GRAPH+AI SUMMIT organized by TigerGraph

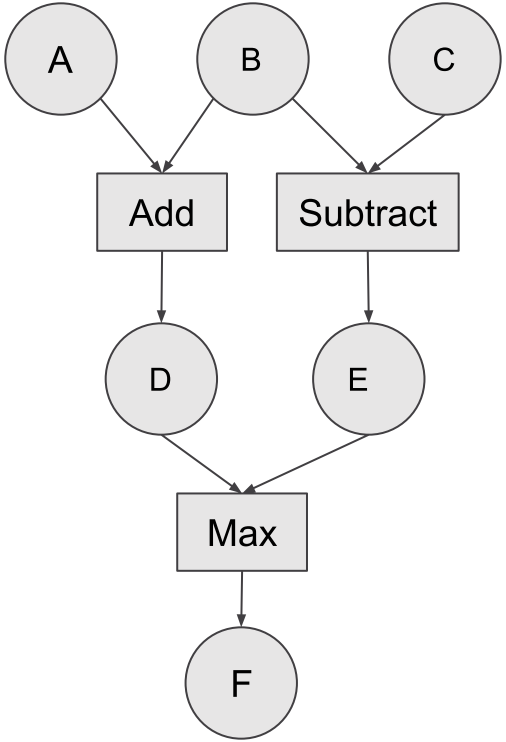
Mining Frequent Subgraphs on the Tax Knowledge Graph

Lalla Mouatadid

Research Scientist

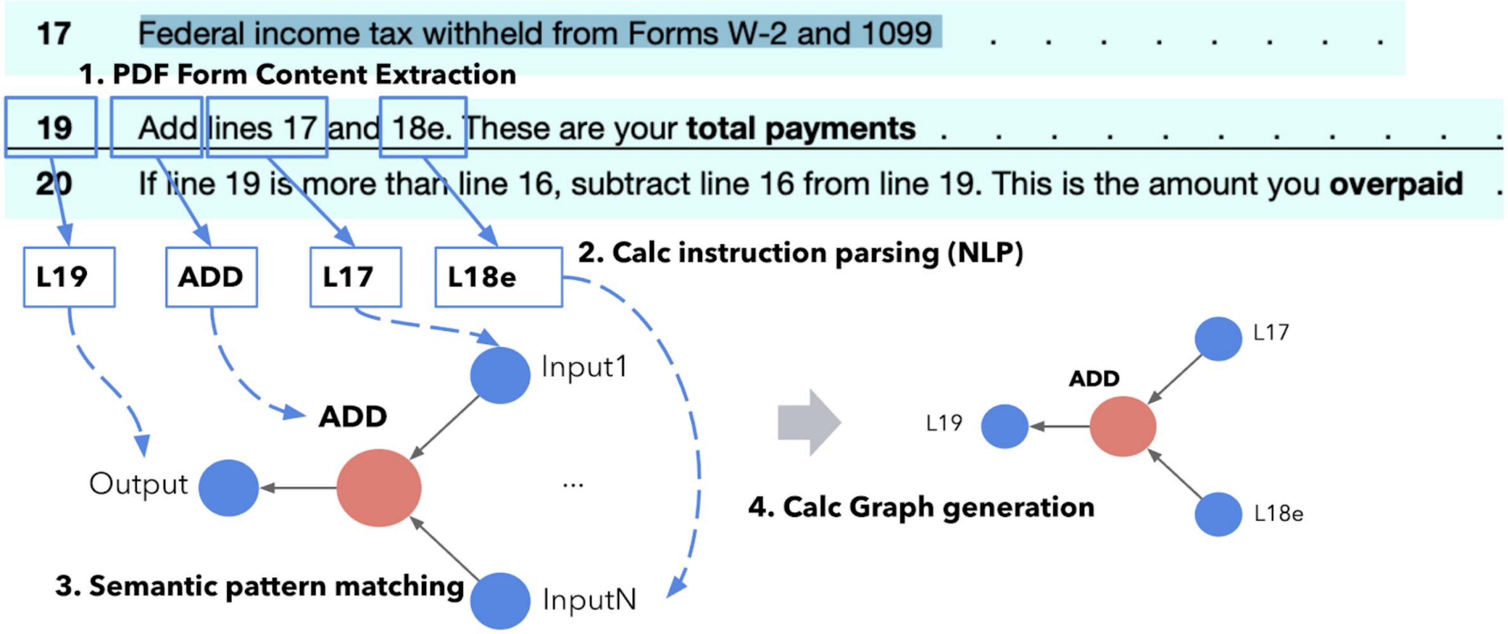
intuit

Declarative Representation of Tax Calculation Rules



- ❑ A set of basic tax specific calculation operations
- ❑ Explanation templates are attached to operations
- ❑ A set of tax form fields
- ❑ Calculations are represented as a declarative acyclic graph
- ❑ Nodes that are not calculated are user entered

Tax Knowledge Graph Construction

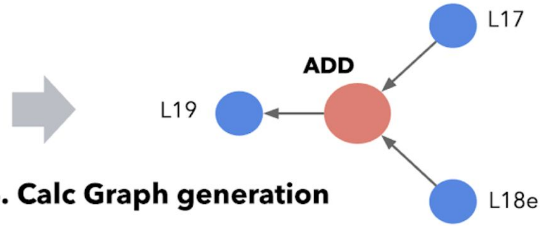


17 Federal income tax withheld from Forms W-2 and 1099 . . . .

19 Add lines 17 and 18e. These are your **total payments** . . . .

20 If line 19 is more than line 16, subtract line 16 from line 19. This is the amount you **overpaid** .

1. PDF Form Content Extraction
L19
ADD
L17
L18e
2. Calc instruction parsing (NLP)
Input1
ADD
Output
...
InputN
3. Semantic pattern matching



Paper: **Tax Knowledge Graph for a Smarter and More Personalized TurboTax, KDD20, Yu et al.**
https://arxiv.org/abs/2009.06103

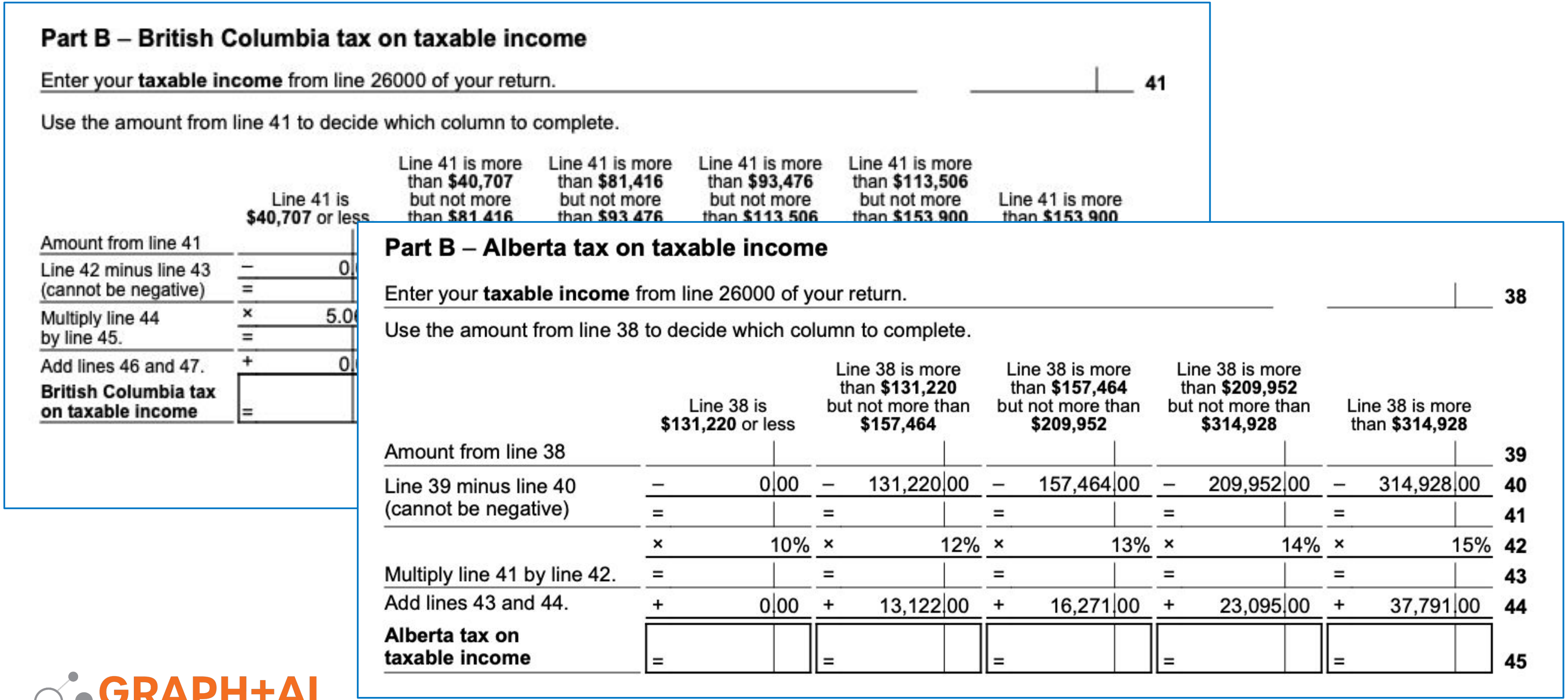
Duplicate Tax Calculations

- ❑ Atomic calculations allow for easy generation of the graph, but the graph is verbose and unnecessarily large.
- ❑ The Tax code gets updated often.
- ❑ Updates and maintenance are expensive and error-prone.

Mine similar repeated Calc patterns to:

- ❑ Tie them together to a “super” Calc
- ❑ Automate and synchronize changes
- ❑ Updating the super Calc to update all its occurrences
- ❑ **Speed & Accuracy & Simplicity**
- ❑ No loss of explanation!

Duplicate Tax Calculations



Part B – British Columbia tax on taxable income

Enter your **taxable income** from line 26000 of your return. 41

Use the amount from line 41 to decide which column to complete.

	Line 41 is **$40,707** or less	Line 41 is more than **$40,707** but not more than **$81 416**	Line 41 is more than **$81,416** but not more than **$93 476**	Line 41 is more than **$93,476** but not more than **$113 506**	Line 41 is more than **$113,506** but not more than **$153 900**	Line 41 is more than **$153 900**
Amount from line 41						
Line 42 minus line 43	– 0.(					
(cannot be negative)	=					
Multiply line 44	× 5.0(					
by line 45.	=					
Add lines 46 and 47.	+ 0.(					
British Columbia tax on taxable income	=					

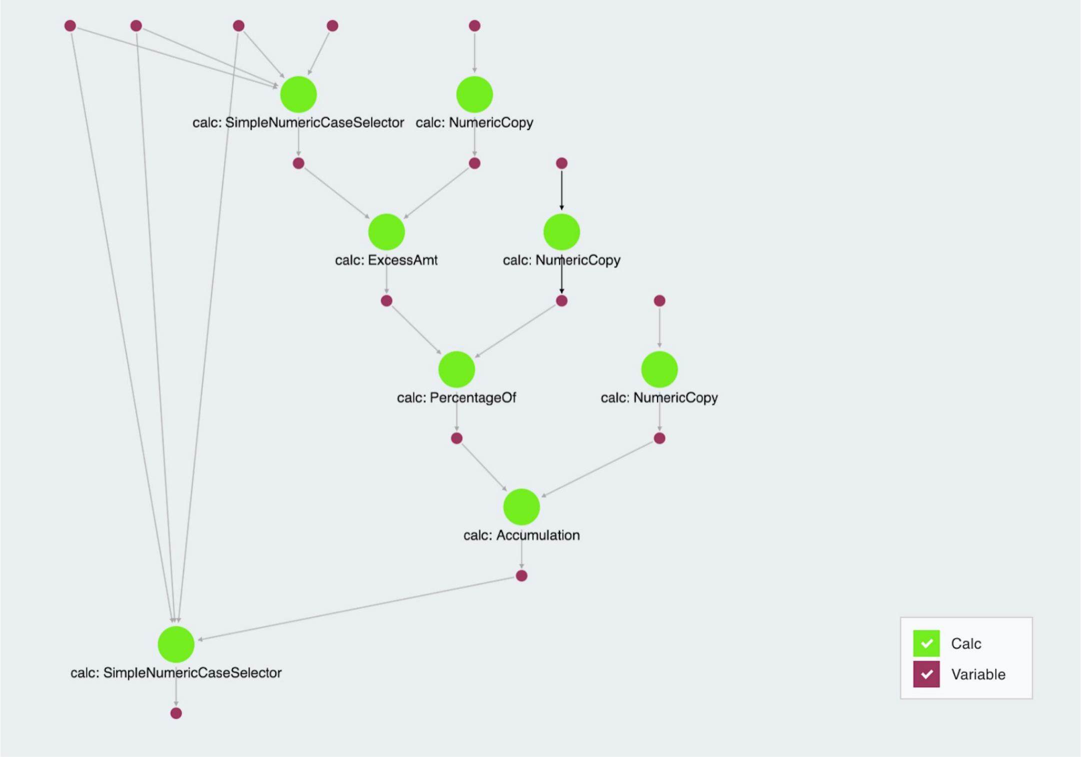
Part B – Alberta tax on taxable income

Enter your **taxable income** from line 26000 of your return. 38

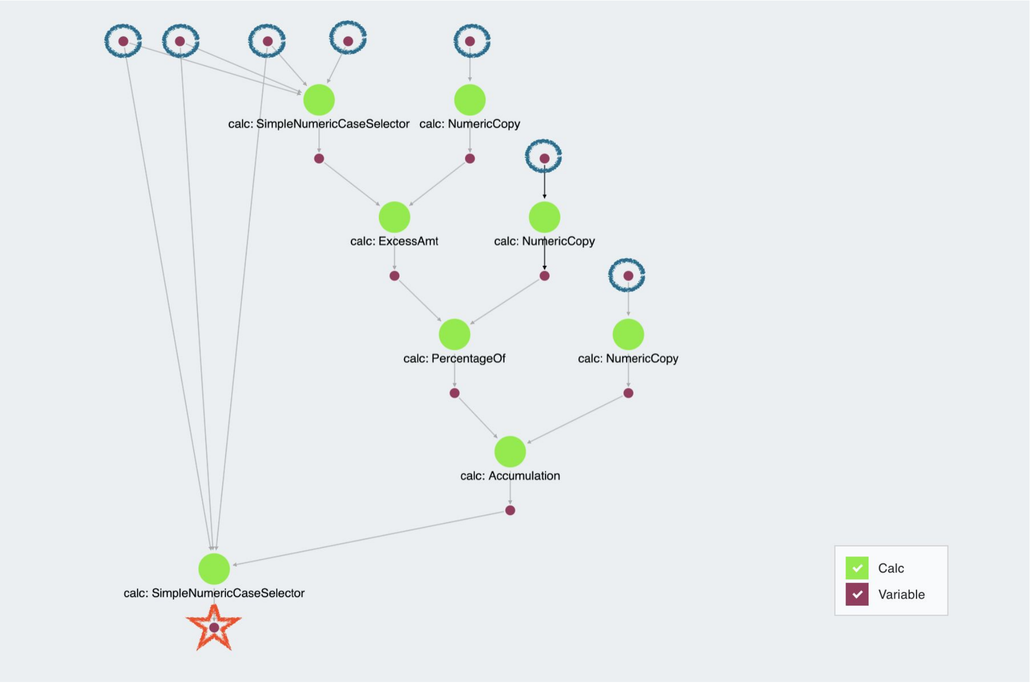
Use the amount from line 38 to decide which column to complete.

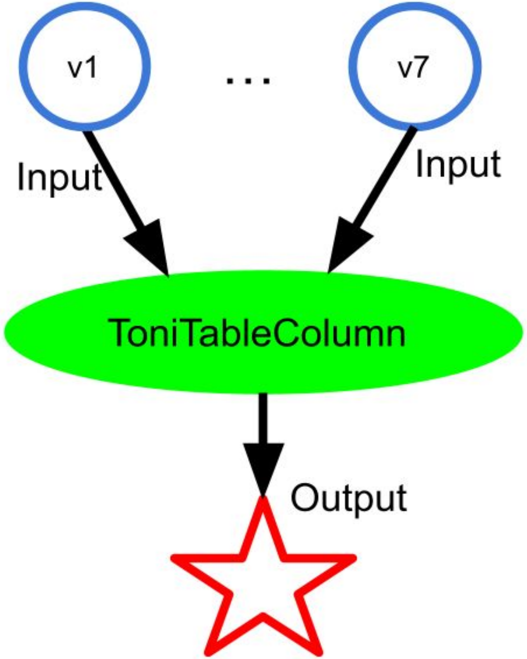
	Line 38 is **$131,220** or less	Line 38 is more than **$131,220** but not more than **$157,464**	Line 38 is more than **$157,464** but not more than **$209,952**	Line 38 is more than **$209,952** but not more than **$314,928**	Line 38 is more than **$314,928**	
Amount from line 38						39
Line 39 minus line 40	– 0|00	– 131,220|00	– 157,464|00	– 209,952|00	– 314,928|00	40
(cannot be negative)	=	=	=	=	=	41
	× 10%	× 12%	× 13%	× 14%	× 15%	42
Multiply line 41 by line 42.	=	=	=	=	=	43
Add lines 43 and 44.	+ 0|00	+ 13,122|00	+ 16,271|00	+ 23,095|00	+ 37,791|00	44
Alberta tax on taxable income	=	=	=	=	=	45

The TONI Column Subgraph

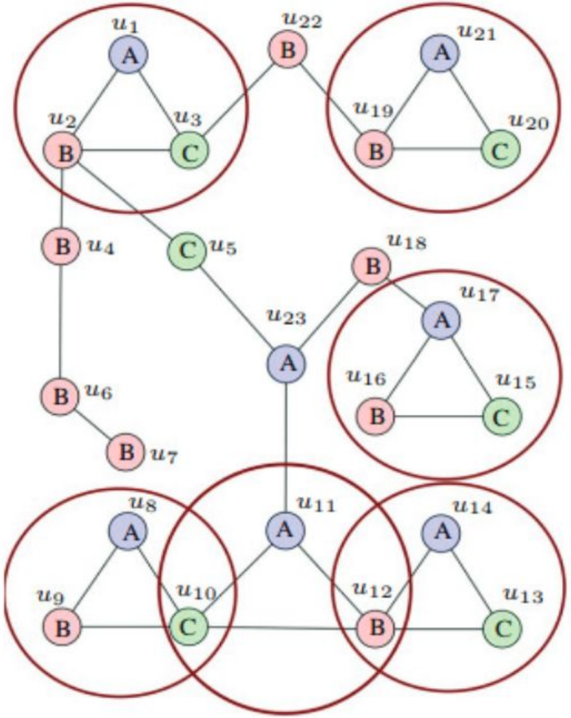


Factoring Calc Patterns in the Tax Graph





Detour: Frequent Subgraph Mining



- ❑ Detect similar patterns to:
  - ❑ Identify ‘like-me’ patterns for better, more accurate predictions.
  - ❑ Find redundant data, factorize the graph.
- ❑ Discover unknown entities.
- ❑ Uncover latent relationships.

Detour: Frequent Subgraph Mining

- The problem of identifying frequent subgraphs in a large network reduces to two main steps:
  - Generate the candidates
  - Check their frequency
- The second step is *subgraph isomorphism*, an NP-hard problem.
- This leaves the first step as the only (realistic in our lifetime) place to optimize frequent subgraphs algorithms.
- Two main techniques:
  - Apriori
  - Pattern Growth

Detour: Frequent Subgraph Mining

- **Apriori:**
  - Mine all subgraphs of size k-1 first, in order to mine subgraphs of size k.
  - Must use BFS.
  - Costly candidate generation (generation of duplicates)
  - Costly graph isomorphism (generation of false positives)
- **Pattern Growth:**
  - Extend a pattern by an edge (backward or forward) and check against this new pattern.
  - Flexible: Can use BFS or DFS (latter uses less memory)
  - Simple but computationally inefficient: Duplicate subgraphs generated over and over again.

Detour: Frequent Subgraph Mining

gSpan Algorithm:

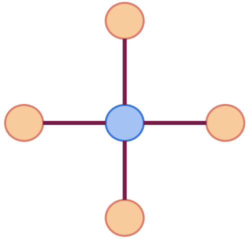
- Construction of canonical DFS codes based on DFS trees:
    - It reduces the generation of duplicate graphs.
    - No need to search previously discovered frequent subgraphs (reducing the search space as the algorithm progresses).
    - Completeness guaranteed without extending any duplicates.
- Every graph owns a canonical DFS code (a “minimal” DFS code).
- Two graphs are isomorphic if and only if they have the same canonical DFS codes.

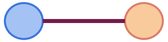
Detour: Frequent Subgraph Mining

- **Graph Dataset** setting:

  Given a collection **C** of graphs and a threshold **t**, return subgraphs that appear in at least **t** graphs in **C**.
- **Why a collection of graphs?**

  To avoid the following scenario:



How many  are there? 1 or 4?

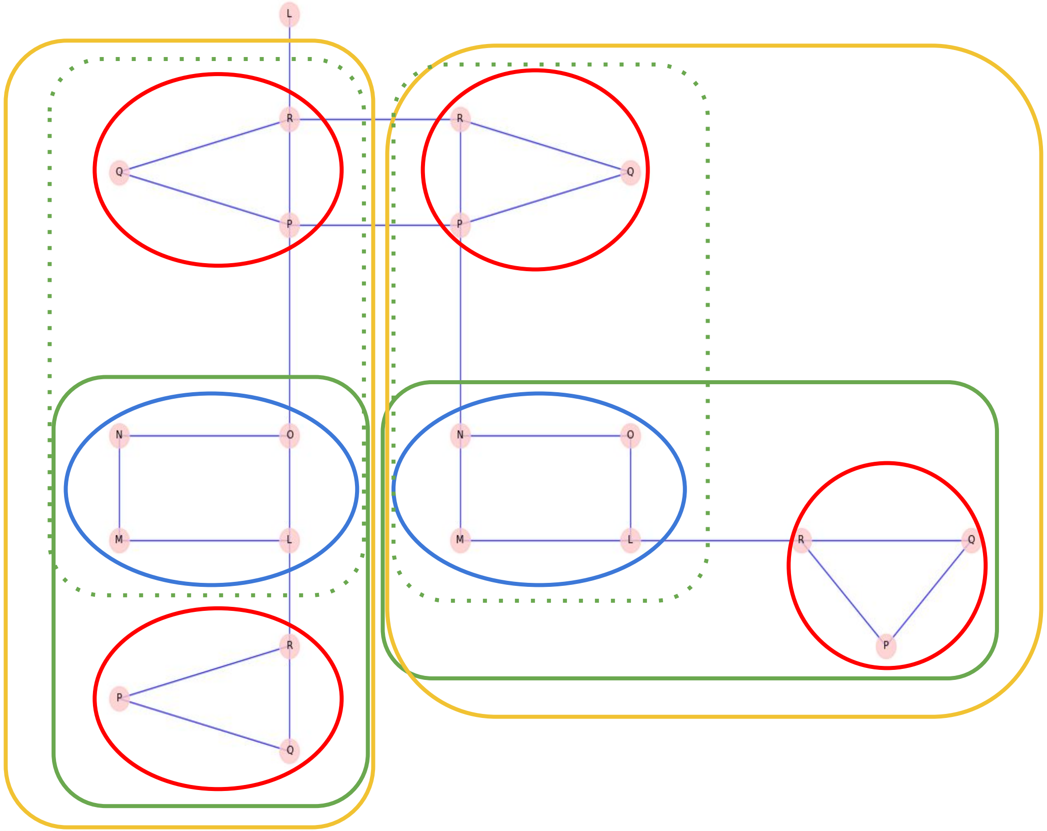
- **Problems:**
  - Does not count subgraphs within the same graph.
  - We have one large Tax graph.

- **Single Graph** setting:

  Given *one* large graph **G** and a threshold **t**, return subgraphs that appear at least **t** times in **G**.

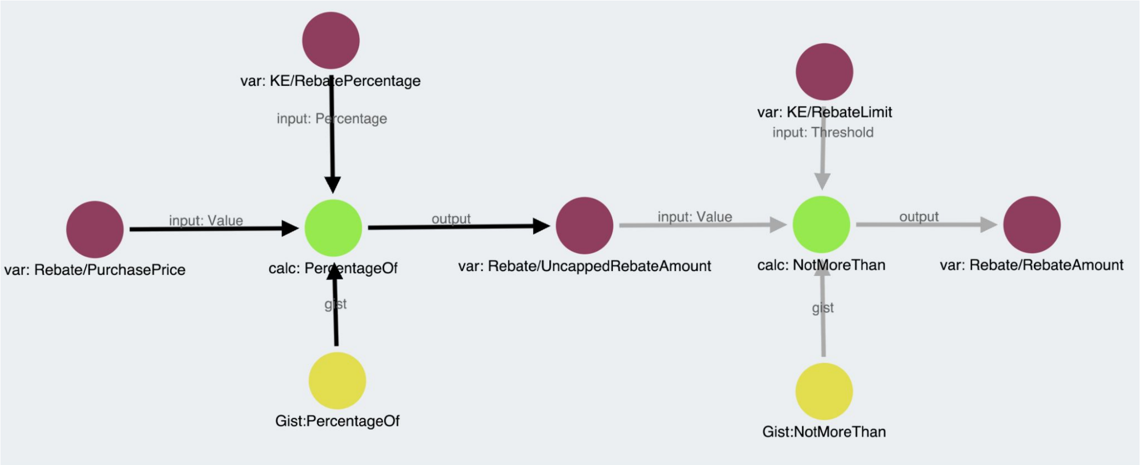
- **GraMi Algorithm**, Elseidy et al. PVDLB’14.

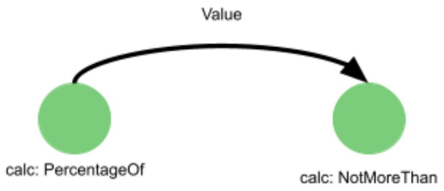
Detour: Frequent Subgraph Mining



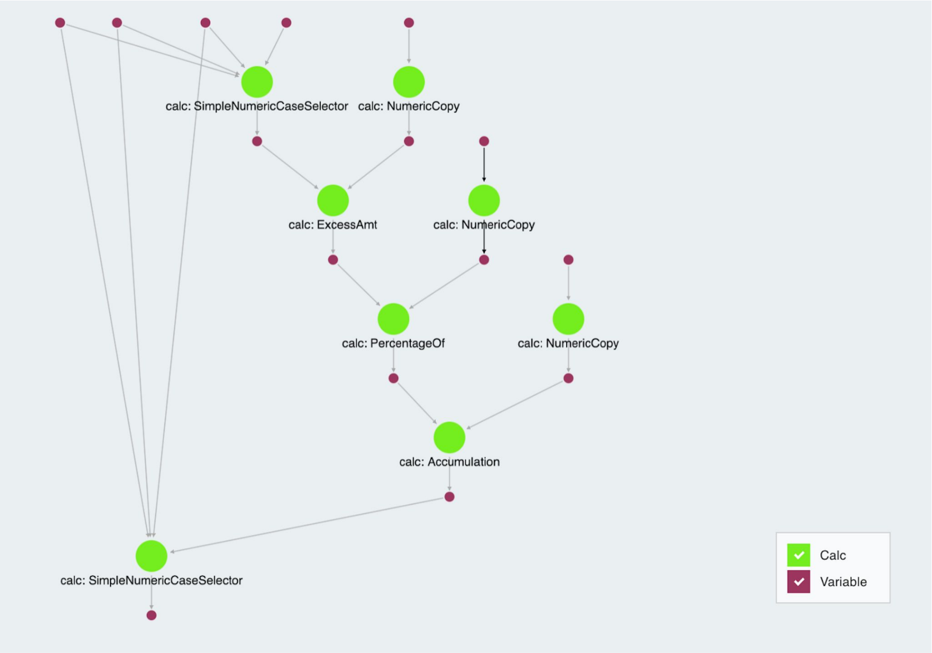
Detour: Frequent Subgraph Mining

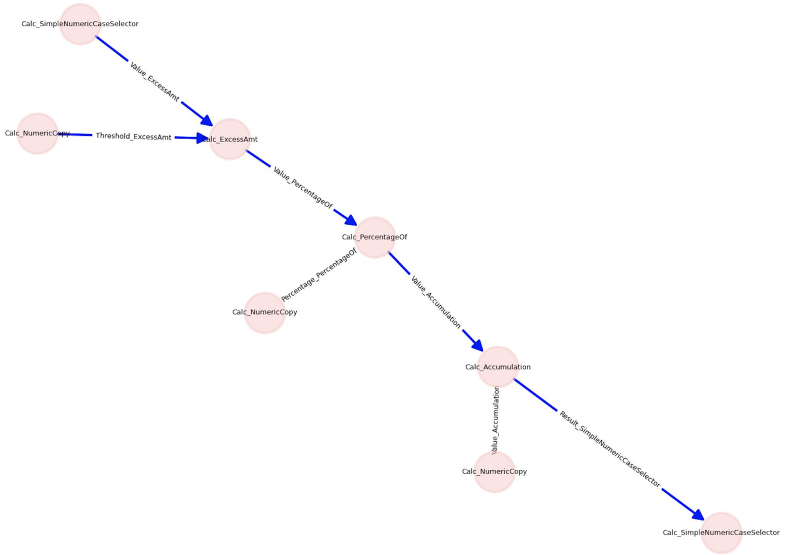
- Build an auxiliary **Calc2Calc** graph:





Repeating Calc Patterns in the Tax Graph





Conclusion

- ❑ Frequent subgraph mining is an NP-hard problem.
- ❑ Refine the graph using the heterogeneous nature of your KG.
- ❑ Useful when the topology of the graph is more important.
- ❑ Explanation is easy and recoverable.
- ❑ But, doesn’t scale well to very large graphs.

Thank You.