Scalable privacy-preserving record linkage: Evaluating MultiBit tree indexing in Atyimo

Victor Orrico [1], Fernanda Eustáquio[2], Bethânia Almeida[2]
Mirlei Silva[1], Robespierre Pita[1,2]

[1]Universidade Federal da Bahia
Instituto de Computação, Departamento de Ciência da Computação

[2]Instituto Gonçalo Moniz
Centro de Integração de Dados e Conhecimentos para Saúde (CIDACS)

orrico.victor@gmail.com, fernanda.eustaquio and bethania.almeida@fiocruz.br

mirlei.moura@ufba.br, robespierre.pita@ufba.br

Abstract. *Privacy-preserving record linkage (PPRL) indexing techniques typically organize Bloom Filters (BF) into data structures to reduce unnecessary comparisons. However, widely used solutions like Multibit Trees (MTB) often face scalability issues with large datasets or high-dimensional BFs, requiring parallel or distributed computation. This study explores the integration of the MTB algorithm into Atyimo, a publicly available Brazilian PPRL tool for merging large-scale administrative databases. We used both simulated and real-world data in our experiments to evaluate Atyimo's effectiveness with MTB in linking routinely collected health records in Brazil. The results show that our Spark DataFrame-based solution builds robust index structures that preserve the linkage accuracy and significantly reduce execution time compared to the baseline.*

1. Introduction

Record linkage (RL) is a computational methodology targeting finding records corresponding to the same entity in multiple sources [Christen 2019]. In big data integration, this step typically compares quasi-identifying attributes (QIDs) across records from different datasets using specific rules to identify matches [Dong and Srivastava 2013]. Unlike direct identifiers, quasi-identifiers (QIDs) — such as name, mother's name, gender, and the municipality of residence — might lead to missed matches, as they are not unique and may contain errors. However, protecting QIDs in administrative databases may be legally required to prevent re-identification [Schnell 2015], demanding strategies from privacy-preserving record linkage (PPRL).

PPRL methods enable data to be merged between two organizations frequently using the encoded QIDs. According to [Schnell 2015], they can either share the encoded records and computational infrastructure or resort to a trusted party. Anyway, the PPRL pipeline often involves six steps. The first step involves selecting variables uniquely identifying individuals in both databases, complying with the schema alignment [Dong and Srivastava 2013]. In the second stage, analysts apply various prepro-

cessing strategies to improve the data quality and address issues regarding semantic ambiguity. Specifically for PPRL, preprocessing the data often includes applying some encoding or encryption technique [Schnell 2015]. The most popular strategy is Bloom filters (BF) [Schnell et al. 2009]. It consists of breaking down the QIDs into bigrams and using an arbitrary number of hash functions to map those bigrams into a binary vector.

There are three fundamental approaches for encoding records into BF. The Attribute-level Bloom Filter (ABF) [Schnell et al. 2009] maps each QID value to a separated BF. Conversely, the Cryptographic Long-term Key (CLK) [Schnell et al. 2011] encodes the whole QID to the same BF, making the resulting BF more resistant to re-identification attacks. Suggested by [Durham et al. 2013], the Record-level Bloom Filter (RLB) improves the hardening of CLK by randomly selecting the bits from the original BF to form the final vector. Other parametrizations regard the size of BF and the strategies for combining different hash functions. Analysts may resort to Double, Triple, or Enhanced Hash protocols [Dillinger and Manolios 2004, Kirsch and Mitzenmacher 2006].

The four remaining steps of PPRL include indexing the databases to reduce the number of comparisons, performing pairwise scoring, classifying the pairs as matching or non-matching, and validating the classification. Addressing privacy protection in indexing techniques is a challenging issue and plays a relevant role in big data scenarios. The primary open demand is drastically decreasing the execution time, using fewer computational resources, and maintaining accuracy [Christen 2011]. In Brazil, the *Atyimo* [Pita et al. 2018b] is a publicly available tool suitable for linking large databases, being initially responsible for merging the first health data sets to the 100 Million Cohort [Barreto et al. 2022]. However, the current version of this Spark-based PPRL solution does not incorporate state-of-the-art indexing and still implements the outdated Spark-RDD library.

This work aims to investigate the impact of an innovative indexing technique, namely the Multibit Tree algorithm (MTB), on the accuracy and performance of RL when applied to integrating Brazillian administrative databases. Our main contribution comprises providing an updated version of Atyimo and implementing the MTB using the Spark Dataframes library. Additional developments include ABF, CLK, and RLB as alternatives to generate BFs and the possibility of using different hashing protocols. Our investigation methodology includes collecting specific metrics of indexing quality, such as pair completeness, pair quality, and reduction rate, to assess the MTB in a simulated scenario with controlled errors and real-world data. We also compared the accuracy of this new version with the previously published baseline. Results indicate that using MTB in Atyimo can improve the scalability of big data integration while maintaining quality, achieving similar accuracy compared to the baseline.

2. Related Work

The first MBT structure proposed for indexing BFs in PPRL was [Schnell 2014]. The experimental setup included two tasks in census databases encoded using CLK. The first task was finding the best matches in a pair of samples with controlled errors, ranging from 0 to 20% of the databases. The authors used F-measure to compare the quality of RL using MTB against Canopy Clustering (CC) and Sorted Neighborhood (SN). The second task involved evaluating the execution time of RL using the different strategies for

linking files with sizes varying between 100.000 and 1.000.000 records. Results show that MTB accomplished a more robust balance between accuracy and performance, achieving a similar execution time to SN, outperforming the CC, and keeping the linkage quality.

In [Christen et al. 2021], the authors also report that MTB outperforms other techniques, even in multiparty linkage scenarios. They highlight that MTB has a linear computational complexity due to the increasing number of databases involved in the linkage. Conversely, the results in [Vatsalan et al. 2017] indicate that execution time increases more than linearly for all PPRL indexing techniques evaluated. Their comparative evaluation shows that the pivot-based metric space approach outperforms the MTB, but it is not as robust when dealing with databases that contain more errors. In conclusion remarks, they argue that analysts must resort to parallel or distributed MTB to address the quadratic complexity problem.

Brazilian researchers have produced three PPRL solutions that implement indexing techniques to address dimensionality challenges: Tucuxi-BLAST, ABEL and Atyimo. Tucuxi-BLAST [Araujo et al. 2022] leverages the Basic local alignment search tool (BLAST) algorithm [Altschul et al. 1990] to achieve efficiency in similarity-based comparison of encoded data. The experimental evaluation included real-world routinely collected data from the Brazilian Health Ministry and simulated data sets. The privacy strategy lies in an encryption step that encodes the QIDs into DNA sequences using a pre-shared symmetric key. The Tucuxi-BLAST tool achieves competitive results of execution time, memory footprint and CPU usage. However, the authors do not present a deep discussion on the resistance to re-identification attacks or provide a scalable solution to handle databases larger than available memory.

The ABEL [Nóbrega et al. 2021] tool implements an iterative comparison over Splited Bloom Filters. This strategy intends to reduce data sharing in a blockchain infrastructure while avoiding unnecessary computations in the RL task. The authors report F1 measurements above 0.89 for several benchmarking data sets and have deeply discussed their capacity of providing privacy, resisting attacks [Nóbrega et al. 2022], and effectiveness. The main limitation for ABEL's wide adoption regards the complexity of building and maintaining a blockchain setup.

Recently, Ranbaduge et al. [Ranbaduge et al. 2023] proposed a deep learning-based PPRL protocol using differentially private Bloom filters and horizontal federated learning. Each data owner trains a local model on perturbed BFs, which are aggregated into a global classifier. The method achieved F1-scores above 0.90 on multiple real-world datasets, outperforming traditional PPRL techniques by 12–20% in linkage quality. It also proved robust to re-identification attacks, with privacy guarantees derived from differential privacy theory. However, the approach relies on training data availability and incurs higher computational costs due to deep learning and model aggregation steps.

The Atyimo [Pita et al. 2018b, Pinto et al. 2015] is a scalable RL solution that links the first health databases to the 100 million cohort [Barreto et al. 2022]. Their spark-based implementation enables the linkage of running on a commodity machine, a supercomputer, or a Spark cluster in a cloud computing environment. However, the original version includes a predicate-based blocking technique that increases RL's space and memory costs and is hard to manage [Pita et al. 2018a]. Additionally, the currently available code

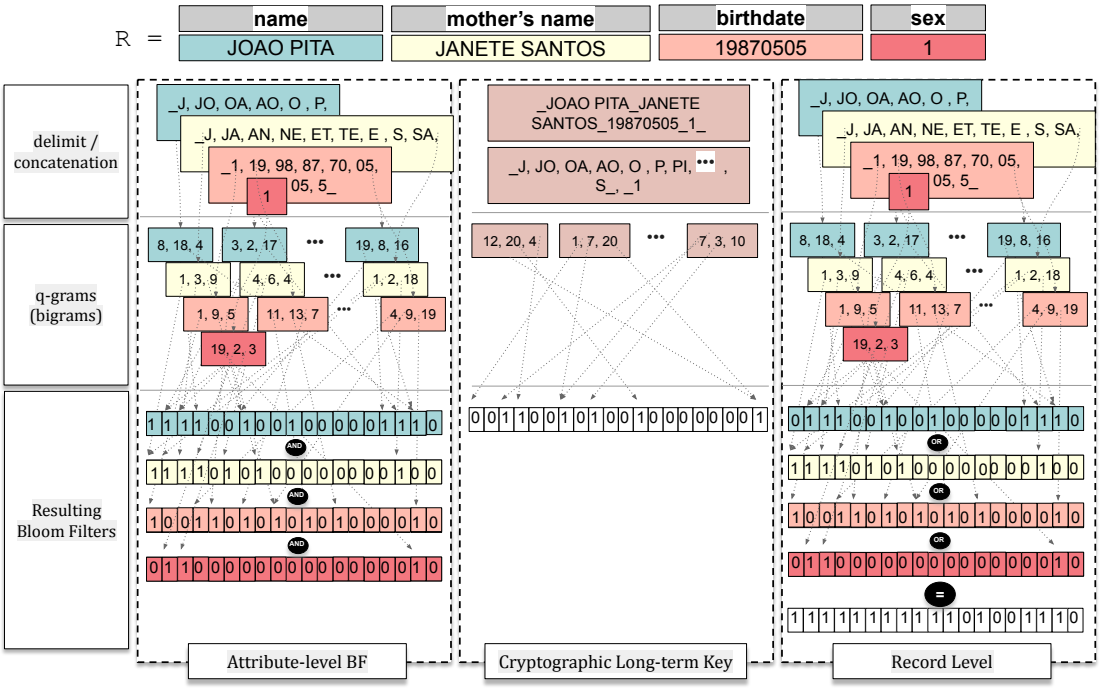


Figure 1. Examples of Bloom Filter Protocols over a q-gram decomposed record, adapted from [Pita et al. 2025]

does not leverage the Spark Dataframe library or any state-of-the-art indexing for PPRL. This work aims to address this gap, following the suggestion of [Vatsalan et al. 2017] and evaluating the performance and quality of the PPRL method using MTB in Brazilian administrative data.

3. Atyimo: The Brazillian PPRL Solution

The name Atyimo combines elements from two languages. In Tupi, *aty* means ”union”, and in Yoruba, *ìmò* means ”knowledge”. The word also resembles *átimo* in Portuguese, which refers to a brief moment, evoking the ideas of speed and precision that are central to the tool's function in record linkage.

As with CIDACS-RL [Barbosa et al. 2020], the version of Atyimo introduced in this paper is limited to the record linkage phase of big data integration, and does not include schema alignment or data fusion, as discussed in [Dong and Srivastava 2013]. The Atyimo implementation is structured into three core modules. The first handles record encoding, allowing users to parameterize the linkage process by selecting a consistent hashing strategy or Bloom Filter protocol. The second module integrates the construction of the MTB indexing structure with the pairwise comparison step, computing similarity scores for each candidate pair. The third module performs ranking of candidate pairs, identifying the most similar ones as matches.

Figure 1 illustrates how the encoding module works. Consider a record $R =$ [”JOAO PITA”, ”19870505”, ”1”, ”JANETE SANTOS”], which contains, respectively, the individual's name, date of birth, sex, and mother's name. Using the *Double Hashing* strategy, the encoding process follows three main steps. In the first step, each field is

decomposed into n-grams. For instance, the value "JOAO PITA" is initially transformed into the set $R_{\text{bigrams\_name}} = [\_J, JO, OA, AO, O\ ,\ P, PI, IT, TA, A\_]$ by applying padding and sliding a n-character window across the string. Then, each bigram in $R_{\text{bigrams\_name}}$ is mapped to multiple positions in the Bloom filter using the equation $\text{position}_i = (\text{int}(h_1(\text{bigram})) + i \cdot \text{int}(h_2(\text{bigram}))) \mod m$, where i is the index of the simulated hash function, ranging from 1 to k (the total number of hash functions). The functions h_1 and h_2 are two distinct hash functions (for example, MD5 and SHA256), and m is the size of the Bloom filter (i.e., the total number of bits). As comprehensively described in [Pita et al. 2025], the Record Level protocol may lead to better levels of privacy when combined with high-sized vectors.

After encoding the records from each database, the different parties involved in the record linkage process —whether individuals or institutions— can share their encrypted records without relying on a trusted third party, thereby potentially mitigating the risk of reidentification. The pairing module operates by comparing these encoded records, using the Sørensen-Dice similarity. This similarity metric is widely used in PPRL tasks [Christen et al. 2020], particularly for comparing Bloom Filters. It quantifies the overlap between two bit arrays by computing the ratio of shared 1-bits to the total number of 1-bits in both filters. Given two Bloom Filters B_1 and B_2, the similarity is defined as $\text{Sørensen-Dice}(B_1, B_2) = \frac{2 \cdot |B_1 \cap B_2|}{|B_1| + |B_2|}$, where $|B_1 \cap B_2|$ is the number of bits set to 1 in both filters, and $|B_1|$, $|B_2|$ denote the number of 1-bits in each filter individually. The resulting coefficient ranges from 0 (no similarity) to 1 (identical filters), enabling an efficient and privacy-preserving estimation of record similarity without access to original attribute values. However, computing this similarity for all possible record pairs across two large datasets is computationally expensive. To address this, efficient indexing techniques are required to reduce the number of comparisons by filtering out unlikely candidate pairs.

4. Multibit-trees

Kristensen et al. [Kristensen et al. 2010] originally proposed Multibit Trees (MTB) to enable efficient searches within large databases containing structural information about chemical molecules. Using MTB in the PPRL pipeline enables nearest neighbors to search in high-dimensional binary space [Schnell 2014]. As described in Algorithm 1 and illustrated in Figure 2, building an MTB structure starts with assigning all BF from the larger database to the root node. Consider a database $D_{N \times p}$ with $N = 8$ records and $p = 16$ bits regarding the size of each BF. Therefore, $D = \{$(1:01101111 01011010), (2:01111111 01011010), (3:00001111 11110101), (4:11110000 11110000), (5:11110111 01011010), (6:01110011 01011010), (7:10000111 11111000), (8:00111000 11110100)$\}$.

The first iteration determines the best *match-bit* by finding the position in all 16-bit BF with the more significant number of 1s. Choosing this bit appropriately will directly impact the balance of BF in the nodes, maximize the differentiation, and improve search efficiency [Schnell 2014]. The bit in sixth position is the best candidate for match-bit in the first level since it has 1s in 6 out of the 8 Bloom filters in D. Splitting D using the values bit 6 will result in $d_{l1,left} = \{(4 : 1111000011110000), (8 : 0011100011110100)\}$ and $d_{l1,right} = \{(1 : 0110111101011010), (2 : 0111111101011010), (3 : 0000111111110101), (5 : 1111011101011010), (6 : 0111001101011010), (7 : 1000011111111000)\}$. The remainder of iterations will do the same operation recursively until they hit a stopping condition, such as the maximum depth, usage of all positions as match-bit, or all leaf nodes

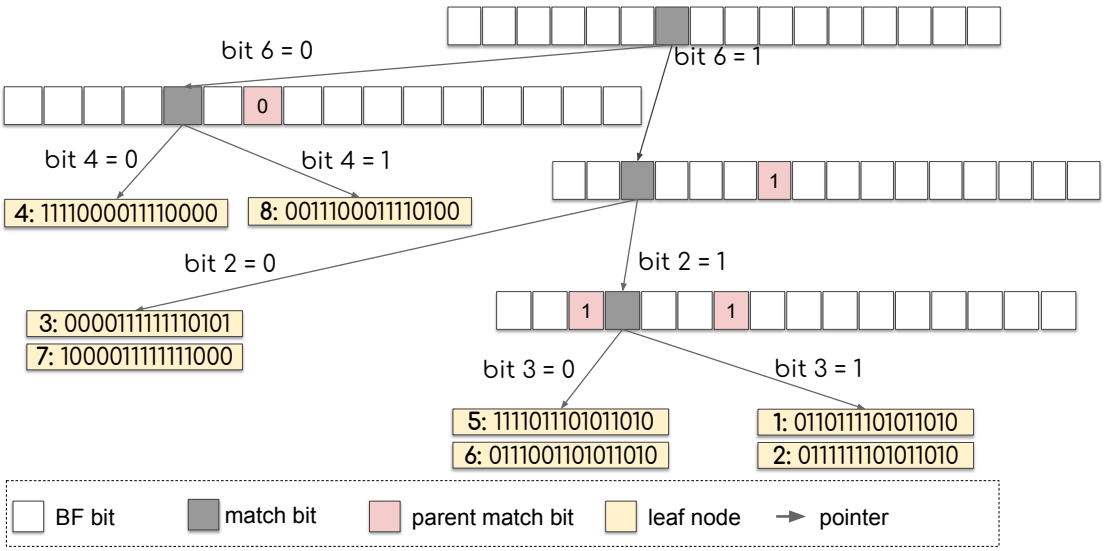


Figure 2. ***MultiBit Tree structure for the example described in Section 4***

holding the minimum amount of BF.

Performing a similarity search within the *MultiBit Tree (MTB)* structure involves a recursive overlap test using the match-bit at each level. For example, consider a search for the most similar Bloom filter (BF) to an encoded record $BF_{search} = 0111001101011110$. The process begins by testing the value of bit 6. This test determines the search direction within the structure, guiding the algorithm to the next bit to evaluate, in this case, bit 2. Since $bit2 = 1$, the MTB structure proceeds to test bit 3. With $bit3 = 0$ in BF_{search}, the search reaches a leaf node containing two candidate records $(5 : 1111011101011010)$ and $(6 : 0111001101011010)$. At this stage, a last-mile search is performed to rank the candidates based on their similarity to BF_{search}. This ranking is computed using the *Dice coefficient*, defined as $2h/(|a| + |b|)$, where h is the number of matching 1s between the two BFs, and $|a|$ and $|b|$ represent the count of 1s in BF_{search} and the candidate BF, respectively. As suggested by [Schnell et al. 2009, Christen et al. 2021], this similarity measure can be used to set a cutoff point, effectively filtering the number of candidates and refining the search results.

The implementation of MTB, depicted in Algorithm 1, in Spark Dataframes, leverages the column-based abstraction of the computing-intensive platform. We used native and user-defined functions to enable the execution of Atyimo in a scalable computational environment. The following sections will detail the main quality aspects and validity framework used to evaluate this methodological development.

The implementation of the Multibit Tree (MTB) [1] in Atyimo was restructured to operate efficiently within Spark's distributed DataFrame environment. To replace the recursive logic traditionally used in tree-based partitioning, we decomposed the MTB construction into a sequence of declarative transformations over DataFrames, leveraging Spark's columnar execution engine. Each node split is expressed as a filter operation on Bloom Filter bit positions, enabling distributed evaluation across partitions and mini-

[1] https://github.com/pierrepita/atyimo-2.0

Algorithm 1 MultiBit Tree

procedure CONSTRUCTION MULTIBIT TREE
  while $i \leq sizeBF$ **do** ▷ sizeBF is the number of bits in the BF
    $sum[i] \leftarrow \sum_{k=1}^{n} BF_i$ ▷ n is the number of BF in the node
    $result[i] \leftarrow sum[i] - \frac{sizeBF}{2}$
  $BitPosition \leftarrow \min(result)$
  while $k \leq n$ **do**
    if $BF_k[BitPosition]$ is 1 **then**
      $listRight.append(BF_k)$
    else
      $listLeft.append(BF_k)$
  if lenght of $listRight \leq 1$ **then return** $listRight$
  else
    $nodeRight \leftarrow Construction(listRight)$
  if lenght of $listLeft \leq 1$ **then return** $listLeft$
  else
    $nodeLeft \leftarrow Construction(listLeft)$

procedure SEARCH MULTIBIT TREE
  for each BF in the second database **do**
    if tree has attribute 'key' **then**
      if $BF[key] == 1$ **then return** $Search(BF, tree.rigth)$
      else
        return $Search(BF, tree.left)$
    else
      return $Similarity = dice(BF, node)$

mizing data shuffling. Bitwise operations are encapsulated within user-defined functions (UDFs), which compute Bloom Filter encodings and extract match-bit values without breaking Spark's parallelism. These UDFs operate on row-level data but integrate seamlessly with Spark's query optimizer, allowing the use of caching, predicate pushdown, and vectorized execution where applicable. This design ensures that both the tree construction and the candidate search stages scale linearly with data volume, supporting high-throughput linkage tasks without reliance on centralized memory or iterative driver-side control.

5. Experimental evaluation

We conducted two sets of experiments to evaluate the use of the Multibit tree in Atyimo to link Brazillian administrative data. The first set explores the quality of the MTB structure and the overall linkage accuracy. The second set of experiments examines the computational efficiency of the Atyimo using MTB over the Spark Dataframe library. Additionally, we compared this new version with prior results. The experimental setup involved an Intel Core i5 and 8GB of RAM. All the experiments were conducted using the 'spark-submit' command to a default standalone Apache Spark 3.5.4 installation.

As illustrated in Figure 3, the sets of experiments are directly related to each type of data set involved. This investigation also covers a comprehensive number of parameters and metrics regarding the effectiveness of the indexed structure and the linkage quality. Our evaluation includes comparing the new version of Atyimo with the legacy version presented in [Pita et al. 2018b] and competing alternatives as the baseline.

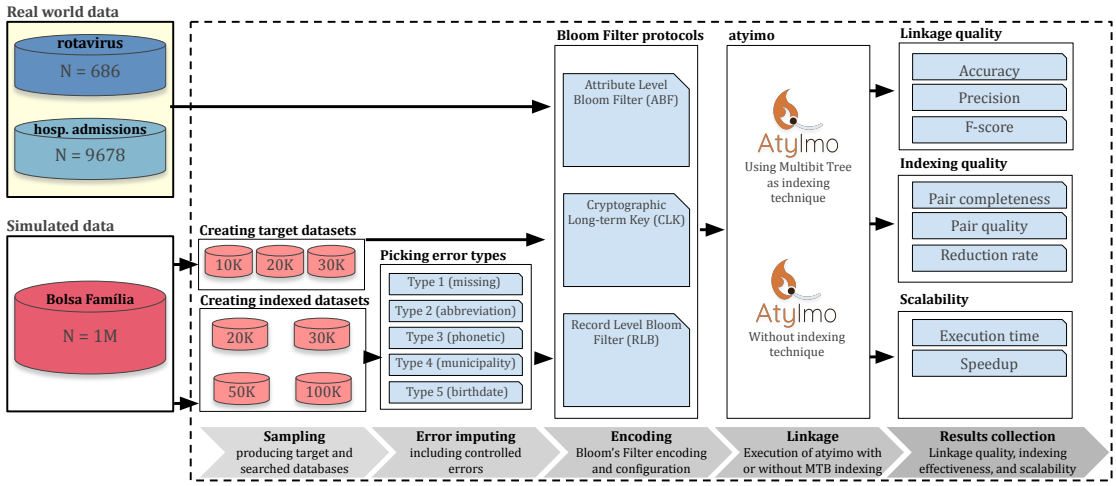


Figure 3. The experimental evaluation pipeline

5.1. Data sets

We used three data sets in our experimental evaluation. The first two files comprise a gold-standard pair of data sets used in [Pita et al. 2018b]. The first database holds 486 records of children treated for diarrhea with positive tests for rotavirus. The records in the second database describe 9678 hospital admissions with several causes, including diarrhea. The Centre for Data and Knowledge Integration for Health (CIDACS) [Barreto et al. 2019] provided restricted access to these databases in a secure environment accessed through a virtual private network.

The first step was building simulated data from the Bolsa Família Program (BFP) payroll records in 2024. We sampled one million beneficiaries from the BF dataset, described by the name, name of the family representative, and the code associated with the municipality of residence. The birthdate of each individual was added using the Synthetic Data Vault (SDV) [Patki 2016].

The third data set consists of a file with simulated records from the Bolsa Família Program (BFP) payroll records in 2024. We sampled one million beneficiaries from the BFP, described by the name, name of the family representative, and the code associated with the municipality of residence. The birthdate of each individual was added using the Synthetic Data Vault (SDV) [Patki 2016]. From the BFP simulated data, we obtained seven samples of different sizes, assigning four as indexed data sets and three as search data. Before the experiment, we generated 12 copies of the indexed data, each incorporating one of three error proportions described in Table 1. Five types of error, depicted in Table 1, are uniformely distributed into the proportion of records with noise.

5.2. Experiments and metrics

According to Table 1, we submitted 150 copies of the real-world pair of data sets to link with the original and new versions of Atyimo, resulting in 300 runs. These copies correspond to the product of five different BF sizes $(500, 1000, 1500, 2000, 2500)$, and five encoding techniques $(ABF, CLK, RLB_{30}, RLB_{40}, RLB_{50})$. The experiments also involved three hashing methods $(DoubleHash, TripleHash, EnhancedHash)$ and the number of hash functions $(2, 3)$.

Table 1. Experimental parameters and evaluation metrics

Experimental Parameter	Value	Source
— Experimental Parameters —		
Indexed dataset size	20K–100K (sim.), 9678 (real)	Dataset
Target dataset size	10K–30K (sim.), 686 (real)	Dataset
Bloom filter size (bits)	400–1600 / 500–2500	Parameter
Encoding method	ABF, CLK, RLB	Both
Sample rate for BF-RLB (%)	30 / 30–50	Sampling
Hashing method	Double / Triple / Enhanced	Method
# Hash functions	2 / 2–3	Method
Error types (%)	10, 25, 50 (Types 1–5)	Only Simulated
Metric	**Formula**	**Type**
— Evaluation Metrics —		
Accuracy	$\frac{TP+TN}{TP+FP+TN+FN}$	Linkage
Precision	$\frac{TP}{TP+FP}$	Linkage
F-score	$\frac{2 \cdot \text{Prec} \cdot \text{Rec}}{\text{Prec}+\text{Rec}}$	Linkage
Pair Completeness (PC)	$\frac{TP}{TP+FN}$	Indexing
Pair Quality (PQ)	$\frac{TP}{TP+FP+TN+FN}$	Indexing
Reduction Ratio (RR)	$\frac{TP+FP+TN+FN}{n \cdot m}$	Indexing

The second set of experiments involved 432 linkage runs for each Atyimo's version compared. This number of executions corresponds to the 36 pairs of simulated data, being 12 indexed with different sizes $(20K, 30K, 50K, 100K)$ and error proportions $(10\%, 25\%, 50\%)$, and the three target data sets with different sizes $(10K, 20K, 30K)$. We only included errors in the indexed files. In this set, the parametrization included three different bloom sizes $(400, 800, 1600)$ and three encoding techniques (ABF, CLK, RLB). Given the expressive number of runs, totalizing 864, the computational resource constraints influenced the number of parameters for this experiment set, reducing the alternatives to comply with our setup.

Table 1 shows the main metrics collected to compare the new Atyimo's version with previously published results and baseline. The accuracy consists of the proportion of well-classified pairs, providing a general overview of the linkage quality. Using precision and f-score alongside the accuracy metric may better support the findings and follow the best practices of the literature [Christen et al. 2021]. High precision often means that the RL could predict correctly the true matches, potentially ignoring false negatives. Alternatively, the f-score presents a harmonized mean within precision and recall.

Christen et al. (2011) [Christen 2011] was the first to present the indexing quality metrics in Table 1 for the RL context. The higher the pair quality (PQ) and pair completeness (PC), the more adequate the indexing technique was to preserve the proportion of true matches. The reduction ratio denotes the computational efficiency of the index structure used. Additionally, we collected the execution time of our runs and tried to analyze the impact of parametrizing the minimum number of records in an MTB leaf.

5.3. Results

Compared with the baseline in [Pita et al. 2018b], the new version of Atyimo maintains comparable results to existing PPRL solutions in the literature. Our proposal consistently

achieved the linkage quality results illustrated in Table 2 for the real-world data sets. The main noteworthy variation was related to execution time. Figure 4.d compares the average time, in seconds, to complete the linking process when the BF size varies. The plot indicates that using MTB can address the complexity boosted by the BF size, decreasing the time to 90%. Additionally, parametrizing the minimal number of records in each leaf node does not affect performance. However, reducing the BF size may substantially increase the number of false positives.

Table 2. Comparative analysis of our proposal when compared to baseline reported in [Pita et al. 2018b]

RL tool	TP	FP	FN	Accuracy	Precision	F-measure
FRIL [Jurczyk et al. 2008]	486	1	0	0.998	0.998	0.999
FRIL with blocking [Jurczyk et al. 2008]	484	0	2	0.996	1.000	0.998
Febrl [Christen 2008]	480	1	6	0.988	0.998	0.993
Febrl with blocking [Christen 2008]	479	0	7	0.986	1.000	0.993
Legacy Atyimo (no blocking)	486	0	0	1.000	1.000	1.000
Legacy Atyimo (blocking)	486	0	0	1.000	1.000	1.000
New Atyimo (no MTB)	486	0	0	1.000	1.000	1.000
New Atyimo (with MTB)	486	0	0	1.000	1.000	1.000

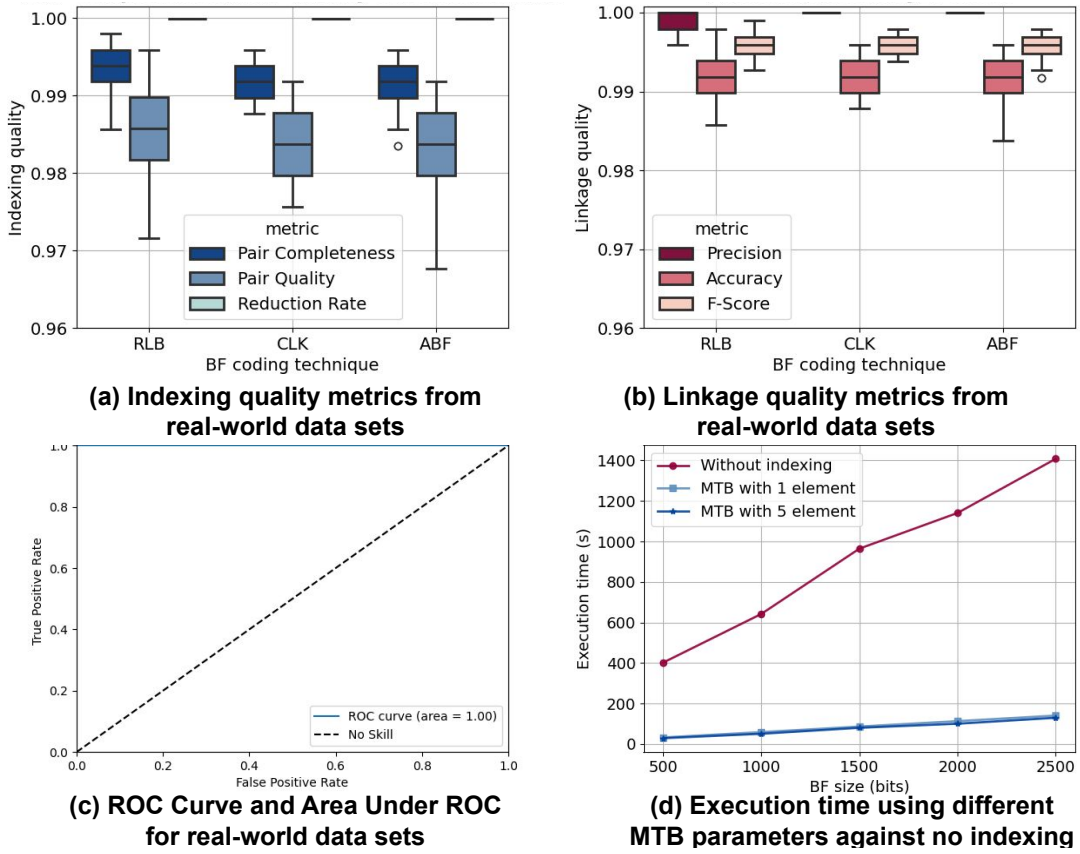
As depicted in Figures 4.a and 4.b, using MTB, the Atyimo achieves a drastic comparison reduction, maintaining satisfactory linkage quality results. We observed a slight variation in precision when using RLB as the BF encoding technique, mostly associated with smaller-sized Bloom filters. Finally, Figure 4.c exhibits an ROC curve for $BFsize = 500$, two hash functions, ABF strategy, and DoubleHasing. The Atyimo with MTB consistently accomplished this result in the majority of runs.

The second experiment set comprises a more comprehensive number of tests, enabling a thorough examination of Atyimo with MTB indexing. We evaluated the sensitivity of the MTB structure due to different proportions of controlled errors. Figure 5.a demonstrates the PC, i.e., the capacity of indexing structure to preserve accuracy, for each type of imputed error from Table 1. As shown in Figure 5.a, the more discriminative the QIDs, the more affected the PC metric for errors in different sizes in the indexed data sets, especially when missing values.

For the results in Figures 5.b, 5.c, and 5.d consider the five error types from Table 1 in 25% of the databases. In Figure 5.b, the x-axis indicates the multiplication factor of the BF size, indexed data set, and target data set. Initial values are 400, 50.000, and 10.000, respectively. Doubling these values, we found the BF size and the number of records in the indexed file influence the increasing execution time (in minutes) of Atyimo with MTB. Similar to the results in real-world data, Figures 5.c and 5.d show that the MTB structure can maintain the RR, leading to better performance and still achieve promising linkage quality metrics. BF encoding techniques do not impact the overall results, indicating that using any of them will only depend on RL's privacy requirements.

6. Discussion

Our work demonstrates that Multibit tree (MTB) structures can scale up Privacy-Preserving Record Linkage (PPRL) tasks for merging administrative Brazillian databases, leading to



(a) Indexing quality metrics from real-world data sets

(b) Linkage quality metrics from real-world data sets

(c) ROC Curve and Area Under ROC for real-world data sets

(d) Execution time using different MTB parameters against no indexing

Figure 4. Results from the first set of experiments: indexing and linkage quality in real-world data sets

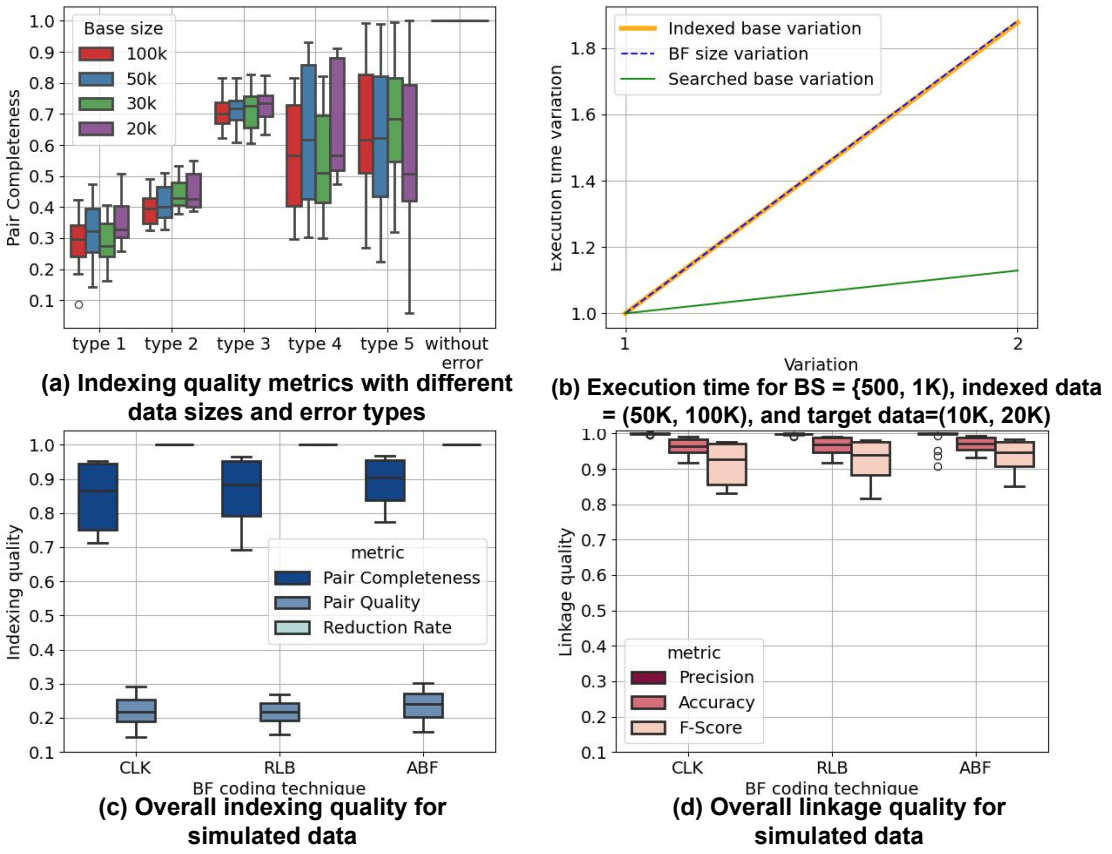
promising results in accuracy. Embedding MTB in an existing Brazillian PPRL, namely Atyimo, may enable several public health studies involving integrated data from two or more organizations without sharing quasi-identifying attributes (QIDs). Compared to Atyimo's legacy version and other well-known tools, the new Atyimo version shows optimistic performance, producing indexing structures capable of addressing lower levels of data quality.

We argue that leveraging a Spark Dataframe-based PPRL implementation will address the reported complexity of existing indexing solutions [Vatsalan et al. 2017]. Using Atyimo with MTB may support several big data analyses in a commodity computer, on-premises cluster, or cloud computing environment.

7. Limitations and future work

The lack of massive real-world data sets with prior knowledge of the link status, namely gold-standard databases, was the main limitation of this work. Even a more comprehensive experiment set using high-fidelity simulated records may not fully capture the complexity of routinely collected administrative data. Additionally, the unavailability of robust computational infrastructure affected the extension of our experiment set, leading to a limited investigation of performance in a scalable environment.

Other limitations of this investigation were related to the unavailability of more recent PPRL software that uses BF, preventing us from extending the compared baseline.



(a) Indexing quality metrics with different data sizes and error types

(b) Execution time for BS = {500, 1K), indexed data = (50K, 100K), and target data=(10K, 20K)

(c) Overall indexing quality for simulated data

(d) Overall linkage quality for simulated data

Figure 5. Results from the first set of experiments: indexing and linkage quality in simulated data sets

As mentioned in [Pita et al. 2018b], there is a lack of alternatives capable of running massive amounts of comparisons, restricting the performance evaluation and comparison.

In future work, we intend to embed new indexing features in Atyimo and make a graphical user interface available to ensure broader adoption. On the other hand, there is a substantial gap between the existing indexing solutions applied in PPRL and the state-of-the-art in the data management community. We intend to explore the potential of indexing algorithms from novel data models, such as document and vector-based systems. Finally, machine learning-enabled solutions, namely learned index structures, can represent an interesting research venue to explore in PPRL applications.

8. Conclusion

In this work, we assessed the potential of a Spark Dataframe-based implementation of Multibit Tree in PPRL tasks. This development has the potential to enable several complex analytical tasks over integrated databases, supporting public health research and policymaking. While our contributions and evaluation primarily focus on Brazilian administrative databases, due to the characteristics of our information systems and population, Atyimo's data integration pipeline can also link databases from different countries. With the maturity of the data protection process in organizations, active research on PPRL can provide updated solutions and sustainability for knowledge production on populational data centers.

References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410.

Araujo, J. D., Santos-e Silva, J. C., Costa-Martins, A. G., Sampaio, V., de Castro, D. B., de Souza, R. F., Giddaluru, J., Ramos, P. I. P., Pita, R., Barreto, M. L., et al. (2022). Tucuxi-blast: Enabling fast and accurate record linkage of large-scale health-related administrative databases through a dna-encoded approach. *PeerJ*, 10:e13507.

Barbosa, G. C. G., Ali, M., Barreto, M., Araujo, B., Reis, S., Sena, S., Ichihara, Y., Pescarini, J., Fiaccone, R., Amorim, L., Pita, R., Smeeth, L., and Barreto, M. (2020). CIDACS-RL: A novel indexing search and scoring-based record linkage system for huge datasets with high accuracy and scalability. *BMC Medical Informatics and Decision Making*, 20.

Barreto, M. L., Ichihara, M., Almeida, B. d. A., Barreto, M., Cabral, L., Fiaccone, R., Carreiro, R., Teles, C., Pitta, R., Penna, G., et al. (2019). The centre for data and knowledge integration for health (cidacs): linking health and social data in brazil. *International journal of population data science*, 4(2):1140.

Barreto, M. L., Ichihara, M. Y., Pescarini, J. M., Ali, M. S., Borges, G. L., Fiaccone, R. L., Ribeiro-Silva, R. d. C., Teles, C. A., Almeida, D., Sena, S., et al. (2022). Cohort profile: the 100 million brazilian cohort. *International journal of epidemiology*, 51(2):e27–e38.

Christen, P. (2008). Febrl- an open source data cleaning, deduplication and record linkage system with a graphical user interface. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1065–1068.

Christen, P. (2011). A survey of indexing techniques for scalable record linkage and deduplication. *IEEE transactions on knowledge and data engineering*, 24(9):1537–1555.

Christen, P. (2019). Data linkage: The big picture. *Harvard Data Science Review*, 1.

Christen, P., Ranbaduge, T., and Schnell, R. (2020). *Linking Sensitive Data: Methods and Techniques for Practical Privacy-Preserving Information Sharing*. Springer Cham.

Christen, P., Ranbaduge, T., and Schnell, R. (2021). Linking sensitive data: Methods and techniques for practical privacy-preserving information sharing: Synopsis by kerina jones. *International Journal of Population Data Science*, 6(2).

Dillinger, P. C. and Manolios, P. (2004). Fast and accurate bitstate verification for spin. In Graf, S. and Mounier, L., editors, *Model Checking Software*, pages 57–75, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dong, X. L. and Srivastava, D. (2013). Big data integration. In *Proceedings - International Conference on Data Engineering*, pages 1245–1248.

Durham, E. A., Kantarcioglu, M., Xue, Y., Toth, C., Kuzu, M., and Malin, B. (2013). Composite bloom filters for secure record linkage. *IEEE transactions on knowledge and data engineering*, 26(12):2956–2968.

Jurczyk, P., Lu, J. J., Xiong, L., Cragan, J. D., and Correa, A. (2008). Fril: a tool for comparative record linkage. In *AMIA annual symposium proceedings*, volume 2008, page 440.

Kirsch, A. and Mitzenmacher, M. (2006). Less hashing, same performance: Building a better bloom filter. volume 4168, pages 456–467.

Kristensen, T. G., Nielsen, J., and Pedersen, C. N. (2010). A tree-based method for the rapid screening of chemical fingerprints. *Algorithms for Molecular Biology*, 5:1–10.

Nóbrega, T., Pires, C. E. S., and Nascimento, D. C. (2021). Blockchain-based privacy-preserving record linkage: enhancing data privacy in an untrusted environment. *Information Systems*, 102:101826.

Nóbrega, T., Pires, C. E. S., and Nascimento, D. C. (2022). Explanation and answers to critiques on: Blockchain-based privacy-preserving record linkage. *Information systems*, 108:101935.

Patki, N. (2016). *The synthetic data vault: generative modeling for relational databases*. PhD thesis, Massachusetts Institute of Technology.

Pinto, C., Pita, R., Melo, P., Sena, S., and Barreto, M. (2015). Correlaçao probabilıstica de bancos de dados governamentais. *Simpósio Brasileiro de Bancos de Dados (SBBD 2015)*, pages 77–85.

Pita, R., Carreiro, R. P., Santos, C. J. C., Protasio, L. d. S., Barreto, M. E., Orrico, V. B., Gomes, J. A. D., Eustáquio, F. S., Sena, S., Barreto, M. L., Ramos, P. I. P., Rangel, D., and Almeida, B. d. A. (2025). Big data linkage no brasil: Aspectos metodológicos e práticos. In *Minicursos do XXV Simpósio Brasileiro de Computação Aplicada à Saúde*, pages 306–345. SBC.

Pita, R., Menezes, L., and Barreto, M. E. (2018a). Applying term frequency-based indexing to improve scalability and accuracy of probabilistic data linkage. In *LADaS@ VLDB*, pages 65–72.

Pita, R., Pinto, C., Sena, S., Fiaccone, R., Amorim, L., Reis, S., Barreto, M. L., Denaxas, S., and Barreto, M. E. (2018b). On the accuracy and scalability of probabilistic data linkage over the brazilian 114 million cohort. *IEEE Journal of Biomedical and Health Informatics*, 22(2):346–353.

Ranbaduge, T., Vatsalan, D., and Ding, M. (2023). Privacy-preserving deep learning based record linkage. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6839–6850.

Schnell, R. (2014). An efficient privacy-preserving record linkage technique for administrative data and censuses. *Statistical journal of the IAOS*, 30:263–270.

Schnell, R. (2015). Privacy-preserving record linkage. *Methodological developments in data linkage*, pages 201–225.

Schnell, R., Bachteler, T., and Reiher, J. (2009). Development of a new method for privacy-preserving record linkage allowing for errors in identifiers. *methods, data, analyses*, 3(2):15.

Schnell, R., Bachteler, T., and Reiher, J. (2011). A novel error-tolerant anonymous linking code. *Available at SSRN 3549247*.

Vatsalan, D., Sehili, Z., Christen, P., and Rahm, E. (2017). Privacy-preserving record linkage for big data: Current approaches and research challenges. *Handbook of big data technologies*, pages 851–895.