Chordal deletion is fixed-parameter tractable

Dániel Marx

Humboldt-Universität zu Berlin

`dmarx@informatik.hu-berlin.de`

June 22, 2006

WG 2006

Bergen, Norway

Graph modification problems

Problems of the following form:

> Given a graph G and an integer k, is it possible to add/delete k edges/vertices such that the result belongs to class $\mathcal{G}$?

- Make the graph bipartite by deleting k vertices.
- Make the graph chordal by adding k edges.
- Make the graph an empty graph by deleting k vertices (VERTEX COVER).
- . . .

Notation for graph classes

A notation introduced by Cai [2003]:

Definition: If $\mathcal{G}$ is a class of graphs, then we define the following classes of graphs:

- $\mathcal{G} + ke$: a graph from $\mathcal{G}$ with k extra edges.
- $\mathcal{G} - ke$: a graph from $\mathcal{G}$ with k edges deleted.
- $\mathcal{G} + kv$: graphs that can be made to be in $\mathcal{G}$ by deleting k vertices.
- $\mathcal{G} - kv$: a graph from $\mathcal{G}$ with k vertices deleted.

Notation for graph classes

A notation introduced by Cai [2003]:

Definition: If $\mathcal{G}$ is a class of graphs, then we define the following classes of graphs:

- $\mathcal{G} + ke$: a graph from $\mathcal{G}$ with k extra edges.
- $\mathcal{G} - ke$: a graph from $\mathcal{G}$ with k edges deleted.
- $\mathcal{G} + kv$: graphs that can be made to be in $\mathcal{G}$ by deleting k vertices.
- $\mathcal{G} - kv$: a graph from $\mathcal{G}$ with k vertices deleted.

Theorem: [Lewis and Yannakakis, 1980] If $\mathcal{G}$ is a nontrivial hereditary graph property, then it is NP-hard to decide if a graph is in $\mathcal{G} + kv$ (k is part of the input).

Parameterized complexity

As most problems are NP-hard, let us try to find efficient algorithms for small values of k. (Better than the $n^{O(k)}$ brute force algorithm.)

Definition: A problem is **fixed-parameter tractable (FPT)** with parameter k if it can be solved in time $f(k) \cdot n^{O(1)}$ for some function f.

Parameterized complexity

As most problems are NP-hard, let us try to find efficient algorithms for small values of k. (Better than the $n^{O(k)}$ brute force algorithm.)

Definition: A problem is **fixed-parameter tractable (FPT)** with parameter k if it can be solved in time $f(k) \cdot n^{O(1)}$ for some function f.

- **Theorem:** [Reed et al.] Recognizing bipartite$+kv$ graphs is FPT.
- **Theorem:** Recognizing empty$+kv$ graphs is FPT (VERTEX COVER).
- **Theorem:** [Cai; Kaplan et al.] Recognizing chordal$-ke$ is FPT.
- **Theorem:** [from Robertson and Seymour] if $\mathcal{G}$ is minor closed, then recognizing $\mathcal{G} + kv$ is FPT.
- **Theorem:** [Cai] If $\mathcal{G}$ is characterized by a finite set of forbidden induced subgraphs, then recognizing $\mathcal{G} + kv$ is FPT.

New result

- **Theorem:** [Reed et al.] Recognizing bipartite$+kv$ graphs is FPT.
- **Theorem:** Recognizing empty$+kv$ graphs is FPT (VERTEX COVER).
- **Theorem:** [Cai; Kaplan et al.] Recognizing chordal$-ke$ is FPT.
- **Theorem:** [from Robertson and Seymour] if $\mathcal{G}$ is minor closed, then recognizing $\mathcal{G} + kv$ is FPT.
- **Theorem:** [Cai] If $\mathcal{G}$ is characterized by a finite set of forbidden induced subgraphs, then recognizing $\mathcal{G} + kv$ is FPT.

New result: Recognizing chordal$+kv$ graphs is FPT.

New result

- **Theorem:** [Reed et al.] Recognizing bipartite$+kv$ graphs is FPT.
- **Theorem:** Recognizing empty$+kv$ graphs is FPT (VERTEX COVER).
- **Theorem:** [Cai; Kaplan et al.] Recognizing chordal$-ke$ is FPT.
- **Theorem:** [from Robertson and Seymour] if $\mathcal{G}$ is minor closed, then recognizing $\mathcal{G} + kv$ is FPT.
- **Theorem:** [Cai] If $\mathcal{G}$ is characterized by a finite set of forbidden induced subgraphs, then recognizing $\mathcal{G} + kv$ is FPT.

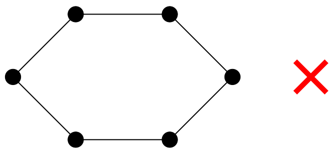
New result: Recognizing chordal$+kv$ graphs is FPT.

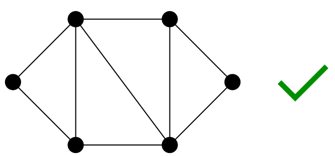
Remark: chordal graphs are not minor closed, and cannot be characterized by finitely many forbidden subgraphs.

Chordal graphs

A graph is **chordal** if it does not contain induced cycles longer than 3 (a “hole”).

- Interval graphs are chordal.
- Intersection graphs of subtrees in a tree $\Leftrightarrow$ chordal graphs.
- The maximum clique size is $k+1$ in a chordal graph $\Leftrightarrow$ the chordal graph has tree width k.
- Chordal graphs are perfect.





Chordal completion

Theorem: [Cai; Kaplan et al.] Recognizing chordal$-\boldsymbol{ke}$ is FPT.

Using the **bounded-height search tree** method.

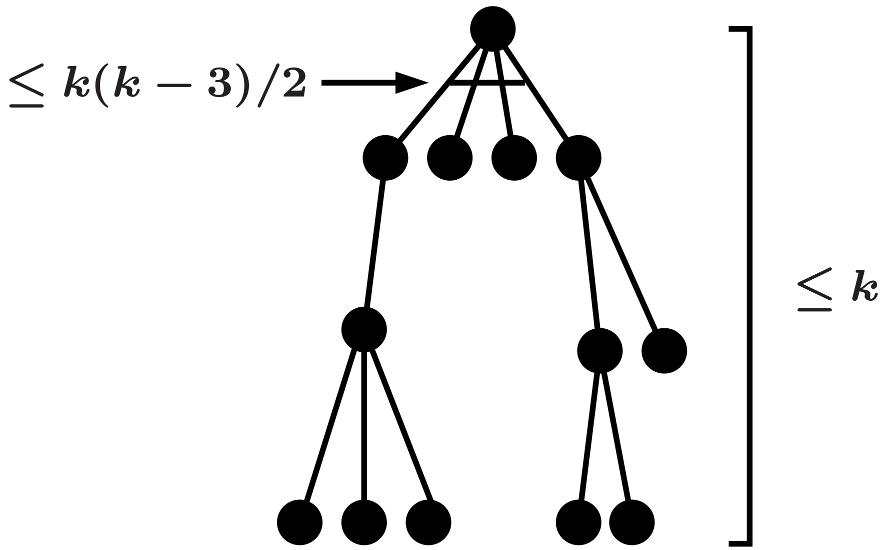
- If there is a hole of size greater than $\boldsymbol{k+3}$: cannot be made chordal with the addition of $\boldsymbol{k}$ edges.
- If there is a hole of size $\boldsymbol{\ell \le k+3}$: at least one chord has to be added. We branch into $\boldsymbol{\ell(\ell-3)/2}$ directions.

Chordal completion

Theorem: [Cai; Kaplan et al.] Recognizing chordal$-ke$ is FPT.

Using the **bounded-height search tree** method.

- If there is a hole of size greater than $k+3$: cannot be made chordal with the addition of k edges.
- If there is a hole of size $\ell \leq k+3$: at least one chord has to be added. We branch into $\ell(\ell-3)/2$ directions.



The size of the search tree can be bounded by a function of k.

$\Downarrow$

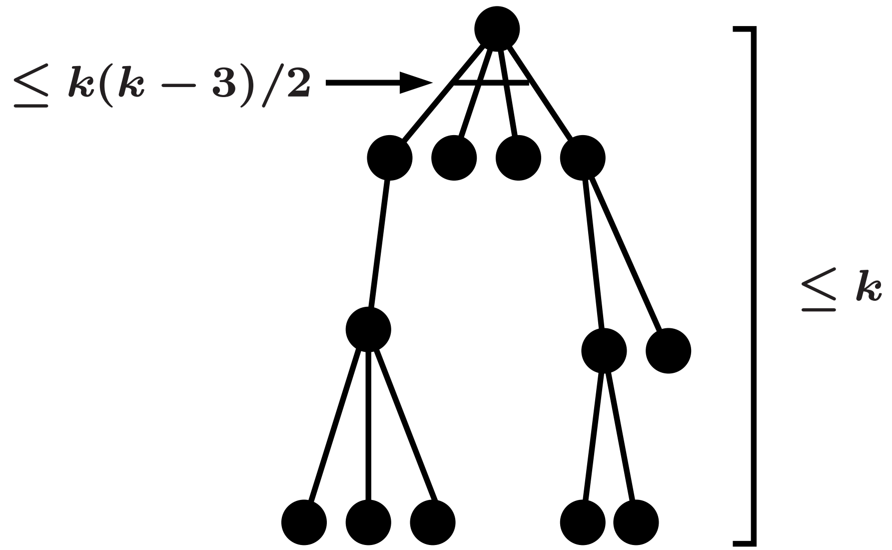
$f(k) \cdot n^{O(1)}$ algorithm

Chordal completion

Theorem: [Cai; Kaplan et al.] Recognizing chordal$-ke$ is FPT.

Using the **bounded-height search tree** method.

- If there is a hole of size greater than $k+3$: cannot be made chordal with the addition of k edges.
- If there is a hole of size $\ell \le k+3$: at least one chord has to be added. We branch into $\ell(\ell-3)/2$ directions.



The size of the search tree can be bounded by a function of k.

$\Downarrow$

$f(k) \cdot n^{O(1)}$ algorithm

For **chordal deletion** we cannot bound the size of the holes!

Techniques

New result: Recognizing chordal+kv graphs is FPT.

We use

- Iterative compression
- Bounded-height search trees
- Courcelle's Theorem for bounded tree width
- Tree width reduction

Iterative compression

Trick introduced by Reed et al. for recognizing bipartite$+kv$ graphs.

Instead of showing that this problem is FPT...

CHORDAL DELETION(G, k)	
Input:	A graph G, integer k
Find:	A set X of k vertices such that $G \setminus X$ is chordal

Iterative compression

Trick introduced by Reed et al. for recognizing bipartite$+kv$ graphs.

Instead of showing that this problem is FPT...

CHORDAL DELETION(G, k)	
Input:	A graph G, integer k
Find:	A set X of k vertices such that $G \setminus X$ is chordal

... we show that the easier "compression" problem is FPT:

CHORDAL COMPRESSION(G, k, Y)	
Input:	A graph G, integer k, **a set Y of $k+1$ vertices such that $G \setminus Y$ is chordal**
Find:	A set X of k vertices such that $G \setminus X$ is chordal

Iterative compression (cont.)

How to solve CHORDAL DELETION with CHORDAL COMPRESSION?

Let $v_1, \ldots, v_n$ be the vertices of G, and let G_i be the graph induced by the first i vertices.

1. Let $i := k$, $X := \{v_1, \ldots, v_k\}$.
2. **Invariant condition:** $|X| = k$, $G_i \setminus X$ is chordal
3. Let $i := i + 1$, $Y := X \cup \{v_i\}$
4. **Invariant condition:** $|Y| = k + 1$, $G_i \setminus Y$ is chordal
5. Call CHORDALCOMPRESSION(G_i, k, Y)
   - If it returns no, then reject.
   - Otherwise let X be the set returned.
6. Go to Step 2.

Small tree width

Given: G and Y with $|Y| = k + 1$ and $G \setminus Y$ is chordal.

Two cases:

- Tree width of G is small ($\leq t_k$)
- Tree width of G is large ($> t_k$)

Small tree width

Given: G and Y with $|Y| = k + 1$ and $G \setminus Y$ is chordal.

Two cases:

- Tree width of G is small ($\leq t_k$)
- Tree width of G is large ($> t_k$)

If tree width is small, then we use

Courcelle's Theorem: If a graph property can be expressed in **Extended Monadic Second Order Logic (EMSO)**, then for every $w \geq 1$, there is a linear-time algorithm for testing this property in graphs having tree width w.

"$G \in$ chordal $+ kv$" can be expressed in EMSO

$$\Downarrow$$

If tree width $\leq t_k$, then the problem can be solved in linear time.

Small tree width

Extended Monadic Second Order Logic: usual logical connectives, vertex-vertex adjacency, edges-vertex incidence, quantification over vertex sets and edge sets.

$$\begin{aligned}
k\text{-chordal-deletion(V,E)} &:= \exists v_1, \ldots v_k \in V, V_0 \subseteq V : [\text{chordal}(V_0) \\
&\quad \wedge (\forall v \in V : v \in V_0 \vee v = v_1 \vee \cdots \vee v = v_k)] \\
\text{chordal}(V_0) &:= \neg(\exists x, y, z \in V_0, T \subseteq E : \text{adj}(x,y) \wedge \text{adj}(x,z) \wedge \\
&\quad \neg\text{adj}(y,z) \wedge \text{connected}(y, z, T, V_0)) \\
\text{connected}(y, z, T, V_0) &:= \forall Y, Z \subseteq V_0 : [(\text{partition}(V_0, Y, Z) \wedge y \in Y \wedge z \in Z) \\
&\quad \rightarrow (\exists y' \in Y, z' \in Z, e \in T : \text{inc}(e, y') \wedge \text{inc}(e, z'))] \\
\text{partition}(V_0, Y, Z) &:= \forall v \in V_0 : (v \in Y \vee v \in Z) \wedge (v \notin Y \vee v \notin Z)
\end{aligned}$$

Large tree width

If tree width of G is large $\Rightarrow$ tree width of $G \setminus Y$ is large $\Rightarrow$ $G \setminus Y$ has a large clique (since it is chordal)

We show that every large clique has a vertex whose deletion does not make the problem easier.

Definition: A vertex $v \in G$ is **irrelevant** if for every X such that $|X| = k$ and $(G \setminus v) \setminus X$ is chordal, it follows that $G \setminus X$ is also chordal.

Large tree width

If tree width of G is large $\Rightarrow$ tree width of $G \setminus Y$ is large $\Rightarrow$ $G \setminus Y$ has a large clique (since it is chordal)

We show that every large clique has a vertex whose deletion does not make the problem easier.

Definition: A vertex $v \in G$ is **irrelevant** if for every X such that $|X| = k$ and $(G \setminus v) \setminus X$ is chordal, it follows that $G \setminus X$ is also chordal.

Equivalent definition: A vertex v is **irrelevant** if whenever $|X| = k$ and $G \setminus X$ has a hole, then $G \setminus X$ has a hole that avoids v.

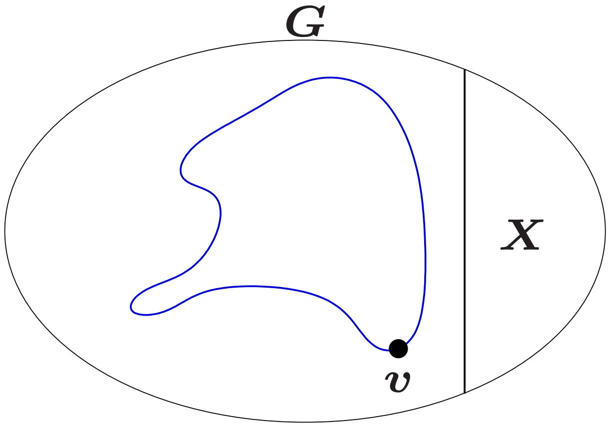
Large tree width

If tree width of G is large $\Rightarrow$ tree width of $G \setminus Y$ is large $\Rightarrow$ $G \setminus Y$ has a large clique (since it is chordal)

We show that every large clique has a vertex whose deletion does not make the problem easier.

Definition: A vertex $v \in G$ is **irrelevant** if for every X such that $|X| = k$ and $(G \setminus v) \setminus X$ is chordal, it follows that $G \setminus X$ is also chordal.

Equivalent definition: A vertex v is **irrelevant** if whenever $|X| = k$ and $G \setminus X$ has a hole, then $G \setminus X$ has a hole that avoids v.



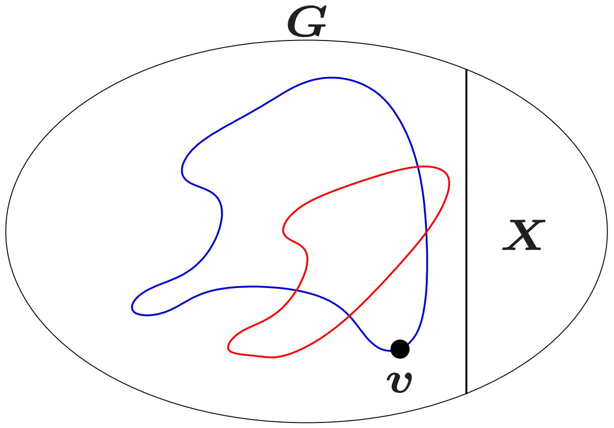
Large tree width

If tree width of G is large $\Rightarrow$ tree width of $G \setminus Y$ is large $\Rightarrow$ $G \setminus Y$ has a large clique (since it is chordal)

We show that every large clique has a vertex whose deletion does not make the problem easier.

Definition: A vertex $v \in G$ is **irrelevant** if for every X such that $|X| = k$ and $(G \setminus v) \setminus X$ is chordal, it follows that $G \setminus X$ is also chordal.

Equivalent definition: A vertex v is **irrelevant** if whenever $|X| = k$ and $G \setminus X$ has a hole, then $G \setminus X$ has a hole that avoids v.



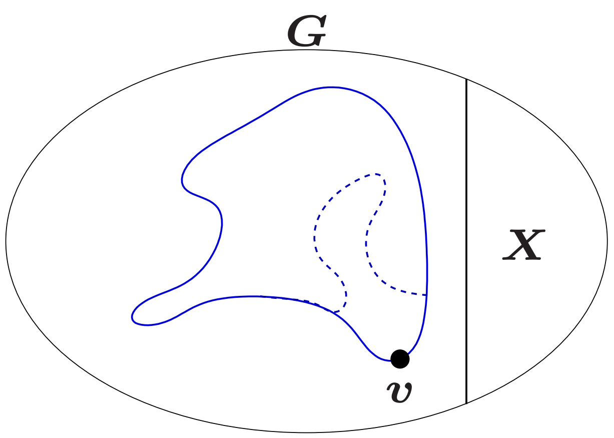
Large tree width

If tree width of G is large $\Rightarrow$ tree width of $G \setminus Y$ is large $\Rightarrow$ $G \setminus Y$ has a large clique (since it is chordal)

We show that every large clique has a vertex whose deletion does not make the problem easier.

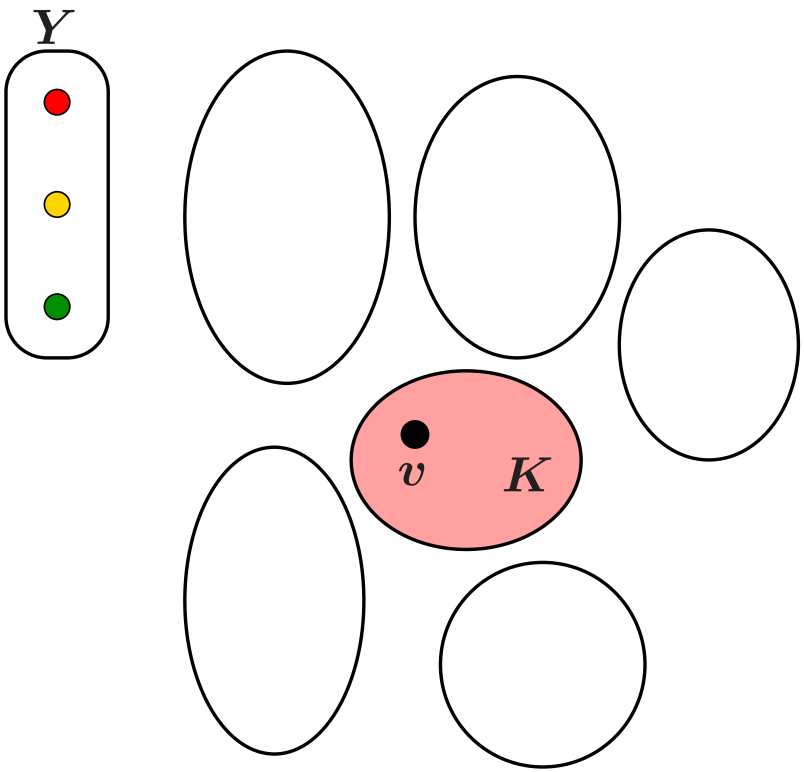
Definition: A vertex $v \in G$ is **irrelevant** if for every X such that $|X| = k$ and $(G \setminus v) \setminus X$ is chordal, it follows that $G \setminus X$ is also chordal.

Equivalent definition: A vertex v is **irrelevant** if whenever $|X| = k$ and $G \setminus X$ has a hole, then $G \setminus X$ has a hole that avoids v.



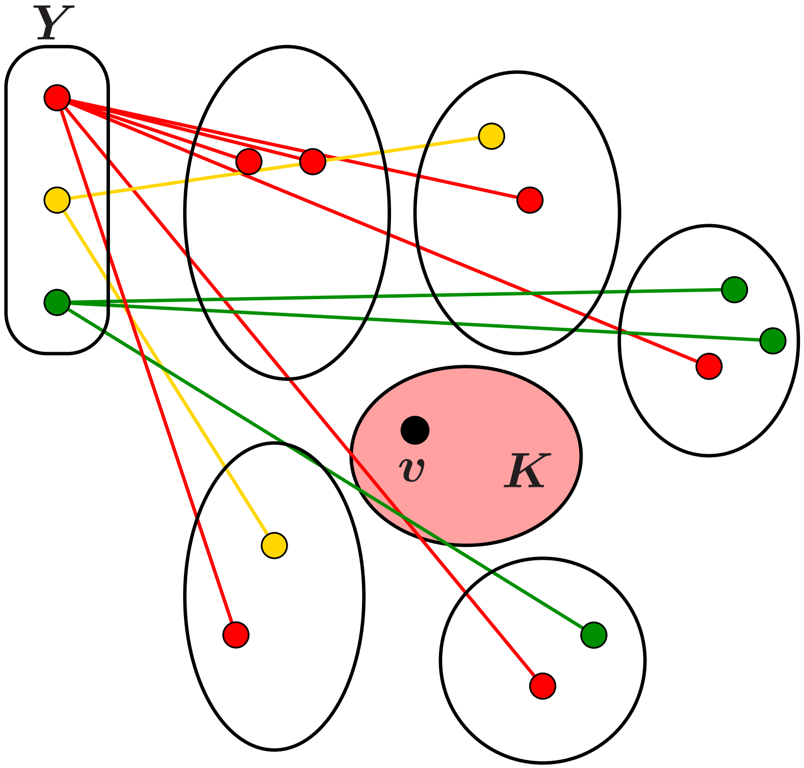
How to find an irrelevant vertex?

Consider $G \setminus X$.



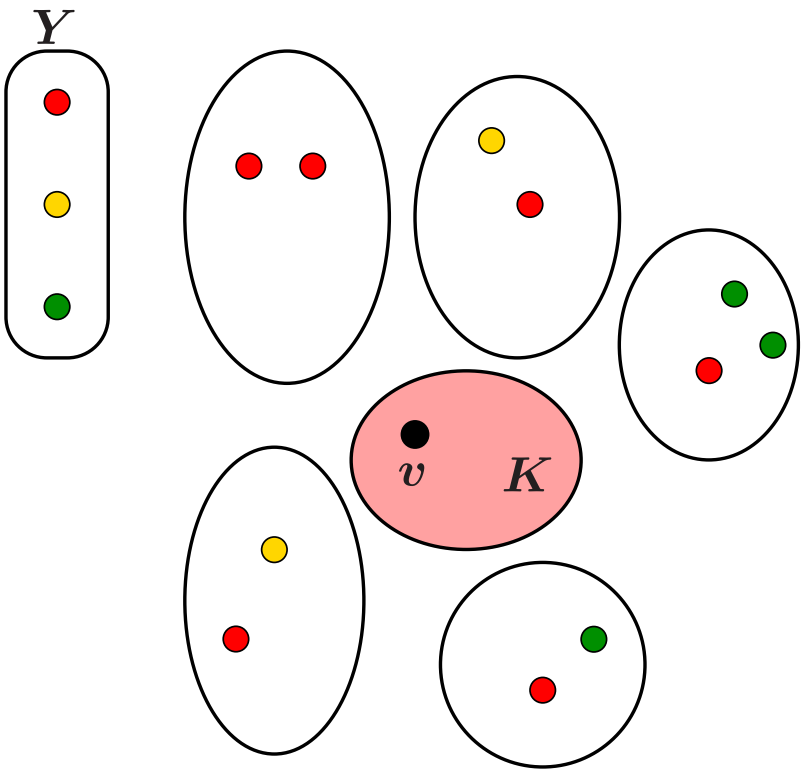
How to find an irrelevant vertex?

Consider $G \setminus X$.



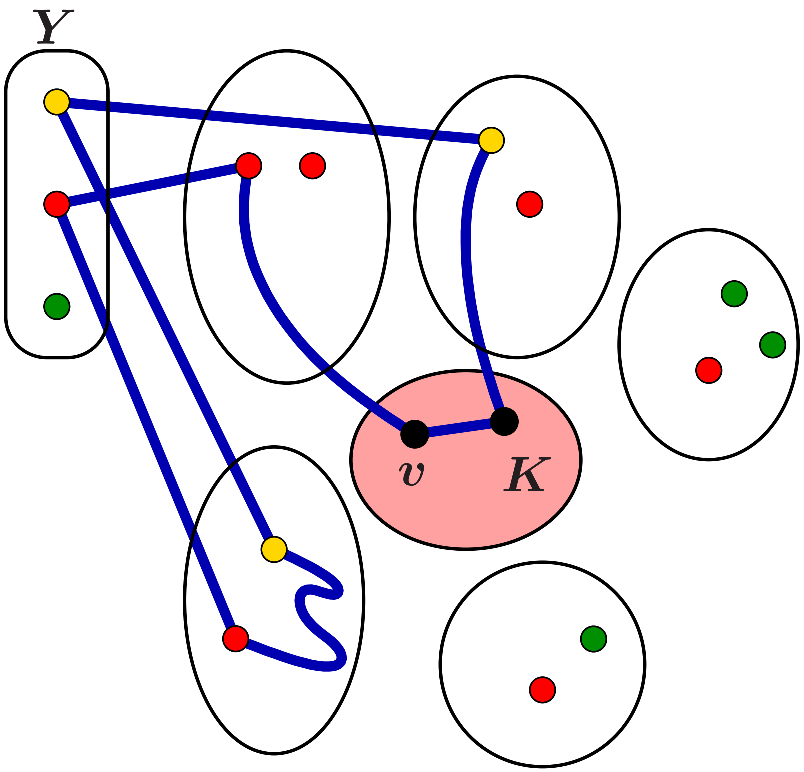
How to find an irrelevant vertex?

Consider $G \setminus X$.



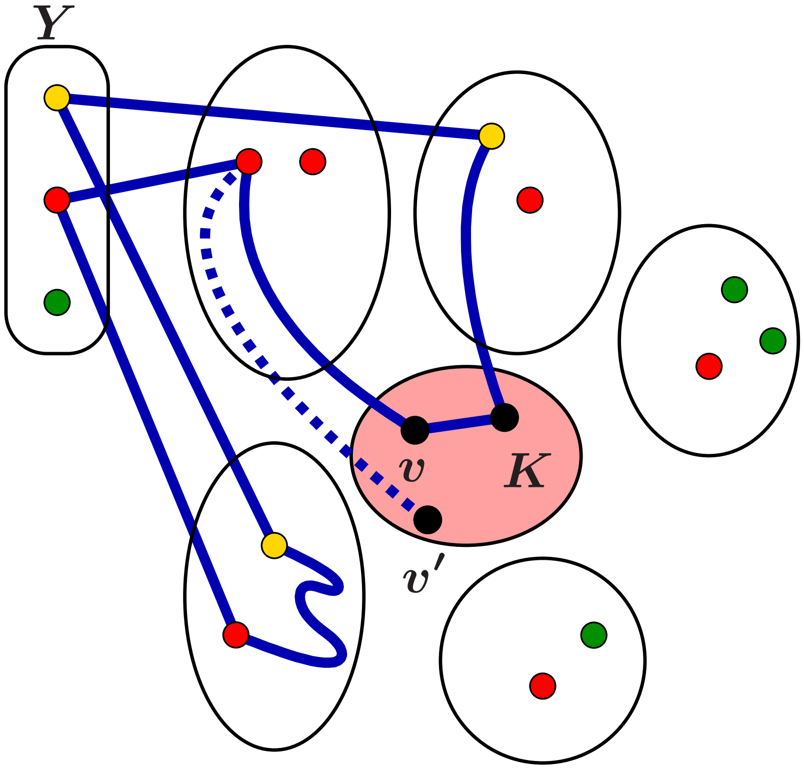
How to find an irrelevant vertex?

Consider $G \setminus X$. Assume that there is a hole going through v.



How to find an irrelevant vertex?

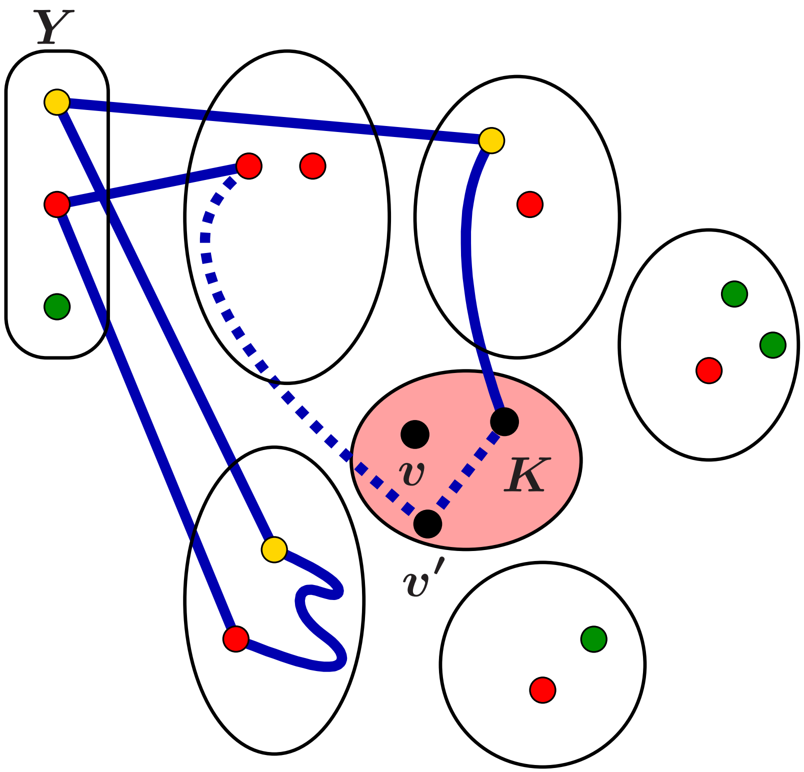
Consider $G \setminus X$. Assume that there is a hole going through v.



To bypass v, we need a $v' \in K$ that can be connected to a neighbor of ● with a path that does not go through a neighbor of ●.

How to find an irrelevant vertex?

Consider $G \setminus X$. Assume that there is a hole going through v.



To bypass v, we need a $v' \in K$ that can be connected to a neighbor of ● with a path that does not go through a neighbor of ●.

Marking vertices

We mark t_k vertices of K such that if there is a “bypass path” in $G \setminus X$, then there is such a path that ends in a marked vertex of K.

$$\Downarrow$$

Any non-marked vertex is irrelevant.

Marking vertices

We mark t_k vertices of K such that if there is a "bypass path" in $G \setminus X$, then there is such a path that ends in a marked vertex of K.

$$\Downarrow$$

Any non-marked vertex is irrelevant.

Dangerous vertex: A neighbor of ●, such that it can be connected to K with a path going through no other neighbor of ●.

- For each dangerous vertex, we mark $k+1$ vertices of the clique such that if K can be reached, then it can be reached at a marked vertex.
- We can do this even for a clique of dangerous vertices.
- The dangerous vertices can be covered by c_k cliques.

Overview

Overview of the algorithm:

- Iterative compression: we can assume that there is a solution of size $k+1$.
- Bounded search tree method.
- Courcelle's Theorem if tree width is small.
- If tree width is large, then an irrelevant vertex can be found.

Conclusions

- Another graph modification problem proved to be FPT.
- General techniques?
- Iterative compression.
- Edge deletion version.
- Interval deletion?