Mining Frequent Trees in Networks

KATHOLIEKE UNIVERSITEIT LEUVEN

DTAI
Declaratieve Talen en Artificiële Intelligentie

Problem

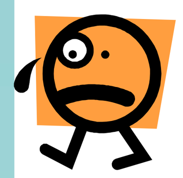
Frequent Tree Mining:

Given a graph G and threshold t, find and list all trees that **occur** in G at least **t** times

Setting:

- Single large networks
- Subgraph isomorphism as matching operator
- Pattern: Labeled rooted trees

Challenges



- Deciding whether a pattern is subgraph isomorphic to G itself is NP-Complete (naïve solution is **O(n^k)**)
- How to measure frequency?
- Main focus was to tackle computational complexity of the problem

Proposal

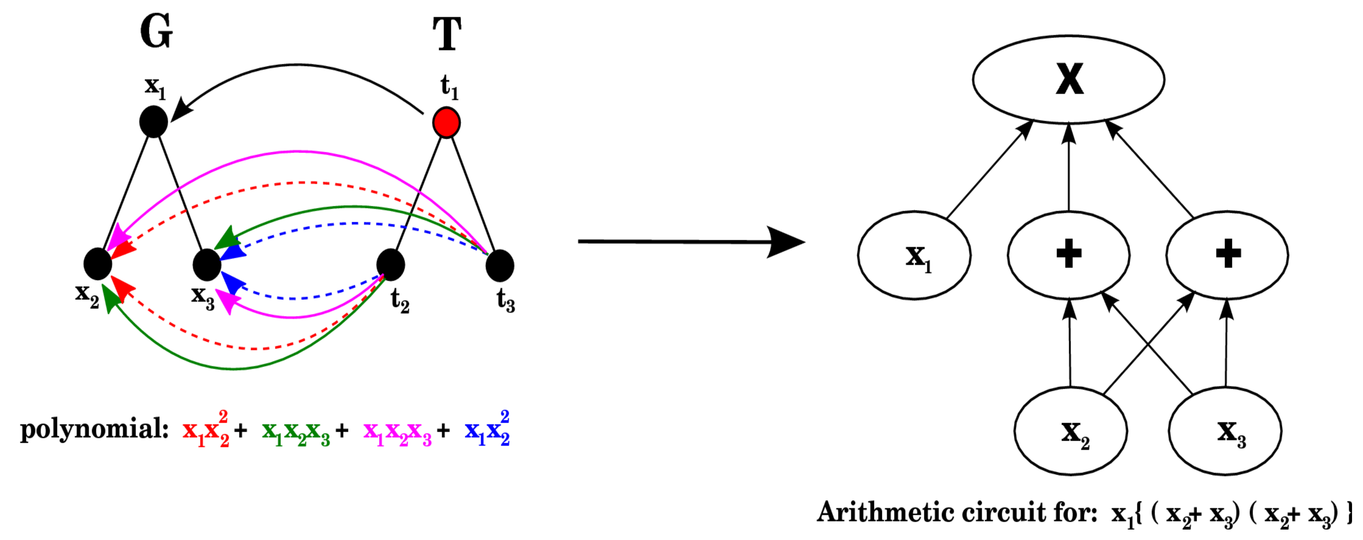
Researcher
Ashraf M. Kibriya

Promoter
Jan Ramon

- Recent advances in parameterized complexity theory, give us (randmized) algebraic methods that for tree patterns, reduce subgraph isomorphism from **O(n^k)** to **O($m.k^2.2^k.\log^2 k$)**
- Combined with a simple anti-monotone frequency measure, we can mine *all* tree patterns with similar complexity, in **delay O($|\Sigma|.m.k^2.2^k.\log^2 k$)**

  → **Frequency:** *Images* of root under isomorphism
- Accuracy can be boosted arbitrarily by repeating subgraph isomorphism multiple times
- Exponential factors are relatively small, to allow for practical applications

Method Overview



Evaluate over $GF(2^{(3+\log k)})[\mathbb{Z}_2^k]$

Delay time:

The time between any two consecutive outputs (including start and 1st output) is an unchanging delay

Results

Sponsors

erc

1) What pattern/networks size can we handle within reasonable time?
   → up to trees size 15 in 10^4 and size 10 in 10^6 networks
2) How does our strategy compare to state of the art methods?
   → outperforms VF2 even for moderate size problems
3) Does implementation really scale as well as given theoretical bounds?
   → approximately, yes
4) What is the influence of different parameters and optimizations?
   → See right