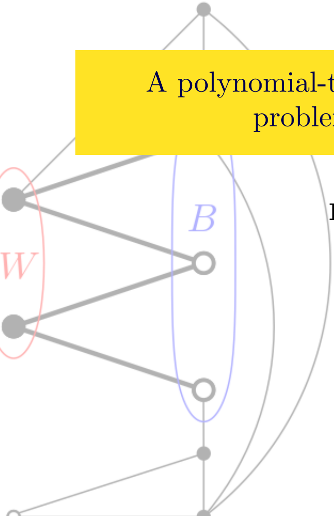
A polynomial-time algorithm for the independent set problem in $\{P_{10}, C_4, C_6\}$-free graphs



Edin Husić and Martin Milanič

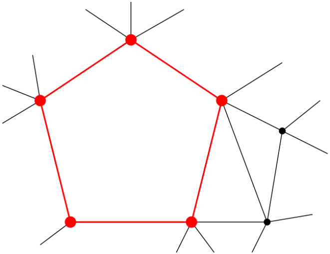
June, 2019

Hereditary classes of graphs, Independent Set

Graphs: finite, simple, undirected.
G is H-free if G does not contain H as *induced* subgraph.
G is $\mathcal{F}$-free if G is H-free for all $H \in \mathcal{F}$.

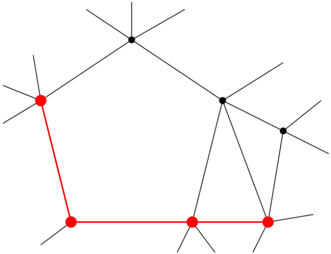
Hereditary classes of graphs, Independent Set

Graphs: finite, simple, undirected.
G is H-free if G does not contain H as *induced* subgraph.
G is $\mathcal{F}$-free if G is H-free for all $H \in \mathcal{F}$.
Cycle C_k (hole if $k \geq 4$)

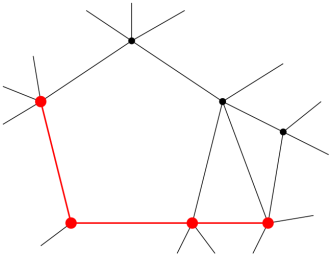


Hereditary classes of graphs, Independent Set

Graphs: finite, simple, undirected.
G is H-free if G does not contain H as *induced* subgraph.
G is $\mathcal{F}$-free if G is H-free for all $H \in \mathcal{F}$.
Cycle C_k (hole if $k \geq 4$), path P_k.



Graphs: finite, simple, undirected.
G is H-free if G does not contain H as *induced* subgraph.
G is $\mathcal{F}$-free if G is H-free for all $H \in \mathcal{F}$.
Cycle C_k (hole if $k \geq 4$), path P_k.



Independent set:

Input:	Graph G.
Task:	Find $\alpha(G)$.

Known hardness results:

Independent Set is NP-hard for:

- Planar graphs of maximum degree 3, triangle-free graphs.
- C_4-free graphs

Known hardness results:

Independent Set is NP-hard for:

- Planar graphs of maximum degree 3, triangle-free graphs.
- C_4-free graphs, or any class of graphs defined by finitely many forbidden induced cycles, i.e., $\{C_{i_1}, \ldots, C_{i_l}\}$-free graphs.

Known hardness results:

Independent Set is NP-hard for:

- Planar graphs of maximum degree 3, triangle-free graphs.
- C_4-free graphs, or any class of graphs defined by finitely many forbidden induced cycles, i.e., $\{C_{i_1}, \ldots, C_{i_l}\}$-free graphs.
- H-free graphs, unless every component of H is a path or subdivision of $K_{1,3}$ (the claw).

Independent Set is polynomial-time solvable for:

- chordal graphs: $\{C_4, C_5, C_6, C_7, \dots\}$-free [Rose, Tarjan, Lueker],
- perfect graphs: $\{C_5, \overline{C_5}, C_7, \overline{C_7}, \dots\}$-free [ellipsoid method, G-L-S], and
- P_6-free graphs [Grzesik, Klimošová, Pilipczuk^2].

Independent Set is polynomial-time solvable for:

- chordal graphs: $\{C_4, C_5, C_6, C_7, \dots\}$-free [Rose, Tarjan, Lueker],
- perfect graphs: $\{C_5, \overline{C_5}, C_7, \overline{C_7}, \dots\}$-free [ellipsoid method, G-L-S], and
- P_6-free graphs [Grzesik, Klimošová, Pilipczuk^2].

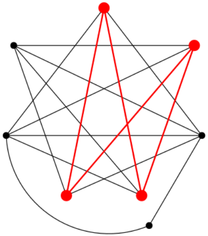
The complexity of the problem is **open** for:

- long-hole-free graphs: $\{C_5, C_6, C_7, \dots\}$-free,
- even-hole-free graphs: $\{C_4, C_6, C_8, \dots\}$-free (β-perfect), and
- P_k-free graphs when $k \geq 7$.

Motivation for even hole-free graphs

Proposition

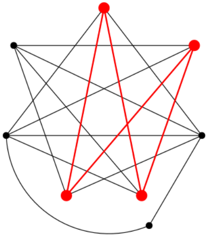
If G is C_4-free then G is $\{\overline{C_6}, \overline{C_7}, \overline{C_8}, \ldots\}$-free.



Motivation for even hole-free graphs

Proposition

If G is C_4-free then G is $\{\overline{C_6}, \overline{C_7}, \overline{C_8}, \dots\}$-free.

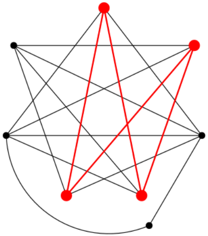


- G is ehf then G is $\{C_4, C_6, \overline{C_6}, C_8, \overline{C_8}, \dots\}$-free.
  (Perfect $= \{C_5, C_7, \overline{C_7}, C_9, \overline{C_9} \dots\}$-free.)

Motivation for even hole-free graphs

Proposition

If G is C_4-free then G is $\{\overline{C_6}, \overline{C_7}, \overline{C_8}, \dots\}$-free.



- G is ehf then G is $\{C_4, C_6, \overline{C_6}, C_8, \overline{C_8}, \dots\}$-free.
  (Perfect $= \{C_5, C_7, \overline{C_7}, C_9, \overline{C_9} \dots\}$-free.)
- even-hole-free graphs are odd signable (balanceable matrices)

Our result

Corollary

INDEPENDENT SET *is polynomial for* $\{$*even-hole*, $P_{10}\}$*-free graphs.*

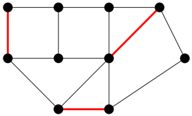
Our result

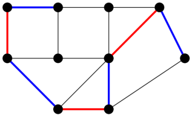
Theorem

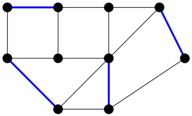
INDEPENDENT SET *is polynomial for* $\{P_{10}, C_4, C_6\}$*-free graphs.*

Corollary

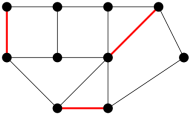
INDEPENDENT SET *is polynomial for* $\{$*even-hole,* $P_{10}\}$*-free graphs.*

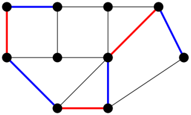


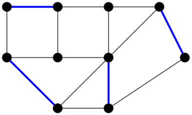




Augmenting Graphs



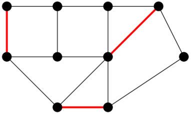


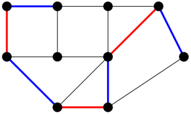


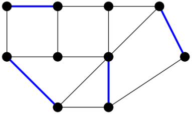
Theorem (Berge)

A matching is maximum $\iff$ it has no augmenting path.

Augmenting Graphs

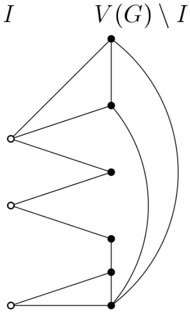


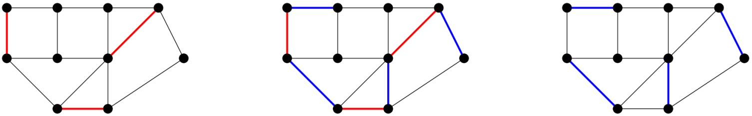




Theorem (Berge)

A matching is maximum $\iff$ *it has no augmenting path.*





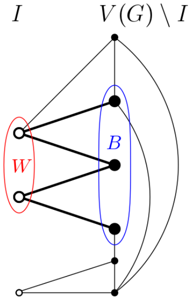
Theorem (Berge)

A matching is maximum $\iff$ *it has no augmenting path.*

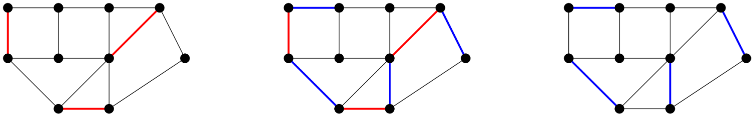
Definition

An induced bipartite graph $H = (W, B; E)$ in G is *augmenting* if

- $W \subseteq I$,
- $B \subseteq V(G) \setminus I$,
- $|W| < |B|$ and
- $N(B) \cap I \subseteq W$.



Augmenting Graphs



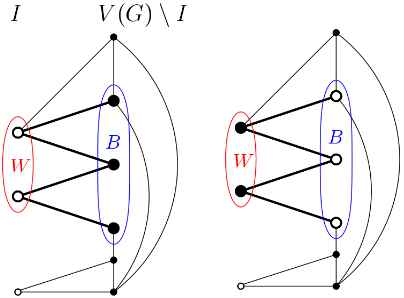
Theorem (Berge)

A matching is maximum $\iff$ *it has no augmenting path.*

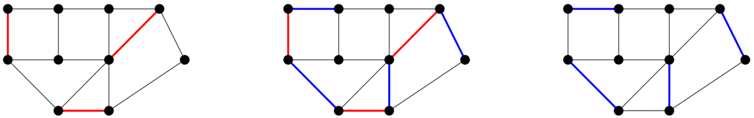
Definition

An induced bipartite graph $H = (W, B; E)$ in G is *augmenting* if

- $W \subseteq I$,
- $B \subseteq V(G) \setminus I$,
- $|W| < |B|$ and
- $N(B) \cap I \subseteq W$.

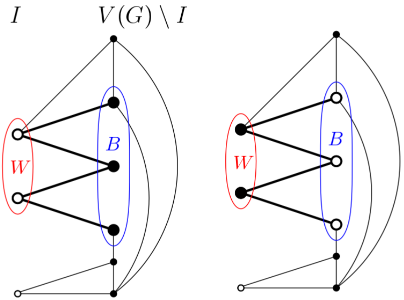


Augmenting Graphs



Theorem (Berge)

A matching is maximum $\iff$ *it has no augmenting path.*



Definition

An induced bipartite graph $H = (W, B; E)$ in G is *augmenting* if

- $W \subseteq I$,
- $B \subseteq V(G) \setminus I$,
- $|W| < |B|$ and
- $N(B) \cap I \subseteq W$..

Theorem (Sbihi, Minty 1980)

An independent set I is maximum $\iff$ I admits no augmenting graph.

Theorem (Sbihi, Minty 1980)

An independent set I is maximum $\iff$ I admits no augmenting graph.

An algorithm for finding a maximum independent set:

Input: A graph G and an independent set I.

1. Check if there I admits augmenting graph:
2. If YES, then augment and repeat.
3. If NO, then output I.

Theorem (Sbihi, Minty 1980)

An independent set I is maximum $\iff$ I admits no augmenting graph.

An algorithm for finding a maximum independent set:

Input: A graph G and an independent set I.

1. Check if there I admits augmenting graph:
2. If YES, then augment and repeat.
3. If NO, then output I.

- $O(n)\cdot$ (time to check if there exists an augmenting graph)
- "Is there an augmenting graph?" is NP-hard !

Theorem (Sbihi, Minty 1980)

An independent set I is maximum $\iff$ I admits no augmenting graph.

An algorithm for finding a maximum independent set:

Input: A graph G and an independent set I.

1. Check if there I admits augmenting graph:
2. If YES, then augment and repeat.
3. If NO, then output I.

- $O(n)\cdot$ (time to check if there exists an augmenting graph)
- "Is there an augmenting graph?" is NP-hard !

But, in special cases the method leads to a polynomial-time algorithm.
It suffices to consider only minimal augmenting graphs: connected, $|W| + 1 = |B|$, degree conditions.

Outline

For a polytime algorithm:

1. Characterize minimal augmenting graphs in the class.
2. Detect if a minimal augmenting graph (of each type) exists.

Outline

For a polytime algorithm:

1. Characterize minimal augmenting graphs in the class.
2. Detect if a minimal augmenting graph (of each type) exists.

We present:

i) polytime algorithm for the problem in $\{P_9, C_4, C_6\}$-free graphs, and

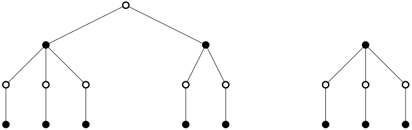
ii) polytime algorithm for the problem in $\{P_{10}, C_4, C_6\}$-free graphs.

Characterize minimal augmenting $\{P_9, C_4, C_6\}$ graphs

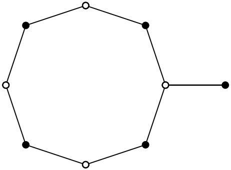
A $\{P_9, C_4, C_6\}$-free augmenting graph is a tree or contains C_8.

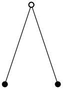
Characterize minimal augmenting $\{P_9, C_4, C_6\}$ graphs

A $\{P_9, C_4, C_6\}$-free augmenting graph is a tree or contains C_8.
The minimal ones are:



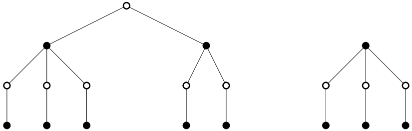
Trees of type $T_{r,s}$ and T_s.



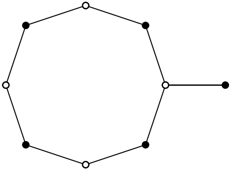


Augmenting P_3.

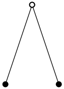
A $\{P_9, C_4, C_6\}$-free augmenting graph is a tree or contains C_8.
The minimal ones are:



Trees of type $T_{r,s}$ and T_s.



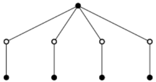
✓



Augmenting P_3. ✓

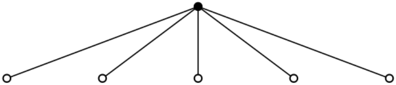
Lemma

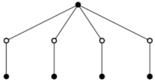
We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.

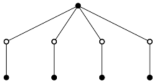


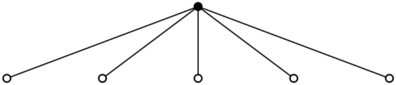


Fix the root x.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.

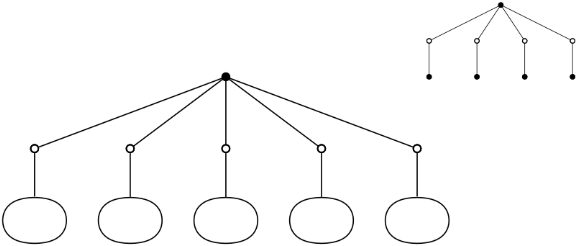




Fix the root x. By definition $N(x) \cap I \subseteq T$.

Lemma

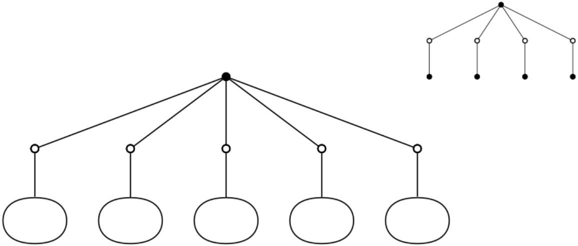
We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$

Lemma

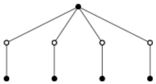
We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.

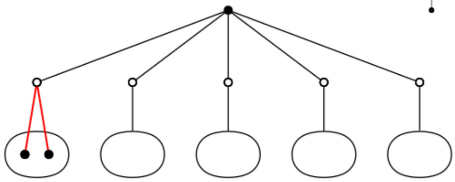


Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.

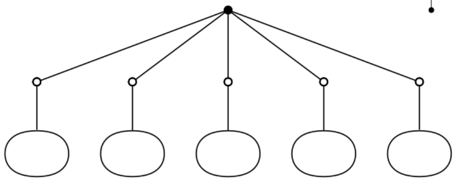


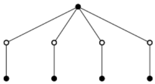


Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.

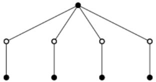


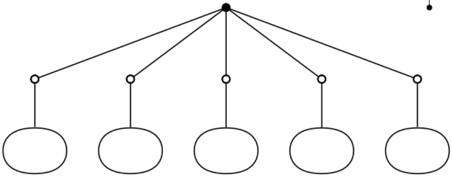


Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



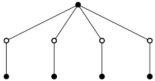


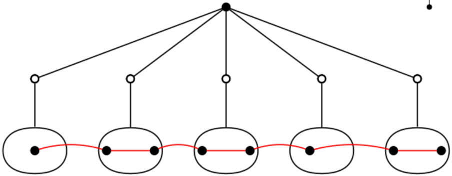
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



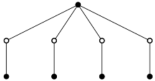


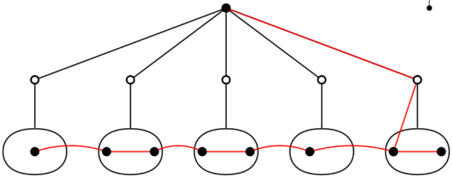
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



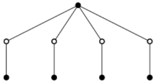


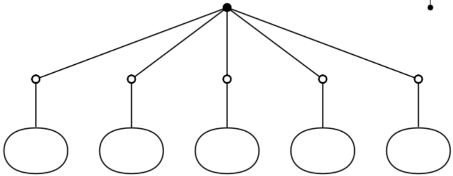
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.



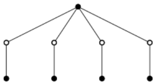


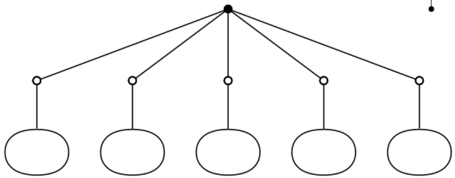
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.





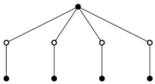
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

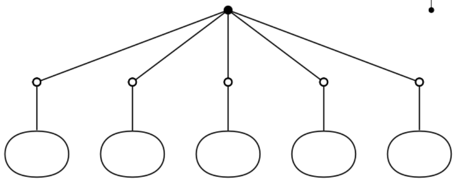
Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

Find a maximum independent set in the graph induced by the cliques.
Polynomial since Independent Set is polynomial in $\{P_8, C_4\}$-free graphs.
[Gerber, Hertz, Lozin '03]

Lemma

We can check in polytime if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_9, C_4, C_6\}$-free graph.





Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \notin I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_8, C_4, C_6\}$-free.

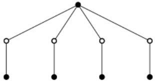
Find a maximum independent set in the graph induced by the cliques.
Polynomial since Independent Set is polynomial in $\{P_8, C_4\}$-free graphs.
[Gerber, Hertz, Lozin '03]

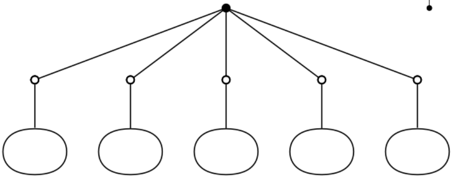
$T_{r,s}$ follows similarly. (Easier)

More general

Lemma

If Independent Set *is polytime solvable for* $\{P_{k-1}\} \cup \mathcal{F}$*-free graphs then we can check (in polytime) if there exists an augmenting tree of type* T_s *or* $T_{r,s}$ *in a* $\{P_k\} \cup \mathcal{F}$*-free graph.*





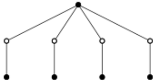
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

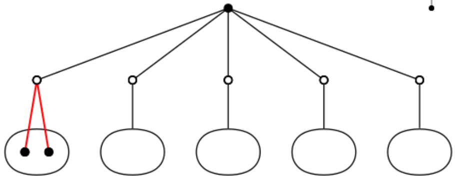
Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.

Lemma

If Independent Set *is polytime solvable for* $\{P_{k-1}\} \cup \mathcal{F}$*-free graphs then we can check (in polytime) if there exists an augmenting tree of type* T_s *or* $T_{r,s}$ *in a* $\{P_k\} \cup \mathcal{F}$*-free graph.*





Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

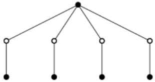
Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

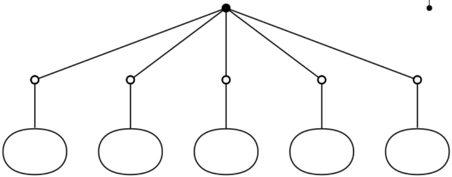
Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.

More general

Lemma

If Independent Set *is polytime solvable for $\{P_{k-1}\} \cup \mathcal{F}$-free graphs then we can check (in polytime) if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_k\} \cup \mathcal{F}$-free graph.*





Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

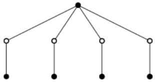
Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

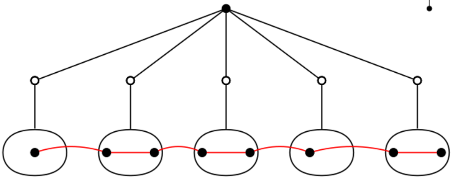
Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.

More general

Lemma

If Independent Set *is polytime solvable for* $\{P_{k-1}\} \cup \mathcal{F}$*-free graphs then we can check (in polytime) if there exists an augmenting tree of type* T_s *or* $T_{r,s}$ *in a* $\{P_k\} \cup \mathcal{F}$*-free graph.*





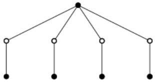
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

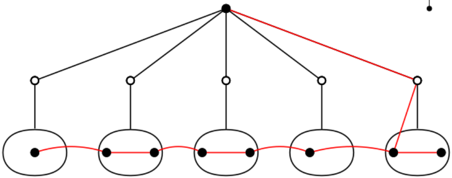
Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.

Lemma

If Independent Set *is polytime solvable for* $\{P_{k-1}\} \cup \mathcal{F}$*-free graphs then we can check (in polytime) if there exists an augmenting tree of type* T_s *or* $T_{r,s}$ *in a* $\{P_k\} \cup \mathcal{F}$*-free graph.*





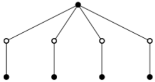
Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

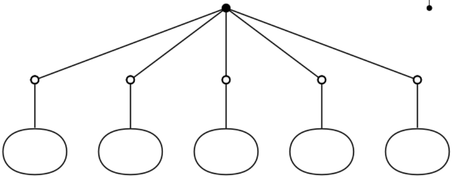
Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.

Lemma

If Independent Set *is polytime solvable for* $\{P_{k-1}\} \cup \mathcal{F}$*-free graphs then we can check (in polytime) if there exists an augmenting tree of type* T_s *or* $T_{r,s}$ *in a* $\{P_k\} \cup \mathcal{F}$*-free graph.*

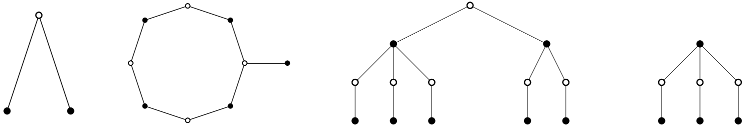




Fix the root x. By definition $N(x) \cap I \subseteq T$.
$K(w) = \{v \in N(G) \setminus I : N(v) \cap I = \{w\}\}$ is a clique.

Claim: Graph induced by $K(w)$'s is $\{P_{k-1}\} \cup \mathcal{F}$-free.

Find the maximum independent set in the graph induced by $K(w)$'s.
Polynomial by assumption.



Theorem

INDEPENDENT SET *is polynomially solvable in class of* $\{P_9, C_4, C_6\}$*-free graphs.*

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

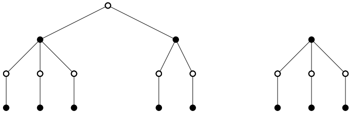
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}.

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

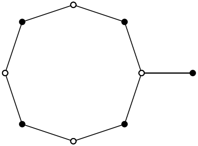
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

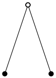
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:



Trees of type $T_{r,s}$ and T_s.



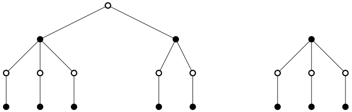
$\checkmark$



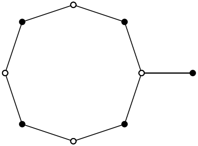
P_3. $\checkmark$

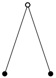
Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:

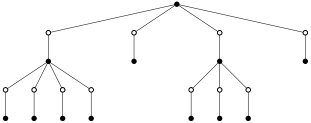


Trees of type $T_{r,s}$ and T_s.

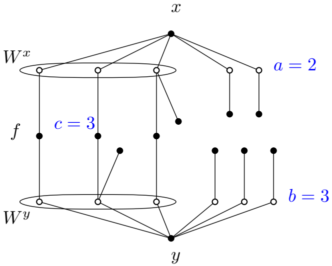




P_3.



"Special" trees of containing P_9.

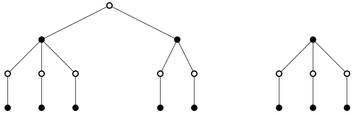


(a,b,c)-crab.

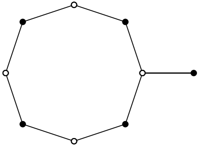
+ Two particular graphs.

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

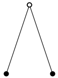
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:



Trees of type $T_{r,s}$ and T_s.



$\checkmark$



P_3. $\checkmark$

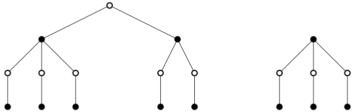
Lemma

If Independent Set *is polytime solvable for $\{P_{k-1}\} \cup \mathcal{F}$-free graphs then we can check (in polytime) if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_k\} \cup \mathcal{F}$-free graph.*

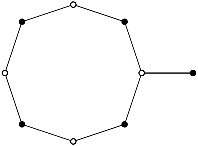
Set $\mathcal{F} = \{C_4, C_6\}$ and $k = 10$.

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

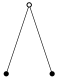
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:



Trees of type $T_{r,s}$ and T_s. ✓



✓



P_3.✓

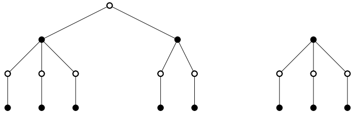
Lemma

If Independent Set *is polytime solvable for $\{P_{k-1}\} \cup \mathcal{F}$-free graphs then we can check (in polytime) if there exists an augmenting tree of type T_s or $T_{r,s}$ in a $\{P_k\} \cup \mathcal{F}$-free graph.*

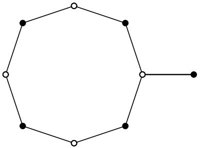
Set $\mathcal{F} = \{C_4, C_6\}$ and $k = 10$.

Characterize minimal augmenting $\{P_{10}, C_4, C_6\}$ graphs

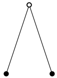
A $\{P_{10}, C_4, C_6\}$-free augmenting graph is either a tree, or contains C_8, or contains C_{10}. The minimal augmenting ones are:



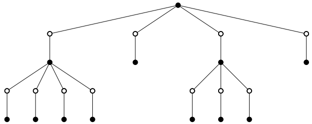
Trees of type $T_{r,s}$ and T_s. ✓



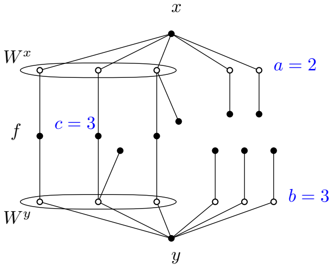
✓



P_3. ✓



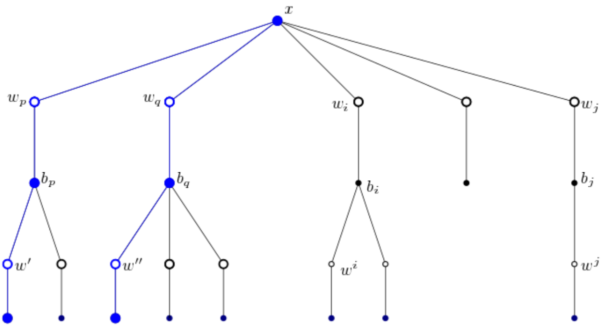
"Special" trees of containing P_9.



(a,b,c)-crab. ✓

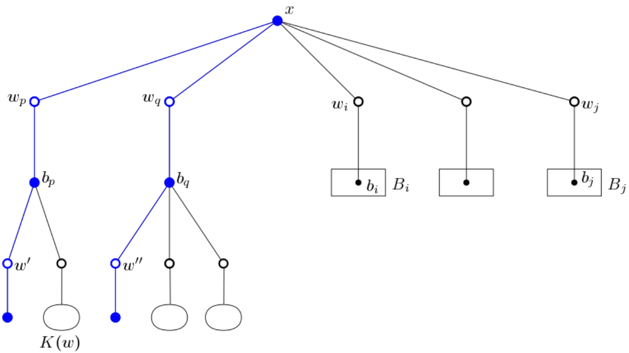
+ Two particular graphs. ✓

Detecting "special" trees containing P_9. (Idea)



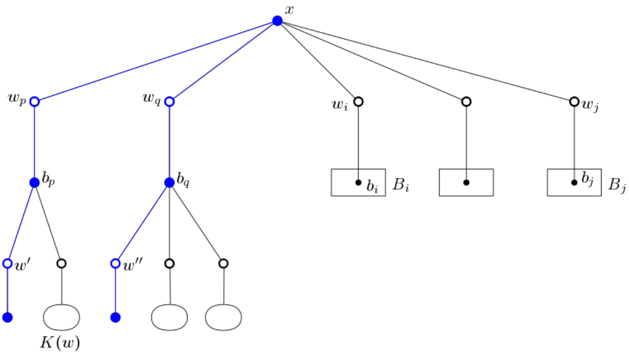
Goal: Find an augmenting tree of depth 4 containing a fixed P_9.

Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.

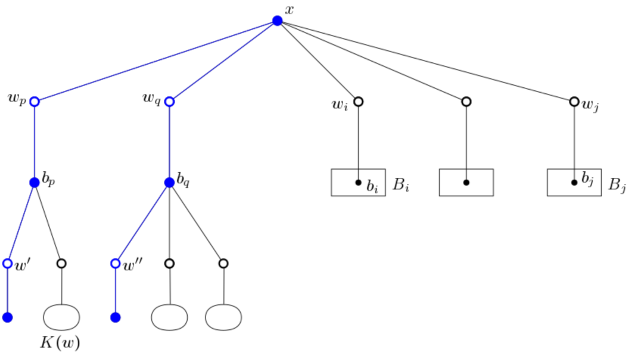
Detecting “special” trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w.

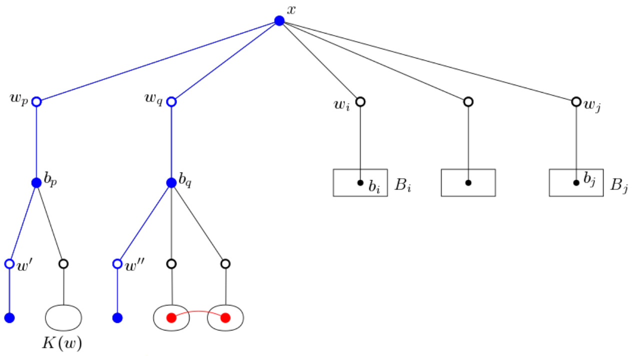
Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w.

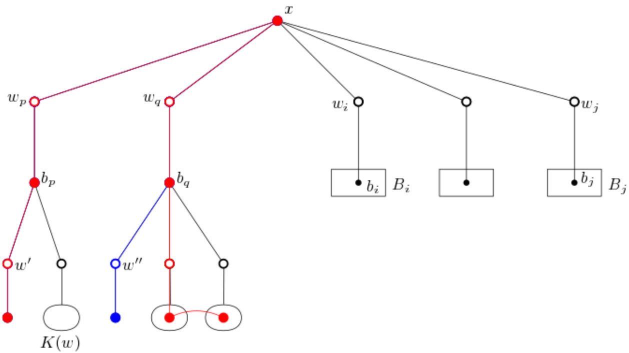
Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

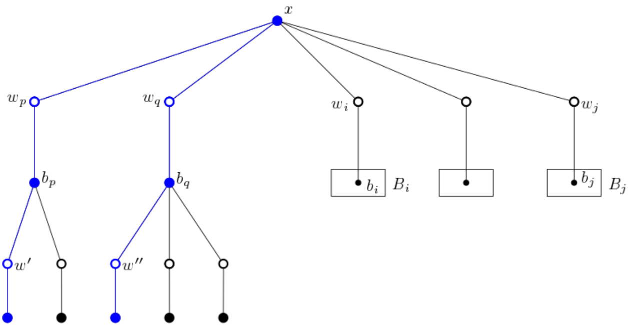
- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w.

Detecting “special” trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

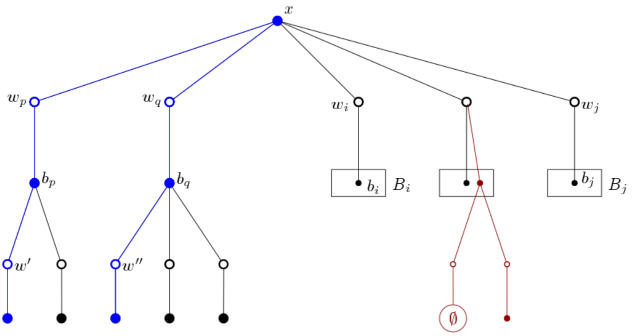
- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w.



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓

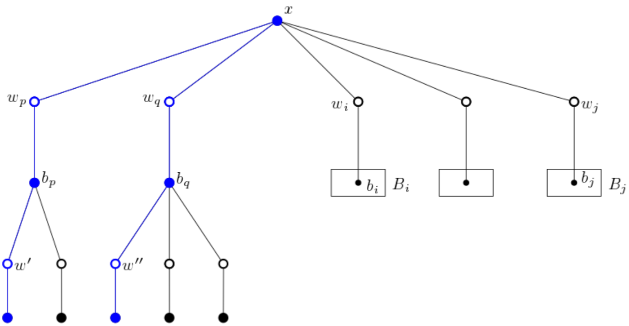
Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$.

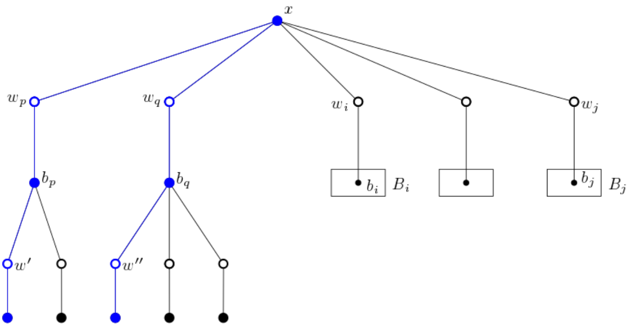
Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$.

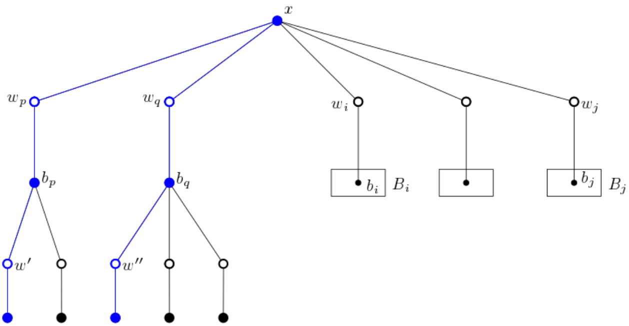
Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$. Not necessarily a clique.

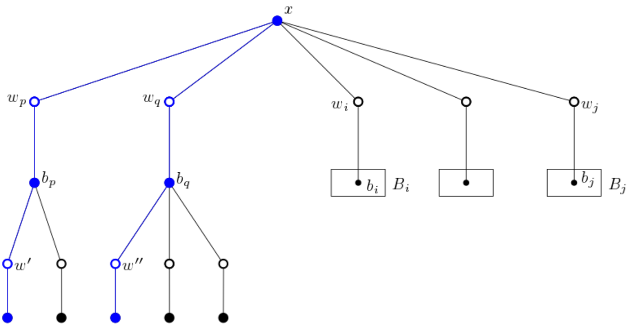
Detecting “special” trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$. Not necessarily a clique.
  Take $G[\cup_{i \neq p,q} B_i]$ and fill every B_i to a clique.

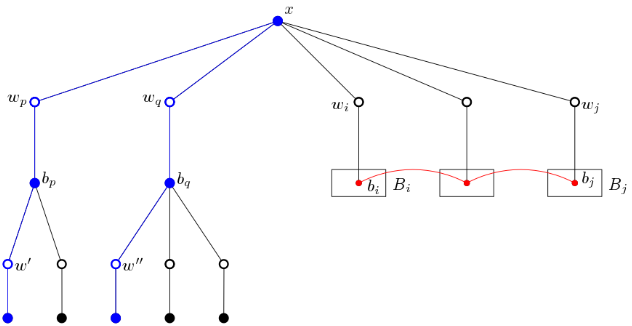
Detecting “special” trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$. Not necessarily a clique. Take $G[\cup_{i \neq p,q} B_i]$ and fill every B_i to a clique. We obtain a perfect graph.

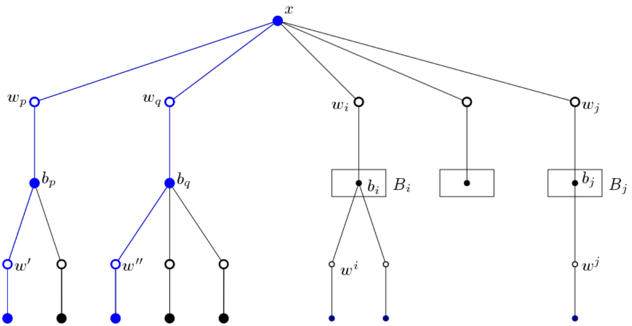
Detecting “special” trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$. Not necessarily a clique.
  Take $G[\cup_{i \neq p,q} B_i]$ and fill every B_i to a clique.
  We obtain a perfect graph.

Detecting "special" trees containing P_9. (Idea)



Goal: Find an augmenting tree of depth 4 containing a fixed P_9.
By definition, the vertices w', w'', and w_i are in the augmenting graph.
Consider all potential vertices: $K(w)$'s or B_i's.

- $K(w)$ is a clique for every w. Moreover, $K(w)$ is adjacent only to w. ✓
- What about B_i's? Clean $\cup_{i \neq p,q} B_i$. Not necessarily a clique.
  Take $G[\cup_{i \neq p,q} B_i]$ and fill every B_i to a clique.
  We obtain a perfect graph.

Thank you for your attention!