Spanning Star Forest Problem

Adam Starak

May 13, 2019

Star definition

Definition

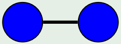
A star is a tree of size at least 2 for which at most one vertex has a degree greater than 1.

Star definition

Definition

A star is a tree of size at least 2 for which at most one vertex has a degree greater than 1.

Examples

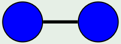


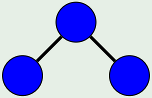
Star definition

Definition

A star is a tree of size at least 2 for which at most one vertex has a degree greater than 1.

Examples



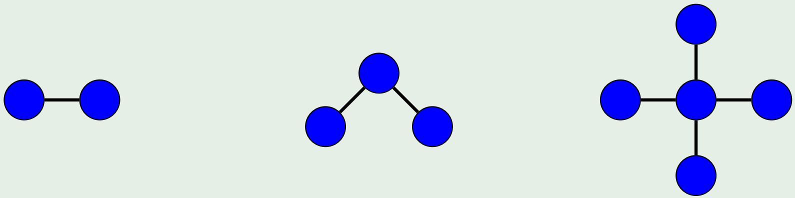


Star definition

Definition

A star is a tree of size at least 2 for which at most one vertex has a degree greater than 1.

Examples

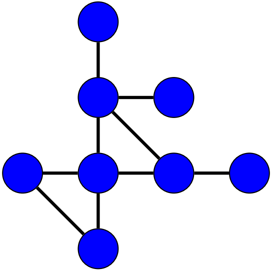


Spanning Star Forest Problem

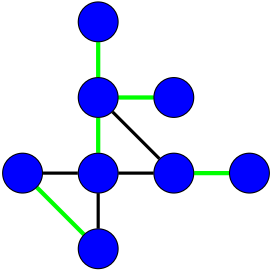
Definition

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

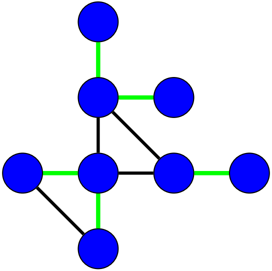
Example



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. Trivial, Every vertex belongs to a tree of size at least 2. Thus, it's degree is at least 1.

Backward implication

Suppose G has no isolated vertices.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Backward implication

Suppose G has no isolated vertices.

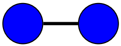
Induction by the number of vertices in G.

Let $n = 2$.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.



Let $n = 2$.

Graph is a correct Spanning Star Forest.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there does not exist a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Backward implication

Suppose G has no isolated vertices.

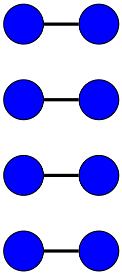
Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there does not exist a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Graph G is a correct Spanning Star Forest.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

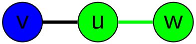
Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.

Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.



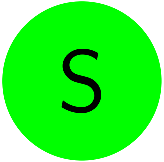
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

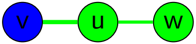
Now, let's try to add vertex v to the solution S.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.



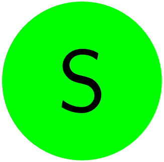
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

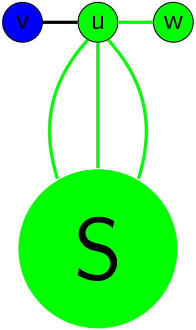
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

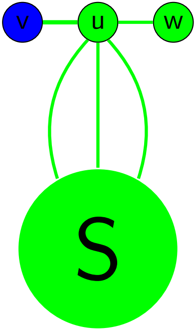
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

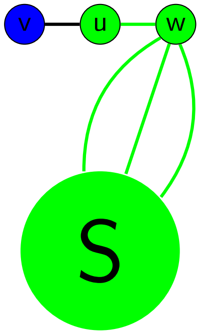
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.



Backward implication

Suppose G has no isolated vertices.

Induction by the number of vertices in G.

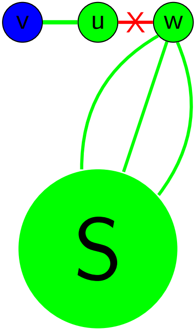
Inductive step.

Assume the condition holds for all graphs of size at most n.

Suppose there exists a vertex v such that $G \setminus \{v\}$ has no isolated vertices.

Let S be a solution for a graph $G \setminus \{v\}$.

Now, let's try to add vertex v to the solution S.



Parametrization by the number of stars

Definition

Given a pair (G, k) find a Spanning Star Forest S such that S has at most k connected components.

Parametrization by the number of stars

Definition

Given a pair (G, k) find a Spanning Star Forest S such that S has at most k connected components.

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.

Parametrization by the number of stars

Definition

Given a pair (G, k) find a Spanning Star Forest S such that S has at most k connected components.

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.

The problem is NP-Complete

NP-completeness - membership in NP

Trivial. Given S check if it is a Spanning Star Forest and if it has at most k connected components.

Construction:

NP-completeness - hardness

Construction: Let (G, k) be an instance of Dominating Set Problem.

NP-completeness - hardness

Construction: Let (G, k) be an instance of Dominating Set Problem.

We create G' as follows: for every isolated vertex v introduce a vertex v' and an edge (v, v').

NP-completeness - hardness

Construction: Let (G, k) be an instance of Dominating Set Problem.

We create G' as follows: for every isolated vertex v introduce a vertex v' and an edge (v, v').

Lemma

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

NP-completeness - hardness

Construction: Let (G, k) be an instance of Dominating Set Problem.

We create G' as follows: for every isolated vertex v introduce a vertex v' and an edge (v, v').

Lemma

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

Backward implication.
Set of centers represents a correct dominating set of G.

NP-completeness: Forward implication

Forward implication.

NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

Assume S contains an isolated vertex. Trivially, $v \in D$.

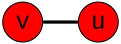
NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

Assume S contains an isolated vertex. Trivially, $v \in D$.



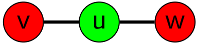
NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

Assume S contains an isolated vertex. Trivially, $v \in D$.



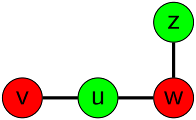
NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

Assume S contains an isolated vertex. Trivially, $v \in D$.



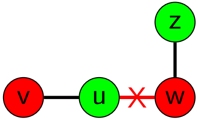
NP-completeness: Forward implication

Forward implication.

Without a loss of generality, if there exists a solution for (G, k), then there exists a solution of minimal size. Let D be such a solution.

We create solution S as follows: For every $v \in V(G') \setminus D$ add an edge (v, u) to the solution where $u \in D$.

Assume S contains an isolated vertex. Trivially, $v \in D$.



Reduction from SSF to Dom-Set

Construction

Given (G, k) transform the instance as following:

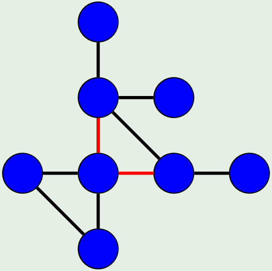
$$(G', k') = \begin{cases} (G, 0), & \text{if } G \text{ contains an isolated vertex.} \\ (G, k), & \text{otherwise.} \end{cases}$$

Spanning Star Forest Extension Problem

Definition

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

Example



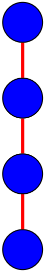
Instance normalization

(R1) If G has an isolated vertex, it is a no instance.

Instance normalization

(R1) If G has an isolated vertex, it is a no instance.

(R2) If G has a path of size 3 made from isolated edges, then it is a no instance

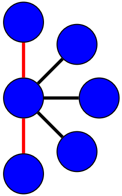


Instance normalization

(R1) If G has an isolated vertex, it is a no instance.

(R2) If G has a path of size 3 made from isolated edges, then it is a no instance

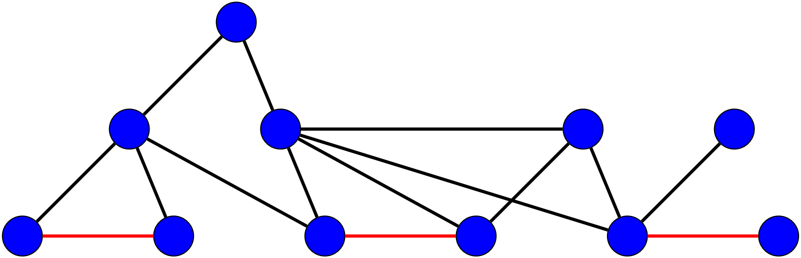
(R3) If G has a path of size 2 made from isolated edges, then remove all the vertices adjacent to the center.



Instance normalization 2

$G_P \;= \{v : u, v \notin V(F) \text{ and } \exists (u,v) \in E(G)\}.$

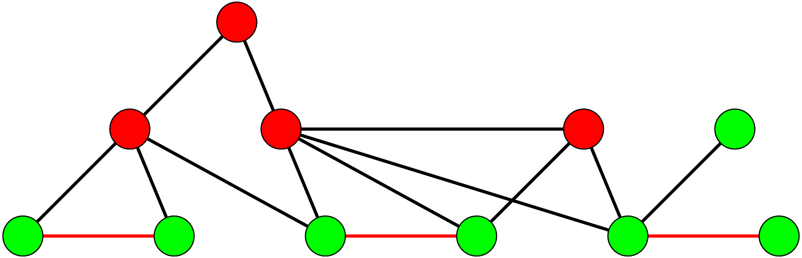
$G_{NP} \;= G \setminus G_P.$



Instance normalization 2

$G_P \; = \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}.$

$G_{NP} \; = G \setminus G_P.$

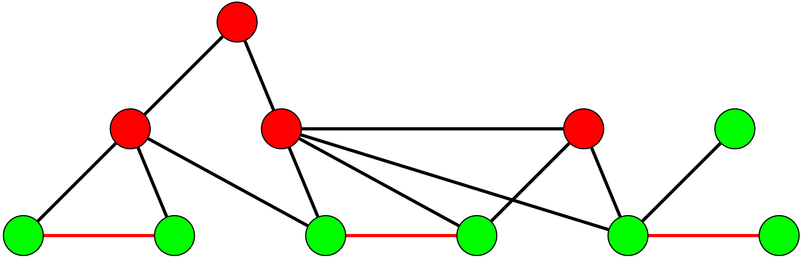


Instance normalization 2

$G_P \;= \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}.$

$G_{NP} \;= G \setminus G_P.$

G_P has always a SSF because it does not contain any isolated edges and for all $v \in V(G_P)$ $deg_{G_P}(v) \neq 0$.



Instance normalization 3

$G_P = \{v : u, v \notin V(F) \text{ and } \exists(u,v) \in E(G)\}.$

$G_{NP} = G \setminus G_P.$

Lemma

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

Instance normalization 3

$G_P = \{v : u, v \notin V(F) \text{ and } \exists (u,v) \in E(G)\}.$

$G_{NP} = G \setminus G_P.$

Lemma

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

Backward implication.

Instance normalization 3

$G_P = \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}$.

$G_{NP} = G \setminus G_P$.

Lemma

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

Backward implication.

Let S be a solution for G_{NP}.

Instance normalization 3

$G_P \; = \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}.$

$G_{NP} \; = G \setminus G_P.$

Lemma

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

Backward implication.

Let S be a solution for G_{NP}.

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

Instance normalization 3

$G_P = \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}.$

$G_{NP} = G \setminus G_P.$

Lemma

(G, F) has a solution if and only if (G_{NP}, F) 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 3

$G_P \;= \{v : u, v \notin V(F) \text{ and } \exists(u, v) \in E(G)\}.$

$G_{NP} \;= G \setminus G_P.$

Lemma

(G, F) has a solution if and only if (G_{NP}, F) 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 3

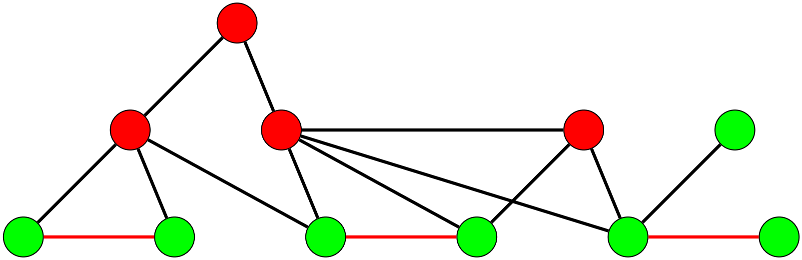
$G_P = \{v : u, v \notin V(F) \text{ and } \exists (u, v) \in E(G)\}.$

$G_{NP} = G \setminus G_P.$

Lemma

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

Forward implication.



SSFE is NP-complete

SSFE is NP-complete

Membership in NP

Trivial. Given a solution S check if all isolated edges are included and if it is a correct SSF.

SSFE - Hardness

SSFE - Hardness

Reduction from 3-SAT: Given formula ϕ do the following.

SSFE - Hardness

Reduction from 3-SAT: Given formula ϕ do the following.

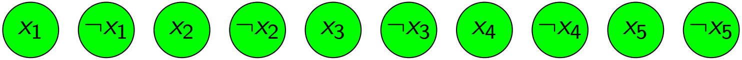
$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$

SSFE - Hardness

Reduction from 3-SAT: Given formula ϕ do the following.

1. For each variable x_i introduce vertices v_{x_i}, $v_{\neg x_i}$.

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

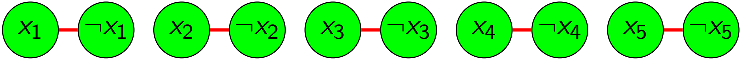


SSFE - Hardness

Reduction from 3-SAT: Given formula ϕ do the following.

1. For each variable x_i introduce vertices v_{x_i}, $v_{\neg x_i}$.
2. For each v_{x_i}, $v_{\neg x_i}$ introduce an isolated edge.

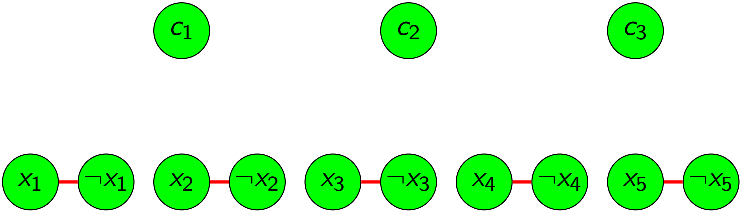
$$(x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_5) \wedge (x_4 \vee x_5 \vee \neg x_1)$$



Reduction from 3-SAT: Given formula ϕ do the following.

1. For each variable x_i introduce vertices v_{x_i}, $v_{\neg x_i}$.
2. For each v_{x_i}, $v_{\neg x_i}$ introduce an isolated edge.
3. For each clause c_i introduce vertex v_{c_i}.

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

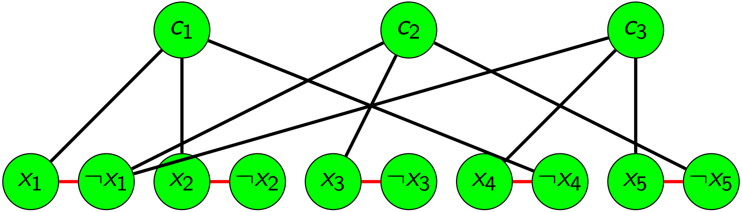


SSFE - Hardness

Reduction from 3-SAT: Given formula ϕ do the following.

1. For each variable x_i introduce vertices v_{x_i}, $v_{\neg x_i}$.
2. For each v_{x_i}, $v_{\neg x_i}$ introduce an isolated edge.
3. For each clause c_i introduce vertex v_{c_i}.
4. If literal l exists in clause c introduce an edge between them.

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



SSFE reverse reduction

SSFE reverse reduction

Theorem

There exists a reduction from 3-SAT to SSFE.

SSFE reverse reduction

Theorem

There exists a reduction from 3-SAT to SSFE.

Corollary

3-SAT and SSFE are equally hard

Parametrizations

Parametrizations

Number of isolated edges $|F|$.

Parametrizations

Number of isolated edges $|F|$.

Number of vertices $|V(G) \setminus V(F)|$.

Parametrizations

Number of isolated edges $|F|$.

Number of vertices $|V(G) \setminus V(F)|$.

Treewidth.

Number of isolated edges $|F|$

Number of isolated edges $|F|$

Simple algorithm

Branch on every isolated edge and check whether chosen vertices form a Spanning Star forest.

Number of isolated edges $|F|$

Simple algorithm

Branch on every isolated edge and check whether chosen vertices form a Spanning Star forest.

Complexity: $\mathcal{O}^*(2^{|F|})$.

Number of isolated edges $|F|$

Simple algorithm

Branch on every isolated edge and check whether chosen vertices form a Spanning Star forest.

Complexity: $\mathcal{O}^*(2^{|F|})$.

SETH

let δ_q be the infimum of the set of constants c for which there exists an algorithm solving q-SAT in time $\mathcal{O}^*(2^{cn})$. Then:

$$\lim_{q \to \infty} \delta_q = 1$$

SSFE parameterized by $|F|$ doesn't invoke kernel.

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

Question: Does there exist an instance (x', k) such that satisfies the following conditions:

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

Question: Does there exist an instance (x', k) such that satisfies the following conditions:

(1) $k' \leq poly(\max |(x_i, k)| + log(t))$

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

Question: Does there exist an instance (x', k) such that satisfies the following conditions:

(1) $k' \leq poly(\max|(x_i, k)| + log(t))$

(2) $(x', k') \in Q$ if and only if $(x_i, k) \in Q$ for some $i \leq t$.

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

Question: Does there exist an instance (x', k) such that satisfies the following conditions:

(1) $k' \leq poly(\max |(x_i, k)| + log(t))$

(2) $(x', k') \in Q$ if and only if $(x_i, k) \in Q$ for some $i \leq t$.

For a more general idea, read chapter 15 from Parameterized Algorithms.

SSFE parameterized by $|F|$ doesn't invoke kernel.

Proving nonexistence of a kernel

Input: $(x_1, k), (x_2, k), ..., (x_t, k)$ where (x_i, k) - instance of a given problem Q.

Question: Does there exist an instance (x', k) such that satisfies the following conditions:

(1) $k' \leq poly(\max|(x_i, k)| + log(t))$

(2) $(x', k') \in Q$ if and only if $(x_i, k) \in Q$ for some $i \leq t$.

For a more general idea, read chapter 15 from Parameterized Algorithms.

Example

Given instances $(G_1, k), (G_2, k), ..., (G_t, k)$ of k-Path problem return $(\bigcup_{i=1}^{t} G_i, k)$

SSFE parameterized by $|F|$ doesn't invoke kernel - Proof

Proof on the board.

SSFE parameterized by $|V(G)| \setminus |V(F)|$

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

Assume there exists a SSF S in $G \setminus \{v\}$.

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

Assume there exists a SSF S in $G \setminus \{v\}$.

Then, we "set" at most k centers.

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

Assume there exists a SSF S in $G \setminus \{v\}$.

Then, we "set" at most k centers.

In graph G $deg_G(v) > k$. So, there exists isolated edge (u, w) such that $deg_S(u) = deg_S(w) = 1$ and either $(v, u) \in E(G)$ or $(v, w) \in E(G)$.

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

Assume there exists a SSF S in $G \setminus \{v\}$.

Then, we "set" at most k centers.

In graph G $deg_G(v) > k$. So, there exists isolated edge (u, w) such that $deg_S(u) = deg_S(w) = 1$ and either $(v, u) \in E(G)$ or $(v, w) \in E(G)$.

Number of edges: $\leq 2k^2$

SSFE parameterized by $|V(G)| \setminus |V(F)|$

Quadratic kernel

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

Proof:

Assume there exists a SSF S in $G \setminus \{v\}$.

Then, we "set" at most k centers.

In graph G $deg_G(v) > k$. So, there exists isolated edge (u, w) such that $deg_S(u) = deg_S(w) = 1$ and either $(v, u) \in E(G)$ or $(v, w) \in E(G)$.

Number of edges: $\leq 2k^2$

Number of edges: $\leq k + 2k^2$.

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:

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

$$f_{v \to p}(u) = \begin{cases} p, & \text{if u=v.} \\ f(u), & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

Leaf node. An empty graph is a correct Spanning Star Forest.

$$dp[t, f] = true$$

SSFE parameterized by $tw(G)$

Introduce vertex $v \in V(F)$.

SSFE parameterized by $tw(G)$

Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

SSFE parameterized by $tw(G)$

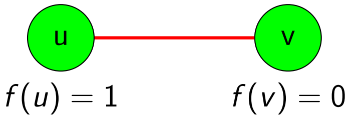
Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

$$dp[t, f_{v \to p}] = \begin{cases} \mathit{false}, & \text{if } u \in X_t \text{ and } f(u) = p. \\ dp[t', f], & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

$$dp[t, f_{v \to p}] = \begin{cases} false, & \text{if } u \in X_t \text{ and } f(u) = p. \\ dp[t', f], & \text{otherwise.} \end{cases}$$



SSFE parameterized by $tw(G)$

Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

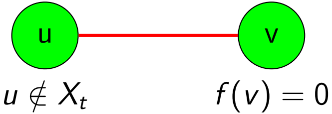
$$dp[t, f_{v \to p}] = \begin{cases} false, & \text{if } u \in X_t \text{ and } f(u) = p. \\ dp[t', f], & \text{otherwise.} \end{cases}$$

$f(u) = 0$ $\quad$ $f(v) = 1$

SSFE parameterized by $tw(G)$

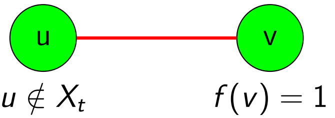
Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

$$dp[t, f_{v \to p}] = \begin{cases} \textit{false}, & \text{if } u \in X_t \text{ and } f(u) = p. \\ dp[t', f], & \text{otherwise.} \end{cases}$$



Introduce vertex $v \in V(F)$.
We assign *false* if both vertices of an isolated vertex (u, v) have the same value.

$$dp[t, f_{v \to p}] = \begin{cases} \mathit{false}, & \text{if } u \in X_t \text{ and } f(u) = p. \\ dp[t', f], & \text{otherwise.} \end{cases}$$



SSFE parameterized by $tw(G)$

Introduce vertex $v \notin V(F)$.

SSFE parameterized by $tw(G)$

Introduce vertex $v \notin V(F)$.
Vertex is isolated. Thus, the only correct possibility is to assign 0.

SSFE parameterized by $tw(G)$

Introduce vertex $v \notin V(F)$.
Vertex is isolated. Thus, the only correct possibility is to assign 0.

$$dp[t, f_{v \to p}] = \begin{cases} dp[t', f], & \text{if } p = 0. \\ false, & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \in F$

SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \in F$
We can skip this part. We assigned proper values during vertex introduction.

SSFE parameterized by $tw(G)$

Forget $v \in V(F)$

SSFE parameterized by $tw(G)$

Forget $v \in V(F)$
The vertex is covered anyway. We get the maxial value.

SSFE parameterized by $tw(G)$

Forget $v \in V(F)$
The vertex is covered anyway. We get the maxial value.

$$dp[t, f_{|v}] = dp[t', f_{v\to 0}] \vee dp[t', f_{v\to 1}]$$

SSFE parameterized by $tw(G)$

Forget $v \notin V(F)$

SSFE parameterized by $tw(G)$

Forget $v \notin V(F)$
The vertex needs to be covered. So, we pass the value if and only if 1 is assigned to vertex v.

Forget $v \notin V(F)$

The vertex needs to be covered. So, we pass the value if and only if 1 is assigned to vertex v.

$$dp[t, f_{|v}] = \begin{cases} dp[t', f], & \text{if f(v)=1.} \\ false, & \text{otherwise.} \end{cases}$$

SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \notin F$**,** $v \notin V(F)$

SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \notin F$, $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center ($f(u) = 1$):

SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \notin F$**,** $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center ($f(u) = 1$):

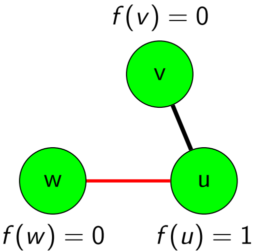
$$dp[t, f_{v\to 1}] = \begin{cases} dp[t', f_{v\to 0}] \vee dp[t', f_{v\to 1}], & \text{if } f(u) = 1 \\ dp[t', f_{v\to 1}], & \text{otherwise.} \end{cases}$$

$$dp[t, f_{v\to 0}] = dp[t', f_{v\to 0}]$$

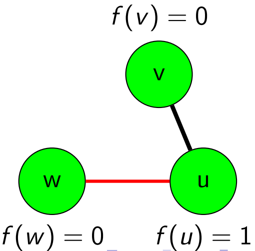
Introduce edge $(u, v) \notin F$**,** $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center $(f(u) = 1)$:

$$dp[t, f_{v\to1}] = \begin{cases} dp[t', f_{v\to0}] \vee dp[t', f_{v\to1}], & \text{if } f(u) = 1 \\ dp[t', f_{v\to1}], & \text{otherwise.} \end{cases}$$
$$dp[t, f_{v\to0}] = dp[t', f_{v\to0}]$$

Before



After



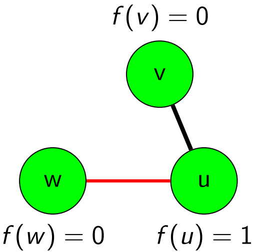
SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \notin F$**,** $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center ($f(u) = 1$):

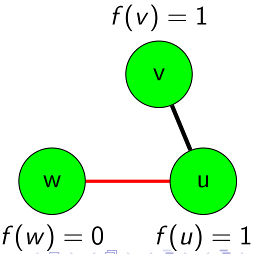
$$dp[t, f_{v\to 1}] = \begin{cases} dp[t', f_{v\to 0}] \vee dp[t', f_{v\to 1}], & \text{if } f(u) = 1 \\ dp[t', f_{v\to 1}], & \text{otherwise.} \end{cases}$$

$$dp[t, f_{v\to 0}] = dp[t', f_{v\to 0}]$$

Before



After

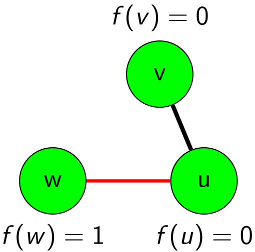


Introduce edge $(u, v) \notin F$, $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center ($f(u) = 1$):

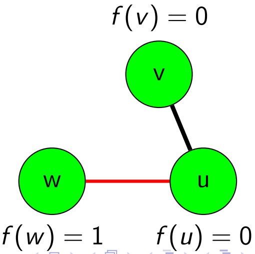
$$dp[t, f_{v\to 1}] = \begin{cases} dp[t', f_{v\to 0}] \vee dp[t', f_{v\to 1}], & \text{if } f(u) = 1 \\ dp[t', f_{v\to 1}], & \text{otherwise.} \end{cases}$$

$$dp[t, f_{v\to 0}] = dp[t', f_{v\to 0}]$$

Before



After

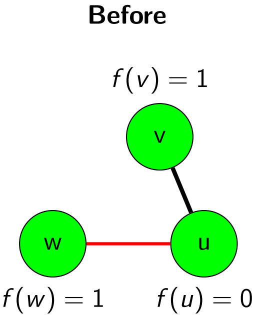


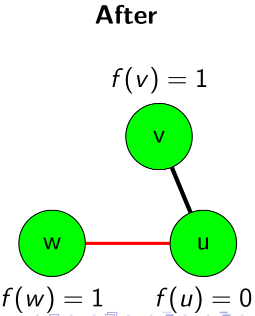
SSFE parameterized by $tw(G)$

Introduce edge $(u, v) \notin F$**,** $v \notin V(F)$
We change the value of vertex v from 0 to 1 if and only if u is a center ($f(u) = 1$):

$$dp[t, f_{v\to 1}] = \begin{cases} dp[t', f_{v\to 0}] \vee dp[t', f_{v\to 1}], & \text{if } f(u) = 1 \\ dp[t', f_{v\to 1}], & \text{otherwise.} \end{cases}$$

$$dp[t, f_{v\to 0}] = dp[t', f_{v\to 0}]$$



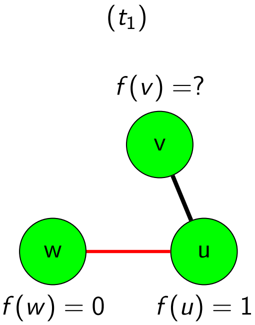


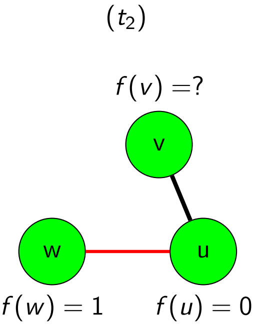
SSFE parameterized by $tw(G)$

Join nodes t_1**,** t_2

SSFE parameterized by $tw(G)$

Join nodes t_1, t_2

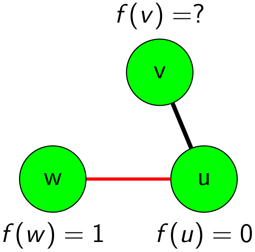




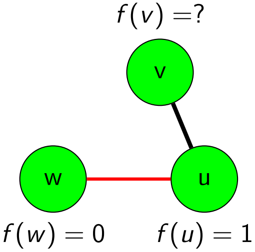
SSFE parameterized by $tw(G)$

Join nodes t_1, t_2

(t_1)



(t_2)

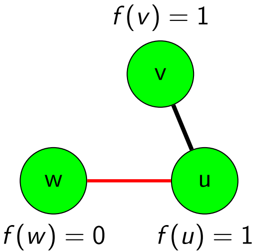


$$\forall v \in V(F) \cap X_t \ f(v) = f_1(v) = f_2(v)$$

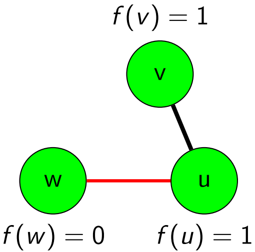
SSFE parameterized by $tw(G)$

Join nodes t_1, t_2

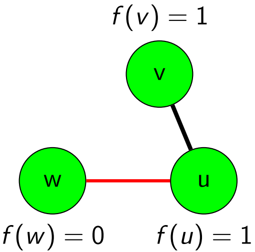
(t_1)



(t_2)

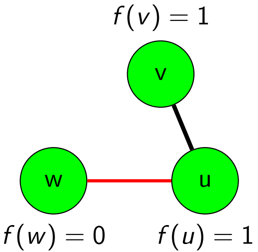


(t)

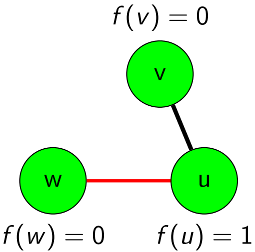


Join nodes t_1, t_2

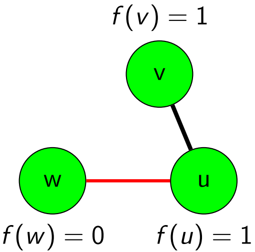
(t_1)



(t_2)



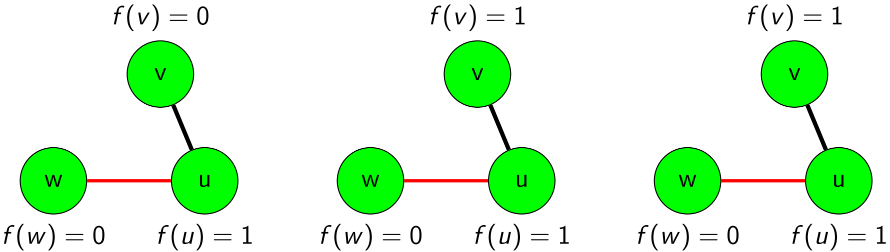
(t)



SSFE parameterized by $tw(G)$

Join nodes t_1**,** t_2

(t_1**)** **(**t_2**)** **(**t**)**



$$C_2 = \forall v ((V(G) \setminus V(F)) \cap X_t) \; f(v) = f_1(v) \vee f_2(v)$$

SSFE parameterized by $tw(G)$

SSFE parameterized by $tw(G)$

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

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 $2^{|X_t|}$ functions. Each can be calculated naively 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 $2^{|X_t|}$ functions. Each can be calculated naively in time $\mathcal{O}^*(2^{|X_t|})$. Applying cover product, we can calculate all of the functions f in time $\mathcal{O}^*(2^|X_t|)$

SSFE parameterized by $tw(G)$

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

There are $2^{|X_t|}$ functions. Each can be calculated naively in time $\mathcal{O}^*(2^{|X_t|})$. Applying cover product, we can calculate all of the functions f in time $\mathcal{O}^*(2^|X_t|)$

SSFE parameterized by $tw(G)$

There exists an algorithm solving SSFE parameterized by treewidth in time $\mathcal{O}^*(2^{tw(G)})$.

SSFE parameterized by $tw(G)$

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

There are $2^{|X_t|}$ functions. Each can be calculated naively in time $\mathcal{O}^*(2^{|X_t|})$. Applying cover product, we can calculate all of the functions f in time $\mathcal{O}^*(2^|X_t|)$

SSFE parameterized by $tw(G)$

There exists an algorithm solving SSFE parameterized by treewidth in time $\mathcal{O}^*(2^{tw(G)})$.

Lower bound

Assuming SETH, there does not exist an algorithm solving SSFE parameterized by $tw(G)$ in time $\mathcal{O}^*((2-\epsilon)^{tw(G)})$.