A Complexity Dichotomy for Finding Disjoint Solutions of Vertex Deletion Problems

Michael R. Fellows[1], Jiong Guo[2],
Hannes Moser[2], and Rolf Niedermeier[2]

[1] University of Newcastle, Australia

[2] Friedrich-Schiller-Universität Jena, Germany

MFCS 2009

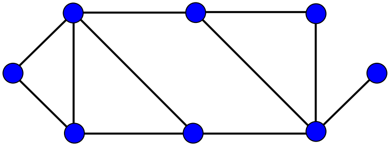
Starting Example

Vertex Bipartization

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that $G[V \setminus X]$ contains no odd-length cycle?

Example



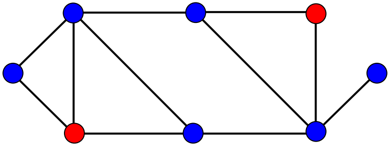
Starting Example

Vertex Bipartization

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that $G[V \setminus X]$ contains no odd-length cycle?

Example



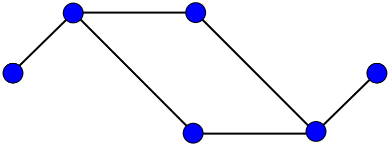
Starting Example

Vertex Bipartization

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that $G[V \setminus X]$ contains no odd-length cycle?

Example



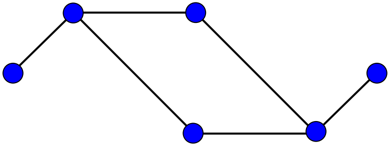
Starting Example

Vertex Bipartization

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that $G[V \setminus X]$ contains no odd-length cycle?

Example



Theorem: Vertex Bipartization is NP-complete

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]

Fixed-Parameter Tractability

Fixed-Parameter Tractability

A parameterized problem with input instance (G, k) is *fixed-parameter tractable* with respect to parameter k if it can be solved in $f(k) \cdot \text{poly}(|G|)$ time.

Fixed-Parameter Tractability

Fixed-Parameter Tractability

A parameterized problem with input instance (G, k) is *fixed-parameter tractable* with respect to parameter k if it can be solved in $f(k) \cdot \text{poly}(|G|)$ time.

Theorem

Vertex Bipartization can be solved in $O(3^k \cdot mn)$ time.

[Reed, Smith, and Vetta, Operations Research Letters 32, 2004]

[Hüffner, Journal of Graph Algorithms and Applications 13, 2009]

[Lokshtanov, Saurabh, and Sikdar, IWOCA 2009]

Fixed-Parameter Tractability

Fixed-Parameter Tractability

A parameterized problem with input instance (G, k) is *fixed-parameter tractable* with respect to parameter k if it can be solved in $f(k) \cdot \text{poly}(|G|)$ time.

Theorem

Vertex Bipartization can be solved in $O(3^k \cdot mn)$ time.

[Reed, Smith, and Vetta, Operations Research Letters 32, 2004]

[Hüffner, Journal of Graph Algorithms and Applications 13, 2009]

[Lokshtanov, Saurabh, and Sikdar, IWOCA 2009]

Technique: "Iterative Compression"

Some Applications of Iterative Compression

- Undirected Feedback Vertex Set in $O(5^k \cdot kn^2)$ time
  [Chen et al., Journal of Computer and System Sciences 74, 2008]
- Directed Feedback Vertex Set in $O(k! \cdot 4^k \cdot k^3 n^4)$ time
  [Chen et al., Journal of the ACM 55, 2008]
- Directed Feedback Vertex Set in Tournaments in $O(2^k \cdot n^2(\log\log n + k))$ time [Dom et al., CIAC 2006]
- Signed Graph Balancing in $O(2^k \cdot m^2)$ time
  [Hüffner, Betzler, and Niedermeier, Journal of Combinatorial Optimization, 2009]
- Chordal Deletion is FPT [Marx, Algorithmica, 2009]
- Cluster Vertex Deletion in $O(2^k \cdot k^6 \log k + nm)$ time
  [Hüffner et al., Theory of Computing Systems, 2009]
- Almost 2-Sat in $O(15^k \cdot km^3)$ time
  [Razgon and O'Sullivan, Journal of Computer and System Sciences, 2009]

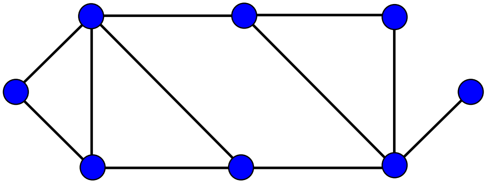
Iterative Compression: Example

Vertex Cover

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that each edge has a least one endpoint in X?

Example



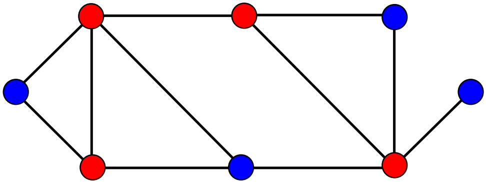
Iterative Compression: Example

Vertex Cover

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that each edge has a least one endpoint in X?

Example



Iterative Compression Framework

Idea

Use a *compression routine* iteratively: Given a solution of size $k+1$ for a graph G, compute a solution of smallest size for G.

[Reed, Smith, and Vetta, Operations Research Letters 32, 2004]

Iterative Compression Framework

Idea

Use a *compression routine* iteratively: Given a solution of size $k + 1$ for a graph G, compute a solution of smallest size for G.

[Reed, Smith, and Vetta, Operations Research Letters 32, 2004]

Iterative Compression Framework

1. Start with an empty vertex set V' and an empty solution X.
2. For each vertex v in $V(G)$:
   2.1 Add v to V' and to X.
   2.2 Use the compression routine to compress the solution X for $G[V']$.
   2.3 If $|X| > k$, then return "no-instance".

Iterative Compression Framework

Idea

Use a *compression routine* iteratively: Given a solution of size $k+1$ for a graph G, compute a solution of smallest size for G.

[Reed, Smith, and Vetta, Operations Research Letters 32, 2004]

Iterative Compression Framework

1. Start with an empty vertex set V' and an empty solution X.
2. For each vertex v in $V(G)$:
   - 2.1 Add v to V' and to X.
   - 2.2 Use the compression routine to compress the solution X for $G[V']$.
   - 2.3 If $|X| > k$, then return "no-instance".

Invariant during the loop for a yes-instance:
After the compression, X is a solution of size at most k for the current graph $G[V']$.

Compression Routine (1)

Task

Given a solution X of size $k+1$, compute a smallest solution X'.

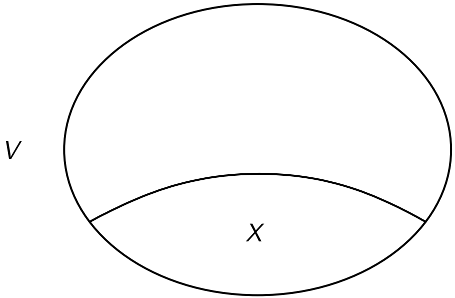
Compression Routine (1)

Task

Given a solution X of size $k+1$, compute a smallest solution X'.

Approach

Try all $2^{|X|}$ partitions of X into a part to exchange (S) and a part to keep ($X \setminus S$).



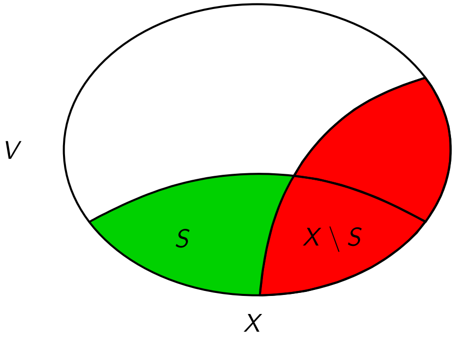
Compression Routine (1)

Task

Given a solution X of size $k + 1$, compute a smallest solution X'.

Approach

Try all $2^{|X|}$ partitions of X into a part to exchange (S) and a part to keep ($X \setminus S$).



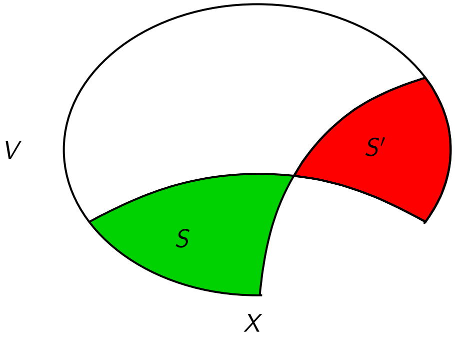
Compression Routine (1)

Task

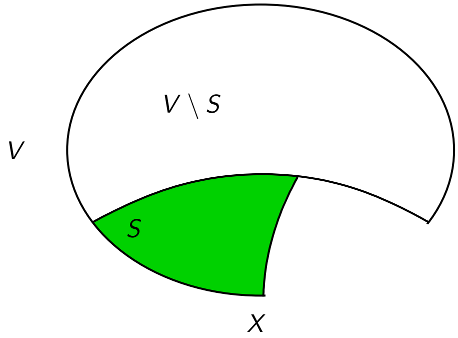
Given a solution X of size $k + 1$, compute a smallest solution X'.

Approach

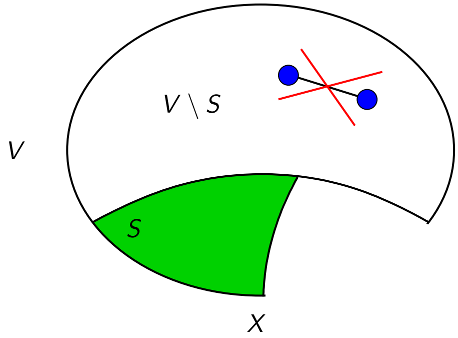
Try all $2^{|X|}$ partitions of X into a part to exchange (S) and a part to keep ($X \setminus S$).



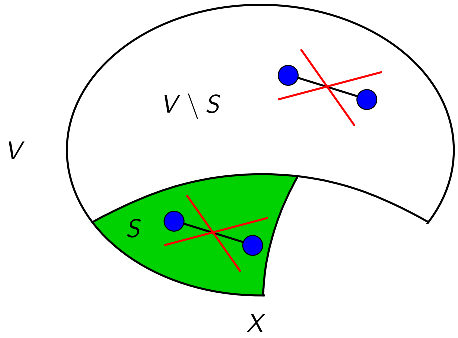
Compression Routine (2)



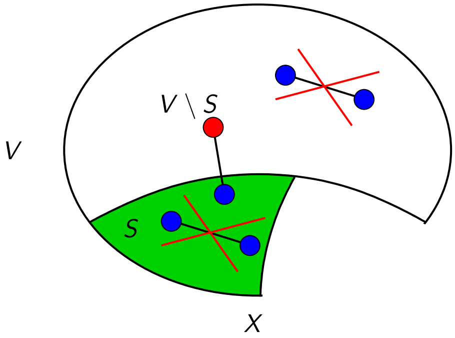
Compression Routine (2)



Compression Routine (2)



Compression Routine (2)



Compression for Vertex Cover

If $|N(S)| < |S|$, then return $N(S)$, else return "no".

Running time: $O(n + m)$

Overall Running Time

Time Consumption

- n iterations of the compression routine;
- 2^{k+1} partitions per iteration;
- time $O(n + m)$ for each compression.

Running time for solving Vertex Cover

$$O(2^k \cdot n(n + m))$$

Compression Routines for Known Applications

- Vertex Bipartization
  brute force: $O(2^k \cdot km)$ time
- Undirected Feedback Vertex Set
  bounded search tree: $O(4^k \cdot n^2)$ time
- Cluster Vertex Deletion
  reduction to the polynomial-time solvable Weighted Maximum Matching problem: $O(m\sqrt{n} \log n)$ time
- Directed Feedback Vertex Set
  brute force combined with bounded search tree: $O(k!4^k \cdot k^2 n^3)$ time
- Directed Feedback Vertex Set in tournaments
  reduction to the polynomial-time solvable Longest Common Subsequence problem: $O(n(\log \log n + k))$ time

Vertex Deletion Problems

A graph property Π is a (possibly infinite) set of graphs.

Π Vertex Deletion

Input: An undirected graph $G = (V, E)$ and a parameter $k \geq 0$.

Question: Can we find a vertex set $X \subseteq V$, $|X| \leq k$, such that $G[V \setminus X] \in \Pi$?

Examples

problem	graph property Π
Vertex Cover	set of all graphs without edges
Feedback Vertex Set	set of all forests (acyclic graphs)
Vertex Bipartization	set of all bipartite graphs
Cluster Vertex Deletion	set of all cluster graphs[1]

[1] Graphs whose connected components are cliques.

Compression Task for Vertex Deletion Problems

Disjoint Π Vertex Deletion

Input: An undirected graph $G = (V, E)$ and a vertex subset $S \subseteq V$ such that $G[V \setminus S] \in \Pi$.

Question: Can we find a vertex set $S' \subseteq V$, such that $S' \cap S = \emptyset$, $|S'| < |S|$ and $G[V \setminus S'] \in \Pi$?

Compression Task for Vertex Deletion Problems

Disjoint Π Vertex Deletion

Input: An undirected graph $G = (V, E)$ and a vertex subset $S \subseteq V$ such that $G[V \setminus S] \in \Pi$.

Question: Can we find a vertex set $S' \subseteq V$, such that $S' \cap S = \emptyset$, $|S'| < |S|$ and $G[V \setminus S'] \in \Pi$?

We restrict our attention to graph properties such that

- for all $V' \subseteq V$, $G \in \Pi \Rightarrow G[V'] \in \Pi$ (Π is *hereditary*)
- If every connected component of a graph is in Π, then the graph is in Π (Π is *determined by the components*)

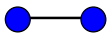
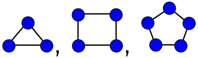
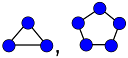
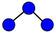
(All the problems on undirected graphs mentioned before are based on such properties.)

Forbidden Induced Subgraph Characterization

For any hereditary graph property Π there exists a set $\mathcal{H}_\Pi$ of inclusion-minimal forbidden induced subgraphs for Π.

[Greenwell, Hemminger, and Klerlein, 1973]

Examples

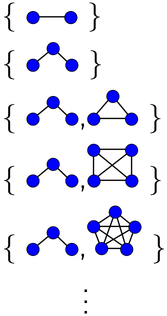
property Π	$\mathcal{H}_\Pi$
set of all graphs without edges	{  }
set of all forests (acyclic graphs)	{ , ...}
set of all bipartite graphs	{ , ...}
set of all cluster graphs	{  }

Complexity Dichotomy for the Compression Task

Main Theorem

Disjoint Π Vertex Deletion is NP-hard unless the set $\mathcal{H}_\Pi$ of forbidden induced subgraphs corresponding to Π contains a single edge or a path of length two—in these cases it is polynomial-time solvable.

P



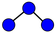
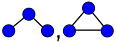
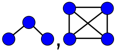
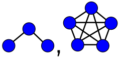
NP-hard

all other graph properties

Example Problems

P	Vertex Cover Cluster Vertex Deletion
NP-hard	Vertex Bipartization Undirected Feedback Vertex Set Chordal Deletion Planar Deletion Bounded-Degree Deletion

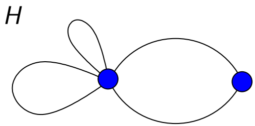
Polynomial-Time Solvable Cases

$\mathcal{H}_\Pi$	compression algorithm
{  }	trivial
{  }	reduction to Maximum Matching [Hüffner et al., Theory of Computing Systems, 2009]
{  }	reduction to Maximum Matching + size limit 2
{  }	reduction to Maximum Matching + size limit 3
{  }	reduction to Maximum Matching + size limit 4
	⋮

NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

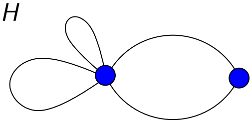
[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]

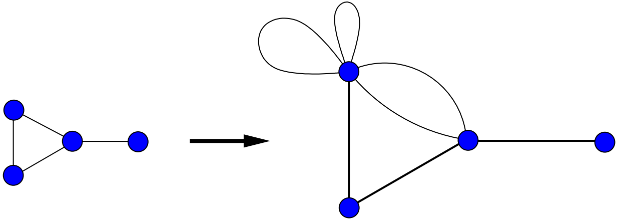


NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]





Vertex Cover instance

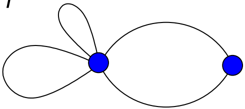
Disjoint Π Vertex Deletion instance

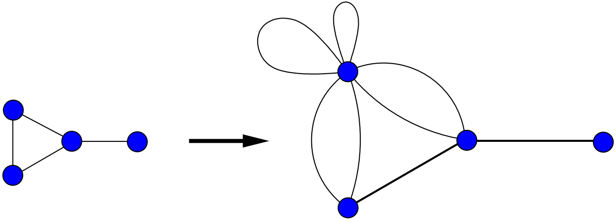
NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]

H



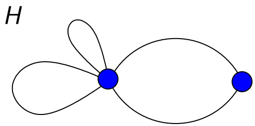


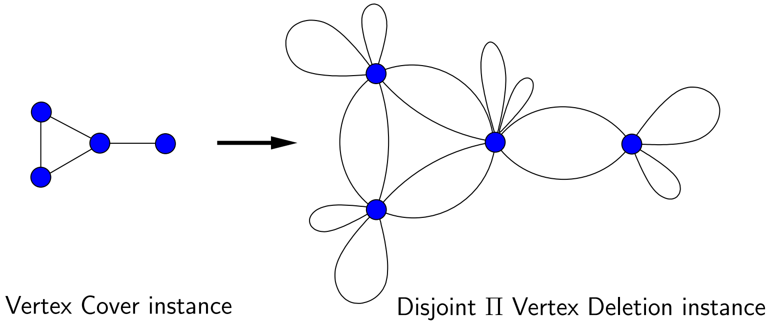
Vertex Cover instance Disjoint Π Vertex Deletion instance

NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]

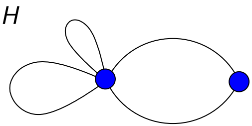


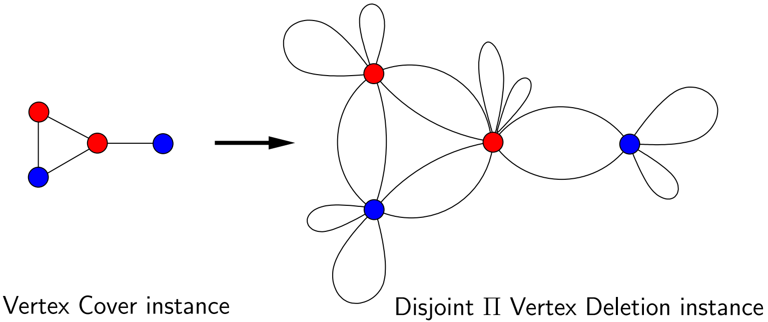


NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]

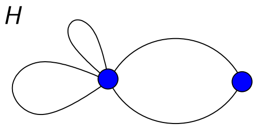


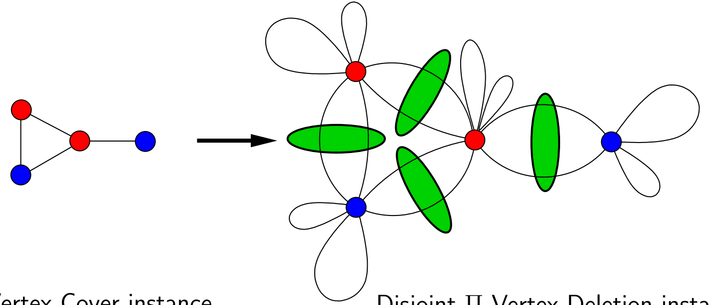


NP-Hardness: Basic Reduction Scheme

Let H be a "properly chosen" graph from $\mathcal{H}_\Pi$.

[Lewis and Yannakakis, Journal of Computer and System Sciences 20, 1980]



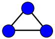


Vertex Cover instance Disjoint Π Vertex Deletion instance

Observation

For the reduction scheme to work, the old solution S must not contain vertices corresponding to vertices in the Vertex Cover instance, but it has to obstruct all forbidden induced subgraphs in $\mathcal{H}_\Pi$.

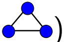
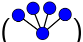
Cases

- All graphs in $\mathcal{H}_\Pi$ contain a K_3 ().
  $\Rightarrow$ reduce from Vertex Cover on K_3-free graphs

Observation

For the reduction scheme to work, the old solution S must not contain vertices corresponding to vertices in the Vertex Cover instance, but it has to obstruct all forbidden induced subgraphs in $\mathcal{H}_\Pi$.

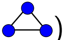
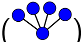
Cases

- All graphs in $\mathcal{H}_\Pi$ contain a K_3 ().
  $\Rightarrow$ reduce from Vertex Cover on K_3-free graphs
- $\mathcal{H}_\Pi$ contains no stars ().
  $\Rightarrow$ reduce from Vertex Cover on K_3-free graphs, where H is a "properly chosen" graph among all K_3-free graphs in $\mathcal{H}_\Pi$

Observation

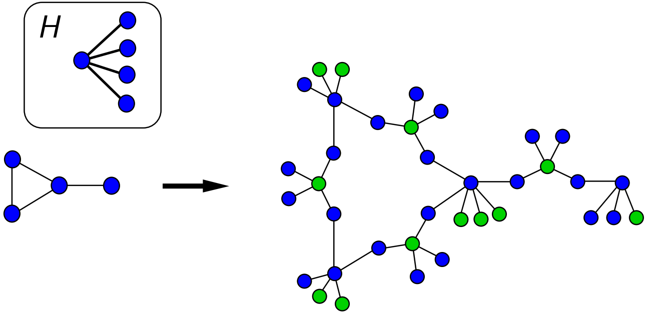
For the reduction scheme to work, the old solution S must not contain vertices corresponding to vertices in the Vertex Cover instance, but it has to obstruct all forbidden induced subgraphs in $\mathcal{H}_\Pi$.

Cases

- All graphs in $\mathcal{H}_\Pi$ contain a K_3 ().
  $\Rightarrow$ reduce from Vertex Cover on K_3-free graphs
- $\mathcal{H}_\Pi$ contains no stars ().
  $\Rightarrow$ reduce from Vertex Cover on K_3-free graphs, where H is a "properly chosen" graph among all K_3-free graphs in $\mathcal{H}_\Pi$
- $\mathcal{H}_\Pi$ contains a star.
  Reduction scheme does not seem to work...

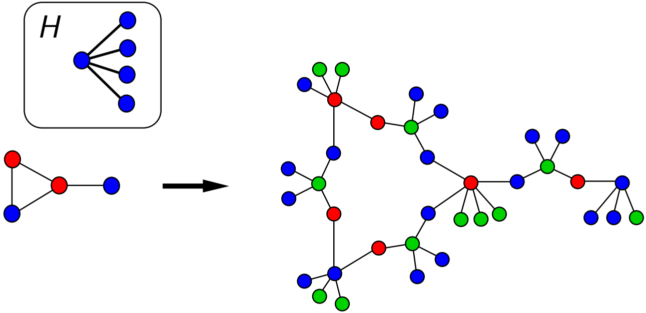
Reduction for Large Stars

$\mathcal{H}_\Pi$ contains a star with at least four leaves $\Rightarrow$ reduction from Vertex Cover on graphs of maximum degree three.



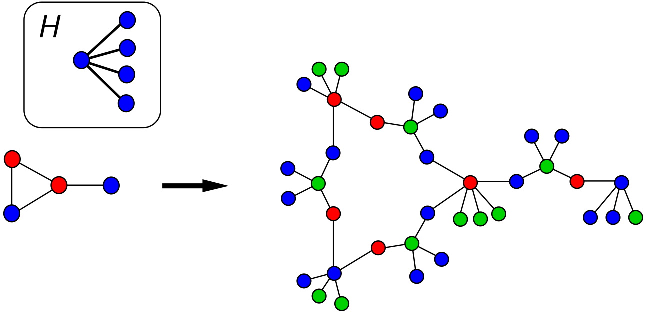
Reduction for Large Stars

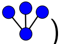
$\mathcal{H}_\Pi$ contains a star with at least four leaves $\Rightarrow$ reduction from Vertex Cover on graphs of maximum degree three.



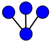
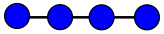
Reduction for Large Stars

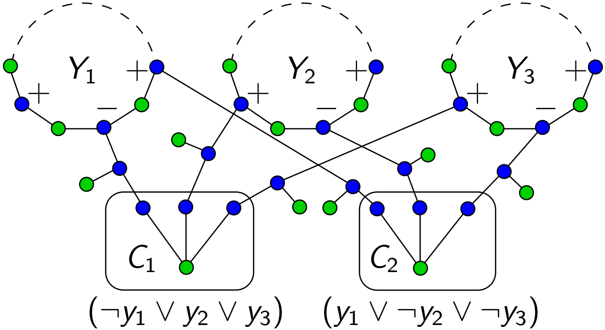
$\mathcal{H}_\Pi$ contains a star with at least four leaves $\Rightarrow$ reduction from Vertex Cover on graphs of maximum degree three.



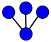
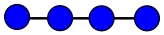
Does not work for stars with three leaves ()...

Reduction for Small Stars

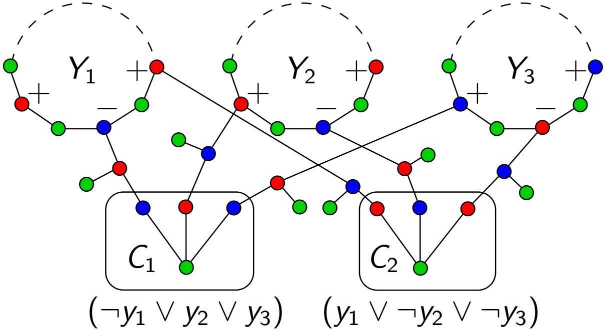
$\mathcal{H}_\Pi$ contains a  and a .
$\Rightarrow$ Reduction from 3-CNF-SAT.



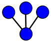
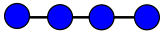
Reduction for Small Stars

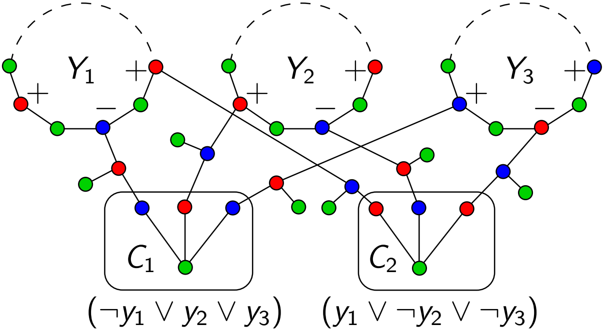
$\mathcal{H}_\Pi$ contains a  and a .

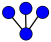
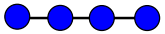
$\Rightarrow$ Reduction from 3-CNF-SAT.



Reduction for Small Stars

$\mathcal{H}_\Pi$ contains a  and a .
$\Rightarrow$ Reduction from 3-CNF-SAT.



$\mathcal{H}_\Pi$ contains a  and no .
$\Rightarrow$ Similar reduction from 3-CNF-SAT.

Outlook

- Study Disjoint Π Vertex Deletion in directed graphs.
- Study Edge Deletion Problems.
- Consider forbidden induced subgraphs with more than one connected component.
- Parameterize Disjoint Π Vertex Deletion by the number of vertices by which the new solution S' should at least differ from the old solution S.

Thank you!