Context-Free Path Querying via Matrix Equations

Yuliya Susanina
Saint Petersburg State University
JetBrains Research
St. Petersburg, Russia
jsusanina@gmail.com

ABSTRACT

Context-free path querying (CFPQ) widely used for graph-structured data analysis in such areas as bioinformatics, static code analysis, graph databases. It is crucial to develop highly efficient algorithms for CFPQ since the size of the input data is typically large. Computational mathematics may be useful because of both rich theoretical foundations and constantly improving modifications. We show how to reduce GFPQ evaluation to solving systems of matrix equations over $\mathbb{R}$ — a problem for which there exist high-performance solutions. Also, we demonstrate the applicability of our approach to real-world data analysis.

CCS CONCEPTS

• **Information systems → Query languages for non-relational engines**; • **Theory of computation → Formal languages and automata theory**; • **Mathematics of computing** → *Computations on matrices.*

KEYWORDS

context-free path querying; graph databases; context-free grammar; nonlinear matrix equations; newton method

ACM Reference Format:
Yuliya Susanina. 2020. Context-Free Path Querying via Matrix Equations. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3318464.3384400

1 INTRODUCTION

Context-free path querying (CFPQ)—a way to specify path constraints in terms of context-free grammars—is gaining popularity in many areas, e. g. bioinformatics [14], graph databases [8, 9, 17] or static code analysis [11, 18]. CFPQ is more expressive than the more commonly used regular languages constrained path query and therefore forms a promising research area. Although many different algorithms for CFPQ have been proposed [2, 9, 12, 16, 19], recent research shows that real-world data handling is still a problem [8].

Recent experiment [10] demonstrates a promising way to get high-performance CFPQ: reduce the problem to well-established problem for which high-performance solutions are available. One such reduction was proposed by Rustam Azimov [2] who showed that CFPQ can be reduced to Boolean matrix multiplication. This way we, can offload the most computationally intensive calculations to high-performance libraries for matrix processing, which utilizes GPGPUs and other modern hardware.

On the other hand, there are several results of applying linear algebra methods to logic programming [1, 13]. In [13] it is shown that the evaluation of a subset of Datalog queries can be reduced to matrix equation solving. This way one can use numerical linear algebra and computational mathematics to improve the performance of query evaluation. If we adapt this technique for CFPQ, we can employ high-performance algorithms for numerical analysis and equation solving to improve the performance of CFPQ. Approximate computational methods can also accelerate CFPQs processing. And most importantly, the popularity of artificial intelligence techniques pushed the development and improvement of many efficient libraries for numerical computing.

In this work we propose a new way to reduce CFPQ to a problem with available high-performance solutions: we show that CFPQ can be reduced to solving the systems of equations over real numbers $\mathbb{R}$. We also assess the feasibility of using both accurate and approximate methods of computational mathematics, such as a Newton method with high-performance implementations. The evaluation of our approach on a set of conventional benchmarks shows that it is comparable with the matrix-based approach and applicable for real-world data processing.

2 PRELIMINARIES

Context-free grammar (CFG) is a triple $G = (N, \Sigma, R)$, where N is a set of nonterminal symbols, Σ is a set of terminal symbols and R is a set of productions of the followings form: $A \to \alpha$, $\alpha \in (N \cup \Sigma)^*$. $\mathcal{L}(G_S)$ denotes a language specified by CFG G with respect to $S \in N$: $\mathcal{L}(G_S) = \{\omega \mid S \Rightarrow^*_G \omega\}$.

Directed graph is a triple $D = (V, E, \sigma)$, where V is a set of vertices, $\sigma \subseteq \Sigma$ is a set of labels, and a set of edges $E \subseteq V \times \sigma \times V$. Denote a path from the node m to the node n in graph D as $m\lambda n$, where λ is the unique word, obtained by concatenating the labels of the edges along this path. P is a set of all paths in D.

CFPQ problem with relational semantics (according Hellings [5]) is for a graph D and a CFG G, to find *context-free relations* $R_A \subseteq V \times V$ such that for each $A \in N$: $R_A = \{(m, n) \mid m\lambda n \in P, \lambda \in \mathcal{L}(G_A)\}$. Every relation R_A is finite, so it can be represented as a Boolean matrix: $(T_A)_{i,j} = 1 \iff (i, j) \in R_A$.

3 EQUATION-BASED APPROACH

We reduce CFPQ to equation solving similarly to the way Taisuke Sato reduces Datalog program evaluation to linear algebra [13]. But

SIGMOD'20, June 14–19, 2020, Portland, OR, USA

ACM ISBN 978-1-4503-6735-6/20/06.
https://doi.org/10.1145/3318464.3384400

unlike Sato, we also consider nonlinear cases and draw attention to the approximate computational methods.

For a given graph and grammar, each terminal and nonterminal specifies a finite relation R_A which can be represented as a Boolean matrix. After that for each production of form

$$N_i \to \beta_0^0 \dots \beta_k^0 \mid \dots \mid \beta_0^l \dots \beta_m^l, \beta_j^i \in \Sigma \cup N$$

we can create an equation

$$T_{N_i} = T_{\beta_0^0} \cdot \dots \cdot T_{\beta_k^0} + \dots + T_{\beta_0^l} \cdot \dots \cdot T_{\beta_m^l}$$

For example, a simple query for describing the $a^n b^n$ language cannot be expressed using some regular grammar, we should use context-free one: $S \to aSb \mid ab$. Boolean matrix equation for this grammar is

$$T_S = T_A T_S T_B + T_A T_B$$

We can solve this equation by naïve iterative process:

$$T_S^0 = \mathbf{0}$$
$$T_S^{k+1} = T_a T_S^k T_b + T_a T_b$$

To be able to apply advanced numerical techniques we can consider another equation over $\mathbb{R}$:

$$\mathcal{T}_S = \epsilon(T_A \mathcal{T}_S T_B + T_A T_B)$$

For the equation we can construct a matrix series $\{\mathcal{T}_S^k\}$:

$$\mathcal{T}_S^0 = \mathbf{0}$$
$$\mathcal{T}_S^{k+1} = \epsilon(T_A \mathcal{T}_S^k T_B + T_A T_B)$$

It converges to $\mathcal{T}_S^*$ when $\mathcal{T}_S^k \le \mathbf{1}$ as a monotonically increasing series of matrices with an upper bound. It can be proved that $((\mathcal{T}_S^{k+1})_{ij} > 0 \iff (T_S^{k+1})_{ij} = 1)$ and $ceil(\mathcal{T}_S^*) = T_S^*$, where $ceil(x)$ returns the least integer greater or equal to x [13].

So, for each rule of the form

$$N_i \to \beta_0^0 \dots \beta_k^0 \mid \dots \mid \beta_0^l \dots \beta_m^l, \beta_j^i \in \Sigma \cup N$$

we can generate an equation

$$\mathcal{T}_{N_i} = \epsilon_{N_i}(T_{\beta_0^0} \cdot \dots \cdot T_{\beta_k^0} + \dots + T_{\beta_0^l} \cdot \dots \cdot T_{\beta_m^l}),$$

where ϵ_{N_i} is chosen such that $\mathcal{T}_{N_i}^k \le \mathbf{1}$ for each k. In real-world cases, we deal with the systems of matrix equations because grammars contain more than one nonterminal. The equations are either linear or nonlinear.

Linear Equations. If the input CFG is linear, each equation is of the form of generalized Sylvester equation: $\sum_{i=1}^k A_i X B_i = C$. For $k = 1, 2$ we can solve it in $O(|V|^3)$ [3]. Otherwise, for $k > 2$ it can be reduced using Kronecker product to solving a linear system $Ax = b$, where A is a matrix of size $(|V|^2 \times |V|^2)$ and time required to compute its solution is $O(|V|^6)$ or $O(|V|^{4+2\omega})$ using more efficient matrix multiplication algorithms (where $O(n^{2+\omega})$ is a complexity of the most performant $n \times n$ matrix multiplication algorithm). The use of sparse matrix representation and approximate methods can be very efficient for solving the equations of this type [4].

For the system of equations, we construct the dependency graph $D_G = (V_G, E_G)$ for nonterminals of a given grammar G and split the set of the equations into disjoint subsets accordingly to the set of strongly connected components in D_G, which can be found in $O(|V_G| + |E_G|)$ [15]. Then we solve our system in stages, for each subset.

Nonlinear Equations. For the nonlinear case we can rewrite each equation of the form $X = \Psi(X)$ to the equivalent $F(X) = X - \Psi(X) = 0$ and use Newton's method for nonlinear functions root finding:

$$X_{i+1} = X_i - (F'(X_i))^{-1} F(X_i) \iff \begin{cases} F'(X_i) H_i = -F(X_i) \\ X_{i+1} = X_i + H_i \end{cases}$$

We start iterations from X_0 as an initial guess, in our case $X_0 = \mathbf{0}$, as our solution is a matrix of small positive numbers. The convergence of this method is quadratic which means that the solution finding can be significantly faster. Even as it is necessary to solve an equation for H_i (also generalized Sylvester equation) on each iteration step, the majority of high-performance implementations do not compute the Jacobian inverse and use its approximate value [7].

The main difficulty in using Newton's method is choosing an appropriate ϵ; to ensure the least positive solution for nonlinear equations ϵ must be smaller than $\frac{1}{|V|}$.

4 EVALUATION

The equation-based approach for CFPQ was implemented and evaluated on **Query 2** from [2]. The equation constructed for this query were solved on the CPU by using two functions from Python package *scipy* [6]: *spsolve* (**sSLV**) to solve as a sparse linear system and *optimize.newton_krylov* (**dNWT**) to find a root of a function.

Table 1: Evaluation results for Query 2 (in ms)

Ontology	|V|	dNWT	sSLV	dGPU	sCPU	sGPU
bio-meas	341	284	35	276	91	24
people-pets	337	73	49	144	38	6
funding	778	502	184	1246	344	27
wine	733	791	171	722	179	6
pizza	671	334	161	943	256	23

We compare (Table 1) our solution with the first matrix-based algorithm implementations described in [2] and conclude that the equation-based approach can be applied to the real-world data and is compared to the matrix-based algorithm. Moreover, we can improve the performance by the means of parallel techniques for matrix operations.

5 CONCLUSION AND FUTURE WORK

We proposed a new approach for context-free path querying, based on solving the systems of equations over $\mathbb{R}$. The evaluation of our approach on a set of conventional benchmarks showed its applicability for the real-world data analysis.

Another direction is to determine the subclasses of (system of) polynomial equations the solution of which can be reduced to CFPQ and try to construct a bidirectional reduction between CFPQ and these subclasses, thereby finding efficient solutions for both these problems.

ACKNOWLEDGMENTS

The research was supported by the Russian Science Foundation grant 18-11-00100.

REFERENCES

[1] Yaniv Aspis. 2018. *A Linear Algebraic Approach to Logic Programming*. Ph.D. Dissertation. Imperial College London.

[2] Rustam Azimov and Semyon Grigorev. 2018. Context-free Path Querying by Matrix Multiplication. In *Proceedings of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA) (GRADES-NDA '18)*. ACM, New York, NY, USA, Article 5, 10 pages. https://doi.org/10.1145/3210259.3210264

[3] R. H. Bartels and G. W. Stewart. 1972. Solution of the Matrix Equation AX + XB = C [F4]. *Commun. ACM* 15, 9 (Sept. 1972), 820–826. https://doi.org/10.1145/361573.361582

[4] Abderrahman Bouhamidi and Khalide Jbilou. 2008. A note on the numerical approximate solutions for generalized Sylvester matrix equations with applications. *Appl. Math. Comput.* 206 (12 2008), 687–694. https://doi.org/10.1016/j.amc.2008.09.022

[5] Jelle Hellings. 2014. Conjunctive Context-Free Path Queries. OpenProceedings.org. https://doi.org/10.5441/002/ICDT.2014.15

[6] Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–2019. SciPy: Open source scientific tools for Python. http://www.scipy.org/ [Online; accessed 5.3.2019].

[7] D. A. Knoll and D. E. Keyes. 2004. Jacobian-free Newton-Krylov Methods: A Survey of Approaches and Applications. *J. Comput. Phys.* 193, 2 (Jan. 2004), 357–397. https://doi.org/10.1016/j.jcp.2003.08.010

[8] Jochem Kuijpers, George Fletcher, Nikolay Yakovets, and Tobias Lindaaker. 2019. An Experimental Study of Context-Free Path Query Evaluation Methods. In *Proceedings of the 31st International Conference on Scientific and Statistical Database Management (SSDBM '19)*. ACM, New York, NY, USA, 121–132. https://doi.org/10.1145/3335783.3335791

[9] Ciro M. Medeiros, Martin A. Musicante, and Umberto S. Costa. 2018. Efficient Evaluation of Context-free Path Queries for Graph Databases. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC '18)*. ACM, New York, NY, USA, 1230–1237. https://doi.org/10.1145/3167132.3167265

[10] Nikita Mishin, Iaroslav Sokolov, Egor Spirin, Vladimir Kutuev, Egor Nemchinov, Sergey Gorbatyuk, and Semyon Grigorev. 2019. Evaluation of the Context-Free Path Querying Algorithm Based on Matrix Multiplication. In *Proceedings of the 2Nd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA) (GRADES-NDA'19)*. ACM, New York, NY, USA, Article 12, 5 pages. https://doi.org/10.1145/3327964.3328503

[11] Thomas Reps. 1997. Program Analysis via Graph Reachability. In *Proceedings of the 1997 International Symposium on Logic Programming (ILPS '97)*. MIT Press, Cambridge, MA, USA, 5–19. http://dl.acm.org/citation.cfm?id=271338.271343

[12] Fred C. Santos, Umberto S. Costa, and Martin A. Musicante. 2018. A Bottom-Up Algorithm for Answering Context-Free Path Queries in Graph Databases. In *Web Engineering*, Tommi Mikkonen, Ralf Klamma, and Juan Hernández (Eds.). Springer International Publishing, Cham, 225–233.

[13] TAISUKE SATO. 2017. A linear algebraic approach to datalog evaluation. *Theory and Practice of Logic Programming* 17, 3 (May 2017), 244–265. https://doi.org/10.1017/s1471068417000023

[14] Petteri Sevon and Lauri Eronen. 2008. Subgraph Queries by Context-free Grammars. *Journal of Integrative Bioinformatics* 5, 2 (June 2008). https://doi.org/10.1515/jib-2008-100

[15] Robert. Tarjan. 1972. Depth-First Search and Linear Graph Algorithms. *SIAM J. Comput.* 1, 2 (1972), 146–160. https://doi.org/10.1137/0201010

[16] Ekaterina Verbitskaia, Ilya Kirillov, Ilya Nozkin, and Semyon Grigorev. 2018. Parser Combinators for Context-free Path Querying. In *Proceedings of the 9th ACM SIGPLAN International Symposium on Scala (Scala 2018)*. ACM, New York, NY, USA, 13–23. https://doi.org/10.1145/3241653.3241655

[17] Mihalis Yannakakis. 1990. Graph-theoretic Methods in Database Theory. In *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '90)*. ACM, New York, NY, USA, 230–242. https://doi.org/10.1145/298514.298576

[18] Qirun Zhang, Michael R. Lyu, Hao Yuan, and Zhendong Su. 2013. Fast Algorithms for Dyck-CFL-reachability with Applications to Alias Analysis. In *Proceedings of the 34th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '13)*. ACM, New York, NY, USA, 435–446. https://doi.org/10.1145/2491956.2462159

[19] X. Zhang, Z. Feng, X. Wang, G. Rao, and W. Wu. 2016. Context-free path queries on RDF graphs. In *International Semantic Web Conference*. Springer, 632–648.