Fine-grained provenance for linear algebra operators

Zhepeng Yan Val Tannen Zachary G. Ives
University of Pennsylvania
{zhepeng, val, zives}@cis.upenn.edu

Abstract

Provenance is well-understood for relational query operators. Increasingly, however, data analytics is incorporating operations expressed through linear algebra: machine learning operations, network centrality measures, and so on. In this paper, we study provenance information for matrix data and linear algebra operations. Our core technique builds upon provenance for aggregate queries and constructs a $K-$semialgebra. This approach tracks provenance by annotating matrix data and propagating these annotations through linear algebra operations. We investigate applications in matrix inversion and graph analysis.

1. Introduction

For many years, data provenance was modeled purely as graphs capturing dataflows among "black box" operations: this representation is highly flexible and general, and has ultimately led to the PROV-DM standard. However, the database community has shown that *fine-grained* provenance with "*white-box*" operations [3] (specifically, for relational algebra operations) enables richer reasoning about the relationship between derived results and data provenance. The most general model for relational algebra queries, *provenance semirings* [7, 1], ensures that equivalent relational algebra expressions, as produced by a query optimizer, have equivalent provenance encodings. Moreover, there is a natural means of computing *trust* [6] and *authoritativeness* [16] scores for data, using provenance semiring expressions. Provenance can even be used to support *incremental updates* to views [6]. Finally, there are probabilistic mechanisms for learning weights or authoritativeness scores for different *provenance tokens* in the semiring model [16].

A natural question is whether the provenance semiring approach, to this point limited to variations on relational queries, can be extended to a greater subset of data analytics tasks, e.g., to other "bulk operations" over collections of data. We present a preliminary set of provenance primitives, for one such class of operations: those from *linear algebra*. Matrices and linear algebra are heavily used in many data analytics applications, including image detection, signal processing, network science, and machine learning. While provenance *libraries* have been developed for matrix operations [17], no techniques currently exist for *automatically* extracting provenance from the various operations, and for reasoning based on this provenance. Our work in this paper develops primitives for matrix operations such as addition, multiplication and inversion. We show building blocks towards applications in principal component analysis (PCA), frequently used for dimensionality reduction, and in computing PageRank-style scores over network graphs.

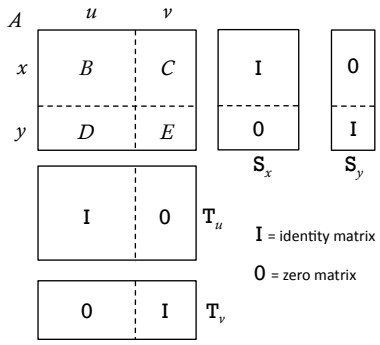

Figure 1. Provenance partitioning and its selectors

2. Tracking matrix data

Consider a $d \times n$ matrix A, and, as is often the case in machine learning, consider an algorithm that begins by computing AA^T. In a typical situation A records n samples from a d-dimensional *feature space*. Suppose we wish to track the provenance of the data by partitioning both the features and samples. In a very simple example the features might be of two types and we associate the *provenance tokens* x, respectively y, with these types of features.

Suppose also that the samples come from two sources and we associate the provenance tokens u, respectively v, with these sources. Again for simplicity, we assume that all the features associated with x correspond to adjacent rows, and similar for y, u, v. This partitions A into four submatrices B, C, D, E, see Figure 1.

Let x, y, u, v be from some set X of provenance tokens and let $\mathbb{N}[X]$ be the provenance semiring of polynomials developed in [7]. Now we wish to annotate the submatrix B with the polynomial xu corresponding to its source and feature subset, and similarly C with xv, D with yu, and E with yv. From this, we will later want to track which sources and feature subsets were used to produce which portions of an output matrix.

Observe that

$$A = \mathsf{S}_x\, B\, \mathsf{T}_u + \mathsf{S}_x\, C\, \mathsf{T}_v + \mathsf{S}_y\, D\, \mathsf{T}_u + \mathsf{S}_y\, E\, \mathsf{T}_v$$

with appropriate *selectors* $\mathsf{S}_x, \mathsf{S}_y, \mathsf{T}_u, \mathsf{T}_v$. Selectors (see Section 4) are certain matrices whose elements are 0 or 1, and serve like a "mask" specifying entries to exclude or include, respectively. In our case, let p, respectively q, be the number of rows of provenance x, respectively of provenance y, where $p + q = d$. Let also k, respectively l, be the number of columns of provenance u,

TaPP 2016, June 8–9, 2016, Washington, DC.

respectively of provenance v, where $k+l=n$. The dimensions of the selectors are $\mathsf{S}_x : d\times p, \mathsf{S}_y : d\times q, \mathsf{T}_u : k\times n, \mathsf{T}_v : l\times n$. Each of them consists of an identity matrix adjacent to a zero matrix, see Figure 1. Note that the selectors do not contain data that is subject to provenance tracking. We used x, y, u, v as subscripts in the notation to make it easier to connect the formulas to the figure.

In this form we can annotate B, etc. with provenance polynomials. Indeed, let $\mathcal{M}$ be the many-sorted [1] ring of matrices of real numbers. In Section 3 we show how to embed $\mathcal{M}$ into a *semialgebra* of matrices annotated with polynomials, namely the *tensor product* $\mathbb{N}[X]\otimes\mathcal{M}$. Indeed, B corresponds to $1\otimes B \in \mathbb{N}[X]\otimes\mathcal{M}$ and annotation with $xu\in\mathbb{N}[X]$ corresponds to multiplication with scalars: $xu*(1\otimes B) = xu\otimes B$. In turn, the selectors, since they do not contain data that needs to be tracked, remain with the neutral annotation, e.g., $1\otimes \mathsf{S}_x$. *For simplicity we denote* $1\otimes M$ *with* M *for any matrix* M. Hence

$$\begin{aligned} A &= \mathsf{S}_x\,(xu*B)\,\mathsf{T}_u + \mathsf{S}_x\,(xv*C)\,\mathsf{T}_v \\ &+ \mathsf{S}_y\,(yu*D)\,\mathsf{T}_u + \mathsf{S}_y\,(yv*E)\,\mathsf{T}_v \end{aligned}$$

The decomposition into submatrices using selectors is nicely compatible with transposition. Indeed we have:

$$\begin{aligned} A^T &= \mathsf{T}_u^T\,(xu*B^T)\,\mathsf{S}_x^T + \mathsf{T}_u^T\,(yu*D^T)\,\mathsf{S}_y^T \\ &+ \mathsf{T}_v^T\,(xv*C^T)\,\mathsf{S}_x^T + \mathsf{T}_v^T\,(yv*E^T)\,\mathsf{S}_y^T \end{aligned}$$

Now to multiply AA^T we can (as will be justified in Section 3) use any identity that is valid in matrix algebra (associativity, distributivity, etc.). We multiply two sums of four products each so that will give a sum of sixteen products. Let's just look at two of them

$$\mathsf{S}_x(xu*B)\mathsf{T}_u\mathsf{T}_u^T(yu*D^T)\mathsf{S}_y^T \quad \mathsf{S}_x(xv*C)\mathsf{T}_v\mathsf{T}_v^T(yv*E^T)\mathsf{S}_y^T$$

We will show in Section 4 that $\mathsf{T}_u\,\mathsf{T}_u^T$ and $\mathsf{T}_v\,\mathsf{T}_v^T$ are actually identity matrices. Moreover, it will follow from Section 3 that

$$(xu{*}B)\,(yu{*}D^T) = xyu^2{*}BD^T \quad (xv{*}C)\,(yv{*}E^T) = xyv^2{*}CE^T$$

With these observations it has become clear that

$$\begin{aligned} AA^T &= \mathsf{S}_x(x^2u^2*BB^T + x^2v^2*CC^T)\mathsf{S}_x^T \\ &+ \mathsf{S}_x(xyu^2*BD^T + xyv^2*CE^T)\mathsf{S}_y^T \\ &+ \mathsf{S}_y(xyu^2*DB^T + xyv^2*EC^T)\mathsf{S}_x^T \\ &+ \mathsf{S}_y(y^2u^2*DD^T + y^2v^2*EE^T)\mathsf{S}_y^T \end{aligned}$$

Where are the other eight products? They become 0 since we show in Section 4 that $\mathsf{T}_u\,\mathsf{T}_v^T$ and $\mathsf{T}_v\,\mathsf{T}_u^T$ are actually 0 matrices!

Finally, to illustrate the flexibility of the framework that we develop in Section 3 we remark that the terms of the expressions can often be rewritten to emphasize various contributions by provenance. For example we can rearrange the expression for AA^T:

$$\begin{aligned} AA^T &= x^2*(u^2*\mathsf{S}_xBB^T\mathsf{S}_x^T + v^2*\mathsf{S}_xCC^T\mathsf{S}_x^T) \\ &+ xy*[\,u^2*(\mathsf{S}_xBD^T\mathsf{S}_y^T + \mathsf{S}_yDB^T\mathsf{S}_x^T) \\ &\qquad + v^2*(\mathsf{S}_xCE^T\mathsf{S}_y^T + \mathsf{S}_yEC^T\mathsf{S}_x^T)\,] \\ &+ y^2*(u^2*\mathsf{S}_yDD^T\mathsf{S}_y^T + v^2*\mathsf{S}_yEE^T\mathsf{S}_y^T) \end{aligned}$$

3. The semialgebra of annotated matrices

The $m\times n$ real matrices for various $m,n\geq 1$ form a *many-sorted ring* $(\mathcal{M},+,\cdot,0,\mathsf{I})$. As usual with matrices, when we write $A+B$ or CD we will assume that the sorts (dimensions) of A,B and C,D are such that the operations are defined.

[1] Because matrices can be added or multiplied only when their dimensions are compatible.

Although in Section 2 we only made use of embedding the ring of matrices into a $\mathbb{N}[X]$-semialgebra, the construction works more generally and this generality might prove useful in future research projects that use different provenance semirings, as in [6].

Let $(K,+_K,\cdot_K,0_K,1_K)$ be a *commutative semiring*. In previous work [1] it was shown, using a tensor product-like construction, how to embed *aggregation domains*, (specifically, commutative monoids) into "most economical" K-semimodules that allow us to track the provenance captured by the elements of K. Here we consider matrices, a more complicated aggregation domain, since it has *two* operations (+ and $\cdot$) so we aim for a richer algebraic structure, that of (many-sorted) *semialgebra*. We shall embed $\mathcal{M}$ into a structure that, like $\mathcal{M}$, forms a ring $(K\otimes\mathcal{M},+,\cdot,\mathsf{0},\mathsf{I})$ (for simplicity we use for the operations the same notation we used for $\mathcal{M}$). In addition, $K\otimes\mathcal{M}$ forms a K-semialgebra, i.e., it also has $*$, a binary operation $K\times(K\otimes\mathcal{M})\to K\otimes\mathcal{M}$ that satisfies the following algebraic identities.

DEFINITION 1. $\forall\ k,k_1,k_2\in K \quad \forall\ A,A_1,A_2\in K\otimes\mathcal{M}$

$$\begin{aligned} k*(A_1+A_2) &= k*A_1+k*A_2 && (1)\\ k*\mathsf{0} &= \mathsf{0} && (2)\\ (k_1+_K k_2)*A &= k_1*A+k_2*A && (3)\\ 0_K*A &= \mathsf{0} && (4)\\ (k_1\cdot_K k_2)*A &= k_1*(k_2*A) && (5)\\ 1_K*A &= A && (6)\\ (k_1*A_1)(k_2*A_2) &= (k_1\cdot_K k_2)*(A_1A_2) && (7) \end{aligned}$$

(In particular, $(K\otimes\mathcal{M},+,\mathsf{0},*)$ *is a (many-sorted)* K*-semimodule.)*

In Section 2 we used the identity $(xu*B)\,(yu*D^T) = xyu^2*BD^T$. Indeed, this is an instance of semialgebra axiom (7) above. We proceed to sketch the construction of $K\otimes\mathcal{M}$.

Tensor product construction We start with $K\times\mathcal{M}$, sorted just like $\mathcal{M}$, denote its elements $k\otimes A$ instead of $\langle k,A\rangle$ and call them "simple tensors". Next we consider the set $\mathrm{Bag}(K\times\mathcal{M})$ of finite bags of such simple tensors, i.e., functions $f: K\times\mathcal{M}\to\mathbb{N}$ whose *support*, i.e., $\mathrm{supp}(f)=\{k\otimes A\in K\times\mathcal{M}\mid f(k\otimes A)\neq 0\}$, is *finite*. Let $\uplus$ be the usual bag union and $\emptyset_+$ be the empty bag. For $f_1,f_2\in\mathrm{Bag}(K\times\mathcal{M})$ define the *("convolution") product* as

$$f_1\star f_2 \;=\; \lambda(k\otimes A).\ \textstyle\sum f_1(k_1\otimes A_1)f_2(k_2\otimes A_2)$$

where the sum is over all $k_1\otimes A_1\in\mathrm{supp}(f_1)$ and $k_2\otimes A_2\in\mathrm{supp}(f_2)$ such that $k_1\cdot_K k_2 = k$ and $A_1A_2=A$. [2] Like matrix multiplication, $\star$ is not commutative.

PROPOSITION 2. $(Bag(K\times\mathcal{M}),\uplus,\star,\emptyset_+,1_K\otimes\mathsf{I})$ *is a semiring.*

Define a function $\iota:\mathcal{M}\to\mathrm{Bag}(K\times\mathcal{M})$ such that $\iota(A)=1_K\otimes A$ (actually, the singleton bag with element $1_K\otimes A$). It is easy to see that ι is injective and is a homomorphism with respect to multiplication. However, it is not a homomorphism with respect to addition. This will be one of the things we will fix when we take equivalence modulo a certain congruence, but there will be a price to pay: ι will not stay injective, in general. Next we define *multiplication with scalars from* K on $\mathrm{Bag}(K\times\mathcal{M})$:

$$k*f = \lambda(k'\otimes A).\ (k\cdot_K k')\otimes A$$

PROPOSITION 3. $(Bag(K\times\mathcal{M}),\uplus,\star,\emptyset_+,1_K\otimes\mathsf{I})$ *satisfies axioms (1,2,5,6,7) of Definition 1 but not axiom (3) or axiom (4).*

Thus we have found that the algebraic structure on bags does not identify enough elements to insure that it is a K-semialgebra and

[2] These algebraic constructions are related to those of [11].

that ι is a semiring homomorphism. Thus, we will enforce the desired identifications by quotienting with a congruence. Let $\sim$ be the smallest congruence on $\mathrm{Bag}(K\times\mathcal{M})$ with respect to $\uplus, \star$ and $*$ that satisfies (for all k_1, k_2, A, A_1, A_2):

$$(k_1 +_K k_2)\otimes A \sim k_1\otimes A \uplus k_2\otimes A \quad (8)$$
$$0_K\otimes A \sim \emptyset_+ \quad (9)$$
$$1_K\otimes(A_1 + A_2) \sim 1_K\otimes A_1 \uplus 1_K\otimes A_2 \quad (10)$$
$$1_K\otimes\mathsf{0} \sim \emptyset_+ \quad (11)$$

Denote by $K\otimes\mathcal{M} = \mathrm{Bag}(K\times\mathcal{M})/\sim$. Its elements could be called "tensors" since they are equivalence classes of bags of simple tensors modulo $\sim$ and this whole construction generalizes a tensor product construction (to the "semi-" situation). Because $\sim$ is a congruence with respect to $\uplus, \star$ and $*$, corresponding operations are defined on the quotient set. As before, we abuse notation by denoting them again by $+, \cdot, *$, We also use ι again for the function that maps every $A \in \mathcal{M}$ to the $\sim$-equivalence class of $1_K\otimes A$.

PROPOSITION 4. *$K\otimes\mathcal{M}$ is a K-semialgebra and $\iota : \mathcal{M} \to K\otimes\mathcal{M}$ is a semiring homomorphism satisfying the following universality property: for any K-semialgebra S and any semiring homomorphism $f : \mathcal{M} \to S$ there exists a unique homomorphism of K-semialgebras $f^* : K\otimes\mathcal{M} \to S$ such that $f = f^* \circ \iota$.*

Therefore , $K\otimes\mathcal{M}$ is the "most economical" (*freely generated*) K-semialgebra that $\mathcal{M}$ can be embeded into. As discussed in [1], ι is not injective, in general. However, it is injective for $K = \mathbb{N}[X]$. This justifies the identification of the annotated matrices $1\otimes A$ with A in Section 2.

4. Provenance propagation by decomposition

We have seen in Section 2 an example, in which a matrix is decomposed into several contiguous blocks, each annotated with a provenance polynomial. We have also illustrated there how to propagate provenance under several operators. However, elements with the same annotation are not necessarily adjacent. In this section, we generalize provenance decomposition and propagation to non-contiguous cases. We will build upon the notion of *selector matrix*.

DEFINITION 5. *A matrix S is a* selector matrix *if it is a zero matrix or can be constructed from inserting rows and columns of zeros into an identity matrix. A selector matrix has at most a 1 in each row and in each column.*

Fix a matrix $A_{m\times n}$. We consider its submatrix defined by a set of rows and columns, as follows. Let $R = \{r_1, r_2, \cdots, r_{m'}\}$ consist of m' distinct row indices and $C = \{c_1, c_2, \cdots, c_{n'}\}$ consist of n' distinct column indices, where $\forall i, r_i \le r_{i+1}$ and $\forall j, c_j \le c_{j+1}$. Let A' be such matrix that $A'(i,j) = A(i,j)$ when $i \in R \wedge j \in C$, and $A'(i,j) = 0$ otherwise. Let A_s be such matrix that $A_s(i,j) = A(r_i, c_j)$. There exist selectors S and T such that

$$\mathsf{S}A_s\mathsf{T} = A',$$

where row r_i of S has a 1 in column i and column c_j of T has a 1 in row j. We visualize S below (T is similarly structured).

$$\mathsf{S}_{m\times m'} = \begin{array}{c} \\ r_1 \\ \\ r_2 \\ \\ \vdots \\ r_{m'} \\ \end{array} \begin{bmatrix} & \mathbf{O}_{(\mathbf{r_1}-\mathbf{1})\times\mathbf{n'}} & & \\ 1 & 0 & \cdots & 0 \\ & \mathbf{O}_{(\mathbf{r_2}-\mathbf{r_1}-\mathbf{1})\times\mathbf{n'}} & & \\ 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \\ & \mathbf{O}_{(\mathbf{m}-\mathbf{r_{m'}})\times\mathbf{n'}} & & \end{bmatrix}$$

PROPOSITION 6. *Given a matrix $A_{m\times n}$, let $\{(R_i, C_i)\}$ be a set of pairs of rows and columns such that $\{R_i\}$ forms a partition of $[1..m]$ and $\{C_i\}$ forms a partition of $[1..n]$. Let A_i be the submatrix defined by R_i and C_i (as above). Then, there exist selector matrices S_i and T_i such that*

$$A = \sum_i \mathsf{S}_i A_i \mathsf{T}_i.$$

This decomposition method enables analysis of provenance propagation under various linear operations. For example,

$$A^T = (\textstyle\sum_i \mathsf{S}_i A_i \mathsf{T}_i)^T = \sum_i (\mathsf{S}_i A_i \mathsf{T}_i)^T = \sum_i \mathsf{T}_i^T A_i^T \mathsf{S}_i^T.$$

For multiplying two matrices $A = \sum_i \mathsf{S}_i A_i \mathsf{T}_i$ and $B = \sum_j \mathsf{U}_j B_j \mathsf{V}_j$,

$$AB = \textstyle\sum_{i,j} \mathsf{S}_i A_i (\mathsf{T}_i\mathsf{U}_j) B_j \mathsf{V}_j. \quad (12)$$

One can verify that $(\mathsf{T}_i\mathsf{U}_j)$ is also a selector matrix. Furthermore, since not all columns in A_i and all rows in B_j contribute to $A_i(\mathsf{T}_i\mathsf{U}_j)B_j$, summands in Equation 12 satisfy the following.

PROPOSITION 7. *When $\mathsf{S}_i A_i (\mathsf{T}_i\mathsf{U}_j) B_j \mathsf{V}_j$ does not evaluate to the zero matrix, there exist submatrix A_i' of A_i and B_j' of B_j such that*

$$\mathsf{S}_i A_i (\mathsf{T}_i\mathsf{U}_j) B_j \mathsf{V}_j = \mathsf{S}_i A_i' B_j' \mathsf{V}_j$$

5. Application: matrix inversion

Tracking provenance through determinant computation and through exact matrix inversion seems to involve provenance that is too complex to be easily usable. Instead, we illustrate in the following example the Jacobi method [4] for iteratively computing solutions to systems of linear equations.[3] We consider the solution to $(A + B)x = b$, where A and B are from two sources, with provenance p and q, respectively:

$$A = p * \begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix}, B = q * \begin{bmatrix}0 & 0\\ -2 & 0\end{bmatrix}, b = \begin{bmatrix}1\\ -1\end{bmatrix}$$

The iterative method first splits $A + B = M - N$, where

$$M = p * \begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}, N = p * \begin{bmatrix}0 & -1\\ -1 & 0\end{bmatrix} + q * \begin{bmatrix}0 & 0\\ 2 & 0\end{bmatrix}$$

Then it iteratively computes $u_{k+1} = M^{-1}Nu_k + M^{-1}b$. Suppose that we start from $u_0 = [0, 0]^T$, then

$$u_1 = p * \begin{bmatrix}\frac{1}{2}\\ -\frac{1}{2}\end{bmatrix}, u_2 = p * \begin{bmatrix}\frac{1}{2}\\ -\frac{1}{2}\end{bmatrix} + p^3 * \begin{bmatrix}\frac{1}{4}\\ -\frac{1}{4}\end{bmatrix} + p^2q * \begin{bmatrix}0\\ -\frac{1}{2}\end{bmatrix}, \cdots$$

Fully accepting all the data, i.e., $p = 1$ $q = 1$, results in the vector u converging to $[\frac{3}{5}, -\frac{1}{5}]^T$. Provenance tracking also enables incremental deletion propagation: when $p = 1, q = 0$, we have $u_k = [1 - \frac{1}{2^k}, -1 + \frac{1}{2^k}]^T$.

6. Application: largest eigenvalue

In principal component analysis (PCA), the largest eigenvalue describes the amount of variance of data projected onto the corresponding eigenvector. Here we illustrate provenance tracking in computing the largest eigenvalue using the power method. Suppose that V is the following matrix:

$$V = p * \begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix} + q * \begin{bmatrix}3 & 1\\ 1 & 3\end{bmatrix},$$

The first summand's eigenvalues $\lambda_1 = 3$ and $\lambda_2 = 1$ while the second's are $\lambda_1 = 4$ and $\lambda_2 = 2$. The power method starts

[3] Matrix inversion can be reduced to solving systems of linear equations

with a vector $\mathbf{u}_0$, iteratively computes $\mathbf{u}_{t+1} = V\mathbf{u}_t$, and returns $\mathbf{u}^1_{t'+1}/\mathbf{u}^1_{t'}$ for some t'. Starting from $\mathbf{u}_0 = [2,1]^T$, we have:

$$\begin{aligned}\mathbf{u}_2^T &= p^2 * [14,13]^T + pq * [38,34]^T + q^2 * [26,22]^T \\ \mathbf{u}_3^T &= p^3 * [41,40]^T + p^2q * [165,161]^T \\ &\quad + pq^2 * [222,210]^T + q^3 * [100,92]^T\end{aligned}$$

This procedure can continue, however, we shall use deletion propagation to verify that three iterations give a pretty good approximation. With $p = 1$ and $q = 0$ we have $\mathbf{u}^1_3/\mathbf{u}^1_2 = 41/14 \approx 3$, while with $p = 0$ and $q = 1$ we have $\mathbf{u}^1_3/\mathbf{u}^1_2 = 100/26 \approx 4$.

7. Application: PageRank

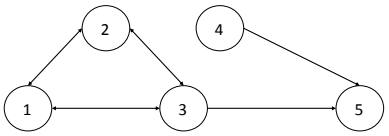

Figure 2. A sample graph.

Consider the sample graph in Figure 2, with transition matrix

$$A = \begin{bmatrix} 0 & 1/2 & 1/3 & 0 & 1/5 \\ 1/2 & 0 & 1/3 & 0 & 1/5 \\ 1/2 & 1/2 & 0 & 0 & 1/5 \\ 0 & 0 & 0 & 0 & 1/5 \\ 0 & 0 & 1/3 & 1 & 1/5 \end{bmatrix}$$

Note that since node 5 has no outgoing edges, entries in the corresponding column are equal to 1/5. The PageRank matrix is

$$M = (1-\alpha)A + \alpha B$$

where $B = \frac{1}{5}[1,1,\cdots,1]^T[1,1,\cdots,1]$ and $\alpha \in (0,1)$.

We now add some annotations. Suppose that A and B are annotated with p_1 and p_2 respectively, to capture the coefficients (i.e., $M = p_1 * A + p_2 * B$). In addition, outgoing edges from nodes 1-3 are annotated with q_1 and the outgoing edge from node 4 is annotated with q_2. The annotated PageRank matrix is given by $M = M_0 + p_1q_1 * M_1 + q_2 * M_2 + p_2 * B$, where

$$\begin{aligned} M_0 &= [1,1,1,1,1]^T[0,0,0,0,1/5] \\ M_1 &= \begin{bmatrix} 0 & 1/2 & 1/3 & 0 & 0 \\ 1/2 & 0 & 1/3 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 0 & 0 \end{bmatrix} \\ M_2 &= [0,0,0,0,1]^T[0,0,0,1,0] \end{aligned}$$

The power method starts from an initial PageRank score vector $v_0 = [\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5}]^T$ and iteratively computes $v_{t+1} = Mv_t$. We can track provenance as in previous examples.

8. Related work

The core building block of our algebraic construction is based on provenance semirings [7]. Our framework goes beyond provenance for aggregate queries [1], by constructing a K-semialgebra instead of just a K-semimodule. Although there are several array database management systems such as SciDB [15] and RasDaMan [2], only until recently have various aspects of provenance been investigated. This line of work includes APIs for encoding and querying for lineage [17] and supporting uncertain array data [14]. Our work differs in *automatically* tracking provenance from the operations, and in preserving the semantics of the matrix operations.

Many systems support distributed machine learning based on MapReduce or iterative linear algebra programs [19, 18, 12, 8, 10, 5, 9]. In addition, LINView [13] proposes methods for incremental computation of iterative linear algebra programs when there are only small changes (low-rank updates) to input matrices. Generalizations of our preliminary work could be incorporated into such platforms based on their linear algebra operations.

9. Contributions and future work

This paper has presented the first steps towards a semantics-preserving notion of fine-grained provenance for linear algebra operations. While more work needs to be done, we believe there is great promise for automatically tracking provenance in a wide variety of applications. In the future, building upon our basic construction, we seek to investigate provenance for more complex linear algebra operators. We will also study how to implement this framework in a real system.

References

[1] Y. Amsterdamer, D. Deutch, and V. Tannen. Provenance for aggregate queries. In *PODS*, pages 153–164, 2011.

[2] P. Baumann, A. Dehmel, P. Furtado, R. Ritsch, and N. Widmann. The multidimensional database system rasdaman. *ACM SIGMOD Record*, 27(2), 1998.

[3] J. Cheney, L. Chiticariu, and W. C. Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4): 379–474, 2009.

[4] P. G. Ciarlet, B. Miara, and J.-M. Thomas. *Introduction to numerical linear algebra and optimisation*. Cambridge University Press, 1989.

[5] A. Ghoting, R. Krishnamurthy, E. Pednault, B. Reinwald, V. Sindhwani, S. Tatikonda, Y. Tian, and S. Vaithyanathan. Systemml: Declarative machine learning on mapreduce. In *ICDE*, 2011.

[6] T. J. Green, G. Karvounarakis, Z. G. Ives, and V. Tannen. Update exchange with mappings and provenance. In *VLDB*, 2007. Amended version available as Univ. of Pennsylvania report MS-CIS-07-26.

[7] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, 2007.

[8] J. M. Hellerstein, C. Ré, F. Schoppmann, D. Z. Wang, E. Fratkin, A. Gorajek, K. S. Ng, C. Welton, X. Feng, K. Li, et al. The MADlib analytics library: or MAD skills, the SQL. *PVLDB*, 5(12), 2012.

[9] B. Huang, S. Babu, and J. Yang. Cumulon: Optimizing statistical data analysis in the cloud. In *SIGMOD*, 2013.

[10] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In *EuroSys*, pages 59–72, 2007.

[11] C. Koch. Incremental query evaluation in a ring of databases. Technical Report 183766, EPFL, 2013. Extended version of the PODS 2010 paper.

[12] T. Kraska, A. Talwalkar, J. C. Duchi, R. Griffith, M. J. Franklin, and M. I. Jordan. MLbase: A distributed machine-learning system. In *CIDR*, 2013.

[13] M. Nikolic, M. ElSeidy, and C. Koch. Linview: incremental view maintenance for complex analytical queries. In *SIGMOD*, 2014.

[14] L. Peng and Y. Diao. Supporting data uncertainty in array databases. In *SIGMOD*, 2015.

[15] M. Stonebraker, J. Becla, D. J. DeWitt, K.-T. Lim, D. Maier, O. Ratzesberger, and S. B. Zdonik. Requirements for science data bases and scidb. In *CIDR*, 2009.

[16] P. P. Talukdar, M. Jacob, M. S. Mehmood, K. Crammer, Z. G. Ives, F. Pereira, and S. Guha. Learning to create data-integrating queries. In *VLDB*, 2008.

[17] E. Wu, S. Madden, and M. Stonebraker. Subzero: a fine-grained lineage system for scientific databases. In *ICDE*, 2013.

[18] R. S. Xin, J. E. Gonzalez, M. J. Franklin, and I. Stoica. GraphX: A resilient distributed graph system on Spark. In *Workshop on Graph Data Management Experiences and Systems*, 2013.

[19] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *NSDI*, 2012.

A. Appendix

We first prove Proposition 2.

LEMMA 8. *Let X be a set. $(Bag(X), \uplus, \emptyset_+)$ is a commutative monoid. Moreover, every non-empty bag can be uniquely expressed a a bag union of singletons, specifically, as the union of all its possibly repreated elements, i.e., $A = \uplus a_i$ for all $a_i \in \mathrm{supp}(A)$, each a_i appearing in the union $A(a_i)$ times.*

Proof We show that the following three axioms hold for any $A, B, C \in \mathrm{Bag}(X)$.

- (commutativity) $A \uplus B = \lambda x.\ A(x) + B(x) = \lambda x.\ B(x) + A(x) = B \uplus A$.
- (associativity) $(A \uplus B) \uplus C = (\lambda x.\ A(x) + B(x)) \uplus C = \lambda x.\ (A(x) + B(x)) + C(x)$
  $= \lambda x.\ A(x) + (B(x) + C(x)) = A \uplus (\lambda x.\ B(x) + C(x)) = A \uplus (B \uplus C)$.
- (identity for plus) Since $\emptyset_+(x) = 0$ for any x, we have $A \uplus \emptyset_+ = \emptyset_+ \uplus B = \emptyset_+$.

Therefore, $(\mathrm{Bag}(X), \uplus, \emptyset_+)$ is a commutative monoid.

Moreover, it follows immediately that for any bag A we have

$$A \;=\; \biguplus_{a \in \mathrm{supp}(A)} A(a)\, a$$

(where for integer $n > 0$ and bag B, $n\,B$ is short for the $\biguplus$ of n copies of B). For the uniqueness of this representation suppose that for some bag A we have $A = \biguplus a_i$ where the a_i's are singleton bags as well as elements of X (in view of our abuse of notation). Then $\mathrm{supp}(A)$ must consist exactly of the a_i's and each a_i appears in the $\biguplus$ exactly $A(a_i)$ times.
□

LEMMA 9. *Let M be a monoid and $A, B \in Bag(M)$.*

(i) $\mathrm{supp}(A \star B)$ *is finite.*

(ii) $(\biguplus a_i) \star (\biguplus b_j) = \biguplus a_i \cdot_M b_j$.

(iii) *For any finite $Y, Z \subseteq M$ such that $\mathrm{supp}(A) \subseteq Y$ and $\mathrm{supp}(B) \subseteq Z$ we have*

$$A \star B \;=\; \lambda x.\ \textstyle\sum A(y)B(z)$$

where the sum is over all $y \in Y$ and $z \in Z$ such that $y \cdot_M z = x$.

Proof For each part:

(i) Let $x \in M$. Because $\mathbb{N}$ is "positive", if $(A \star B)(x) \neq 0$ then at least one of the numbers in the sum $\sum A(y)B(z)$ is non-zero so there exist $y \in \mathrm{supp}(A)$ and $z \in \mathrm{supp}(B)$ such that $y \cdot_M z = x$. Therefore, $\cdot_M$ defines a surjection from $\mathrm{supp}(A) \times \mathrm{supp}(B)$ to $\mathrm{supp}(A \star B)$ which makes the latter finite.

(ii) To check this, observe that an x appears in $\biguplus a_i \cdot_M b_j$ exactly as many times as there are appearances of a y in $\biguplus a_i$ and of a z's in $\biguplus b_j$ such that $x = y \cdot_M z$.

(iii) Indeed, each $y \in Y \setminus \mathrm{supp}(A)$ has $A(y) = 0$ and thus contributes 0 to the sum. Same with $z \in Z \setminus \mathrm{supp}(B)$.

□

PROPOSITION 2. $(Bag(K \times \mathcal{M}), \uplus, \star, \emptyset_+, 1_K \otimes 1\!\!1)$ *is a semiring.*

Proof Let M be $K \times \mathcal{M}$. We have already seen (Lemma 8) that $(\mathrm{Bag}(M), \uplus, \emptyset_+)$ is a commutative monoid. We need to show that $(\mathrm{Bag}(M), \star, 1_M)$ is also a monoid (not necessarily commutative), that $\star$ distributes over $\uplus$, and that $\emptyset_+$ is the "zero" element for $\star$. Let A, B and C be any elements in $\mathrm{Bag}(M)$.

- We now show that $(\mathrm{Bag}(M), \star, 1_M)$ is a monoid as well, in other words, that $\star$ is associative and that 1_M is the "neutral element" for $\star$. We have

$$(A \star B) \star C = \lambda x. \sum_{(u \cdot_M v) \cdot_M w = x} A(u)B(v)C(w)$$
$$= \lambda x. \sum_{u \cdot_M (v \cdot_M w) = x} A(u)B(v)C(w) \;=\; A \star (B \star C).$$

  We also have, because $\mathrm{supp}(1_M) = \{1_M\}$:

$$A \star 1_M \;=\; \lambda x. \sum_{y \cdot_M 1_M = x} A(y)\,1 \;=\; \lambda x.\, A(x) \;=\; A,$$

  and similarly $1_M \star A = A$.
- For distributivity, we have

$$\begin{aligned} A \star (B \uplus C) &= \lambda x. \sum_{u \cdot_M v = x} A(u)(B(v) + C(v)) \\ &= \lambda x. \sum_{u \cdot_M v = x} A(u)B(v) + A(u)C(v) \\ &= (A \star B) \uplus (A \star C). \end{aligned}$$

  where we make us Lemma 9, Part (iii), to extend the unions from $\mathrm{supp}(B)$ and $\mathrm{supp}(C)$ to $\mathrm{supp}(B) \cup \mathrm{supp}(C) = \mathrm{supp}(B \uplus C)$. Similarly $(A \uplus B) \star C = (A \star C) \uplus (B \star C)$.
- Since $\mathrm{supp}(\emptyset_+)$ is empty, we have $A \star \emptyset_+ = \emptyset_+ \star B = \emptyset_+$.

Hence, $(\mathrm{Bag}(M), \uplus, \star, \emptyset_+, 1_M)$ is a semiring.
□

We now proceed to Proposition 3.

LEMMA 10. *The function $\iota : \mathcal{M} \to Bag(K \times \mathcal{M})$ that maps $s \in \mathcal{M}$ to the singleton bag with elements $1_K \otimes s$ is injective and is a homomorphism between the multiplicative structures of $\mathcal{M}$ and $Bag(K \times \mathcal{M})$.*

Proof ι is clearly injective. Let $s_1, s_2 \in \mathcal{M}$, we have

$$\begin{aligned} \iota(s_1)\iota(s_2) &= (1_K \otimes s_1)(1_K \otimes s_2) \;=\; (1_K \cdot_K 1_K) \otimes (s_1 \cdot_{\mathcal{M}} s_2) \\ &= 1_K \otimes (s_1 \cdot_{\mathcal{M}} s_2) \;=\; \iota(s_1 \cdot_{\mathcal{M}} s_2). \end{aligned}$$

□

PROPOSITION 3. $(Bag(K \times \mathcal{M}), \uplus, \star, \emptyset_+, 1_K \otimes 1\!\!1)$ *satisfies axioms (1,2,5,6,7) of Definition 1 but not axiom (3) or axiom (4).*

Proof We show the following for any $k, k' \in K$, $A = \biguplus k_i \otimes s_i \in \mathrm{Bag}(K \times \mathcal{M})$ and $A' = \biguplus k'_j \otimes s'_j \in \mathrm{Bag}(K \times \mathcal{M})$.

- $k * (A \uplus A') = (k * A) \uplus (k * A')$ directly by definition of $*$.
- $k * \emptyset_+ = \emptyset_+$ also directly by definition of $*$.
- We have

  $(k +_K k') * A = \biguplus ((k +_K k') \cdot_K k_i) \otimes s_i = \biguplus (k \cdot_K k_i +_K k' \cdot_K k_i) \otimes s_i$

and we also have

$$(k * A) \uplus (k' * A) = \biguplus (k \cdot_K k_i) \otimes s_i \uplus (k' \cdot_K k_i) \otimes s_i$$

To complete the verification of axiom (3) we would like to have

$$(k_1 +_K k_2) \otimes s \quad = \quad k_1 \otimes s \uplus k_2 \otimes s$$

but this does not hold in general.

- We have $0_K * A = \biguplus (0_K \cdot_K k_i) \otimes s_i = \biguplus 0_K \otimes s_i$. To complete the verification of axiom (4) we would like to have

$$0_K \otimes s \quad = \quad \emptyset_+$$

but again this does not hold in general.

- $(k \cdot_K k') * A = \biguplus ((k \cdot_K k') \cdot_K k_i) \otimes s_i = \biguplus (k \cdot_K (k' \cdot_K k_i)) \otimes s_i$
  $= k * \biguplus (k' \cdot_K k_i) \otimes s_i = k * (k' * A)$.
- $1_K * A = \biguplus (1_K \cdot_K k_i) \otimes s_i = \biguplus k_i \otimes s_i = A$.
- $(k * A) \star A' = (\biguplus_i (k \cdot_K k_i) \otimes s_i) \star (\biguplus_j k'_j \otimes s'_j)$
  $= \biguplus_{i,j} ((k \cdot_K k_i) \otimes s_i)(k'_j \otimes s'_j)$
  $= \biguplus_{i,j} ((k \cdot_K k_i) \cdot_K k'_j) \otimes (s_i \cdot_{\mathcal{M}} s'_j)$
  $= \biguplus_{i,j} (k \cdot_K (k_i \cdot_K k'_j)) \otimes (s_i \cdot_{\mathcal{M}} s'_j)$
  $= k * \biguplus_{i,j} (k_i \cdot_K k'_j) \otimes (s_i \cdot_{\mathcal{M}} s'_j) = k * (A \star A')$
- $A \star (k * A') = (\biguplus_i k_i \otimes s_i) \star (\biguplus_j (k \cdot_K k'_j) \otimes s'_j)$
  $= \biguplus_{i,j} (k_i \otimes s_i)((k'_j \cdot_K k) \otimes s'_j)$
  $= \biguplus_{i,j} (k_i \cdot_K (k \cdot_K k'_j)) \otimes (s_i \cdot_{\mathcal{M}} s'_j)$
  $= \biguplus_{i,j} (k \cdot_K (k_i \cdot_K k'_j)) \otimes (s_i \cdot_{\mathcal{M}} s'_j) = k * (A \star A')$.

Note that for the last axiom we have made essential use of the fact that K is a *commutative* semiring.
□
Finally, we prove Proposition 4.

PROPOSITION 4. *$K \otimes \mathcal{M}$ is a K-semialgebra and $\iota : \mathcal{M} \to K \otimes \mathcal{M}$ is a semiring homomorphism satisfying the following universality property: for any K-semialgebra S and any semiring homomorphism $f : \mathcal{M} \to S$ there exists a unique homomorphism of K-semialgebras $f^* : K \otimes \mathcal{M} \to S$ such that $f = f^* \circ \iota$.*

Proof As is the case with any algebraic congruence, equational axioms that hold in the original structure continue to hold in the quotient structure. Therefore, $(K \otimes \mathcal{M}, +_{K\otimes\mathcal{M}}, \cdot_{K\otimes\mathcal{M}}, 0_{K\otimes\mathcal{M}}, 1_{K\otimes\mathcal{M}})$ is also a semiring, $*_{K\otimes\mathcal{M}}$ satisfies axioms (1,2,5,6,7) of semialgebras and ι is a homomorphism for the multiplicative structures. Moroever, after the quotienting, axioms (3,4) also hold as we have shown in the proof of Proposition 3 and ι is also a homomorphism for the additive structures because (8,9) now hold in $K \otimes \mathcal{M}$.

We now proceed to the second part. We use $+_S, \cdot_S, *_S$ for operators in S and $+_{K\otimes\mathcal{M}}, \cdot_{K\otimes\mathcal{M}}, *_{K\otimes\mathcal{M}}$ for operators in $K \otimes \mathcal{M}$. Define f^* first on bags of simple tensors as follows

$$f^*(\sum k_i \otimes s_i) = \sum k_i *_S f(s_i).$$

The first total sum uses $+_{K\otimes\mathcal{M}}$ and the second uses $+_S$. Thus $f^*(\iota(s)) = f^*(1_K \otimes s) = 1_K *_A f(s) = f(s)$. Then, we check that f^* is a homomorphism with respect to $+_S, \cdot_S$ and $*_S$, as follows.

- $f^*((\sum k_i \otimes s_i) +_{K\otimes\mathcal{M}} (\sum k'_j \otimes s'_j))$
  $= (\sum k_i *_S f(s_i)) +_S (\sum k'_j *_S f(s'_j))$
  $= f^*(\sum k_i \otimes s_i) +_S f^*(\sum k'_j \otimes s'_j)$.
- $f^*((\sum k_i \otimes s_i) \cdot_{K\otimes\mathcal{M}} (\sum k'_j \otimes s'_j))$
  $= f^*(\sum_{i,j} (k_i \otimes s_i) \cdot_{K\otimes\mathcal{M}} (k'_j \otimes s'_j))$
  $= f^*(\sum_{i,j} (k_i \cdot_K k'_j) \otimes (s_i \cdot_{\mathcal{M}} s'_j))$
  $= \sum_{i,j} (k_i \cdot_K k'_j) *_S f(s_i \cdot_{\mathcal{M}} s'_j)$
  $= \sum_{i,j} (k_i \cdot_K k'_j) *_S (f(s_i) \cdot_S f(s'_j))$
  $= \sum_{i,j} (k_i *_S f(s_i)) \cdot_S (k'_j *_S f(s'_j))$
  $= (\sum_i (k_i *_S f(s_i))) \cdot_S (\sum_j (k'_j *_S f(s'_j)))$
  $= f^*(\sum k_i \otimes s_i) \cdot_S f^*(\sum k'_j \otimes s'_j)$.
- For any $k \in K$, $f^*(k *_{K\otimes\mathcal{M}} (\sum k_i \otimes s_i))$
  $= f^*(\sum (k \cdot_K k_i) \otimes s_i) = \sum (k \cdot_K k_i) *_S f(s_i)$
  $= k *_S (\sum k_i *_S f(s_i)) = k *_S f^*(\sum k_i \otimes s_i)$.

This implies that for f^* to preserve $\sim$ it suffices to preserve (8,9,10,11), which is checked as follows.

- $f^*((k_1 +_K k_2) \otimes s) = (k_1 +_K k_2) *_S f(s)$
  $= k_1 *_S f(s) +_S k_2 *_S f(s) = f^*(k_1 \otimes s +_{K\otimes\mathcal{M}} k_2 \otimes s)$
- $f^*(0_K \otimes s) = 0_K *_S f(s) = \emptyset_+ = f^*(0_{K\otimes\mathcal{M}})$.
- $f^*(1_K \otimes (s_1 +_S s_2)) = 1_K *_S f(s_1 +_S s_2)$
  $= 1_K *_S (f(s_1) +_S f(s_2)) = 1_K *_S f(s_1) +_S 1_K *_S f(s_2)$
  $= f^*(1_K \otimes s_1 +_{K\otimes\mathcal{M}} 1_K \otimes s_2)$
- Since f is a semiring homomorphism, for any s we have $f(s) = f(s + 0_S) = f(s) +_S f(0_S)$, which yields that $f(0_S)$ must be $\emptyset_+$. Now, $f^*(1_K \otimes 0_S) = 1_K *_S f(0_S) = \emptyset_+ = f^*(0_{K\otimes\mathcal{M}})$.

Since f^* preserves $\sim$ it can be defined as above by picking a representative from each equivalence class. Now let $g : K \otimes S \to A$ be another K-semialgebra homomorphism such that $g \circ \iota = f$. Then

$$\begin{aligned} & g(\sum k_i \otimes s_i) = g(\sum k_i *_{K\otimes\mathcal{M}} (1_K \otimes s_i)) \\ = \quad & \sum k_i *_S g(\iota(s_i)) = \sum k_i *_S f(s_i) = f^*(\sum k_i \otimes s_i) \end{aligned}$$

hence $g = f^*$, thus verifying the uniqueness of f^*.
□