Support Computation for Mining Frequent Subgraphs in a Single Graph

Mathias Fiedler and Christian Borgelt

Intelligent Data Analysis and Graphical Models Research Unit
European Center for Soft Computing
c/ Gonzalo Gutiérrez Quirós s/n, 33600 Mieres, Spain
christian.borgelt@softcomputing.es
http://www.borgelt.net/

Contents

- **Frequent Subgraph Mining**
- **Subgraph Support and Overlap Graphs**
  - Embeddings (Subgraph Isomorphisms)
  - Monotonicity of Subgraph Support
  - Maximum Independent Set Support: Overlapping Embeddings
  - Harmful Overlap Support: Existence of Equivalent Ancestors
- **Subgraph Support Computation**
  - Finding Equivalent Ancestor Embeddings
  - Restriction to Connected Subgraphs
- **Experimental Results**
- **Summary**

Frequent Subgraph Mining

Problem:

Given: a database of (attributed) graphs and
a minimum support value (frequency)

Desired: the set of all frequent subgraphs
(subgraphs that have a support greater than the minimum support)

Basic Search Procedure

- Start with a single node (try all labels/attributes).
- In each step add an edge (and maybe a node).
  (Try all possibilities, but avoid redundant search.)
- Prune infrequent subgraphs, extend frequent subgraphs.
- Standard support definition:
  number of graphs in the database that contain the subgraph.
  (Not applicable to the single graph setting, which is considered here.)

Embeddings (Subgraph Isomorphisms)

A **labeled** or **attributed graph** is a triple $G = (V, E, l)$, where

- V is the set of vertices,
- $E \subseteq V \times V - \{(v, v) \mid v \in V\}$ is the set of edges, and
- $l : V \cup E \to L$ assigns labels from the set L to nodes and edges.

Let $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$ be two labeled graphs.
A **subgraph isomorphism** of S to G is an injective function $f : V_S \to V_G$ with

- $\forall v \in V_S : l_S(v) = l_G(f(v))$ and
- $\forall (u, v) \in E_S : (f(u), f(v)) \in E_G \wedge l_S((u, v)) = l_G((f(u), f(v)))$.

Let f_1 and f_2 two subgraph isomorphisms of S to G and
$V_1 = \{v \in V_G \mid \exists u \in V_S : v = f_1(u)\}$ and $V_2 = \{v \in V_G \mid \exists u \in V_S : v = f_2(u)\}$.
f_1 and f_2 are called

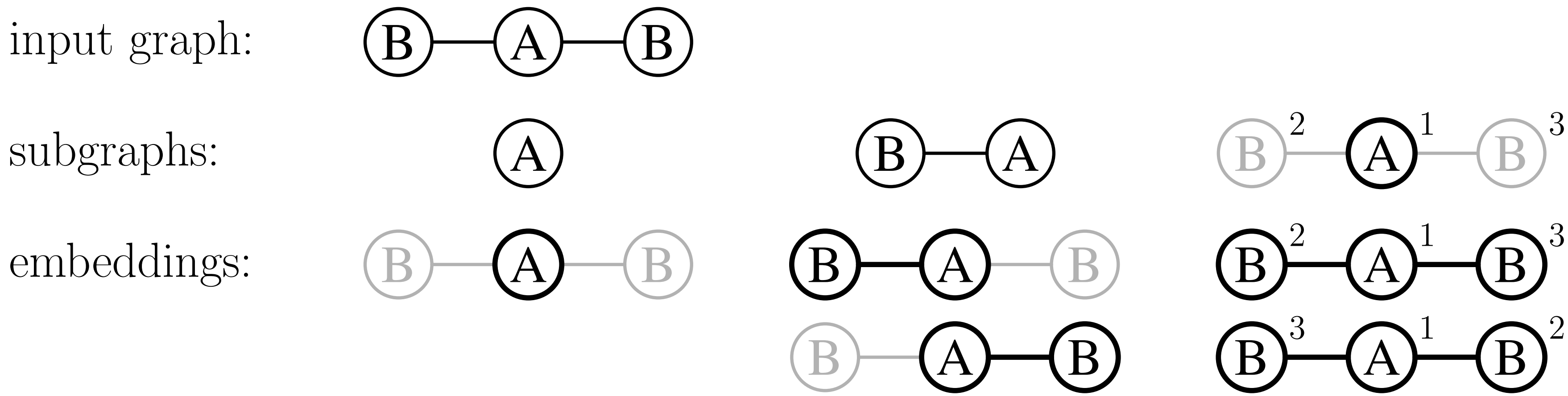
- **overlapping**, written $f_1 \circledcirc f_2$, iff $V_1 \cap V_2 \neq \emptyset$,
- **equivalent**, written $f_1 \circ f_2$, iff $V_1 = V_2$,
- **identical**, written $f_1 \equiv f_2$, iff $\forall v \in V_S : f_1(v) = f_2(v)$.

Anti-Monotonicity of Subgraph Support

Most natural definition of subgraph support in a single graph setting:
number of embeddings (subgraph isomorphisms)

Problem: The number of embeddings of a subgraph is not anti-monotone.

Example:



But: Anti-monotonicity is vital for the efficiency of frequent subgraph mining.

Question: How should we define subgraph support in a single graph?

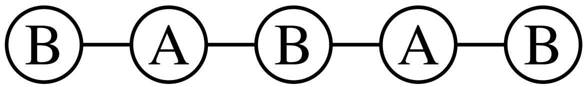
Overlap Graphs of Embeddings

Let $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$ be two labeled graphs and let V_O be the set of all embeddings (subgraph isomorphisms) of S into G.

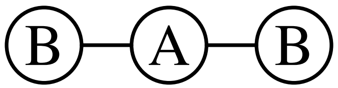
The **overlap graph** of S w.r.t. G is the graph $O = (V_O, E_O)$, which has the set V_O of embeddings as its node set and the edge set $E_O = \{(f_1, f_2) \mid f_1, f_2 \in V_O \wedge f_1 \not\equiv f_2 \wedge f_1 \circledcirc f_2\}$.

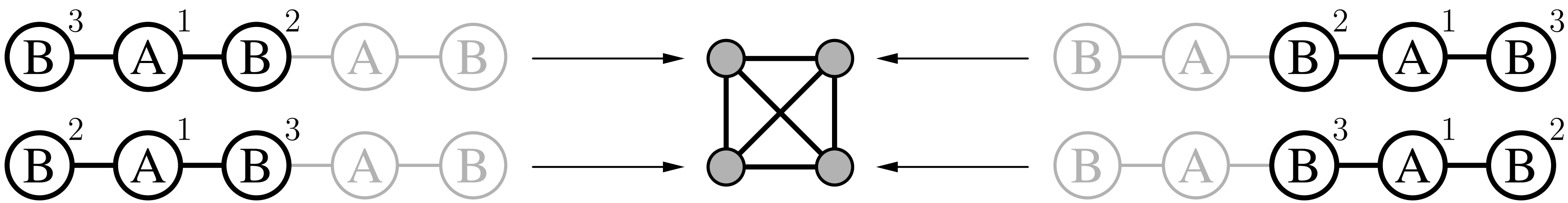
Example:

input graph:



subgraph:





Maximum Independent Set Support

Let $G = (V, E)$ be an (undirected) graph with node set V
and edge set $E \subseteq V \times V - \{(v, v) \mid v \in V\}$.

An **independent node set** of G is a set $I \subseteq V$ with $\forall u, v \in I : (u, v) \notin E$.

I is a **maximum independent node set** iff

- it is an independent node set and
- for all independent node sets J of G it is $|I| \geq |J|$.

Notes: Finding a maximum independent node set is an NP-complete problem.
However, a greedy algorithm usually gives very good approximations.

Let $O = (V_O, E_O)$ be the overlap graph of the embeddings
of a labeled graph $S = (V_S, E_S, l_S)$ into a labeled graph $G = (V_G, E_G, l_G)$.

The **maximum independent set support** (or **MIS-support** for short)
of S w.r.t. G is the size of a maximum independent node set of O.

Anti-Monotonicity of MIS-Support: Preliminaries

Let $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$ be two labeled graphs.

Let $T = (V_T, E_T, l_T)$ a (non-empty) proper subgraph of S
(that is, $V_T \subset V_S$, $E_T = (V_T \times V_T) \cap E_S$, and $l_T \equiv l_S|_{V_T \cup E_T}$).

Let f be an embedding of S into G.

An embedding f' of the subgraph T is called a **T-ancestor** of the embedding f iff $f' \equiv f|_{V_T}$, that is, if f' coincides with f on the node set V_T of T.

Observations:

For given G, S, T and f the T-ancestor f' of the embedding f is uniquely defined.

Let f_1 and f_2 be two (non-identical, but maybe equivalent) embeddings of S into G.

f_1 and f_2 overlap if there exist overlapping T-ancestors f_1' and f_2' of the embeddings f_1 and f_2, respectively.

(Note: The inverse implication does not hold generally.)

Anti-Monotonicity of MIS-Support: Proof

Theorem: MIS-support is anti-monotone.

Proof: We have to show that the MIS-support of a subgraph S w.r.t. a graph G cannot exceed the MIS-support of any (non-empty) proper subgraph T of S.

- Let I be an arbitrary independent node set of the overlap O graph of S w.r.t. G.
- The set I induces a subset I' of the nodes of the overlap graph O' of an (arbitrary, but fixed) subgraph T of the considered subgraph S, which consists of the (uniquely defined) T-ancestors of the nodes in I.
- It is $|I| = |I'|$, because no two elements of I can have the same T-ancestor.
- With similar argument: I' is an independent node set of the overlap graph O'.
- As a consequence, since I is arbitrary, every independent node set of O induces an independent node set of O' of the same size.
- Hence the maximum independent node set of O' must be at least as large as the maximum independent node set of O.

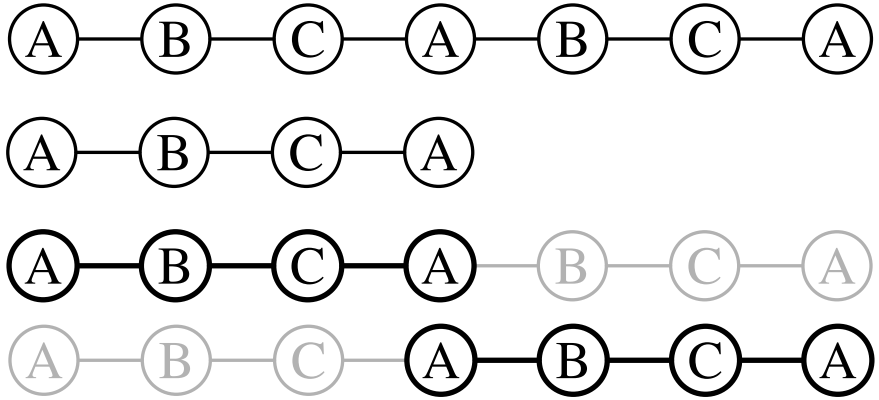
Harmful and Harmless Overlaps of Embeddings

Not all overlaps of embeddings are harmful:

input graph:

subgraph:

embeddings:



Let $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$ be two labeled graphs and let f_1 and f_2 be two embeddings (subgraph isomorphisms) of S to G.

f_1 and f_2 are called **harmfully overlapping**, written $f_1 \bullet\!\!\bullet f_2$, iff

- they are equivalent or
- there exists a (non-empty) proper subgraph T of S, so that the T-ancestors f_1' and f_2' of f_1 and f_2, respectively, are equivalent.

Harmful Overlap Graphs and Subgraph Support

Let $G = (V_G, E_G, l_G)$ and $S = (V_S, E_S, l_S)$ be two labeled graphs and let V_H be the set of all embeddings (subgraph isomorphisms) of S into G.

The **harmful overlap graph** of S w.r.t. G is the graph $H = (V_H, E_H)$, which has the set V_H of embeddings as its node set and the edge set $E_H = \{(f_1, f_2) \mid f_1, f_2 \in V_H \wedge f_1 \not\equiv f_2 \wedge f_1 \mathbin{\bullet\!\!\bullet} f_2\}$.

Let $H = (V_H, E_H)$ be the harmful overlap graph of the embeddings of a labeled graph $S = (V_S, E_S, l_S)$ into a labeled graph $G = (V_G, E_G, l_G)$.

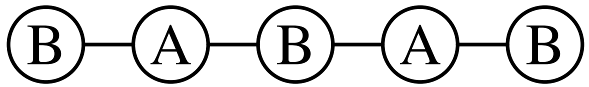
The **harmful overlap support** (or **HO-support** for short) of the graph S w.r.t. G is the size of a maximum independent node set of H.

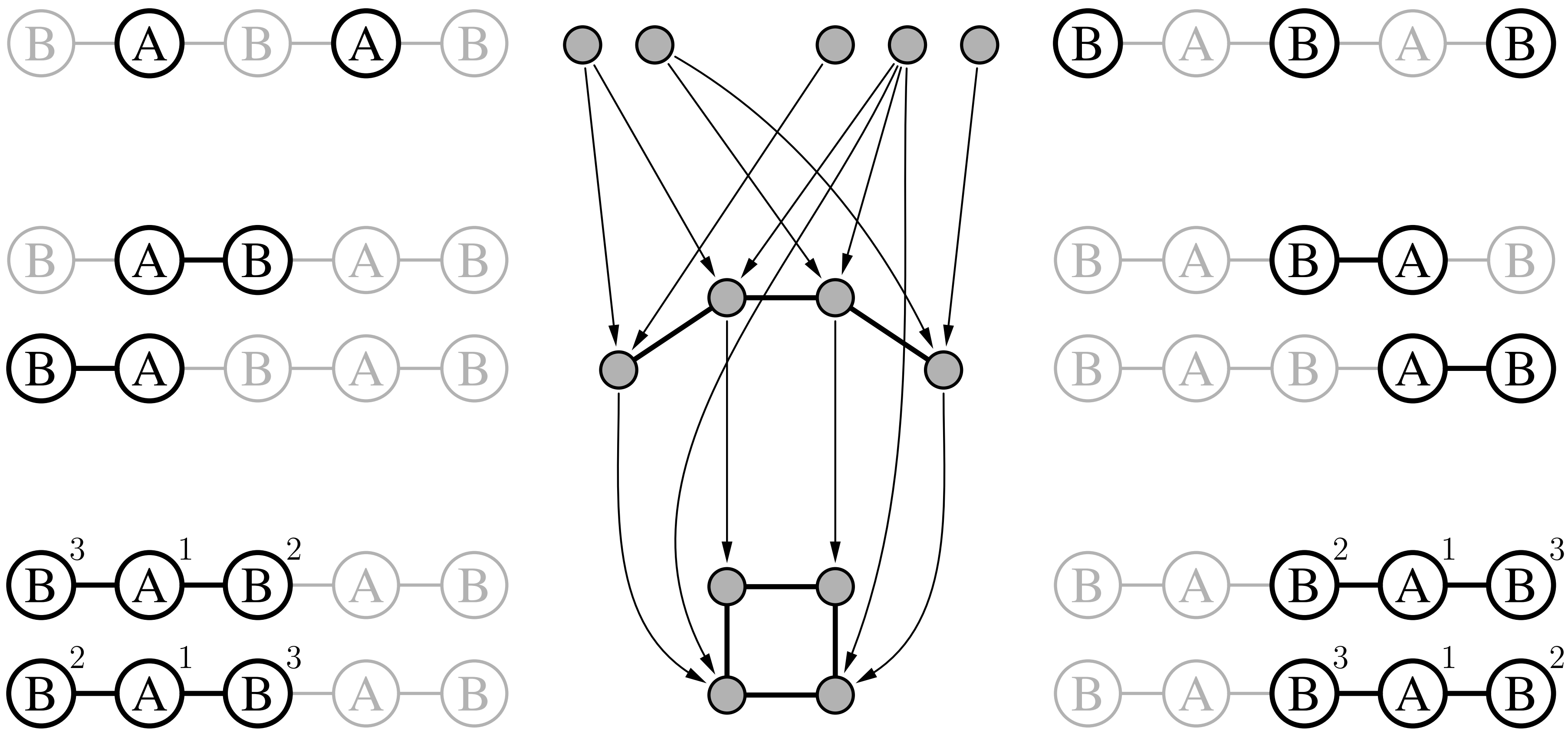
Theorem: HO-support is anti-monotone.

Proof: Identical to proof for MIS-support.
(The same two observations hold, which were all that was needed.)

Harmful Overlap Graphs and Ancestor Relations

input graph:





Subgraph Support Computation

Checking whether two embeddings overlap is easy, but:

How do we check whether two embeddings overlap harmfully?

Core ideas of the harmful overlap test:

- Try to construct a subgraph $S_E = (V_E, E_E, l_E)$ that yields equivalent ancestors of two given embeddings f_1 and f_2 of a graph $S = (V_S, E_S, l_S)$.
- For such a subgraph S_E the mapping $g : V_E \to V_E$ with $v \mapsto f_2^{-1}(f_1(v))$, where f_2^{-1} is the inverse of f_2, must be a bijective mapping.
- More generally, g must be an **automorphism** of S_E, that is, a subgraph isomorphism of S_E to itself.
- Exploit the properties of automorphism to exclude nodes from the graph S that cannot be in V_E.

Subgraph Support Computation

Input: Two (different) embeddings f_1 and f_2 of a labeled graph $S = (V_S, E_S, l_S)$ into a labeled graph $G = (V_G, E_G, l_G)$.

Output: Whether f_1 and f_2 overlap harmfully.

1) Form the sets $V_1 = \{v \in V_G \mid \exists u \in V_S : v = f_1(u)\}$ and $V_2 = \{v \in V_G \mid \exists u \in V_S : v = f_2(u)\}$.

2) Form the sets $W_1 = \{v \in V_S \mid f_1(v) \in V_1 \cap V_2\}$ and $W_2 = \{v \in V_S \mid f_2(v) \in V_1 \cap V_2\}$.

3) If $V_E = W_1 \cap W_2 = \emptyset$, return *false*, otherwise return *true*.

- V_E is the node set of a subgraph S_E that induces equivalent ancestors.
- Any node $v \in V_S - V_E$ cannot contribute to such equivalent ancestors.
- Hence V_E is a maximal set of nodes for which g is a bijection.

Restriction to Connected Subgraphs

The search for frequent subgraphs is usually restricted to **connected graphs**.

We cannot conclude that no edge is needed if the subgraph S_E is not connected: there may be a connected subgraph of S_E that induces equivalent ancestors of the embeddings f_1 and f_2.

Hence we have to consider subgraphs of S_E in this case.
However, checking all possible subgraphs is prohibitively costly.

Computing the edge set E_E of the subgraph S_E:

1) Let $E_1 = \{(v_1, v_2) \in E_G \mid \exists (u_1, u_2) \in E_S : (v_1, v_2) = (f_1(u_1), f_1(u_2))\}$
and $E_2 = \{(v_1, v_2) \in E_G \mid \exists (u_1, u_2) \in E_S : (v_1, v_2) = (f_2(u_1), f_2(u_2))\}$.

2) Let $F_1 = \{(v_1, v_2) \in E_S \mid (f_1(v_1), f_1(v_2)) \in E_1 \cap E_2\}$
and $F_2 = \{(v_1, v_2) \in E_S \mid (f_2(v_1), f_2(v_2)) \in E_1 \cap E_2\}$.

3) Let $E_E = F_1 \cap F_2$.

Restriction to Connected Subgraphs

Lemma: Let $S_C = (V_C, E_C, l_C)$ be an (arbitrary, but fixed) connected component of the subgraph S_E and let $W = \{v \in V_C \mid g(v) \in V_C\}$

(reminder: $\forall v \in V_E : g(v) = f_2^{-1}(f_1(v))$, g is an automorphism of S_E)

Then it is either $W = \emptyset$ or $W = V_C$.

Proof: (by contradiction)

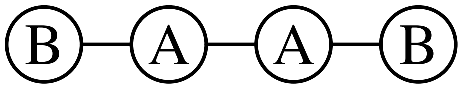
- Suppose that there is a connected component S_C with $W \neq \emptyset$ and $W \neq V_C$.
- Choose two nodes $v_1 \in W$ and $v_2 \in V_C - W$.
- v_1 and v_2 are connected by a path in S_C, since S_C is a connected component. On this path there must be an edge (v_a, v_b) with $v_a \in W$ and $v_b \in V_C - W$.
- It is $(v_a, v_b) \in E_E$ and therefore $(g(v_a), g(v_b)) \in E_E$ (g is an automorphism).
- Since $g(v_a) \in V_C$, it follows $g(v_b) \in V_C$.
- However, this implies $v_b \in W$, contradicting $v_b \in V_C - W$.

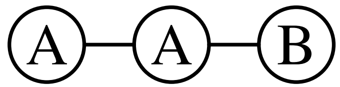
Further Optimization

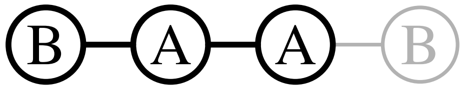
The test can be further optimized by the following simple insight:

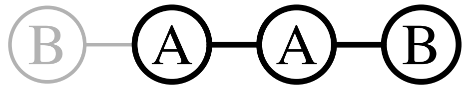
- Two embeddings f_1 and f_2 overlap harmfully if $\exists v \in V_S : f_1(v) = f_2(v)$, because then such a node v alone gives rise to equivalent ancestors.
- This test can be performed very quickly, so it should be the first step.
- Additional advantage:
  connected components consisting of isolated nodes can be neglected afterwards.

A simple example of harmful overlap without identical images:

input graph: 

subgraph: 

embeddings: 



Note that the subgraph inducing equivalent ancestors can be arbitrarily complex even if $\forall v \in V_S : f_1(v) \neq f_2(v)$.

Final Procedure for Harmful Overlap Test

Input: Two (different) embeddings f_1 and f_2 of a labeled graph $S = (V_S, E_S, l_S)$ into a labeled graph $G = (V_G, E_G, l_G)$.

Output: Whether f_1 and f_2 overlap harmfully.

1) If $\exists v \in S : f_1(v) = f_2(v)$, return *true*.

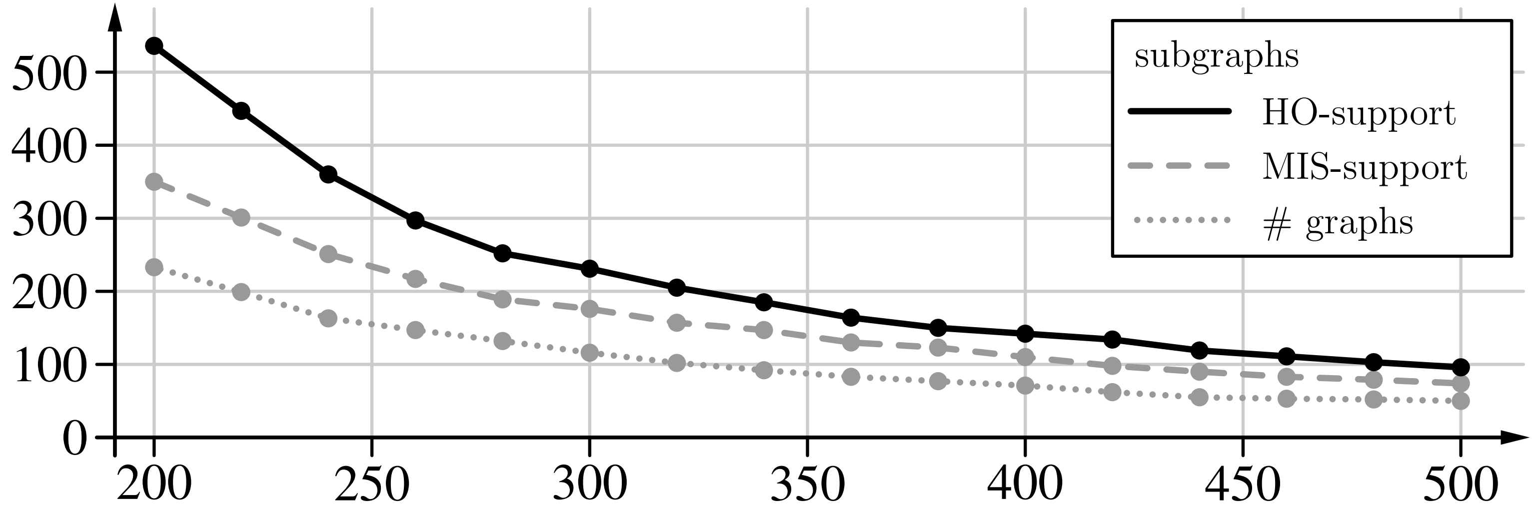
2) Form the edge set E_E of the subgraph S_E (as described above) and form the (reduced) node set $V'_E = \{v \in V_S \mid \exists u \in V_S : (v, u) \in E_E\}$. (Note that V'_E does not contain isolated nodes.)

3) Let $S^i_C = (V^i_C, E^i_C)$, $1 \leq i \leq n$, be the connected components of $S'_E = (V'_E, E_E)$.

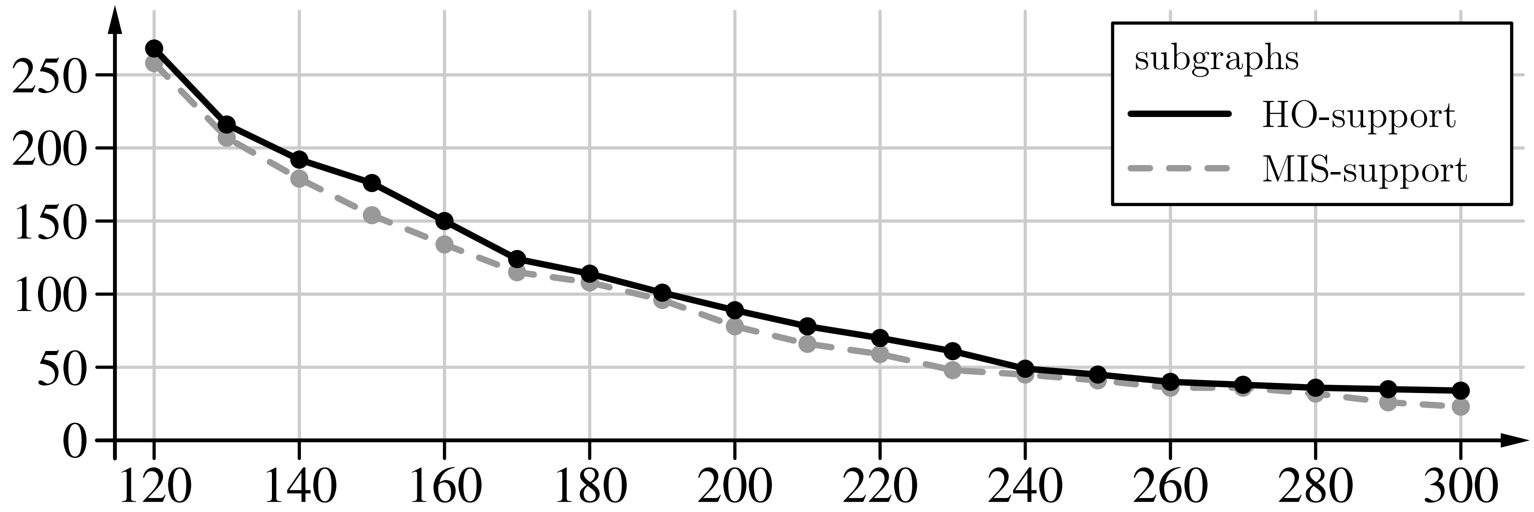
   If $\exists i; 1 \leq i \leq n : \exists v \in V^i_C : g(v) = f_2^{-1}(f_1(v)) \in V^i_C$, return *true*, otherwise return *false*.

Experimental Results

Index Chemicus 1993



Tic-Tac-Toe win



Summary

- Defining subgraph support in the single graph setting:
  maximum independent node set of an overlap graph of the embeddings
- Simple **proof** that **MIS-support** is **anti-monotone**:
  look at induced independent node sets for substructures
- Definition of **harmful overlap support** of a subgraph:
  existence of equivalent ancestor embeddings
- Simple **procedure** for testing whether two embeddings overlap harmfully.
- Simple **proof** that **harmful overlap support** is **anti-monotone**.
- Restriction to **connected substructures** and optimizations.

Software: MoSS — Molecular Substructure Miner

http://www.borgelt.net/moss.html