Hereditary efficiently dominatable graphs

Martin Milanič

UP FAMNIT and UP PINT, University of Primorska

Raziskovalni matematični seminar, UP FAMNIT, 14. november 2011

Efficient dominating sets

$G = (V, E)$: finite, simple, undirected graph

a vertex $v \in V$ dominates itself and all its neighbors

A set $D \subseteq V$ is an efficient dominating set in G if every vertex in V is dominated by exactly one vertex in D:

$$|N[v] \cap D| = 1$$

for all $v \in V$.

- Biggs 1973 (perfect codes in distance-transitive graphs)

(1-)perfect code / perfect (independent) dominating set

Efficient dominating sets

$G = (V, E)$: finite, simple, undirected graph

a vertex $v \in V$ dominates itself and all its neighbors

A set $D \subseteq V$ is an efficient dominating set in G if every vertex in V is dominated by exactly one vertex in D:

$$|N[v] \cap D| = 1$$

for all $v \in V$.

- Biggs 1973 (perfect codes in distance-transitive graphs)

(1-)perfect code / perfect (independent) dominating set

Efficient dominating sets

$G = (V, E)$: finite, simple, undirected graph

a vertex $v \in V$ dominates itself and all its neighbors

A set $D \subseteq V$ is an efficient dominating set in G if every vertex in V is dominated by exactly one vertex in D:

$$|N[v] \cap D| = 1$$

for all $v \in V$.

- Biggs 1973 (perfect codes in distance-transitive graphs)

(1-)perfect code / perfect (independent) dominating set

Efficient dominating sets

$G = (V, E)$: finite, simple, undirected graph

a vertex $v \in V$ dominates itself and all its neighbors

A set $D \subseteq V$ is an efficient dominating set in G if every vertex in V is dominated by exactly one vertex in D:

$$|N[v] \cap D| = 1$$

for all $v \in V$.

- Biggs 1973 (perfect codes in distance-transitive graphs)

(1-)perfect code / perfect (independent) dominating set

Efficient dominating sets

Equivalently:

- D is an independent set of vertices such that
- every vertex outside D has a unique neighbor in D.

Efficient dominating sets

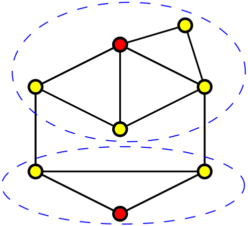
Equivalently:

- D is an independent set of vertices such that
- every vertex outside D has a unique neighbor in D.

Equivalently:

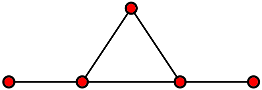
$$\{N[v] \mid v \in D\}$$

forms a partition of V.

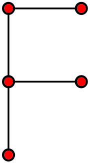


Examples

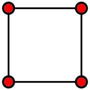
Some small graphs do not contain any efficient dominating sets:



bull



fork

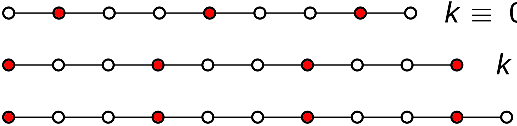


C_4

Paths and cycles

All paths contain efficient dominating sets:

P_k



$k \equiv 0 \mod 3$

$k \equiv 1 \mod 3$

$k \equiv 2 \mod 3$

C_k contains an efficient dominating set $\iff k \equiv 0 \mod 3$.

Complexity

G is efficiently dominatable if it contains an efficient dominating set.

All efficient dominating sets of G are of the same size:

- every efficient dominating set is a minimum dominating set.

Determining whether G is efficiently dominatable is NP-complete, even for:

- planar cubic graphs,
- planar bipartite graphs,
- chordal bipartite graphs,
- chordal graphs,
- line graphs of planar bipartite graphs of max degree three.

Complexity

G is efficiently dominatable if it contains an efficient dominating set.

All efficient dominating sets of G are of the same size:

- every efficient dominating set is a minimum dominating set.

Determining whether G is efficiently dominatable is NP-complete, even for:

- planar cubic graphs,
- planar bipartite graphs,
- chordal bipartite graphs,
- chordal graphs,
- line graphs of planar bipartite graphs of max degree three.

Complexity

G is efficiently dominatable if it contains an efficient dominating set.

All efficient dominating sets of G are of the same size:

- every efficient dominating set is a minimum dominating set.

Determining whether G is efficiently dominatable is NP-complete, even for:

- planar cubic graphs,
- planar bipartite graphs,
- chordal bipartite graphs,
- chordal graphs,
- line graphs of planar bipartite graphs of max degree three.

Complexity

... but polynomially solvable for:

- trees, interval graphs, series-parallel graphs,
- split graphs, block graphs, circular-arc graphs,
- permutation graphs, trapezoid graphs,
- cocomparability graphs, distance-hereditary graphs,
- AT-free graphs,
- graphs of bounded treewidth or clique-width.

Relation to hereditary classes

The efficiently dominatable graphs do not form a hereditary class:

Hereditary efficiently dominatable graphs

G is hereditary efficiently dominatable (HED) if every induced subgraph of G is efficiently dominatable.

We are interested in:

- characterizations,
- algorithmic aspects.

Hereditary efficiently dominatable graphs

G is hereditary efficiently dominatable (HED) if every induced subgraph of G is efficiently dominatable.

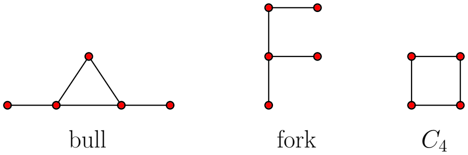
We are interested in:

- characterizations,
- algorithmic aspects.

Hereditary efficiently dominatable graphs

Proposition

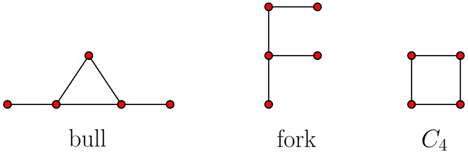
Every HED graph is (bull, fork, C_{3k+1}, C_{3k+2})-free.



Hereditary efficiently dominatable graphs

Proposition

Every HED graph is (bull, fork, C_{3k+1}, C_{3k+2})-free.



The converse holds as well.

To prove this, we first study the structure of (bull, fork, C_4)-free graphs.

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

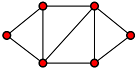
paths and cycles

by applying a sequence of the following operations:

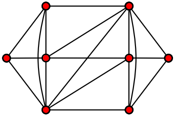
- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

Rafts and semi-rafts

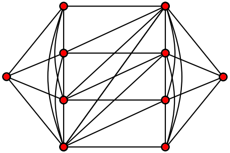
Rafts of order 2, 3 and 4:



R_2



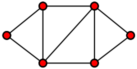
R_3



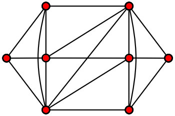
R_4

Rafts and semi-rafts

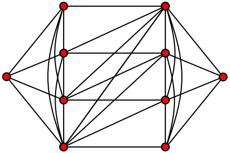
Rafts of order 2, 3 and 4:



R_2

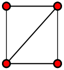


R_3

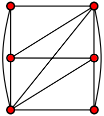


R_4

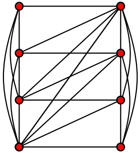
Semi-rafts of order 2, 3 and 4:



S_2

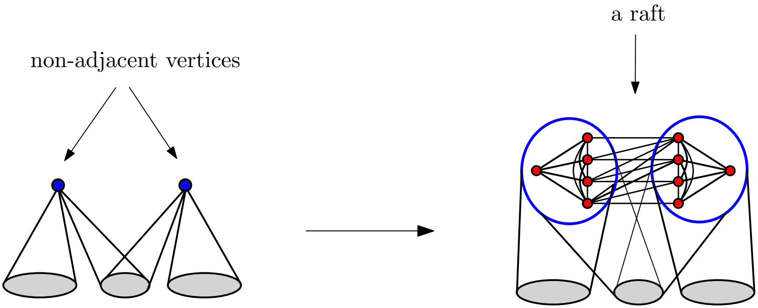


S_3

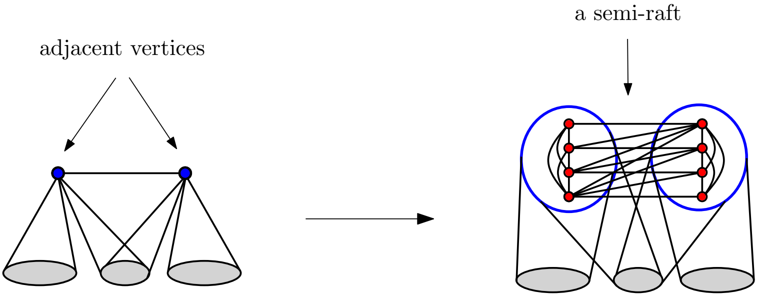


S_4

Raft expansion



Semi-raft expansion



A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

Sketch of proof

G: a minimal counterexample.

Case 1. G **contains an induced cycle of order at least 5**

Easy.

- C: shortest induced cycle of order at least 5
- Analyzing the neighborhood of C shows that $G = C$.

Sketch of proof

G: a minimal counterexample.

Case 1. G contains an induced cycle of order at least 5

Easy.

- C: shortest induced cycle of order at least 5
- Analyzing the neighborhood of C shows that $G = C$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.

- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free,
  therefore it is either disconnected or
  contains a dominating vertex,
  which is impossible by minimality.

- If $k \geq 5$ then analyzing the neighborhood of P shows that
  $G = P$.

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

Case 2. The only possible induced cycle in G is C_3.

$P = P_k$: a longest induced path in G.

- $k \geq 4$ since otherwise G is (P_4, C_4)-free, therefore it is either disconnected or contains a dominating vertex, which is impossible by minimality.
- If $k \geq 5$ then analyzing the neighborhood of P shows that $G = P$.

Sketch of proof

If $k = 4$ then analyzing the neighborhood of P shows that G is an induced subgraph of the following 14-vertex graph:

Sketch of proof

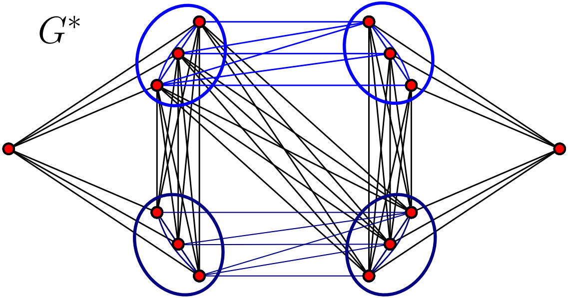
If $k = 4$ then analyzing the neighborhood of P shows that G is an induced subgraph of the following 14-vertex graph:

Sketch of proof

If $k = 4$ then analyzing the neighborhood of P shows that G is an induced subgraph of the following 14-vertex graph:

Sketch of proof

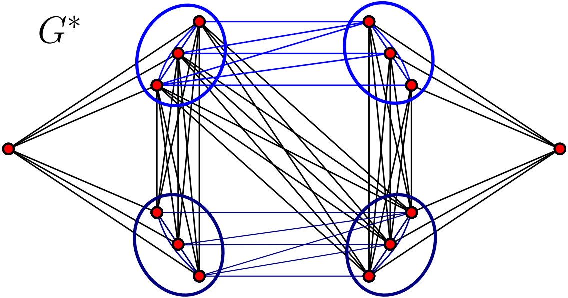
If $k = 4$ then analyzing the neighborhood of P shows that G is an induced subgraph of the following 14-vertex graph:



G^* arises from a double semi-raft expansion applied to raft R_2.

Sketch of proof

If $k = 4$ then analyzing the neighborhood of P shows that G is an induced subgraph of the following 14-vertex graph:



G^* arises from a double semi-raft expansion applied to raft R_2.

A decomposition theorem

Theorem

Let G be a (bull, fork, C_4)-free graph. Then, G can be built from

paths and cycles

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_{3k+1}, C_{3k+2})-free graph. Then, G can be built from

paths and $\{$cycles C_{3k} ; $k \in \mathbb{N}\}$

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

A decomposition theorem

Theorem

Let G be a (bull, fork, C_{3k+1}, C_{3k+2})-free graph. Then, G can be built from

paths and $\{$cycles C_{3k} ; $k \in \mathbb{N}\}$

by applying a sequence of the following operations:

- *disjoint union of two graphs,*
- *duplicating a vertex,*
- *adding a dominating vertex,*
- *raft expansion,*
- *semi-raft expansion.*

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Characterization of HED graphs

The set of efficiently dominatable graphs is closed under each of the operations used in the theorem:

- disjoint union of two graphs,
- duplicating a vertex,
- adding a dominating vertex,
- raft expansion,
- semi-raft expansion.

Corollary

Every (bull, fork, C_{3k+1}, C_{3k+2})-free graph is efficiently dominatable.

Theorem

The class of hereditary efficiently dominatable graphs equals the class of (bull, fork, C_{3k+1}, C_{3k+2})-free graphs.

Finding efficient dominating sets efficiently?

Is there an efficient algorithm

for finding an efficient dominating set

in a given efficiently dominatable graph?

No (unless P = NP).

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given efficiently dominatable graph?

No (unless P = NP).

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given efficiently dominatable graph?

No (unless P = NP).

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given efficiently dominatable graph?

No (unless P = NP).

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given hereditary efficiently dominatable graph?

Yes! We will see two approaches.

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given hereditary efficiently dominatable graph?

Yes! We will see two approaches.

Finding efficient dominating sets efficiently?

Is there an efficient algorithm
for finding an efficient dominating set
in a given hereditary efficiently dominatable graph?

Yes! We will see two approaches.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or $C_4 \to G$ is not HED
- while G is decomposable, decompose $\to$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or $C_{3k+2} \to$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or $C_{3k} \to$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or $C_4 \rightarrow G$ is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or $C_{3k+2} \rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or $C_{3k} \rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

A polynomial-time robust algorithm

Input: a graph G
Output: either an efficient dominating set in G, or a proof that G is not hereditary efficiently dominatable.

Algorithm:

- if G contains an induced bull, fork, or C_4 $\rightarrow$ G is not HED
- while G is decomposable, decompose $\rightarrow$ compute a set $\mathcal{H}$ of indecomposable induced subgraphs of G
- if there exists an $H \in \mathcal{H}$ such that $H = C_{3k+1}$ or C_{3k+2} $\rightarrow$ G is not HED
- otherwise, each $H \in \mathcal{H}$ is either P_k or C_{3k} $\rightarrow$ we can find an ED set in every H; these sets can be mapped to an ED set in G.

Another approach

efficient domination number
= maximum number of vertices that can be efficiently dominated
$= \max\{|D \cup N(D)| \mid D \subseteq V$ independent, every $v \in V \setminus D$ has at most one neighbor in $D\}$

The efficient domination problem:

Given a graph G, compute the efficient domination number of G.

Another approach

efficient domination number
= maximum number of vertices that can be efficiently dominated
= $\max\{|D \cup N(D)| \mid D \subseteq V$ independent, every $v \in V \setminus D$ has at most one neighbor in $D\}$

The efficient domination problem:

Given a graph G, compute the efficient domination number of G.

Another approach

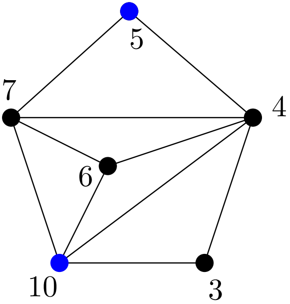
efficient domination number
= maximum number of vertices that can be efficiently dominated
$= \max\{|D \cup N(D)| \mid D \subseteq V$ independent, every $v \in V \setminus D$ has at most one neighbor in $D\}$

The efficient domination problem:

Given a graph G, compute the efficient domination number of G.

The weighted independent set problem

WEIGHTED INDEPENDENT SET (WIS) Problem:
Input: $G = (V, E)$, $w : V \to \mathbb{N}$
Task: Compute $\alpha_w(G)$ = max weight of an independent set.



$$\alpha_w(G) = 15$$

Reduction to the WIS problem

G^2 – square of a graph G:

- $V(G^2) = V(G)$,
- $uv \in E(G^2) \iff d_G(u,v) \leq 2$.

What are the independent sets in G^2?

Observation

Efficient domination number of G =
maximum weight of an independent set in G^2 where

$$w(x) = |N[x]|$$

for all $x \in V(G)$.

Reduction to the WIS problem

G^2 – square of a graph G:

- $V(G^2) = V(G)$,
- $uv \in E(G^2) \iff d_G(u, v) \leq 2$.

What are the independent sets in G^2?

Observation

Efficient domination number of G =
maximum weight of an independent set in G^2 where

$$w(x) = |N[x]|$$

for all $x \in V(G)$.

Reduction to the WIS problem

The efficient domination problem is polynomially solvable in every class of graphs X such that
the WIS problem is polynomially solvable in the class

$$\{G^2 \mid G \in X\}.$$

Theorem

The WIS problem is polynomially solvable for claw-free graphs.

Minty 1980 + Nakamura–Tamura 2001
Oriolo–Pietropaoli–Stauffer 2008
Nobili–Sassano 2010
Faenza–Oriolo–Stauffer 2011

Reduction to the WIS problem

The efficient domination problem is polynomially solvable in every class of graphs X such that
the WIS problem is polynomially solvable in the class

$$\{G^2 \mid G \in X\}.$$

Theorem

The WIS problem is polynomially solvable for claw-free graphs.

Minty 1980 + Nakamura–Tamura 2001
Oriolo–Pietropaoli–Stauffer 2008
Nobili–Sassano 2010
Faenza–Oriolo–Stauffer 2011

Reduction to the WIS problem

The efficient domination problem is polynomially solvable in every class of graphs X such that
the WIS problem is polynomially solvable in the class

$$\{G^2 \mid G \in X\}.$$

Theorem

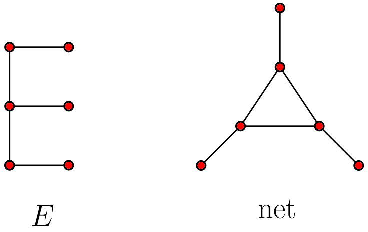
The WIS problem is polynomially solvable for claw-free graphs.

Minty 1980 + Nakamura–Tamura 2001
Oriolo–Pietropaoli–Stauffer 2008
Nobili–Sassano 2010
Faenza–Oriolo–Stauffer 2011

(E, net)-free graphs

Proposition

If G is (E, net)-free then G^2 is claw-free.



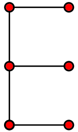
Corollary

The ED number can be computed in polynomial time for (E, net)-free graphs.

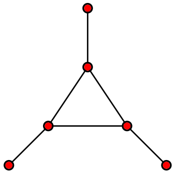
(E, net)-free graphs

Proposition

If G is (E, net)-free then G^2 is claw-free.



E



net

Corollary

The ED number can be computed in polynomial time for (E, net)-free graphs.

More polynomial results

The same approach can be used to show that the efficient domination problem is polynomial for:

- cocomparability graphs,
- interval graphs,
- circular-arc graphs,
- trapezoid graphs,
- strongly chordal graphs,
- AT-free graphs.

All these graph classes are closed under taking squares, and the WIS problem is polynomial on each of them.

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

Summary

- Characterizations of hereditary efficiently dominatable graphs.
- HED graphs can be recognized in polynomial time by:
  (1) expressing their defining property in MSOL,
  (2) using the fact that they are of bounded clique-width,
  (3) applying a theorem of Courcelle-Makowsky-Rotics (2000).
  Is there a more direct polynomial-time algorithm for recognizing hereditary efficiently dominatable graphs?
- What is the complexity of recognizing (C_{3k+1}, C_{3k+2})-free graphs?

The end

Hvala!