Edge Elimination Schemes of Weighted Graph Classes

Martin Milanič
University of Primorska, Koper, Slovenia

Workshop on Graph Modification
[algorithms, experiments and new problems]

23 – 24 January 2020, Bergen, Norway

Joint work (in progress) with:

Jesse Beisegel, BTU Cottbus-Senftenberg, Germany

Nina Chiarelli, University of Primorska, Koper, Slovenia

Ekkehard Köhler, BTU Cottbus-Senftenberg, Germany

Matjaž Krnc, University of Primorska, Koper, Slovenia

Nevena Pivač, University of Primorska, Koper, Slovenia

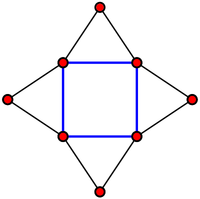
Robert Scheffler, BTU Cottbus-Senftenberg, Germany

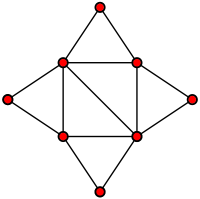
Martin Strehler, BTU Cottbus-Senftenberg, Germany

Background and motivation

Chordal Graphs

A graph G is chordal if every cycle in G of length at least four has a chord.





Chordal graphs are well-known to possess many good structural and algorithmic properties.

1965, Fulkerson and Gross:

- A graph $G = (V, E)$ is chordal if and only if it has a **perfect elimination ordering.**

A linear ordering $<$ of V such that
for all $x < y < z$:

$$xy \in E \text{ and } xz \in E \quad \Longrightarrow \quad yz \in E$$

1965, Fulkerson and Gross:

- A graph $G = (V, E)$ is chordal if and only if it has a **perfect elimination ordering.**

Or, in terms of the adjacency matrix A of G:
for all $x < y < z$:

$$A_{yz} \geq \min\{A_{xy}, A_{xz}\}$$

2017, Laurent and Tanigawa:

- extended to weighted graphs the notion of perfect elimination ordering.

A **perfect elimination ordering** of an edge-weighted graph $G = (V, E)$ given by a weighted adjacency matrix A:

a linear ordering $<$ of V such that for all $x < y < z$:

$$A_{yz} \geq \min\{A_{xy}, A_{xz}\}$$

This framework captures common vertex elimination orderings of families of chordal graphs, Robinsonian matrices, and ultrametrics.

A symmetric matrix A is a **Robinsonian similarity** if its rows and columns can be (simultaneously) permuted so that for all $x < y < z$:

$$A_{xz} \leq \min\{A_{xy}, A_{yz}\}$$

Special case: adjacency matrices of unit interval graphs.

1969, Roberts:

- A graph $G = (V, E)$ is a **unit interval graph** if and only if there is a linear ordering $<$ of V such that for all $x < y < z$:

$$xz \in E \quad \Longrightarrow \quad xy \in E \text{ and } yz \in E$$

Note: if

$$A_{xz} \le \min\{A_{xy}, A_{yz}\}$$

then

$$A_{yz} \ge \min\{A_{xy}, A_{xz}\}\,.$$

Hence, every Robinsonian similarity has a perfect elimination ordering.

Special case (adjacency matrices of graphs):

every unit interval ordering of a graph is also a perfect elimination ordering.

Theorem (Laurent and Tanigawa, 2017)

The following conditions are equivalent for a weighted graph (G, w)*:*

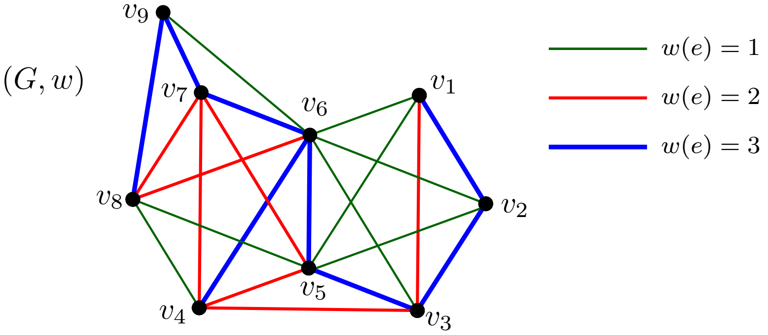
1. (G, w) *has a perfect elimination ordering.*
2. *There exists an ordering of the vertices that is a common perfect elimination ordering of all level graphs.*

A ***k*-weighted graph** is a pair (G, w) where $G = (V, E)$ is a graph and w is a weight function $E \to \{1, \dots, k\}$.

The i-th **level graph** of (G, w) is the graph (V, F_i) where F_i consists of edges of G of weight $\geq i$.

In particular, if a weighted graph (G, w) has a perfect elimination ordering, then **all level graphs are chordal**.

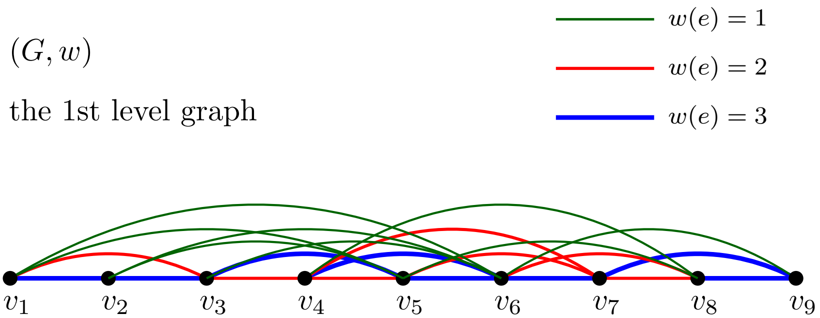
Example:



Example:

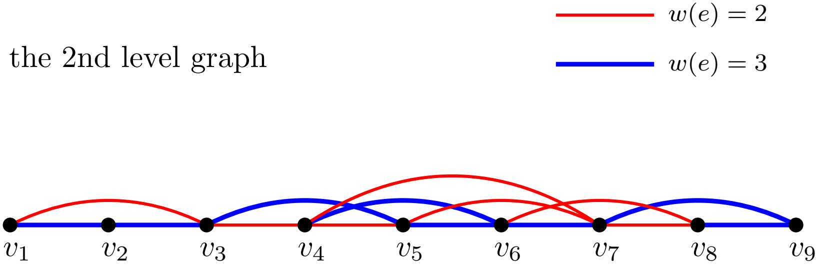
(G, w)

the 1st level graph



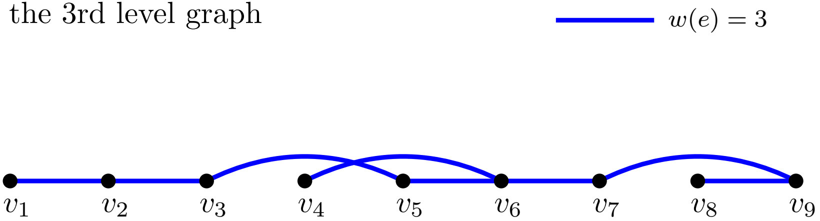
Example:

the 2nd level graph



Example:

the 3rd level graph



A new concept and main questions

We propose the following generalization.

$\mathcal{G}$ – a graph class (for example, the class of chordal graphs)

A weighted graph (G, w) is **level-$\mathcal{G}$** if all its level graphs are in $\mathcal{G}$.

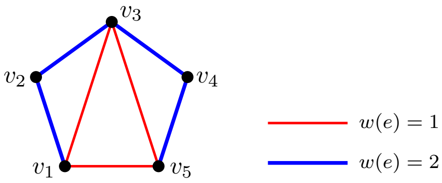
This definition can be applied to any graph class $\mathcal{G}$

(it is not limited to graph classes defined using elimination orderings).

For example, every weighted graph with a perfect elimination ordering is level-chordal.

But not vice versa:

(G, w)



This weighted graph is level-chordal.

However, every perfect elimination ordering of the 2nd level graph starts with v_1 or v_5, neither of which can start a perfect elimination ordering in G (= the 1st level graph).

$\Longrightarrow$ (G, w) does not have a perfect elimination ordering

For a given graph class $\mathcal{G}$, the following are the **main questions of interest**:

1. Can we efficiently recognize level-$\mathcal{G}$ weighted graphs?

   Observation:
   Level-$\mathcal{G}$ weighted graphs are recognizable in polynomial time if and only if graphs in $\mathcal{G}$ are recognizable in polynomial time.

   A better question:
   Can level-$\mathcal{G}$ weighted graphs be recognized **in linear time**?

2. A structural question that can help in this regard:

Can we delete edges from a given level-$\mathcal{G}$ weighted graph one at a time, from lightest to heaviest, so that **all the intermediate graphs** are in $\mathcal{G}$?

We call such a sequence of edge deletions a **sorted $\mathcal{G}$-safe edge elimination scheme**.

Theorem

The following two conditions are equivalent for a graph class $\mathcal{G}$:

1. *Every level-$\mathcal{G}$ weighted graph has a sorted $\mathcal{G}$-safe edge elimination scheme.*
2. *$\mathcal{G}$ is gap monotone.*

A graph class $\mathcal{G}$ is said to be **gap monotone** if for every two graphs $G = (V, E)$ and $G' = (V, E \cup F)$ in $\mathcal{G}$, where $E \cap F = \emptyset$,

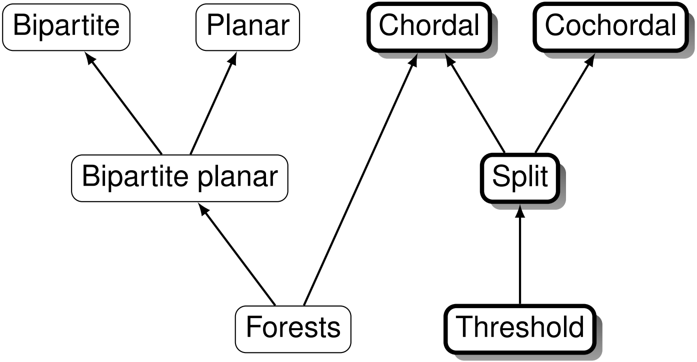
graph G can be obtained from G' by a sequence of edge deletions such that all intermediate graphs are in $\mathcal{G}$.

- **Equivalently:** if $F \neq \emptyset$, then $\exists e \in F$ such that $G' - e \in \mathcal{G}$.

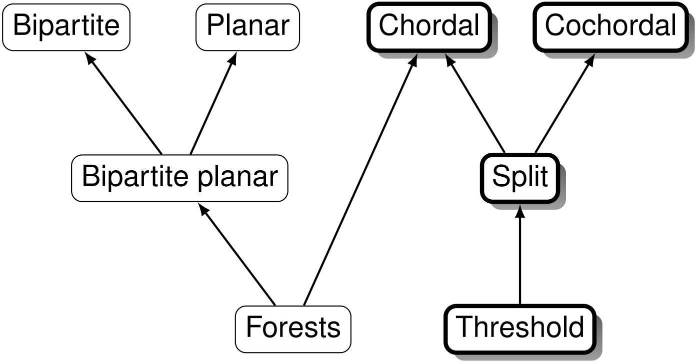
Two observations:

- monotone $\implies$ gap monotone, but not vice versa (monotone = closed under edge deletions)
- If a graph class is gap monotone, then so is its complementary class.

Some gap monotone graph classes:



Some gap monotone graph classes:

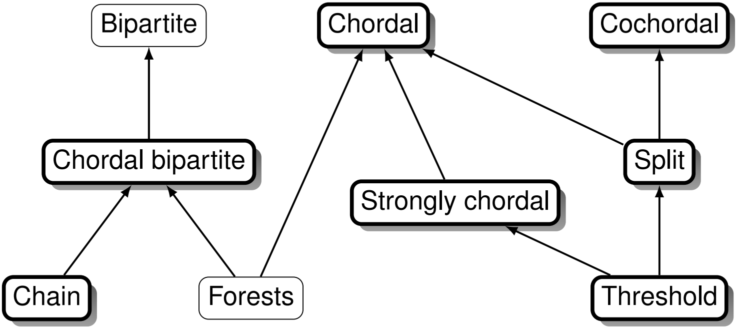


Chordal: **1976, Rose, Tarjan, and Lueker**, **1991, Bakonyi and Constantinescu**

Split: **2009, Heggernes, Mancini**

Threshold: **2009, Heggernes, Papadopoulos**

Some gap monotone graph classes:

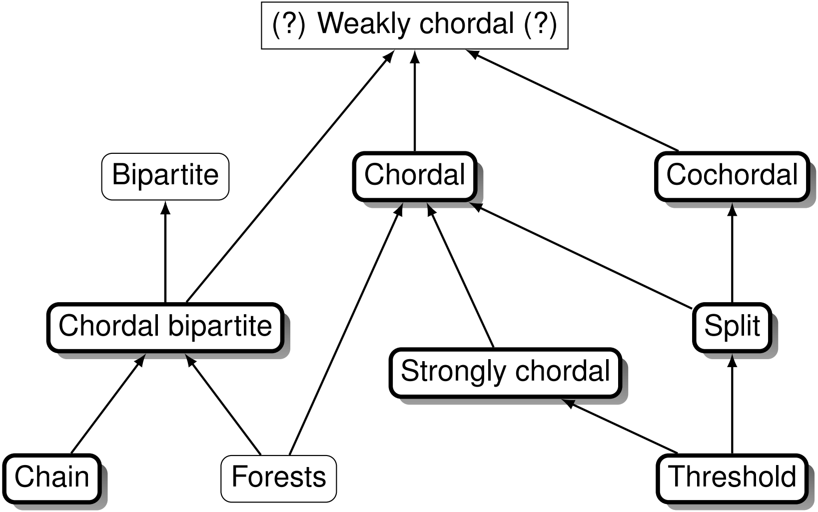


Threshold, chain: **2009, Heggernes, Papadopoulos**

Chordal bipartite, strongly chordal:
2011, Heggernes, Mancini, Papadopoulos, Sritharan

(Heggernes, Mancini, Papadopoulos, and Sritharan refer to the **gap monotonicity** property as **sandwich monotonicity**.)

Open question: Is the class of weakly chordal graphs gap monotone?



Further motivation for gap monotonicity:

- The property allows for one graph in the class to be dynamically changed to another one by successive edge additions (or removals) so that all intermediate graphs are in the class.

  Thus, **dynamic graph algorithms** designed for changing one edge at a time can be applied.

  For example: in **2008, Ibarra** gave fully dynamic algorithms for chordal graphs and split graphs

Let us return to the main question:

Can level-$\mathcal{G}$ weighted graphs be recognized **in linear time**?

Theorem

The following two conditions are equivalent for a graph class $\mathcal{G}$:

1. *Every level-$\mathcal{G}$ weighted graph has a sorted $\mathcal{G}$-safe edge elimination scheme.*
2. *$\mathcal{G}$ is gap monotone.*

Thus, if $\mathcal{G}$ is a gap monotone graph class, then the fact that a weighted graph is level-$\mathcal{G}$ can be certified by a sorted $\mathcal{G}$-safe edge elimination scheme.

We show that the classes of

threshold graphs, **split graphs**, and **chain graphs**

admit particularly simple sorted $\mathcal{G}$-safe edge elimination schemes.

Two general concepts

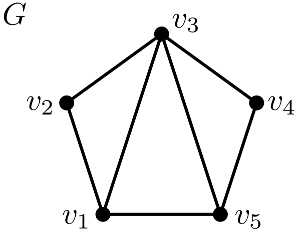
Definition

Let G be a graph and let F be a set of edges of G.

A **degree-minimal edge in F** is an edge $xy \in F$ such that:

1. vertex x has the smallest degree in G among all vertices incident to an edge in F, and
2. the degree of y in G is the smallest among all neighbors of x that are adjacent to x via an edge in F.

Example:



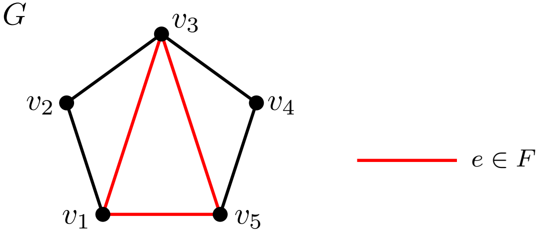
Definition

Let G be a graph and let F be a set of edges of G.

A **degree-minimal edge in F** is an edge $xy \in F$ such that:

1. vertex x has the smallest degree in G among all vertices incident to an edge in F, and
2. the degree of y in G is the smallest among all neighbors of x that are adjacent to x via an edge in F.

Example:



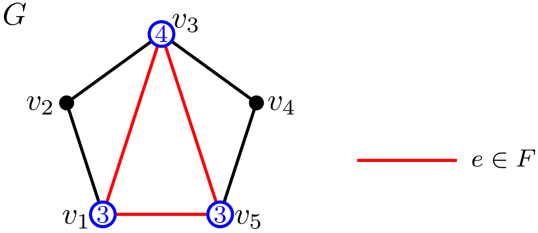
Definition

Let G be a graph and let F be a set of edges of G.

A **degree-minimal edge in F** is an edge $xy \in F$ such that:

1. vertex x has the smallest degree in G among all vertices incident to an edge in F, and
2. the degree of y in G is the smallest among all neighbors of x that are adjacent to x via an edge in F.

Example:

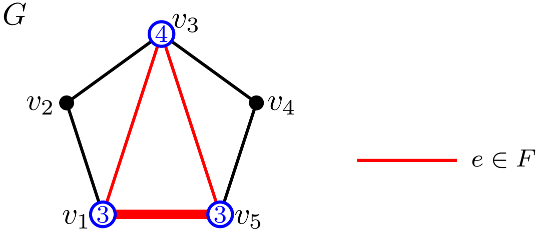


Definition
Let G be a graph and let F be a set of edges of G.

A **degree-minimal edge in F** is an edge $xy \in F$ such that:

1. vertex x has the smallest degree in G among all vertices incident to an edge in F, and
2. the degree of y in G is the smallest among all neighbors of x that are adjacent to x via an edge in F.

Example:



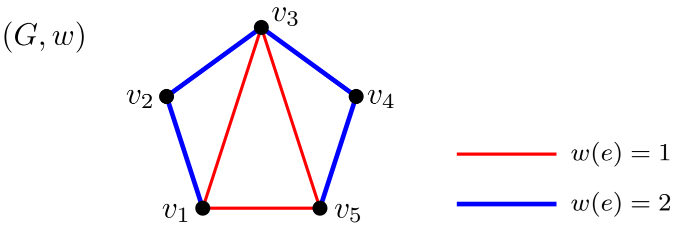
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



Definition

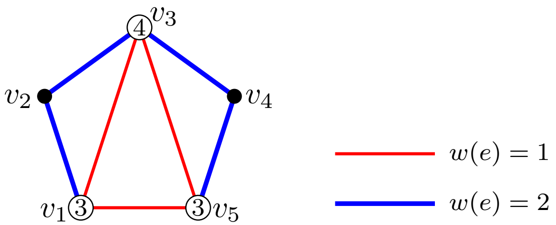
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

(G, w)



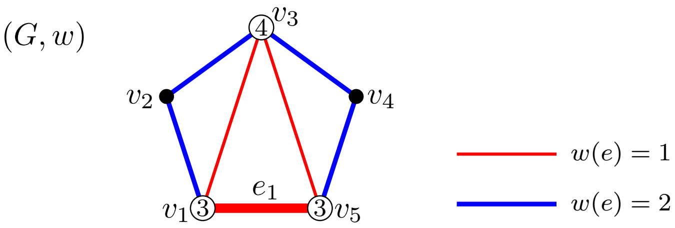
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



Definition

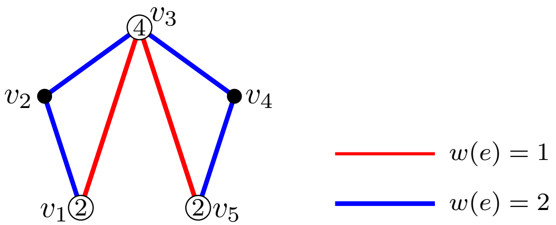
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

$G - e_1$



Definition

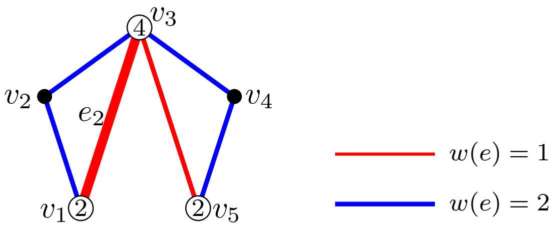
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

$G - e_1$



Definition

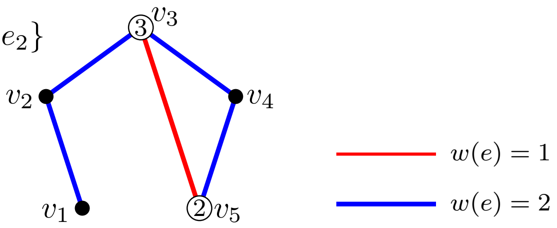
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

$G - \{e_1, e_2\}$



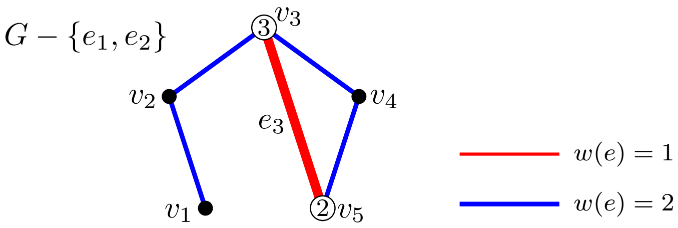
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



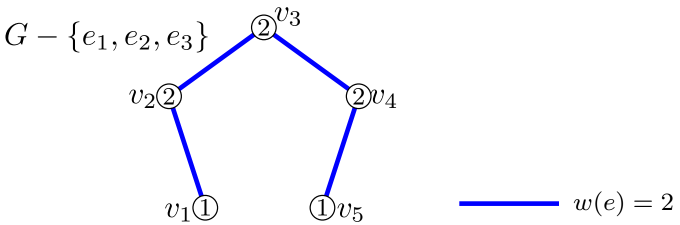
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



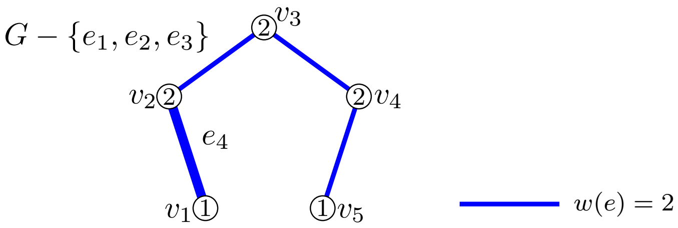
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

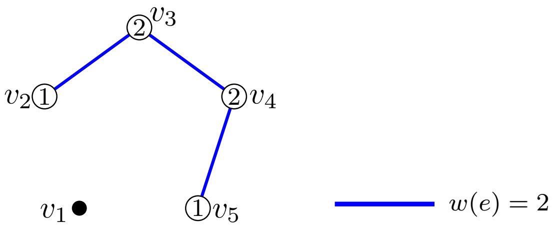


Definition
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

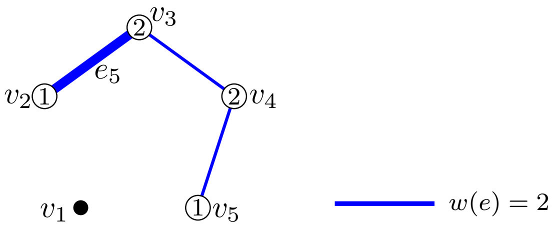


Definition
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



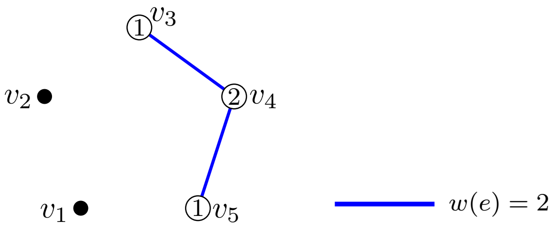
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

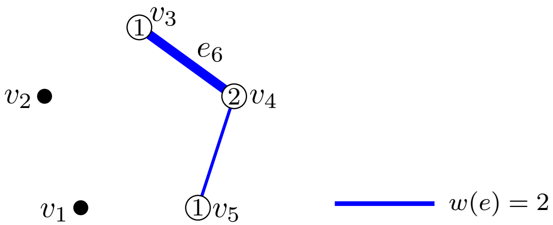


Definition
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



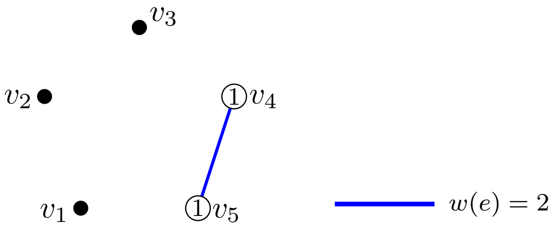
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:

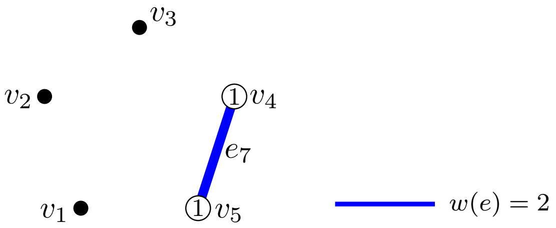


Definition
Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



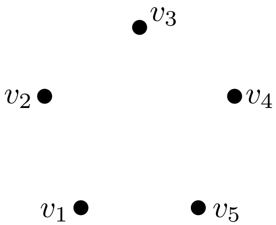
Definition

Let (G, w) be a weighted graph.

A linear ordering $\tau = (e_1, \ldots, e_m)$ of the edges of G is said to be a **degree-minimal edge elimination scheme (dmees)** of (G, w) if for every $i \in \{1, \ldots, m\}$,

edge e_i is a degree-minimal edge in the set of all minimum-weight edges in the graph $G - \{e_1, \ldots, e_{i-1}\}$.

Example:



A linear-time algorithm

Theorem

There exists an algorithm with the following specifications:

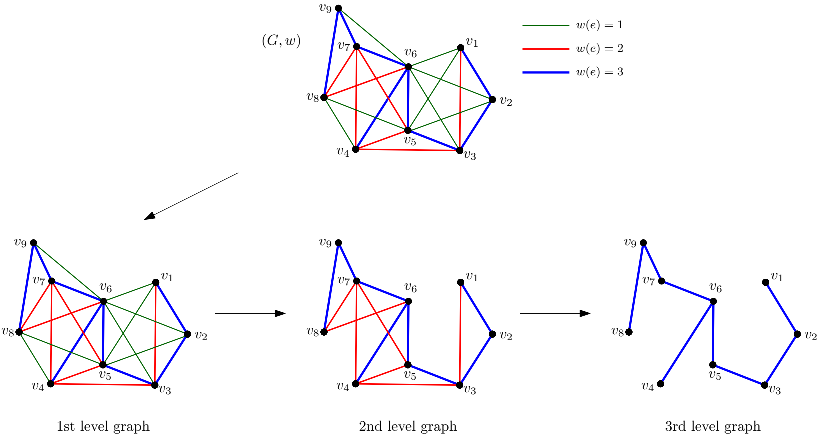
Input: *A weighted graph* $(G = (V, E), w)$.

Output: *A degree-minimal edge elimination scheme of G.*

Running time: $\mathcal{O}(|V| + |E|)$.

The idea of the algorithm:

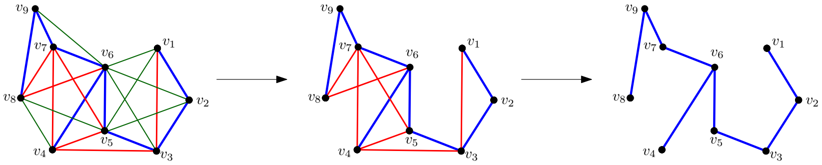
1. Consider the *k* level graphs.



The idea of the algorithm:

2. For each level graph, choose a vertex of smallest degree, remove all the edges of weight *i* incident with it and iterate.

This defines an ordering of vertices within each level graph.



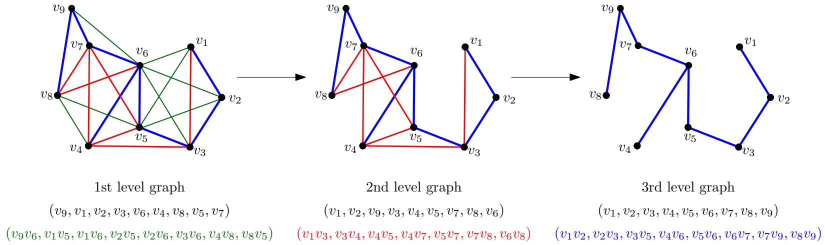
1st level graph
$(v_9, v_1, v_2, v_3, v_4, v_7, v_8, v_5, v_6)$

2nd level graph
$(v_1, v_2, v_9, v_4, v_5, v_8, v_3, v_6, v_7)$

3rd level graph
$(v_1, v_4, v_8, v_2, v_3, v_5, v_7, v_9, v_6)$

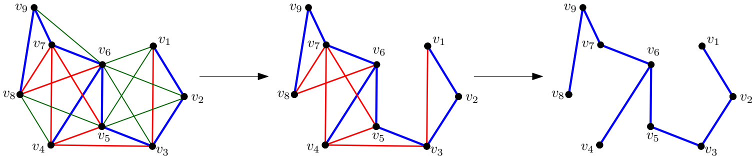
The idea of the algorithm:

3. This also defines an ordering of the edges of weight *i* according to when they were deleted in the *i*-th level graph.



The idea of the algorithm:

4. Finally, reorder the edges within each star if necessary, to satisfy the second constraint from the definition of degree-minimality.



1st level graph

$(v_9, v_1, v_2, v_3, v_6, v_4, v_8, v_5, v_7)$

$(v_9v_6, v_1v_5, v_1v_6, v_2v_5, v_2v_6, v_3v_6, v_4v_8, v_8v_5)$

2nd level graph

$(v_1, v_2, v_9, v_3, v_4, v_5, v_7, v_8, v_6)$

$(v_1v_3, v_3v_4, v_4v_5, v_4v_7, v_5v_7, v_7v_8, v_6v_8)$

3rd level graph

$(v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

$(v_1v_2, v_2v_3, v_3v_5, v_4v_6, v_5v_6, v_6v_7, v_7v_9, v_8v_9)$

Why do we care?

Recall:

- A graph class $\mathcal{G}$ is **gap monotone** if for every two graphs $G = (V, E)$ and $G' = (V, E \cup F)$ in $\mathcal{G}$, where $E \cap F = \emptyset$,

  graph G can be obtained from G' by a sequence of edge deletions such that all intermediate graphs are in $\mathcal{G}$.

  Equivalently: if $F \neq \emptyset$, then $\exists e \in F$ such that $G' - e \in \mathcal{G}$.

- For gap monotone graph classes, the fact that a weighted graph is level-$\mathcal{G}$ can be certified by a sorted $\mathcal{G}$-safe edge elimination scheme.

An edge e in a graph $G \in \mathcal{G}$ is said to be **$\mathcal{G}$-safe** if $G - e \in \mathcal{G}$.

Theorem

Let $\mathcal{G}$ be a graph class such that for every two graphs $G = (V, E)$ and $G' = (V, E \cup F)$ in $\mathcal{G}$, where $E \cap F = \emptyset$, every degree-minimal edge in F is $\mathcal{G}$-safe.

(In particular, $\mathcal{G}$ is gap monotone.)

Then, for every level-$\mathcal{G}$ weighted graph (G, w), every dmees is also a sorted $\mathcal{G}$-safe edge elimination scheme.

We show that the condition of the theorem is satisfied for the classes of threshold graphs, split graphs, and chain graphs.

- This gives a unifying approach to proving that these graph classes are gap monotone.
- It also leads to **linear-time recognition algorithms** for level-threshold, level-split, and level-chain weighted graphs.

- A graph is **threshold** if it admits weights on vertices and a threshold t such that

  a subset of vertices is independent if and only if its total weight is at most t.

- A graph is **split** if its vertex set can be partitioned into a clique and an independent set.

- A graph is **chain** if it is a bipartite graph

  having a bipartition (X, Y) such that the neighborhoods of vertices in X are nested, that is, $X = \{x_1, \ldots, x_p\}$ such that

$$N(x_1) \subseteq N(x_2) \subseteq \ldots \subseteq N(x_p)\,.$$

To obtain linear running time, we proceed as follows:

1. We compute a degree-minimal edge elimination scheme τ.
2. We check whether τ is a sorted $\mathcal{G}$-safe edge elimination scheme.

The details of step 2 are class specific.

For **split graphs**, we use a dynamic recognition algorithm by Ibarra.

The algorithm relies on the fact that split graphs are characterized by their degree sequences.

Theorem (Hammer, Simeone, 1981)

Let $d_1 \geq d_2 \geq \ldots \geq d_n$ *be the degree sequence of a graph* G*. Also, let* $h = \max\{i \ : \ d_i \geq i - 1\}$*.*

Then, G *is a split graph if and only if*

$$\sum_{i=1}^{h} d_i = h(h-1) + \sum_{i=h+1}^{n} d_i\,.$$

For **threshold graphs**, we can use a dynamic recognition algorithm by Shamir and Sharan (2004).

The algorithm relies on the fact that threshold graphs are precisely the split cographs.

It uses the dynamic recognition algorithm for **cographs** developed by the authors and the algorithm by Ibarra for split graphs.

For **threshold graphs**, a more direct algorithm can be developed using the fact that

a graph is threshold if and only if it can be generated from the one-vertex graph by successive additions of universal and isolated vertices.

Summary

1. We generalized the concept of graph classes to graphs equipped with an ordered partition of their edges:

   a weighted graph is **level-$\mathcal{G}$** if all its level graphs are in $\mathcal{G}$.

2. A particularly nice situation arises in the case of gap monotone graph classes:

   edges can be eliminated not only block by block
   but one edge at a time, respecting the weights.

   In this case, **dynamic graph algorithms** can be applied.

Summary

3. We gave a linear-time algorithm for computing a **degree-minimal edge elimination scheme (dmees)** of an arbitrary weighted graph.

4. For the weighted graphs that are level-threshold, level-split, or level-chain, every dmees is also a **sorted $\mathcal{G}$-safe edge elimination scheme**.

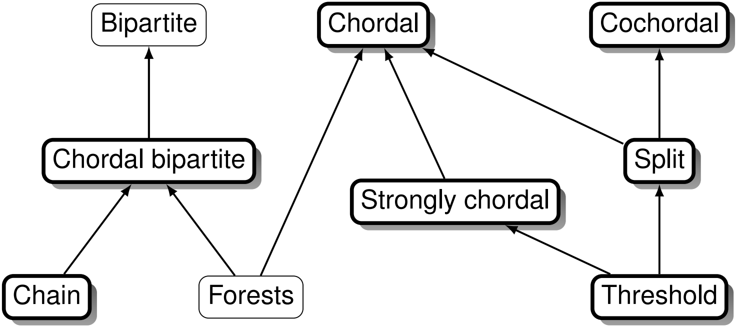
5. This leads to linear-time recognition algorithms.

Two open questions

1. Is there a linear-time algorithm to recognize level-chordal weighted graphs?

2. Is the class of weakly chordal graphs gap monotone?

Thank you!


Bipartite
Chordal
Cochordal
Chordal bipartite
Split
Strongly chordal
Chain
Forests
Threshold