

Context-Free Path Querying: Obstacles on the Way to Adoption

Semyon Grigorev

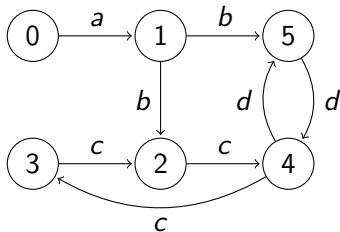
JetBrains Research, Programming Languages and Tools Lab
St. Petersburg State University

https://research.jetbrains.org/groups/plt_lab/

16.07.2021

Formal Language Constrained Path Querying

Navigational queries in edge-labelled graph

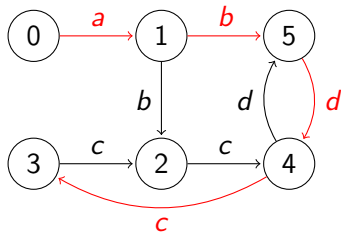


- Path to find:

$$0 \xrightarrow{a} v_0 \xrightarrow{b} v_1 \underbrace{\xrightarrow{d} v_2 \xrightarrow{c} v_3 \dots v_k}_{c \text{ or } d \text{ in arbitrary order}} \xrightarrow{c} v$$

Formal Language Constrained Path Querying

Navigational queries in edge-labelled graph

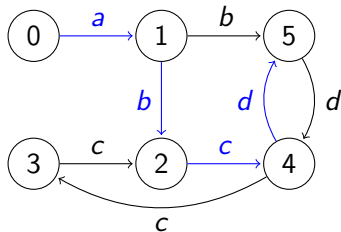


- Path to find:

$$0 \xrightarrow{a} v_0 \xrightarrow{b} v_1 \underbrace{\xrightarrow{d} v_2 \xrightarrow{c} v_3 \dots v_k}_{c \text{ or } d \text{ in arbitrary order}} \xrightarrow{c} v$$

Formal Language Constrained Path Querying

Navigational queries in edge-labelled graph

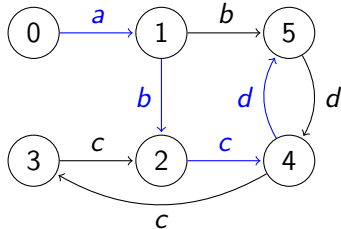


- Path to find:

$$0 \xrightarrow{a} v_0 \xrightarrow{b} v_1 \underbrace{\xrightarrow{d} v_2 \xrightarrow{c} v_3 \dots v_k}_{c \text{ or } d \text{ in arbitrary order}} \xrightarrow{c} v$$

Formal Language Constrained Path Querying

Navigational queries in edge-labelled graph



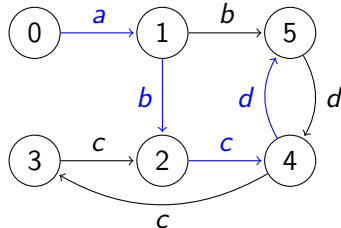
- $w(v_0 \xrightarrow{l_0} v_1 \xrightarrow{l_1} \dots \xrightarrow{l_{k-1}} v_k) = l_0 l_1 \dots l_{k-1}$
- $Q = \{(v_i, v_j) \mid \exists \pi = v_i \rightarrow \dots \rightarrow v_j; w(\pi) \in \mathcal{L}\}$,
where \mathcal{L} — formal language
 - ✓ Regular, RPQ $(ab(c \mid d)^*)$
 - ⚙ **Context-Free**, CFPQ $(a^n b^n)$
 - ⌚ Multiple Context-Free $(a^n c^m b^n d^m)$

- Path to find:

$$0 \xrightarrow{a} v_0 \xrightarrow{b} v_1 \underbrace{\xrightarrow{d} v_2 \xrightarrow{c} v_3 \dots v_k}_{c \text{ or } d \text{ in arbitrary order}} \xrightarrow{c} v$$

Formal Language Constrained Path Querying

Navigational queries in edge-labelled graph



- Path to find:

$$0 \xrightarrow{a} v_0 \xrightarrow{b} v_1 \underbrace{\xrightarrow{d} v_2 \xrightarrow{c} v_3 \dots v_k}_{c \text{ or } d \text{ in arbitrary order}} \xrightarrow{c} v$$

- $w(v_0 \xrightarrow{l_0} v_1 \xrightarrow{l_1} \dots \xrightarrow{l_{k-1}} v_k) = l_0 l_1 \dots l_{k-1}$
- $Q = \{(v_i, v_j) \mid \exists \pi = v_i \rightarrow \dots \rightarrow v_j; w(\pi) \in \mathcal{L}\}$, where \mathcal{L} — formal language
 - ✓ Regular, RPQ $(ab(c \mid d)^*)$
 - ⚙ **Context-Free**, CFPQ $(a^n b^n)$
 - ⌚ Multiple Context-Free $(a^n c^m b^n d^m)$

Variations:

- All-pairs
- Multiple source
- Reachability
- All paths
- ...

Applications of Context-Free Path Querying

Hierarchy analysis: variations of the *same-generation queries* is the essence of CFPQ

¹Thomas Reps. 1997. “Program Analysis via Graph Reachability”.

²Mihalis Yannakakis. 1990. “Graph-theoretic Methods in Database Theory”.

Applications of Context-Free Path Querying

Hierarchy analysis: variations of the *same-generation queries* is the essence of CFPQ

Graph structured data analysis

- Introduced by M. Yannakakis in 1990¹
- Biological data analysis
- Data provenance analysis
- ...

¹Thomas Reps. 1997. "Program Analysis via Graph Reachability".

²Mihalis Yannakakis. 1990. "Graph-theoretic Methods in Database Theory".

Applications of Context-Free Path Querying

Hierarchy analysis: variations of the *same-generation queries* is the essence of CFPQ

Graph structured data analysis

- Introduced by M. Yannakakis in 1990¹
- Biological data analysis
- Data provenance analysis
- ...

Static code analysis

- Introduced by T. Reps in 1997²
- Interprocedural points-to analysis
- Interprocedural alias analysis
- Type inference related tasks
- ...

¹Thomas Reps. 1997. "Program Analysis via Graph Reachability".

²Mihalis Yannakakis. 1990. "Graph-theoretic Methods in Database Theory".

Applications of Context-Free Path Querying

Hierarchy analysis: variations of the *same-generation queries* is the essence of CFPQ

Graph structured data analysis

- Introduced by M. Yannakakis in 1990¹
- Biological data analysis
- Data provenance analysis
- ...

Static code analysis

- Introduced by T. Reps in 1997²
- Interprocedural points-to analysis
- Interprocedural alias analysis
- Type inference related tasks
- ...

Graph databases

¹Thomas Reps. 1997. "Program Analysis via Graph Reachability".

²Mihalis Yannakakis. 1990. "Graph-theoretic Methods in Database Theory".

There is No Unified Infrastructure to Compare CFPQ Solutions

- ? Which algorithm is better for the specific problem?
- ? How to assess if a newly developed algorithm is better than the existing ones?
- ? ...

³Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019

There is No Unified Infrastructure to Compare CFPQ Solutions

- ? Which algorithm is better for the specific problem?
- ? How to assess if a newly developed algorithm is better than the existing ones?
- ? ...
- 💡 **Unified benchmarks:** dataset, workflow, ...

³Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019

There is No Unified Infrastructure to Compare CFPQ Solutions

- ? Which algorithm is better for the specific problem?
- ? How to assess if a newly developed algorithm is better than the existing ones?
- ? ...
- 💡 **Unified benchmarks:** dataset, workflow, ...
- ! The only attempt to compare different algorithms: “An Experimental Study of Context-Free Path Query Evaluation Methods”³

³Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019

There is No Unified Infrastructure to Compare CFPQ Solutions

- ? Which algorithm is better for the specific problem?
- ? How to assess if a newly developed algorithm is better than the existing ones?
- ? ...
- 💡 **Unified benchmarks:** dataset, workflow, ...
- ! The only attempt to compare different algorithms: “An Experimental Study of Context-Free Path Query Evaluation Methods”³
 - 😞 “We conclude that state of the art solutions are not able to cope with large graphs as found in practice.”

³Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019

There is No Unified Infrastructure to Compare CFPQ Solutions

- ? Which algorithm is better for the specific problem?
- ? How to assess if a newly developed algorithm is better than the existing ones?
- ? ...
- 💡 **Unified benchmarks:** dataset, workflow, ...
- ! The only attempt to compare different algorithms: “An Experimental Study of Context-Free Path Query Evaluation Methods”³
 - 😞 “We conclude that state of the art solutions are not able to cope with large graphs as found in practice.”

Difficulties

- Data is spread over projects and papers in different communities
- There is a huge number of different subclasses of the problem
 - ▶ all-pairs, single source, multiple source, ...
 - ▶ reachability, single path, all path, ...

³Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019

There is No Support of CFPQ in Real-World Graph Databases

? Which database or graph analysis system should you choose?

☹️ H. Miao and A. Deshpande: “Though the problem has been first studied in our community [40], there is little follow up and support in the context of modern graph databases ...”⁴

Difficulties

? How to choose an appropriate algorithm for query engine?

⚙️ Benchmarks for querying algorithms

? How to express context-free constraints in graph query language?

? Syntactic features to express context-free language constraints

? Semantics of query language

⁴H. Miao and A. Deshpande, “Understanding Data Science Lifecycle Provenance via Graph Segmentation and Summarization”, 2019

There is No Support of CFPQ in Real-World Graph Databases

? Which database or graph analysis system should you choose?

🙄 H. Miao and A. Deshpande: “Though the problem has been first studied in our community [40], there is little follow up and support in the context of modern graph databases ...”⁴

Difficulties

? How to choose an appropriate algorithm for query engine?

⚙️ Benchmarks for querying algorithms

? How to express context-free constraints in graph query language?

? Syntactic features to express context-free language constraints

? Semantics of query language

⚙️ GQL standard

⁴H. Miao and A. Deshpande, “Understanding Data Science Lifecycle Provenance via Graph Segmentation and Summarization”, 2019

Our Results

- ✓ Collection of linear algebra based algorithms for CFPQ
 - ▶ SuiteSparse is utilized for sparse linear algebra subroutines
 - ▶ Published: https://github.com/JetBrains-Research/CFPQ_PyAlgo

⁵Arseniy Terekhov et al. 2019. "Multiple-Source Context-Free Path Querying in Terms of Linear Algebra".

Our Results

- ✓ Collection of linear algebra based algorithms for CFPQ
 - ▶ SuiteSparse is utilized for sparse linear algebra subroutines
 - ▶ Published: https://github.com/JetBrains-Research/CFPQ_PyAlgo
- ✓ Full-stack support of CFPQ⁵
 - ▶ On top of RedisGraph: query engine is extended with CFPQ algorithm
 - ▶ openCypher is extended to support CFPQ

⁵Arseniy Terekhov et al. 2019. "Multiple-Source Context-Free Path Querying in Terms of Linear Algebra".

Our Results

- ✓ Collection of linear algebra based algorithms for CFPQ
 - ▶ SuiteSparse is utilized for sparse linear algebra subroutines
 - ▶ Published: https://github.com/JetBrains-Research/CFPQ_PyAlgo
- ✓ Full-stack support of CFPQ⁵
 - ▶ On top of RedisGraph: query engine is extended with CFPQ algorithm
 - ▶ openCypher is extended to support CFPQ
- ⚙ Dataset for CFPQ benchmarking: early stages
 - ▶ Synthetic graphs
 - ★ Theoretical worst case
 - ★ Complicated cases
 - ▶ Real-world graphs
 - ★ Static code analysis
 - ★ Biological data analysis
 - ▶ Published: https://github.com/JetBrains-Research/CFPQ_Data

⁵Arseniy Terekhov et al. 2019. "Multiple-Source Context-Free Path Querying in Terms of Linear Algebra".

Our Results Evaluation

- All-pairs reachability queries
 - *geospecies*, *taxonomy* — biological data
 - *crypto*, *drivers*, *fs* — points-to analysis
 - Time in seconds
- GPU: Geforce GTX 1070, 1.5GHz, 8Gb RAM, 1920 CUDA cores
 - CPU: Intel core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM

Graph	#V	#E	Neo4j ⁶	RedisGraph ⁷	Lin.al. CPU ⁸	Lin.al. GPU ⁹
geospecies	450 609	2 311 461	6 953.9	80.1	7.1	0.8
taxonomy	5 728 398	14 922 125	n.a.	⚙️	1.1	0.7
crypto	3 464 970	5 976 774	n.a.	⚙️	84.8	28.1
drivers	4 273 803	7 415 538	n.a.	⚙️	269.9	62.5
fs	4 177 416	7 218 746	n.a.	⚙️	165.1	47.7

⁶Jochem Kuijpers et al. 2019. “An Experimental Study of Context-Free Path Query Evaluation Methods”.





⁷Arseniy Terekhov et al. 2019. “Multiple-Source Context-Free Path Querying in Terms of Linear Algebra”.

⁸Standalone linear algebra based algorithm on CPU, using SuiteSparse.

⁹Standalone linear algebra based algorithm on GPU, using spbla.

Our Results Evaluation

- All-pairs reachability queries
 - *geospecies*, *taxonomy* — biological data
 - *crypto*, *drivers*, *fs* — points-to analysis
 - Time in seconds
- GPU: Geforce GTX 1070, 1.5GHz, 8Gb RAM, 1920 CUDA cores
 - CPU: Intel core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM

Graph	#V	#E	Neo4j ⁶	RedisGraph ⁷	Lin.al. CPU ⁸	Lin.al. GPU ⁹
geospecies	450 609	2 311 461	6 953.9	80.1	7.1	0.8
taxonomy	5 728 398	14 922 125	n.a.		1.1	0.7
crypto	3 464 970	5 976 774	n.a.		84.8	28.1
drivers	4 273 803	7 415 538	n.a.		269.9	62.5
fs	4 177 416	7 218 746	n.a.		165.1	47.7

⁶Jochem Kuijpers et al. 2019. “An Experimental Study of Context-Free Path Query Evaluation Methods”.





⁷Arseniy Terekhov et al. 2019. “Multiple-Source Context-Free Path Querying in Terms of Linear Algebra”.

⁸Standalone linear algebra based algorithm on CPU, using SuiteSparse.

⁹Standalone linear algebra based algorithm on GPU, using spbla.

Our Results Evaluation

- All-pairs reachability queries
 - *geospecies*, *taxonomy* — biological data
 - *crypto*, *drivers*, *fs* — points-to analysis
 - Time in seconds
- GPU: Geforce GTX 1070, 1.5GHz, 8Gb RAM, 1920 CUDA cores
 - CPU: Intel core i7-6700 CPU, 3.4GHz, DDR4 64Gb RAM

Graph	#V	#E	Neo4j ⁶	RedisGraph ⁷	Lin.al. CPU ⁸	Lin.al. GPU ⁹
geospecies	450 609	2 311 461	6 953.9	80.1	7.1	0.8
taxonomy	5 728 398	14 922 125	n.a.		1.1	0.7
crypto	3 464 970	5 976 774	n.a.		84.8	28.1
drivers	4 273 803	7 415 538	n.a.		269.9	62.5
fs	4 177 416	7 218 746	n.a.		165.1	47.7

⁶Jochem Kuijpers et al. 2019. “An Experimental Study of Context-Free Path Query Evaluation Methods”.

⁷Arseniy Terekhov et al. 2019. “Multiple-Source Context-Free Path Querying in Terms of Linear Algebra”.

⁸Standalone linear algebra based algorithm on CPU, using SuiteSparse.

⁹Standalone linear algebra based algorithm on GPU, using spbla.

- ⚙️ Benchmarking of linear algebra based algorithms
 - ▶ Comparison of different algorithms for different query semantics
 - ▶ Investigation of scalability on multicore machines
 - ▶ Estimation of performance on GPGPU
- ⚙️ Developing and evaluating GLL-based CFPQ algorithm for Neo4j
 - ▶ Multiple-source
 - ▶ All paths and reachability-only

Benchmarking of linear algebra based algorithms

- ▶ Comparison of different algorithms for different query semantics
- ▶ Investigation of scalability on multicore machines
- ▶ Estimation of performance on GPGPU

Developing and evaluating GLL-based CFPQ algorithm for Neo4j

- ▶ Multiple-source
- ▶ All paths and reachability-only

Describing semantics of (subset of) openCypher in terms of linear algebra (in Coq)

Utilizing multiple context-free languages as path constraints

What Should We Do?

- ▶ **A** Publish unified benchmarks for formal language constrained path querying algorithms
 - ▶ Graphs: synthetic and real-world
 - ▶ Queries: templates and real-world queries
 - ▶ Tasks: all-pairs, single source, reachability, ...

What Should We Do?

- A** Publish unified benchmarks for formal language constrained path querying algorithms
 - ▶ Graphs: synthetic and real-world
 - ▶ Queries: templates and real-world queries
 - ▶ Tasks: all-pairs, single source, reachability, ...
- A** Organize language constrained path querying competition
 - ▶ Part of existing competition for graph processing systems
 - ▶ Involve static analysis community

What Should We Do?

- A** Publish unified benchmarks for formal language constrained path querying algorithms
 - ▶ Graphs: synthetic and real-world
 - ▶ Queries: templates and real-world queries
 - ▶ Tasks: all-pairs, single source, reachability, ...
- A** Organize language constrained path querying competition
 - ▶ Part of existing competition for graph processing systems
 - ▶ Involve static analysis community
- ?** Provide graph database support
 - ▶ Different algorithms for different systems
 - ▶ Syntax and semantics of query languages