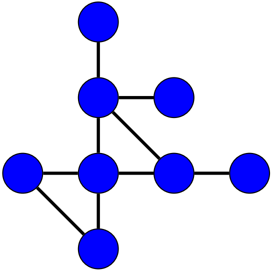


Examples

$(G, 3)$ for Dominating Set.

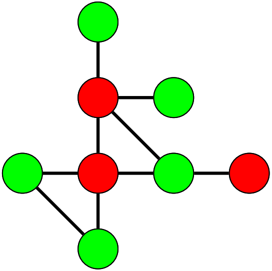


$(G, 3)$ for Minimal Spanning Star Forest.

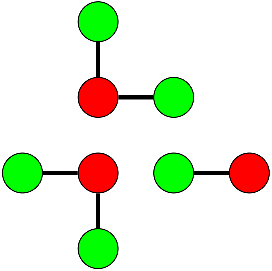


Examples

$(G, 3)$ for Dominating Set.

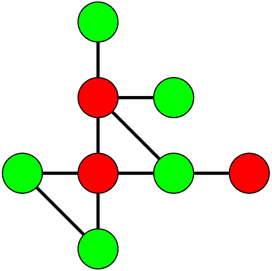


$(G, 3)$ for Minimal Spanning Star Forest.

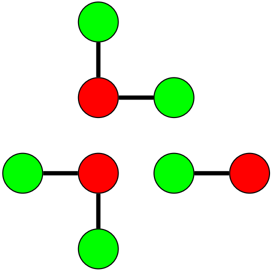


Examples

$(G, 3)$ for Dominating Set.



$(G, 3)$ for Minimal Spanning Star Forest.

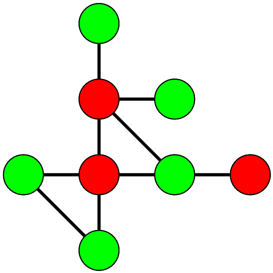


Observation

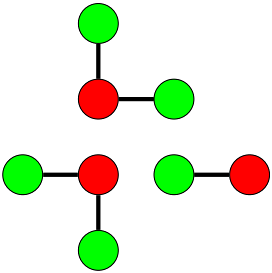
There is a relationship between:

Examples

$(G, 3)$ for Dominating Set.



$(G, 3)$ for Minimal Spanning Star Forest.



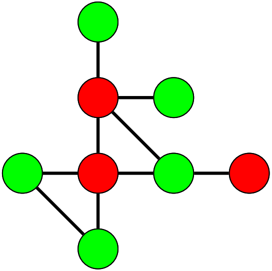
Observation

There is a relationship between:

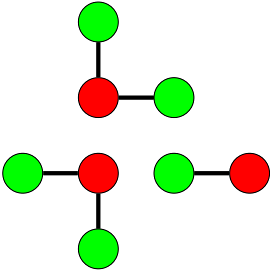
- **Centers** and **dominating** vertices.

Examples

$(G, 3)$ for Dominating Set.



$(G, 3)$ for Minimal Spanning Star Forest.



Observation

There is a relationship between:

- **Centers** and **dominating** vertices.
- **Rays** and **dominated** vertices.

Minimal Spanning Star Forest is NP-complete

Theorem

Minimal Spanning Star Forest is NP-complete.

Minimal Spanning Star Forest is NP-complete

Theorem

Minimal Spanning Star Forest is NP-complete.

Construction.

Minimal Spanning Star Forest is NP-complete

Theorem

Minimal Spanning Star Forest is NP-complete.

Construction.

Replace every isolated vertex $\mathbf{v}$ with an isolated edge $(\mathbf{v}, \mathbf{v}')$.

Minimal Spanning Star Forest is NP-complete

Theorem

Minimal Spanning Star Forest is NP-complete.

Construction.

Replace every isolated vertex $\mathbf{v}$ with an isolated edge $(\mathbf{v}, \mathbf{v}')$.

Lemma

(G, k) has a solution if and only if (G', k) has one.

Minimal Spanning Star Forest is NP-complete

Theorem

Minimal Spanning Star Forest is NP-complete.

Construction.

Replace every isolated vertex $\mathbf{v}$ with an isolated edge $(\mathbf{v}, \mathbf{v}')$.

Lemma

(G, k) has a solution if and only if (G', k) has one.

Proof on the board.

Reverse reduction

Construction

$$(G', k') = \begin{cases} (G, 0), & \text{if } G \text{ contains an isolated vertex.} \\ (G, k), & \text{otherwise.} \end{cases}$$

Reverse reduction

Construction

$$(G', k') = \begin{cases} (G, 0), & \text{if } G \text{ contains an isolated vertex.} \\ (G, k), & \text{otherwise.} \end{cases}$$

Lemma

(G, k) has a solution if and only if (G', k) has one.

Reverse reduction

Construction

$$(G', k') = \begin{cases} (G, 0), & \text{if } G \text{ contains an isolated vertex.} \\ (G, k), & \text{otherwise.} \end{cases}$$

Lemma

(G, k) has a solution if and only if (G', k) has one.

Corollary

Dominating Set and Minimal Spanning Star Forest are interreductible.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Suppose contrary, $\mathcal{A}$ is the algorithm.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Suppose contrary, $\mathcal{A}$ is the algorithm.

(G, k) - Dominating Set instance.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Suppose contrary, $\mathcal{A}$ is the algorithm.

(G, k) - Dominating Set instance.

(G', k) - Minimal Spanning Star Forest instance

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Suppose contrary, $\mathcal{A}$ is the algorithm.

(G, k) - Dominating Set instance.

(G', k) - Minimal Spanning Star Forest instance

Apply algorithm $\mathcal{A}$.

Transfering theorems

Theorem

Unless SETH fails, there is no algorithm solving Dominating Set in time $\mathcal{O}^*(n^{k-\epsilon})$ for $\epsilon > 0$.

The theorem is also true for Minimal Spanning Star Forest.

Proof.

Suppose contrary, $\mathcal{A}$ is the algorithm.

(G, k) - Dominating Set instance.

(G', k) - Minimal Spanning Star Forest instance

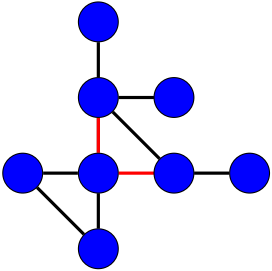
Apply algorithm $\mathcal{A}$.

Contradiction.

Spanning Star Forest Extension Problem

Definition

Given a graph G and a set of **forced edges** $M \subseteq E(G)$, find a Spanning Star Forest S such that $M \subseteq E(S)$.

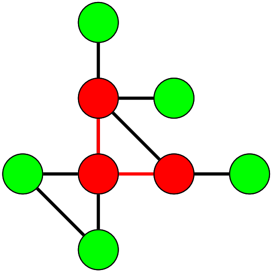


Spanning Star Forest Extension Problem

Definition

Given a graph G and a set of **forced edges** $M \subseteq E(G)$, find a Spanning Star Forest S such that $M \subseteq E(S)$.

$F = V(M)$
$R = V \setminus F$



Instance normalization

(R1) Isolated vertex $\rightarrow$ no instance.

Instance normalization

(R1) Isolated vertex $\rightarrow$ no instance.

(R2) $G[M]$ not a Star Forest $\rightarrow$ no instance.

Instance normalization

(R1) Isolated vertex $\to$ no instance.

(R2) $G[M]$ not a Star Forest $\to$ no instance.

(R3) M has a star S of size ≥ 3 $\to$ remove rays' edges and contract rays.

Instance normalization

(R1) Isolated vertex $\rightarrow$ no instance.

(R2) $G[M]$ not a Star Forest $\rightarrow$ no instance.

(R3) M has a star S of size $\geq 3 \rightarrow$ remove rays' edges and contract rays.

Corollary

After exhaustive application of the R3, M is a matching.

Instance normalization 2

Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

Instance normalization 2

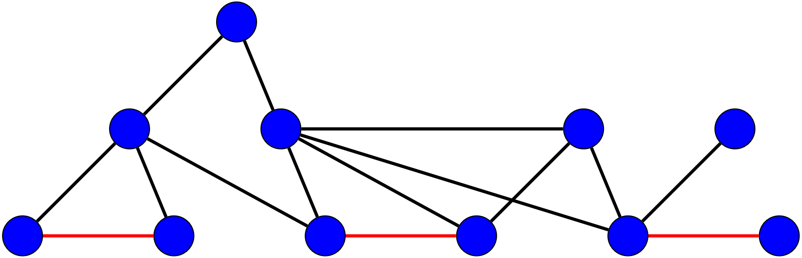
$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

$G_P = G \setminus G_{NP}$

Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

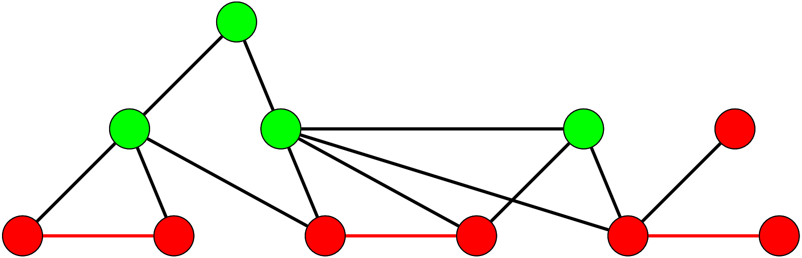
$G_P = G \setminus G_{NP}$



Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

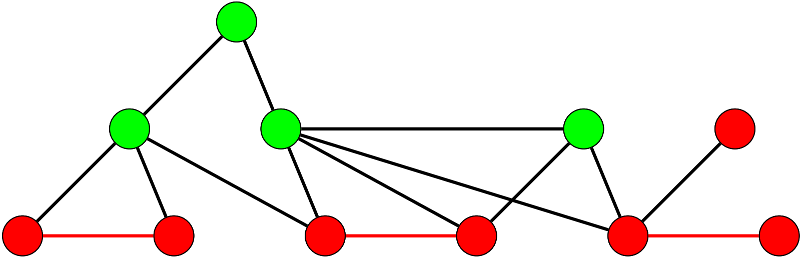
$G_P = G \setminus G_{NP}$



Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

$G_P = G \setminus G_{NP}$



Corollary

G_P has always a solution.

Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

$G_P = G \setminus G_{NP}$

Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$

$G_P = G \setminus G_{NP}$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Instance normalization 2

$$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$$
$$G_P = G \setminus G_{NP}$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Instance normalization 2

$$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$$
$$G_P = G \setminus G_{NP}$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Instance normalization 2

$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$
$G_P = G \setminus G_{NP}$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Partition G into (G_P, G_{NP}).

Instance normalization 2

$$
\begin{aligned}
G_{NP} &= \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\} \\
G_P &= G \setminus G_{NP}
\end{aligned}
$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Partition G into (G_P, G_{NP}).

Find a solution S' for G_P.

Instance normalization 2

$$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$$
$$G_P = G \setminus G_{NP}$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Partition G into (G_P, G_{NP}).

Find a solution S' for G_P.

$S \cup S'$ is a solution for G.

Instance normalization 2

$$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$$
$$G_P = G \setminus G_{NP}$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Partition G into (G_P, G_{NP}).

Find a solution S' for G_P.

$S \cup S'$ is a solution for G.

Forward implication.

Instance normalization 2

$$G_{NP} = \{v : v \in F \text{ or } v \text{ is isolated in } G \setminus F\}$$
$$G_P = G \setminus G_{NP}$$

Lemma

(G, M) has a solution if and only if (G_{NP}, M) has one.

Backward implication.

Let S be a solution for G_{NP}.

Partition G into (G_P, G_{NP}).

Find a solution S' for G_P.

$S \cup S'$ is a solution for G.

Forward implication.

Proof on the board.

Reduction Rules

(R1) Isolated vertex $\rightarrow$ no instance.

(R2) $G[M]$ not a Star Forest $\rightarrow$ no instance.

(R3) M has a star S of size ≥ 3 $\rightarrow$ remove rays' edges and contract rays.

Reduction Rules

(R1) Isolated vertex $\rightarrow$ no instance.

(R2) $G[M]$ not a Star Forest $\rightarrow$ no instance.

(R3) M has a star S of size ≥ 3 $\rightarrow$ remove rays' edges and contract rays.

(R4) Apply $G = G_{NP}$.

SSFE is NP-complete

Theorem

There exists a reduction from CNF-SAT to SSFE.

SSFE is NP-complete

ϕ CNF-formula.

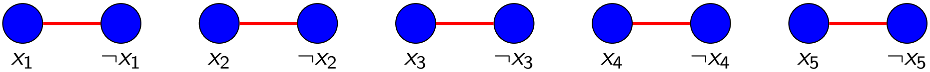
$$(x_1 \lor x_2 \lor \neg x_4) \land (\neg x_1 \lor x_3 \lor \neg x_5) \land (x_4 \lor x_5 \lor \neg x_1)$$

SSFE is NP-complete

ϕ CNF-formula.

Variable $\mathbf{x_i}$ $\rightarrow$ **vertices** $x_i, \neg x_i$ and **marked edge** $(x_i, \neg x_i)$.

$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$



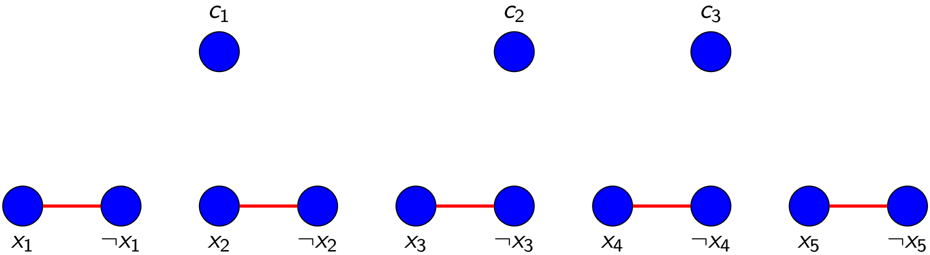
SSFE is NP-complete

ϕ CNF-formula.

Variable $\mathbf{x_i}$ $\rightarrow$ **vertices** $x_i, \neg x_i$ and **marked edge** $(x_i, \neg x_i)$.

Clause C_i $\rightarrow$ **vertex** c_i.

$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$



SSFE is NP-complete

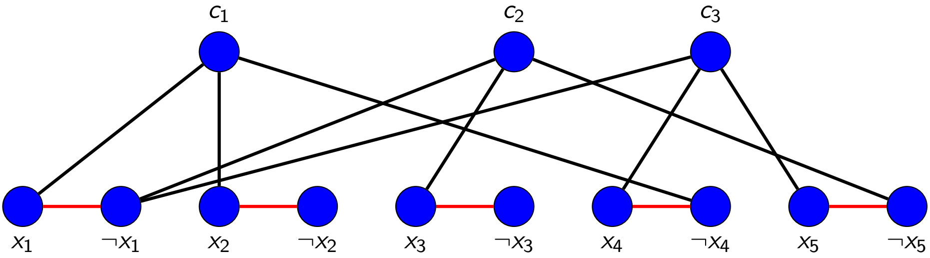
ϕ CNF-formula.

Variable $\mathbf{x_i}$ $\rightarrow$ **vertices** $x_i, \neg x_i$ and **marked edge** $(x_i, \neg x_i)$.

Clause C_i $\rightarrow$ **vertex** c_i.

literal $\neg x_k$ in clause c_i $\rightarrow$ **edge** $(\neg x_k, c_i)$.

$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$

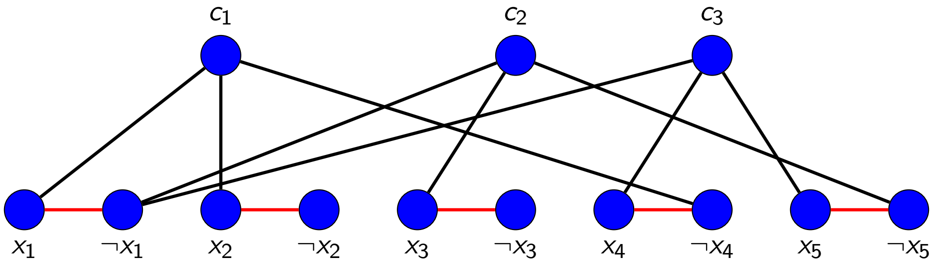


SSFE is NP-complete

Lemma

ϕ is satisfiable if and only there exists a SSFE.

$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$



Reverse reduction

Observation

There is a relationship between:

- **M** and **variables**.
- **R** and **clauses**.

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Construction.

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Construction.

$(G, M) \rightarrow$ SSFE instance

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Construction.

$(G, M) \rightarrow$ SSFE instance

$(G', M') \rightarrow$ Exhaustive application of (R1)-(R5).

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Construction.

$(G, M) \rightarrow$ SSFE instance

$(G', M') \rightarrow$ Exhaustive application of (R1)-(R5).

$\phi \rightarrow$ formula of $|M'|$ variables and $|R|$ clauses.

Reverse reduction

Theorem

There is a reduction from SSFE to CNF-SAT

Construction.

$(G, M) \rightarrow$ SSFE instance

$(G', M') \rightarrow$ Exhaustive application of (R1)-(R5).

$\phi \rightarrow$ formula of $|M'|$ variables and $|R|$ clauses.

Lemma

(G, M) has a SSF $\iff \phi$ is satisfiable.

Proof on the board.

Wrap up

Wrap up

Corollary

Spanning Star Forest Extension and CNF-SAT are interreductible.

Wrap up

Corollary

Spanning Star Forest Extension and CNF-SAT are interreductible.

CNF-SAT	SSFE
n variables	n marked edges
m clauses	$\lvert R \rvert = m$

Wrap up

Corollary

Spanning Star Forest Extension and CNF-SAT are interreductible.

CNF-SAT	SSFE
n variables	n marked edges
m clauses	$\|R\| = m$

Observation 1

Theorems for CNF-SAT paramterized by the number of **variables** can be transfered to Spanning Star Forest Extension parameterized by the number of **marked edges**.

Wrap up

Corollary

Spanning Star Forest Extension and CNF-SAT are interreductible.

CNF-SAT	SSFE
n variables	n marked edges
m clauses	$\lvert R \rvert = m$

Observation 2

Theorems for CNF-Sat parameterized by the number of **clauses** can be transfered to Spanning Star Forest Extension paramtereized by $\lvert \mathbf{R} \rvert$.

Summary

So far we proved that the following problems are interreductible:

Summary

So far we proved that the following problems are interreductible:

Dominating Set	Minimal SSF
CNF-Sat n variables	SSFE $\|M\|$
CNF-Sat m clauses	SSFE $\|R\|$

$|M|$ parametrization

Theorem

CNF-Sat with n variables doesn't admit any kernel.

$|M|$ parametrization

Theorem

CNF-Sat with n variables doesn't admit any kernel.

Consequently, nor does SSFE parameterized by $|M|$.

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), ..., (x_p, k)$

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), ..., (x_p, k)$

Output: (x', k).

such that:

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), \ldots, (x_p, k)$

Output: (x', k).

such that:

(1) $k' \leq POLY(k + \log(p))$.

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), \ldots, (x_p, k)$

Output: (x', k).

such that:

(1) $k' \leq POLY(k + \log(p))$.

(2) $(x', k') \in \mathcal{P} \iff (x_i, k) \in \mathcal{P}$ for some $1 \leq i \leq p$

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), ..., (x_p, k)$

Output: (x', k).

such that:

(1) $k' \leq POLY(k + \log(p))$.

(2) $(x', k') \in \mathcal{P} \iff (x_i, k) \in \mathcal{P}$ for some $1 \leq i \leq p$

Theorem

SSFE parameterized by $|M|$ doesn't admit a kernel.

$|M|$ parametrization

Proving nonexistence of a kernel for a problem $\mathcal{P}$

Design an algorithm $\mathcal{A}$:

Input: $(x_1, k), (x_2, k), \ldots, (x_p, k)$

Output: (x', k).

such that:

(1) $k' \leq POLY(k + \log(p))$.

(2) $(x', k') \in \mathcal{P} \iff (x_i, k) \in \mathcal{P}$ for some $1 \leq i \leq p$

Theorem

SSFE parameterized by $|M|$ doesn't admit a kernel.

Proof on the board.

$|R|$ parametrization

Theorem

CNF-Sat parameterized with m clauses has a linear kernel.

$|R|$ parametrization

Theorem

CNF-Sat parameterized with m clauses has a linear kernel.

Consequently, so does SSFE parameterized by $|R|$.

$|R|$ parametrization

Quadratic kernel

If there exists a vertex $v \in R$ such that $deg_G(v) > k$, then remove v.

Proof on the board.

$|R|$ parametrization

Quadratic kernel

If there exists a vertex $v \in R$ such that $deg_G(v) > k$, then remove v.

Proof on the board.

Corollary

After exhaustive application of the rule, an instance contains:

$|R|$ parametrization

Quadratic kernel

If there exists a vertex $v \in R$ such that $deg_G(v) > k$, then remove v.

Proof on the board.

Corollary

After exhaustive application of the rule, an instance contains:

$\leq 2k^2$ edges.

$|R|$ parametrization

Quadratic kernel

If there exists a vertex $v \in R$ such that $deg_G(v) > k$, then remove v.

Proof on the board.

Corollary

After exhaustive application of the rule, an instance contains:

$\leq 2k^2$ edges.

$\leq k + 2k^2$ vertices.

SSFE parameterized by $tw(G)$

Dynamic table. $dp[t, f]$ where t is node from a tree decomposition and $f : X_t \to \{true, false\}$ defined as follows:

SSFE parameterized by $tw(G)$

Dynamic table. $dp[t, f]$ where t is node from a tree decomposition and $f : X_t \to \{true, false\}$ defined as follows:

$$f(v) = \begin{cases} true, & v \in F \text{ and } v \text{ is a center.} \\ true, & v \in R \text{ and } v \text{ is a part of a Star Tree.} \\ false, & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

Dynamic table. $dp[t, f]$ where t is node from a tree decomposition and $f : X_t \to \{true, false\}$ defined as follows:

$$f(v) = \begin{cases} true, & v \in F \text{ and } v \text{ is a center.} \\ true, & v \in R \text{ and } v \text{ is a part of a Star Tree.} \\ false, & \text{otherwise.} \end{cases}$$

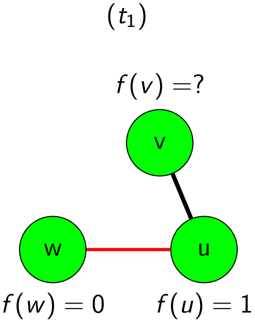
Assume we have everything finished **except for a join node**.

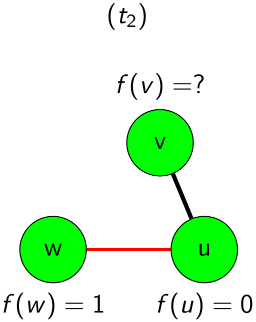
SSFE parameterized by $tw(G)$

Join nodes t_1, t_2

SSFE parameterized by $tw(G)$

Join nodes t_1**,** t_2

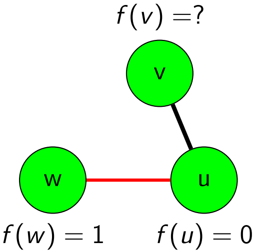




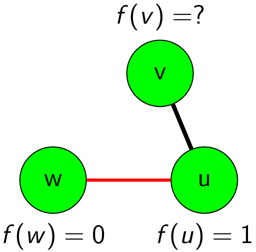
SSFE parameterized by $tw(G)$

Join nodes t_1**,** t_2

(t_1)

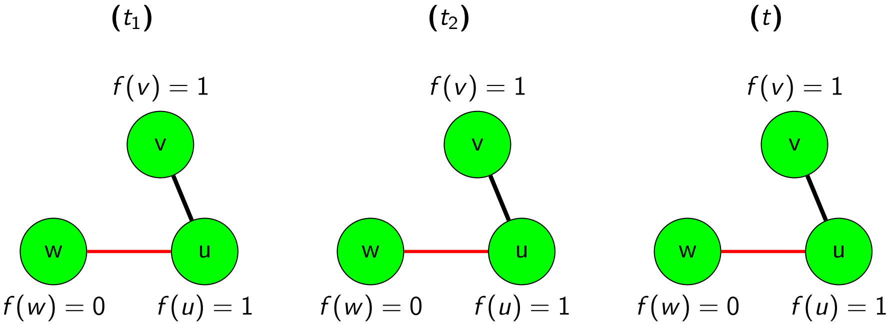


(t_2)



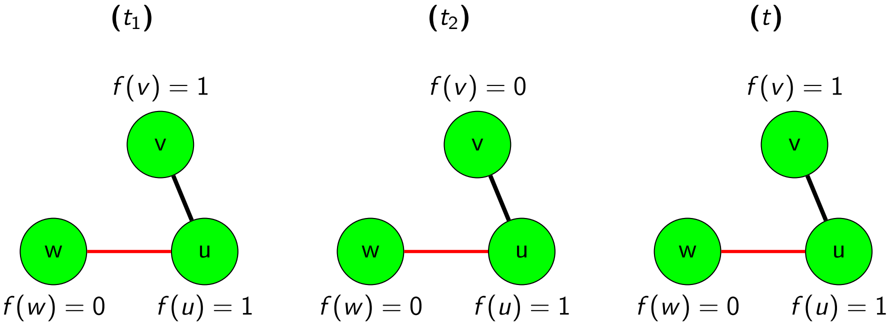
$C_1 =$ For every vertex $v \in (F \cap X_t)$ it holds $f(v) = f_1(v) = f_2(v)$

Join nodes t_1, t_2



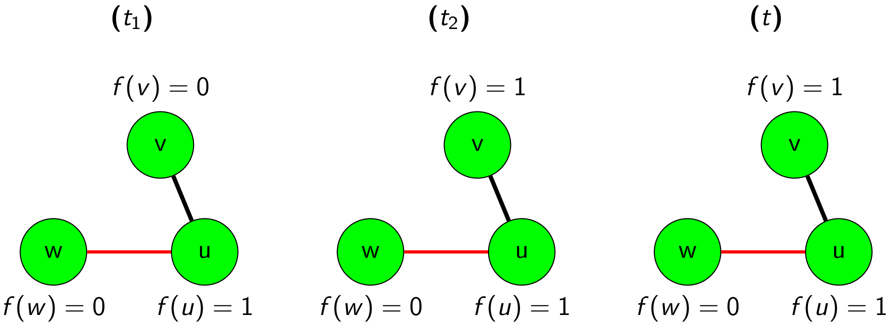
SSFE parameterized by $tw(G)$

Join nodes t_1, t_2



SSFE parameterized by $tw(G)$

Join nodes t_1, t_2



$C_2 =$ For all $v \in R$ it holds $f(v) = f_1(v) \vee f_2(v)$.

SSFE parameterized by $tw(G)$

SSFE parameterized by $tw(G)$

$$dp[t, f] = \begin{cases} \bigvee_{f_1, f_2} dp[t_1, f_1] \wedge dp[t_2, f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

$$dp[t, f] = \begin{cases} \bigvee_{f_1, f_2} dp[t_1, f_1] \wedge dp[t_2, f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ false, & \text{otherwise.} \end{cases}$$

There are $\mathbf{2}^{|\mathbf{X_t}|}$ functions.

SSFE parameterized by $tw(G)$

$$dp[t,f] = \begin{cases} \bigvee_{f_1,f_2} dp[t_1,f_1] \wedge dp[t_2,f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ false, & \text{otherwise.} \end{cases}$$

There are $\mathbf{2^{|X_t|}}$ functions.

Naive approach: Each f in $\mathcal{O}^*(2^{|X_t|})$.

SSFE parameterized by $tw(G)$

$$dp[t,f] = \begin{cases} \bigvee_{f_1,f_2} dp[t_1,f_1] \wedge dp[t_2,f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ false, & \text{otherwise.} \end{cases}$$

There are $\mathbf{2^{|X_t|}}$ functions.

Naive approach: Each f in $\mathcal{O}^*(2^{|X_t|})$.

Total complexity: $\mathcal{O}^*(4^{|X_t|})$.

SSFE parameterized by $tw(G)$

$$dp[t,f] = \begin{cases} \bigvee_{f_1,f_2} dp[t_1,f_1] \wedge dp[t_2,f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

There are $\mathbf{2^{|X_t|}}$ functions.

Naive approach: Each f in $\mathcal{O}^*(2^{|X_t|})$.

Total complexity: $\mathcal{O}^*(4^{|X_t|})$.

Cover product: all functions f in time $\mathcal{O}^*(2^{|X_t|})$

SSFE parameterized by $tw(G)$

$$dp[t,f] = \begin{cases} \bigvee_{f_1,f_2} dp[t_1,f_1] \wedge dp[t_2,f_2], & \text{if } C_1 \text{ and } C_2 \text{ holds.} \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

There are $\mathbf{2^{|X_t|}}$ functions.

Naive approach: Each f in $\mathcal{O}^*(2^{|X_t|})$.

Total complexity: $\mathcal{O}^*(4^{|X_t|})$.

Cover product: all functions f in time $\mathcal{O}^*(2^{|X_t|})$

Assuming leaf, introduce and forget are easy:

Lower bound

SSFE parameterized by $tw(G)$

SSFE parameterized by treewidth can be solved in $\mathcal{O}^*(2^{tw(G)})$.

Lower bound

SSFE parameterized by $tw(G)$

SSFE parameterized by treewidth can be solved in $\mathcal{O}^*(2^{tw(G)})$.

Lower bound

Assuming SETH, $\mathcal{O}^*(2^{tw(G)})$ is optimal.

Proof on the board.