Probabilistic data structures applied to implicit graph representation

Juan P. A. Lopes[1]*, **Fabiano S. Oliveira**[1]†, **Paulo E. D. Pinto**[1]†

[1]IME/DICC, Universidade do Estado do Rio de Janeiro (UERJ)
Av. São Francisco Xavier, 524, Maracanã – 20550-013 – Rio de Janeiro – RJ

jlopes@cos.ufrj.br, fabiano.oliveira@ime.uerj.br, pauloedp@ime.uerj.br

Abstract. *In recent years, probabilistic data structures have been extensively employed to handle large volumes of streaming data in a timely fashion. However, their use in algorithms on giant graphs has been poorly explored. We introduce the concept of probabilistic implicit graph representation, which can represent large graphs using much less memory asymptotically by allowing adjacency test to have a constant probability of false positives or false negatives. This is an extension from the concept of implicit graph representation, comprehensively studied by Muller and Spinrad. Based on that, we also introduce two novel representations using probabilistic data structures. The first uses Bloom filters to represent general graphs with the same space complexity as the adjacency matrix (outperforming it however for sparse graphs). The other uses MinHash to represent trees with lower space complexity than any deterministic implicit representation. Furthermore, we prove some theoretical limitations for the latter approach.*

1. Introduction

The interest in probabilistic data structures has increased in the recent years. That is a direct consequence to the emergence of applications that deal with large volumes of streaming data. In those applications, it is often required the ability to answer queries quickly, which is infeasible to do by querying over stored data due to high latency. Nevertheless, such volumes do not generally fit into memory either. Probabilistic data structures offer a good compromise for many applications, allowing less memory and CPU usage at cost of decreased accuracy. In [Lopes 2017], we surveyed some of those data structures concerning the problems of testing membership of an item in a (large) set and that of determining the cardinality of (large) sets. We proposed the application of two data structures (Bloom filters and MinHash) to the graph representation problem [Spinrad 2003]. Our main contribution was a probabilistic representation for trees that has better space complexity than the optimal deterministic representation. That will be further elaborated in Section 2.

1.1. Bloom filter

Bloom filter is a data structure that represents a set S and allows testing elements for set membership with some probability of false positives, but no false negatives [Bloom 1970].

*Currently D.Sc. student at COPPE/UFRJ.

†Partially supported by FAPERJ.

Dissertation available at:
http://www.ime.uerj.br/~pauloedp/MEST/probabilistic_ds.pdf

It is implemented as an array M of m bits and k pairwise independent hash functions $h_i : S \to [1; m]$ for all $1 \leq i \leq k$. The insertion of an element x is performed by computing k hash values and setting these indexes in the array to 1, that is $M[h_i(x)] \leftarrow 1$ for all $1 \leq i \leq k$. The membership query for some element x is done by verifying whether all bits in positions given by the hash values are 1, that is $M[h_i(x)] = 1$ for all $1 \leq i \leq k$. If at least one bit is 0, that means with certainty that x is not in the set. If all bits are 1, it is assumed that the element is in the set, although that may not be the case (a false positive). The probability of a false positive when n elements are already stored (event FALSEP) can be determined from the probability of collisions in all k hash values, that is

$$\Pr[\text{FALSEP}] = \Pr\left[\bigwedge_{1 \leq i \leq k} M[h_i(x)] = 1\right] = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - \mathrm{e}^{-kn/m}\right)^k$$

Defining $q = m/n$, that is, q as the ratio between the size of M in bits and the number of stored elements, it is possible to show that the probability of false positives is minimized when $k = q \ln 2$. So, $\Pr[\text{FALSEP}] \approx (1 - e^{-\ln(2)})^{q \ln(2)} \approx (0.6185)^q$. That is, setting the dimension of M to 10 bits per element and using 7 hash functions, it is possible to estimate set membership with less than 1% of false positives.

Bloom filters are commonly used in database systems, both to avoid fetch of non-existing data, and to optimize communication costs in distributed joins. In summary, Bloom filters are useful in contexts where performance gain in negative queries make up for the cost of false positives.

1.2. MinHash

MinHash is a probabilistic data structure that represents sets A and B and allows estimating their Jaccard coefficient $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ [Broder 1997]. The estimation is done by computing a signature (a k-tuple of hash values) for each set S, using k pairwise independent hash functions $h_1, \ldots, h_k$. Each element in the signature is given by $h_i^{\min}(S) = \min\{h_i(x) : x \in S\}$ for all $1 \leq i \leq k$. The probability of two sets A and B having a common signature element can be shown to be equal to their Jaccard coefficient, that is $\Pr[h_i^{\min}(A) = h_i^{\min}(B)] = J(A, B)$, for all $1 \leq i \leq k$. Given two sets A, B, let X_i denote the Bernoulli random variable such that $X_i = 1$ if $h_i^{\min}(A) = h_i^{\min}(B)$, and $X_i = 0$ otherwise. The set $\{X_1, \ldots, X_k\}$ consists of an independent set of unbiased estimators for $J(A, B)$, in such a way that increasing k decreases the estimator variance. The error bounds for the estimation of $J(A, B)$ can be proved using the Chernoff inequalities. In special, to achieve an error factor of θ with a probability greater than $1 - \delta$, k should be defined such that $k \geq \frac{2+\theta}{\theta^2} \times \ln(2/\delta)$.

MinHash original motivation is still its most useful application: detect plagiarism. It is possible to evaluate the similarity of two documents only by comparing their MinHash signatures in constant time. It can also be used in conjunction with HyperLogLog [Flajolet et al. 2008] to estimate the cardinality of set intersection without having both sets in the same machine [Lopes et al. 2016a]. This was another contribution of this work.

2. Probabilistic implicit graph representations

An *implicit graph representation* is a vertex labelling scheme that allows testing the adjacency between any two vertices efficiently by just comparing their labels [Muller 1988, Kannan et al. 1992, Spinrad 2003]. More formally, given a graph class $\mathcal{C}$ with $2^{f(n)}$ graphs with n vertices, a representation is said to be *implicit* if

1. it is *space optimal*, that is, it requires $O(f(n))$ bits to represent graphs in $\mathcal{C}$;
2. it *distributes information evenly* among vertices, that is, each vertex is represented by a *label* using $O(f(n)/n)$ bits;
3. the *adjacency test is local*, that is, when testing the adjacency of any two vertices, only their labels are used in the process.

According to this definition, *adjacency matrix* is an implicit representation of the class containing all graphs, because there are $2^{\Theta(n^2)}$ graphs of n vertices and the adjacency matrix can represent them using $\Theta(n^2)$ bits. On the other hand, the *adjacency list* is not an implicit representation, because it requires $\Theta(m \log n)$ bits to represent the same graph class, which may need $\Theta(n^2 \log n)$ bits in the worst case (e.g. complete graphs). In contrast, adjacency list is space optimal to represent trees, as $O(m \log n) = O(n \log n)$ for trees and there are $2^{\Theta(n \log n)}$ trees of n vertices, but it is still not an implicit representation because it does not distribute information evenly: each tree vertex may use $\Theta(n \log n)$ bits to represent its adjacency in an adjacency list (e.g. center vertices of stars).

In [Lopes 2017], we explored the concept of *probabilistic implicit graph representations*, which extends the concept of implicit representations by relaxing one of the properties: the adjacency test is *probabilistic*, meaning that it has a constant probability of resulting in false negatives or false positives. A 0% chance of false positives and negatives implies an ordinary implicit representation. The main benefit of probabilistic representations is the ability to trade accuracy for memory, that is, to achieve more space efficient representations by allowing some incorrect results. We present two novel probabilistic implicit representations, each based on a distinct probabilistic data structure.

2.1. Representation based on Bloom filter

For this probabilistic implicit representation, Bloom filters are applied to represent the adjacency set. That is, for each vertex, a Bloom filter is created, representing the set of vertices adjacent to it. The set of Bloom filters of all vertices constitutes an implicit probabilistic representation. This representation requires $\Theta(\sum_{v \in V(G)} d(v) = 2m)$ bits to represent any graph, which makes it equivalent to the adjacency matrix in the worst case (e.g. complete graphs). However, this representation has better space complexity for sparse graphs than the deterministic one. In fact, it is better for any graph where $m = o(n^2)$. Also, it has the property of not allowing false negatives in adjacency tests. That is, it will never fail to report an existing edge, although it may report the existence of non-existing edges with a small probability.

2.2. Representation based on MinHash

We introduced a representation based on MinHash in which the main idea is, for any graph $G = (V, E)$ in a class $\mathcal{C}$ and for some pair of constants $0 \leq \delta_A < \delta_B \leq 1$, to find representing sets $S_v \neq \emptyset$ for each $v \in V$ such that the following two conditions hold: (i) $J(S_u, S_v) \geq \delta_B$ if and only if $(u, v) \in E$, and (ii) $J(S_u, S_v) \leq \delta_A$ if and only

if $(u, v) \notin E$. Therefore, no pairwise Jaccard coefficient among those representing sets should lie within the interval $(\delta_A; \delta_B)$. This way, the adjacency (u, v) can be tested by determining $J(S_u, S_v)$ and comparing it with δ_A and δ_B. However, we shall use MinHash to provide not the exact value, but an estimation of the Jaccard coefficients. Therefore, the actual idea to test adjacency is to assume that $(u, v) \in E$ if $J(S_u, S_v) > \delta$ for some $\delta_A \leq \delta \leq \delta_B$. Note that only the signatures of the representing sets must be stored and they require a constant number of elements. Furthermore, those signatures can be represented with a constant number of bits [Li and König 2010], and therefore a representation based on MinHash requires $O(n)$ bits to represent any class for which such representing sets exist.

In [Lopes 2017], we presented an algorithm to build such representing sets for trees with $\delta_A = 1/3$ and $\delta_B = 1/2$. Given a tree T, the construction is performed recursively starting at an arbitrary vertex v, to which S_v is defined with ℓ arbitrary distinct elements, where $\ell = \min\{2^r : r \in \mathbb{N} \mid 2\Delta(T) \leq 2^r\}$. Transforming T into a rooted tree having v as the root, for each level, the procedure alternates between choosing S_u as a subset of S_p (*selection* phase) and choosing S_u as a superset of S_p (*extension* phase), where p is the parent of u in T. Figure 1 exemplifies this construction.

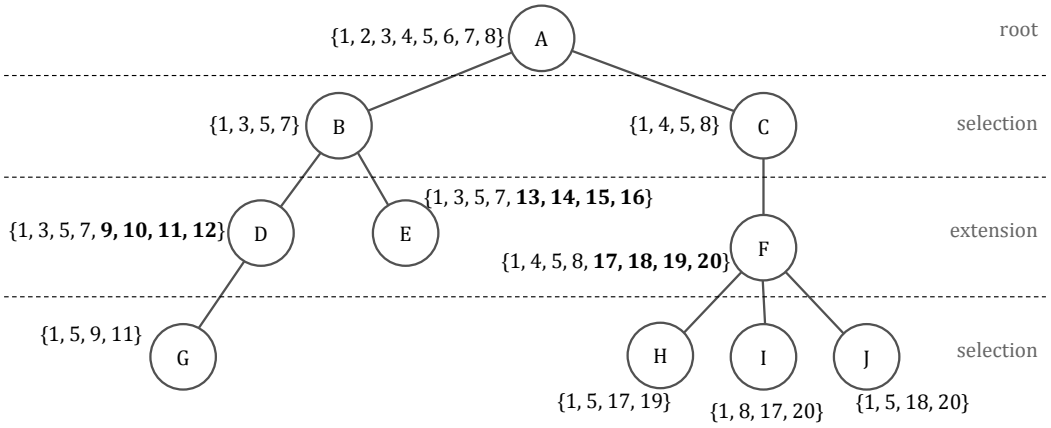

Figure 1. Example of representing sets for a given tree.

The selection phase is done as follows. For a set $S_p = \{a_1, \ldots, a_x\}$, $x/2$ subsets $U_1, \ldots, U_{x/2}$ are selected from it, each with $x/2$ elements, such that each pair of subsets has $x/4$ common elements. This way, $J(U_i, U_j) = 1/3$ for all $1 \leq i < j \leq x/2$ and $J(U_i, S_p) = 1/2$ for all $1 \leq i \leq x/2$. Thus, each child of p must be assigned a distinct subset among those as its representing set. The efficient implementation of such selection procedure is based on the representation by a binary string u_i, with length $x/2$, of a subset $U_i \subset S_p$, such that if the j^{th} bit of u_i has the value b, then a_{2j-1+b} belongs to U_i. The generation of the strings that represent $U_1, \ldots, U_{x/2}$ can be done through an iterative process, in which, starting from a 1×1 matrix, in each step, the matrix is fourfolded, inverting the bits of the lower right quadrant. The construction process, for a set $S_p = \{1, \ldots, 8\}$, is illustrated in Figure 2. The extension phase is done through the inclusion of unique $|S_p|$ elements among all already defined representing sets.

The MinHash signature is then computed for each representing set. Those signatures are then used as labels for each corresponding vertex. As this labelling scheme

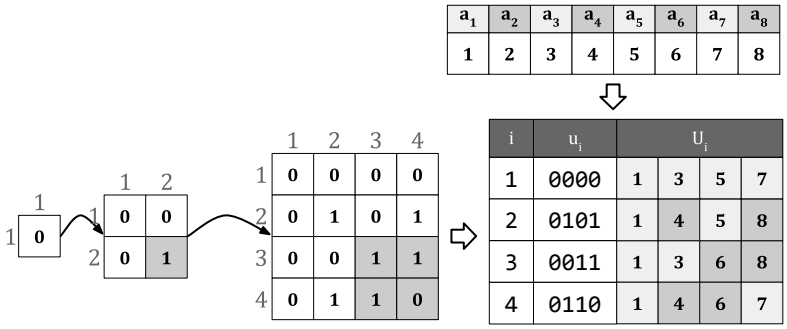

Figure 2. Example of a subset selection of $S_p = \{1, \ldots, 8\}$.

requires only $O(n)$ to probabilistically represent trees, a class with $2^{\Theta(n \log n)}$ graphs of n vertices, we can say that it has better space complexity than the optimal deterministic representation.

The theoretical predictions about this representation were verified through three practical experiments. The experiments aimed to validate the rate of false positives and negatives as some parameter changed. The results can be seen in the Figure 3.

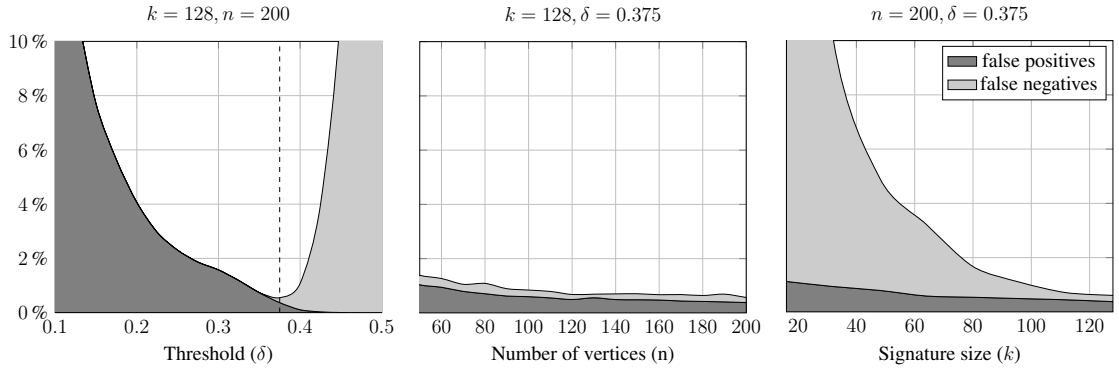

Figure 3. Rate of false negatives and false positives varying parameters.

2.3. Considerations on bipartite graphs

[Spinrad 2003] shows that any hereditary graph class with $2^{\Theta(n^2)}$ members of n vertices should entirely include either the bipartite, co-bipartite or split graphs. Also, it is possible to transform any graph $G = (V, E)$ into a bipartite graph $G' = (V', E')$ such that $V' = \{v_1, v_2 : v \in V\}$, and $E' = \{(v_1, u_2), (u_1, v_2) : (v, u) \in E\}$. Any efficient representation of G' can be used to efficiently represent G. This makes the search for a probabilistic implicit representation for bipartite graphs specially appealing. However, we proved the non-existence of some representations. For example, it is impossible to construct a MinHash-based representation with $\delta_A = 0.4$ and $\delta_B = 0.6$ for a graph as simple as a complete bipartite $K_{3,3}$. Our proof is based on the formulation of a corresponding integer linear programming problem, which turns out to be infeasible. This suggests that a further investigation concerning this probabilistic implicit graph representation is that of characterizing the class of graphs that can be represented by it.

3. Conclusion

This paper summarizes the contributions in [Lopes 2017]. A few probabilistic data structures were studied, and empirical evidence confirming their theoretical behavior was provided. Moreover, a probabilistic extension to the theory of implicit graph representations was introduced with a successful result regarding the representation of trees.

The results from this research were published in some conferences: 1. *Estruturas de Dados Probabilísticas para Representação de Conjuntos*, published at *I Encontro de Teoria da Computação* [Lopes et al. 2016b]; 2. *Estimativa de Cardinalidade da Interseção de Conjuntos Utilizando as Estruturas MinHash e HyperLogLog*, published at *XXXVI Congresso Nacional de Matemática Aplicada e Computacional* [Lopes et al. 2016a]; 3. *Representações Implícitas Probabilísticas de Grafos*, published at *II Encontro de Teoria da Computação* [Lopes et al. 2017]. A full report of the contributions is in the process of being submitted to a journal.

References

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426.

Broder, A. Z. (1997). On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29. IEEE.

Flajolet, P., Fusy, É., Gandouet, O., and Meunier, F. (2008). Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. *DMTCS Proceedings*.

Kannan, S., Naor, M., and Rudich, S. (1992). Implicit representation of graphs. *SIAM Journal on Discrete Mathematics*, 5(4):596–603.

Li, P. and König, A. C. (2010). b-bit minwise hashing. In *Nineteenth International World Wide Web Conference (WWW 2010)*. Association for Computing Machinery, Inc.

Lopes, J. P. A. (2017). Estruturas de dados probabilísticas aplicadas à representação implícita. Master's thesis, Universidade do Estado do Rio de Janeiro.

Lopes, J. P. A., Oliveira, F. S., and Pinto, P. E. D. (2016a). Estimativa de cardinalidade da interseção de conjuntos utilizando as estruturas minhash e hyperloglog. In *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, volume 5. SBMAC.

Lopes, J. P. A., Oliveira, F. S., and Pinto, P. E. D. (2016b). Estruturas de dados probabilísticas para representação de conjuntos. In *Anais do XXXVI Congresso da Sociedade Brasileira de Computação*. Sociedade Brasileira de Computação.

Lopes, J. P. A., Oliveira, F. S., and Pinto, P. E. D. (2017). Representações implícitas probabilísticas de grafos. In *Anais do XXXVII Congresso da Sociedade Brasileira de Computação*. Sociedade Brasileira de Computação.

Muller, J. H. (1988). *Local structure in graph classes*. PhD thesis, Georgia Institute of Technology.

Spinrad, J. P. (2003). *Efficient graph representations*. American mathematical society.