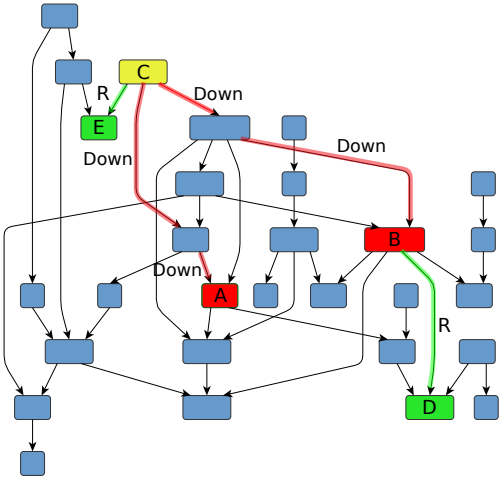
GLL-based Context-Free Path Querying for Neo4j

Vadim Abzalov, Vlada Pogozhelskaya, Vladimir Kutuev,
Olga Bachishche, **Semyon Grigorev**

Saint Petersburg State University

October 31, 2025

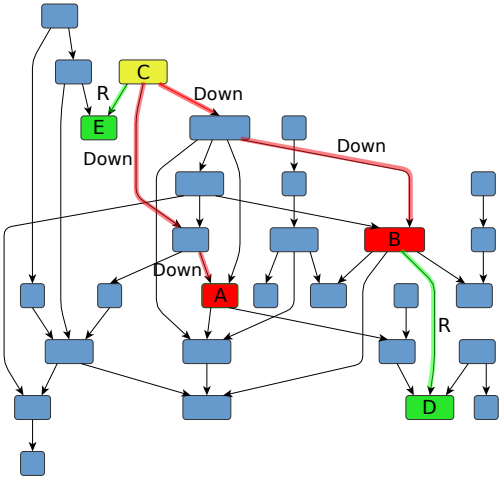
Formal Language Constrained Path Querying



Navigation through an edge-labeled graph

- **Path** specifies a **word** formed by the labels of the edges
- **Paths constraint** is a **language**: the word specified by the path should be in the given language
- The expressiveness of constraints is related to **formal languages classes**

Regular Path Queries (RPQ)

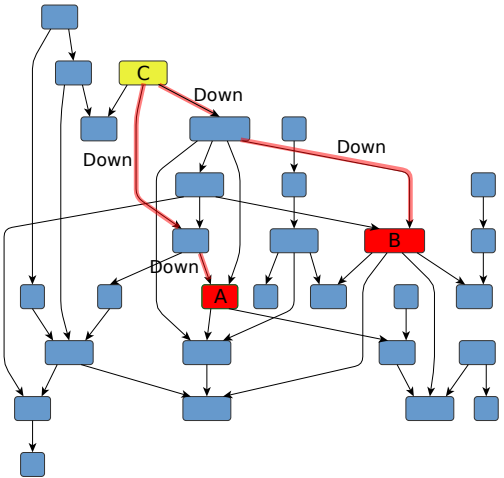


Regular languages as constraints

- Which nodes are reachable from **C** by arbitrary number of **R** and **Down** edges?
- Regular language $\mathcal{L} = (R \mid Down)^*$

- Part of GQL and SQL/PGQ (ISO/IEC 9075-16:2023)

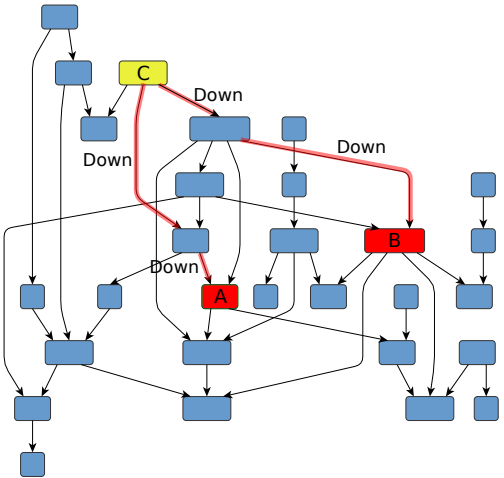
Context-Free Path Queries (CFPQ)



Context-free languages as constraints

- Are nodes A and B on the same level of hierarchy?
- Is there a path of form $\overline{\textbf{Down}}^n\,\textbf{Down}^n$ between A and B?
- Context-free grammar:
  $SameLvl \to \overline{Down}\ SameLvl\ Down \mid \varepsilon$

Context-Free Path Queries (CFPQ)



Context-free languages as constraints

- Are nodes A and B on the same level of hierarchy?
- Is there a path of form $\overline{\textbf{Down}}^n\,\textbf{Down}^n$ between A and B?
- Context-free grammar:
  $SameLvl \to \overline{Down}\ SameLvl\ Down \mid \varepsilon$

Applications

- Static code analysis [T. Reps, et al, 1995]
- Graph segmentation [H. Miao, et al, 2019]
- Bio data analysis [P. Sevon, et al, 2008]
- ...

Problem Statement

- J. Kuijpers, et al[1]: existing algorithms are too slow to be used in practical applications (in the context of Neo4j)
- Reachability in the focus
  - Paths needed in some applications
  - Not for all pairs, but for specified start vertices

? How to create faster multiple source context-free all paths querying algorithm?

[1] Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019. An Experimental Study of Context-Free Path Query Evaluation Methods.

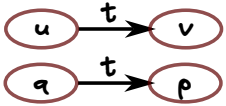
Proposed Solution

- **Generalized LL (GLL)**[2] as a base
  - Arbitrary grammars (including left-recursive and ambiguous) without transformations
  - **Shared Packed Parse Forest (SPPF)** is a native representation of all paths
  - Directed — native support of source vertices
- **Recursive State Machine (RSM)** to represent constraints
  - Instead of grammar in (E)BNF

[2] A. Afroozeh, Anastasia Izmaylova. Faster, Practical GLL Parsing. 2015

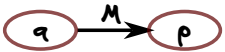
Generalized LL for CFPQ: The Idea

Current descriptor: (u, q, g)



Just read the terminal

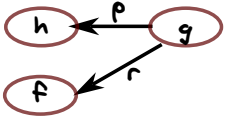
New descriptor: (v, p, g)
// For all terminal edges



Call: start handling of M on position u

New descriptor: (u, r, h)
// r: start state for M
// For all Nonterminal edges

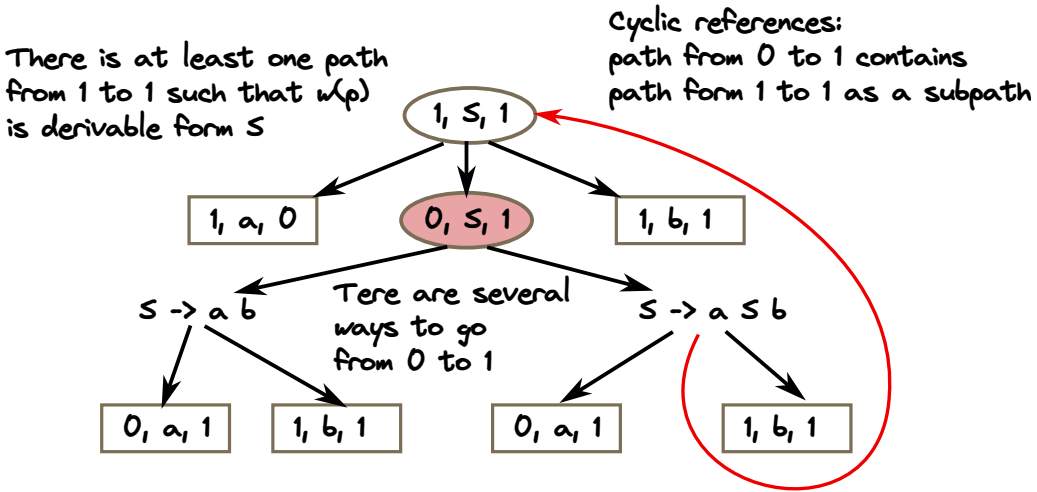
q is a final state



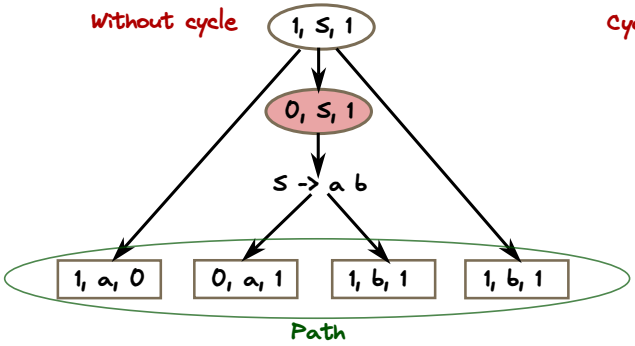
Pop is not destructive.
Just move pointer
alongside outgoing edges

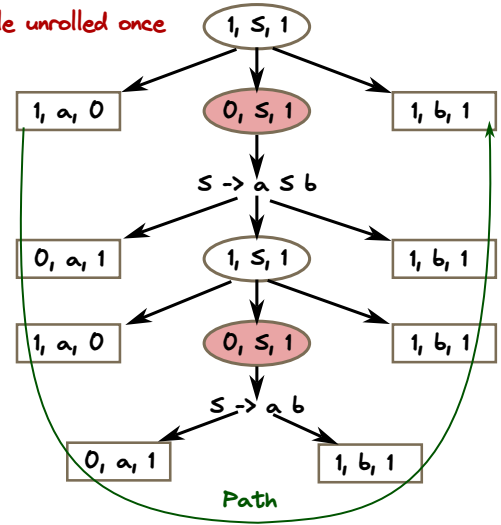
New descriptors: (u, p, h)
(u, r, f)
// For each outgoing edge

SPPF is a Representation of All Paths of Interest



Trees And Paths





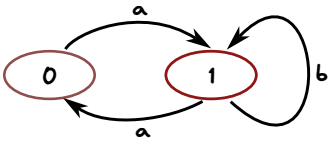
Trees extracted from SPPF for the following paths:

$1 \xrightarrow{a} 0 \xrightarrow{a} 1 \xrightarrow{b} 1 \xrightarrow{b} 1$

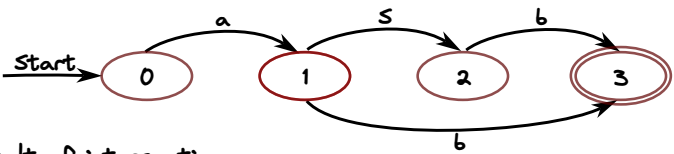
$1 \xrightarrow{a} 0 \xrightarrow{a} 1 \xrightarrow{a} 0 \xrightarrow{a} 1 \xrightarrow{b} 1 \xrightarrow{b} 1 \xrightarrow{b} 1 \xrightarrow{b} 1$

Context-Free Languages Are Closed Under Intersection With Regular Ones

Input graph: regular language

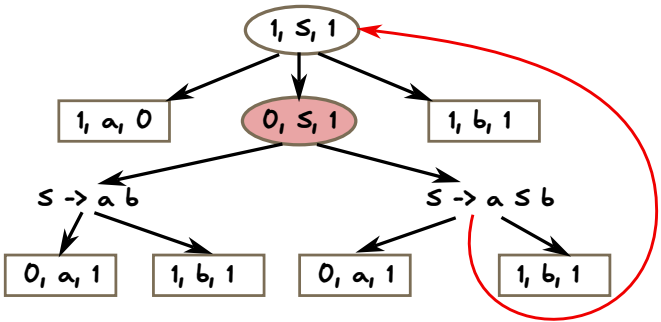


Query: context-free language



Result of intersection

SPPF



Respective context-free grammar

(1, S, 1) -> (1, a, 0) (0, S, 1) (1, b, 1)
(0, S, 1) -> (0, a, 1) (1, b, 1)
(0, S, 1) -> (0, a, 1) (1, S, 1) (1, b, 1)

Nonterminals

Terminals (edges)

Implementation Details

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

Evaluation Setup

- Ubuntu 18.04, Intel Core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM
- Graphs are stored in RedisGraph augmented with our extensions
- Queries are generated with template for the given size of the start set
- The union of all start sets is denoted V

Evaluation Setup

- Ubuntu 18.04, Intel Core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM
- Graphs are stored in RedisGraph augmented with our extensions
- Queries are generated with template for the given size of the start set
- The union of all start sets is denoted V

Graph	#V	#E	Q
core	1323	4342	g_1
pathways	6238	18 598	g_1
gohierarchy	45 007	980 218	g_1
enzyme	48 815	109 695	g_1
eclass_514en	239 111	523 727	g_1
geospecies	450 609	2 311 461	*geo*
go	272 770	534 311	g_1

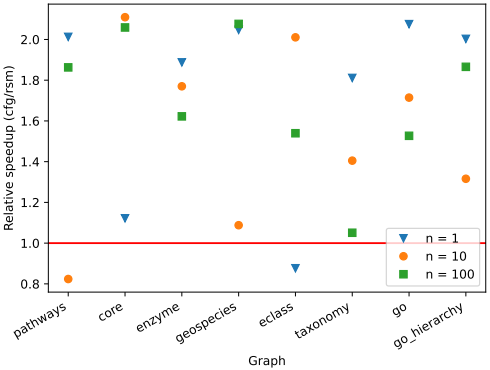
Evaluation Setup

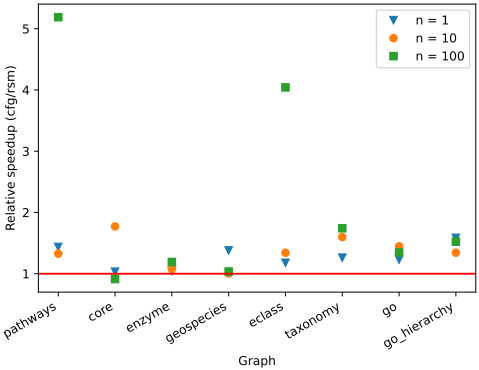
- Ubuntu 18.04, Intel Core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM
- Graphs are stored in RedisGraph augmented with our extensions
- Queries are generated with template for the given size of the start set
- The union of all start sets is denoted V

Graph	#V	#E	Q
core	1323	4342	g_1
pathways	6238	18 598	g_1
gohierarchy	45 007	980 218	g_1
enzyme	48 815	109 695	g_1
eclass_514en	239 111	523 727	g_1
geospecies	450 609	2 311 461	*geo*
go	272 770	534 311	g_1

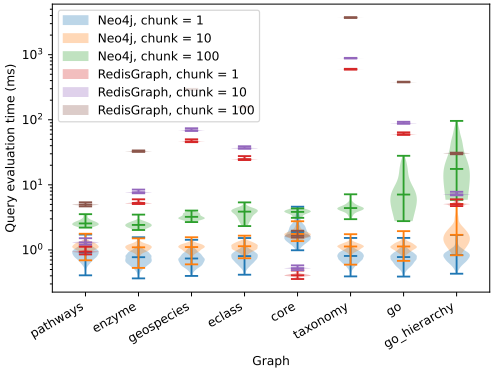
```
PATH PATTERN S =
  ()-/ [<:SubClassOf [~S | ()] :SubClassOf] | [<:Type [~S | ()] :Type] /->()
MATCH (src)-/ ~S /->()
WHERE {id_from} <= src.id and src.id <= {id_to}
RETURN count(*)
```

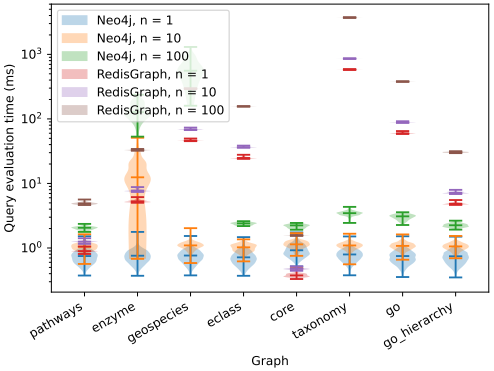
Multiple sources CFPQ reachability speedup (RSM over CFG) on RDF graphs



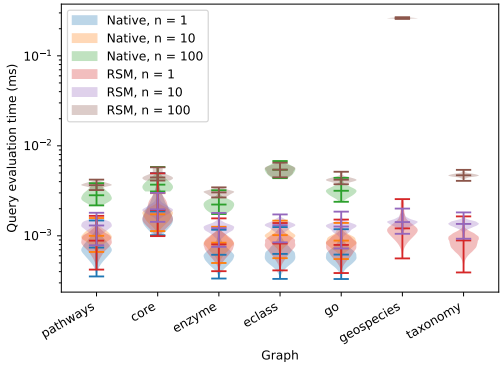


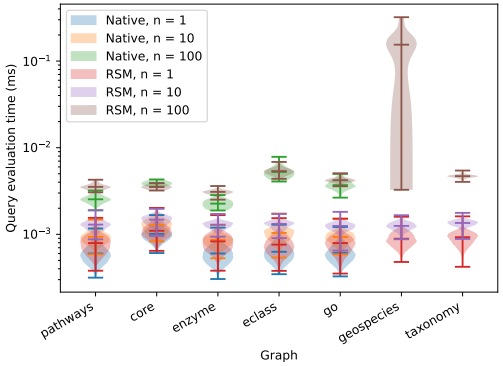
Multiple sources CFPQ reachability results for queries related to RDF analysis





Multiple source RPQ reachability results for queries related to RDF analysis and respective query (native solution failed with OOM on last two graphs)





Conclusion

- Full-stack support for CFPQ in real-world applications which use RedisGraph database with Cypher query language
  - No more context-free grammars
  - No more custom graph formats and storages
- Reasonable performance of context-free path queries
  - Multiple-source scenario
  - Space-time ratio can be tuned
- Context-free path queries can be used in applications with well-established tools

Future Research

- Mechanization of Cypher semantics in Coq
  - Semantics which includes path patterns
  - Goal: prove correctness of translation to linear algebra

Future Research

- Mechanization of Cypher semantics in Coq
  - Semantics which includes path patterns
  - Goal: prove correctness of translation to linear algebra
- Integration of tensor-based CFPQ algorithm[3] to RedisGraph
  - The algorithm constructs paths, not only reachability facts
  - The algorithm should be modified to get multiple-source version

[3] Egor Orachev, Ilya Epelbaum, R. Azimov and S. Grigorev. 2020. Context-Free Path Querying by Kronecker Product.

Future Research

- Mechanization of Cypher semantics in Coq
  - Semantics which includes path patterns
  - Goal: prove correctness of translation to linear algebra
- Integration of tensor-based CFPQ algorithm[3] to RedisGraph
  - The algorithm constructs paths, not only reachability facts
  - The algorithm should be modified to get multiple-source version
- Detailed evaluation
  - Include more graphs and queries, including RPQs
  - Evaluate the scalability of the solution
  - Compare with other graph query engines

[3] Egor Orachev, Ilya Epelbaum, R. Azimov and S. Grigorev. 2020. Context-Free Path Querying by Kronecker Product.

Contact Information

- Try it out (Docker image with extended RedisGraph):
  `https://hub.docker.com/r/simpletondl/redisgraph`
- RedisGraph extended with CFPQ:
  `https://github.com/YaccConstructor/RedisGraph`
- Cypher parser extended with path patterns:
  `https://github.com/YaccConstructor/libcypher-parser`

Contact Information

- Try it out (Docker image with extended RedisGraph):
  `https://hub.docker.com/r/simpletondl/redisgraph`
- RedisGraph extended with CFPQ:
  `https://github.com/YaccConstructor/RedisGraph`
- Cypher parser extended with path patterns:
  `https://github.com/YaccConstructor/libcypher-parser`

- Semyon Grigorev: s.v.grigoriev@spbu.ru
- Arseniy Terekhov: simpletondl@yandex.ru
- Vlada Pogozhelskaya: pogozhelskaya@gmail.com
- Vadim Abzalov: vadim.i.abzalov@gmail.com
- Timur Zinnatulin: teemychteemych@gmail.com

Contact Information

- Try it out (Docker image with extended RedisGraph):
  `https://hub.docker.com/r/simpletondl/redisgraph`
- RedisGraph extended with CFPQ:
  `https://github.com/YaccConstructor/RedisGraph`
- Cypher parser extended with path patterns:
  `https://github.com/YaccConstructor/libcypher-parser`

Thanks!

- Semyon Grigorev: s.v.grigoriev@spbu.ru
- Arseniy Terekhov: simpletondl@yandex.ru
- Vlada Pogozhelskaya: pogozhelskaya@gmail.com
- Vadim Abzalov: vadim.i.abzalov@gmail.com
- Timur Zinnatulin: teemychteemych@gmail.com