UNIVERSIDADE FEDERAL DA BAHIA

DISSERTAÇÃO DE MESTRADO

APLICAÇÃO DE TÉCNICAS DE ETL PARA A INTEGRAÇÃO DE DADOS COM ÊNFASE EM BIG DATA NA ÁREA DE SAÚDE PÚBLICA

CLÍCIA DOS SANTOS PINTO

Mestrado Multiinstitucional em Ciência da Computaçao

Salvador
05 de março de 2015

MMCC-Msc-2015

CLÍCIA DOS SANTOS PINTO

APLICAÇÃO DE TÉCNICAS DE ETL PARA A INTEGRAÇÃO DE DADOS COM ÊNFASE EM BIG DATA NA ÁREA DE SAÚDE PÚBLICA

Esta Dissertação de Mestrado foi apresentada ao Mestrado Multiinstitucional em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: MARCOS ENNES BARRETO

Salvador
05 de março de 2015

Ficha catalográfica.

Clícia dos Santos Pinto

APLICAÇÃO DE TÉCNICAS DE ETL PARA A INTEGRAÇÃO DE DADOS COM ÊNFASE EM BIG DATA NA ÁREA DE SAÚDE PÚBLICA/ CLÍCIA DOS SANTOS PINTO– Salvador, 05 de março de 2015.

55p.: il.

Orientador: MARCOS ENNES BARRETO.
Dissertação (mestrado)– UNIVERSIDADE FEDERAL DA BAHIA, INSTITUTO DE MATEMÁTICA, 05 de março de 2015.

TÓPICOS PARA FICHA CATALOGRÁFICA.
I. BARRETO, Marcos E. II. UNIVERSIDADE FEDERAL DA BAHIA. INSTITUTO DE MATEMÁTICA. III Título.

NUMERO CDD

TERMO DE APROVAÇÃO

CLÍCIA DOS SANTOS PINTO

APLICAÇÃO DE TÉCNICAS DE ETL PARA A INTEGRAÇÃO DE DADOS COM ÊNFASE EM BIG DATA NA ÁREA DE SAÚDE PÚBLICA

Esta Dissertação de Mestrado foi julgada adequada à obtenção do título de Mestre em Ciência da Computação e aprovada em sua forma final pelo Mestrado Multiinstitucional em Ciência da Computação da Universidade Federal da Bahia.

Salvador, 05 de março de 2015

Prof. Dr. Carlos Antônio de Souza Teles Santos
UEFS

Prof. Dr. Murilo do Carmo Boratto
UNEB

Prof. Dr. Frederico Araujo Durão
DCC/UFBA

À minha irmã, por todas as vezes que precisei do seu apoio e por todas as vezes que se colocou à disposição antes mesmo que fosse necessário.

AGRADECIMENTOS

Agradeço ao Professor Marcos Barreto, pelo seu empenho no desenvolvimento desta pesquisa e pelo direcionamento em todas as etapas deste mestrado. Ao Professor Davide Rasella, por todo apoio dispensado, pelo tempo dedicado, pelos ensinamentos e pela oportunidade de trabalhar em uma pesquisa com implicações tão relevantes no cenário social brasileiro. Ao Robespierre Dantas, pela sua dedicação na parceria deste trabalho e por tudo que pude aprender com a nossa convivência diária. Aos colegas do Laboratório de Sistemas Distribuídos pela assistência, experiências compartilhadas e esforço dedicado na produção dos textos científicos.

Aos amigos de sempre Adryelle Lomes, Aryanne Gastino, Drielle Oliveira, Matheus Bragança e aos amigos que encontrei na UFBA, muito obrigada por me lembrar que nenhum homem é uma ilha. À Rafaela, pelo companheirismo e por estar ao meu lado desde o início desta caminhada. Também agradeço à Jhenifer e à Tamires, pela nossa conviência. Vocês três tornaram essa jornada muito mais fácil.

À minha mãe Edneuza, pelo seu incentivo e por me oferecer tudo que eu precisei para me dedicar integralmente a este trabalho. Não há como expressar minha gratidão em palavras. Ao meu pai Cláudio e a toda minha família, por todo suporte que tive ao longo destes dois anos. Me orgulho de ser hoje, um pouco de cada um de vocês. Ao Bruno, meu querido companheiro, por todas as vezes que abreviou a distância e se fez presente. Você é meu maior exemplo de empenho, esforço e dedicação. Parafraseio Fernando Pessoa, na certeza de que tudo valeu a pena: *Quem quer passar além do Bojador, tem que passar além da dor.*

A maior riqueza do homem é sua incompletude. Nesse ponto sou abastado.

—MANOEL DE BARROS

RESUMO

Transformar os dados armazenados em informações úteis tem sido um desafio cada vez maior e mais complexo a medida em que o volume de dados produzidos todos os dias aumenta. Nos últimos anos, conceitos e tecnologias de Big Data têm sido amplamente utilizados como solução para o gerenciamento de grandes quantidades de dados em diferentes domínios. A proposta deste trabalho diz respeito à utilização de técnicas de ETL (extração, transformação e carga) no desenvolvimento de um módulo de pré-processamento para o pareamento probabilístico de registros em bases de dados na área de Saúde Pública. A utilização da ferramenta de processamento distribuído do *Spark* garante o tratamento adequado para o contexto de Big Data em que esta pesquisa está inserida, gerando respostas em tempo hábil.

Palavras-chave: Big Data, ETL, pré-processamento, correlação de registros, Spark

ABSTRACT

Transforming stored data into useful information has been a growing challenge as the volume of data produced daily increases. In recent years , Big Data concepts and technologies have been widely used as a solution for managing large amounts of data in different domains. The purpose of this work concerns the use of ETL (Extract, Transform and Load) techniques in developing a efficient pre-processing module for probabilistic record linkage of public health databases. The use of Spark high-performance processing tool guarantees the proper treatment to the context of Big Data in which this research is inserted, generating responses in a timely manner.

Keywords: Big Data, ETL, pre-processing, record linkage, Spark

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

INTRODUÇÃO

1.1 INTRODUÇÃO

A gestão da informação vem se destacando como necessidade fundamental em diversos setores organizacionais onde o grande desafio é transformar dados armazenados em informações práticas e acessíveis. A integração de dados é um problema comum em áreas de negócio em que se precisa correlacionar registros em diversas bases de dados, muitas vezes provenientes de diferentes fontes. A consolidação de dados de diferentes fontes é um requisito ligado aos projetos de *Data Warehouse* (SINGH, 1999) que permite organizar os dados coorporativos de maneira integrada, mantendo uma única versão e gerando uma única fonte de dados que será usada para abastecer os repositórios de dados chamados *Data Marts*.

Analisar grande quantidade de dados, muitas vezes, não é uma tarefa trivial e vem se tornando uma prática cada vez mais desafiadora à medida que se produz mais informações. As técnicas de extração, transformação, armazenamento, processamento, recuperação e distribuição precisam ser cada vez mais acuradas principalmente quando estão relacionadas com a manipulação de uma enorme quantidade de dados. Muitas vezes os dados que se quer analisar são provenientes de fontes diferentes, com estruturas e formatos diversos. Por isto, observa-se atualmente um grande esforço em diversas áreas da tecnologia da informação no sentido de se prover arquiteturas capazes de resolver estes problemas de forma eficiente.

Implementações voltadas para suporte a Big Data vêm sendo amplamente discutidas no contexto de saúde pública. Em termos gerais, muito se tem falado sobre estratégias de processamento intensivo de dados com o objetivo de se melhorar a prevenção e tratamento de doenças. O Brasil tem avançado significativamente nos últimos anos na gestão da informação relacionada à área de saúde pública. O DATASUS (DATASUS, 2015), por exemplo, é um esforço do governo brasileiro que busca coletar, processar e disseminar informações que podem ser úteis para análise de situações sanitárias, tomadas de decisão baseadas em evidência e elaboração de programas de ações de saúde.

No domínio de setores públicos de saúde existe uma crescente necessidade na integração de dados utilizando técnicas de ETL (*Extraction, Transformation and Loading*) (DOAN; HALEVY; ZACHARY, 2012), a fim de se extrair das enormes bases de dados existentes, informações úteis. Avanços nesta área podem representar um benefício tanto do ponto de vista operacional, no desenvolvimento das tecnologias compatíveis com os conceitos de *Business Intelligence* (GARTNER, 2015), gerando respostas rápidas para as tomadas de decisão; como também, em consequência, vários benefícios do ponto de vista político-social.

A utilização de métodos de relacionamento de bases de dados com a finalidade de recuperar informações de uma mesma entidade (uma pessoa, por exemplo), é um campo de vasta pesquisa em diversos domínios. A não existência de uma identificação única, capaz de tornar determinístico o relacionamento entre as bases, tem motivado a pesquisa sobre metodologias capazes de fazer uso de diversos atributos como chaves de comparação e estabelecer uma probabilidade de correlação entre os registros.

Atualmente, existe um consenso sobre a influência que a pobreza exerce na saúde. É fato que as populações mais pobres estão mais sucetíveis a doenças infecciosas como HIV/AIDS, malária, tuberculose, hanseníase, infecções parasitárias e outros. Nos últimos anos, diversos programas de transferência de renda foram implementados como um esforço no sentido de se reduzir os níveis de pobreza, como é o caso do Programa Bolsa Família (PBF, 2015), mas até então nenhum estudo avaliou de fato o impacto de tais programas sobre a morbimortalidade por tuberculose e hanseníase. O caso de uso vinculado à esta pesquisa tem como objetivo identificar e, caso exista, avaliar o impacto do programa de transferência de renda do Bolsa Família na redução da morbimortalidade por estas doenças.

Ao relacionar bases de dados das folhas de pagamento do Bolsa Família e dos dados socioeconômicos do Cadastro Único com os dados de morbimortalidade dos indivíduos, é possível não só avaliar a relação existente e o real impacto, como também concluir seus determinantes sociais. Para prover a integração de todas as bases de dados envolvidas utiliza-se uma metodologia de correlação que compara os registros de diferentes bases par a par e avalia se estes se referem ou não a uma mesma entidade. Considerando que a etapa de comparações par a par é a que exige maior demanda computacional, faz-se necessário utilizar métodos que minimizem a sobrecarga desta fase, evitando comparações desnecessárias. Para isto, existem algumas abordagens que utilizam blocagem para agrupar os registros de acordo com um ou vários critérios de similaridade. Associado à blocagem existem outros métodos cujos objetivos são aumentar a acurácia, facilitar as comparações e prover anonimização para as informações nominais. Estas etapas e outras, também relacionadas ao processo de Extração, Transformação e Carga e ao projeto de *Data Warehouse*, buscam sistematizar e organizar as informações provenientes de bases de dados de fontes diferentes, auxiliando as tomadas de decisão e fases subsequentes.

1.2 MOTIVAÇÃO

Os setores de vigilância epidemiológica e saúde pública no Brasil têm experimentado um aumento significativo na quantidade de informações armazenadas. Ao longo da história

foram criadas várias bases de dados de diferentes indicadores como é o caso, por exemplo do sistema de internações hospitalares e dos fatores de mortalidade da população. As inúmeras bases de dados que atendem às demandas desses setores contém registros desvinculados e independentes, tornando extremamente difícil, por exemplo, identificar uma mesma pessoa em bases de dados de diferentes fontes.

A grande dificuldade no relacionamento entre estas bases de dados se deve principalmente a não existência de um identificador único para registros de saúde. Alguns esforços já estão sendo movidos no sentido de se criar uma identificação nacional única e obrigatória para relacionar os sistemas de saúde com o cidadão, como é o caso do Cartão SUS, que busca integrar, modernizar e facilitar a comunicação entre os sistemas públicos de saúde. Entretanto, esta ainda não é uma realidade e as dificuldades na integração da informação são ainda um desafio. Em adição a isto, as políticas públicas de saúde e os grupos de pesquisa têm uma real necessidade em transpor tais impedimentos no sentido de obter as informações necessárias em tempo hábil e com resultados acurados.

A dificuldade no relacionamento entre estas bases de dados aumenta quando consideramos o tamanho dos sistemas reais. O CadÚnico (MDS, 2014) por exemplo contém mais de 100 milhões de registros em sua versão mais atual. Portanto, encontrar uma mesma entidade em bases desta dimensão, sem uma chave que permita uma busca determinística, implica em uma enorme quantidade de comparações.

Muito se têm discutido sobre métodos e estratégias para a correlação de registros em bases de diferentes esquemas em que não exista uma chave de relacionamento. Entretanto uma etapa igualmente importante está na preparação dos dados e na infraestrutura de integração. Ao projeto de *Data Warehouse*, análise dos dados, extração, transformação e carga é preciso dedicar atenção especial para garantir a qualidade do processo e validar os resultados obtidos.

O processamento e transferência de dados em escala tão grande quanto esta, representa um enorme desafio para a comunidade científica, de modo que os sistemas e os protocolos tradicionais existentes hoje, não são suficientes para suprir as necessidades exigidas por estas aplicações. Nos últimos anos muitas tecnologias emergiram com a finalidade de oferecer suporte ao tratamento de dados intensivos, no que diz respeito à manipulação, visualização, interpretação e processamento. Os sistemas distribuídos de larga escala estão em constante evolução, buscando oferecer suporte à computação intensiva e transpor os desafios de distribuição dos dados, transporte e gerência dos recursos.

Tendo isto em vista, somos confrontados com a necessidade de se implantar uma infraestrutura adequada para gestão da informação, que atenda à demanda e sirva de suporte às etapas posteriores de processamento, utilizando para isto os benefícios da computação paralela e dos paradigmas de processamento de dados intensivos. Através dos esforços dedicados nesta pesquisa será possível aplicar técnicas de *Data Warehouse* e integração de dados capazes de gerir grandes bases nas áreas de saúde pública de maneira integrada e escalável.

1.3 OBJETIVOS

Este trabalho tem como objetivo principal atuar nas etapas de pré processamento das bases de dados fornecendo os métodos necessários para aumentar a eficiência da comparação de registros e geração dos pares. Serão abordadas nesta pesquisa questões específicas das diferentes etapas relacionadas com a aplicação dos métodos de ETL: análise dos dados, limpeza, padronização, preservação da privacidade, transformação através do método de filtros de Bloom e testes de similaridade através do coeficiente de Dice. O estudo, sistematização e aplicação de tais etapas, são empregados com o propósito de oferecer suporte para a execução dos algoritmos de correlação probabilística e determinística.

A infraestrutura, objeto desta pesquisa, busca utilizar as vantagens da tecnologia de processamento distribuído para computação de dados em larga escala e servir como base para um estudo de caso vinculado, que tem como objetivo identificar e, caso exista, avaliar o impacto do programa de transferência de renda do Bolsa Família na redução da morbimortalidade por tuberculose e hanseníase.

1.3.1 Objetivos Específicos

Os objetivos específicos deste trabalho incluem:

1. Organizar e analisar os dados utilizando métodos estatísticos para avaliação de frequência, *missing* e valores ausentes, favorecendo um estudo da qualidade dos parâmetros;
2. Eleger as variáveis mais relevantes para a correlação probabilística em cada base de dados envolvida no processo;
3. Desenvolver rotinas de eliminação de registros duplicados (deduplicação) em todas as bases de dados utilizadas na correlação;
4. Desenvolver rotinas de limpeza dos dados no que diz respeito à substituição dos valores ausentes; padronização do formato de datas, códigos e nomes; remoção de caracteres especiais; substituição de nomes comuns;
5. Aplicar métodos de comparação de campos com preservação da privacidade, garantindo anonimização dos dados identificados;
6. Realizar blocagem dos arquivos, com o objetivo de reduzir a quantidade total de comparações;
7. Exportar os arquivos pré-processados para serem utilizados pelas rotinas de comparação;

1.4 ORGANIZAÇÃO DO TRABALHO

A organização deste trabalho segue a estrutura explicada a seguir. O Capítulo 2 apresenta uma revisão da literatura descrevendo os principais conceitos relacionados à integração de dados, ETL e à correlação probabilística de registros. O Capítulo 3 contextualiza este trabalho no Estudo de Caso em questão, justificando a necessidade e aplicabilidade

de todos os métodos desenvolvidos. Os detalhes sobre o desenvolvimento de todos os módulos para as etapas do pré-processamento são descritos no Capítulo 4. Este Capítulo descreve, inclusive a metodologia e avaliação dos resultados. O Capítulo 5, finalmente, apresenta uma síntese dos resultados, as discussões mais relevantes as limitações e principais dificuldades encontradas, os trabalhos que ainda estão em desenvolvimento e os que serão realizados no futuro.

Capítulo 2

REFERENCIAL TEÓRICO

2.1 INTEGRAÇÃO DE DADOS E PROJETOS DE WAREHOUSING

Ao problema de se relacionar informações referentes a uma mesma entidade localizadas em fontes diferentes, convencionou-se chamar Integração de Dados. O objetivo desta disciplina é prover acesso a um conjunto de fontes de dados heterogêneas e autônomas a fim de que, sobre as bases de dados deste novo ambiente, seja possível executar consultas de forma eficiente (CAVALCANTI; FELL; DORNELAS, 2005). A utilização de boas estratégias de integração de dados implica em avanços em diversos campos não apenas da ciência, mas também para administradores de um modo geral que precisam otimizar a informação para tomadas de decisão. Nesse sentido, setores de *business inteligence* têm traçado estratégias e fornecido métodos para tornar um enorme volume de dado bruto de diversas fontes em informações significativas.

O processo de integração de dados representa um enorme desafio. Em primeiro lugar, executar operações sobre bancos de dados distribuídos de forma eficiente ainda é um grande problema. Quando esses bancos vêm de fontes diferentes o problema, em relação ao poder de processamento de consultas, é ainda maior. Em segundo lugar, a organização lógica dos bancos podem diferir bastante. Mesmo quando dois bancos possuem a mesma finalidade, representando as mesmas entidades, seus esquemas tendem a ser diferentes um do outro. Isto ocorre porque a representação das entidades é uma tarefa que utiliza métodos de interpretação e considerações subjetivas de cada projetista. A representação dos dados certamente não será idêntica em diferentes bancos. Um exemplo clássico é a representação de endereço: na dúvida se a melhor representação para esta entidade é transformá-la em um único atributo (string) ou criar uma classe com diferentes atributos, frequentemente nos deparamos com diferentes representaçõeses para este caso. O fato é que integração de dados só é possível se for possível contornar todas estas questões de heterogeneidade semântica (DOAN; HALEVY; IVES, 2012).

A Figura 2.1 ilustra a arquitetura do processo de integração. A origem dos dados pode ser bancos de dados estruturados, formulários HTML ou aplicações que utilizam um banco de dados. As camadas denominadas *wrappers* são responsáveis por solicitar

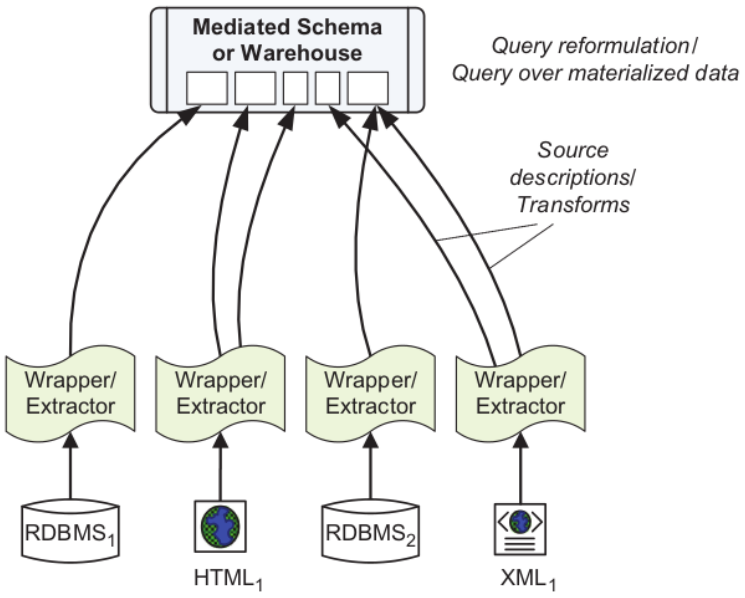


Figura 2.1 Arquitetura do processo de integração de dados e warehouse (DOAN; HALEVY; IVES, 2012)

e analisar as fontes de dados. Para prover a comunicação entre as fontes de dados e o sistema *warehouse*, é necessário que haja uma camada de transformação. Esta camada será responsável por enviar consultas, obter respostas, e caso necessário, aplicar transformações sobre estas respostas de modo a adequá-las ao propósito. O usuário que deseja interagir com o sistema de integração de dados, terá apenas a visão do esquema mais externo (esquema mediador). Os atributos existentes nas fontes de dados não são relevantes para a aplicação, mas apenas um subconjunto destes.

Usualmente, um banco de dados possui apenas uma parte da informação sobre determinada entidade, uma vez que eles são projetados para antender a um domínio específico de determinada aplicação. A integração provê resultados de grande valor, através da combinação de informações de diversas fontes. Motivados por isto, os gestores têm encontrado no projeto de *Data Warehouse* (DW) um modo de melhor explorar estas informações. *Data Warehouse* descreve o uso de um conjunto de dados integrados, agregados, não voláteis e orientados a assunto (SRIVASTAVA; CHEN, 1999). Seu conceito, popularizado e primeiro definido por (INMON, 1992), preconiza a visualização e modelagem dos dados a partir de múltiplas perspectivas envolvendo desde a etapa de extração dos dados, padronização, até o armazenamento e apresentação para o usuário final. A Figura 2.2 apresenta uma visão simplificada do esquema de *warehousing* e a relação deste com os serviços que o faz funcionar.

Segundo (DOAN; HALEVY; IVES, 2012), as ferramentas propostas para o que convencionou-se chamar *Extract, Transform and Load* (ETL), servem aos propósitos de um DW no que diz respeito a:

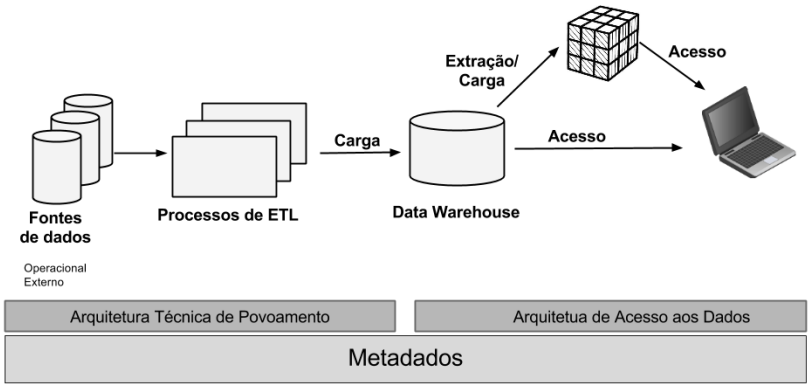


Figura 2.2 Relação entre o projeto de data warehouse e os serviços associados

1. Filtros de Integração: Gerencia a importação dos diversos formatos de arquivos externos, que muitas vezes não derivam de bancos de dados relacionais.
2. Transformação: Filtra os dados e, normalmente, envolve conceitos semelhantes ao mapeamento de esquemas. Pode modificar a estrutura da arquitetura de origem, fornecendo todo o suporte para que os dados sejam acessíveis nas etapas posteriores.
3. Deduplicação: Utiliza conceitos de *Data Matching* para verificar quando dois registros em ambientes diferentes dizem respeito à mesma entidade. O propósito desta etapa é evitar redundâncias.
4. Caracterização: Oferece uma visualização das propriedades dos dados dentro do ambiente *warehouse*. Para isto pode utilizar tabelas, histogramas ou outras informações.
5. Gerenciamento de Qualidade: Oferece ferramentas para realizar testes a fim de constatar se os dados estão corretos, completos e consistentes.

O *DataMart* é o produto gerado como resultado do projeto de *Data Warehouse*. Nele estão contidos os dados operacionais próprios de cada domínio, que serão úteis nas tomadas de decisão. Geralmente, estes dados precisam ser acessados rapidamente, por isto devem ser altamente indexados ou armazenados em Sistemas Gerenciadores de Bancos de Dados Multidimensionais (SGBDM) (COSTA, 2001). O *Data Mart* é, portanto, o responsável por tornar a informação de fato acessível através da reestruturação dos dados.

2.2 O QUARTO PARADIGMA: COMPUTAÇÃO INTENSIVA DE DADOS

As informações digitais produzidas hoje no mundo, fomentada pelas diversas aplicações têm atingido proporções extraordinárias e continua se expandindo. A quantidade de informações que existem hoje armazenadas em formato digital já são frequentemente tratadas em termos de distância entre a Terra e a Lua. Os setores corporativos já compreenderam que esta quantidade de dados proliferados em diversos campos podem re-

presentar descobertas ou benefícios importantes e não devem ser negligenciados. Se, no futuro, os dataceters terão que enfrentar grandes desafios para armazenar a quantidade de informações que serão produzidas, uma vez que a capacidade de armazenamento disponível já está em seu limite, processar as os dados existentes hoje e extrair deles todas as informações que se deseja ainda é um problema complexo e custoso.

O crescente volume de dados trouxe para computação científica o surgimento de um novo paradigma, uma vez que as soluções tradicionais de protocolos e sistemas não são suficientes para oferecer o tratamento adequado destes dados. Por exemplo, na área de genética, um grande desafio diz respeito ao sequenciamento do genoma. Considere que cada indivíduo tem um DNA único com bilhões de pares básicos, mas apenas algumas partes destes são sequenciados, correspondendo a algumas centenas de milhares de pares, gerando cerca de 25 KB. Esta operação exige um custo e tempo consideráveis para ser executada. Sabe-se que cerca de 1% do genoma leva à proteção de proteínas e descobrir qual a função dos outros 99% ainda é uma questão em aberto. As simulações científicas estão gerando uma enorme quantidade de informações, o que torna as ciências experimentais extremamente dependentes do processo computacional. Neste contexto científico, os dados são capturados por instrumentos ou gerados por simulações, processados por software, armazenados por mídias eletrônicas e analisados por métodos de gerenciamento estatístico. Disciplinas como bioinformática emergiram nos dias de hoje como áreas científicas totalmente orientadas por dados, justificando a necessidade e urgência pelos avanços em computação intensiva de dados. Partindo deste ponto em que TI e ciência se encontram, o quarto paradigma pode ser definido em termos de três atividades: captura, curadoria e análise (HEY; TANSLEY; TOLLE, 2011).

Ao longo do tempo diversas soluções foram propostas pela comunidade de Sistemas Distribuídos com o objetivo de gerenciar grandes volumes de dados em tempo hábil. Este cenário motiva diversos problemas, como por exemplo o gargalo de I/O causado pela tendência ao uso de *storage* centralizado e a exigência pela distribuição dos processos para executar as tarefas em menor tempo de resposta. Além destas, questões secundárias como consumo energético também tem constituído assunto de muitas pesquisas, de forma que o volume é apenas um de muitos desafios da área de computação intensiva de dados.

A caracterização de *Big Data* não inclui uma especificação de tamanho que a quantidade de dados deve ocupar. Por isto, a sua definição inclui a confirmação de um volume de dados que excede a capacidade de gerenciamento das ferramentas e soluções tradicionais, utilizado em determinada aplicação. O termo *Big Data* vem sendo amplamente mencionado no meio acadêmico e industrial a fim de se descrever o crescimento, a disponibilidade e o uso destas informações estruturadas ou não. É comum se utilizar cinco características principais para representar sistemas *Big Data*: velocidade, volume, variedade, veracidade e valor [Dijcks 2013].

Em *Big Data*, os dados podem vir de fontes distintas e o seu gerenciamento além de processamento e armazenamento, envolve questões adicionais como governança, segurança e políticas. O desafio é administrar o grande volume de dados e minerar as informações em um menor tempo de requisição. Os setores de inteligência empresarial têm obtido grande vantagem na utilização deste paradigma em seus processos de recolhimento, organização, análise e utilização de dados e informações que são úteis às tomadas

de decisão. É importante destacar que *Big Data* e *data warehouse* são abordagens complementares e não sinônimos. Enquanto o primeiro é uma abstração que engloba um conjunto de aplicações, metodologias e ferramentas destinadas ao armazenamento e processamento de grande quantidade de dados de forma rápida, o segundo pode ser visto como um método auxiliar para este processo. Outra diferença é que o processo de extração, transformação e carga ligado aos projetos de *data warehouse* pode se dar de modo mais lento até que as informações estejam disponíveis não enquanto que em *Big Data*, velocidade é um atributo fundamental (SILVA, 2012).

2.2.1 O Paradigma MapReduce e o Modelo Distribuído

Com a constatação de que as tecnologias tradicionais não são adequadas para tratar o imenso volume de dados que caracteriza a computação de dados intensivos, algumas tecnologias têm se destacado no suporte a estas necessidades. Do ponto de vista de processamento e análise o *MapReduce* (DEAN; GHEMAWAT, 2008) e Hadoop (WHITE, 2004) despontaram como referências mais populares.

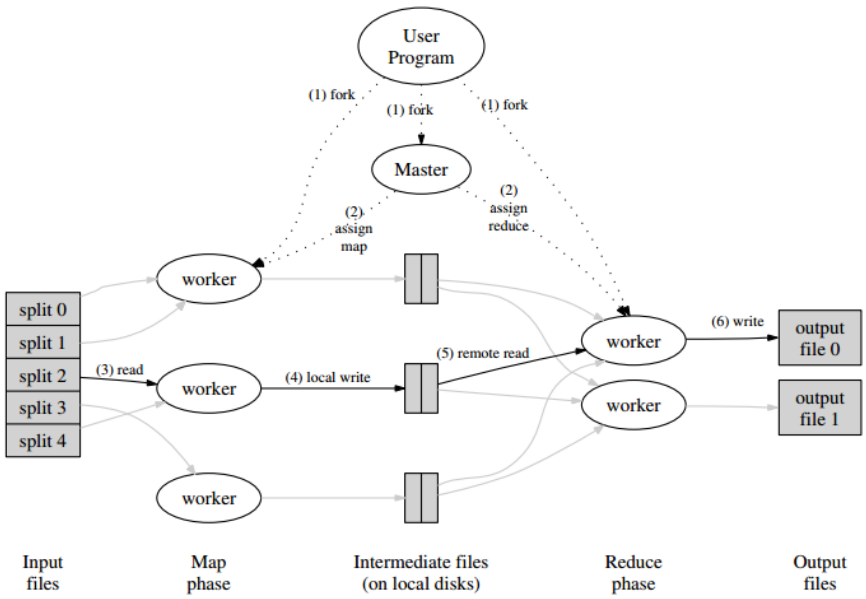


Figura 2.3 Arquitetura de execução do *MapReduce* (DEAN; GHEMAWAT, 2008)

MapReduce é um paradigma de programação para processamento de dados intensivos, proposto para uso em ambientes de memória distribuída, apesar de implementações sobre ambientes de memória compartilhada já terem sido apresentas. Desta forma sua proposta permite escalabilidade massiva através de computação em cluster. O conceito básico está em subdividir um grande problema em problemas menores, independentes o suficiente para serem resolvidos por *workers*[1]. Depois, cada worker retorna o resultado gerado,

[1]Parte integrante da computação, destinada a resolver uma tarefa. Pode ser entendido como *threads*

que será reagrupado. Este modelo serve como base de programação para o framework proprietário desenvolvido e mantido pela Google Inc (GOOGLE. . . , 2015) e também para o framework mantido pela Apache Software Foundation, o Hadoop que se tornou popular com sua implementação de código aberto.

A popularidade do *Apache Hadoop* tem sua explicação principal na economia que promove. Se antes, processar grandes conjuntos de dados exigia supercomputadores ou um hardware muito especializado, hoje o desafio e os custos estão direcionados na contratação de especialistas e cientistas de dados. Neste contexto, é comum a utilização de softwares conhecidos como *commodities*[1], motivo pelo qual existe uma grande preocupação para que o software seja inteligente o suficiente para lidar com as falhas. Para gerenciar o grande volume de arquivos de forma distribuída e segura, o *Hadoop* utiliza um sistema de arquivos especial chamado HDFS (*Hadoop Distributed File System*) (BORTHAKUR, 2008). O HDFS foi construído sobre o modelo "escreva uma vez, leia muitas vezes", ou seja, é aconselhável que os dados escritos sofram poucas modificações para se evitar sobrecargas. O sistema de arquivos do *Hadoop* é ideal para aplicações nas quais seja necessário ler uma quantidade muito grande de dados.

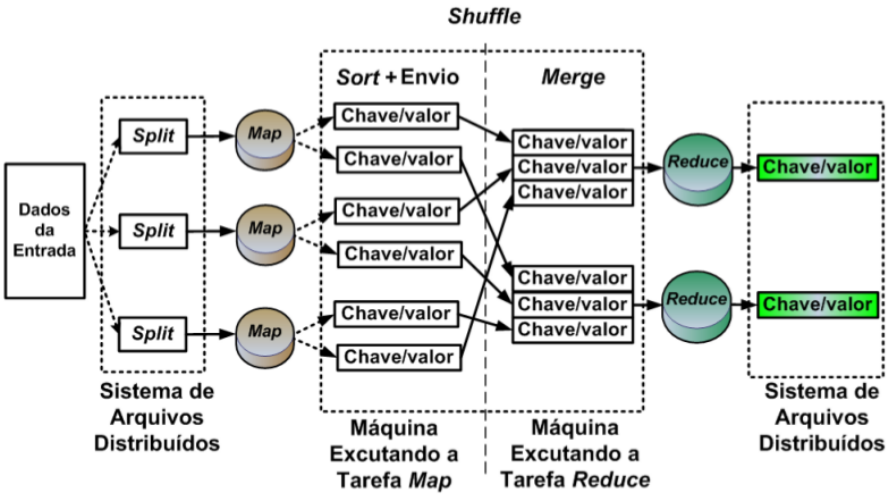


Figura 2.4 Fluxo de dados do *MapReduce* (ANJOS et al., 2013)

Comumente, o *MapReduce* é capaz de ler qualquer arquivo em formato texto desde que haja um fluxo (*stream*) de dados de entrada que possa ser transformado em pares chave-valor, embora frequentemente sejam usados arquivos com uma formatação especial de separação como ".csv". O *MapReduce* surgiu com a proposta de prover abstração do paralelismo de tarefas e controle de falhas e lidar com dados distribuídos. Esta primitiva de programação é baseada em linguagens funcionais como LISP e tem como base duas opererações principais: um *Map*, escrito pelo usuário, que computa dados de entrada

em um núcleo de processamento, *cores* em processamento *multi-core*, ou máquinas em um *cluster*.

[1]Componente ou dispositivo de custo relativamente baixo, por isto, encontrado com grande disponibilidade.

gerando um conjunto de pares chave/valor e um *Reduce*, também escrito pelo usuário, que agrupa o resultado do map gerando um resultado final também expresso em pares chave/valor. À biblioteca *MapReduce* cabe o trabalho de agrupar todos os pares intermediários com a mesma chave e passar o resultado para a função *Reduce*. Esta tarefa é denominada *Shuffle* e como ilustrado na Figura 2.4, é constituída de um processo *sort*, que ordena as chaves e serializa os dados e um *merge*, que agrupa as chaves de modo adequado. O *Map* produz dados intermediários que são gravados no disco local.

A Figura 2.4 demonstra a execução deste modelo. Cada nó no sistema é chamado de *worker* e pode assumir diversas funções. O *master* é o responsaável por atribuir tarefas *Map* e *Reduce* além de armazenar estruturas capazes de manter o estado de cada worker, colaborando para a tolerância a falhas. As falhas em um *worker* são verificadas através de *heartbeat* enviadas ao *master*. As falhas do *master*, por sua vez, são controladas através de *checkpoints* (DEAN; GHEMAWAT, 2008). Uma vez que a função *Reduce* está dependente do *merge* da etapa anterior, ele só acontece depois que todas as tarefas *Map* terminarem de executar.

2.2.2 Alternativa ao MapReduce: Apache Spark

O *MapReduce* obteve bastante êxito e popularidade nos setores industrial e acadêmico, que puderam, ao implantar sua infraestrutura, analisar terabytes de dados em inúmeros computadores de baixo custo. Na prática, novos cenários continuam a desafiar os limites deste modelo de programação e tecnologias alternativas têm surgido com o propósito de melhor atender à demanda de processamento intensivo de dados.

Em diversas aplicações, como por exemplo algoritmos iterativos de aprendizagem de máquina e ferramentas de *data analysis*, é necessário submeter uma série de operações paralelas reutilizando um conjunto de dados. Para tais cenários, o Apache Spark (ZAHARIA et al., 2010) foi apresentado, como uma solução para computação em cluster capaz de prover uma abordagem mais apropriada e manter as características de escalabilidade e tolerância a falhas do *MapReduce*. Mais uma vez, a natureza do problema implica diretamente na escolha do modelo de programação adequado. A solução tradicional do paradigma *MapReduce* apresenta algumas deficiências quando é necessário lidar com *jobs iterativos* uma vez que tais *jobs* precisam ler várias vezes diretamente do disco, implicando em perda de desempenho (ZAHARIA et al., 2010).

O *Spark* utiliza uma abstração para representar uma coleção de objetos, somente para leitura, chamada RDD (*Resilient Distributed Dataset*). Um RDD é construído à partir dos dados armazenados em local confiável, o que permite que um RDD seja facilmente reconstruído em caso de falhas. É possível construir um RDD de quatro maneiras: a partir de um arquivo existeme em um sistema de arquivos distribuído, como o HDFS do *Hadoop*; a partir do sistema de arquivos local; usando transformações sobre outro RDD existente e alterando a persistência do RDD através das ações *cache* ou *save* (ZAHARIA et al., 2010).

Diversas operações paralelas podem ser executadas sobre os RDD que representam os dados. A função *reduce* combina os elementos de um conjunto de dados usando uma função associativa, retornando um resultado ao programa principal. A função *collect*,

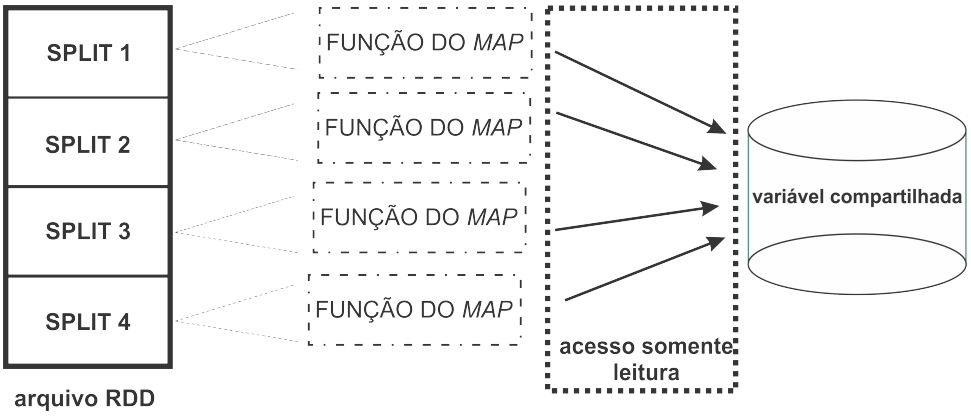


Figura 2.5 Abstração e acesso aos arquivos no Spark

por sua vez, envia todos os elementos do *dataset* ao programa principal. Diferente do que acontece no *MapReduce* tradicional, a função *reduce* no *Spark* não fornece resultados intermediários. Ao invés disso, ele produz um resultado final que pode ser coletado apenas pelo programa principal.

No Spark as operações são executadas sempre sobre os objetos do tipo RDD. As operações suportadas são classificadas em transoformações e ações. A primeira, cria um novo RDD a partir de um já existente e a segunda, retorna um valor ao programa principal depois da computação. A Tabela 2.1 resume as principais operações utilizadas neste trabalho. É simples portanto, entender que um *map* é uma transformação que executa determinada função sobre cada elemento de um RDD e retorna um novo RDD com os respectivos resultados.

	Operação	**Significado**
Transformações	map(func)	*Retorna um novo conjunto de dados formado a partir da passagem de cada elemento do conjunto original para a função func.*
	mapPartitionsWithIndex(func)	Executa sobre blocos do conjunto RDD, que levam consigo o índice daquela partição.
Ações	collect()	Retorna todos os elementos do conjunto de dados como uma matriz, para o programa principal.
	count()	Retorna o número de elementos no conjunto de dados.

Tabela 2.1 Operações do Spark utilizadas neste trabalho

O *reduce*, então, é uma ação que agrupa os resultados representados em um RDD e envia uma única resposta ao programa principal. Os dados utilizados por uma função dentro do *map* devem ser enviados na execução. Considerando que normalmente, cada nó que executa uma transformação lida com cópias de variáveis que muitas vezes não precisam ser atualizadas, o *Spark* permite que este acesso também seja possível através de *variáveis compartilhadas*, como *broadcast*, por exemplo. Quando transformadas em *broadcast* uma variável, somente leitura, é acessível por todas as máquinas, dispensando o envio de várias cópias e melhorando o custo da comunicação. Esta é uma boa estratégia para aplicações que lidam com arquivos muito grandes. A figura 2.5 ilustra o fluxo de

dados no Spark e exemplifica o modo como ele trata o acesso aos arquivos utilizando a vantagem de variáveis compartilhadas dentro de transformações como o *map*.

2.3 CORRELAÇÃO PROBABILÍSTICA DE REGISTROS

O problema de se relacionar dois ou mais registros que contém informaçoes suficientemente identificadoras e que representam uma mesma entidade no mundo real, em bancos de dados diferentes é conhecido como *Data Matching*. É comum encontrar o termo relacionamento de registros (*record linkage*) entre estatísticos e epidemiologistas enquanto que cientistas da computação estão mais familiarizados com o termo emparelhamento de dados (*data matching*) ou problema de identidade de objeto.

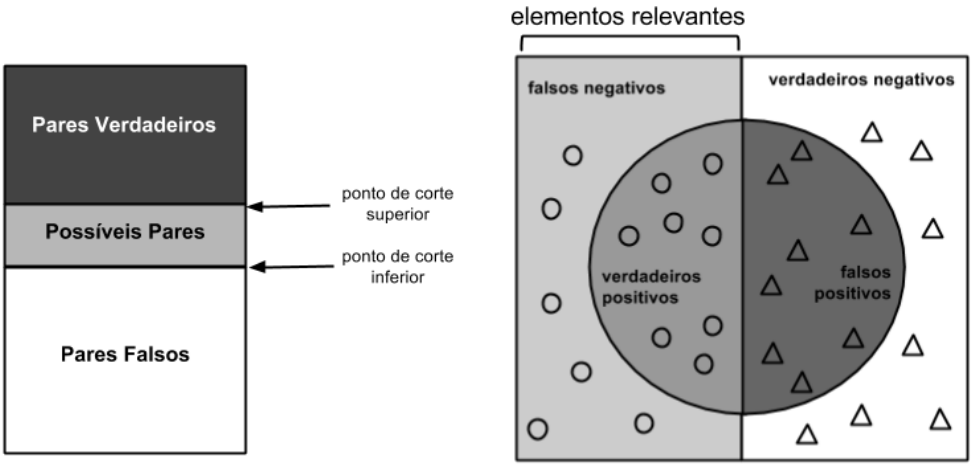
Pode-se formalizar o conceito de correlação como o processo de comparação entre dois ou mais registros que contém informações de identificação a fim de se verificar se esses registros referem-se à mesma entidade. Em outras palavras, se X e Y são duas tabelas relacionais, e considerando que cada linha ou registro de ambas as bases descrevem propriedades de uma entidade, é possível dizer que uma tupla x pertencente a X é pareada com a tupla y pertencente a Y se elas representam a mesma entidade no mundo real (DOAN; HALEVY; ZACHARY, 2012). O objetivo do método é encontrar todas as correlações verdadeiras entre X e Y.

As aplicações práticas deste método podem ser observadas em diversos setores. Em estratégias de avaliação de impacto, por exemplo, é necessário muitas vezes se utilizar métodos de pesquisa individual de modo que a hipótese, para ser provada, precisa ser observada em todo o grupo da análise. Portanto, o relacionamento de registros, ou record linkage é um dos métodos capazes de seguir coortes de indivíduos através da supervisão de bases de dados que contêm resultados contínuos (ROMERO, 2008). O grupo de interesse também pode ser observado individualmente ao logo de um tempo com o propósito de obter maior precisão na avaliação ou observar variações nas características dos elementos amostrais. A esta situação dá-se o nome de estudo longitudinal. Ao processo de relacionamento de dados para estudos longitudinais chamaremos pareamento longitudinal de registros.

Na prática, utiliza-se o método de correlação de registros para observar eventos e provar hipóteses. Por exemplo, os registros contidos nos sistemas que notificam óbitos ou nascimentos podem ser pareados com os registros das bases do Sistema de Informação sobre Mortalidade (SIM) ou Sistema de Informações sobre Nascidos Vivos (SINASC), para determinar se eles de fato foram registrados ou existe erro ou mal preenchimento em alguma das duas bases. Do mesmo modo, caso se espere conhecer os indivíduos internados com determinada doença, que são beneficiários do Cadastro Único, é necessário parear os registros do Sistema de Internações Hospitalares com os registros do CadÚnico caso a caso em busca das correlações positivas. Nesse sentido pode existir relações 1-para-1, como no caso da associação entre um nascido vivo e seu registro de óbito ou ainda 1-para-muitos, como o registro de um paciente e suas internações (SANTOS, 2008).

No passado, as técnicas de *record linkage* eram executadas manualmente ou através de regras. Aos poucos, métodos computacionais para a correlação foram sendo aprimorados, reduzindo ou eliminando a revisão manual, tornando o processo facilmente reproduzível e

incorporando vantagens como melhor controle de qualidade, consistência e principalmente velocidade. Uma pessoa que conduz um processo de correlação de registros é sensível o suficiente para perceber que mesmo havendo erros de digitação, ausência de uma letra no campo de nome, abreviações ou erros em data de nascimento, dois registros de fato representam uma mesma entidade. Os métodos computacionais desenvolvidos para este fim têm como ambição serem tão bons e tão sensíveis a erros, quanto qualquer processo de verificação humano (WINKLER, 2014). Assim, se dois registros possuem erros ou inconsistências em algum atributo de correlação, os registros ainda podem ser corretamente relacionados através da verificação de informações adicionais contidas nos demais atributos.



(a) Classificação dos pares após a correlação

(b) Elementos relevantes após a correlação

Figura 2.6 Classificação e relevância dos pares após a etapa de decisão

A abordagem probabilística para correlação de registros tem como principal incentivo a não existência de uma chave única capaz de relacionar uma mesma entidade em duas bases distintas. Por isto, faz-se necessário utilizar um conjunto de atributos através dos quais uma probabilidade de correspondência pode ser definida. Deve existir um conjunto de variáveis sobre as quais duas tuplas que representam uma mesma entidade sejam pareadas. Esse método requer um processo de raciocínio sobre as chaves envolvidas que implique na tomada da decisão de correspondência, não-correspondência ou indeterminação. Este, por exemplo, é o problema de se determinar se o registro "Maria dos Santos Oliveira, Rua Caetano Moura, Salvador" e "Maria S. Oliveira, rua Caetano Moura, Salvador" são referências à mesma pessoa. A desvantagem dessa abordagem é que ela leva um tempo muito maior para ser executada do que métodos não-probabilísticos além de que entender ou depurar abordagens probabilísticas é um processo muito mais difícil. O grande desafio do relacionamento é tentar parear registros de bases de dados com diferentes esquemas obtendo uma acurácia ótima. São muitos os problemas que dificultam a comparação, tais como abreviações, diferentes convenções de nomes, omissões (*missing*) e erro de transcrição e coleta. Além disso um outro grande problema é escalar

os algoritmos para grandes conjuntos de dados.

Quando duas bases contém o mesmo identificador único, o relacionamento entre elas pode se dar de através de correlação exata, o que chamamos relacionamento determinístico. A chave da correlação determinística pode ser um atributo ou um conjunto de atributos, dependendo da qualidade da variável utilizada como chave. O método probabilístico tradicional foi primeiramente proposto por (FELLEGI; SUNTER, 1969) sendo esta a base para a maioria dos modelos desenvolvidos posteriormente e para este trabalho. Este método utiliza um conjunto de atributos comuns dos registros para identificar pares verdadeiros. Um grande complicador do processo de correlação probabilística são as possíveis inconsistências no preenchimento dos campos. O método precisa ser sensível o suficiente para contornar estes problemas, relacionando corretamente dois registros, mesmo que haja pequenas diferenças entre eles.

Um par é considerado verdadeiro se a combinação dos atributos gerar chaves predominantemente semelhantes; falso se a combinação gerar chaves comprovadamente distantes e indeterminado se não for possível classificá-lo em verdadeiro ou falso. Seguindo o que foi colocado, a Figura 2.6(a) demonstra a representação padrão gerada pela decisão da correspondência. Algumas abordagens mais acuradas dispensam a classificação dos pares em "possíveis pares". De fato, uma solução de correlação probabilística de registros que seja 100% confiável, não existe. A Figura 2.6(b) chama atenção para o fato de que alguns elementos relevantes podem ser incorretamente inseridos no grupo de pares falsos. Da mesma forma, pares que não se referem a mesma entidade podem ser incorretamente insidiidos no grupo de pares verdadeiros. Considerando a classificação de Verdadeiros Positivos (VP), Verdadeiros Negativos (VN), Falsos Positivos (FP) e Falsos Negativos (FN) é preciso definir dois conceitos importantes utilizados para avaliar o resultado gerado por algum método de comparaçãoo de registros: Sensibilidade e precisão

$$sensibilidade = \frac{\sum VP}{\sum VP + \sum FN} \tag{2.1}$$

$$precisão = \frac{\sum VP}{\sum VP + \sum FP} \tag{2.2}$$

2.3.1 Transformações e Métodos de Comparação

Na prática é comum que pares de strings, relacionadas a uma mesma entidade, contenha variações de digitação ("Rafael" e "Raphael", por exemplo). Por isto, as funções comparadoras de string devem ser capazes de lidar com tais sequências aproximadas, a fim de se obter a vantagem de recuperar registros que seriam perdidos em uma comparação exata caractere por caractere. Jaro, em sua pesquisa, demonstrou que cerca de 25% do nome e 20% do sobrenome contém alguma divergência, em registros que são comprovadamente pares verdadeiros (JARO, 1989). Diversas metodologias de comparação de strings emergiram com a proposta de estabelecer uma medida contínua de similaridade equivalente à faixa [0, ..., 1], em detrimento das medidas binárias [0,1] que classificam os pares apenas como verdadeiros pareados ou falsos pareados (DURHAM et al., 2012).

A recuperação da informação por similaridade utilizando strings aproximadas envolve

a utilização de diversas técnicas. Uma bastante difundida diz respeito ao uso de funções que retornam um código fonético para uma cadeia de caracteres. A justificativa para as pesquisas em código fonético está embasada no fato de que existe uma grande variação na escrita de nomes e palavras em relação a sua pronúncia. Os algoritmos desenvolvidos para atender a este propósito buscam normalizar as palavras na medida da similaridade da sua pronúncia. Apesar disto, alguns estudos apontam para uma desvantagem da comparação de strings usando este método, contida no fato de que codificação fonética possivelmente produz uma taxa maior de falsos positivos do que métodos que não o utiliza (SCHNELL; BACHTELER; REIHER, 2009)

2.3.2 Preservação da Privacidade

Em diversos contextos e aplicações, como em áreas de saúde pública por exemplo, existem regulamentações e políticas de privacidade que exigem confidencialidade sobre as variáveis identificadoras dos indivíduos, justificando a utilização de algum tipo de criptografia. O processo de correlação de registros nesse caso, é conhecido como *Privacy-preserving Record Linkage* (PPRL)(CLIFTON et al., 2004) e impõe ao método tradicional de correlação, novos requisitos que garantam anonimização. Esta abordagem *encode-and-compare*, exige que os campos selecionados para a comparação sejam transformados em campos codificados e depois comparados para verificação de semelhança.

Muitas soluções envolvem procedimentos de criptografia padrão como funções hash de uma única mão. Um protocolo utilizando mapeamento em hash aplicado em cada bigrama da string foi proposto por (CHURCHES; CHRISTEN, 2013). São utilizadas funções hash com combinações particulares e em um primeiro passo, as strings passam por um processo de padronização e pré-processamento e depois, o conjunto de bigramas é formado. De toda forma, a principal preocupação do método de correlação de registros utilizando preservação da privacidade é calcular o quão similares são duas strings que estão encriptadas. Para resolver esta preocupação, (SCHNELL; BACHTELER; REIHER, 2009) sugere a utilização da metodologia de Filtros de Bloom, proposta inicialmente por Burton H Bloom (BLOOM, 1970). Além desta, outras soluções vem sido utilizadas para resolver problemas do relacionamento e comparação de registros.

2.3.3 O Método de n-gramas

Diversos domínios e áreas do conhecimento relacionadas com processamento estatístico de linguagem natual como mineração de texto, utilizam métodos baseados em n-gramas para extração de informação. Um n-grama pode ser entendido como uma sequência contígua ou não contígua, de tokens, mantendo a ordem e as posições com que surgem. Um corpus, por sua vez, pode ser entendido como uma grande coleção de textos e diz respeito a associação de vários documentos a uma língua, a um alfabeto ou conjunto de símbolos, e a um conjunto de regras que agrupam os diferentes símbolos em unidades lexicográficas (NUNO, 2002). A dimensão de um n-grama diz respeito à distância, incluindo posições vazias, ocupada entre o primeiro e o último token. O termo bigrama será utilizado aqui para definir um n-grama de comprimento 2, e trigrama, um n-grama de comprimento 3. Sendo assim, podemos dizer, por exemplo, que [AP] é um bigrama para a palavra

“APPLE” enquanto que [APP], um trigrama. A compreensão do uso de n-gramas para comparação de palavras, está fundamentada no fato de que registros muito parecidos possuem um grande número de n-gramas em comum.

Questões relacionadas à escolha dos n-gramas, como por exemplo a eliminação de espaços e pontuação, depende da natureza do problema e está vinculada a uma fase anterior de normalização. Considerando que o modelo de n-gramas estabelece uma relação probabilística a uma palavra n considerando as n-1 palavras anteriores, esta abordagem pode ser vista como uma medida de redundância ortográfica e ao mesmo tempo uma forma de codificar a ordem das letras nas palavras. O estudo de bigramas tem alimentado uma vasta área de pesquisas em relação a outras modalidades de n-grama. A relevância do uso de bigramas está justificada no estudo de frequência de ocorrência de unidades sublexicais. Sem entrar em detalhes, a constatação de que, em uma determinada língua, certos padrões ocorrem com mais frequência do que outros define os bigramas como uma medida de redundância ortográfica. Motivados por isto, diversos autores têm destacado a importância dos bigramas em seus estudos sobre modelos de reconhecimento de palavras (GUARALDO; REIS, 2009).

A frequência de ocorrência dos bigramas, portanto é uma metodologia comum para identificação de padrões. Baseado nisto, uma estratégia para calcular similaridade entre strings é a verificação de bigramas comuns entre dois conjuntos de bigramas. Em outras palavras este método se resume na verificação de quantos bigramas semelhantes duas strings possuem. Se por exemplo, temos que os bigramas da palavra ”Maria”são ”M”, ”Ma”, ”ar”, ”ri”, ”ia”e ”a”e para ”Marina”, ”M”, ”Ma”, ”ar”, ”ri”, ”in”, ”na”e ”a”, pode-se dizer que a porcentagem que representa a quantidade de bigramas semelhantes é também a expressão da similaridade entre as duas strings. Abordagens como está são interessantes, mas não costumam serem úteis na prática. A comparação entre string é uma tarefa computacionalmente custosa e a perda de desempenho pode tomar grandes proporções considerando a aplicação deste método em grandes bases de dados.

2.3.4 O Método de Filtros de Bloom

Filtro de Bloom é um método utilizado para se fazer verificações de *membership*. O método consiste em uma estrutura de dados representada como um vetor de bits, de tamanho n, inicialmente marcados com 0. O que pode representar uma vantagem em problemas de diversos domínios é o fato de que este método não permite falsos negativos. Em outras palavras, dois registros exatamente iguais sempre vai produzir filtros exatamente iguais. O contrário não é válido, pois a existência de falsos positivos é possível, uma vez que registros diferentes podem produzir filtros semelhantes, como será abordado em seguida.

Para compor a codificação, cada string é decomposta em n-gramas (consideremos aqui a decomposição em bigramas). Cada bigrama é submetido à um conjunto de k funções hash que delimitará uma posição específica (de 1 a n) no vetor para aquele bigrama. Desta forma, cada elemento influencia uma posição de acordo com as funções de hash que foram submetidas. Comumente, um bigrama pode influenciar mais de uma posição no vetor. Uma vez que o módulo que mapeia cada bigrama em posições específicas seja

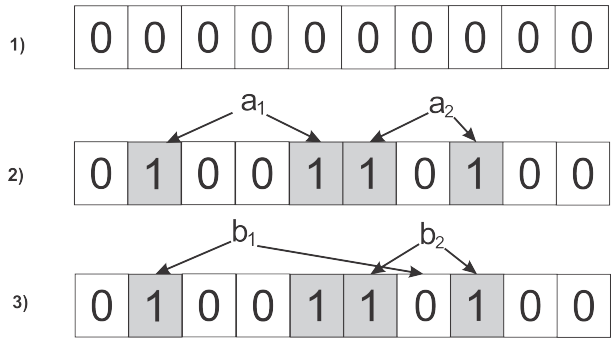


Figura 2.7 Exemplo de preenchimento do filtro de Bloom utilizando duas funções hash

mantido privado e as funções utilizadas sigam as recomendações para prevenir ataques de dicionário, é possível garantir com alguma certeza, que a codificação gerada é capaz de manter a privacidade das informações dos indivíduos. A Figura 2.7 demonstra o preenchimento de um Filtro de Bloom em que cada elemento (bigrama) influencia duas posições diferentes. Uma vez que a escolha da posição que determinado bigrama vai influenciar é de origem randômica, dois bigramas iguais certamente influenciarão bits de mesma posição, mas não existe impedimento para que dois bigramas diferentes influenciem bits de posição igual. Este fato é um evento probabilístico previsível, determinante para o aumento da ocorrência de falsos positivos.

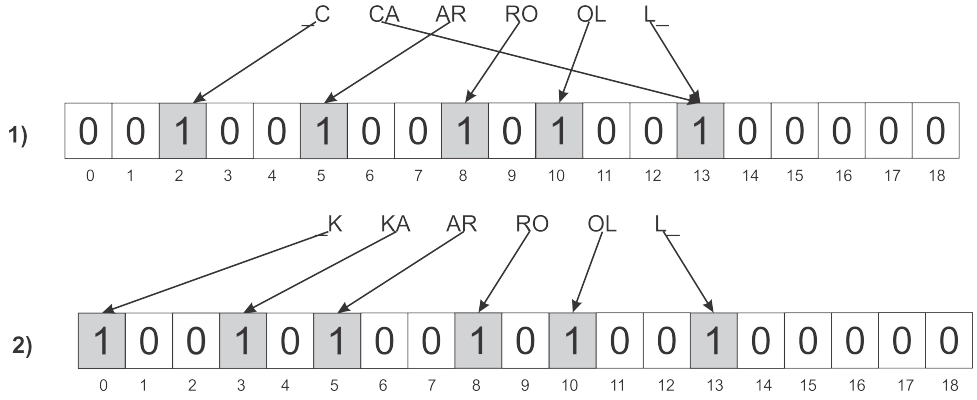


Figura 2.8 Exemplo de mapeamento dos bigramas no filtro de Bloom

Para verificar se um elemento, neste caso um bigrama, existe em um filtro, é preciso consultar se todos os bits influenciados por aquele bigrama estão marcados com 1's. Na Figura 2.7 o vetor foi preenchido com os elementos *a1* e *a2*. Ao se consultar a existência de dois outros elementos neste vetor, verificamos que *b1* não está ali representado. Neste caso, para comparar dois registros, é preciso garantir que eles concordem no número de bits que compõem o filtro, no número de funções hash e em quais delas foram aplicadas. Para avaliar o índice de similaridade entre dois registros pode-se utilizar o coeficiente de Dice que é dado por (3.3) , sendo α e β os bits mapeados pelas funções de hash para os bigramas do vetor da primeira string e da segunda string, respectivamente.

$$DICE = 2(\frac{|\alpha \cap \beta|}{|\alpha + \beta|}) \tag{2.3}$$

A Figura 2.8 ilustra a codificação de dois nomes para filtros de Bloom usando bigramas, com um espaço em branco em cada extremidade, e um vetor de tamanho 19. É possível observar que os bigramas em comuns entre as duas strings são mapeadas sempre em posição igual. Os bits que equivalem a bigramas diferentes serão marcados em posições de origem randômica sendo comum suas representações em posições diferentes, como no exemplo da Figura em que o bigrama _C influenciou a o bit da posição 2 e o bigrama _K o de posição 0. De acordo com o coeficiente de Dice, o índice de similaridade para esse exemplo é 0,728. É fácil notar que três fatores influenciam diretamente no número de falsos positivos gerados: O tamanho do vetor, o número de funções hash utilizadas e o número de elementos no conjunto mapeados no vetor (SCHNELL; BACHTELER; REIHER, 2009).

Este protocolo de comparação de registros utilizando os paradigmas de preservação de privacidade pressupõe a existência de dois detentores de bases de dados A e B. Estes serão os responsáveis por decompor os registros em conjuntos de bigramas, estabelecer o tamanho l do vetor de bits e mapear cada bigrama utilizando k funções hash em posições específicas. Já foi previamente demonstrado que duas funções hash são necessárias para criar um filtro de Bloom com k funções hash sem com isto, aumentar o número de falsos positivos (KIRSCH; MITZENMACHER, 2008). Então é possível obter os k valores de hash através da função

$$g_i(x) = (h_1(x) + ih_2(x)) \ mod \ l \tag{2.4}$$

onde i varia de *0* a *k-1*, e l é o tamanho do vetor de bits. O protocolo é considerado seguro uma vez que as comparações são realizadas por um terceiro módulo C que não conhece e nem tem acesso à construção do filtro através do mapeamento dos bigramas utilizando as funções de hash. Considerando que são utilizadas funções de hash de uma via (ou seja, não existe método para recuperar a string original a partir do vetor de bits), ataques de dicionário não são possíveis, exceto se forem executados por A ou B. Entretanto ataques de frequência não estão descartados, pois a frequência de 1 em determinada posição reproduz a frequência do bigrama na string original (SCHNELL; BACHTELER; REIHER, 2009). Por isto a escolha do número k de funções hash implica diretamente na proteção e segurança do método.

Capítulo

3

ESTUDO DE CASO

3.1 COMPUTAÇÃO INTENSIVA DE DADOS NA ÁREA DE SAÚDE PÚBLICA

3.1.1 Visão Geral

Nos últimos anos, programas de transferência de renda como o Bolsa Família vêm sendo utilizados como alternativa para apoiar famílias em situação de pobreza e extrema pobreza e oferecer a estas famílias um piso mínimo de consumo. É observado que estes programas de transferência de renda apresentam contribuições positivas em áreas como saúde e educação e estudos que comprovem esta assertiva são desejáveis e necessários para o aprimoramento de políticas públicas. A pesquisa que foi utilizada como estudo de caso deste trabalho busca sobretudo avaliar a eficácia do programa de Bolsa Família sobre doenças infecciosas ligadas à pobreza, a saber, tuberculose e hanseníase. Muito se tem pesquisado sobre o impacto do PBF sobre a saúde dos seus beneficiários concentrando-se em aspectos nutricionais e da saúde da criança, mas até o momento nenhum estudo avaliou o impacto do programa com a morbimortalidade pelas doenças infecciosas tuberculose e hanseníase, que representam um grande problema de saúde pública no Brasil.

Foram utilizadas para esta pesquisa, bases de dados do Cadastro Único (CadÚnico), Sistema de Informação de Agravos de Notificação (SINAN), Sistema de Informações Hospitalares (SIH), Sistema de Informações de Mortalidade (SIM) e das Folhas de Pagamento do Bolsa Família (PBF).

Os objetivos desta pesquisa compreendem, além da avaliação do impacto, o desenvolvimento de tecnologias para utilizar grandes bases de dados como o CadÚnico e o Bolsa Família em conjunto com outras bases relacionadas à saúde. Espera-se com esta pesquisa identificar fatores de risco que possam levar às doenças, estimar os efeitos de determinadas intervenções, planejar campanhas, estabelecer relações entre epidemias em diferentes regiões ou populações entre outras possibilidades. É importante que as intervençõess planejadas atuem, além da assistência à saúde, sobre os determinantes sociais relacionados com a morbimortalidade de doenças ligadas à pobreza. Espera-se também desenvolver

uma prática inovadora de avaliação e monitoramento do Programa Bolsa Família permitindo uma observação mais imediata dos seus efeitos sobre as populações beneficiárias, com potencial para incrementar a eficiência e a transparência pública de suas gestões.

3.1.2 Os Desafios do Estudo Longitudinal

Nos últimos anos, com o crescimento da disponibilidade de bases de dados informatizadas na área de saúde, cresceu também a inclinação pelo relacionamento de diversas bases de dados com o objetivo de integrar as informações contidas nelas e extrair informações úteis na investigação da ocorrência de certos eventos e monitoração de desfechos. Em muitas investigações é necessário confrontar as informações contidas nos registros de duas ou mais bases diferentes, com a finalidade de se estabelecer ou não a correspondência dos pares de registros.

A qualidade do estudo da avaliação do impacto do Programa Bolsa Família na morbimortalidade pelas doenças infecciosas está diretamente relacionada com a quantidade e qualidade das informações que se pode extrair das bases de dados. Por exemplo, o Sistema de Notificação e Agravos (SINAN) é responsável por notificar e investigar doenças e agravos em todo o território nacional, entretanto é possível que existam pacientes cuja internação seja registrada no Sistema de Internações Hospitalares e a notificação, equivocadamente não exista no SINAN. Portanto, a utilização de diversas bases de dados, neste contexto tem o objetivo de fornecer informações complementares que só existem em sistemas diferentes, mas também contornar problemas como erros ou privação das informações disponíveis.

A utilização de um número identificador único capaz de distinguir os indivíduos em contextos variados nos setores de saúde pública, é praticamente inexistente no Brasil. Por isto o método utilizado para estabelecer esta correlação precisa utilizar diversos atributos disponíveis e estabelecer uma probabilidade de concordância entre cada par de registro na busca pelo par que possui maior probabilidade de se referir à mesma pessoa. Este método de correlação probabilística comumente utilizado, deve ser aplicado neste estudo para estabelecer as relações entre os pares, com o objetivo final de obter um banco de dados único e integrado, que contenha informações parciais de todos os arquivos originais envolvidos no processo.

Com a finalidade de investigar hipóteses e levantar afirmativas, os setores epidemiológicos utilizam estudos experimentais ou observacionais. Estudos descritivos são um método de estudo observacional cuja finalidade é determinar a distribuição das doenças ou condições relacionadas à saúde, segundo o tempo, o lugar ou as características do indivíduo (LIMA-COSTA; BARRETO, 2003). O estudo de caso apresentado, requer a observação detalhada e acompanhento de cada indivíduo. O recebimento do benefício é o fator de exposição que deve ser considerado. Por isto, é preciso acompanhar o grupo de indivíduos que receberam o benefício (e também aqueles que não receberam, para que seja estabelecido o grupo de controle) ao longo do tempo. Se as características relacionadas às doenças apresentam diferença entre um grupo e outro, é possível aos pesquisadores obter conclusões da relação entre doença e fator. Para analisar esta evolução de cada grupo de indivíduo são utilizados estudo de coorte.

A observação dos indivíduos beneficiários ou não beneficiários, ao longo do tempo e em todas as bases de dados envolvidas, compreende um processo de alto custo computacional e operacional. Neste momento, é importante atentar para o fato de que as bases de dados envolvidas nesta pesquisa são consideravelmente grandes. Existe hoje, mais de 50 milhões de indivíduos sendo beneficiados pelo Programa Bolsa Família. Além disso existe também a necessidade de seguir todos os indivíduos que já receberam o benefício, acompanhando as características relevantes para o estudo como o tempo e o valor recebido. Sabendo que as folhas de pagamento são emitidas mensalmente, é fácil constatar a complexidade de se visitar todos os arquivos disponíveis em busca de todos os indivíduos, a fim de se estabelecer este mapa.

O estudo longitudinal impõe ao método de correlação probabilística de registros uma complexidade ainda maior. Por isto, para os fins desta pesquisa, faz-se necessário o desenvolvimento de uma infraestrutura capaz de distribuir as tarefas através do hardware disponível e que utilize as vantagens da computação intensiva de dados para gerar respostas em tempo hábil.

Capítulo 4

TÉCNICAS DE PRÉ-PROCESSAMENTO

4.1 AVALIAÇÃO DA QUALIDADE DOS DADOS

O primeiro passo, antes de se pensar nas estratégias de relacionamento entre as bases, é conhecer sua estrutura e avaliar a qualidade dos dados. O objetivo nesta etapa é detectar anomalias, verificar se os dados estão íntegros e avaliar a qualidade do preenchimento de todos os atributos para, enfim, conseguir uma visão geral sobre quais informações de fato, são possíveis de se extrair e quais atributos podem ser utilizados com maior eficiência. Para a finalidade da correlação probabilístico de registros, proposto por esse trabalho, a etapa de análise da qualidade dos dados é de indispensável importância, pois é através dela que as variáveis mais adequadas para a comparação serão selecionadas.

A tendência mais intuitiva é procurar pelas variáveis que contenham informações de documentos, como RG e CPF. Entretanto, muitas vezes, informações que seriam tão específicas na identificação de um indivíduo não estão disponíveis, seja pela natureza dos domínios que realizam os cadastros, seja pelo mal preenchimento do profissional que realiza a entrevista ou simplesmente pela não exitência. Consideremos por exemplo um sistema de cadastro que contemple crianças de baixa renda. É muito provável que a maioria destas crianças não tenham nenhum documento de identificação além do registro de nascimento. Além disto, pode-se erroneamente imaginar que a inclusão de atributos identificadores pode aumentar a precisão da correlação. Contudo, se a análise estatística revelar uma grande incidência de valores ausentes neste campo, toda a etapa de linkage é comprometida, uma vez que o método utilizado é a correlação coletiva.

É esperado que todas as bases de dados envolvidas neste trabalho tenham qualidade duvidosa, isto é, os registros contidos nas bases podem conter duplicatas e as variáveis possuírem muitos valores ausentes. Tendo isto em vista, técnicas de limpeza de base de dados também devem ser empregadas a fim de identificar partes dos dados que são sujas, incompletas, incorretas ou irrelevantes e então substituí-las ou removê-las. O emprego destas técnicas é útil no sentido de tornar a execução mais rápida e os resultados mais precisos e corretos. (RANDALL; FERRANTE; SEMMENS, 2013) mostrou em sua pesquisa que as etapas de preparação das bases de dados são de irrefutável importância para

aumentar a qualidade da correlação de registros na classificação dos pares no grupo de *maching* e *non-matching*, colaborando para a redução do número de falsos positivos e falsos negativos. Em (GILL; STATISTICS, 2001), o autor ainda estima que o processo de limpeza é um dos principais passos no processo de correlação podendo representar 75% do esforço total do processo de linkage.

4.1.1 Descrição das Bases de Dados

As bases de dados selecionadas para a pesquisa de análise de impacto utilizada por este trabalho como estudo de caso, são geradas pelos sistemas de informação sob domínio do Ministério do Desenvolvimento Social e Combate à Fome (MDS) e Ministério da Saúde do Brasil. A Tabela 4.1 descreve todas as bases utilizadas, e os anos de abrangência. A aquisição desses dados se deu através de solicitações e acordos formais firmados com os Ministérios e expressa avaliação do Comitê de Ética em Pesquisa do Instituto de Saúde Coletiva da Universidade Federal da Bahia, que detém a responsabilidade sobre a pesquisa de Avaliação de Impacto.

Databases	Anos de Abrangência
SIH (Hospitalizações)	1998 a 2011 (todas as UFs)
SINAN (Notificações)	2000 to 2010 (todas as UFs)
SIM (Mortalidade)	2000 to 2010 (todas as UFs)
CadÚnico (Dados Socioeconômicos)	2007 to 2013

Tabela 4.1 Bases de dados utilizadas e seus anos de abrangência

O Cadastro Único identifica e caracteriza as famílias de baixa renda do Brasil. Entende-se família de baixa renda aquelas com renda igual ou inferior a meio salário mínimo por pessoa ou renda mensal familiar de até três salários mínimos. Ele é o principal instrumento para concessão de benefícios por parte do Governo Federal e o ponto de partida para a análise dos principais fatores que caracterizam a pobreza, permitindo a adoção de políticas públicas mais eficientes para a proteção social destes indivíduos. O cadastro é realizado a nível municipal, ficando sob responsabilidade das prefeituras. Ao longo de sua existência duas versões foram utilizadas, a versão 6.05 e a versão 7. As bases de dados exportadas pelas duas versões destes sistemas possuem estruturas e informações bastante diferentes considerando seu processo de aperfeiçoamento. Por isto, as análises descritivas tiveram início com os módulos da versão 7.

O módulo utilizado para a maioria dos testes desenvolvidos por esta pesquisa é composto pelas tabelas do ano de 2011 do CadÚnico. A Tabela 4.2 foi extraída da análise de frequência do CadÚnico 2011 e mostra uma descritiva para suas principais variáveis identificadoras. O Número de Identificação Social (NIS) é uma chave atribuída pela Caixa Econômica Federal a todos os registrados no CadÚnico para que possam desfrutar de benefícios e programas sociais. Este número é único e pessoal, portanto, é o atributo necessário para se relacionar das bases do CadÚnico com as folhas de pagamento do Programa Bolsa Família (PBF), processo que é feito de forma determinística. Esta correlação determinística é simples de ser projetada. Quando uma família é registrada

Atributo	Descrição	Ausentes
NIS	Número de Identificação Social	0,7
NAME	Nome do paciente	0
MUNIC_RES	Município de residência	0
SEXO	Sexo do paciente	0
RG	Registro Geral	48,7
CPF	Cadastro de Pessoa Física	52,1

Tabela 4.2 Análise descritiva do CadÚnico 2011

no CadÚnico ela precisa que um responsável familiar seja identificado (preferivelmente a mulher que tem a liderança da família). Este responsável familiar é, preferivelmente, o titular do recebimento do benefício do Bolsa Família. Através do seu Número de Identificação Social fica simples encontrar todos os seus dependentes, também beneficiários. O relacionamento entre o CadÚnico e todas as outras bases é feito de forma probabilística pois não existe uma chave, como o NIS, nestas tabelas. Por isto é preciso selecionar os atributos que sejam comuns às duas bases e que possuam um grande poder discriminatório. A variável sexo, por exemplo possui uma ótima qualidade de preenchimento, entretanto separa os registros em apenas dois grupos. Em outras palavras, não é possível julgar se duas pessoas são a mesma pessoa apenas observando a variável sexo. Portanto, esta é uma variável que deve ser usada como um critério extra para desempate e não como principal.

Atributo	Descrição	Ausentes (%)
MUNIC_RES	Número de Identificação Social	0
NASC	Data de nascimento	0
SEXO	Sexo	0
NOME	Nome completo	0
LOGR	rua que compõe endereço	0,9
NUM_LOGR	Number of house	16,4
COMPL_LOGR	Informações adicionais de endereço	80,7

Tabela 4.3 Análise descritiva do SIH 2011

Apenas uma base do CadÚnico analisada (versão 7, ano de 2011), possui mais de 115 milhões de registros. A manipulação de bases desta dimensão é incomum em ambientes de trabalho habituais. Ferramentas de análise estatística tradicionais em computadores comuns não apresentam bom desempenho executando operações simples sobre tais bases, pois muitas vezes o software exige disponibilidade de memória ou armazenamento além da existente. As bases do CadÚnico se apresentam em formato texto, com delimitadores de campo, o que facilita a sua manipulação através da utilização de qualquer software capaz de editar texto. A Tabela 4.3 apresenta uma análise de frequência para a base do Sistema de Informações Hospitalares (SIH) que contém internações por tuberculose e hanseníase. As bases do SIH também se apresentam em formato texto e sua tabela

para o ano de 2011 possui mais de 61 mil registros. A partir desta análise e considerando a coexistência dos atributos e qualidade do preenchimento, foram selecionados *nome*, *data de nascimento* e *município de residência* para compor a estrutura que realizará a comparação entre os registros. Além da coexistência em todas as bases, escolha destas variáveis também é justificada pela sua qualidade de preenchimento em relação ao número de valores ausentes. Esta primeira correlação entre CadÚnico e SIH, busca encontrar o grupo de indivíduos hospitalizados por tuberculose ou hanseníase que estão registradas no CadÚnico de modo que suas informações socioeconômicas possam ser resgatadas e em segundo plano, seja feita a verificação de recebimento ou não do benefício do Bolsa Família. A Tabela 4.4 foi resumida da análise de frequência da base do SIM e representa um total de 205.744 observações. Semelhante ao caso anterior, a correlação do CadÚnico com o SIM, busca identificar o grupo de indivíduos com óbito registrado sob o diagnóstico de tuberculose e hanseníase que existem no CadÚnico.

Atributo	**Descrição**	**Ausentes (%)**
MUNIC_RES	Município de residência	0
NASC	Data de nascimento	1,21
NOME	Nome completo	9,64

Tabela 4.4 Análise descritiva do SIM 2011

A observação destas bases aponta para algumas conclusões qualitativa. É possível observar, por exemplo, que o SIH apresenta uma baixa qualidade no preenchimento das suas variáveis. Um estudo um pouco mais prático também sugere que algumas variáveis tendem a apresentar menos confiabilidade do que outras como endereço, considerando que esta é uma informação cedida pelo paciente no ato da internação e quase sempre não existe meios de comprovação. Muitas vezes um paciente se desloca para receber atendimento em outro município e o cadastro de sua residência acaba sendo comprometido por falsas informações. Estas são conclusões que devem ser ponderadas nas decisões de projeto. É preciso estar atento para o fato de que, muitas vezes, o endereço ou município de residência será diferente entre dois registros que se referem à mesma pessoa, quer seja pela omissão da informação correta, quer seja pela migração. O processo de correlação precisa ser habilidoso o suficiente para contornar problemas como este.

4.2 ESTRUTURA E ORDEM DAS CORRELAÇÕES

O objetivo geral da utilização de todas essas as bases de dados é utilizar informações redundantes para observar as ocorrências de tuberculose e hanseníanse. O objetivo deste trabalho, por sua vez é fornecer o suporte para a sistematização, pré-processamento e transformações necessárias para que as comparações sejam realizadas e um quadro geral possa ser montado, de modo que os profissionais de estatística e epidemiologia possam, então, realizar as análises que sejam de suas competências na avaliação do impacto e conclusões diversas.

A Figura 4.1 mostra o panorama que se deseja alcançar ao se relacionar todas as bases. Se considerarmos o objetivo de identificar beneficiários do Bolsa Família com

CADUNICO	SIH	SIM	SINAN	PBF	...
registro_pessoa1	x			x	
registro_pessoa2					
registro_pessoa3		x	x		
registro_pessoa4				x	
registro_pessoa5				x	
registro_pessoa6		x			
...					

Figura 4.1 Acompanhamento das ocorrências na correlação entre as cinco bases

alguma ocorrência de tuberculose ou hanseníase, apenas três registros comporiam o quadro. Apesar disto, existem subquadros com ocorrências que podem ser úteis para a análise como por exemplo o quadro de pessoas registradas no CadÚnico que foram internados, que tiveram óbito registrado ou que foram notificados com o diagnóstico da doença estudada. Do ponto de vista epidemiológico, a análise de impacto reivindica que a correlação aconteça em granularidade ainda menor, pois um fator decisivo para tal análise é o tempo de exposição do evento. Em outras palavras, para compor o quadro geral é preciso observar o momento da ocorrência do evento (data da internação, do óbito ou da notificação) e a exposição ao programa (quantidade em meses do recebimento do benefício e o valor acumulado em determinado tempo). Para isto uma medida necessária é uma correspondência determinística que acompanhe aquele beneficiário ao longo de um perído de tempo, considerando e identificando inclusive, os intervalos em que ele não recebeu benefício algum.

O quadro geral é construído através do relacionamento entre as bases independentes e da composição dos subquadros. Uma vez que os esquemas das bases de dados são muito diferentes e também são diferentes as conclusões sobre a qualidade do preenchimento de seus atributos, as características dos parâmetros do relacionamento entre cada uma delas podem divergir. O método probabilísto de comparação desenvolvido por este trabalho exige que um conjunto de chaves seja selecionado a cada subgrupo pareado. Até o momento do desenvolvimento deste texto, dois subquadros foram compostos, referentes às colunas SIH e SIM da Figura 4.1, valendo-se dos métodos estudados e desenvolvidos neste trabalho.

Para a geração dos dois subgrupos foram utilizados os atributos *nome*, *data de nascimento* e o código IBGE do *município de residência*. Considerando que a comparação do *município de residência* se dá de forma determinística (método exposto na seção 4.3.3), as variações no *nome* e na *data de nascimento* serão determinantes na inserção ou não dos pares dos grupos M ou U.

4.3 EXTRAÇÃO, TRANSFORMAÇÃO E CARGA

A etapa dedicada ao projeto de ETL busca organizar todas as bases de dados existentes e consolidá-las em um formato e estrutura específicos de modo que elas estejam prontas e habilitadas para serem utilizadas pelo módulo que realizará a correlação, sem erros e com máxima eficiência. No processo de ETL as etapas de extração e transformação dos dados compreendem os procedimentos de limpeza, conversões de formatos, extração de atributos e todo o pré-processamento necessário, como substituição de valores ausentes e padronização, que pode representar o processo mais demorado na contrução de um Data Warehouse. A última fase desta etapa tem o objetivo de consolidar uma nova base totalmente limpa, padronizada e anonimizada que será utilizada para gerar os pares que serão comparados e classificados.

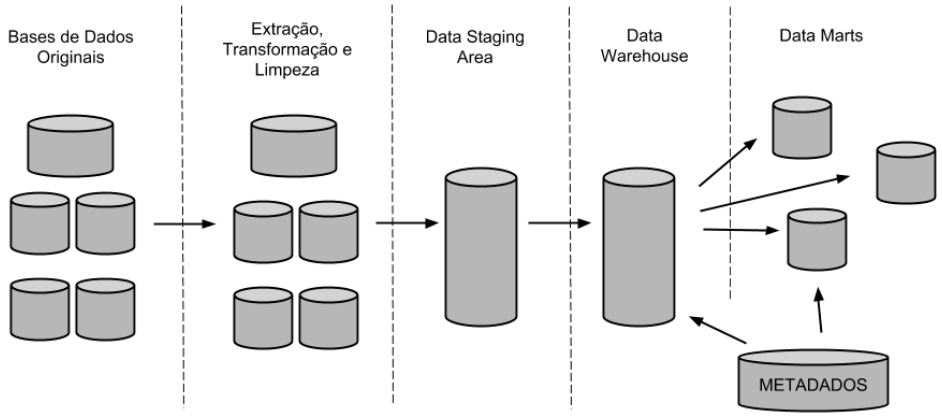


Figura 4.2 Arquitetura padrão de um ambiente de DW

Existem diversas abordagens para se estruturar um ambiente warehouse e ao longo do tempo, diversas arquiteturas foram surgindo. A Figura 4.2 demonstra a arquitetura padrão que se baseia na extração, transoformação e carga dos dados provenientes do ambiente operacional e de fontes externas para uma área de *staging*, seguidos da construção do DW. Os *data marts* são seguramente consistentes, uma vez que são construídos com base nos dados mantidos pelo DW.

Proporcionando uma visão mais detalhada do fluxo relacionado às etapas de ETL, A Figura 4.3 descreve as fases responsáveis por transformar o dado bruto em um conjunto de dados pré processados prontos para ser utilizado pela etapa de comparação.

4.3.1 A etapa de Extração e Merge

Um grande complicador, que impede que as bases de dados com uma quantidade muito grande de linhas e colunas sejam manipulados livremente através das ferramentas dos softwares comuns e a memória necessária RAM para ler os arquivos inteiros. A fim de se evitar sobrecargas de memória e operações desnecessárias, foi desenvolvido um módulo "*ExtractAndMerge*" para ser aplicado a todas as bases antes que seja realizado qualquer operação sobre elas.

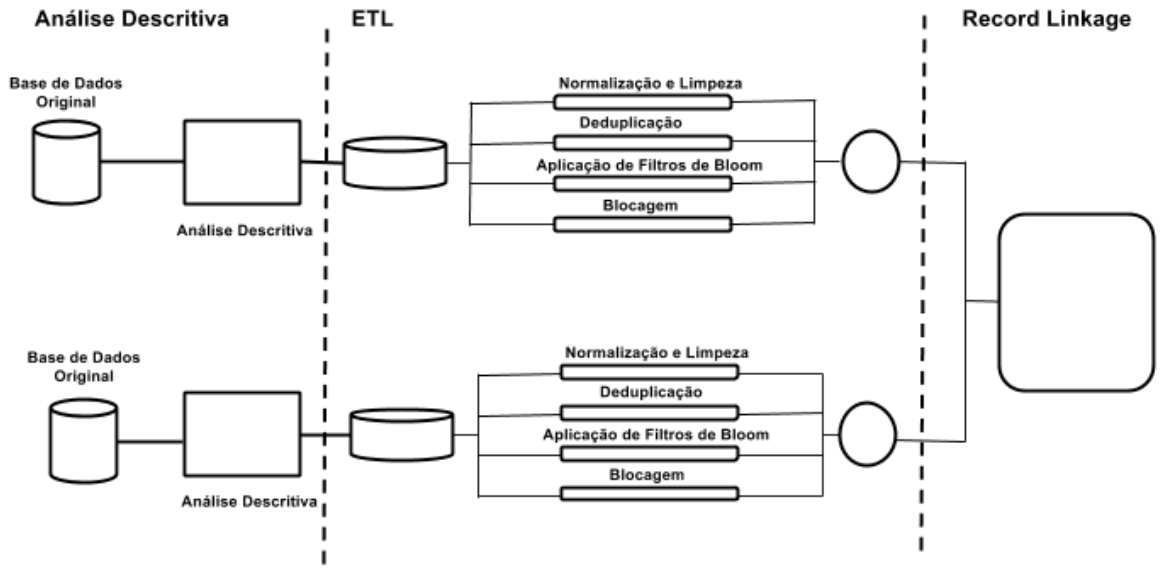


Figura 4.3 Fluxo da Etapa de Extração, Transformação e Carga

Este módulo tem três propósitos principais: Reduzir o tamanho dos arquivos manipulados, proporcionando um rápido acesso às informações requeridas; Organizar todos nados necessários em um único arquivo, pois muitas vezes as informações necessárias no processo da correlação probabilística (ou determinística) estão dispersas em mais de uma tabela; E por fim, gerar arquivos com o mesmo esquema, ou seja, mesma quantidade de colunas, mesmos atributos e mesmo padrão.

Esta fase de extração é fundamental para a administração do processo de anonimização e rgeração do *Data Mart*. Uma decisão importante está na associação de uma chave de recuperação, à cada registro. Esta chave pode ser um atributo único existente em cada tabela, como o Número de Identificação Social para as bases do CadÚnico ou o Número da Internação, para as bases do SIH. Entretanto, questionando novamente a segurança e preservação de privacidade recomenda-se que sejam geradas chaves numéricas aleatórias e únicas para cada registro armazenado.

4.3.2 Normalização e Limpeza

A etapa de limpeza e padronização utiliza como entrada um arquivo contendo a base de dados em formato bruto. Não podemos perder de vista que o primeiro passo para a geração desses dados passa pelo processo humano, na entrevista ou coleta. É claro que este processo está sujeito a falhas, pois muitas vezes a maior preocupação do profissional não está na qualidade dos dados. Um exemplo disto está na frequência com que o número "999999999-99"é encontrado em Data Warehouses para representar CPFs ou na divergência na representação do atributo sexo ("M"e "F"; "H"e "M"; "1"e "2"). Por isto, é esperado que exista muito lixo e inconsistência compondo os valores dos atributos além de representações diferentes para uma mesma informação.

Algoritmo 1: Rotinas de Pré Processamento

```
[1]  Function principal()
[2]      arq_RDD = transformaEmRDD(caminho_do_arquivo)
[3]      registros = arquivo_RDD.map(PadronizaRegistros).map(Bloom)
[4]  End
[5]  Function PadronizaRegistro(registro)
[6]      chave = registro[coluna_da_chave]
[7]      munic = registro[coluna_do_municipio]
[8]      nome = registro[coluna_do_nome]
[9]      dt_nasc = registro[coluna_da_data]
[10]     nome = padronizaNome(nome) dt_nasc = padronizaData(dt_nasc)
[11]     result = nome + dt_nasc + municipio + chave
[12]     return result
[13] End
[14] Function BloomAndBlocking(registro)
[15]     chave = registro[coluna_da_chave]
[16]     munic = registro[coluna_do_municipio]
[17]     nome = registro[coluna_do_nome]
[18]     dt_nasc = registro[coluna_da_data]
[19]     munic = registro[coluna_do_municipio]
[20]     bitsVector = getBloom(nome, tamN) + getBloom(dt_nasc, tamD)
[21]     outputFile = munic + ".bloom"p = OpenFile(outputFile)
[22]     p.Write(bitsVector + chave)
[23] End
```

Se os dados não forem tratados e transformados de maneira correta, nesta etapa, todo o fluxo das etapas subsequentes será comprometido uma vez que não será possível retornar aos valores corretos sem que o processo seja totalmente refeito, resultando em conclusões incorretas ou tomadas de decisões equivocadas. Pode-se dizer então, que o processo de limpeza (*data cleaning*) serve para corrigir as imperfeições existentes nas bases de dados transacionais, gerando dados corretos e informações mais acuradas.

A fase de pré-processamento dos dados realizada nas bases do CadÚnico, SIH e SIM (todos de 2011), tem a finalidade de extrair de todas as tabelas apenas as variáveis de interesse. Este é um modo de dispor de três arquivos com esquemas idênticos para que os módulos posteriores possam ser utilizados genericamente. Além disso, algumas bases possuem centenas de atributos e realizar operações sobre elas seria inútil, incorrendo apenas em consumo desnecessário de recursos. O pré-processamento realizado sobre os valores contidos nos atributos incluem:

1. Substituição dos valores ausentes: Todos os valores em branco existentes nas variáveis de interesse foram substituídos por valores simbólicos como "NOME AUSENTE", para o campo nome ou "99999999", para data de nascimento, respeitados os tipos e formatos.
2. Padronização no formato de datas: Todas as datas foram substituídas para o padrão "ANO,MES e DIA".
3. Padronização no formato do código de município: Existem algumas divergências entre a utilização do código IBGE com cinco ou seis dígitos. Todos os códigos de município foram transformados para códigos de cinco dígitos.
4. Remoção de caracteres especiais: Acentos e caracteres como hífen ou cedilha foram removidos em todas as variáveis de interesse. Os nomes foram todos convertidos para caixa alta e os espaçamentos extra foram removidos.
5. Substituição de nomes comuns: Sobrenomes muito comuns foram abreviados de modo que "CARLA DA SILVA DOS SANTOS", por exemplo, não apresente alta similaridade com "MARIA DA SILVA DOS SANTOS".

A partir destes procedimentos uma nova base, incluindo apenas as variáveis necessárias, é gerada contendo valores limpos e prontos para serem utilizados pelo processo de transformação para comparação dos campos. O Algoritmo 1 demonstra os passos executados pelas rotinas de pré-processamento, incluindo a blocagem, discutida na seção 4.3.3. Fica claro, porém que apenas as inconsistências estruturais são solucionadas por esta etapa. A nova base gerada também inclui uma chave de identificação que será exportada junto com o par na comparação. Através desta chave será possível recuperar todas as informações dos registros originais e colaborar na geração do *Data Mart*.

4.3.3 Blocagem

Considerando que $| A |$ e $| B |$ representam a quantidade de registros contidos nas bases A e B, respectivamente, a quantidade total de comparações necessárias no processo da

correlação de registros é | A | x | B |. Portanto a quantidade de pares candidatos crescerá de forma quadrática. Considerando a correlação CadÚnico x SIH, com a quantidade de registros descritos na seção 4.1.1, teríamos aproximadamente $7,076x10^{12}$ comparações. Entretanto, grande parte destas comparações são desnecessárias, porque geralmente em cada registro existem informações ou chaves capazes de criar agrupamentos que podem ser comparados entre si. Os métodos de blocagem tem a finalidade de diminuir a quantidade de pares candidatos através da utilização da chave para criação de subconjuntos de registros.

Um dos métodos mais utilizados é a Blocagem Padrão (*Standard Blocking*). Este método cria um bloco para cada chave única e agrupa todos os registros de mesma chave em bloco igual. As chaves são extraídas dos próprios atributos (como por exemplo sobrenome, ou ano de nascimento) e criadas na ordem sequencial em que aparecem. Uma chave de blocagem também pode ser composta por combinação de atributos, por exemplo sobrenome e idade (BAXTER; CHRISTEN; CHURCHES,).

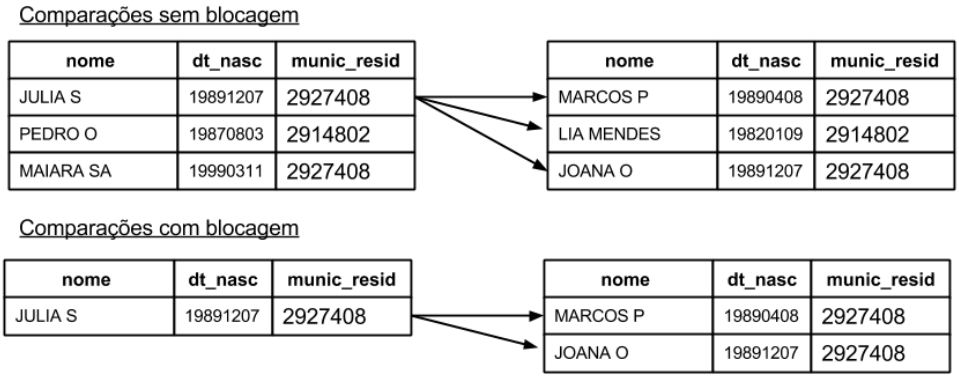


Figura 4.4 Blocagem por município para a chave 2927408

A blocagem utilizada por esta pesquisa utiliza o código do município de resiência para agrupar os registros. A vantagem é que esta chave cria blocos suficientemente pequenos reduzindo significativamente a quantidade de comparações. Cada vez que um registro é lido o valor de sua variável referente ao código do município de residência é selecionado. Se já existir algum bloco com aquela chave o registro é incluído no bloco existente, senão um novo bloco é criado. A Figura 4.4 demonstra como a estratégia de blocagem pode ser útil para reduzir as comparações. Se antes cada registro precisaria ser comparado com todos os registros da segunda base, agora ele é comparado apenas com quem compartilha de sua chave. Considerando então, que n é a quantidade de registros em duas bases pareadas e considerando também que o processo de correlação produz b blocos, todos do mesmo tamanho, incluindo n/b registros, pode-se concluir que o número total de comparações é $O\left(\frac{n^2}{b}\right)$. Uma análise ingênua considera que no Brasil existam 5570 municípios, portanto, utilizando o método de blocagem por município o problema prático da correlação CadÚnico x SIH se resumiria em aproximadamente $12,7x10^8$ comparações.

É importante deixar claro que a escolha da chave de blocagem não é trivial, pois existe um grande impasse envolvido. Por um lado, chaves que criam uma grande quantidade de

blocos produzirá grupos com poucos registros e consequentemente, poucas comparações precisarão ser realizadas. Contudo, a probabilidade de um registro ser incluído em um bloco incorreto é muito grande. O resultado perderá acurácia e a sensibilidade do método será comprometida. Por outro lado, a escolha de chaves pouco discriminatórias, como sexo ou UF da residência, produzirá poucos grupos de tamanho muito grande, exigindo ainda comparações supérfluas e incorrendo em tempo de execução muito alto.

4.3.4 Utilização dos Filtros de Bloom e Preservação da Privacidade

A utilização de bases de dados envolvendo informações pessoais dos indivíduos vem crescendo no meio científico e nas diversas pesquisas. Com isto, cresce também a responsabilidade com questões éticas que dizem respeito à privacidade e à segurança dos dados. As bases de dados fornecidas pelo SUS, por exemplo, contém informações sensíveis como diagnósticos de doenças infecciosas, cujo acesso não pode ser facilitado de nenhuma forma a pessoas não autorizadas. Pensando nisto, surge a necessidade de se utilizar um método de correlação de registros que permita a preservação da privacidade mesmo durante a etapa de comparação, de modo que mesmo com verificações manuais não seja possível recuperar a informação que identifique o indivíduo.

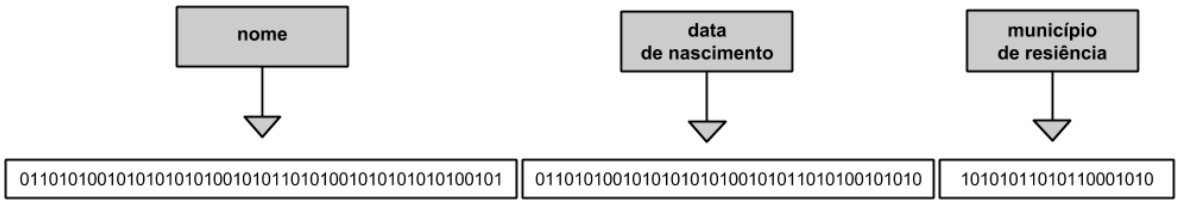


Figura 4.5 Versão do filtro de Bloom utilizado

O desafio primordial na correlação de registros é encontrar um método capaz de traduzir perfeitamente as pequenas diferenças existentes entre duas *strings* através da comparação de suas formas codificadas. A exatidão desta transformação fornecerá resultados mais acurados. Isto é o que chamamos correção do algoritmo. O protocolo de comparação de *string* desenvolvido por este trabalho codifica o texto em seu formato inteligível, em um vetor de bits através dos filtros de Bloom (BLOOM, 1970). A versão utilizada, como pode ser observada na Figura 4.5 destina a cada atributo, que no contexto das execuções apresentadas aqui são bigramas, uma faixa específica do filtro de modo que um elemento em um atributo não influencie a faixa do filtro destinada a outro atributo. Para mapear cada bigrama em posições específicas em sua faixa do vetor, k funções hash foram utilizadas. Isto permite que as comparações de cada um desses campo se dê de forma mais individual possível.

Os detalhes teóricos da construção do filtro estão descritos na seção 2.3.4. Interessa aqui, saber que o protocolo desenvolvido gera filtros para cada campo (*field*) de acordo com o tamanho n do filtro e da quantidade k de funções hash especificadas. Nestas execuções foram utilizadas duas funções hash, considerando que o tamanho do filtro é o menor capaz de manter a eficiência do método. O Algoritmo 2 demonstra como funciona o

procedimento que transforma os campos (texto plano) em um vetor de bits. Os elementos existentes na base que se deseja parear são submetidos obrigatoriamente, a este mesmo procedimento.

Algoritmo 2: Preenchimento do Filtro de Bloom

```
[1]  preencheFiltro(field, n, k)
[2]  x = obtemProximoBigrama(field)
[3]  while x == !vazio do
[4]      for j:1...k do
[5]          i ← h_j(x);
[6]          if B_i == 0then
[7]              B_i← 1;
[8]          end
[9]      end
[10] end
[11] x = obtemProximoBigrama(field)
[12] return B
[13]
```

O resultado desta etapa consiste na inserção de cada registro, representando os três atributos que serão comparados, nos arquivos correspondentes a cada bloco: "cadUnicoMunicipioX.bloom". "sihMunicipioX.bloom"e "simMunicipioX.bloom". Todas as execuções acontecem no ambiente distribuído do Spark, por isto não há como recuperar a posição de cada registro no arquivo original. Logo, faz-se necessário que cada registro codificado leve consigo também uma chave de recuperação. Finalmente, estes arquivos em formato bloom ficarão disponíveis para serem utilizados pelas rotinas de processamento que realizará as comparações e classificação dos pares.

4.3.4.1 Atribução dos Pesos Duas ocorrências que caracterizam um mesmo indivíduo, em bases distintas, pode conter diversos campos identificadores comuns, úteis no processo de correlação. Como demonstrado anteriormente, a seleção das variáveis que serão utilizadas na etapa da correlação de registros, estão sujeitas à análise da qualidade dos dados. A inclusão de mais variáveis, de forma indiscriminada, não implica necessariamente em um resultado mais acurado além de comprometer o desempenho das etapas de comparação. É importante ressaltar também que as variáveis comuns, capazes de identificar uma pessoa, tem relevância diferente no processo de decisão. Por exemplo, a variável sexo pode existir em duas bases de dados, entretanto, comumente, esta informação discrimina os registros em apenas dois conjuntos. Por outro lado uma variável como município de nascimento é bastante específica, e portanto, mais discriminatória.

Levando isto em conta, é comum se estabelecer pesos diferentes para cada atributo participante da correlação como uma estratégia de medição da contribuição de cada campo. Por exemplo, suponha que dois registros, em bases de dados distintas, tenham nome “MARIA APARECIDA DE SOUZA”, o fato de estas duas pessoas residirem em

mesmo município não é tão determinante para classificar o pareamento como verdadeiro. Por outro lado, o fato delas possuírem mesma data de nascimento deve ter uma influência muito maior no processo de decisão. O método da correlação probabilística de registros deve ser suficiente para contornar o problema da existência de erros (ou da ausência) de campos inteiros para a comparação. Levando isto em conta, o módulo desenvolvido neste trabalho considera a importância de cada campo individualmente. Desse modo os erros existentes no município de residência, por exemplo, têm um impacto muito menor do que a mesma proporção de erro na data de nascimento ou no nome do indivíduo.

Além da constatação de que os pesos devem ser proporcionais à importância do atributo no processo de decisão, outro fato que deve ser considerado diz respeito ao tamanho ideal do que o filtro de Bloom deve ter para ser suficientemente representativo. A escolha do tamanho ideal para cada campo no filtro, não é uma tarefa trivial e, nesse contexto, existe um grande impasse entre acurácia e tempo de execução, pois se por um lado um filtro muito grande diminuiria as taxas de erro do método, aumentaria significamente o tempo do processo de transformação e comparação. Sabendo que existe um tamanho de filtro ideal para cada atributo, é necessário então, atribuir tamanhos diferentes para cada atributo, no filtro, mantendo as proporções e respeitando as características dos valores contidos em cada um deles. Levando isto em conta, foi realizada uma sequência de testes empíricos em bases de dados conhecidas e pré-definidas, a fim de se estabelecer o peso e tamanho ideal para cada atributo depois da transformação.

Peso de Cada Atributo no Filtro	**Nenhuma Correspondência Esperada**		**Cinco Correspondências Exatas Esperadas**		**Cinco Correspondências Esperadas com um Caracter Incorreto**	
	Pares Esperados	**Pares Encontrados**	**Pares Esperados**	**Pares Encontrados**	**Pares Esperados**	**Pares Encontrados**
20x20x20	0	310	5	347	5	348
30x30x30	0	29	5	41	5	42
40x40x40	0	11	5	17	5	16
50x50x50	0	0	5	5	5	5
50x50x40	0	0	5	5	5	5
50x40x40	0	0	5	5	5	5
50x40x30	0	0	5	5	5	5
50x30x30	0	2	5	6	5	6

Tabela 4.5 Comparação dos pesos atribuídos aos campos

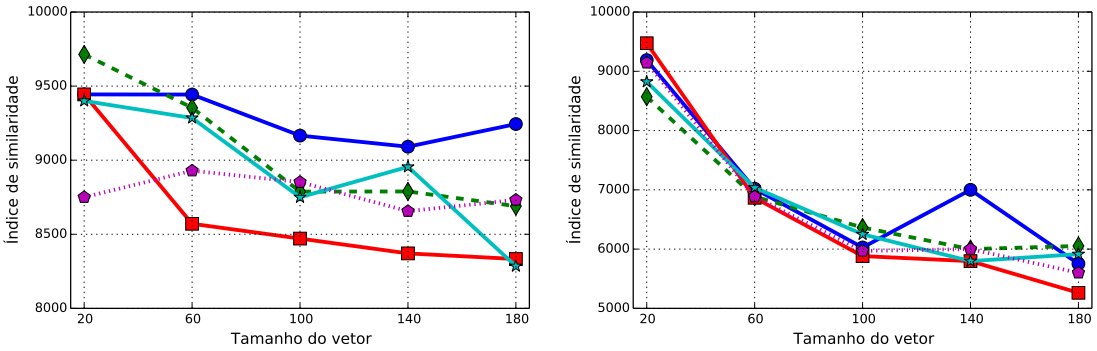
Para a observação e validação da acurácia e do filtro de bloom, foram construídas duas bases de dados com 50 e 20 registros, contendo nome, município de residência e data de nascimento. Para a primeira execução, nenhum registro continha valores iguais ou parecidos de forma que caracterizasse uma mesma entidade. Logo, o resultado perfeito deveria mostrar nenhuma correspondência verdadeira. O teste seguinte incluiu cinco registros idênticos, que deveriam ser pareados de forma exata. O último teste acrescentou a estes mesmos cinco registros idênticos nas duas bases, erros de adição e remoção de caracteres bem como erros de transposição entre dois caracteres vizinhos. O propósito é combinar tamanhos diferentes para cada atributo para encontrar a acurácia ótima em um tamanho suficiente do vetor de bits que represente bem a transformação do campo e mantenha as proporções dos pesos. Este estudo mostra, como esperado, que vetores

muito pequenos não são suficientemente representativos pois a probabilidade de que uma posição de bit seja influenciada por dois bigramas diferentes é muito maior. Fica claro que a utilização de um vetor muito pequeno, neste caso, seria o principal responsável pela grande quantidade de falsos positivos que seriam gerados. O vetor de tamanho 50-40-20 dentre os testados apresentou melhores resultados tanto observado a representatividade do vetor de bits em relação ao valor original quanto em relação aos pesos escolhidos.

4.3.4.2 Testes de Similaridade Uma vez que todos os vetores de bits são gerados para todos os registros existentes em cada bloco, o processo de comparação atribui uma pontuação entre cada par candidato. Essa pontuação, dada pelo índice de *Dice* 4.1, também conhecido como Soresen-Dice, é o fator utilizado na decisão de classificação, de modo que h se refere aos 1's em comum entre os dois vetores, a se refere aos 1's no primeiro vetor e b, 1's no segundo vetor. Aqueles pares que obtiverem índice de similaridade acima do ponto de corte serão colocados no grupo M = *Matching* e aqueles que estiverem abaixo do ponto de corte, serão colocados no grupo U = *Non-Matching*.

$$(\frac{2 * h}{|a + b|}) \tag{4.1}$$

Obviamente, a classificação dos pares nos dois grupos M e U está sujeita à análise estatística do ponto de corte ideal. Em outras palavras é preciso observar tanto a representatividade do filtro, no que diz respeito a capacidade do vetor de traduzir com precisão o texto plano da forma mais correta possível, quanto a qualidade das bases de dados envolvidas, pois o ponto de corte utilizado para bases com excelente qualidade não deve ser o mesmo para bases sujas derivadas de um processo descuidado de preenchimento. A constatação de que o tamanho do filtro de Bloom influencia diretamente na geração de falsos positivos é um assunto levantado por (SCHNELL; BACHTELER; REIHER, 2009) e foi comprovado na prática com os testes apresentados na seção 4.3.4.1 .



(a) Variação do Índice de Similaridade Para correlação com dois Erros.

(b) Variação do Índice de Similaridade Para correlação com seis Erros.

Figura 4.6 Variação do índice de similaridade.

O gráfico da Figura 4.6(a) demonstra a variação do índice de similaridade quando comparamos registros diferentes, todos com dois erros de substituição ou seja, os caracte-

res correspondentes são trocados por caracteres não correspondentes. É possível observar que filtros muito pequenos tendem a ter índice de similaridade mais altos. Isso se deve às coincidências nas posições mapeadas para bigramas diferentes. Quanto maior o filtro, mais representativo ele é. Essa diferença fica ainda mais evidente quando observamos o gráfico da Figura 4.6(b) em que todos os cinco registros agora, possuem erro de seis caracteres, indicando possível dúvida ou certeza de não-correspondência. A Figura 4.6 mostra que quando um filtro é muito pequeno, registros diferentes são erroneamente incluídos no grupo M (falsos positivos). Para este teste, cinco registros contendo apenas nome foi utilizado.

4.3.5 Deduplicação

O termo deduplicação é utilizado para descrever o processo de eliminação de registros que se repetem em uma mesma base de dados. Em uma base de dados a repetição de um registro por pode ocorrer por erro do sistema de informação ou pelo controle de um evento diferente. Quando esta ocorrência é esperada, as informações fornecidas pelo sistema permitem que se elimine todas as réplicas mantendo apenas um registro. Entretanto, a identificação do registro repetido, muitas vezes, é um processo dificultoso. Considere, por exemplo, que um cadastro seja realizado a nível municipal (como acontece no CadÚnico). É possível que alguma pessoa ao migrar de município seja novamente e equivocadamente cadastrada. A presença de réplicas pode trazer diversos prejuízos como a degradação do desempenho das fases de pré-processamento e comparação. O processo de deduplicação é uma etapa importante no processo de correlação de registros por dois motivos principais: Pode reduzir significativamente a quantidade de pares candidatos e consequentemente, reduzir o tempo total da execução e colabora na geraçao de um grupo confiável de verdadeiros pareados de modo que exista apenas um par verdadeiro.

As réplicas existentes tanto no CadÚnico quanto no SIH, nas versões utilizadas por esta pesquisa, possuem réplicas previstas que puderam ser eliminadas através de operações simples utilizando as próprias variáveis de identificação contidas nas bases de dados. Para as bases mais poluídas, em que não existam variáveis de controle de réplicas, métodos mais elaborados precisam ser utilizados. A deduplicação também é um método de correspondência probabilística que procura em uma mesma base registros que se refiram a uma mesma entidade. Neste cenário, a correlação pode ser elaborada de modo muito mais confiável, levando-se em conta que mais chaves podem ser selecionadas. Como é possível observar na Figura 4.7, o processo de deduplicação exige menos comparações do que o processo de correlação entre duas bases diferentes, resultando em um total de (n 1) x (n 2)/2 comparações. (SANTOS, 2008) alerta para o fato de que não existe muita referência, na literatura, que aborde o conceito de paralelização para a deduplicação de registros. O Algoritmo 3 demonstra o funcionamento da rotina de deduplicação que pode ser utilizada na limpeza das bases de dados mais poluídas.

4.3.6 Comparações e Recuperação da Informação

Ainda que o objetivo deste trabalho esteja no estudo e desenvolvimento dos métodos de organização e pré-processamento das bases de dados, é de extrema importância expor a

Algoritmo 3: Procedimento de Deduplicação

```
[1]  Function principal()
[2]      arquivo_RDD = transformaEmRDD(caminho_do_arquivo)
[3]      arquivo_B_RDD =
         transformaEmVariavelCompartilhada(arquivo_RDD)
[4]      result = arquivo_RDD.mapPartitionsWithIndex(compare)
[5]  End
[6]  Function compareDistribuido(index, bloco)
[7]      foreach linha_arquivo1 in bloco do
[8]          linha_arquivo2 = retornaLinhaInicial(index)
[9]          while linha_arquivo2 != vazio do
[10]             dice = verificaSimilaridade(linha_arquivo2, linha)
[11]             if dice > ponto_de_corte then
[12]                 elimina registro
[13]             end
[14]             linha_arquivo2 = getNextLine()
[15]         end
[16]     end
[17] End
```

metodologia geral das etapas de comparações.

De fato, o maior desafio envolvido nesta pesquisa está no gerenciamento e manipulação da enorme quantidade de informações contidas em todas as bases de dados utilizadas. Como pode ser observado pela Figura 4.8, a etapa de comparação observa todos os pares possíveis contidos em cada bloco e atribui a cada par uma pontuação que equivale à similaridade entre eles. No exemplo da Figura, foi utilizado o texto plano contendo o nome para tornar mais simples a observação das diferenças entre cada *string*. Vale ressaltar que as comparações reais ocorrem entre os vetores de bits codificados através de filtros de Bloom. A Figura demonstra também o índice de similaridade do *Dice* sendo aplicado aos registros do Bloco1. A utilização de um produto sobre resultado do cálculo que gera o índice Dice faz com que 10.000 seja o valor máxima similaridade. Considerando que o número de pares candidatos cresce de forma quadrática, é fácil constatar que a etapa de pré-processamento precisa se preocupar em reduzir o tempo de execução exigido pelos comparadores de string. Assumimos a comparação do município de residência um modo de correspondência exata realizada durante a blocagem, portanto a comparação entre os registros neste módulo está relacionada à probabilidade de concordância ou discordância quando comparamos o nome completo e data de nascimento do indivíduo.

Como foi exposto na seção 4.3.4.1, o estudo do ponto de corte ideal envolve uma análise da taxa de verdadeiros e falsos positivos incluídos nos grupos selecionados. Uma vez que a decisão pelo ponto de corte é feita, todos os registros acima dele são considerados uma correspondência verdadeira. É possível que, acima do ponto de corte, um mesmo registro seja pareado a dois ou mais registros. Sabendo que a relação entre os registros é unívoca,

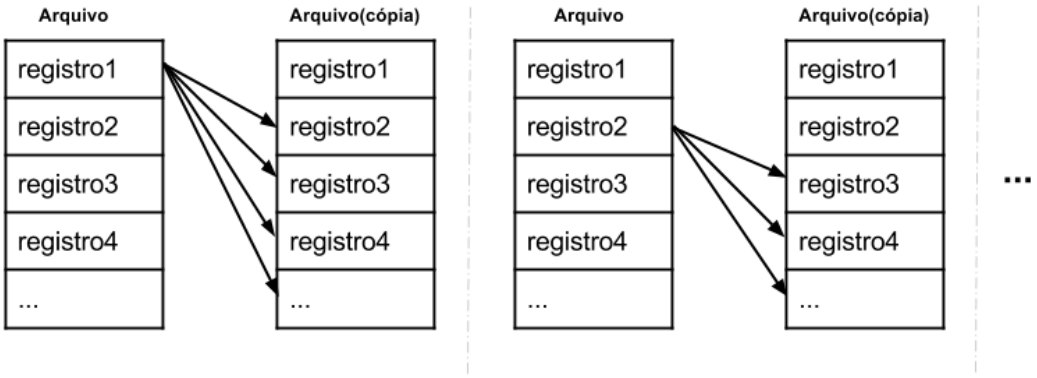


Figura 4.7 Esquema das comparações na deduplicação

é mantido no conjunto apenas o par cujo índice de similaridade seja o maior. Uma vez que os grupos M e U estejam completos uma nova etapa é inicializada a fim de se recuperar todas as informações relevantes existentes nas duas bases de dados. Nessa fase, a base que correlaciona os pares verdadeiros contém os registros anonimizados através dos filtros de Bloom e uma chave de recuperação diferente associada a cada componente do par. A rotina de recuperação, por último, é responsável por buscar a informação original nas duas bases, através das chaves existentes no par. A disposição das chaves ao longo das tabelas permite recuperar apenas as informações úteis para uma finalidade específica e aumentar ou modificar as informações conforme haja necessidade. A Figura 4.9 mostra um exemplo da estrutura do Data Mart gerado. As variáveis que compõem a nova base de dados podem ser incluídas através das chaves. Este DataMart gerado será, enfim, utilizado pelos pesquisadores estatísticos e epidemiólogos para suas investigações.

A grande vantagem das soluções apresentadas por este trabalho está no tratamento dado aos arquivos que contém uma grande quantidade de registros. O ambiente distribuído Spark permite que todas as operações, tanto das etapas de pré-processamento quanto da comparação e classificação, sejam executas utilizando os benefícios da computação paralela e distribuída. Os algoritmos desenvolvidos podem ser facilmente escalados para bases de dados ainda maiores sem que haja modificações conceituais. Além disso o Spark provê uma camada de tolerância a falhas indispensável quando se utiliza aplicações que executam durante muitas horas ou em hardware não confiável.

A abstração do objeto RDD é utilizada para gerenciar as funções *map* através do arquivo menor (o SIH, neste caso) enquanto que a o arquivo maior torna-se visível por todas as operações paralelas em diversas funções através da sua transformação em variável compartilhada do tipo *broadcast*. Este tipo de variável pode ser utilizado sempre que for necessário manter uma variável em memória para todos os nós que executam tarefas. Deste modo o custo com a transferência de dados, e a sobrecarga que este processo implica, são reduzidos. A cada vez que uma ação é executada sobre um objeto RDD, seu valor é recomputado. Entretanto, aqui foi utilizado o método *cache* para fazer com que um RDD persista na memória de modo que os próximos acessos aos elementos sejam muito mais rápidos. Em cenários em que se precisa realizar operações várias vezes sobre um

PARES	DICE
MARIA DE LURDES GUIMARAES - MARIA LURDES GUIMARAES	9831.0
MARIA DE LURDES GUIMARAES - MARIA SANTANA GUIMARAES	9000.0
MARIA DE LURDES GUIMARAES - CAMILA O DOS S	5660.0
JOSE A A - MARIA LURDES GUIMARAES	3810.0
JOSE A A - MARIA SANTANA GUIMARAES	3256.0
JOSE A A - CAMILA O DOS S	4444.0
FELIPE DOS S P - MARIA LURDES GUIMARAES	4898.0
FELIPE DOS S P - MARIA SANTANA GUIMARAES	5600.0
FELIPE DOS S P - CAMILA O DOS S	6512.0

Figura 4.8 Exemplo da comparação dos pares de cada bloco

mesmo elemento em um RDD (como é o caso da comparação na correlação de registros), esta pode representar uma grande vantagem. A compatibilidade do Spark com linguagens de programação como Python, também colaborou para a redução do tempo na curva de aprendizado. Para utilizar o modelo de programação Spark para Python foi empregada a API PySpark, cuja exigência é a utilização da versão 2.6 ou superior para o Python.

4.4 AVALIAÇÃO DOS RESULTADOS

É indispensável avaliar o resultado da correlação no estudo de caso real, pois somente através disto é possível concluir a sobre a aplicabilidade e contribuição desta pesquisa, bem como sobre a qualidade dos métodos desenvolvidos. No CadÚnico a deduplicação realizada considerou apenas os registros que possuem Número de Identificação Social válido. Para isto foi utilizada a variável que representa o estado cadastral da família para excluir os cadastros que constavam como "validando NIS", "aguardando NIS", "excluído", "sem registro civil"e "em cadastramento". A base do SIH, por sua vez permite que uma mesma pessoa seja inserida no sistema mais de uma vez, como no caso de renovação da internação. Neste caso foram mantidos sempre as ocorrências mais recentes de cada registro duplicado. Os principais parâmetros utilizados na aplicação dos módulos elaborados, nas bases reais foram: 1) A seleção das bases de dados, 2) A seleção dos atributos utilizados nas comparações, 3) As definições das blocagens 4) Os critérios de transformação do filtro de Bloom 5) O ponto de corte 5) A classificação e recuperação das informações.

Os atributos utilizados na comparação foram *nome completo*, *data de nascimento* e

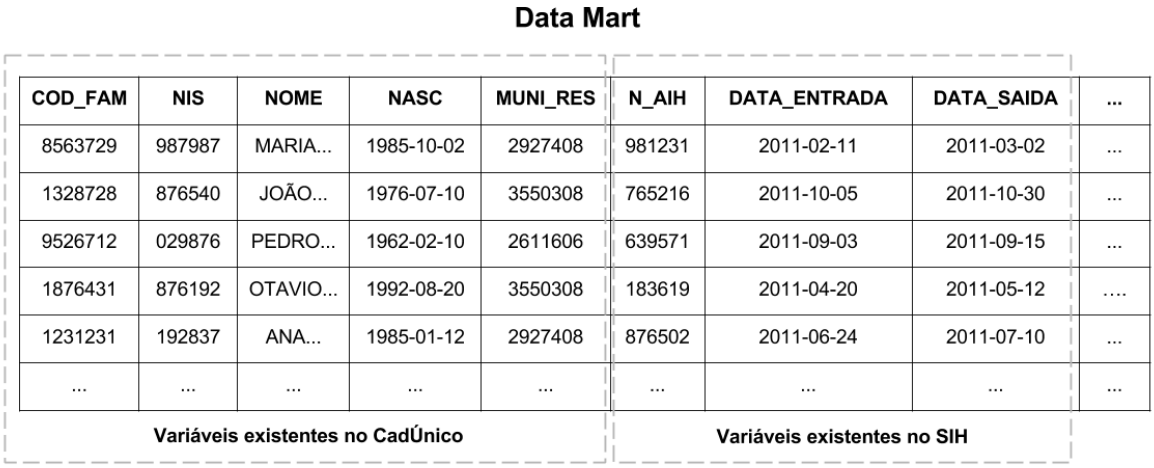


Figura 4.9 Exemplo da estrutura de um Data Mart

município de residência, sendo que este último pode ser entendido como um atributo de correlação exata, pois foi utilizado como chave na geração dos blocos. Um quarto atributo como o *nome da mãe* seria útil para melhorar a qualidade das comparações entretanto atributos com esta capacidade discriminatória não existem em comum nas duas bases. Foi utilizado a blocagem padrão por município, muito embora os módulos que executam as blocagens por predicado já fazem parte desta infraestrutura. Para as transformações, foi utilizado um vetor de 110 bits, sendo a distribuição 50-40-20 equivalentes ao nome, data de nascimento e municipio de residência.

Para a correlação CadÚnico x SIH de 2011, foi selecionado todos os pares com índice de similaridade superior a 8.700 para análise do ponto de corte. Todos os pares acima desta faixa são considerados *link*, ou seja, pares que são aceitos como prováveis pares positivos. Abaixo desta faixa, por sua vez estão os *non-link*, ou seja, pares que são aceitos como prováveis pares negativos. A avaliação utilizada para definir *match* e *non-match* é de natureza subjetiva e foi empregada por uma única pessoa. Todas as avaliações de acurácia ficam sob a responsabilidade de um grupo composto por estatísticos. A ausência de um *padrão ouro* na análise dos resultados torna este processo ainda mais crítico.

Dice	**Sensibilidade(%)**	**Especificidade (%)**	**Acurácia (%)**
> 8800	100,0	51,0	52,0
> 9000	100,0	53,8	57,0
> 9500	100,0	89,3	94,0
> 9800	99,0	98,0	98,5
> 9850	97,0	99,0	98,0

Tabela 4.6 Sensibilidade, especificidade e acurácia para diferentes faixas de similaridade

A sensibilidade pode ser interpretada como a fração dos verdadeiros positivos do total

de pares incluídos no grupo dos "positivos"que contem também falsos negativos, enquanto que a especificidade está relacionada à fração dos verdadeiros negativos do total de pares incluídos no grupo dos "negativos", que contem também falsos negativos. Sendo assim estas duas medidas podem ser dadas pelas equações 4.2 e 4.3.

$$Sensibilidade = (\frac{VP}{|VP + FN|}) \tag{4.2}$$

$$Especificidade = (\frac{VN}{|VN + FP|}) \tag{4.3}$$

O resultado apresentado na Tabela 4.6, demonstra a análise do estudo da acurácia obtida por esta correlação, considerando sensibilidade e especificidade. Levando-se em conta a inviabilidade de se analisar individualmente todos os pares gerados pela etapa de correspondência (mesmo aqueles que têm Dice superior a 8.700), foram geradas amostras de cada faixa do índice de similaridade, através das quais a análise manual foi viabilizada. O melhor resultado possível seria aquele com 100% de acurácia. Para isto, os resultados deveriam apresentar sensibilidade e especificidade máximas. Entretanto, não existem métodos de correlação probabilística que garanta acurácia máxima, pois isto está diretamente relacionado com a qualidade do preenchimento das variáveis. O melhor resultado da aplicação deste método está na faixa cujo índice de similaridade é maior que 9800. Se utilizarmos este valor como ponto de corte, temos acurácia de 98,5%. Apesar disto, faz-se necessário aplicar outros métodos que resgatem os pares verdadeiros que ficaram fora desta faixa.

	CadÚnico	**SIH**
Tamanho Aproximado (em linhas)	87 milhões	61 mil
Padronização, Anonimização e Blocagem	2310.4 s	36.5 s
Comparações	9,03 horas	
Recuperação dos Registros	1,31 horas	

Tabela 4.7 Tempo de execução de cada etapa relacionada à correlação de registros

A solução distribuída do Spark foi a base do processamento utilizada em todas as fases deste trabalho. Sua vantagem pôde ser observada também na execução de tarefas *in-mememory*, evitando o armazenamento intermediário do disco. A Tabela 4.7 mostra o tempo de execução para todas as etapas da correlação de registros executando em ambiente Spark para a correlação realizada entre o CadÚnico e o SIH.

O hardware principal utilizado na execução de todas as etapas da correlação de registros é um *cluster* com 8 processadores Intel Xeon E74820, 16 cores, 126 GB de RAM e armazenamento contendo discos de 10 TB conectados por protocolo NFS. Além do cluster, dispusemos de uma estação de trabalho para testes, com processador i5, 4 GB de RAM e 300 GB de disco.

Capítulo 5

CONSIDERAÇÕES FINAIS

5.1 RESULTADOS E DISCUSSÕES

A pesquisa apresentada nesta Dissertação corresponde a um pequeno módudo do cenário geral em que ela está incluída. O objetivo do projeto principal envolve o desenvolvimento de uma plataforma completa que permita realizar correlação de registros entre diversas bases de dados através de um portal, de modo que os próprios pesquisadores possam ajustar os parâmetros de acordo com a necessidade, utilizando as vantagens da computação em nuvem para ter acesso rápido e simplificado à uma infraestrutura capaz de prover computação eficiente e escalável. A Figura 5.1 demonstra todos os módulos envolvidos na concepção da plataforma em questão.

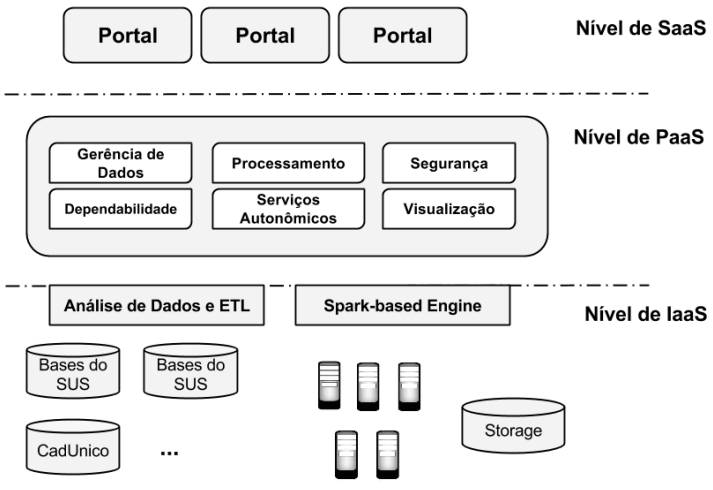


Figura 5.1 Módulos do Projeto de Integração de Dados e Correlação de Registros

O escopo deste trabalho está na organização e pré-processamento das bases de dados, o que compõe as primeiras fases de todo o processo que envolve o relacionamento de

registros. Todos os módulos desenvolvidos foram utilizados no relacionamento CadÚnico x SIH e CadÚnico x SIM, gerando como resultado *Data Marts* utilizados pela equipe que envolve epidemiologistas e estatísticos do Instituto de Saúde Coletiva da Universidade Federal da Bahia. A primeira estratégia de blocagem utilizada considerou agrupar os registros por UFs, entretanto a blocagem que utiliza como chave o código IBGE do município de residência gerou melhores resultados agrupando uma classe muito mais específica de indivíduos e reduzindo significativamente o número de pares candidatos e consequentemente o tempo gasto nas comparações.

Essa modalidade de blocagem foi uma estratégia bem sucedida para reduzir a complexidade das comparações. A escolha do município de residência como chave foi encorajada pela qualidade do preenchimento deste atributo nas bases do CadÚnico. Como este cadastro é de responsabilidade das prefeituras municipais a exportação para o sistema de informação leva consigo o código do município. Portanto, não há erros no preenchimento deste valor. Entretanto, podem existir erros de preenchimento do município nas bases onde esta informação é cedida pela própria pessoa. Este é o caso dos dados contidos no Sistema de Informação sobre Internações Hospitalares. É possível que um paciente ao se internar em outra cidade que não a sua, informe a cidade de internação como sendo sua cidade de residência. Para contornar este tipo de problema pode ser utilizado a estratégia alternativas de blocagem por predicados (HERNANDEZ; STOLFO, 1998) que considera a utilização de uma combinação de atributos para gerar as chaves que formarão os grupos. Desta forma, mesmo que um atributo não contenha o valor correto, a inserção deste registro naquele grupo pode ser salva por outro atributo que compõe a chave. Compreendendo a blocagem por predicados como uma disjunção de conjunções, um exemplo de predicado que pode ser empregado é $P = (nome \wedge município\_de\_residência) \vee (sobrenome \wedge ano\_de\_nascimento)$.

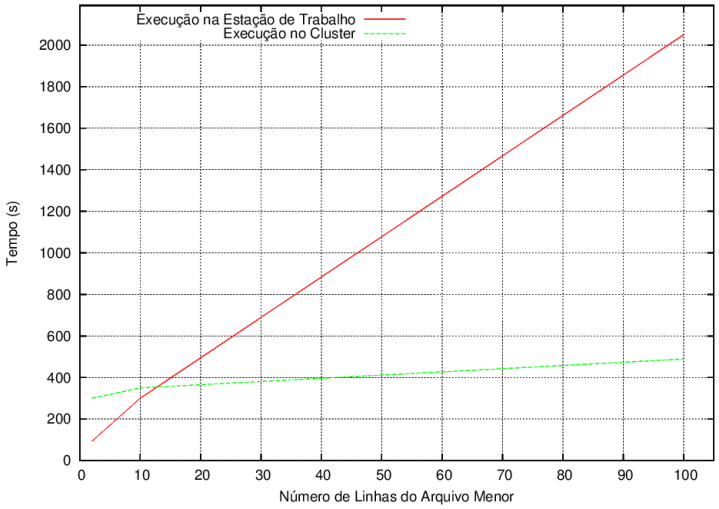


Figura 5.2 Desempenho do Spark em Arquivos de Tamanhos Diferentes

Importante destacar também que os módulos desenvolvidos nesta pesquisa, executados sobre a plataforma distrubuída do *Spark*, apresentam bons resultados quando executados

sobre arquivos muito grandes, que contém uma grande quantidade de registros. Apesar do desempenho da ferramenta, na manipulação de bases de dados pequenas não ter sido cuidadosamente avaliado é possível constatar que este cenário implica em uma degradação muito maior para a aplicação, gerada pelo escalonador de tarefas, como pode ser observado na Figura 5.2. Uma rotina de correlação foi executada sobre uma base de dados de 1 milhão de registros e outra que varia em quantidade de linhas, como expresso no gráfico. A execução distribuída no cluster executa em tempo ainda maior que a execução em uma máquina simples, para arquivos pequenos. Isto pode ser explicado pelo tempo necessário para criar e alocar os *workers* e gerenciar a distribuição dos processos e dos dados na criação dos *splits* nos arquivos de entrada. Desta forma, não existem garantias de que os tempos de resposta para bases pequenas sejam ainda satisfatórios.

5.2 LIMITAÇÕES E DESAFIOS

Apesar dos testes apresentados neste trabalho envolverem apenas duas bases, muito tempo foi dedicado também, na organização e compreensão das outras que não foram apresentadas. Uma etapa fundamental foi entender a estrutura, as características e o significado de cada uma delas. Alguns dicionários de variáveis foram facilmente adquiridos, enquanto que outros, para as bases do SUS principalmente, precisaram ser construídos através de investigação e observação. As diferentes versões das bases do CadÚnico também exigiram tempo de investigação em relação à composição das tabelas, distribuição das informações e criação de scripts de exportação.

Outro ponto de contratempo foram as barreiras de hardware, principalmente no que diz respeito à armazenamento. Os arquivos que contém as bases de dados em sua versão original não tratada, exigem um espaço considerável de armazenamento (cerca de 500 GB). Conforme as os módulos que compõem este arquivo vão sendo extraídos para utilização, uma quantidade muito maior de espaço em disco é exigida. Por sua vez, os módulos que realizam o pré-processamento, blocagem e a correlação dos registros produzem ainda uma quantidade de arquivos muito grande. Portanto, é importante dedicar atenção à esta necessidade a fim de se evitar percalços.

5.3 TRABALHOS FUTUROS

O trabalho apresentado por esta Dissertação está inserido em um projeto maior que se encontra ainda em fase inicial. Portanto, existem muitas questões que estão incluídas no escopo de trabalhos futuros. Em relação à qualidade do processo de comparação, o próximo passo a ser implantado é a utilização de outras variáveis para compor as chaves de comparações. A variável "sexo"será utilizada através de correspondência exata, enquanto que variáveis como "nome da mãe"serão utilizadas em bases nas quais elas coexistam.

A estratégia de blocagem por predicados será utilizada com a finalidade de se cobrir a maioria absoluta dos pares reais ao mesmo tempo em que restringe a quantidade de pares candidatos. Existem implementações em fase de projeto que consideram a utilização do município de residência combinado com o ano de nascimento, bem como o sobrenome. A utilização de código fonético (BuscaBR ou Metaphone) na geração das chaves de blocagem

tem o objetivo de evitar que os erros de preenchuimento sejam responsáveis pela inclusão daquele registro em um bloco incorreto.

Além disto, existem algumas decisões de implementação que podem aumentar significativamente a qualidade do filtro de Bloom gerado e consequentemente reduzir a taxa de falsos positivos. A primeira resolução diz respeito à quantidade de funções *hash* utilizadas. Como foi declarado anteriormente, cada função *hash* implica na modificação de uma posição diferente para cada bigrama. A utilização de uma quantidade maior de funções implicará, portanto, em uma probabilidade menor de que bigramas diferentes influenciem posições iguais. Contudo, a decisão de se aumentar a quantidade de funções hash, vem acompanhada da necessidade de se aumentar também o tamanho do vetor de bits utilizado para que o filtro não perca sua representabilidade em relação ao texto plano. Uma vez que mais posições são influenciadas por um bigrama, a quantidade de posições vazias (ou preenchidas com "0") diminui, justificando a necessidade de se aumentar o tamanho do filtro.

Para se ter um controle maior sobre a representação do vetor de bits e de cada campo do texto plano transformado em código, serão utilizados componentes ainda menores na transformação e comparação dos registros. Um nome, por exemplo, pode ser dividido em nome, nome do meio e sobrenome, a fim de se efetuar comparações individuais em cada um destes campos. Seguindo a sugestão de (SCHNELL; BACHTELER; REIHER, 2009), serão utilizados, nos testes futuros, filtros de 500 bits e 15 funções *hash* para cada uma das unidades de comparações existentes nos campos. Os novos resultados serão comparados com os resultados apresentados por este trabalho, em termos de sensibilidade e especificidade.

Um outro ponto de ajuste que pode ser considerado nos trabalhos futuros, diz respeito ao tratamento adequado que deve ser dado à faixa cinzenta, conforme pode ser verficado na da Figura 2.6. Na maioria das vezes existe uma faixa, expressa no índice de similaridade, na qual é extremamente difícil classificar os pares como verdadeiros ou falsos. É preciso, portanto, utilizar métodos de refinamento que sejam mais cuidadosos nas comparações dos pares existentes neste grupo. Uma estratégia que pode ser adotada é se utilizar um segundo método de compraração de strings como códigos fonéticos, para se obter um resultado mais confiável e que seja útil na decisão correta de correlação.

Finalmente, pretende-se incluir a este trabalho suporte a banco de dados relacionais de modo a facilitar o gerenciamento das informações antes e depois de realizada a correlação e, consequentemente, facilitar a geração dos Data Marts através das exportações das chaves.

5.4 CONCLUSÕES

Este trabalho apresentou a aplicação de técnicas de ETL, com relação às etapas de pré-processamento de bases de dados, tendo como principal objetivo o aumento da a eficiência das fases da correlação probabilística de registros. Além disto, esta pesquisa envolveu um estudo integral das ferramentas e dos métodos de correlação e principalmente de transformação e comparação de campos. Com a preocupação da preservação da privacidade, foram utilizados métodos que garantiram a total anonimização dos valores identificado-

res dos indivíduos, de modo que uma terceira parte pudesse realizar a comparação dos registros ao mesmo tempo em que fossem mantidas as políticas de segurança.

A solução desenvolvida foi arranjada em módulos, cada um com sua função específica. Os módulos seguem uma ordem de execução que compreende desde a organização das bases de dados, transformação dos registros, até a exportação dos pares candidatos para os módulos que realizam a comparação. A ordem e a execução individual destes, permite que cada passo seja aplicado, omitido ou substituído. De mesmo modo, a estrutura das implementações permite que todos os parâmetros sejam facilmente reconfigurados conforme a necessidade. Os módulos desenvoldidos por esta pesquisa têm o propósito princial de servir às etapas posteriores da correlação de resgistros e por isto, são facilmente integradas aos outros módulos desenvolvidos em conjunto.

Os algoritmos foram desenvolvidos utilizando a API do Spark para a linguagem Python. A abstração de RDD é utilizada para se manipular os arquivos, tornando a aplicação facilmente escalável para bases maiores uma vez que as tarefas são realocadas e a divisão dos dados reformulada de acordo com os parâmetros de entrada e características do hardware. A plataforma distribuída do Spark foi fundamental para possibilitar o tratamento de bases de dados grandes, como as utilizadas nestes testes. Todas as implementações apresentadas por este trabalho foram aplicadas na plataforma do *Spark*, incluindo também, aplicações auxiliares não apresentadas, como sort, merge, inserção de chaves de recuperação, entre outros.

Com o objetivo de que compartilhar com a comunidade científica os progressos alcançados durante todo o tempo de dedicação a esta pesquisa, algum esforço foi empregado na elaboração de textos científicos. O resultado disto foi a publicação do trabalho apresentado nesta dissertação, na conferência internacional EDBT/ICDT (*Extending Database Technology*) (EDBT/ICDT, 2015) 2015.

A aplicação das técnicas apresentadas neste trabalho apresentou resultados satisfatórios, tanto do ponto de vista da compatibilidade em relação ao tempo de resposta gerado pelas operações executadas sobre bases do tamanho do CadÚnico (mais de 100 milhões de registros), quanto pelos resultados das comparações dos registros aproximados. Os resultados apresentados são uma iniciativa inovadora na área de correlação de registros em bases de saúde, no Brasil, pois ainda não existem muitos trabalhos que contextualizam esta necessidade no cenário de Big Data, considerando aspectos como distribuição das tarefas e paralelização de todos os processos que envolvem o problema. O êxito obtidos nos resultados apresentados neste trabalho e as propostas dos trabalhos futuros, apontam para aplicões mais promissoras que poderão servir diversas contextos e domínios, colaborando nas tomadas de decisão para órgãos governamentais ou não.

REFERÊNCIAS BIBLIOGRÁFICAS

ANJOS, J. C. S. dos et al. O 4º paradigma e a computação intensiva de dados. 2013.

BAXTER, R.; CHRISTEN, P.; CHURCHES, T. In: . [S.l.: s.n.].

BLOOM, B. H. Space/time trade-offs in hash coding with allowable errors. 1970.

BORTHAKUR, D. Hdfs architecture guide. *HADOOP APACHE PROJECT http://hadoop. apache. org/common/docs/current/hdfs design. pdf*, 2008.

CAVALCANTI, G.; FELL, A.; DORNELAS, J. Data warehouse: uma ferramenta de tecnologia de informação para as organizações. 2005.

CHURCHES, T.; CHRISTEN, P. Some methods for blindfolded record linkage. 2013.

CLIFTON, C. et al. Privacy-preserving data integration and sharing. In: *Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. New York, NY, USA: ACM, 2004. (DMKD '04), p. 19–26. ISBN 1-58113-908-X. Disponível em: ⟨http://doi.acm.org/10.1145/1008694.1008698⟩.

COSTA, A. *Aspectos de Criação e Carga de um Ambiente de Data Warehouse*. Dissertação (Mestrado) — Universidade Federal do Rio De Janeiro – UFRJ, 2001.

DATASUS. 2015. ⟨http://datasus.saude.gov.br/⟩. Accessed: 2015-02-07.

DEAN, J.; GHEMAWAT, S. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, ACM, v. 51, n. 1, p. 107–113, 2008.

DOAN, A.; HALEVY, A.; IVES, Z. *Principles of Data Integration*. Morgan Kaufmann, 2012. (Morgan Kaufmann). ISBN 9780124160446. Disponível em: ⟨http://books.google.com.br/books?id=5Rg679tjhFQC⟩.

DOAN, A.; HALEVY, A.; ZACHARY, I. *Principles of Data Integration*. [S.l.]: Morgan Kaufmann, 2012. ISBN 0124160441.

DURHAM, E. et al. Quantifying the correctness, computational complexity, and security of privacy-preserving string comparators for record linkage. *Inf. Fusion*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 13, n. 4, p. 245–259, out. 2012. ISSN 1566-2535. Disponível em: ⟨http://dx.doi.org/10.1016/j.inffus.2011.04.004⟩.

EDBT/ICDT. 2015. ⟨http://http://edbticdt2015.be/⟩. Accessed: 2015-02-07.

FELLEGI, I. P.; SUNTER, A. B. A theory for record linkage. *Journal of the American Statistical Association*, v. 64, p. 1183–1210, 1969.

GARTNER. 2015. ⟨http://www.gartner.com/⟩. Accessed: 2015-02-07.

GILL, L.; STATISTICS, G. B. O. for N. Book. *Methods for automatic record matching and linkage and their use in national statistics*. [S.l.]: London : National Statistics, 2001. Bibliography: p. 139 - 151.

GOOGLE Inc. 2015. ⟨http://google.com/about/company/⟩. Accessed: 2015-02-07.

GUARALDO, C. N.; REIS, F. R. Contagem da frequência dos bigramas em palavras de quatro a seis letras do português brasileiro. 2009.

HERNANDEZ, M. A.; STOLFO, S. J. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Min. Knowl. Discov.*, Kluwer Academic Publishers, Hingham, MA, USA, v. 2, n. 1, p. 9–37, jan. 1998. ISSN 1384-5810. Disponível em: ⟨http://dx.doi.org/10.1023/A:1009761603038⟩.

HEY, T.; TANSLEY, S.; TOLLE, K. *O Quarto Paradigma - Descobertas Científicas Na Era da Escience*. [S.l.]: OFICINA DE TEXTOS, 2011. ISBN 9788579750281.

INMON, W. H. *Building the Data Warehouse*. New York, NY, USA: John Wiley & Sons, Inc., 1992. ISBN 0471569607.

JARO, M. Advances in record-linkage methodology as applied to matching the 1985census of tampa, florida. 1989.

KIRSCH, A.; MITZENMACHER, M. Less hashing, same performance: Building a better bloom filter. *Random Struct. Algorithms*, John Wiley & Sons, Inc., New York, NY, USA, v. 33, n. 2, p. 187–218, set. 2008. ISSN 1042-9832. Disponível em: ⟨http://dx.doi.org/10.1002/rsa.v33:2⟩.

LIMA-COSTA, M. F.; BARRETO, S. M. Tipos de estudos epidemiolÃ: conceitos bÃ¡sicos e aplicaÃ§Ãna Ã¡rea do envelhecimento. *Epidemiologia e ServiÃde SaÃ*, scielo, v. 12, p. 189 – 201, 12 2003. ISSN 1679-4974. Disponível em: ⟨http://scielo.iec.pa.gov.br/scielo.php?script=sci_arttext&pid=S1679-49742003000400003&nrm=iso⟩.

MDS. *Cadastro Único para Programas Sociais do Governo Federal*. 2014. [Online; accessed 12-december-2014]. Disponível em: ⟨http://www.mds.gov.br/bolsafamilia/cadastrounico⟩.

NUNO, A. *Extracção Eficiente de Padrões Textuais Utilizando Algoritmos e Estruturas de Dados Avançadas*. Dissertação (Mestrado) — Universidade Nova de Lisboa/Faculdade de Ciências e Tecnologia, 2002.

PBF. 2015. ⟨http://www.mds.gov.br/bolsafamilia⟩. Accessed: 2015-02-07.

RANDALL, S. M.; FERRANTE, A. M.; SEMMENS, J. The effect of data cleaning on record linkage quality. *BMC Medical Informatics and Decision Making*, v. 13, 06 2013.

ROMERO, J. *Utilizando o Relacionamento de Bases de Dados para Avaliação de Políticas Públicas: Uma Aplicação para o Programa Bolsa Família*. Dissertação (Doutorado) — UFMG/Cedeplar, 2008.

SANTOS, W. *Um algoritmo paralelo e eficiente para o problema de pareamento de dados*. Dissertação (Mestrado) — Universidade Federal de Minas Gerias, 2008.

SCHNELL, R.; BACHTELER, T.; REIHER, J. Privacy-preserving record linkage using bloom filters. *BMC Medical Informatics and Decision Making*, v. 9, p. 41, 2009.

SILVA, A. M. *Big Data Versus Data Warehouse type @ONLINE*. 2012. Disponível em: ⟨http://angmaximo.wordpress.com/2012/11/28/big-data-data-warehouse/⟩.

SINGH, H. *Interactive Data Warehousing*. Prentice Hall PTR, 1999. ISBN 9780130803719. Disponível em: ⟨http://books.google.com.br/books?id=jksqAQAAMAAJ⟩.

SRIVASTAVA, J.; CHEN, P.-Y. Warehouse creation-a potential roadblock to data warehousing. *IEEE Trans. on Knowl. and Data Eng.*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 11, n. 1, p. 118–126, jan. 1999. ISSN 1041-4347. Disponível em: ⟨http://dx.doi.org/10.1109/69.755620⟩.

WHITE, T. *Hadoop: The Definitive Guide*. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2004. ISBN 0596521979, 9780596521974.

WINKLER, W. E. Matching and record linkage. 2014.

ZAHARIA, M. et al. Spark: cluster computing with working sets. In: *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. [S.l.: s.n.], 2010. p. 10–10.