Unified Algorithm to Solve several Graph Problems with Relational Queries

Wellington Cabrera, Carlos Ordonez

Department of Computer Science, University of Houston, USA

Abstract. Several important graph algorithms can be solved as an iteration of vector-matrix multiplication over different semirings. On this basis, we show that the Bellman-Ford (single source shortest paths), reachability, PageRank, and topological sort algorithms can be expressed as relational queries, to solve analytic graph problems in relational databases. As a main contribution, we present a general algorithm that unifies all graph algorithms aforementioned.

1 Introduction

Relational databases remain the most common technology to store transactional and analytical databases, due to optimized I/O, robustness and security control. Although the common understanding is that relational database queries are not sufficient to express important graphs algorithms, some recent work has addressed the problem of solving graph algorithms with SQL queries. In our previous work [5], we proposed optimized recursive queries to compute matrix powers, to solve two important graph problems in the DBMS: Transitive closure and all pairs shortest path. In [6], the authors use recursive SQL to find constrained paths in RDF data, stored in a relational DBMS. Jindal et all [3] present some graph algorithms using a columnar database, showing that DBMS are competitive with specialized graph systems, namely Giraph and GraphLab. In this work we demonstrate that several graph algorithms can be expressed as a succession of efficient SPJA queries that are defined over the foundation of graph algorithms with vector-matrix operations. We believe that efficient graph algorithms for relational databases are a contribution that will support in-database graphs analytics in large data sets stored in databases.

2 Definitions

Graph Dataset Let $G = (V, E)$ be a directed graph, where V is a set of vertices and E is a set of edges, considered as an ordered pairs of vertices. Let $n = |V|$ vertices and $m = |E|$ edges. An edge (i, j) in E links two vertices in V and has a direction. We use E to name both the set of edges of G as well as the adjacency matrix, depending on the context. In general, real-life graphs are sparse. Therefore, it is reasonable to store the adjacency matrix E

in a sparse representation, which saves both computing, memory and storage resources. Several methods to represent sparse matrices are well known, and the interested reader may check [1]. In our work, the adjacency matrix of a graph G is stored in a table $E(i, j, v)$, with primary key (i, j). The sparse matrix representation avoids storing zeroes. Thus, the space required for table E is $\mathrm{O}(m)$, much smaller than the space necessary in the dense representation, $\mathrm{O}(n^2)$.

We consider a matrix *sparse* when the number of non-zero cells is $\mathrm{O}(n)$. Likewise, we consider a graph as sparse when the number of edges is $\mathrm{O}(n)$. Certain classes of graph are clearly sparse. For instance, both trees and spanning trees are graphs where $m = n - 1$. Many graph problems are solved computing powers of E. For large graphs computing such powers is challenging; even if E is a sparse graph, it is possible that after k multiplications by itself, the number of non-zero entries increases to $\mathrm{O}(n^2)$. Thus, E^k is not necessarily sparse.

Matrix Multiplication and Semirings Semirings are algebraic structures consisting of a set R, an additive operator $\oplus$ with identity element 0, a product operator $\otimes$ with identity element 1, and commutative, associative and distributive properties holding for the two operators in the usual manner, succinctly represented as $(R, \oplus, \otimes, 0, 1)$. For instance, the regular matrix multiplication is defined under $(\mathrm{R}, +, \times, 0, 1)$. A general definition of matrix multiplication expands it to any semiring. On the min-plus semiring, also known as "tropical semiring", min is the additive operator $\oplus$, and $+$ is the product operator $\otimes$. The min-plus semiring is used to solve shortest path problems, as in [2]. The boolean semiring, with $\vee$ (logical OR) as $\oplus$ operator and $\vee$ (logical OR) as $\otimes$ operator, is used frequently in linear algebra to represent some graph algorithms.

3 Graphs Algorithms computed with Relational Queries

We use as starting point known linear algebra approaches for Bellman-Ford, reachability and PageRank, as well as a novel way to express Topological Sort with matrix operations. On this foundation, we will show how all these algorithms can be expressed as relational queries. As is already known, Bellman-Ford and reachability can be solved by powers of the adjacency matrix and a vector-matrix multiplication. Topological sort also can be solved in the same manner, as we will show shortly. Likewise, PageRank can be solved by powers of the transition matrix. These four algorithms can be solved in a general way by Equation 1:

$$S_k = S_0 \cdot E^k = S_0 \cdot E \cdot E \cdot ... \tag{1}$$

where S_0 stores the initial state and S_k stores the final result. The last iteration is k. Instead of solving E^k, it is more efficient to compute S_k with iterative vector-matrix products: $S_d \leftarrow S_{d-1} \cdot E$ for $d = 1 \ldots k$. Algorithms are below.

Bellman-Ford: Shortest paths from a source s. Initialization: $S_0[i] = 0$ if $i = s, \infty$ otherwise. After k successive vector matrix multiplications (under the min $+$ semiring), the vector S_k contains the minimun distance from s to every vertex.

Reachability: Initialization: $S_0[i] = 0$ if $i = s$, 0 otherwise. Vector matrix multiplications are computed under the natural numbers semiring. At iteration d, $S_d[i]$ contains 1 if there exists a path of length d from the s to the vertex i.

PageRank: It is well known that PageRank can be computed as powers of a modified transition matrix [4]. Since PageRank is conceived as a Markov process, it can be computed as an iterative process that stops when the Markov chain stabilizes. Let $S_0[i] = 1/n$. The transition matrix T is defined as $T_{i,j} = E_{j,i}/outdeg(j)$ when $E_{j,i} = 1$; otherwise $T_{i,j} = 0$. Let A be a $n \times n$ matrix, whose cells contain always $1/n$, and $0 < p < 1$ the damping factor; let $T^{'} = T + D$, where D is a matrix such that $D_{i,j} = 1/n$ if the column j is a 0 column. The power method can be applied on $T^{''}$ defined as: $T^{''} = (1-p)T^{'} + pA$, computing $S_d \leftarrow S_{d-1} \cdot T^{"}$ iteratively, from $d = 1$ until stabilization of the Markov chain.

Topological Sort: S_d holds the relative ordering for the vertices, and $S_d[j] = 0$ when the relative order of j is not yet determined. Initialization: $S_0[j] = 1$ if outdeg$(j) = 0$, otherwise $S_0[j] = 1$. The algorithm can be outlined in this manner: For each vertex $j \in S_{d-1}$, find every father vertex π such that $\forall i$ successor of $\pi, i \in S$, and let $S_d[\pi] = 1$. Repeat the process until $\forall i \in V$, $S[i] \neq 0$. The topological order is given by $S_0 + S_1 + ...S_k$. This algorithm can be expressed as in Equation 1, where S_k is solved with iterative vector-matrix multiplication. In this case, the $\oplus$ operation is the logical AND: $\wedge$, and the $\otimes$ is the logical implication $\Rightarrow$. As result of the vector-matrix multiplication, $S_d[j] = 1$ if for all j' successors of j , $S_d[j'] = 1$. Although the operators $\wedge$ and $\Rightarrow$ cannot comprise a semiring (the commutative property for $\Rightarrow$ does not hold), it is possible to express $S_d \cdot E$ with SPJA queries, like in the previous algorithms.

Table 1. Comparison of four graphs algorithms

...	Bellman-Ford	Reachability	PageRank	Topological Sort
Computed as	$S_d \leftarrow S_{d-1} \cdot E$	$S_d \leftarrow S_{d-1} \cdot E$	$S_d \leftarrow S_{d-1} \cdot T$	$S_d \leftarrow S_{d-1} \cdot E$
Semiring op.$(\oplus, \otimes)$	$\min, +$	$+, \times$	$+, \times$	$\wedge, \Rightarrow$
Value $S[i]$	distance s to i	1 iif $\exists$ a path s to i	probability	order
S_0 defined as	$S_0[s] = 0$	$S_0[s] = 1$	$S_0[i] = 1/n$	$S_0[i]$=1 iff outdeg(i)=1
$\lvert S_0 \rvert$	1	1	n	$\lvert \{i \in V \mid outdeg(i) = 0\} \rvert$
Output	S_k	$S_0 + S_1 + ..S_k$	S_k	$S_0 + S_1 + ..S_k$
Time complexity	$O(kn \log n)$	$O(kn)$ or $O(kn \log n)$	$O(kn \log n)$	$O(kn \log n)$
Scope	from source s	from source s	$\forall i \in V$	$\forall i \in V$

4 Unified Algorithm

Considering that S and E are stored in relational tables $S(j, v)$ and $E(i, j, v)$, the vector-matrix product can be clearly computed with a relational query as: $\pi_{S.i,E.j:sum(S.v*E.v)}(S \bowtie_{S.j=E.i} E)$. Table 1 shows a comparison of the properties of the four algorithms of our concern. All of them are computed by a combined

Algorithm 1: Unified Algorithm

Input: Table E, S_0, R_0, $f(), g(), \otimes, \epsilon, unionFlag$. Optional input: s (source)
Output: Table $Rd(j, v)$
$d \leftarrow 0$; $\Delta \leftarrow 1$;
while $\Delta > \epsilon$ **do**
  $d \leftarrow d + 1$;
  $S_d \leftarrow \pi_{j:g()(S_{d-1}.v \otimes E.v)}(S_{d-1} \bowtie_{j=i} E)$
  if *unionFlag* **then**
    $R_d \leftarrow \pi_{j:sum(v)}(S_d \cup R_{d-1})$
  else
    $R_d \leftarrow S_d$
  end
  $\Delta \leftarrow f(R_{d-1}, S_{d-1}, R_d, S_d)$
end

join/aggregation between S and E (or T") as the main operation of the iteration. This SPJA query computes a vector-matrix multiplication on different semirings, depending on the specific algorithm, with time complexity O($nlogn$). The meaning of the value in $S[i]$ is different in every case. While Bellman-Ford and reachability return results concerning an unique source vertex, topological sort and PageRank return results with respect to the entire graph. The execution time of these algorithms will depend on n and the number of iterations k. For instance, reachability may finish in less iterations than Bellman-Ford, for the same graph. It is possible to conceptualize a Unified Algorithm to solve these four graphs problems, as presented in Algorithm 5. The main operation remains the matrix multiplication, with parameters $\oplus, \otimes$. Other input parameters are the initial vector S_0. and unionFlag, wich controls if the output is S_d or the cumulative sum of $S_0 + S_1 ... + S_d$. F

5 Conclusions

We demonstrate that Bellman-Ford, reachability, PageRank and Topological Sort can be expressed as an iteration of relational queries, to solve graph problems in data sets stored in databases. Based on the conceptual foundation of vector-matrix products under different semirings, these algorithms run as SPJA queries; they are secondary memory algorithms, not limited by the system RAM. The efficiency of our algorithms is supported by: sparse storage, which avoids wasting both space and computing time; conditions for early termination, which avoids useless iterations; and light weight relational queries. Moreover, we compared these algorithms and proposed a Unified Algorithm to solve them. Further research will identify and integrate into the Unified Algorithm more algorithms that follow the vector-matrix multiplication pattern. Besides, we will study these algorithms in large and complex graphs, especially in parallel clusters.

References

1. Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*, volume 11. Siam, 2000.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.
3. A. Jindal, P. Rawlani, E. Wu, S. Madden, A. Deshpande, and M. Stonebraker. Vertexica: your relational friend for graph analytics! *Proceedings of the VLDB Endowment*, 7(13):1669–1672, 2014.
4. S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the 12th Int. Conf. on World Wide Web*, pages 261–270. ACM, 2003.
5. C. Ordonez, W. Cabrera, and A. Gurram. Comparing columnar, row and array DBMSs to process recursive queries on graphs. *Information Systems*, pages –, 2016.
6. N. Yakovets, P. Godfrey, and J. Gryz. Evaluation of SPARQL property paths via recursive SQL. In *Proc. AMW*, 2013.