Bart Jansen

Vertex Cover Kernelization Revisited:

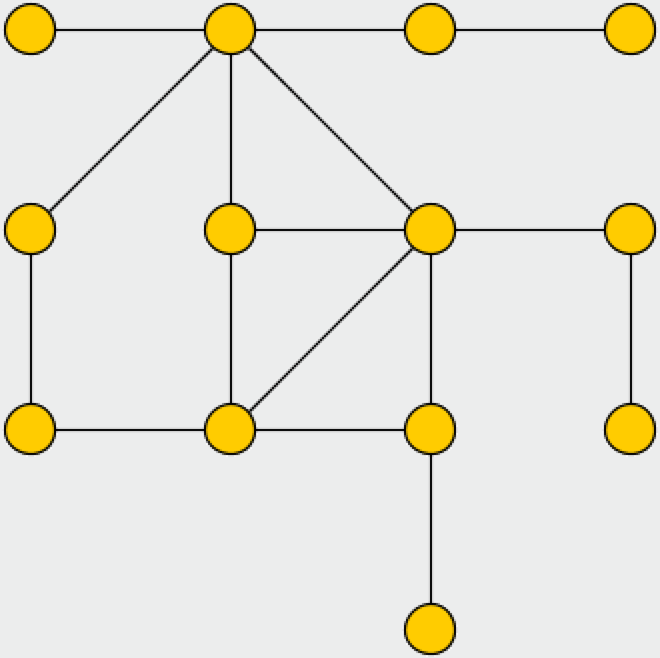
Upper and Lower Bounds for a Refined Parameter

Joint work with Hans Bodlaender

STACS 2011, Dortmund
March 10th, 2011

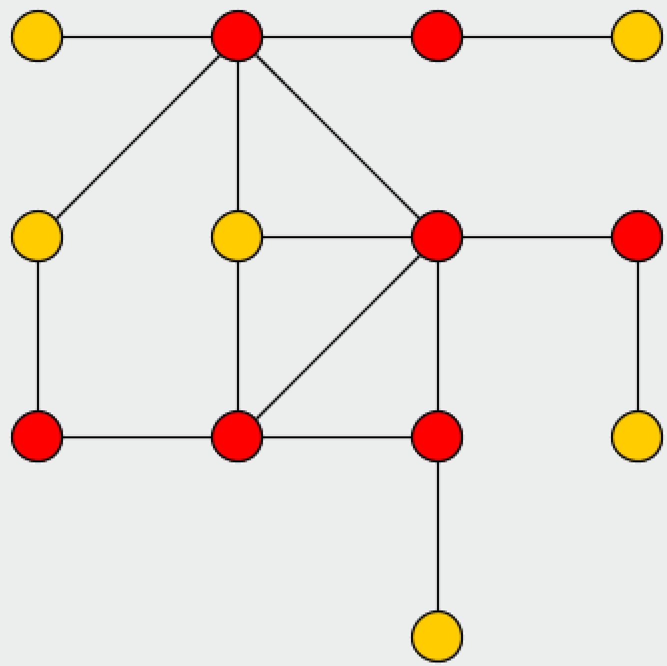
Vertex Cover

- Input: Graph G, integer k
- Question: Does G have a vertex cover of size ≤ k?



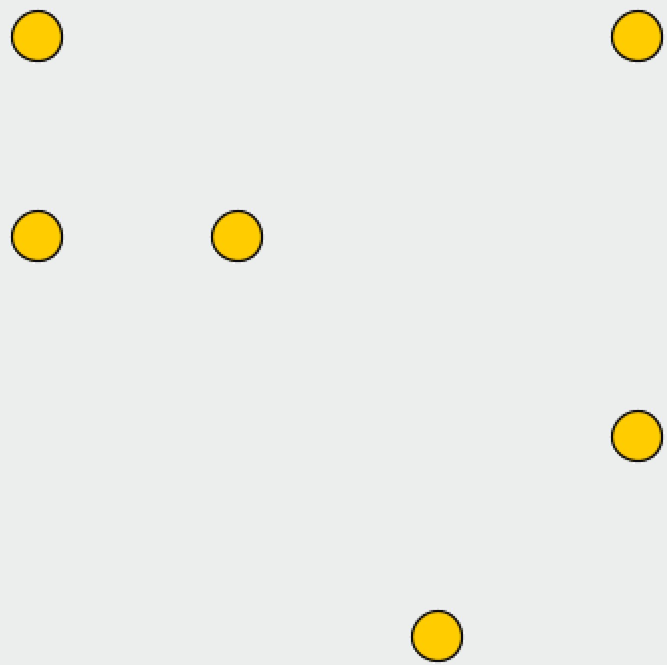
Vertex Cover

- Input: Graph G, integer k
- Question: Does G have a vertex cover of size ≤ k?



Vertex Cover

- Input: Graph G, integer k
- Question: Does G have a vertex cover of size ≤ k?



S is a Vertex Cover of G ⇔ Graph G – S is edgeless

Preprocessing for Vertex Cover

- Preprocess by computing a small, equivalent instance in polynomial time
- Reduce (G,k) to an equivalent instance on f(k) vertices

Preprocessing for Vertex Cover

- Preprocess by computing a small, equivalent instance in polynomial time
- Reduce (G,k) to an equivalent instance on f(k) vertices

$O(k^2)$

- Sam Buss [SIAM J. Comput. 1993]
- High-degree rule

Universiteit Utrecht

[Faculty of Science
Information and Computing Sciences]

Preprocessing for Vertex Cover

- Preprocess by computing a small, equivalent instance in polynomial time
- Reduce (G,k) to an equivalent instance on f(k) vertices

$O(k^2)$	• Sam Buss [SIAM J. Comput. 1993] • High-degree rule
2k	• Chen, Kanj and Jia [J. Algorithms 2001] • Linear-programming theorem by Nemhauser and Trotter

Preprocessing for Vertex Cover

- Preprocess by computing a small, equivalent instance in polynomial time
- Reduce (G,k) to an equivalent instance on f(k) vertices

$O(k^2)$	• Sam Buss [SIAM J. Comput. 1993] • High-degree rule
2k	• Chen, Kanj and Jia [J. Algorithms 2001] • Linear-programming theorem by Nemhauser and Trotter
3k	• Abu-Khzam, Fellows, Langston and Suters [Theory Comput. Syst. 2007] • Combinatorial algorithm by *Crown Reduction*

Preprocessing for Vertex Cover

- Preprocess by computing a small, equivalent instance in polynomial time
- Reduce (G,k) to an equivalent instance on f(k) vertices

$O(k^2)$	• Sam Buss [SIAM J. Comput. 1993] • High-degree rule
2k	• Chen, Kanj and Jia [J. Algorithms 2001] • Linear-programming theorem by Nemhauser and Trotter
3k	• Abu-Khzam, Fellows, Langston and Suters [Theory Comput. Syst. 2007] • Combinatorial algorithm by *Crown Reduction*

- Evidence that factor 2 is optimal under UGC

Relevant values of k

- Consider instance (G,k) of Vertex Cover
- Compute a 2-approximation set S in linear time
  - $VC(G) \leq |S| \leq 2\,VC(G)$

Relevant values of k

- Consider instance (G,k) of Vertex Cover
- Compute a 2-approximation set S in linear time
  - VC(G) ≤ |S| ≤ 2 VC(G)

k < |S|/2

- k < VC(G)
- Output **NO**

Relevant values of k

- Consider instance (G,k) of Vertex Cover
- Compute a 2-approximation set S in linear time
  - $VC(G) \leq |S| \leq 2\,VC(G)$

$k < \|S\|/2$	$k \geq \|S\|$
• $k < VC(G)$ • Output **NO**	• $k \geq VC(G)$ • Output **YES**

Relevant values of k

- Consider instance (G,k) of Vertex Cover
- Compute a 2-approximation set S in linear time
  - VC(G) ≤ |S| ≤ 2 VC(G)

k < |S|/2	k ≥ |S|
• k < VC(G) • Output **NO**	• k ≥ VC(G) • Output **YES**

- In interesting situations we have:
  - VC(G) / 2 ≤ k ≤ 2 VC(G)

Relevant values of k

- Consider instance (G,k) of Vertex Cover
- Compute a 2-approximation set S in linear time
  - VC(G) ≤ |S| ≤ 2 VC(G)

k < |S|/2	k ≥ |S|
• k < VC(G) • Output **NO**	• k ≥ VC(G) • Output **YES**

- In interesting situations we have:
  - VC(G) / 2 ≤ k ≤ 2 VC(G)
- So for relevant instances **k is Θ(VC(G))**

Alternative parameterizations

- Existing results guarantee that the size of instance (G,k) of Vertex Cover can be reduced to O(VC(G)) vertices
  - In polynomial time, without changing the answer

Alternative parameterizations

- Existing results guarantee that the size of instance (G,k) of Vertex Cover can be reduced to O(VC(G)) vertices
  - In polynomial time, without changing the answer
- VC(G) is just a measure of the complexity of a graph

Alternative parameterizations

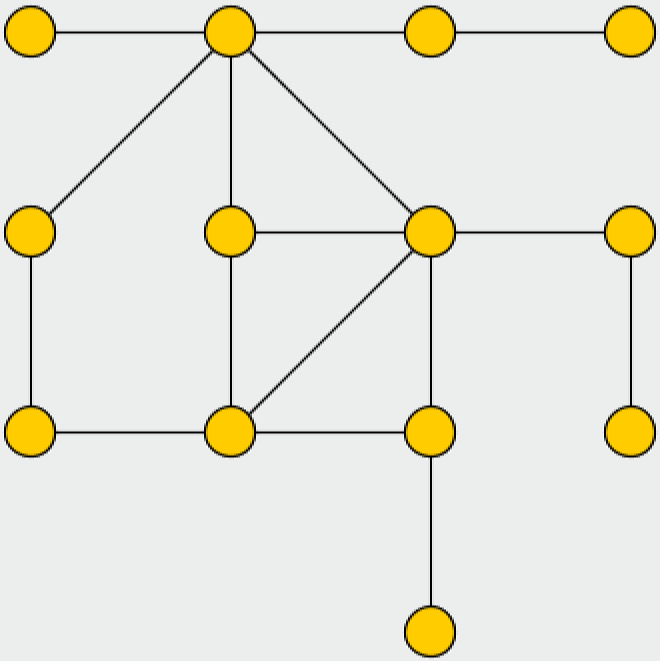
- Existing results guarantee that the size of instance (G,k) of Vertex Cover can be reduced to O(VC(G)) vertices
  - In polynomial time, without changing the answer
- VC(G) is just a measure of the complexity of a graph
- Take any measure Π which maps graphs to ***N***, and ask:
  - Can we reduce an instance (G,k) to poly[Π(G)] vertices?

Alternative parameterizations

- Existing results guarantee that the size of instance (G,k) of Vertex Cover can be reduced to O(VC(G)) vertices
  - In polynomial time, without changing the answer
- VC(G) is just a measure of the complexity of a graph
- Take any measure Π which maps graphs to $\boldsymbol{N}$, and ask:
  - Can we reduce an instance (G,k) to poly[Π(G)] vertices?
- Stronger data reduction if we can ensure $\Pi(G) \leq VC(G)$

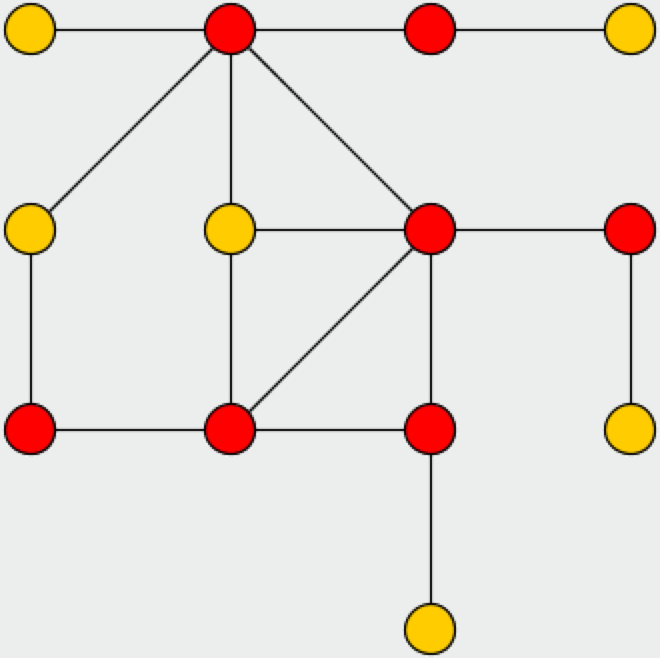
Graph parameters

- **Vertex Cover Number VC(G)**
- Size of a smallest set S such that G – S is edgeless



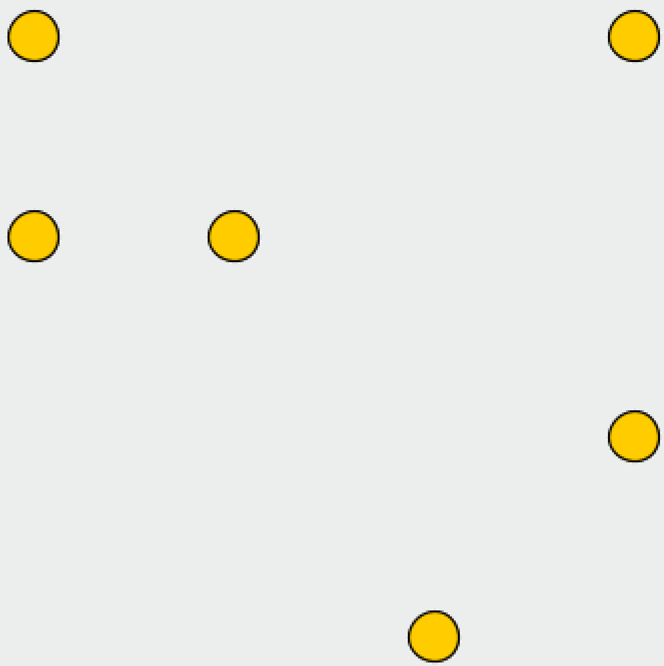
Graph parameters

- **Vertex Cover Number VC(G)**
- Size of a smallest set S such that G – S is edgeless



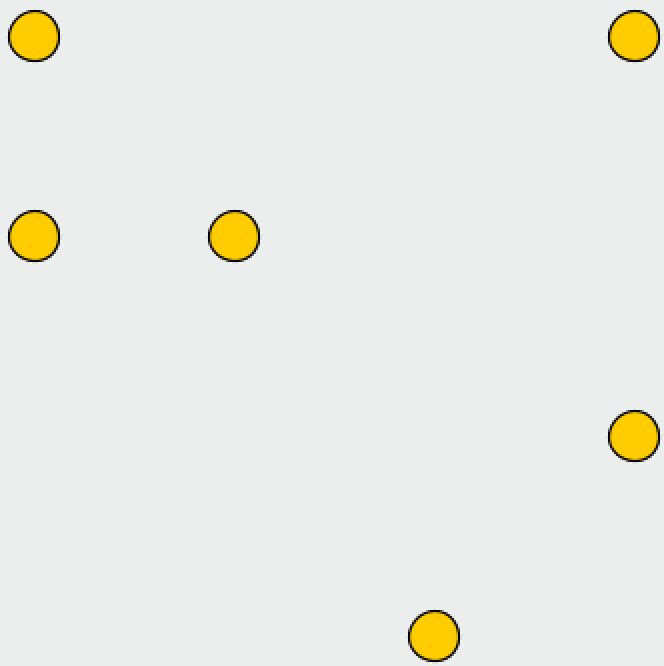
Graph parameters

- **Vertex Cover Number VC(G)**
- Size of a smallest set S such that G – S is edgeless

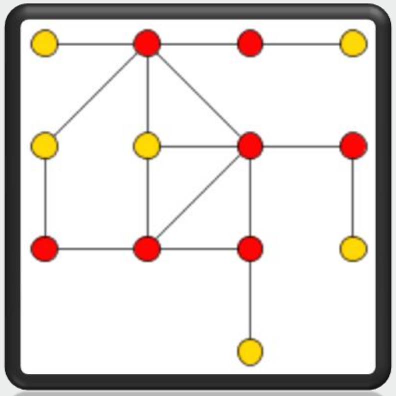


Graph parameters

- **Vertex Cover Number VC(G)**
- Size of a smallest set S such that G – S is edgeless



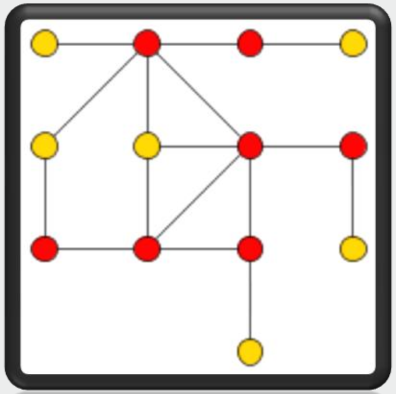
Vertex Cover
Edgeless Graphs



Graph parameters

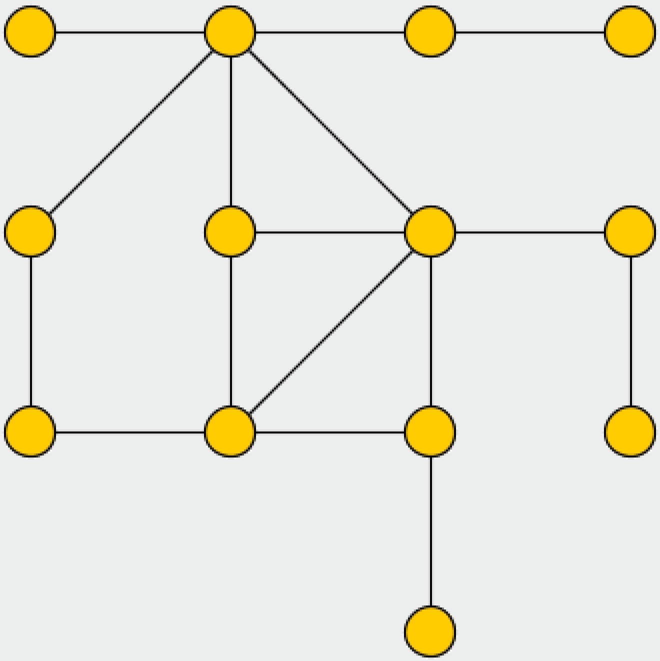
- **Feedback Vertex Set Number FVS(G)**
- Size of a smallest set S such that G – S is a forest (acyclic)

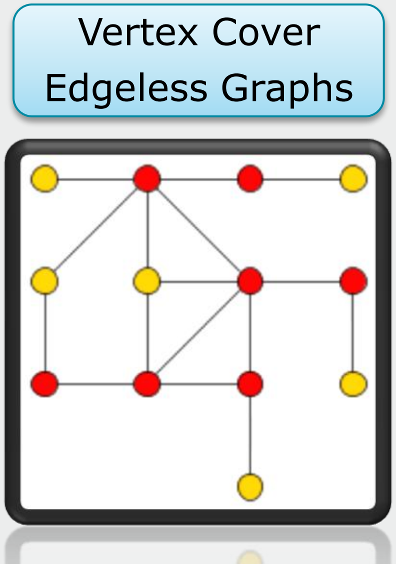
Vertex Cover
Edgeless Graphs



Graph parameters

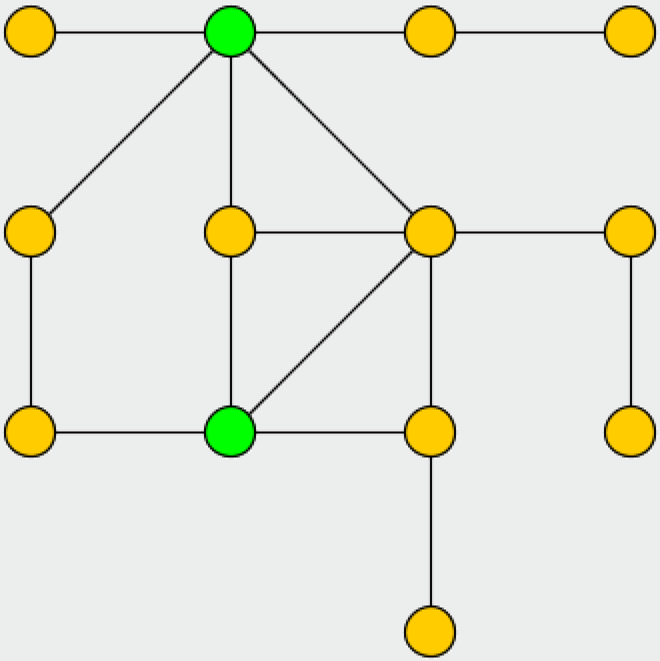
- **Feedback Vertex Set Number FVS(G)**
- Size of a smallest set S such that G – S is a forest (acyclic)



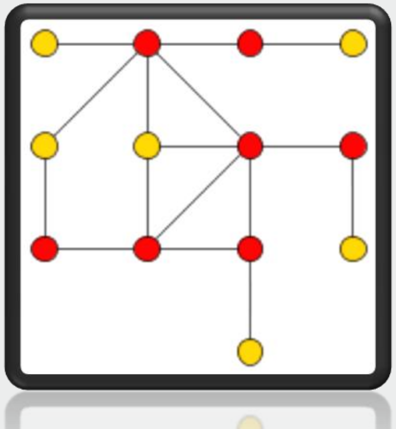


Graph parameters

- **Feedback Vertex Set Number FVS(G)**
- Size of a smallest set S such that G – S is a forest (acyclic)

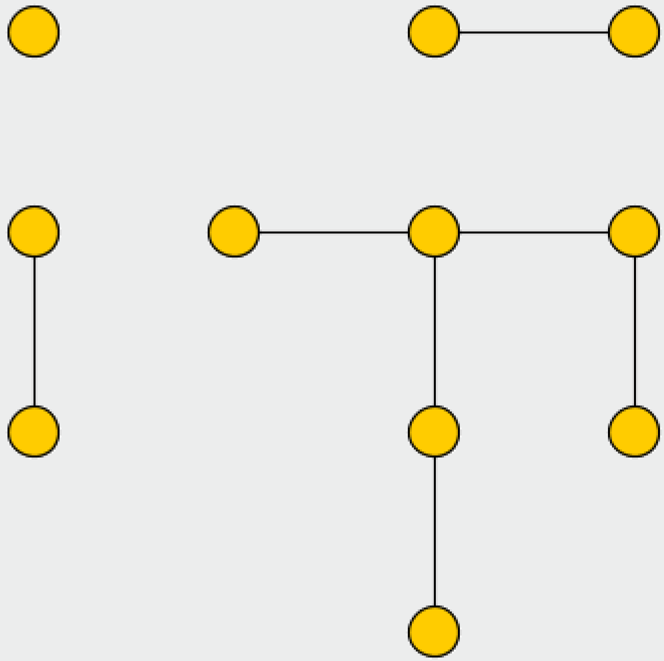


Vertex Cover
Edgeless Graphs

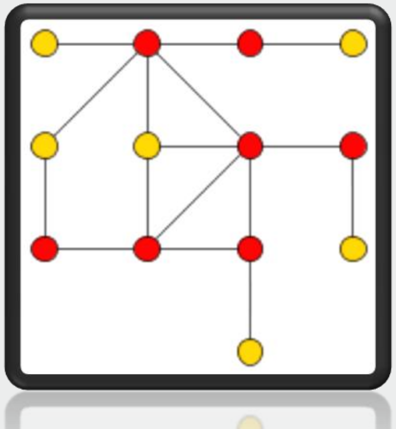


Graph parameters

- **Feedback Vertex Set Number FVS(G)**
- Size of a smallest set S such that G – S is a forest (acyclic)

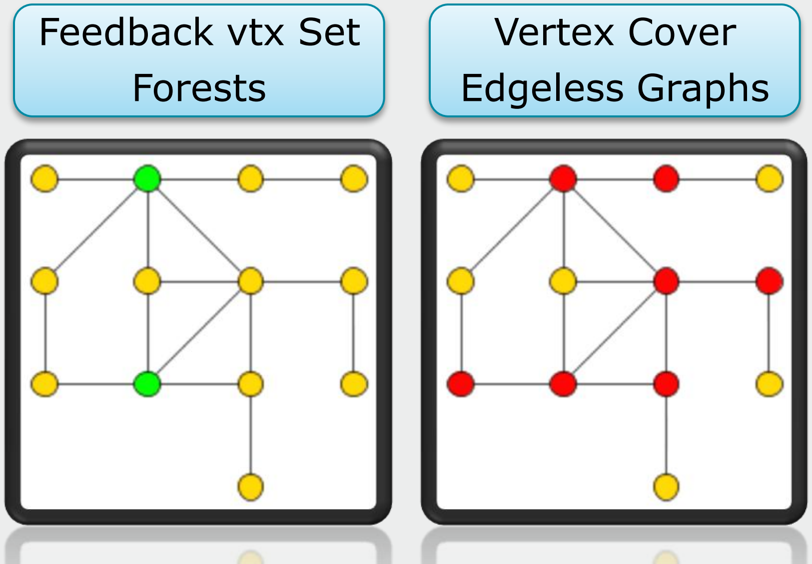


Vertex Cover
Edgeless Graphs



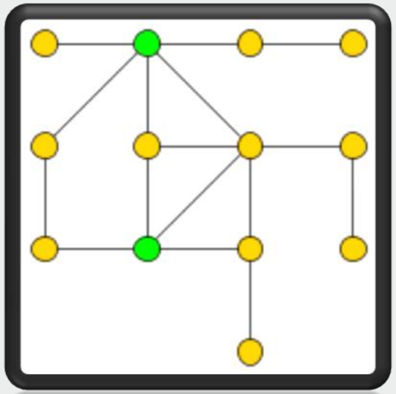
Graph parameters

- **Feedback Vertex Set Number FVS(G)**
- Size of a smallest set S such that G – S is a forest (acyclic)

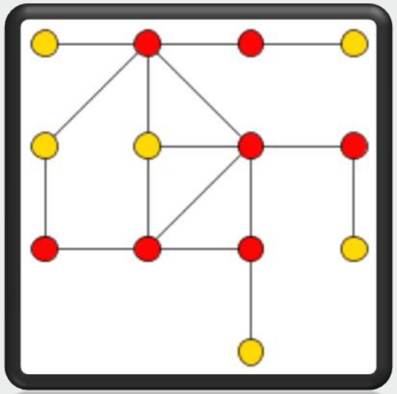


A refined parameter

Feedback vtx Set
Forests

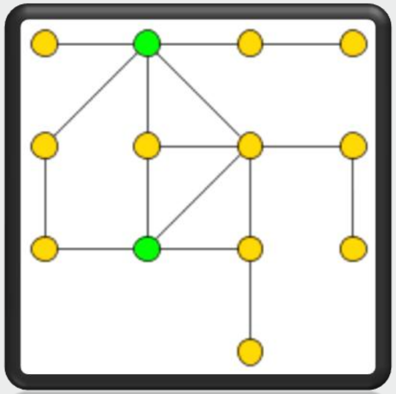


Vertex Cover
Edgeless Graphs



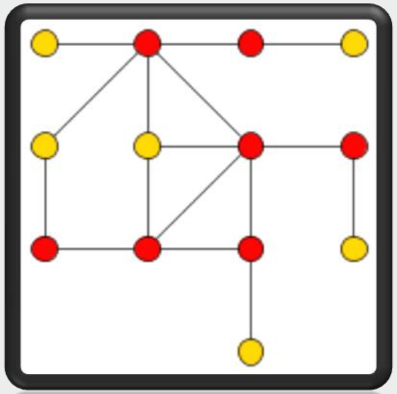
A refined parameter

Feedback vtx Set
Forests



≤

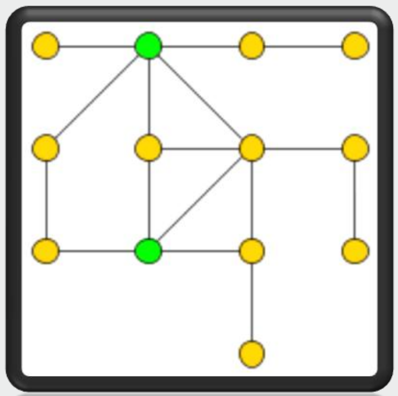
Vertex Cover
Edgeless Graphs



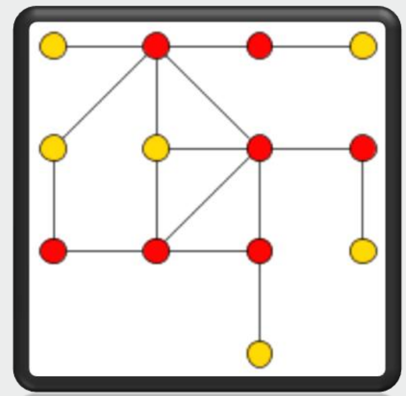
A refined parameter

Feedback vtx Set
Forests

Vertex Cover
Edgeless Graphs

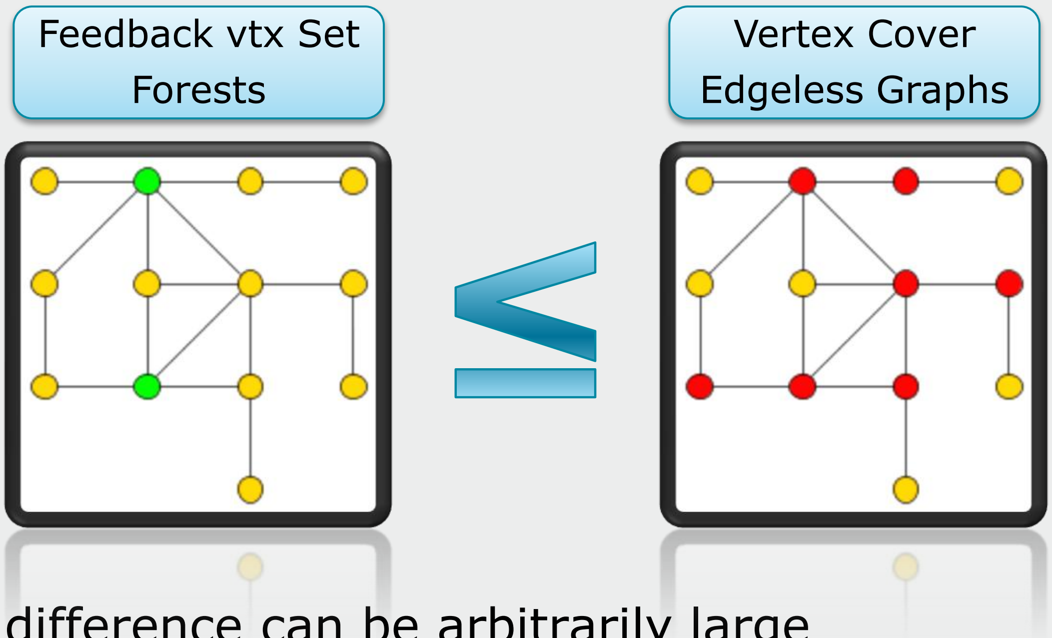


$\leq$



- The difference can be arbitrarily large

A refined parameter



- The difference can be arbitrarily large
- The feedback vertex number is a **refined** parameter

Our results

1. The size of an instance can efficiently be reduced to a polynomial in the size of the minimum FVS

Our results

1. The size of an instance can efficiently be reduced to a polynomial in the size of the minimum FVS

(G,k) → p-time → O [min VC(G), FVS(G)^3] vertices

Our results

1. The size of an instance can efficiently be reduced to a polynomial in the size of the minimum FVS

(G,k) → p-time → O [min VC(G), FVS(G)^3] vertices

- In the language of parameterized complexity:
  - Vertex Cover parameterized by the size of a feedback vertex set admits a cubic-vertex kernel

Our results

2. The **Weighted** Vertex Cover problem cannot be reduced to an instance on poly[VC(G)] vertices *
    - (unless the polynomial hierarchy collapses)
    - where VC(G) is the **cardinality** of a minimum vertex cover

Our results

2. The **Weighted** Vertex Cover problem cannot be reduced to an instance on poly[VC(G)] vertices *****
    - (unless the polynomial hierarchy collapses)
    - where VC(G) is the **cardinality** of a minimum vertex cover

- Reduction to O((W-VC(G)) vertices *is* possible

Our results

2. The **Weighted** Vertex Cover problem cannot be reduced to an instance on poly[VC(G)] vertices *
    - (unless the polynomial hierarchy collapses)
    - where VC(G) is the **cardinality** of a minimum vertex cover

- Reduction to O((W-VC(G)) vertices *is* possible
    - where W-VC(G) is the **weight** of a minimum vertex cover [Chlebík and Chlebíková, Disc Appl M 2008]

Our results

2. The **Weighted** Vertex Cover problem cannot be reduced to an instance on poly[VC(G)] vertices *
    - (unless the polynomial hierarchy collapses)
    - where VC(G) is the **cardinality** of a minimum vertex cover

- Reduction to O((W-VC(G)) vertices *is* possible
    - where W-VC(G) is the **weight** of a minimum vertex cover [Chlebík and Chlebíková, Disc Appl M 2008]

- Weighted Vertex Cover can be solved in $2^{VC(G)}$ poly(n) time

Our results

2. The **Weighted** Vertex Cover problem cannot be reduced to an instance on poly[VC(G)] vertices *****
   - (unless the polynomial hierarchy collapses)
   - where VC(G) is the **cardinality** of a minimum vertex cover

- Reduction to O((W-VC(G)) vertices *is* possible
  - where W-VC(G) is the **weight** of a minimum vertex cover [Chlebík and Chlebíková, Disc Appl M 2008]

- Weighted Vertex Cover can be solved in $2^{VC(G)}$ poly(n) time

*** This strenghtens the result as given in the paper**

Sketch of the reduction rules

THE UPPER BOUNDS

Outline of the reduction algorithm

- Input: an instance (G,k) of Vertex Cover

Outline of the reduction algorithm

- Input: an instance (G,k) of Vertex Cover

1) Apply the Nemhauser-Trotter reduction
   - This effectively deletes vertices, so FVS(G) is not increased

Outline of the reduction algorithm

- Input: an instance (G,k) of Vertex Cover

1) Apply the Nemhauser-Trotter reduction
   - This effectively deletes vertices, so FVS(G) is not increased

2) Compute a 2-approximate Feedback Vertex Set X
   - [Bafna, Berman and Fujito, SIAM J Disc M 1999]

Outline of the reduction algorithm

- Input: an instance (G,k) of Vertex Cover

1) Apply the Nemhauser-Trotter reduction
   - This effectively deletes vertices, so FVS(G) is not increased

2) Compute a 2-approximate Feedback Vertex Set X
   - [Bafna, Berman and Fujito, SIAM J Disc M 1999]

3) Use the structure of X within G to apply reduction rules
   - When no rules apply, the instance is provably small

Change of perspective

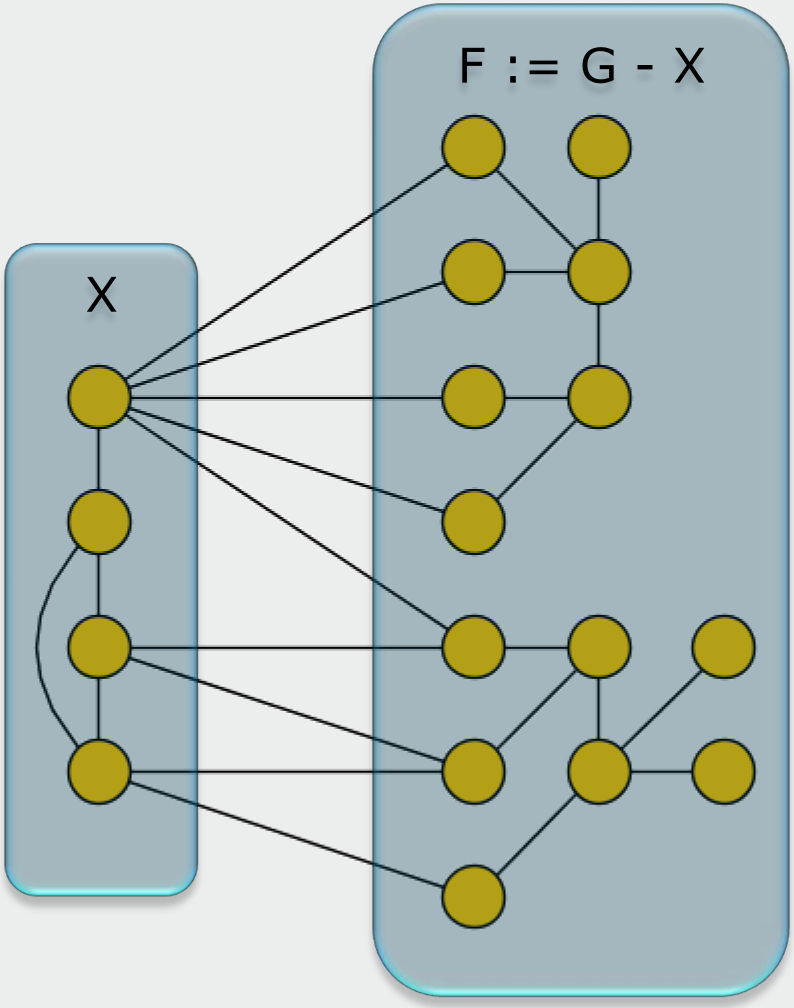
- Instance (G,k) of Vertex Cover is equivalent to asking "Does G have an Independent Set of size n – k?"

Change of perspective

- Instance (G,k) of Vertex Cover is equivalent to asking "Does G have an Independent Set of size n – k?"
- Reduction rules are easier to formulate in Independent Set perspective
  - Interpret (G,k) as an instance (G, n – k) of Independent Set
  - Apply reduction rules to obtain a small instance of Independent Set (G', n' – k')
  - Equivalent to the small Vertex Cover instance (G', k')

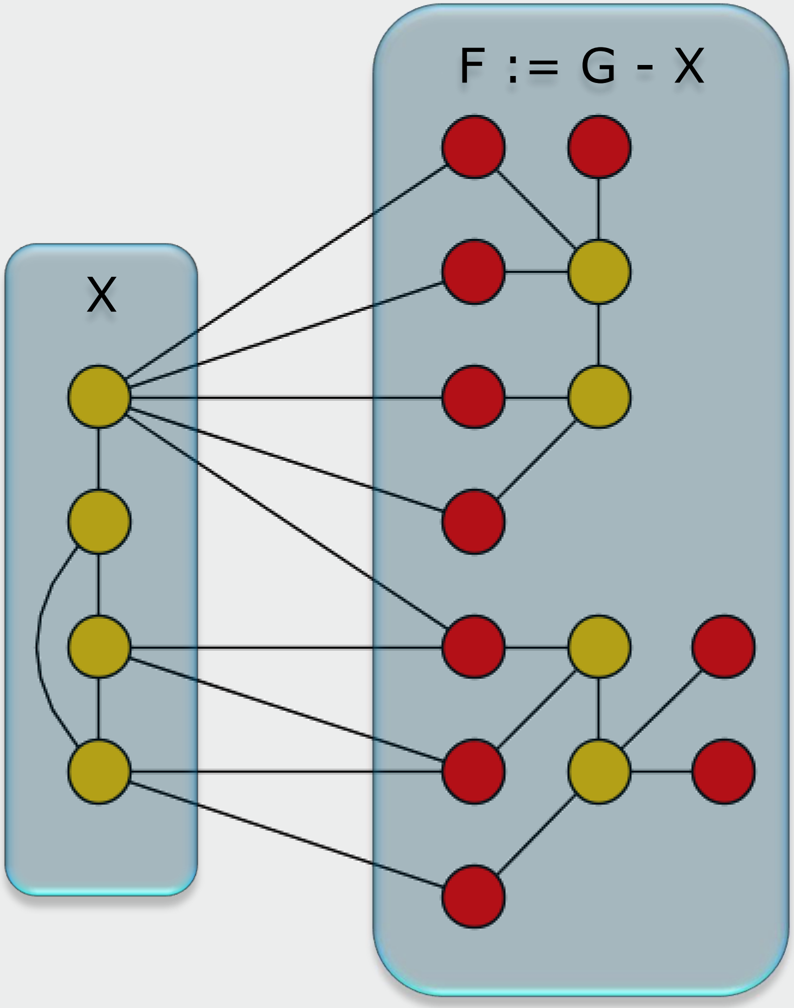
Structure of an instance: a canonical solution

- Let forest F := G – X
  - Maximum Independent Set (MIS) of F is poly-time computable



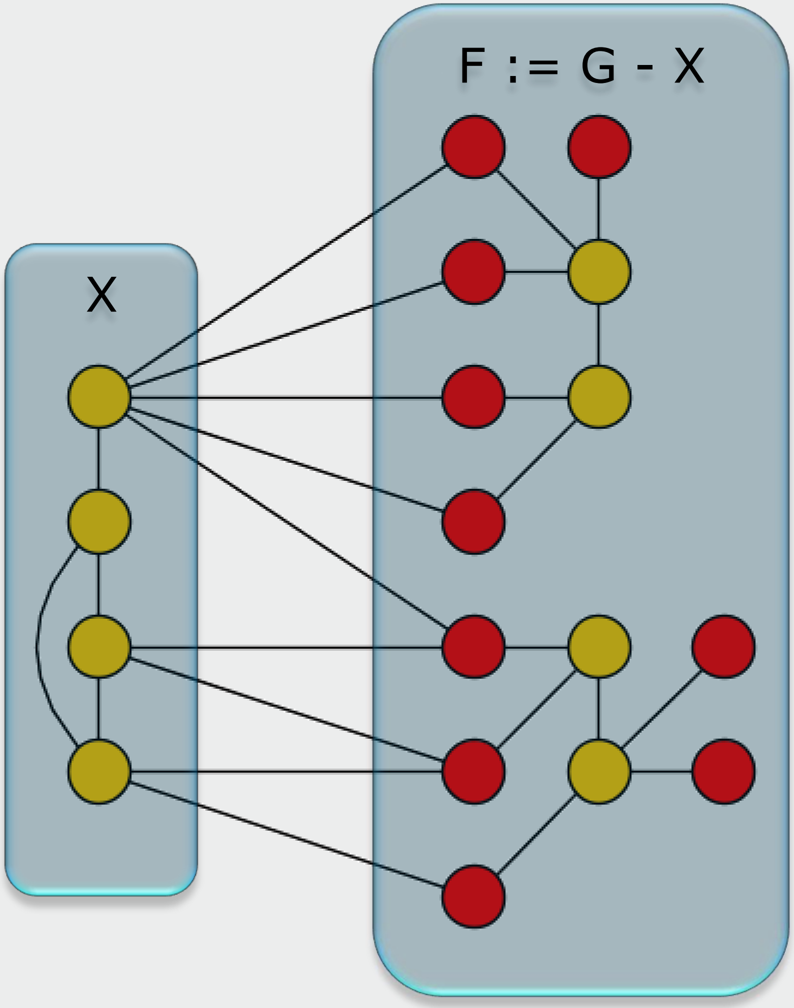
Structure of an instance: a canonical solution

- Let forest F := G – X
  - Maximum Independent Set (MIS) of F is poly-time computable
- Canonical solution=MIS(F)



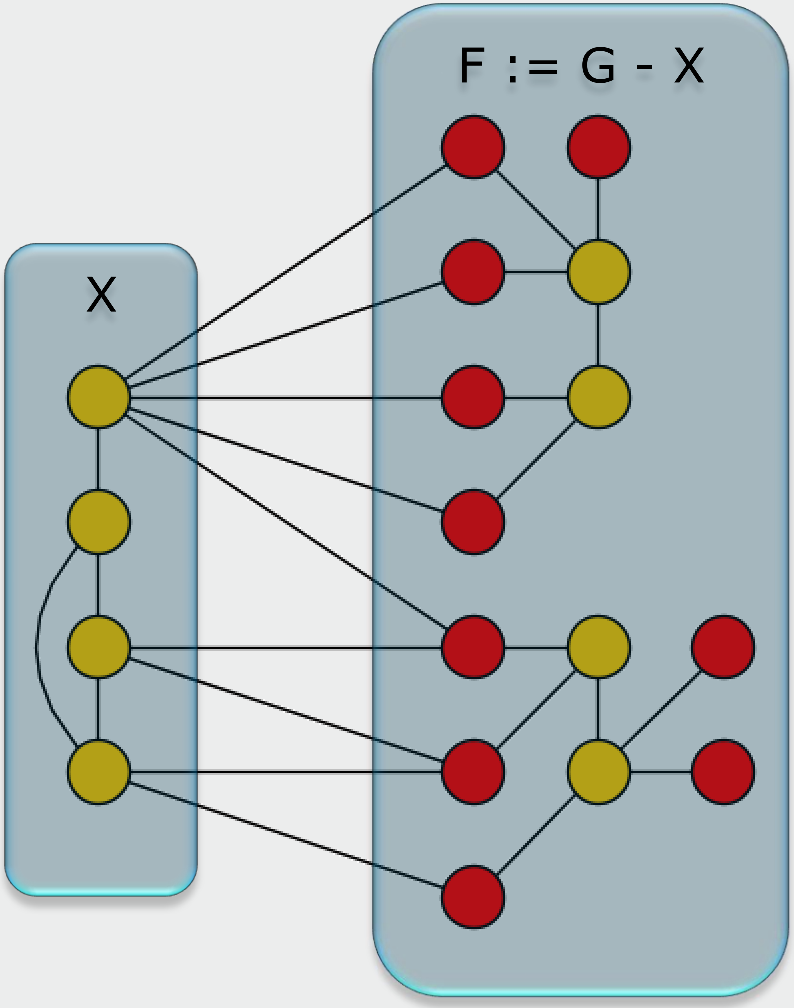
Structure of an instance: a canonical solution

- Let forest F := G – X
  - Maximum Independent Set (MIS) of F is poly-time computable
- Canonical solution=MIS(F)
- Better solutions may exist using some vertices of X



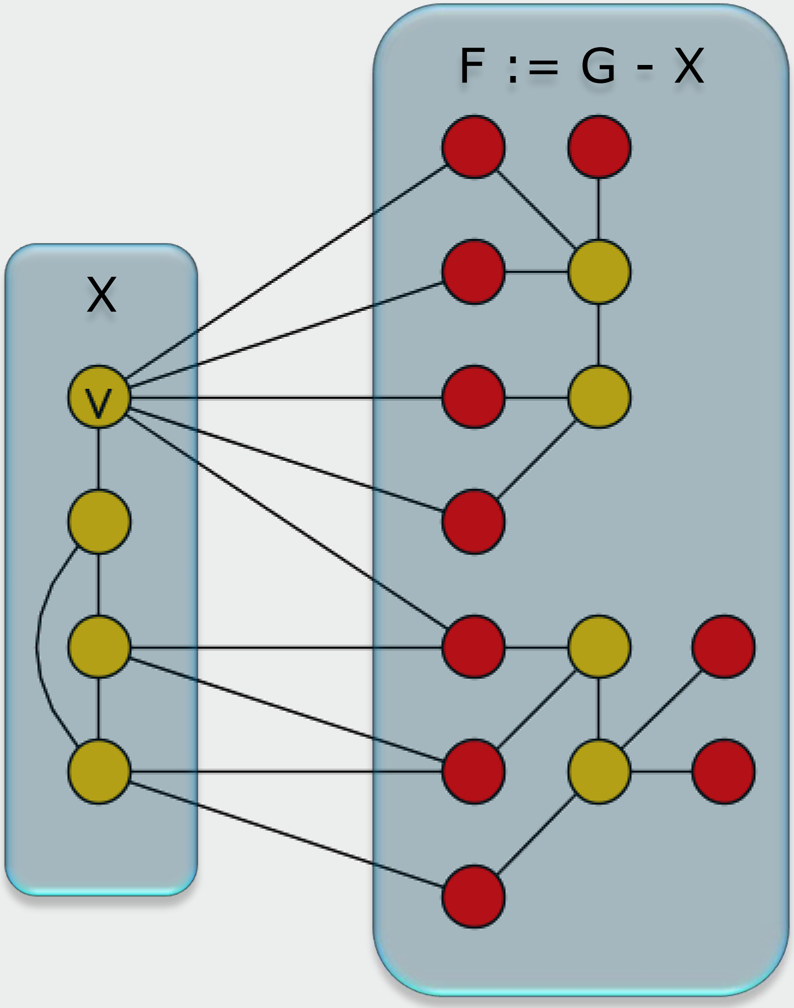
Structure of an instance: a canonical solution

- Let forest F := G – X
  - Maximum Independent Set (MIS) of F is poly-time computable
- Canonical solution=MIS(F)
- Better solutions may exist using some vertices of X
- We can test the effect of using single vertex



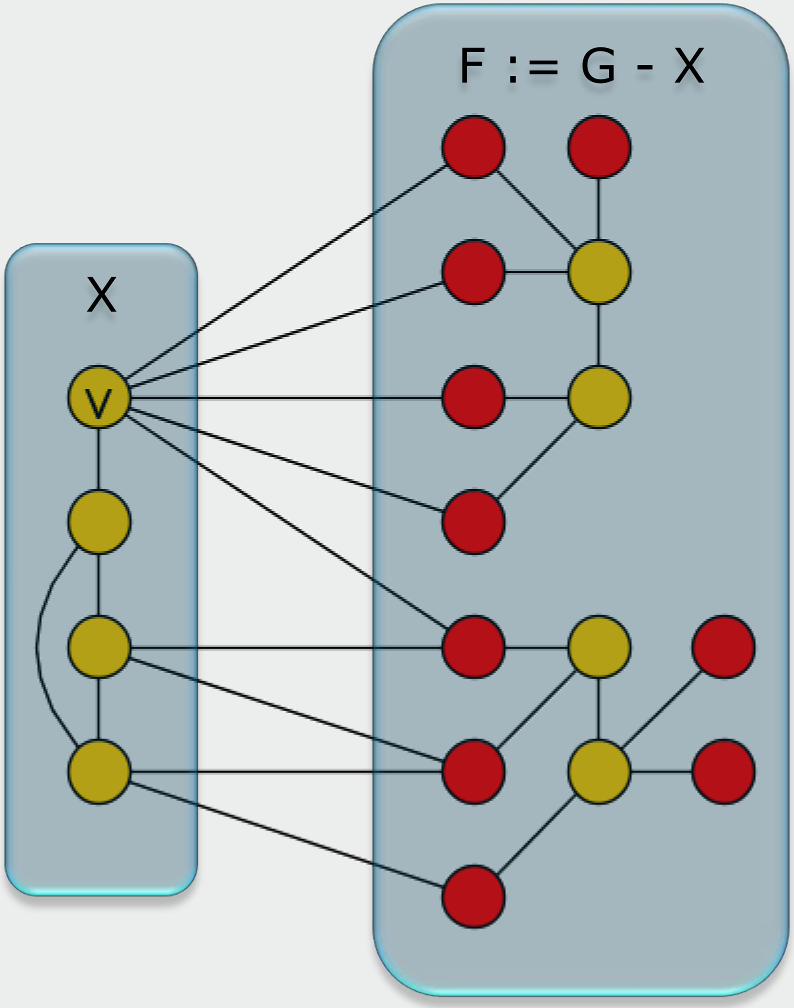
Using vertices from X

- Consider using vertex v in X in an independent set



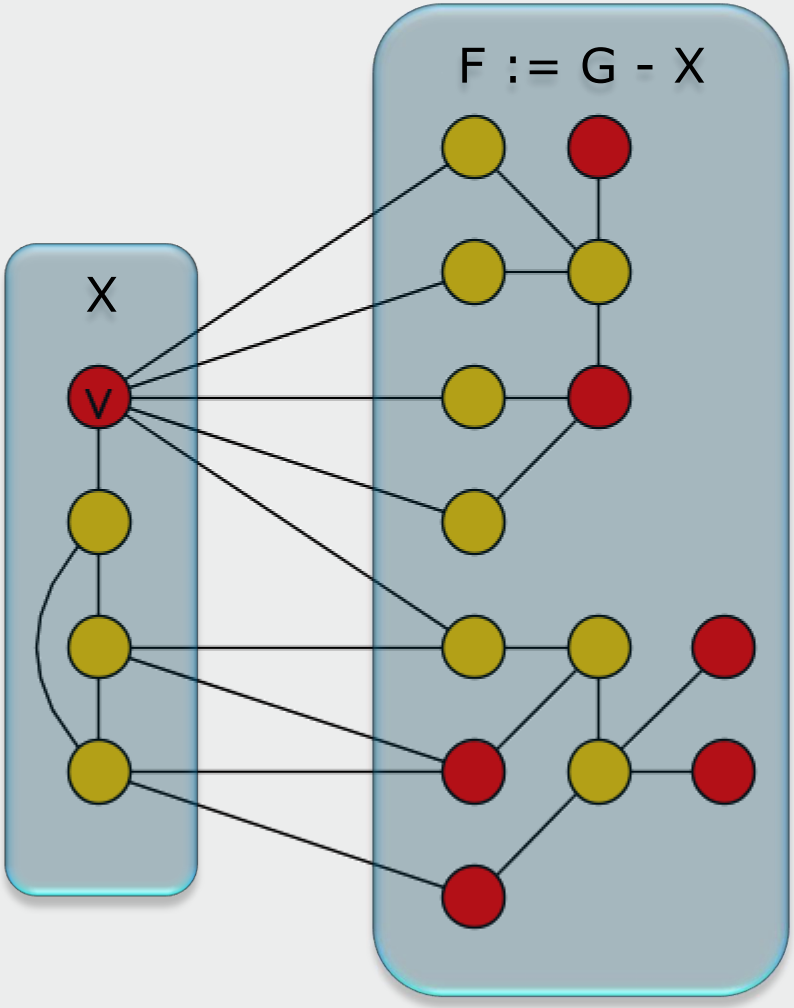
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v



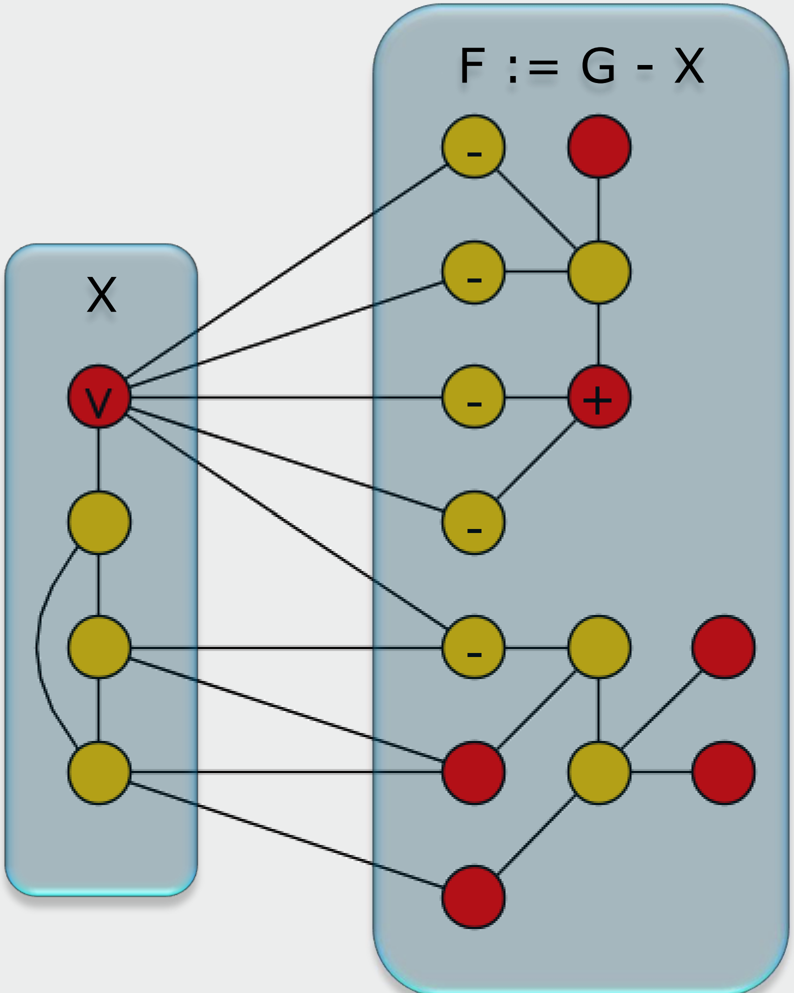
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v



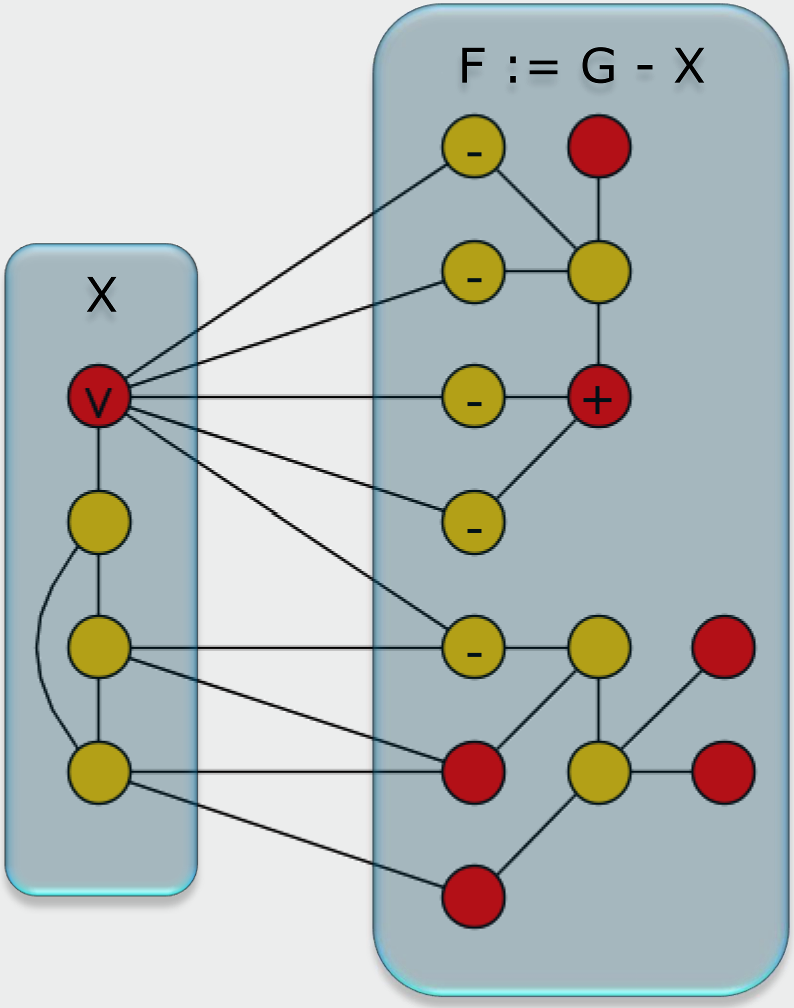
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v



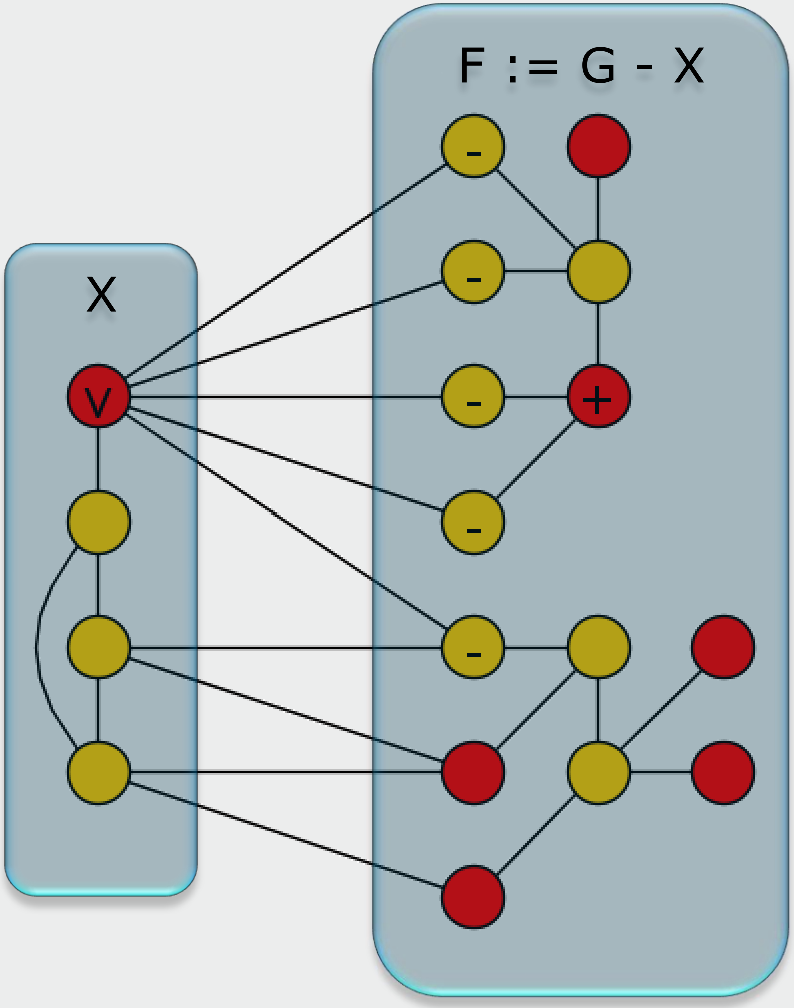
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v
- Compare canonical solution to MIS(F – N(v))



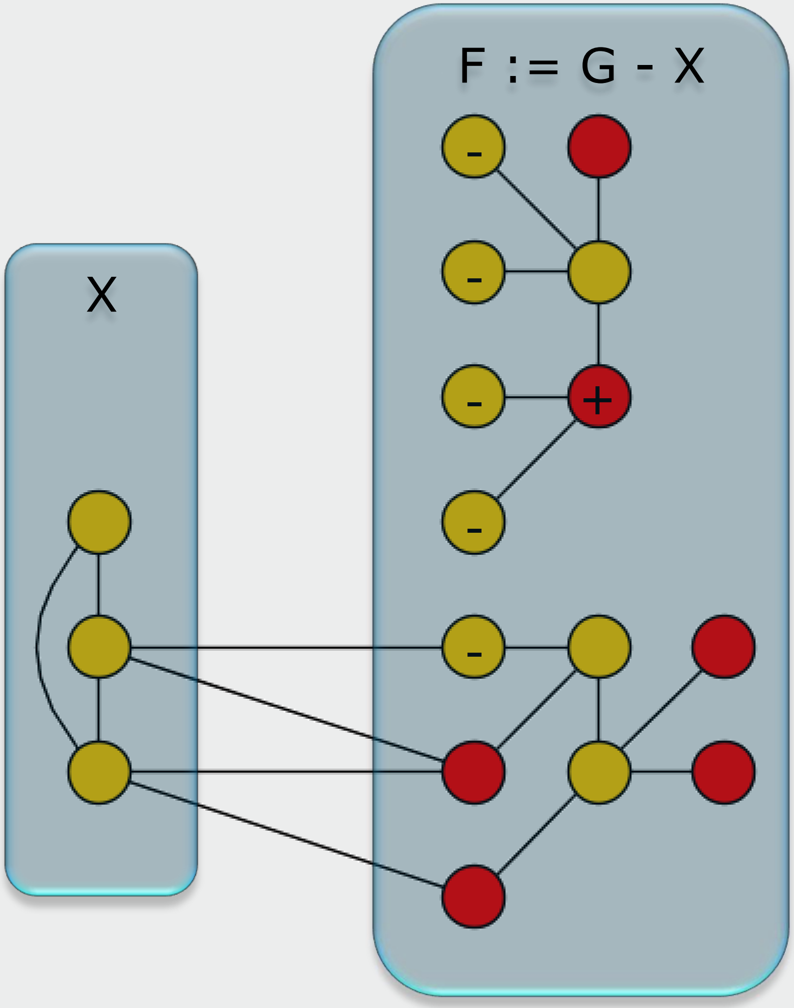
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v
- Compare canonical solution to MIS(F – N(v))
- If difference ≥ |X|:
  - Solutions containing v are not better than canonical
  - Delete v from the instance



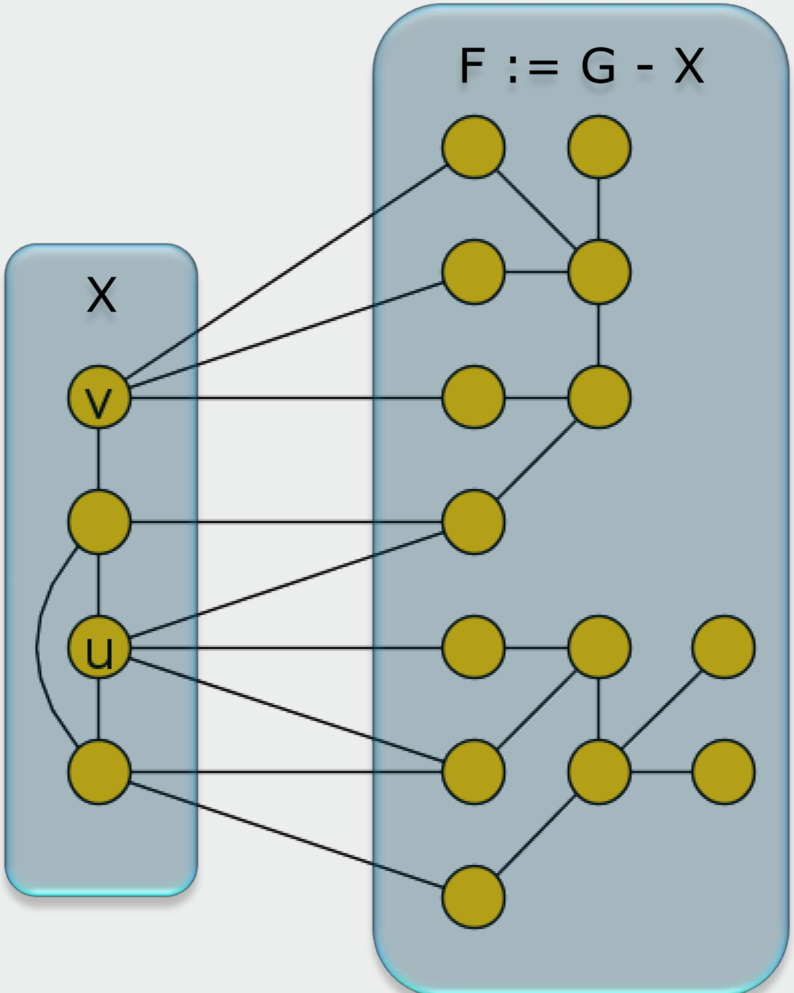
Using vertices from X

- Consider using vertex v in X in an independent set
- This IS cannot use any neighbors of v
- Compare canonical solution to MIS(F – N(v))
- If difference ≥ |X|:
  - Solutions containing v are not better than canonical
  - Delete v from the instance



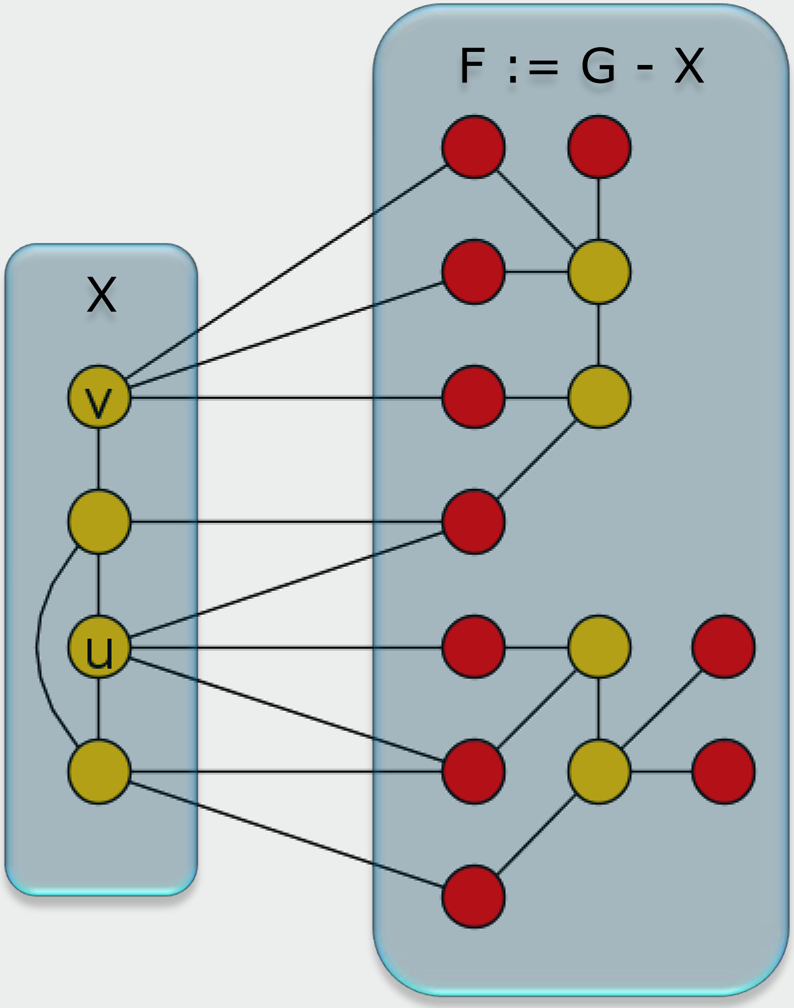
Using pairs of vertices from X

- Different situation



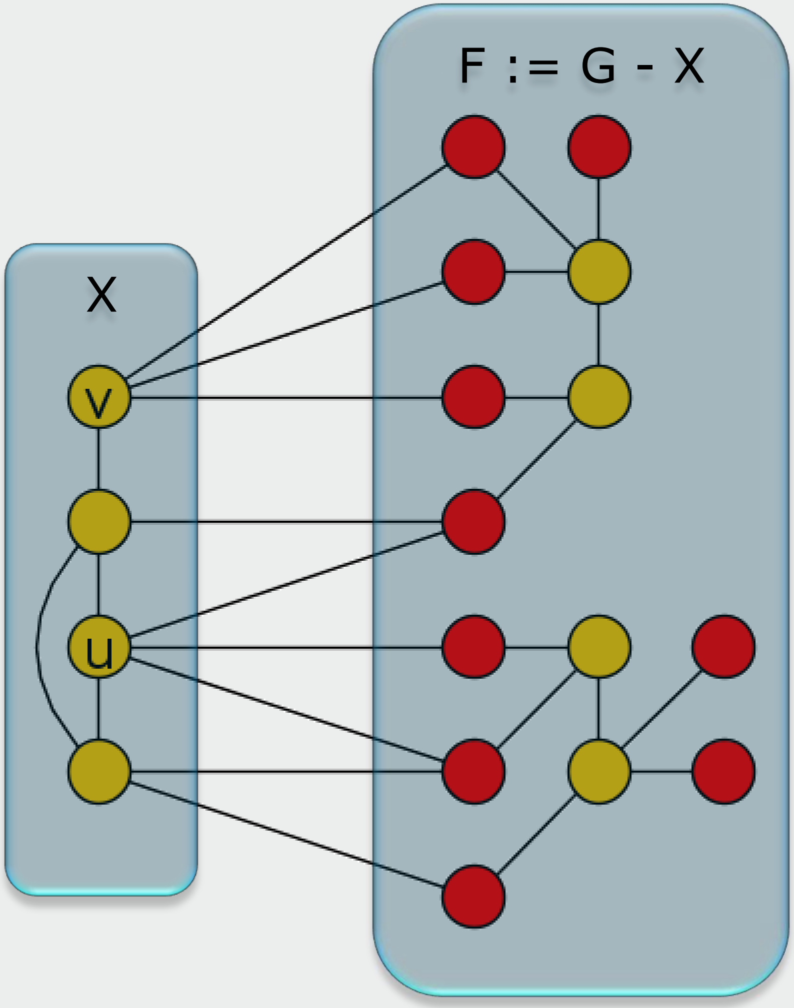
Using pairs of vertices from X

- Different situation



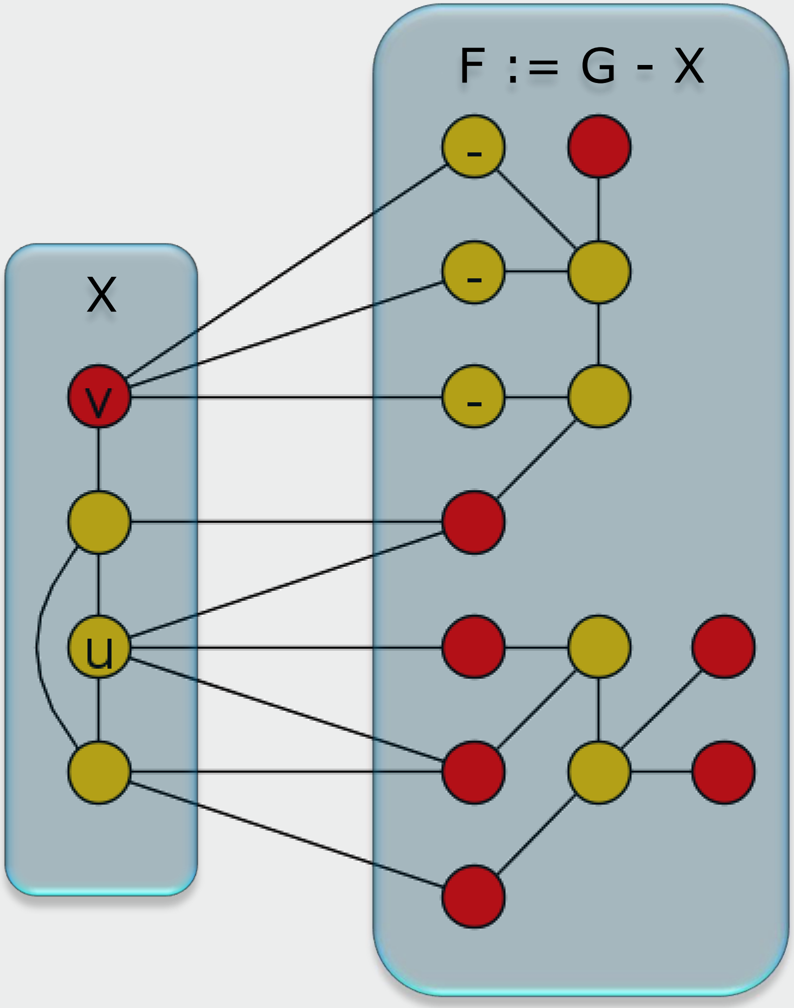
Using pairs of vertices from X

- Different situation
- No single vertex triggers the reduction rule



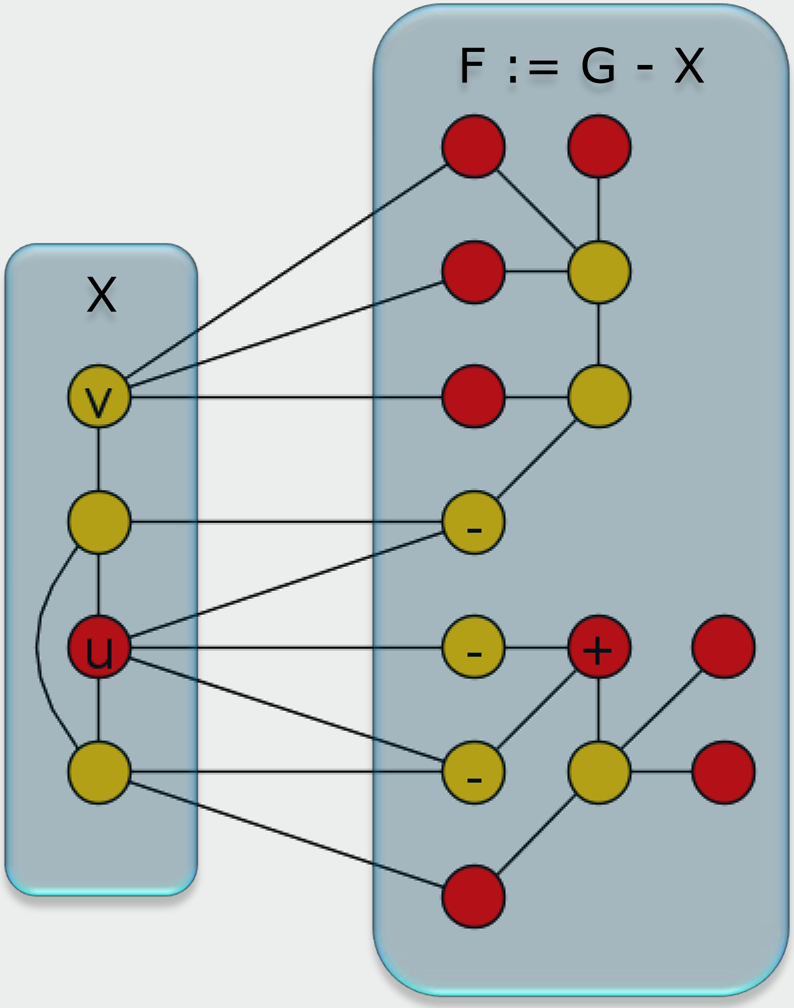
Using pairs of vertices from X

- Different situation
- No single vertex triggers the reduction rule



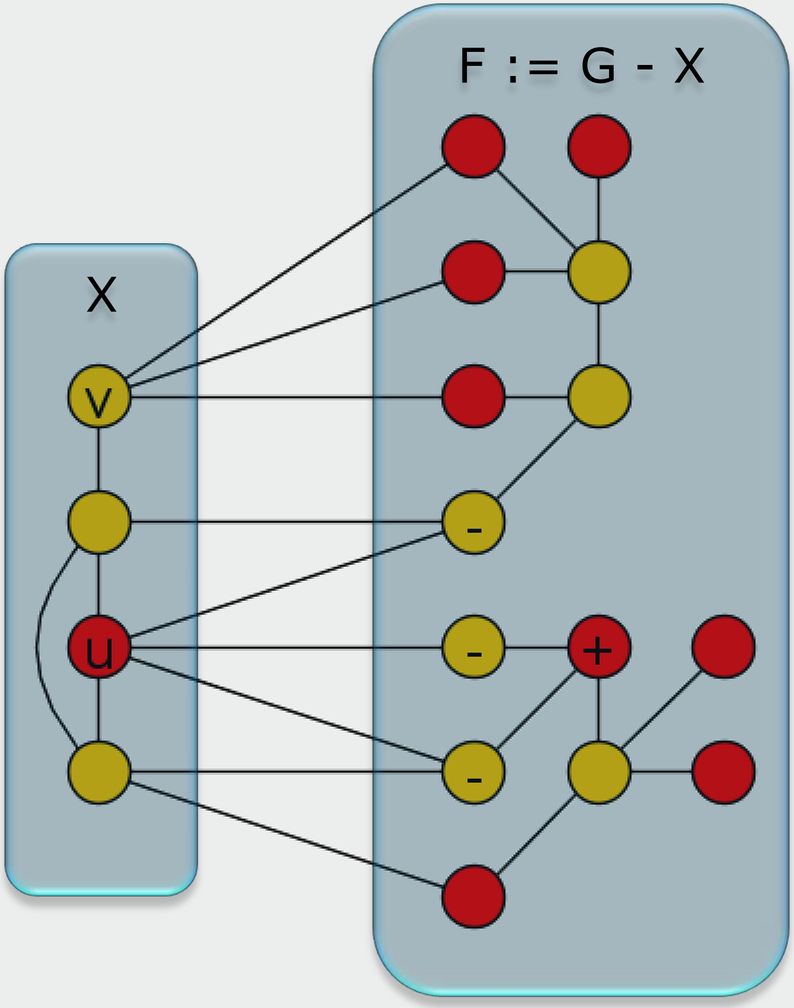
Using pairs of vertices from X

- Different situation
- No single vertex triggers the reduction rule



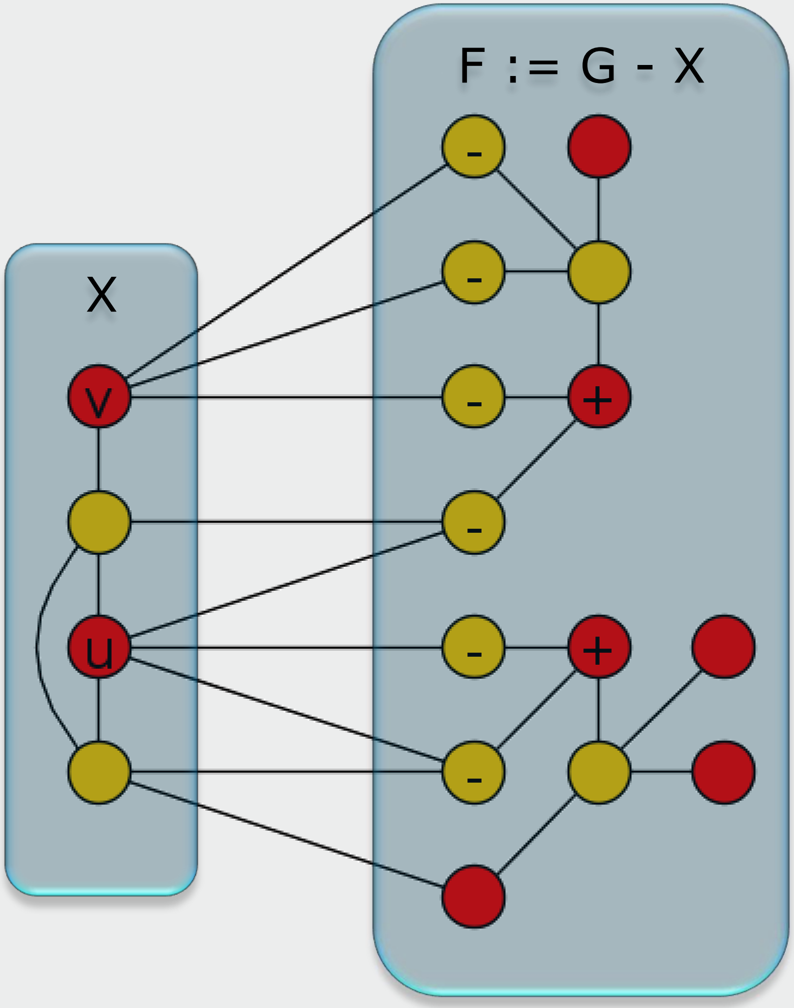
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent



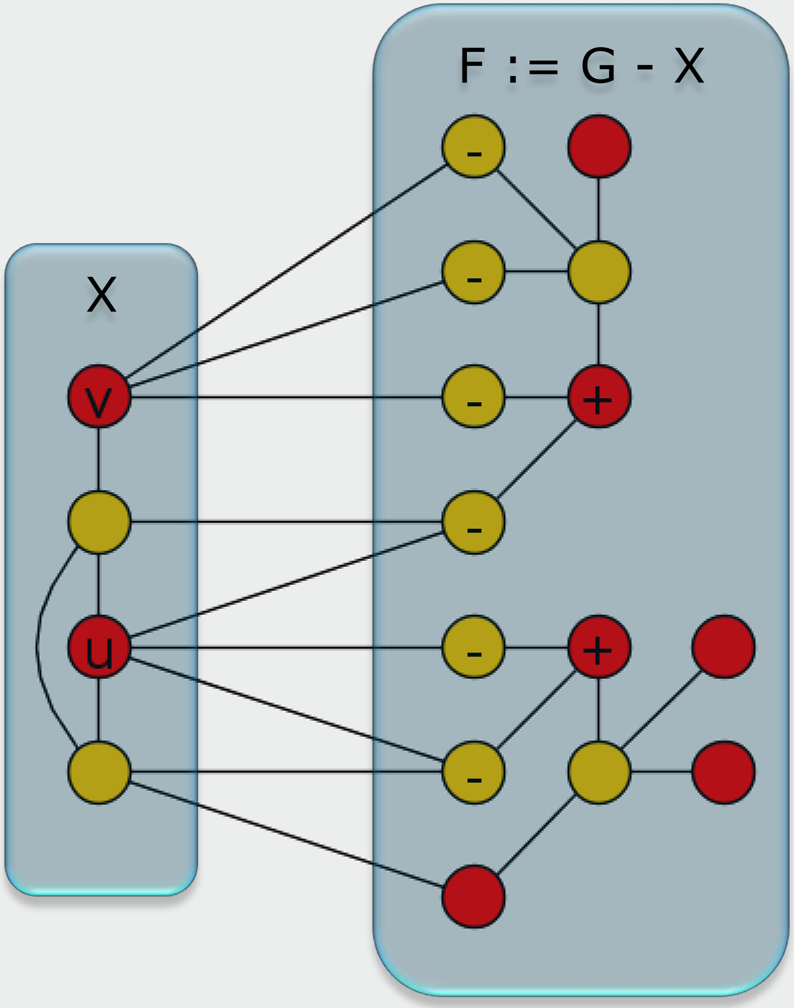
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent



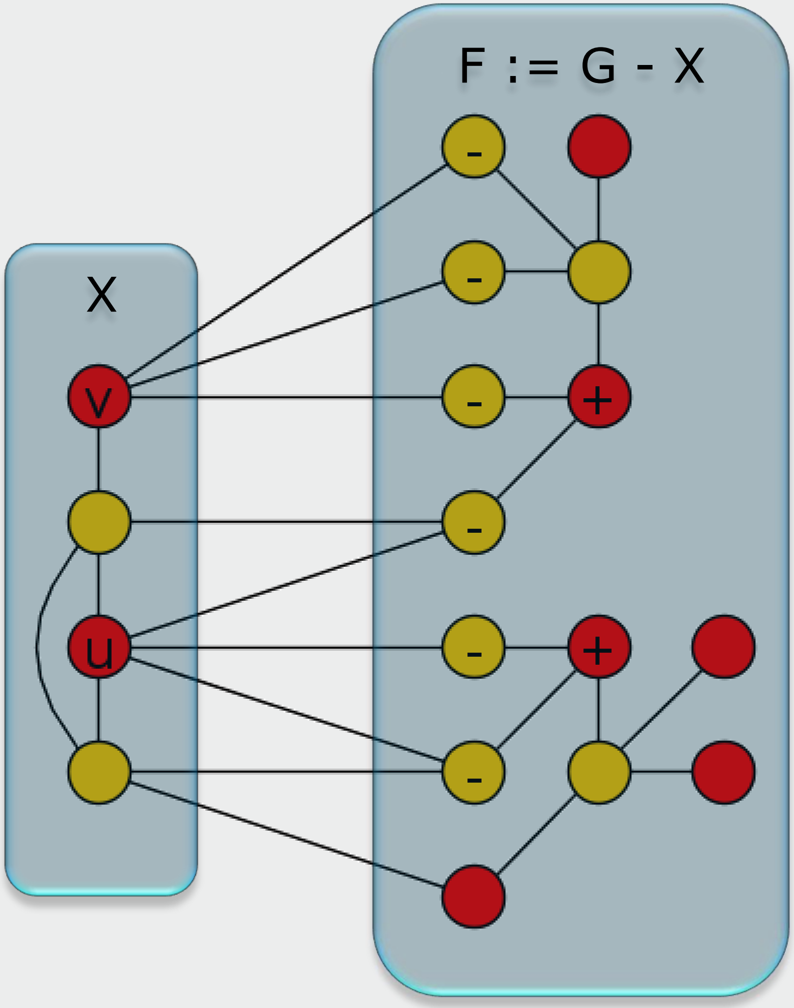
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent
- Compare canonical solution to MIS(F – N(u,v))



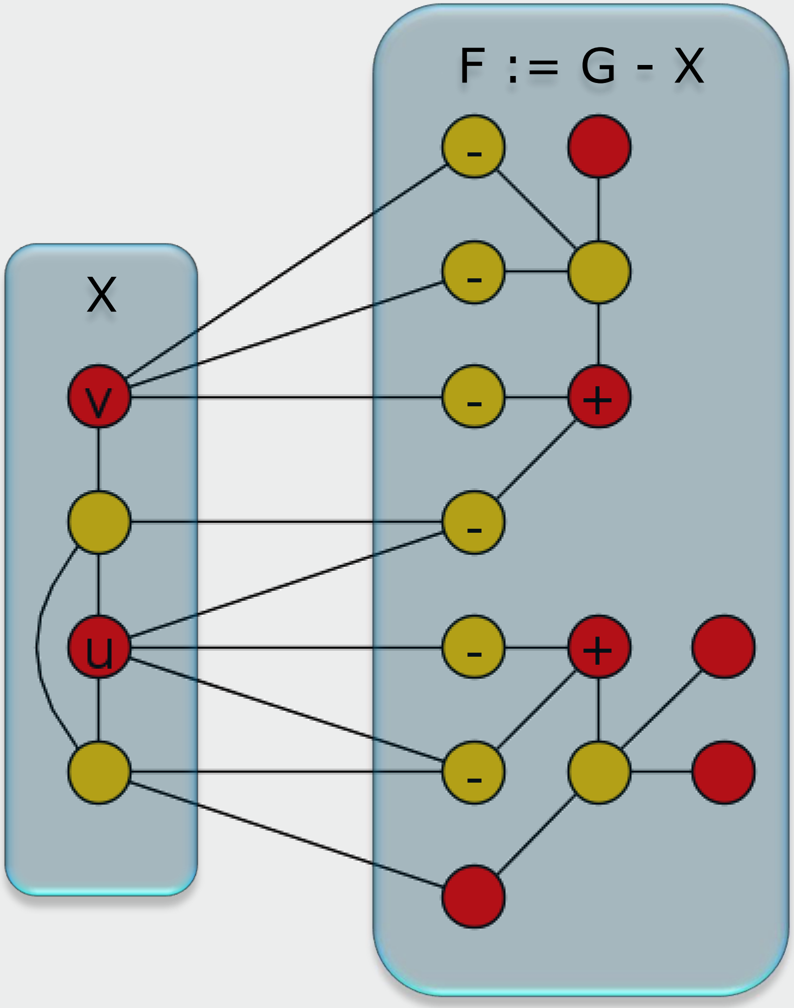
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent
- Compare canonical solution to MIS(F – N(u,v))
- If difference ≥ |X|:
  - Solutions containing {u,v} are not better than canonical



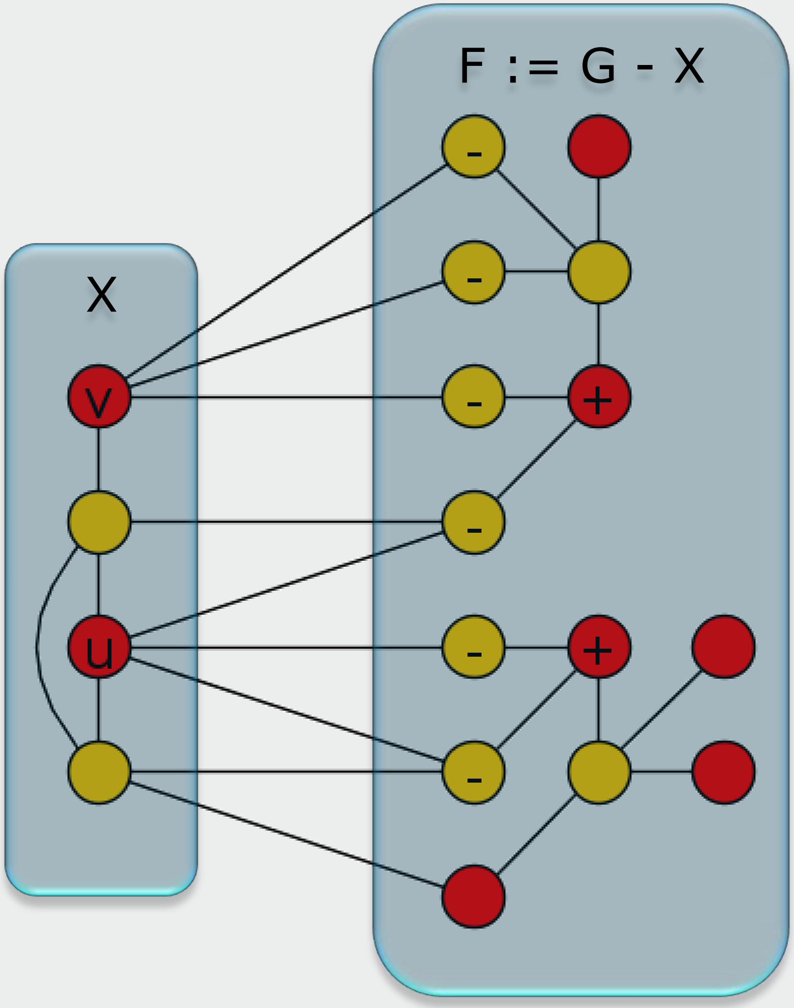
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent
- Compare canonical solution to MIS(F – N(u,v))
- If difference ≥ |X|:
  - Solutions containing {u,v} are not better than canonical
  - Exists optimal solution which does not use both



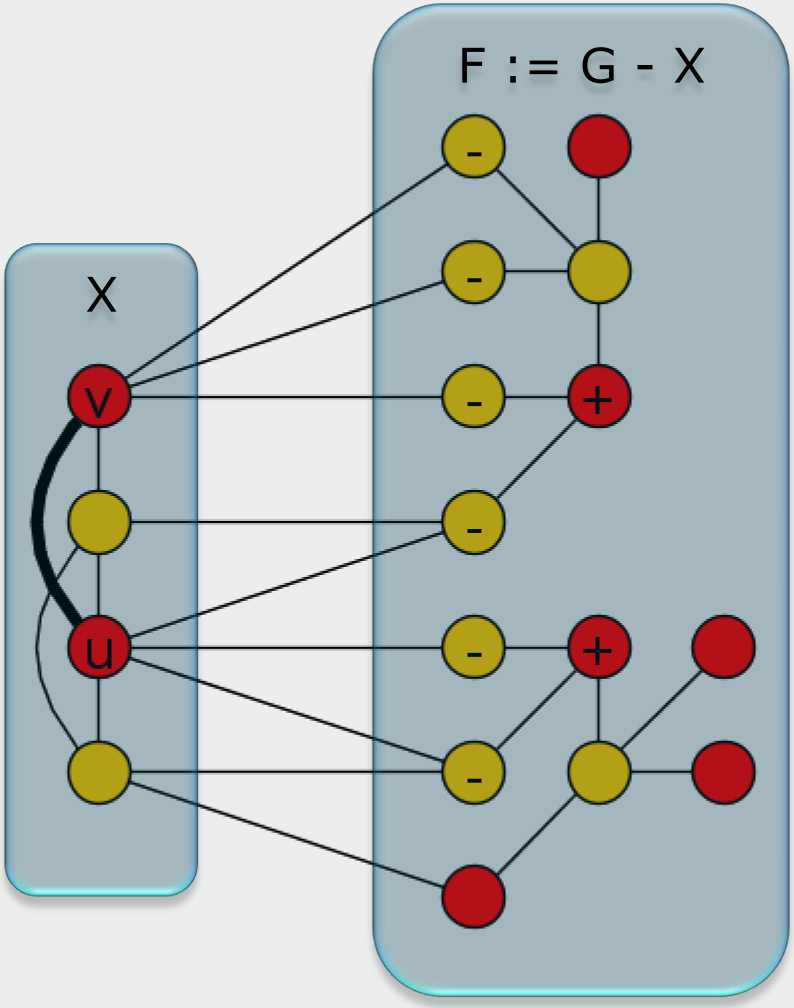
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent
- Compare canonical solution to MIS(F – N(u,v))
- If difference ≥ |X|:
  - Solutions containing {u,v} are not better than canonical
  - Exists optimal solution which does not use both
  - Add edge {u,v}



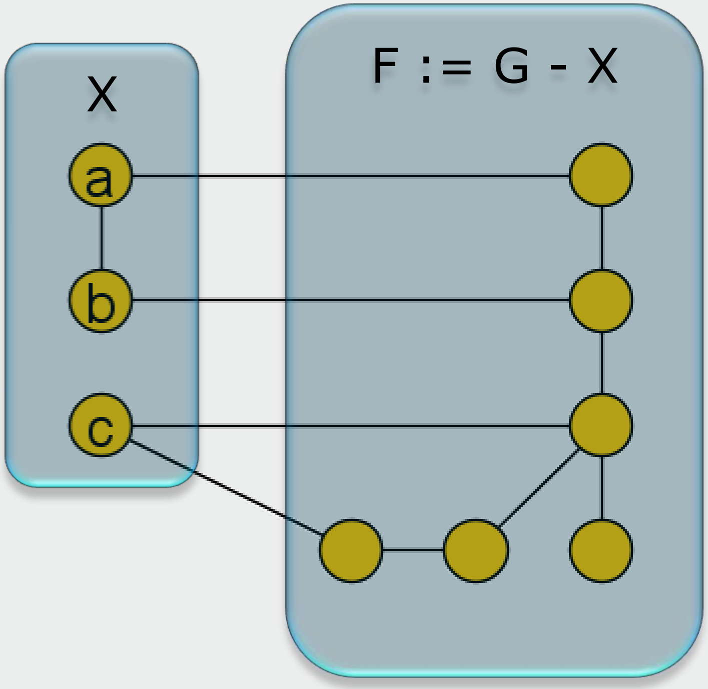
Using pairs of vertices from X

- Consider using {u,v} from X in the independent set
  - Impossible if {u,v} adjacent
- Compare canonical solution to MIS(F – N(u,v))
- If difference ≥ |X|:
  - Solutions containing {u,v} are not better than canonical
  - Exists optimal solution which does not use both
  - Add edge {u,v}



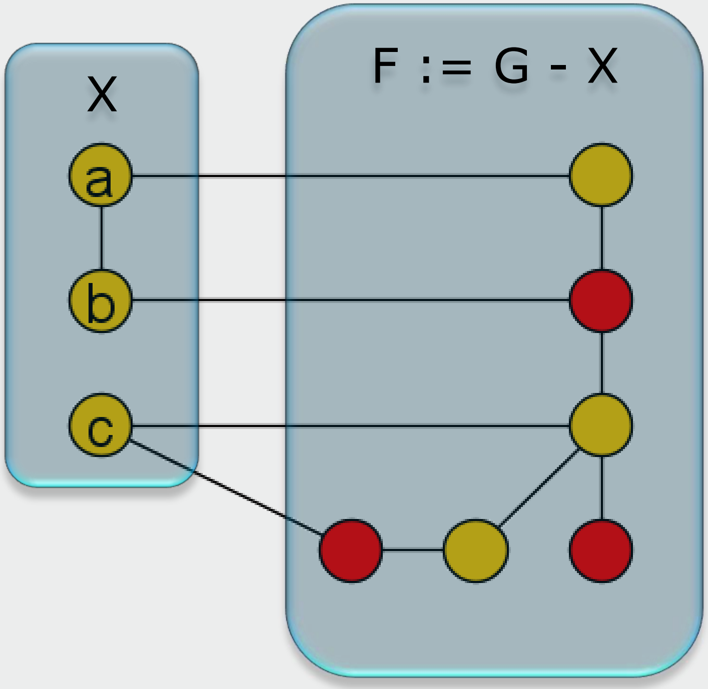
Deleting trees from F: An example

- Consider this tree T in forest F



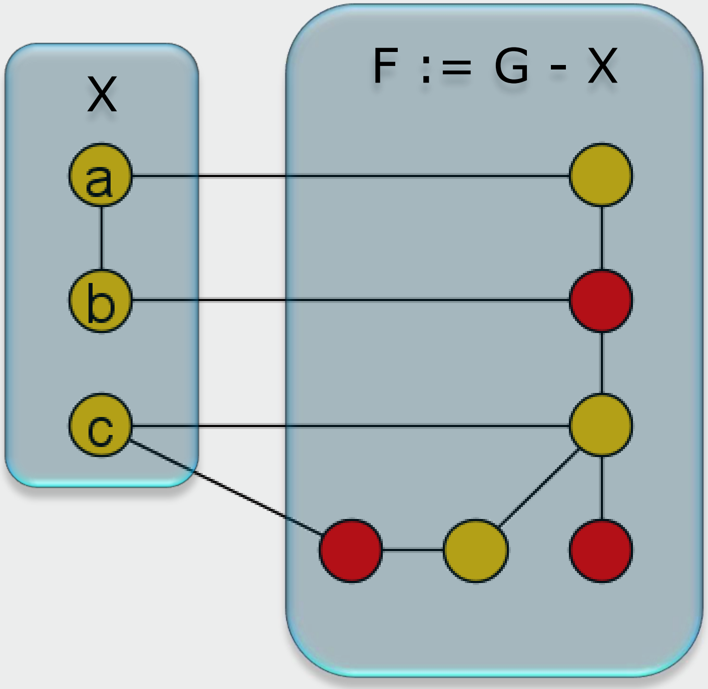
Deleting trees from F: An example

- Consider this tree T in forest F



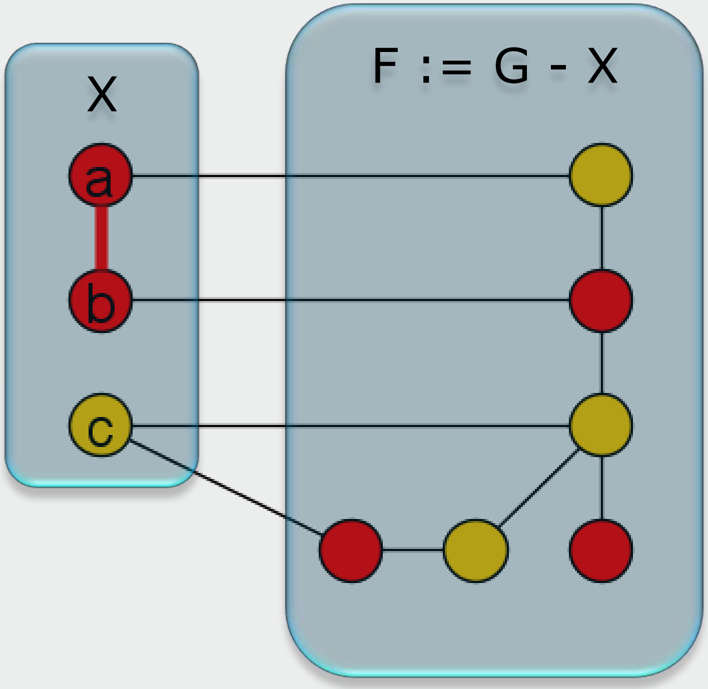
Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T



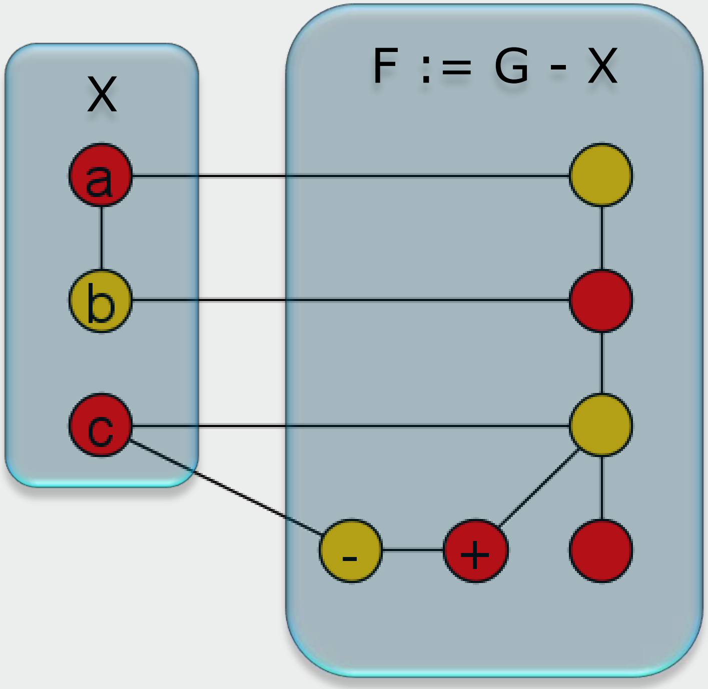
Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T



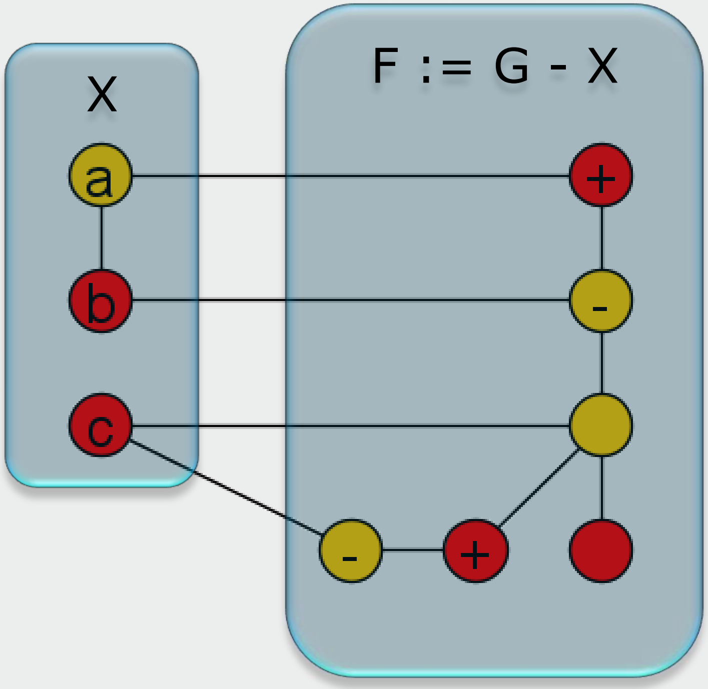
Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T



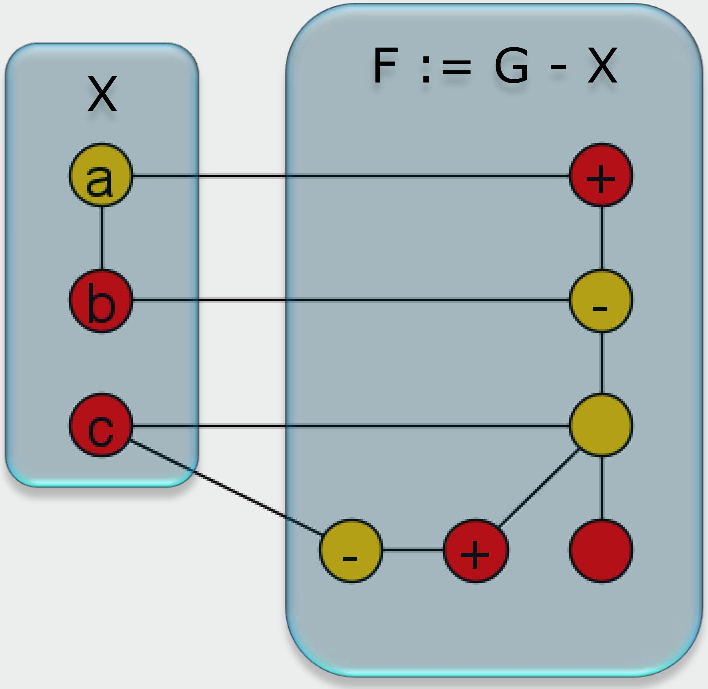
Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T



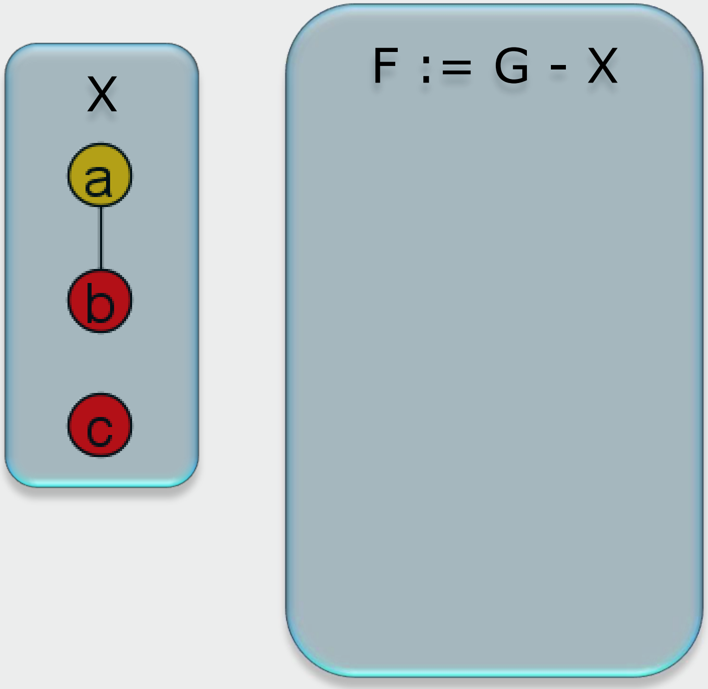
Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T
- Delete T, decrease k by MIS(T)



Deleting trees from F: An example

- Consider this tree T in forest F
- **Any** independent set in X can be augmented with MIS(T) vertices from T
- Delete T, decrease k by MIS(T)



Deleting trees from F: the rule

- If there is a tree T in the forest F, such that:
  - for all non-adjacent pairs {u,v} in X:
    MIS(T) = MIS(T – N(u,v))
- Then delete T from the instance, decrease k by MIS(T)

- Justified by the following lemma:
  - If there is an independent set X' ⊆ X such that
    MIS(T) > MIS(T – N(X'))
  - then there is such a set of size at most 2

Overview of the reduction process

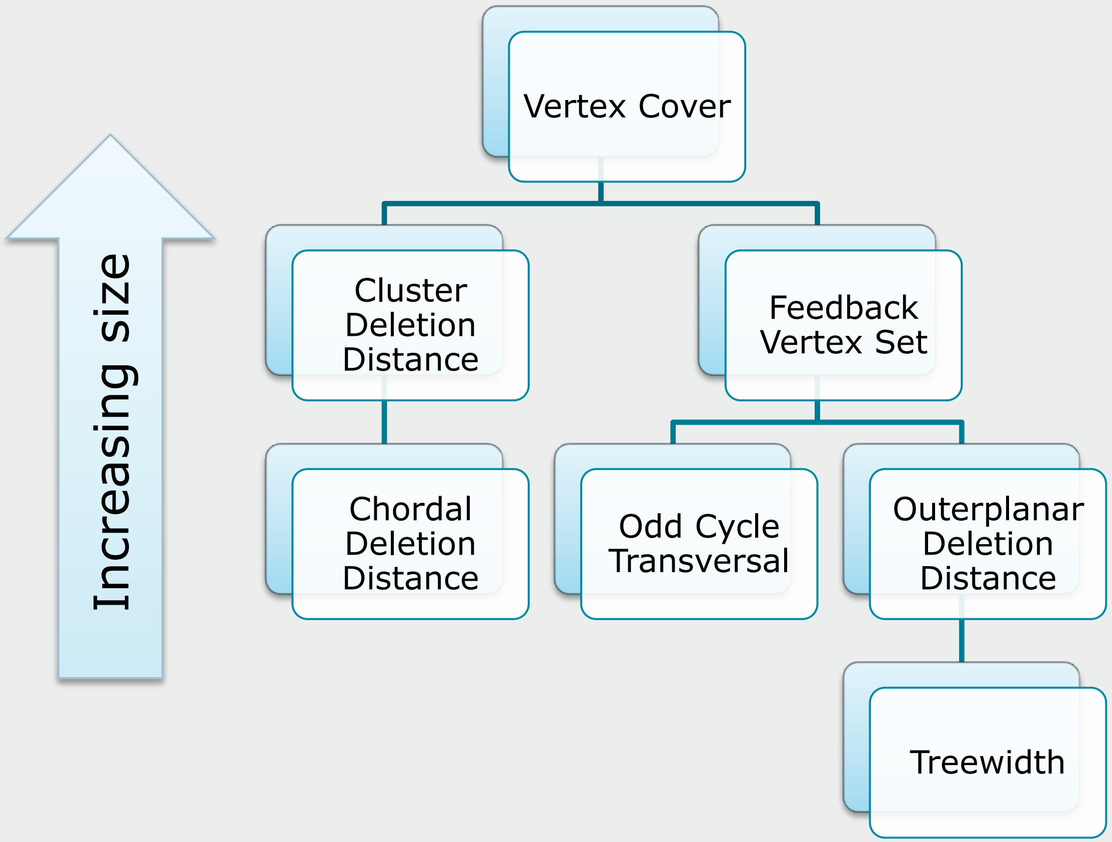
- Two more rules to simplify the trees in F
- Effect of the rules:
  - For each vertex v in X, the amount you have to "pay" in F for using v is at most |X|
  - Similar for pairs of vertices in X
  - But for each tree, some pair makes you pay in that tree

Overview of the reduction process

- Two more rules to simplify the trees in F
- Effect of the rules:
  - For each vertex v in X, the amount you have to "pay" in F for using v is at most |X|
  - Similar for pairs of vertices in X
  - But for each tree, some pair makes you pay in that tree
- Long proof shows that |F| is $O(|X|^3)$ after reduction
  - Size of vertex set is $|X| + O(|X|^3)$

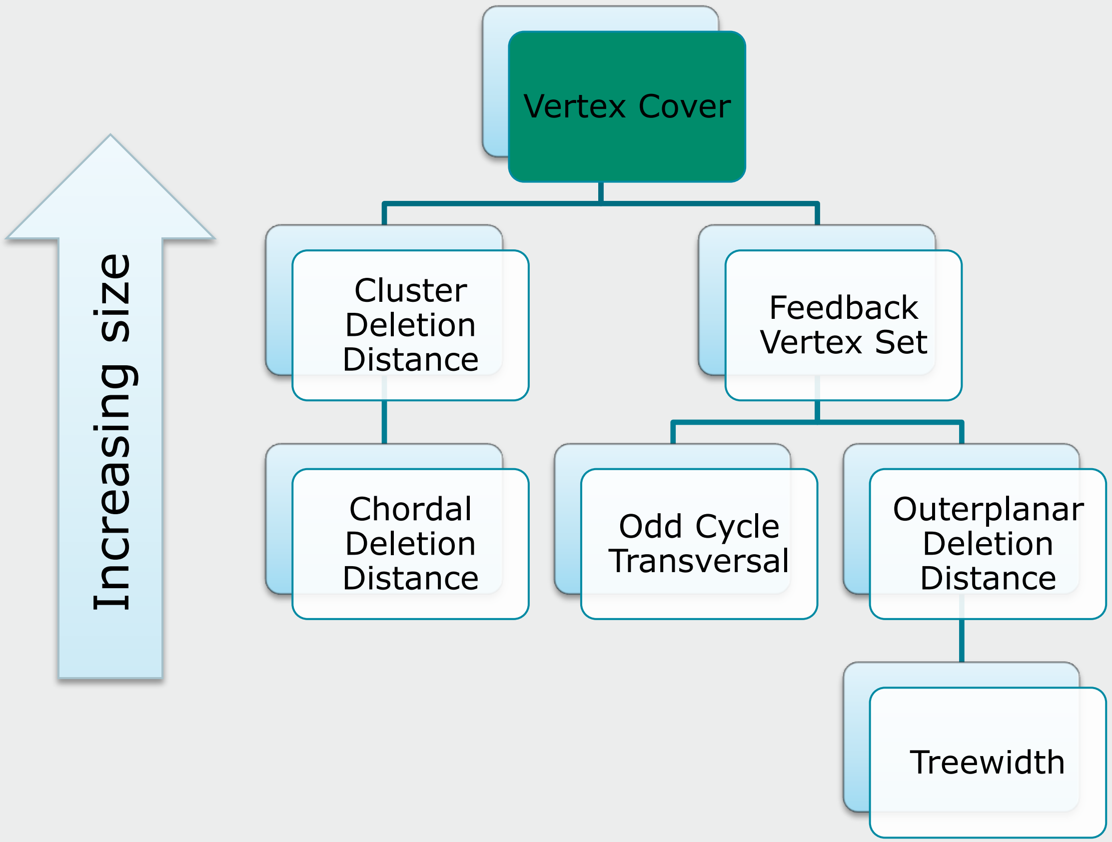
CONCLUSION AND DISCUSSION

Kernelizability of (Unweighted) Vertex Cover



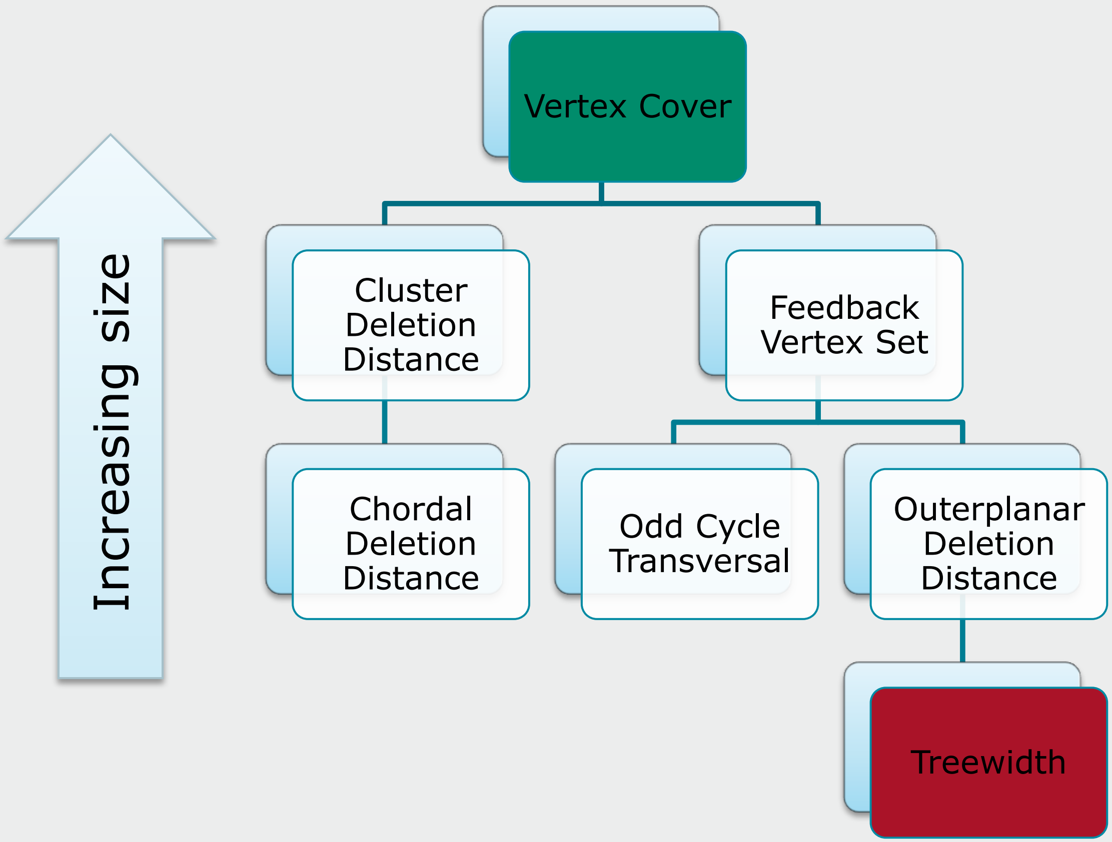
All parameterizations are fixed-parameter tractable

Kernelizability of (Unweighted) Vertex Cover



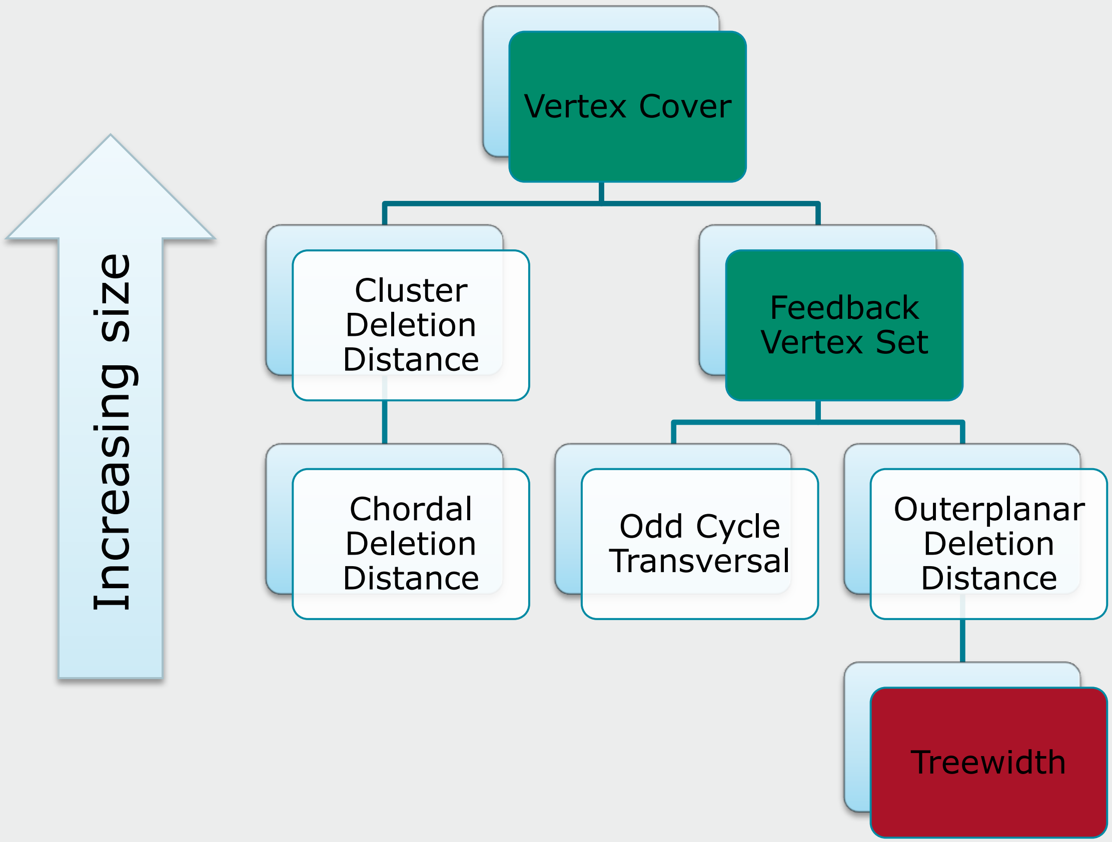
All parameterizations are fixed-parameter tractable

Kernelizability of (Unweighted) Vertex Cover



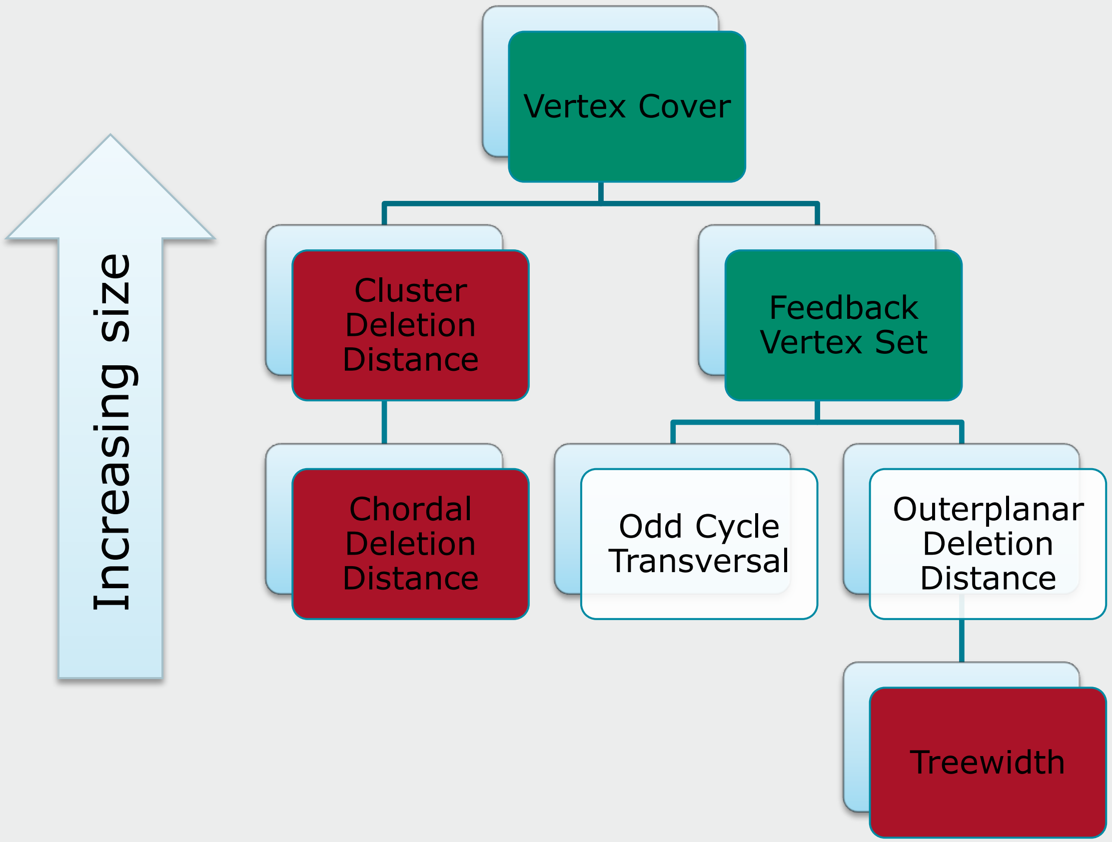
All parameterizations are fixed-parameter tractable

Kernelizability of (Unweighted) Vertex Cover



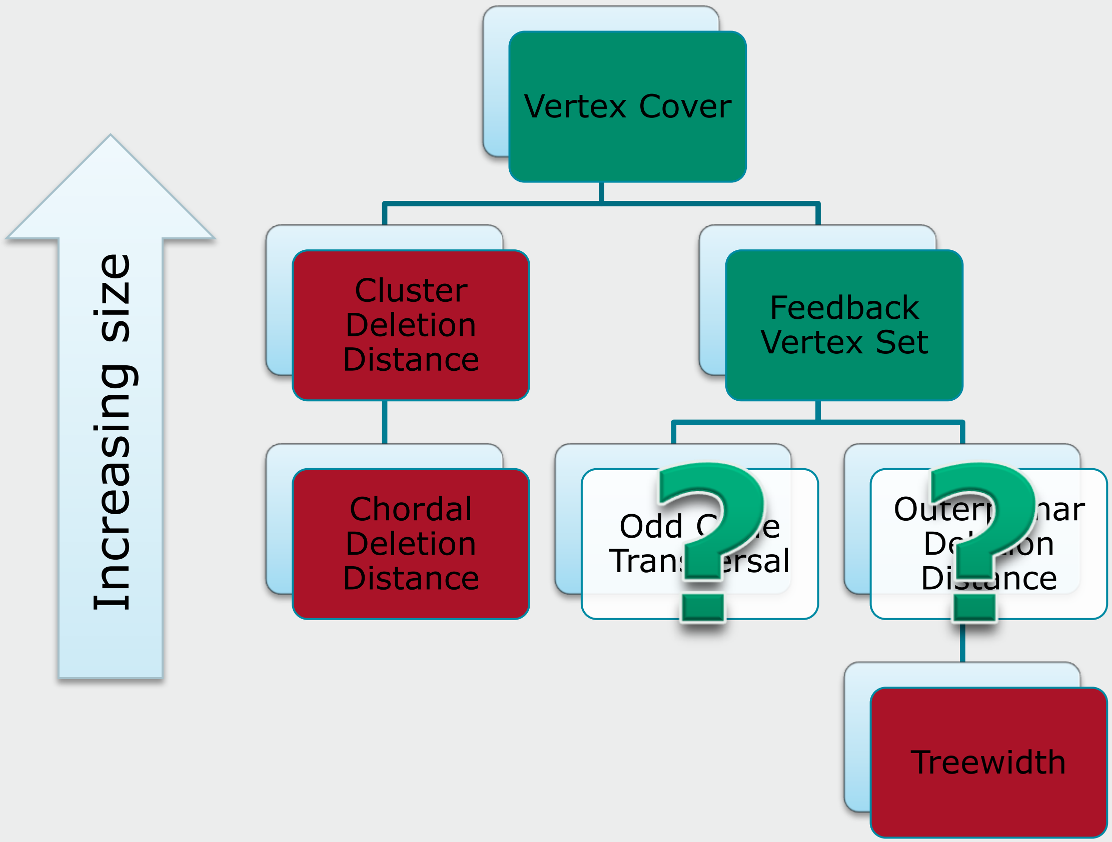
All parameterizations are fixed-parameter tractable

Kernelizability of (Unweighted) Vertex Cover



All parameterizations are fixed-parameter tractable

Kernelizability of (Unweighted) Vertex Cover



All parameterizations are fixed-parameter tractable

Conclusion

- We have studied data reduction for Vertex Cover using a “refined” parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices

Conclusion

- We have studied data reduction for Vertex Cover using a "refined" parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices
- Usage of vertex weights affects kernelizability

Conclusion

- We have studied data reduction for Vertex Cover using a "refined" parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices
- Usage of vertex weights affects kernelizability
  - No polynomial kernel for weighted problem parameterized by VC-size (unless…)

Conclusion

- We have studied data reduction for Vertex Cover using a "refined" parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices
- Usage of vertex weights affects kernelizability
  - No polynomial kernel for weighted problem parameterized by VC-size (unless…)
- Hierarchy of parameters to explore

Conclusion

- We have studied data reduction for Vertex Cover using a "refined" parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices
- Usage of vertex weights affects kernelizability
  - No polynomial kernel for weighted problem parameterized by VC-size (unless…)
- Hierarchy of parameters to explore
- Open problems:
  - Deletion distance to bipartite/outerplanar graphs
  - Improve the degree of the polynomial: cubic to quadratic?

Conclusion

- We have studied data reduction for Vertex Cover using a "refined" parameter: Feedback Vertex Number
  - Kernel with $O(|X|^3)$ vertices
- Usage of vertex weights affects kernelizability
  - No polynomial kernel for weighted problem parameterized by VC-size (unless…)
- Hierarchy of parameters to explore
- Open problems:
  - Deletion distance to bipartite/outerplanar graphs
  - Improve the degree of the polynomial: cubic to quadratic?

Thank you!