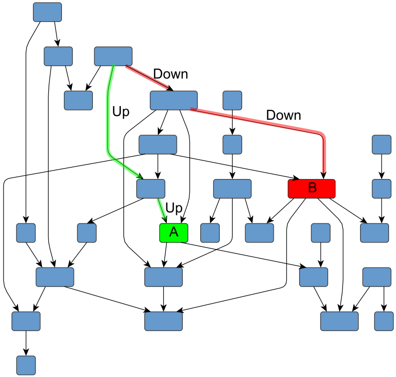
Context-Free Path Querying by Matrix Multiplication

Rustam Azimov, Semyon Grigorev
presents Kate Verbitskaia

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

June 10, 2018

Language-Constrained Path Filtering



Navigation through a graph

- Are nodes A and B on the same level of hierarchy?
- Is there a path of form **Up**^{n} **Down**^{n}?
- Find all paths of form **Up**^{n} **Down**^{n} which start from the node A

Language-Constrained Path Filtering: Relational Query Semantics

- $\mathbb{G} = (\Sigma, N, P)$ — context-free grammar in normal form
  - $A \to BC$, where $A, B, C \in N$
  - $A \to x$, where $A \in N, x \in \Sigma$
- $G = (V, E, L)$ — directed graph
  - $v \xrightarrow{l} u \in E$
  - $L \subseteq \Sigma$
- $\omega(\pi) = \omega(v_0 \xrightarrow{l_0} v_1 \xrightarrow{l_1} \cdots \xrightarrow{l_{n-2}} v_{n-1} \xrightarrow{l_{n-1}} v_n) = l_0 l_1 \cdots l_{n-1}$
- $R_A = \{(n, m) \mid \exists n \pi m$, such that $\omega(\pi) \in L(\mathbb{G}, A)\}$

Regular language constraints

- Widely spread
  - Graph databases query languages (SPARQL, Cypher, PGQL)
  - Network analysis
- Still in active development
  - OpenCypher: `https://goo.gl/5h5a8P`
  - Scalability, huge graphs processing
  - Derivatives for graph querying: *Maurizio Nole and Carlo Sartiani.* Regular path queries on massive graphs. 2016

Context-free language constraints

- Graph databases and semantic networks (Context-Free Path Querying, CFPQ)
  - *Sevon P., Eronen L.* "Subgraph queries by context-free grammars." 2008
  - *Hellings J.* "Conjunctive context-free path queries." 2014
  - *Zhang X. et al.* "Context-free path queries on RDF graphs." 2016
- Static code analysis (Language Reachability Framework)
  - *Thomas Reps et al.* "Precise interprocedural dataflow analysis via graph reachability." 1995
  - *Qirun Zhang et al.* "Efficient subcubic alias analysis for C." 2014
  - *Dacong Yan et al.* "Demand-driven context-sensitive alias analysis for Java." 2011
  - *Jakob Rehof and Manuel Fahndrich.* "Type-base flow analysis: from polymorphic subtyping to CFL-reachability." 2001

Open Problems

- Development of efficient algorithms
- Effective utilization of GPGPU and parallel programming
- Lifting up the limitations on the input graph and the query language

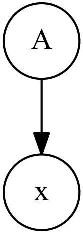
The algorithm

Algorithm 1 Context-free recognizer for graphs

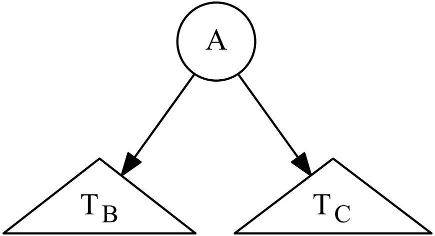
function CONTEXTFREEPATHQUERYING(D, G)
 $n \leftarrow$ the number of nodes in D
 $E \leftarrow$ the directed edge-relation from D
 $P \leftarrow$ the set of production rules in G
 $T \leftarrow$ the matrix $n \times n$ in which each element is $\varnothing$
 for all $(i, x, j) \in E$ **do**   ▷ Matrix initialization
  $T_{i,j} \leftarrow T_{i,j} \cup \{A \mid (A \rightarrow x) \in P\}$
 while matrix T is changing **do**
  $T \leftarrow T \cup (T \times T)$ ▷ Transitive closure T^{cf} calculation
 return T

Derivation step

$A \to x$



$A \to BC$



Matrix multiplication

- Subset multiplication, $N_1, N_2 \subseteq N$
  - $N_1 \cdot N_2 = \{A \mid \exists B \in N_1, \exists C \in N_2 \text{ such that } (A \to BC) \in P\}$
- Subset addition: set-theoretic union.
- Matrix multiplication
  - Matrix of size $|V| \times |V|$
  - Subsets of N are elements
  - $c_{i,j} = \bigcup_{k=1}^{n} a_{i,k} \cdot b_{k,j}$

Transitive closure

- $a^{cf} = a^{(1)} \cup a^{(2)} \cup \cdots$
- $a^{(1)} = a$
- $a^{(i)} = a^{(i-1)} \cup (a^{(i-1)} \times a^{(i-1)}),\ i \geq 2$

Algorithm Correctness

Theorem

Let $D = (V, E)$ be a graph and let $G = (N, \Sigma, P)$ be a grammar.
Then for any i, j and for any non-terminal $A \in N$, $A \in a^{cf}_{i,j}$ iff $(i, j) \in R_A$.

Theorem

Let $D = (V, E)$ be a graph and let $G = (N, \Sigma, P)$ be a grammar.
The Algorithm 1 terminates in a finite number of steps.

Algorithm Complexity

Theorem

Let $D = (V, E)$ be a graph and let $G = (N, \Sigma, P)$ be a grammar. The Algorithm 1 calculates the transitive closure T^{cf} in $O(|V|^2|N|^3(BMM(|V|) + BMU(|V|)))$.

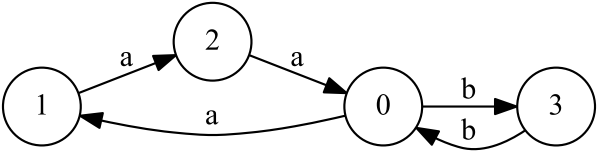
- $BMM(n)$ — number of elementary operations executed by the algorithm of multiplying two $n \times n$ Boolean matrices.
- $BMU(n)$ — number of elementary operations, executed by the matrix union operation of two $n \times n$ Boolean matrices

Algorithm Complexity: the Worst Case

Input graph: two cycles connected via a shared node

- first cycle has $2^k + 1$ edges labeled a
- second cycle has 2^k edges labeled b

$$\begin{array}{lcl} S & \rightarrow & a\ S\ b \\ & | & a\ b \end{array}$$



Evaluation

- dGPU (dense GPU): row-major matrix representation and a GPU for matrix operation calculation.
- sCPU (sparse CPU): CSR format for sparse matrix representation and a CPU for matrix operation calculation.
- sGPU (sparse GPU): CSR format for sparse matrix representation and a GPU for matrix operation calculation.

Evaluation: Same Generation Queries

Query 1 retrieves the concepts on the same layer

$$
\begin{array}{rcl}
S & \rightarrow & \mathit{subClassOf}^{-1}\ S\ \mathit{subClassOf} \\
 & \mid & \mathit{type}^{-1}\ S\ \mathit{type} \\
 & \mid & \mathit{subClassOf}^{-1}\ \mathit{subClassOf} \\
 & \mid & \mathit{type}^{-1}\ \mathit{type}
\end{array}
$$

Query 2 retrieves concepts on the adjacent layers

$$
\begin{array}{rcl}
S & \rightarrow & B\ \mathit{subClassOf} \\
S & \mid & \mathit{subClassOf} \\
B & \rightarrow & \mathit{subClassOf}^{-1}\ B\ \mathit{subClassOf} \\
B & \mid & \mathit{subClassOf}^{-1}\ \mathit{subClassOf}
\end{array}
$$

Evaluation: Query 1

Ontology	V	E	#results	GLL(ms)	dGPU
skos	144	323	810	10	56
generations	129	351	2164	19	62
travel	131	397	2499	24	69
univ-bench	179	413	2540	25	81
atom-primitive	291	685	15454	255	19
biomedical-measure-primitive	341	711	15156	261	26
foaf	256	815	4118	39	15
people-pets	337	834	9472	89	39
funding	778	1480	17634	212	141
wine	733	2450	66572	819	204
pizza	671	2604	56195	697	110
g_1	6224	11840	141072	1926	—
g_2	5864	19600	532576	6246	—
g_3	5368	20832	449560	7014	—

Evaluation: Query 2

Ontology	V	E	#results	GLL(ms)	dGPU
skos	144	323	1	1	10
generations	129	351	0	1	9
travel	131	397	63	1	31
univ-bench	179	413	81	11	55
atom-primitive	291	685	122	66	36
biomedical-measure-primitive	341	711	2871	45	27
foaf	256	815	10	2	53
people-pets	337	834	37	3	14
funding	778	1480	1158	23	124
wine	733	2450	133	8	72
pizza	671	2604	1262	29	94
g_1	6224	11840	9264	167	–
g_2	5864	19600	1064	46	–
g_3	5368	20832	10096	393	–

Pros/Cons

Something about how this algorithm is matrix operations independent and great and stuff

Contact Information

- Semyon Grigorev: s.v.grigoriev@spbu.ru
- Rustam Azimov: st013567@student.spbu.ru

- YaccConstructor: https://github.com/YaccConstructor