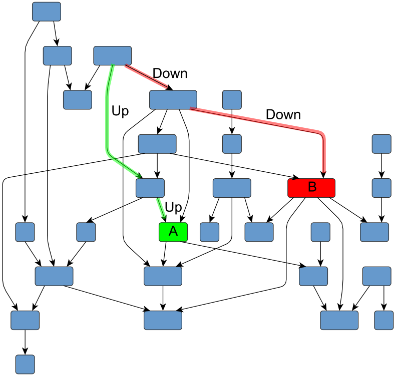
F# OpenCL Type Provider

Kirill Smirenko, **Semyon Grigorev**

JetBrains Research, Programming Languages and Tools Lab
Saint Petersburg University

September 27, 2018

GPGPU



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

Problem: GPGPU in applicatiions

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

Existing tools

- 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 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

    $T_{i,j} \leftarrow T_{i,j} \cup \{A \mid (A \rightarrow x) \in P\}$

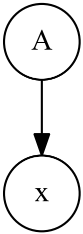
  while matrix T is changing **do**

    $T \leftarrow T \cup (T \times T)$ ▷ Transitive closure T^{cf} calculation

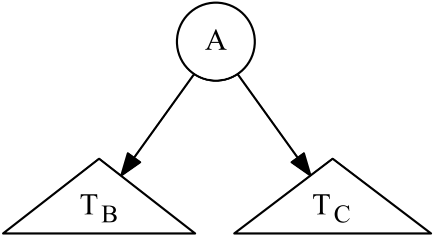
  return T

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

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

The algorithm

Algorithm Context-free recognizer for graphs

```
function CONTEXTFREEPATHQUERYING(D, G)
    n ← the number of nodes in D
    E ← the directed edge-relation from D
    P ← the set of production rules in G
    T ← the matrix n × n in which each element is ∅
    for all (i, x, j) ∈ E do                          ▷ Matrix initialization
        T_{i,j} ← T_{i,j} ∪ {A | (A → x) ∈ P}
    while matrix T is changing do
        T ← T ∪ (T × T)                  ▷ Transitive closure T^{cf} calculation
    return T
```

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 terminates in a finite number of steps.

Architecture

Limitations

- !!!
- !!!
- !!!
- !!!

Examples

future work

- !!!
- !!!
- !!!
- !!!

Summary

- Algorithm for context-free path querying
- Works on any input graph
- Supports any context-free constraints
- Is independent of matrix representation
- Can utilize GPGPU easily and efficiently

Contact Information

- Semyon Grigorev: s.v.grigoriev@spbu.ru
- Kirill Smirenko: k.smirenko@gmail.com

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