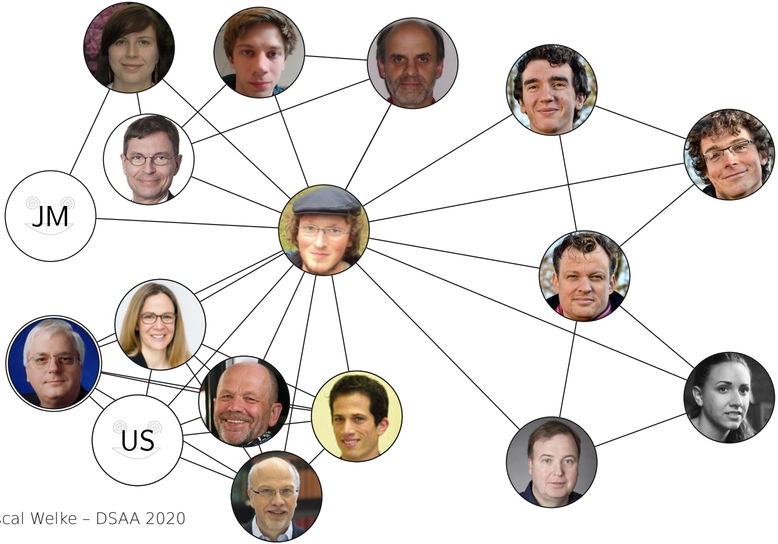
Efficient Frequent Subgraph Mining in Transactional Databases

Pascal Welke

DSAA 2020

UNIVERSITÄT BONN

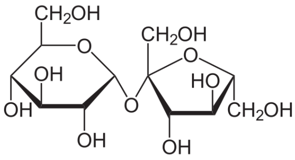
Example: Co-authorship Networks



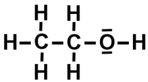
A *graph* $G = (V, E)$ consists of a set of vertices V and a set of edges E connecting pairs of vertices.

(Pictures taken from the home pages of the authors)

Example: Chemical Molecules



Saccharose

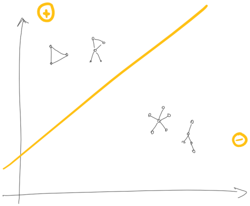


Ethanol

(commons.wikimedia.org)

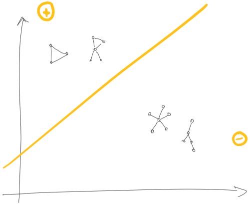
How to Learn From Graphs?

- Similarity based learning methods
  - "close by objects behave similarly"



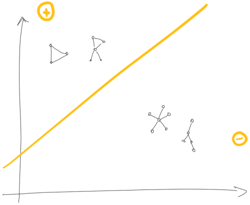
How to Learn From Graphs?

- Similarity based learning methods
  - "close by objects behave similarly"
  - $\rightarrow$ What does "close by" mean if objects are graphs?

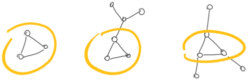


How to Learn From Graphs?

- Similarity based learning methods
  - "close by objects behave similarly"
  - $\rightarrow$ What does "close by" mean if objects are graphs?

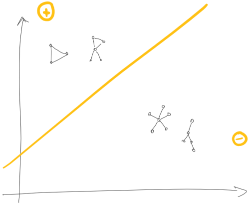


- Identification of relevant patterns

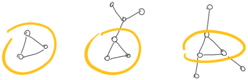


How to Learn From Graphs?

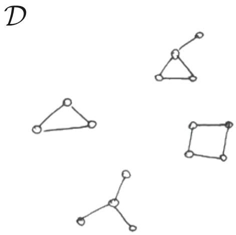
- Similarity based learning methods
  - "close by objects behave similarly"
  - $\rightarrow$ What does "close by" mean if objects are graphs?



- Identification of relevant patterns
  - What is a pattern?
  - What is relevant?

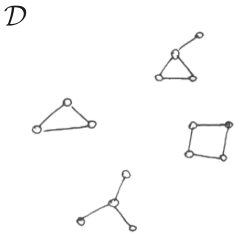


- Let's say we have a few graphs (in a graph database $\mathcal{D}$)



Frequent Subgraph Mining

- Let's say we have a few graphs (in a graph database $\mathcal{D}$)
- *Frequent subgraphs* are a reasonable choice to define similarities in a domain of graphs
  Deshpande et al (2005)



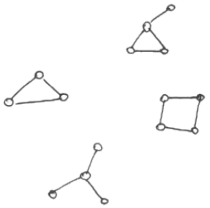
- Let's say we have a few graphs (in a graph database $\mathcal{D}$)
- *Frequent subgraphs* are a reasonable choice to define similarities in a domain of graphs

Deshpande et al (2005)

Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

$\mathcal{D}$



Frequent Subgraph Mining

- Let's say we have a few graphs (in a graph database $\mathcal{D}$)
- *Frequent subgraphs* are a reasonable choice to define similarities in a domain of graphs

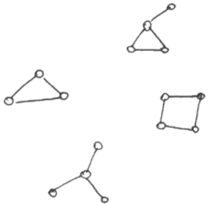
Deshpande et al (2005)

Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

List all connected graphs $P \in \mathcal{P}$ that are subgraph isomorphic to at least t graphs in $\mathcal{D}$.

$\mathcal{D}$



2-frequent subgraphs of $\mathcal{D}$

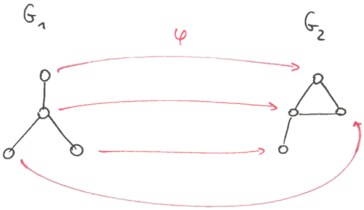
Definition

A *subgraph isomorphism* is an injective mapping

$$\varphi : V(G_1) \to V(G_2)$$

such that

$$(v_1, v_2) \in E(G_1) \Rightarrow (\varphi(v_1), \varphi(v_2)) \in E(G_2)$$



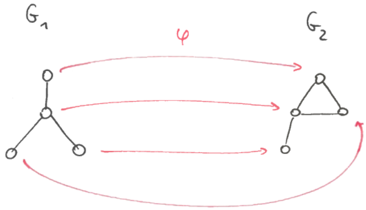
Definition

A *subgraph isomorphism* is an injective mapping

$$\varphi : V(G_1) \to V(G_2)$$

such that

$$(v_1, v_2) \in E(G_1) \Rightarrow (\varphi(v_1), \varphi(v_2)) \in E(G_2)$$



Deciding whether one exists, is *NP-hard*.

A brief history of exact FCSM algorithms

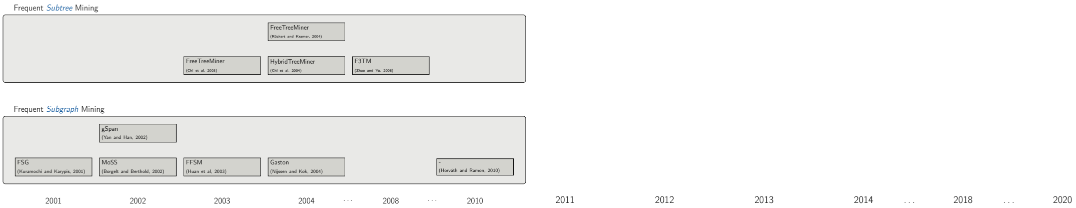
Frequent *Subtree* Mining

- FreeTreeMiner (Chi et al, 2003)
- FreeTreeMiner (Rückert and Kramer, 2004)
- HybridTreeMiner (Chi et al, 2004)
- F3TM (Zhao and Yu, 2008)

Frequent *Subgraph* Mining

- FSG (Kuramochi and Karypis, 2001)
- gSpan (Yan and Han, 2002)
- MoSS (Borgelt and Berthold, 2002)
- FFSM (Huan et al, 2003)
- Gaston (Nijssen and Kok, 2004)
- \- (Horváth and Ramon, 2010)

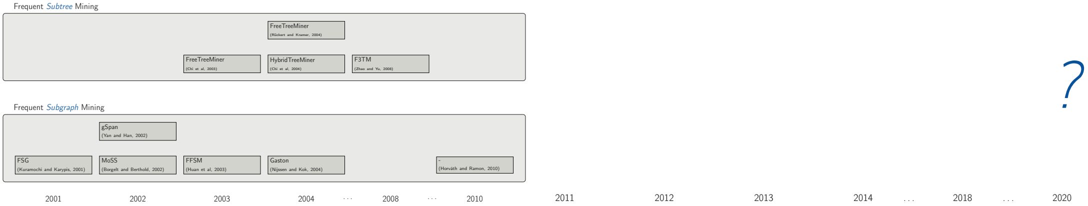
2001 2002 2003 2004 . . . 2008 . . . 2010



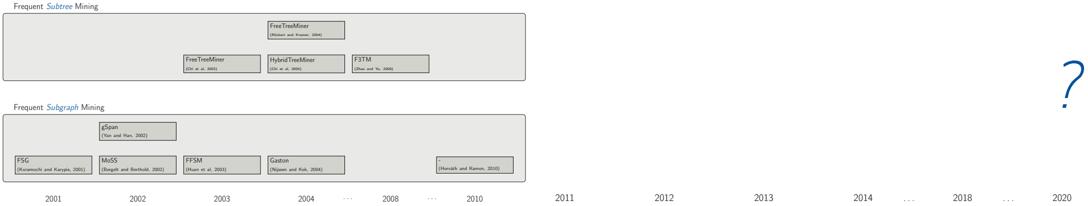
Frequent Subtree Mining
FreeTreeMiner
(Rückert and Kramer, 2004)
FreeTreeMiner
(Chi et al, 2003)
HybridTreeMiner
(Chi et al, 2004)
F3TM
(Zhao and Yu, 2008)
Frequent Subgraph Mining
gSpan
(Yan and Han, 2002)
FSG
(Kuramochi and Karypis, 2001)
MoSS
(Borgelt and Berthold, 2002)
FFSM
(Huan et al, 2003)
Gaston
(Nijssen and Kok, 2004)
-
(Horváth and Ramon, 2010)
2001
2002
2003
2004
...
2008
...
2010
2011
2012
2013
2014
...
2018
...
2020

A brief history of exact FCSM algorithms

Efficient Frequent Subgraph Mining in Transactional Databases



- All these methods enumerate the full set of frequent subtrees/subgraphs



- All these methods enumerate the full set of frequent subtrees/subgraphs
- ...so why are we here, ten years later and discussing about this?

A quick quizz

- An acquaintance of yours has 200 graphs

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each
  - 30-120 edges each

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each
  - 30-120 edges each
  - 100+ (vertex) labels

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each
  - 30-120 edges each
  - 100+ (vertex) labels
  - ...and wants to find frequent patterns with five or more vertices

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each
  - 30-120 edges each
  - 100+ (vertex) labels
  - ...and wants to find frequent patterns with five or more vertices
- can he do it?

A quick quizz

- An acquaintance of yours has 200 graphs
  - 10-30 vertices each
  - 30-120 edges each
  - 100+ (vertex) labels
  - ...and wants to find frequent patterns with five or more vertices
- can he do it?

Nope! Tinnes (2020)

Structure of this Talk

1. *Computational complexity* of frequent subgraph/subtree mining
   - what notions of efficiency are useful?
   - is there any hope?

Structure of this Talk

1. *Computational complexity* of frequent subgraph/subtree mining
   - what notions of efficiency are useful?
   - is there any hope?
2. *Exact algorithms* and their problems
   - a quick and general look at counting subgraph support

Structure of this Talk

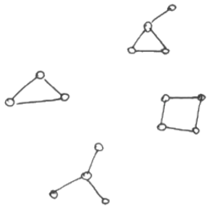
1. *Computational complexity* of frequent subgraph/subtree mining
   - what notions of efficiency are useful?
   - is there any hope?
2. *Exact algorithms* and their problems
   - a quick and general look at counting subgraph support
3. Some more current *inexact* solutions
   - *efficient* for *arbitrary* graph databases
   - though *incomplete*, comparable predictive performance to exact frequent subgraphs

Part 1.
Computational Complexity of Frequent Subgraph Mining

Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

$\mathcal{D}$

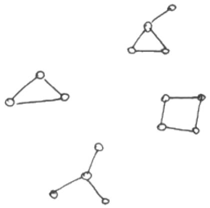


Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

List all connected graphs $P \in \mathcal{P}$ that are *subgraph isomorphic* to at least t graphs in $\mathcal{D}$.

$\mathcal{D}$



2-frequent subgraphs of $\mathcal{D}$

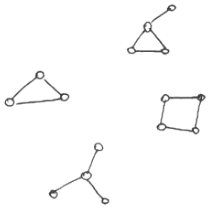
1. Frequent Subgraph Mining (Recall)

Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

List all connected graphs $P \in \mathcal{P}$ that are *subgraph isomorphic* to at least t graphs in $\mathcal{D}$.

$\mathcal{D}$



Subgraph Isomorphism is *NP-hard*.

2-frequent subgraphs of $\mathcal{D}$

UNIVERSITÄT BONN

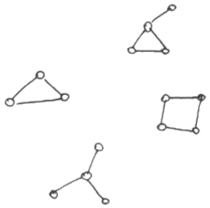
1. Frequent Subgraph Mining (Recall)

Frequent Connected Subgraph Mining (FCSM)

Given a dataset of graphs $\mathcal{D} \subseteq \mathcal{G}$ and an integer threshold $t \leq |\mathcal{D}|$

List all connected graphs $P \in \mathcal{P}$ that are *subgraph isomorphic* to at least t graphs in $\mathcal{D}$.

$\mathcal{D}$



Subgraph Isomorphism is *NP-hard*.
(There is a bit more to it Horváth et al (2007))

2-frequent subgraphs of $\mathcal{D}$

- There can be exponentially many frequent patterns
  - we need a bound in the output size

A few more detailed considerations

- There can be exponentially many frequent patterns
  - we need a bound in the output size
- Output polynomial time
  - finish in polynomial time in *input size and output size*

UNIVERSITÄT BONN

A few more detailed considerations

- There can be exponentially many frequent patterns
  - we need a bound in the output size
- Output polynomial time
  - finish in polynomial time in *input size and output size*
- Polynomial delay
  - *the time between finding two patterns* is polynomial in the input size

A few more detailed considerations

- There can be exponentially many frequent patterns
  - we need a bound in the output size
- Output polynomial time
  - finish in polynomial time in *input size and output size*
- Polynomial delay
  - *the time between finding two patterns* is polynomial in the input size
- Incremental polynomial time
  - something in between

Problems with FCSM

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) ...

Problems with FCSM

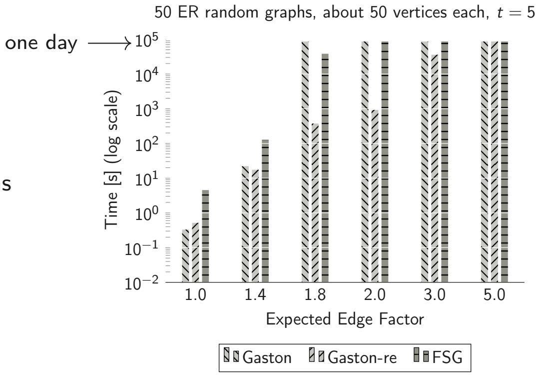
- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) ...
- *However:*

Problems with FCSM

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) ...
- *However:*
  - *only works* for (very simple, chemical) graphs

Problems with FCSM

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) . . .
- *However:*
  - *only works* for (very simple, chemical) graphs
  - *explodes* on most other datasets

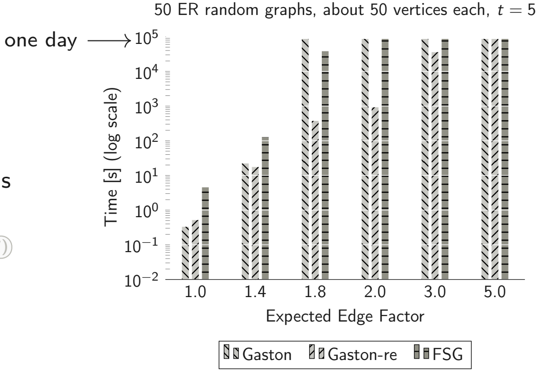


(Welke (2019))

UNIVERSITÄT BONN

Problems with FCSM

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) . . .
- *However:*
  - *only works* for (very simple, chemical) graphs
  - *explodes* on most other datasets
- *Computationally Intractable* (Horváth et al (2007))

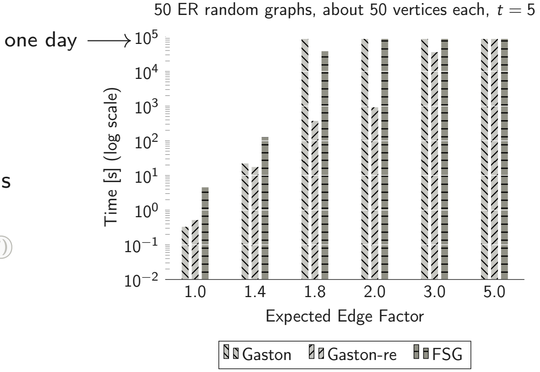


(Welke (2019))

UNIVERSITÄT BONN

Problems with FCSM

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) . . .
- *However:*
  - *only works* for (very simple, chemical) graphs
  - *explodes* on most other datasets
- *Computationally Intractable* (Horváth et al (2007))
  - previous work deals with this using some properties of very simple graphs

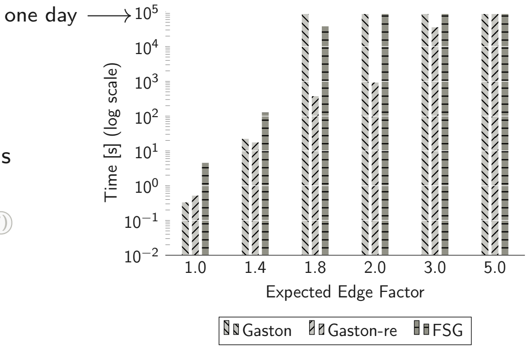


(Welke (2019))

UNIVERSITÄT BONN

- Software exists, e.g.
  - FSG (Kuramochi and Karypis (2001))
  - gSpan (Yan and Han (2002))
  - Gaston (Nijssen and Kok (2005)) ...
- *However:*
  - *only works* for (very simple, chemical) graphs
  - *explodes* on most other datasets
- *Computationally Intractable* (Horváth et al (2007))
  - previous work deals with this using some properties of very simple graphs

50 ER random graphs, about 50 vertices each, $t = 5$



(Welke (2019))

⇒ *There is no system that can reliably mine all frequent subgraphs for arbitrary graph databases of small to medium sized graphs*

Part 2.

Exact Frequent Subgraph Mining Algorithms (& Problems)

The Timeline, again...

Frequent *Subtree* Mining

FreeTreeMiner
(Chi et al, 2003)

FreeTreeMiner
(Rückert and Kramer, 2004)

HybridTreeMiner
(Chi et al, 2004)

F3TM
(Zhao and Yu, 2008)

Frequent *Subgraph* Mining

FSG
(Kuramochi and Karypis, 2001)

gSpan
(Yan and Han, 2002)

MoSS
(Borgelt and Berthold, 2002)

FFSM
(Huan et al, 2003)

Gaston
(Nijssen and Kok, 2004)

-
(Horváth and Ramon, 2010)

2001 2002 2003 2004 ... 2008 ... 2010

The Timeline, again...

Frequent *Subtree* Mining

2001	2002	2003	2004	...	2008	...	2010
			FreeTreeMiner (Rückert and Kramer, 2004)				
		FreeTreeMiner (Chi et al, 2003)	HybridTreeMiner (Chi et al, 2004)		F3TM (Zhao and Yu, 2008)		

Frequent *Subgraph* Mining

2001	2002	2003	2004	...	2008	...	2010
	gSpan (Yan and Han, 2002)						
FSG (Kuramochi and Karypis, 2001)	MoSS (Borgelt and Berthold, 2002)	FFSM (Huan et al, 2003)	Gaston (Nijssen and Kok, 2004)				- (Horváth and Ramon, 2010)

polynomial delay | incr. polynomial time | exponential in output size

UNIVERSITÄT BONN

The Timeline, again...

Frequent *Subtree* Mining

- only for forest transactions: FreeTreeMiner (Chi et al, 2003)
- FreeTreeMiner (Rückert and Kramer, 2004)
- only for forest transactions: HybridTreeMiner (Chi et al, 2004)
- F3TM (Zhao and Yu, 2008)

Frequent *Subgraph* Mining

- FSG (Kuramochi and Karypis, 2001)
- gSpan (Yan and Han, 2002)
- MoSS (Borgelt and Berthold, 2002)
- FFSM (Huan et al, 2003)
- Gaston (Nijssen and Kok, 2004)
- not implemented: – (Horváth and Ramon, 2010)

2001 2002 2003 2004 ... 2008 ... 2010

polynomial delay | incr. polynomial time | exponential in output size

Part 3.

Efficient Inexact Mining Methods

1. Make the database simpler or smaller (Chen et al (2009)) (Welke et al (2019))

Possible Ways Out

1. Make the database simpler or smaller (Chen et al (2009)) (Welke et al (2019))
2. Change the embedding operator (Li and Wang (2015)) (Schulz et al (2018))

Possible Ways Out

1. Make the database simpler or smaller (Chen et al (2009)) (Welke et al (2019))
2. Change the embedding operator (Li and Wang (2015)) (Schulz et al (2018))
3. Restrict the pattern language and allow one-sided error (Schulz et al (2018)) (Welke et al (2019)) (Welke et al (2020))

A More Recent Timeline

Frequent *Subtree* Mining

- –
  (Schulz et al, 2018)
- PS
  (Welke et al, 2018)
- BPS
  (Welke et al, 2019)
- HOPS
  (Welke et al, 2020)

Frequent *Subgraph* Mining

- SUMMARIZE-MINE
  (Chen et al, 2009)
- MUSE
  (Zou et al, 2010)
- REAFUM
  (Li and Wang, 2015)

2009 2010 ··· 2015 ··· 2018 2019 2020

polynomial delay

exponential in output size

Conclusion & Outlook

- Frequent subgraph mining is an inherently difficult problem
  - not possible in output polynomial time

Conclusion & Outlook

- Frequent subgraph mining is an inherently difficult problem
  - not possible in output polynomial time
- Exact methods exist for a long time, but they only work on simple graph databases
  - use *output exponential* time & space

Conclusion & Outlook

- Frequent subgraph mining is an inherently difficult problem
  - not possible in output polynomial time
- Exact methods exist for a long time, but they only work on simple graph databases
  - use *output exponential* time & space
- Relaxations of the problem result in efficient algorithms
  - polynomial delay
  - restriction to tree patterns
  - one-sided error

Conclusion & Outlook

- Frequent subgraph mining is an inherently difficult problem
  - not possible in output polynomial time
- Exact methods exist for a long time, but they only work on simple graph databases
  - use *output exponential* time & space
- Relaxations of the problem result in efficient algorithms
  - polynomial delay
  - restriction to tree patterns
  - one-sided error
- Can we combine exact and approximate tools to achieve “one-size-fits-all” mining?

Conclusion & Outlook

- Frequent subgraph mining is an inherently difficult problem
  - not possible in output polynomial time
- Exact methods exist for a long time, but they only work on simple graph databases
  - use *output exponential* time & space
- Relaxations of the problem result in efficient algorithms
  - polynomial delay
  - restriction to tree patterns
  - one-sided error
- Can we combine exact and approximate tools to achieve "one-size-fits-all" mining?
- Can we devise efficient methods for non-tree patterns?

References I

C Borgelt, M Berthold (2002) Mining molecular fragments: Finding relevant substructures of molecules. In: ICDM, doi: 10.1109/ICDM.2002.1183885

C Chen, C Lin, M Fredrikson, M Christodorescu, X Yan, J Han (2009) Mining graph patterns efficiently via randomized summaries. PVLDB 2(1):742–753, URL http://www.vldb.org/pvldb/2/vldb09-80.pdf

Y Chi, Y Yang, R Muntz (2003) Indexing and mining free trees. In: ICDM

Y Chi, Y Yang, R Muntz (2004) HybridTreeMiner: an efficient algorithm for mining frequent rooted trees and free trees using canonical forms. In: SSDBM, doi: 10.1109/SSDM.2004.1311189

M Deshpande, M Kuramochi, N Wale, G Karypis (2005) Frequent substructure-based approaches for classifying chemical compounds. TKDE 17(8):1036–1050, doi: 10.1109/tkde.2005.127

T Horváth, J Ramon (2010) Efficient frequent connected subgraph mining in graphs of bounded tree-width. Theor Comp Sci 411(31–33):2784–2797, doi: 10.1016/j.tcs.2010.03.030

T Horváth, B Bringmann, L De Raedt (2007) Frequent hypergraph mining. In: ILP, Springer, doi: 10.1007/978-3-540-73847-3_26

J Huan, W Wang, J Prins (2003) Efficient mining of frequent subgraphs in the presence of isomorphism. In: ICDM, doi: 10.1109/ICDM.2003.1250974

M Kuramochi, G Karypis (2001) Frequent subgraph discovery. In: ICDM, doi: 10.1109/ICDM.2001.989534

R Li, W Wang (2015) REAFUM: representative approximate frequent subgraph mining. In: SDM, doi: 10.1137/1.9781611974010.85

S Nijssen, J Kok (2004) A quickstart in frequent structure mining can make a difference. In: KDD, doi: 10.1145/1014052.1014134

S Nijssen, J Kok (2005) The gaston tool for frequent subgraph mining. Electronic Notes in Theor Comp Sci 127(1):77–87, doi: 10.1016/j.entcs.2004.12.039

U Rückert, S Kramer (2004) Frequent free tree discovery in graph data. In: SAC, doi: 10.1145/967900.968018

References II

T Schulz, T Horváth, P Welke, S Wrobel (2018) Mining tree patterns with partially injective homomorphisms. In: ECMLPKDD, doi: 10.1007/978-3-030-10928-8_35

C Tinnes (2020) Personal communication

P Welke (2019) Efficient frequent subtree mining beyond forests. PhD thesis, University of Bonn

P Welke, T Horváth, S Wrobel (2018) Probabilistic frequent subtrees for efficient graph classification and retrieval. Mach Learn 107(11):1847–1873, doi: 10.1007/s10994-017-5688-7

P Welke, T Horváth, S Wrobel (2019) Probabilistic and exact frequent subtree mining in graphs beyond forests. Mach Learn 108(7):1137–1164, doi: 10.1007/s10994-019-05779-1

Pascal Welke, Florian Seiffarth, Michael Kamp, Stefan Wrobel (2020) HOPS: Probabilistic subtree mining for small and large graphs. In: KDD, doi: 10.1145/3394486.3403180

X Yan, J Han (2002) gSpan: Graph-based substructure pattern mining. In: ICDM, doi: 10.1109/icdm.2002.1184038

P Zhao, J Yu (2008) Fast frequent free tree mining in graph databases. WWW 11(1):71–92, doi: 10.1007/s11280-007-0031-z

Z Zou, J Li, H Gao, S Zhang (2010) Mining frequent subgraph patterns from uncertain graph data. TKDE 22(9):1203–1218, doi: 10.1109/tkde.2010.80