Mine 'Em All: A Note on (Complexity of) Mining All Graphs

Ondřej Kuželka[1] and Jan Ramon[2]
[1]Cardiff University, [2]KU Leuven

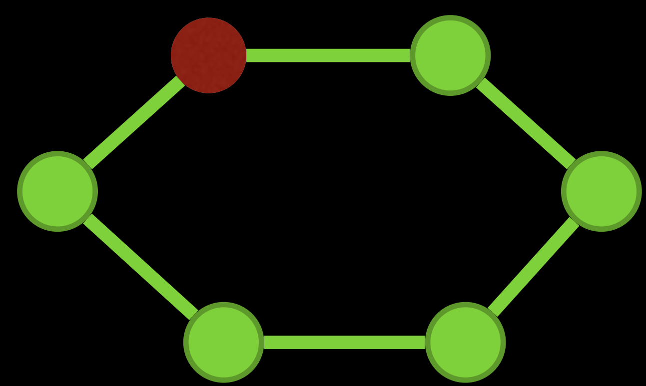
Question

- **When can *graph mining* with an *intractable pattern matching operator be fast*?**

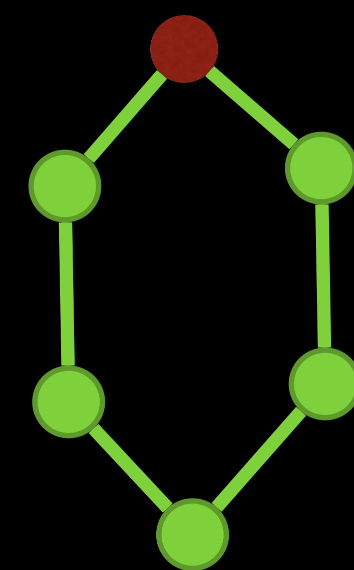
- Motivation: Horváth & Ramon have shown that frequent bounded-treewidth graphs can be mined in incremental-polynomial time even though subgraph isomorphism is NP-hard for them.

Preliminaries

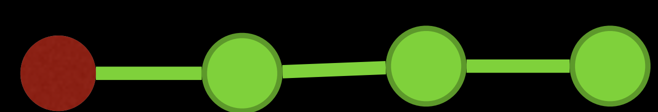
- **Isomorphism**



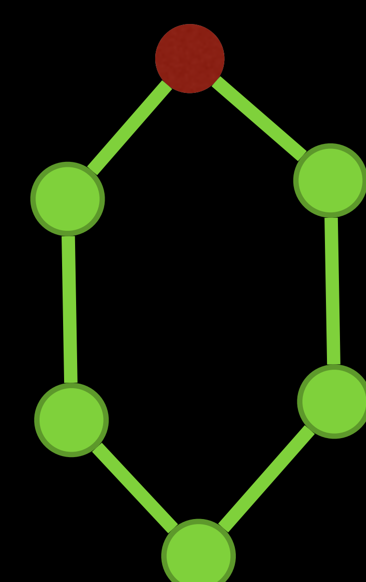
$\sim$



- **Subgraph isomorphism:**



$\preccurlyeq$

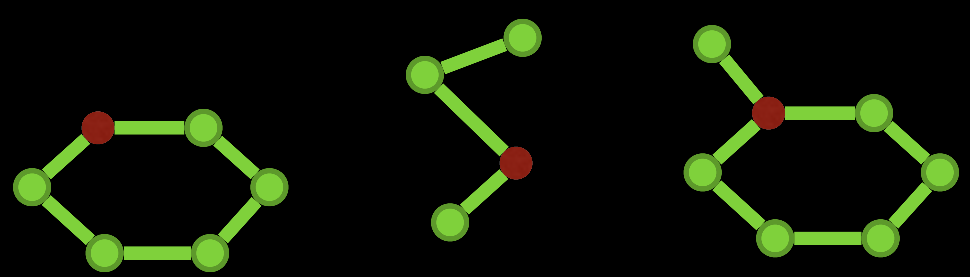


- **+ other matching operators (homeomorphism, minor embedding, induced operators…)**

Frequent Graph Mining

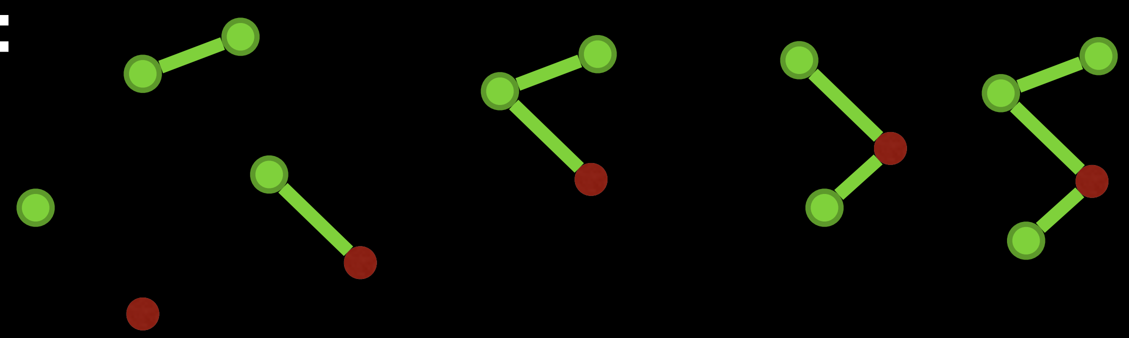
- **Given:** a database *DB* of graphs and a frequency threshold *t*

DB:



- **Task:** Output all nonisomorphic connected graphs subgraph isomorphic to at least *t* graphs from *DB*.

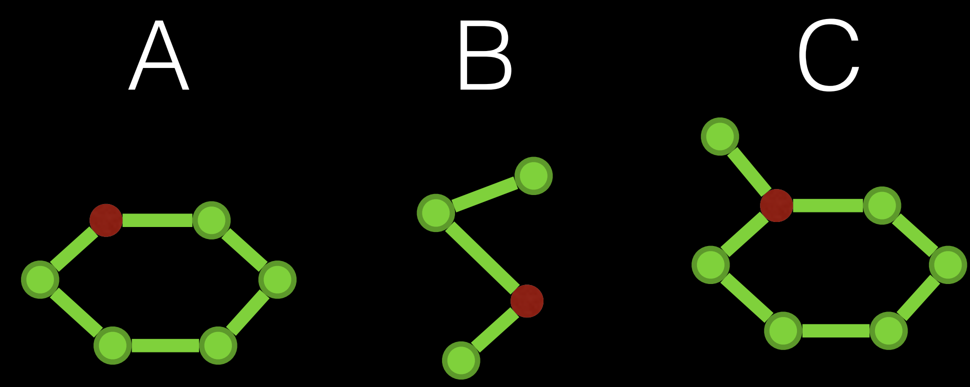
Example (t=3):



How Typical FGM Algos. Work

Example (t=2):

DB:



Candidates (1):  

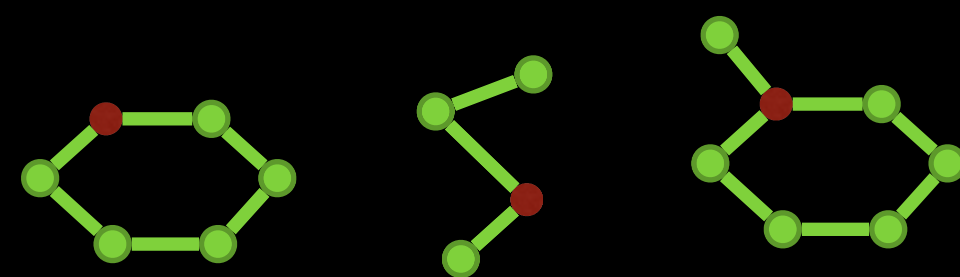
Occurrences: {A,B,C} and {A,B,C}

How Typical FGM Algos. Work

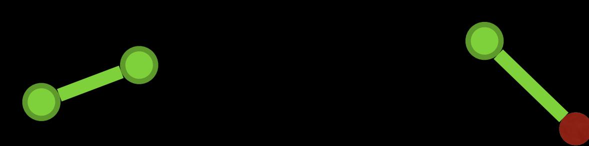
Example (t=2):

A B C

DB:



Candidates (2):



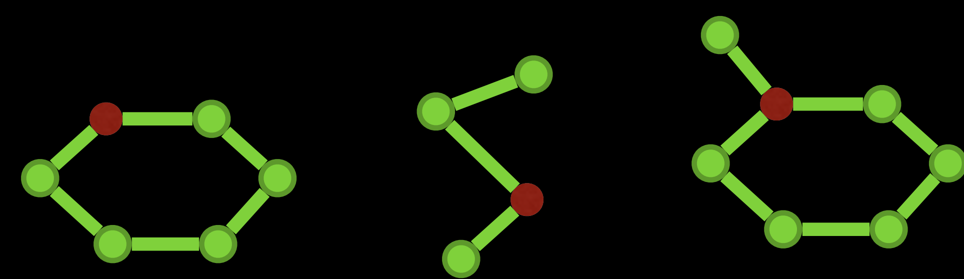
Occurrences: {A,B,C} and {A,B,C}

How Typical FGM Algos. Work

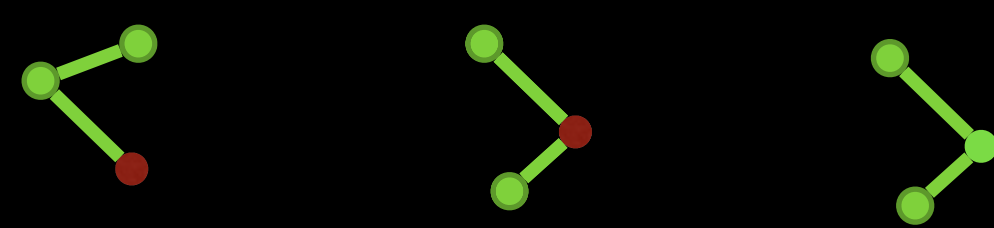
Example (t=2):

A B C

DB:



Candidates (3):



etc…

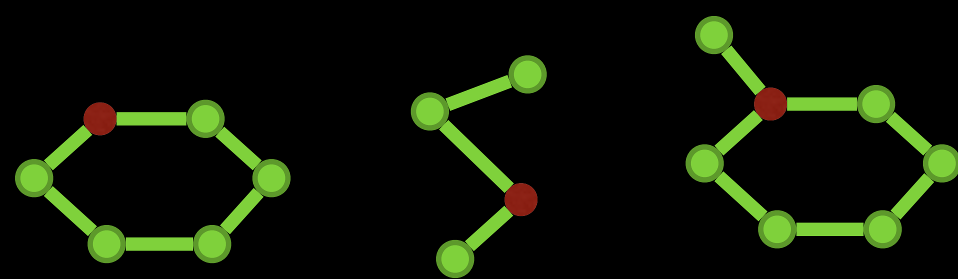
Occurrences: {A,B,C} and {A,B,C} and {A,C}

How Typical FGM Algos. Work

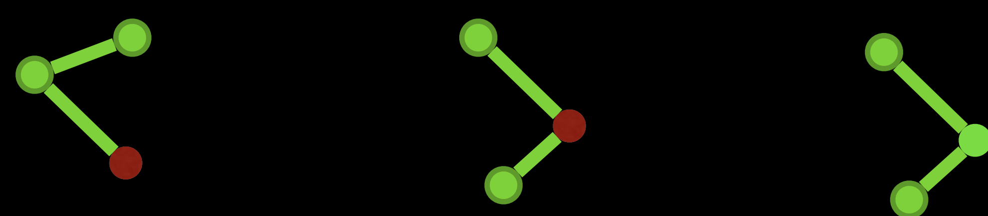
Example (t=2):

A B C

DB:



Candidates (3):



etc…

Occurrences: {A,B,C} and {A,B,C} and {A,C}

Such an algorithm needs to be able to:

- remove isomorphic candidates (iso. not known to be in P)
- compute occurrences using subgraph isomorphism (NP-hard)

Complexity of FGM

- **Complexity measures:**

  - **Polynomial delay:** if the time between printing the next fr. graph (or terminating) is bounded by a polynomial of the size of input,

    implies

  - **Incremental polynomial time:** if the time between printing next fr. graph (or terminating) is bounded by a polynomial of the size of input and of the size of output so far,

    implies

  - **Output polynomial time:** if the algorithm finishes in time polynomial in the combined size of input and the entire output.

Known Results

DB $\subseteq$ All graphs

NOT EVEN OUTPUT-POLY TIME POSSIBLE!

Interesting cases

DB $\subseteq$ Bounded-treewidth graphs

INCREMENTAL-POLY TIME!
[Horvath & Ramon, 2010]

Despite NP hard subgraph iso.

Poly delay???

???

Open questions!

DB $\subseteq$ Hereditary graph classes with poly-time subgraph iso.

POLY DELAY!

Change of Perspective

- **A more general problem (Ordered graph mining):**
  - Output all nonisomorphic connected graphs with frequency at least 1 and their occurrences in *DB* (*i.e. which DB graphs they match by subgraph iso.*):
    - **F -> I:** from frequent to infrequent (**generalizes FGM**)
    - **I -> F:** from infrequent to frequent (**generalizes IGM**)
    - **S -> L:** from smallest to largest
    - **L -> S:** from largest to smallest

(If you cannot solve a problem, George Pólya in "How to Solve It" suggests studying a more general problem.)

Available Results

(From correspondence between FGM and F -> I)

	All Graphs	Planar Graphs	Bounded-Treewidth Graphs
S -> L	??	??	IncPoly [Horvath and Ramon, 2010]
L -> S	??	??	??
F -> I	Not IncPoly unless P=NP [known]	??	IncPoly [Horvath and Ramon, 2010]
I -> F	??	??	??

New Results and Corollaries

Corollaries of our theorems

	All Graphs	Planar Graphs	Bounded-Treewidth Graphs
S -> L	Not IncPoly unless FPT = W[1]	??	IncPoly [Horvath and Ramon, 2010]
L -> S	IncrPoly iff GI in P, Poly delay if CANON in P	Poly delay	Poly delay
F -> I	Not IncPoly unless P=NP [known]	Not IncPoly unless P=NP	IncPoly [Horvath and Ramon, 2010]
I -> F	Not IncPoly unless P=NP	Not IncPoly unless P=NP	Not IncPoly unless P=NP

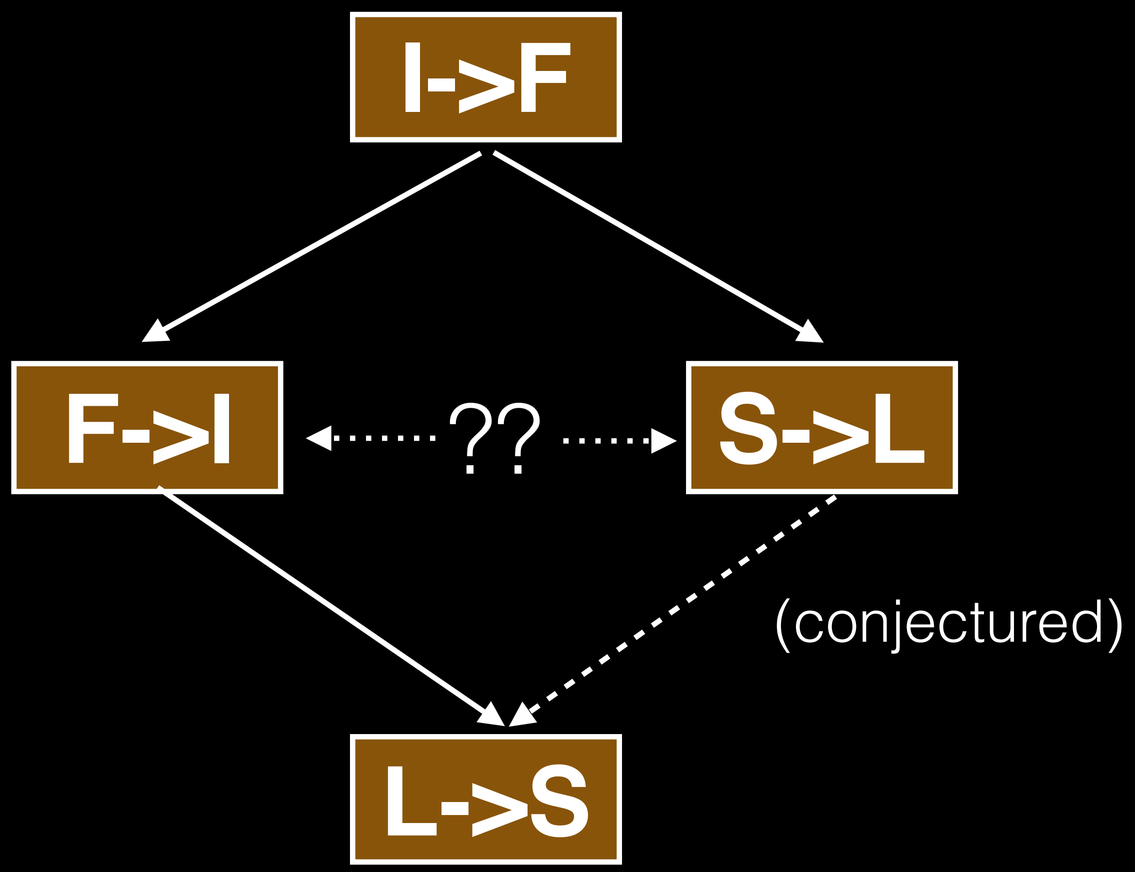
Positive

Negative

(More general results in the paper.)

Relative Hardness

- Difficulty of the problems for the considered classes of graphs:



Large to Small (Details)

Require: database DB of transaction graphs
Ensure: all connected (induced) subgraphs and their occurrences

let ALL be a data structure for storing graphs and their occurrences (as described in the main text).
for $G \in DB$ **do**
    $\mathsf{ADD}(G, \{\mathsf{ID}(G)\}, ALL)$
endfor
let m be the maximum order[1] of a graph in DB.
for $(l := m; l > 0;\ l := l - 1)$ **do**
    for $H \in \mathsf{KEYS}(l, ALL)$ **do**
        $OCC \leftarrow \mathsf{GET}(H, ALL)$
        $\mathsf{PRINT}(H, OCC)$
        for $H' \in \mathsf{REFINE}(H)$ **do**
            if H' is connected **then**
                $\mathsf{ADD}(H', OCC, ALL)$
            endif
        endfor
    endfor
    $\mathsf{DELETE}(l, ALL)$
endfor

- **Simple**, yet **poly-delay** algorithm for bounded TW graphs, planar graphs, ….
- It achieves poly-delay with NP-hard pattern matching operators and even if FGM cannot be solved in output-poly time (planar graphs).
- It may be combined with constraints such as maximum graph diameter which even leads to practical algorithms
- It can be generalised to **(induced) homeomorphism** and **(induced) minor emb.**

Conclusions

- **Theory:**
  - **New results for complexity** of graph mining with NP-hard pattern matching operators (**some pretty surprising**).
  - We have proved analogical results for **induced subgraph isomorphism**, **(induced) homeomorphism** and **(induced) minor embedding**

- **Practice:**
  - Both the **positive** and **negative** results give guidelines e.g. for developing practical subgraph kernels.
  - **Larger-to-smaller** algorithm:
    - practically useful for mining subgraphs of bounded diameter
    - surprisingly also useful for mining **all** induced subgraphs of molecules of up to 25 non-hydrogen atoms (+ bigger molecules with additional hacks)

Thank you!