Estudo do *framework* Hadoop aplicado em um ambiente de *Big Data*

Sérgio Rafael Hautrive Righi[1], Gustavo Stangherlin Cantarelli[1]

[1]Ciência da Computação - Centro Universitário Franciscano

Santa Maria - RS - Brasil

{sergiohautrive, gus.cant}@gmail.com

Abstract. *Daily big companies produce and store a huge amount of data, but there was no way that these companies were able to use this data in a timely manner to carry out decision-making. For this and other reasons that companies like Google Inc. and Yahoo! Inc. have developed some of the technology as MapReduce and Hadoop, respectively, in order to make it possible for these data often valuable to be processed simply and quickly . This paper will discuss concepts such as Big Data, MapReduce and Hadoop with the aim of studying the ways in which the Hadoop framework contributes in Big Data environments. And to validate a tool will be developed to show the functioning of these technologies and how they work together. At the end contributions were found showing that Hadoop is a framework that seeks to increase transparency in development through its subprojects that provide support in carrying out tasks.*

Resumo. *Diariamente grandes empresas produzem e armazenam uma enorme quantidade de dados, porém não havia uma maneira com que essas empresas conseguissem utilizar esses dados, em tempo hábil, para realizar tomadas de decisões. É por esse e outros motivos que empresas como a Google Inc. e a Yahoo! Inc. desenvolveram algumas das tecnologias como, MapReduce e Hadoop, respectivamente, com objetivo de tornar possível que esses dados, muitas vezes valiosos sejam processados de maneira simples e rápida. Neste trabalho foram abordados conceitos como os de Big Data, MapReduce* e *Hadoop com o objetivo de estudar de que maneiras o framework Hadoop contribui em ambientes de Big Data. E para sua validação será desenvolvida uma ferramenta para mostrar o funcionamento dessas tecnologias e como elas trabalham em conjunto. Ao final foram encontradas contribuições que mostram que o Hadoop é um framework que busca tornar mais transparente o desenvolvimento através de seus subprojetos que fornecem suporte na realização de tarefas.*

1 Introdução

A quantidade de dados que são gerados a cada dia está cada vez maior. Com base nisto, e segundo [IBM, 2013], a Google Inc. processa mais de vinte e quatro *petabytes* de dados diariamente, enquanto o Facebook, recebe aproximadamente dez milhões de novas fotos a cada hora. Observa-se também que diariamente no Facebook, seus usuários clicam no botão “Curtir” ou postam comentários pelo menos três bilhões de vezes. Dado esse aumento constante no volume de dados armazenados, que cresce a cada dia, devem ser

desenvolvidas técnicas para que o processamento destas informações ocorra em um tempo aceitável.

Foi com o objetivo de tornar o processamento de grandes volumes de dados mais rápido, e de maneira simples, que o MapReduce foi criado. Fazendo o uso dos benefícios do processamento distribuído, o MapReduce possibilita que seja feita a divisão de um grande problema em tarefas menores que são processadas individualmente.

Baseado neste modelo de programação a Yahoo! Inc., em 2005, tomou a iniciativa de desenvolver um *framework*, o Hadoop (que logo após foi incorporado ao projeto Apache, permitindo que outros desenvolvedores contribuam para a sua melhoria). A criação do Hadoop foi embasada no MapReduce criado pela Google Inc. e teve como objetivo tornar o desenvolvimento de sistemas para processamento de grandes volumes de dados (*Big Data*) de forma paralela e distribuída mais fácil para o programador.

Este trabalho foi desenvolvido com o intuito realizar um estudo sobre o *framework* Hadoop e mostrar o funcionamento das tecnologias citadas acima e como elas interagem entre si. Para isto serão utilizadas algumas ferramentas, são elas: as *Application Programming Interface* (APIs) JavaScript do Google para criação de mapas e gráficos e uma máquina virtual rodando a distribuição Hadoop Hortonworks. Também será utilizado um conjunto grande de dados obtidos em um portal de dados abertos do governo brasileiro [PBDA, 2015] onde nele estão disponibilizados dados sobre serviços sociais públicos que o governo realiza, como informações sobre saúde, educação, economia, segurança e outros.

A partir das ferramentas apresentadas acima e para que fosse possível realizar a validação do trabalho (mostrar o uso do *framework* Hadoop) proposto, foi necessário o desenvolvimento de um sistema simples que teve como base o mapa do Brasil, onde ao interagirem com ele é possível visualizar as informações retiradas da base de dados, de maneira rápida e intuitiva. Estes dados estão divididos em regiões e estados e apresentam informações (referente a edição da prova escolhida para este trabalho, de 2012) como média de alunos inscritos, média de idade dos alunos, média de notas por área (ciências da natureza, ciências humanas, linguagens e códigos e matemática), entre outros.

Nas próximas seções serão abordados, de forma mais aprofundada, os temas que fazem parte do estudo teórico necessário para o trabalho, bem como trabalhos relacionados, entre outros tópicos que servirão para melhorar a compreensão do que será desenvolvido.

2 MapReduce

O MapReduce é um modelo de programação e um *framework*, proposto pela Google Inc., para processamento paralelo de grandes volumes de dados, podendo ser aplicado a diversas tarefas do mundo real [Dean e Ghemawat, 2008].

Este modelo de programação é utilizado em problemas que podem ser fragmentados em partes menores, sendo essas pequenas partes, processadas em paralelo e de forma

abstraída para o programador, tendo ele somente o trabalho de implementar as funções *Map* e *Reduce*, deixando a cargo do *framework* gerenciar como serão distribuídas as tarefas.

De modo geral, a ideia do MapReduce consiste em dividir e processar um conjunto grande de dados, sendo isto realizado pelas funções *Map* e *Reduce*. A função *Map* utiliza as partes dos arquivos como entrada, efetua o processamento paralelo, e gera como saída as tuplas formadas pelos campos (chave, valor). A função *Reduce* é responsável por fornecer o resultado final, agrupando as tuplas geradas pela etapa anterior que possuírem valores iguais.

A distribuição é gerenciada pelo *framework*, que utiliza arquivos distribuídos e trocas de mensagens para realizar a execução. O processamento é dividido em três etapas: mapeamento, agrupamento e redução. Na primeira etapa é feita a divisão dos blocos de entrada em funções *Map*, onde os dados recebidos como entrada são processados e transformados em novas tuplas (chave, valor). Na segunda etapa os dados são agrupados de acordo com suas chaves e enviados para etapa de redução. Nesta etapa cada função *Reduce* recebe uma tupla (chave, valor), executa uma função pré-definida pelo usuário e gera como saída uma lista de tuplas (chave, valor) [Andrade, T., 2012]. Na Tabela 1, é mostrado alguns dos benefícios da utilização do MapReduce.

Tabela 1. Benefícios da utilização do MapReduce [Hortonworks, 2015]

Benefício	**Descrição**
Simplicidade	Pode ser desenvolvido em diferentes linguagens como: Java, C++ ou Python e *jobs* (programas) MapReduce são fáceis de serem executados.
Escalabilidade	Pode processar grandes quantidades de dados, podendo esses dados, estarem armazenados em Hadoop *Distributed File System* (HDFS) ou em um em *cluster* (conjunto de computadores que trabalham em conjunto para processar uma tarefa).
Velocidade	O processamento paralelo faz com que problemas que levavam dias para serem resolvidos, sejam solucionados em horas.
Recuperação	Por ser um sistema com tolerância a falhas, se uma máquina não está disponível, o sistema procura outra que possui o mesmo par (chave, valor).
Movimentação de dados mínima	Move os processos sobre o HDFS e não o contrário, onde tarefas de processamento ocorrem no nó onde os dados estão, reduzindo as operações de E/S e contribuindo para o melhor processamento do Hadoop.

3 Hadoop

Sua origem veio a partir do Apache *Nutch*, um projeto de motor de buscas, mantido pela Apache Software *Foundation*, que fazia parte de um projeto de biblioteca de indexação de páginas chamado Lucene, ambos desenvolvidos pelos mesmos criadores do Hadoop. O Apache Nutch possui um sistema de arquivos baseado no *Google File System* (GFS), o *Nutch Distributed File System* (NDFS) que tornava possível o tratamento de grandes

volumes de páginas e, pouco tempo depois também já possuía o MapReduce incorporado ao sistema. Visto que esses projetos poderiam ser utilizados também para outros fins, o Hadoop foi criado abrangendo características de ambos os projetos e seu sistema de arquivos foi chamado de Hadoop *Distributed File System* (HDFS) [Alecrim, 2015].

A biblioteca de software Apache Hadoop é um *framework* (embasado no MapReduce) que permite o processamento distribuído de grandes conjuntos de dados através de *clusters* de computadores usando modelos de programação simples [Apache Hadoop, 2015].

É um projeto de código aberto implementado em Java, criado inicialmente pela Yahoo! Inc. e atualmente mantido pela Apache Software *Foundation*, com o objetivo de oferecer uma solução para problemas de *Big Data.* É formado, principalmente, por duas partes: o Hadoop *Distributed File System* (HDFS), que é um sistema de arquivos distribuído e confiável, responsável pelo armazenamento de dados, e o Hadoop MapReduce, responsável pela análise e processamento dos dados. Na Figura 1 é mostrado, de forma abstrata, as etapas de funcionamento do Hadoop onde ocorrem as operações de entrada, mapeamento, ordenação e agregação, redução e por fim são gerados os arquivos de saída resultantes das operações que foram executadas.

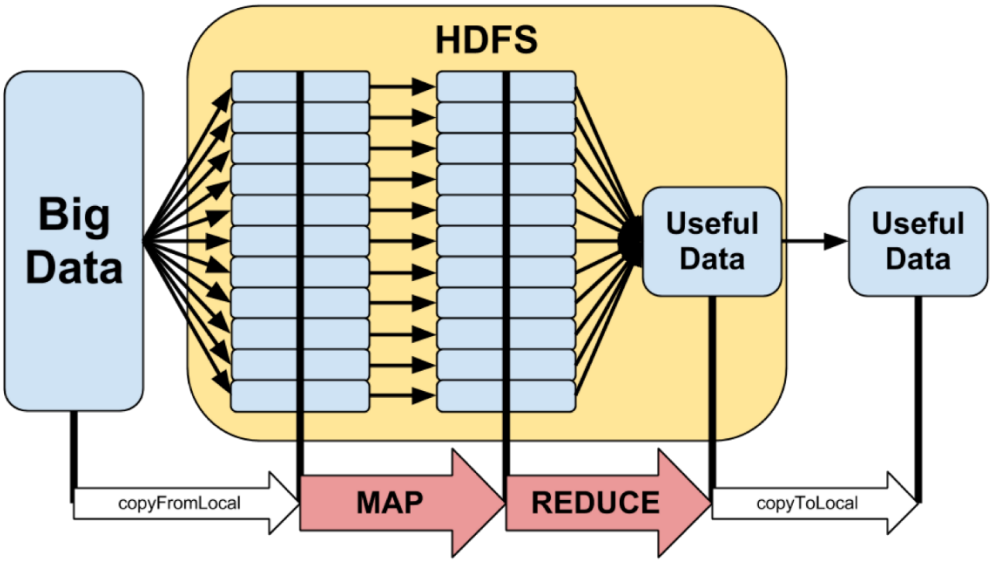

Figura 1. Funcionamento do Hadoop MapReduce [Lockwood G., 2015].

Também existem outros subprojetos que fornecem abstração de maior nível, facilitando o desenvolvimento [Apache Hadoop, 2015a] que são:

- **Ambari:** Visa tornar a gestão do *cluster* Hadoop mais simples através de uma interface web intuitiva, que possibilita o monitoramento e configuração dos recursos do *cluster* de forma simples e rápida.

- **HBase:** É um banco de dados, não relacional (NoSQL) e orientada a colunas, que fornece acesso em tempo real de leitura e escrita a grandes conjuntos de dados (bilhões de linhas e milhões de colunas). Como características estão o fato

de prover mecanismos de tolerância a falhas, fácil integração com aplicações utilizando APIs e rapidez. [Apache HBase, 2015]

- **HCatalog:** É uma camada de gerenciamento que fornece abstração ao usuário apresentando em uma visão relacional os dados armazenados dentro do HDFS, garantindo que o usuário não precise se preocupar sobre onde e em que formato estão os dados [Apache HCatalog, 2015].

- **Hive:** É um *data warehouse* distribuído, que facilita a utilização de grandes volumes de dados (*datasets*) em ambientes de armazenamento distribuído. Possui uma linguagem de consulta muito parecida com a SQL, chamada de HiveQL, que tem o objetivo de facilitar a estruturação e consulta dos dados.

- **Hue:** É uma interface web que permite a utilização do ambiente Hadoop diretamente no navegador, podendo ser associada a qualquer distribuição do Hadoop [Hue, 2014].

- **Pig:** Uma plataforma para grandes volumes de dados que possui uma linguagem de programação de alto nível para análise de dados. Também possui um compilador que transforma aplicações nela desenvolvidas em uma sequência de programas MapReduce.

- **Spark:** É um mecanismo para processamento de grandes volumes de dados em memória, possui APIs em diversas linguagens de programação, podendo ser utilizado em diversas aplicações do mundo real como, aprendizagem de máquina ou cargas de trabalho SQL para acesso rápido a conjuntos de dados iterativos [Apache Spark, 2015].

- **Sqoop:** É um serviço que permite que uma base de dados armazenada em um banco relacional seja transferida para dentro do Hadoop de forma simples e rápida. No Quadro 1 é mostrado o comando, para PostgreSQL, utilizado para importar a base relacional para o Hadoop. (versão do *driver* utilizado para conexão, 9.2-1002).

Quadro 1. Comando para importação da base de dados relacional utilizando o Apache Sqoop

```
sqoop import --connect jdbc:postgresql://endereco:porta/nome_do_banco
             --username nome_do_usuario
             --password senha_do_usuario
             --table nome_da_tabela
             -- --schema nome_do_esquema
```

- **Tez:** Este projeto é um serviço que visa construir uma estrutura complexa, grafo dirigido sem ciclo, de tarefas para processamento de dados construído sobre o Hadoop YARN, com o objetivo de dar mais desempenho na realização de consultas.

- **Zookeeper:** É um serviço de alto desempenho centralizado que mantém informações de configurações do grupo, com o objetivo de coordenar e proporcionando a sincronização para aplicações distribuídas.

É a partir destas tecnologias que o Hadoop fornece ao programador um alto nível de abstração, tornando desnecessário que o mesmo possua conhecimento aprofundado sobre sistemas distribuídos [Gasparotto M. H., 2014]. Segundo os analistas Gualtieri e Yuhanna (2014), acredita-se que o Hadoop seja uma plataforma flexível de gestão de dados voltada para grandes empresas que possuam diferentes dados estruturados, não estruturados e binários [Gualtieri e Yuhanna, 2014]. O *framework* Hadoop, atualmente na versão 2.7.1 (versão estável 2.7.0), é formado pelos seguintes módulos: Hadoop *Common*, Hadoop *Distributed File System*, Hadoop YARN e Hadoop MapReduce [Apache Hadoop, 2015a].

3.1 Hadoop *Common*

É um conjunto de utilitários e bibliotecas que dão suporte aos demais módulos do Hadoop. Sendo um dos módulos mais importantes do sistema, também chamado de Hadoop *Core*, da mesma forma que os demais módulos, tem como premissa que falhas de hardware acontecem com frequência e por este motivo devem ser tratadas de forma automática, a nível de software, pelo *framework*.

3.2 Hadoop *Distributed File System*

Os arquivos HDFS são divididos em dois tipos de nós de armazenamento. Um *namenode* (nó mestre) e um ou mais *datanodes* (nós operários). O mestre comanda os demais nós e sabe onde cada bloco está armazenado. Os operários, armazenam e recuperam os dados conforme o nó mestre solicitar, também são responsáveis por enviar, periodicamente, a lista dos blocos armazenados ao nó mestre.

Sendo parte fundamental para o funcionamento do sistema, o nó mestre, deve possuir mecanismos que garantam a ele ser tolerante a falhas. Sendo assim, o Hadoop fornece essa garantia de duas maneiras. A primeira é efetuando *backups* dos arquivos para que, em caso de falhas, o sistema possa ser restaurado. E a segunda é utilizando um nó mestre secundário que realize a junção, periódica, entre a imagem do sistema de arquivos e o arquivo de *log* (armazena dados sobre edições realizadas), evitando que o *log* fique muito grande. Visto que o nó principal é praticamente igual ao secundário, em caso de falha, o nó secundário pode assumir seu lugar apenas copiando os *metadados* (informações sobre o conteúdo dos arquivos) do nó principal [Gasparotto M. H., 2014].

3.3 Hadoop YARN

O Hadoop YARN é um *framework* para gerenciamento dos recursos do *cluster*. Também chamado de MapReduce 2.0, tem como ideia fundamental separar as duas principais funções do *JobTracker*, gestão de recursos e monitoramento, em partes independentes. A ideia básica é ter o um *ResourceManager* (RM) para todos e um *ApplicationMaster* (AM) para cada *job* MapReduce. O *ResourceManager* é reponsável por distribuir os recursos no sistema e o *ApplicationMaster* fica com a tarefa de negociar os recursos com o

ResourceManager e trabalhar em conjunto com o *NodeManager* (monitor de recursos) para executar e monitorar as tarefas [Apache Hadoop, 2015b]. A Figura 2 apresenta a estrutura básica do Hadoop e seus subprojetos.

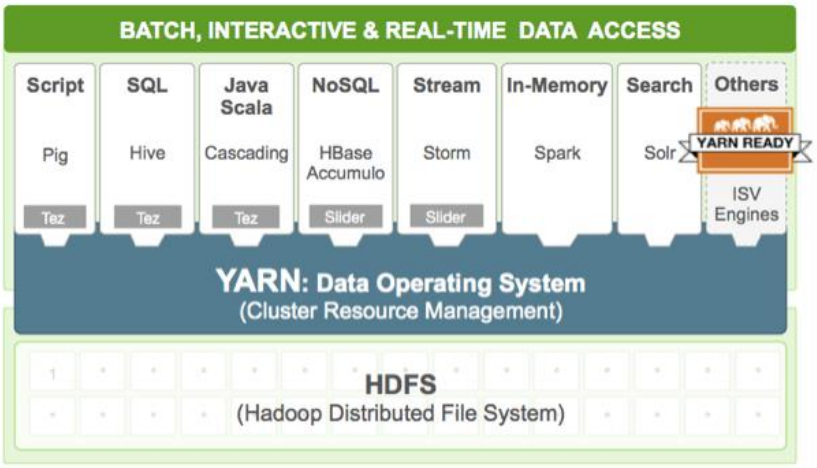

Figura 2 – Estrutura básica do *framework* Hadoop [YARN, 2015]

3.4 Hadoop MapReduce

Como descrito na seção 2 deste trabalho, o MapReduce é um modelo de programação e um framework para processamento paralelo, destinado, principalmente, para processamento de grandes volumes de dados através das funções *Map* e *Reduce*.

3.5 Distribuições Hadoop

Existem diversas distribuições do Hadoop (Cloudera, Hortonworks, MapR entre outras). A Cloudera é uma distribuição que utiliza muitos aspectos do Hadoop original, porém foram adicionadas melhorias como, por exemplo, o Cloudera *Manager* (aplicativo de gerenciamento, projetado para facilitar a administração centralizada dos dados [Cloudera, 2015]) e um motor SQL que roda dados relacionais no Hadoop, o Impala. Já a distribuição Hortonworks tem a distribuição do Hadoop mais próxima do original, tendo como meta construir um ambiente Hadoop e evoluir o código *open source*. Por último tem-se a distribuição MapR que por sua vez não é tão popular quando as outras citadas anteriormente, porém, caracteriza-se pela sua arquitetura e recursos de processamento. Outra característica é possuir recursos únicos como suporte a *Network File Systems* (NFS), uma plataforma para recuperação de desastres e alta disponibilidade de dados [CIO, 2014].

4 Big Data

Com o aumento constante do volume de dados gerados e que devem ser armazenados, o termo *Big Data* está cada vez mais presente em eventos sobre bancos de dados. O *Big Data* é um conceito utilizado para caracterizar uma grande quantidade de dados, que geralmente apresenta grandes dificuldades ao ser processado em um banco de dados convencional [Schneider, R. D. 2012], desta maneira, devem ser desenvolvidas soluções

que tornem possível armazenar, processar e posteriormente analisar esses dados em tempo hábil, a fim de auxiliar em tomadas de decisões a partir dos resultados obtidos.

Muitas vezes não é sabido que *Big Data* é um conceito presente em toda parte, onde na realização de tarefas cotidianas como, navegar na web, acessar redes sociais ou mesmo ao utilizar aplicativos, entre outras, são geradas quantidades muito grandes de informações. Essas informações possuem um grande valor e devem ser armazenadas e processadas em tempo hábil, de modo que este tipo de informação proporcione diversas possibilidades para quem as possui. Uma delas, e talvez a mais comum, é a possibilidade de apresentar conteúdos relevantes de acordo com o perfil de cada usuário, utilizado com frequência em empresas, principalmente da internet [Ianni, V., 2013].

Este conceito defende que ao utilizar um grande conjunto de dados para extrair informações, desde que em tempo de processamento aceitável, podem ser alcançados melhores resultados, que na maioria das vezes não seriam alcançados ao utilizar um volume menor.

Para tornar o conceito de Big Data mais claro, ele pode ser resumido utilizando cinco Vs: volume, velocidade, variedade, veracidade e valor [Alecrim, 2015].

- **Volume:** A quantidade de dados deve ser grande (depende do ambiente que esteja trabalhando), para realizar a extração de informações destes dados, que antes não eram aproveitados.
- **Velocidade:** O processamento destes dados deve ser feito em um tempo aceitável, pois dependendo da aplicação, não adianta nada resultados sobre informações desatualizada.
- **Variedade:** Os dados podem ser de diversos formatos diferentes, sendo eles estruturados (armazenados em bancos de dados relacionais) e não estruturados (documentos, imagens, entre outros). Podem, esses dados, terem sido obtidos de diferentes fontes (contanto que sejam fontes confiáveis).
- **Veracidade:** Os dados a serem analisados devem ser confiáveis, pois não adianta nada recuperar uma informação que não seja verdadeira, portanto, é necessário que os processos garantem a sua consistência.
- **Valor:** As informações que são retiradas deste conjunto de dados devem trazer benefícios, pois é desnecessário processar uma quantidade grande de dados se elas não forem úteis.

5 Trabalhos Relacionados

Nesta seção são apresentados objetivos e resultados de outros trabalhos que possuem assuntos que também serão abordados no presente trabalho.

5.1 Análise de Estratégias de Acesso a Grandes Volumes de Dados

No trabalho de Oliveira *et al*. (2014), foi investigado o efeito de alternativas de armazenamento e particionamento de grandes volumes de dados no desempenho de *dataflows* (fluxo de funcionamento ou como os dados se movem em uma aplicação). Em particular, observa-se que, através da análise do processo de *dataflows* em grandes volumes de dados, o modelo de paralelismo adotado pelo MapReduce, pode não produzir a melhor estratégia de execução. Os autores concluíram que atualmente existem diversas estratégias diferentes, cada uma possuindo vantagens e desvantagens, como desempenho, custo de processamento, entre outras. Com isso pôde ser observado que o custo com leitura e gravação de dados é muito relevante se for considerado ao custo total de processamento.

5.2 MapReduce - Conceitos e Aplicações

No trabalho de Andrade (2012), foram apresentados alguns conceitos e aplicações do MapReduce. Inicialmente conceitualizando sobre sua origem e o porquê de ser necessário que o modelo de programação seja utilizado para processamento de grandes volumes de dados. Ao longo do artigo também foi apresentado como é o funcionamento, *frameworks* (é um conjunto de conceitos usados para resolver um problema específico) e onde esse modelo pode ser aplicado. O autor Andrade concluiu que o MapReduce tem sido utilizado com sucesso para diferentes propósitos. E que este sucesso venha por diversos motivos, são eles: fácil utilização, transparência nos detalhes de paralelismo, fácil modelagem de problemas utilizando as funções *Map* e *Reduce* e por ser escalável para *clusters* computacionais.

5.3 Análise comparativa de ambientes e linguagens para computação intensiva de dados na nuvem

No trabalho de Dantas e Barreto (2013), foi realizada uma análise comparativa entre *frameworks* de processamento paralelo de grandes volumes de dados, realizada levando em consideração alguns tópicos como: comunicação entre processos, tolerância a falhas, desempenho, entre outros. O objetivo é que a partir dos pontos fortes e fracos seja possível criar um módulo de suporte baseado em *cloud computing* para atender aplicações de *data intensive* (aplicações que utilizam paralelismo para processamento de grandes volumes de dados). Os autores Dantas e Barreto concluíram que este estudo comparativo servirá de base para a concepção de um módulo de suporte ao desenvolvimento de aplicações *data intensive*.

5.4 *Big Data*: Transformando Dados em Decisões

No trabalho de Volpato *et al.* (2014), foram abordados conceitos de *Big Data* como, volume, variedade e velocidade em lidar com dados digitais, estruturados ou não, e como estes agregados com veracidade e valor podem ser importantes no processo de tomada de decisão. Após a realização do estudo os autores concluíram que, o *Big Data,* é muito importante, e que por meio dele, grandes organizações podem utilizar informações

armazenadas que antes não estavam sendo utilizadas para obterem resultados de qualidade que auxiliem a mesma no processo de tomada de decisão.

5.5 *Big Data*: Utilizando a Internet para Tomada de Decisões

No trabalho de Chatalov e Rufino (2013), foi apresentada uma descrição da ferramenta Hadoop, mostrar como se dá seu funcionamento e objetivo, utilizando revisões bibliográficas e sites para realizar o estudo da mesma. Também foi de interesse do trabalho estudar o termo *Big Data*, destacando sua importância e de que maneira o Hadoop pode extrair informações em grandes volumes de dados. Ao final da pesquisa realizada neste trabalho os autores Chatalov e Rufino concluíram que dado o crescente aumento nos dados gerados diariamente e que, em sua maioria não são aproveitados, é preciso desenvolver técnicas para realizar o processamento desses dados em tempo aceitável. É por isso que a utilização do Hadoop vem crescendo, pois torna possível a extração de informações importantes dessas grandes bases de dados de forma instantânea, analisando e filtrando para auxiliar na melhor tomada de decisão.

5.6 Conclusão dos Trabalhos Relacionados

Nos trabalhos apresentados nesta seção observou-se que, de forma geral, visam encontrar formas de manipular e gerenciar grandes volumes de dados com objetivo de encontrar a melhor solução para o Big Data utilizando diferentes técnicas e para diferentes situações. Também é abordado como a utilização do *framework* Hadoop vem crescendo cada vez mais pois torna a extração de informações, dessas grandes bases de dados, possível e em tempo aceitável, possibilitando que grandes organizações consigam assim, tomar as melhores decisões com base nos dados que elas possuem.

6 Metodologia

Nesta seção estarão descritas algumas das ferramentas como, linguagens de programação, Sistema Gerenciador de Banco de Dados (SGBD), tecnologias como, Hadoop, e o diagrama entidade relacionamento necessárias para realizar o desenvolvimento deste trabalho e como cada uma delas foi utilizada. Também será descrito o estudo de caso que foi aplicado este estudo.

Para o desenvolvimento da ferramenta que servirá para validar este trabalho serão utilizadas as linguagens de programação Javascript e *Hypertext Preprocessor* (PHP). A linguagem Javascript foi escolhida pela facilidade de incorporação de mapas e de gráficos por meio das *Application Programming Interface* (APIs) do Google (*Maps* e *Charts*). Com a utilização da API do Google *Maps* é possível incorporar os mapas e gráficos do Google em páginas web de maneira simples e rápida, visto que, fornece diversos utilitários para a manipulação dos mapas [Google *Developers*, 2013] e a API do Google *Charts* é uma ferramenta poderosa, simples de usar e gratuita [Google *Developers*, 2015] para a criação de gráficos, podendo ser customizada de maneira a adequar-se a cada necessidade. A linguagem PHP também foi utilizada e, com ela, foi possível realizar a incorporação com o Hive e realizar as consultas a partir dos filtros utilizando a biblioteca Javascript JQuery, que, juntamente com Ajax, possibilita realizar requisições assíncronas

ao servidor. Também foi utilizada uma biblioteca PHP, chamada ThriftSQL[1], sua utilização possibilitou a conexão entre a aplicação que foi desenvolvida e o *data warehouse* (Hive) do Hadoop.

Outro ponto importante a destacar é como o *framework* Hadoop foi utilizado neste trabalho. Como função principal, o Hadoop, realizou o processamento dos dados utilizando o modelo MapReduce e seus demais módulos de apoio. Mas antes de realizar o processamento, foi preciso importar a base de dados relacional para dentro do *framework*, e para isto, foi utilizado um serviço chamado Sqoop. Após a importação e para armazenamento dos dados dentro do Hive (que roda em cima do sistema de arquivos do Hadoop, o HDFS) foram utilizados aproximadamente seiscentos blocos de armazenamento.

A distribuição do Hadoop escolhida para o desenvolvimento deste trabalho foi a Hortonworks, versão 2.3. Os motivos que levam a está escolha foram o ambiente de gerenciamento, bastante documentação, é uma distribuição recente (2011), sendo assim não há tantos trabalhos a utilizando. Outro fator que levou a esta escolha foi por ser uma distribuição próxima do código original do Hadoop. Para sua utilização é necessário a criação de uma máquina virtual. Isto ocorre a partir da importação de um arquivo (no formato "Open Virtualization Format Distribution Package ou .ova") disponibilizado no site oficial da distribuição Hortonworks[2]. O arquivo é disponibilizado para dois softwares de virtualização, o VMware e o VirtualBox. Neste trabalho foi escolhido utilizar o VMware e esta escolha se deu por ser um software que oferece eficiência, fácil gerenciamento e configuração [VMware, 2015]. A configuração que da máquina virtual que foi utilizada está descrita na Tabela 2.

Tabela 2. Configurações da Máquina Virtual

	Descrição
Nome	Hortonworks HDP 2.3
Sistema Operacional	CentOS 64 bits
Memória Principal	8GB
Processador	Intel Core i7 4790k
Número de Núcleos	8
Disco Rígido	150GB

Para poder realizar este trabalho, foi necessário ter uma base de dados para realizar a extração das informações. E para isto, foi utilizada uma base de dados adquirida em um portal de dados abertos do governo brasileiro [PBDA, 2015]. O conteúdo destes dados é

[1] Disponível em: https://github.com/Automattic/php-thrift-sql.
[2] Disponível em: http://www.hortonworks.com.

referente ao Exame Nacional do Ensino Médio (ENEM) e foi escolhida a edição da prova do ano de 2012.

A quantidade de registros contidos na base de dados é de aproximadamente trinta e seis milhões de registros, totalizando por volta de doze gigabytes de dados, número suficiente para que seja realizado um estudo do Hadoop. No Quadro 2 pode ser observado, um exemplo de consulta SQL, que foi implementada no sistema que foi utilizado a fim de validar este trabalho.

Quadro 2 – Exemplo de consulta SQL

```
SELECT count(cor_raca), cor_raca
FROM informacao.pessoa a
JOIN localizacao.cidade b ON a.fk_cidade = b.id_cidade
JOIN localizacao.estado c ON b.fk_estado = c.id_estado
JOIN localizacao.regiao d ON c.fk_regiao = d.id_regiao
WHERE id_estado = 22
GROUP BY cor_raca
ORDER BY cor_raca
```

O SGBD escolhido para receber os dados contidos na base de dados foi o PostgreSQL. Os motivos que levaram a sua escolha foi que, além de ser de código aberto, é um SGBD robusto, seguro e com diversas funcionalidades [PostgreSQL, 2015]. Na Figura 3 é mostrado um protótipo do Diagrama Entidade Relacionamento (DER) referente a base de dados que foi utilizada neste trabalho com o objetivo de mostrar a estrutura em que os dados estão organizados dentro do banco de dados.

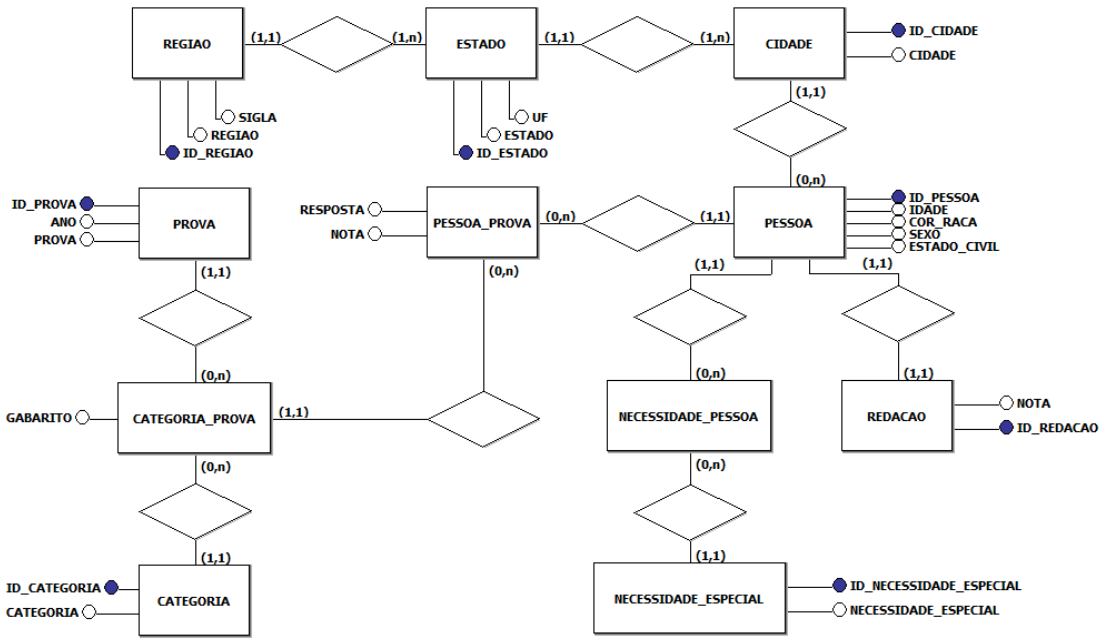

Figura 3. Diagrama Entidade Relacionamento (DER)

Por fim e com o intuito de validar o trabalho proposto, foi desenvolvido um sistema utilizando as ferramentas descritas ao longo desta seção e, que permite, ao interagir com

o mapa da Figura 4, por meio de eventos como, por exemplo, de *scroll* ou *click* sejam obtidas informações, pré-estabelecidas, a respeito da base de dados utilizada.

O filtro em que essas informações irão se basear é de acordo com o evento que for executado, se for um *scroll*, por exemplo, irá depender do nível do *zoom* (os níveis são região, estado e cidade respectivamente). Se for um evento de *click*, irá depender de qual região, estado ou cidade o usuário clicou.

O sistema fornece informações tendo como fonte de conteúdo a base de dados que foi citada nesta seção e a partir disto é possível apresentar, por exemplo, qual foi a nota média dos alunos nas provas (por categorias e geral) na região Sudeste do país, qual foi o estado ou cidade que obteve o melhor índice de frequência nas provas, em qual região os alunos alcançaram melhores resultados na prova de matemática, entre outras informações a respeito dos candidatos inscritos. Com base nisto e, ao analisar as estatísticas apresentadas, é possível concluir, por exemplo, em uma determinada região do país é preciso dar mais ênfase na área de ciências da natureza e diversas outras informações (o resultado obtido irá depender da escolha feita pelo usuário do sistema durante a interação com o mapa), auxiliando em tomadas de decisão.

A seguir, na Figura 4, é mostrada a interface do sistema que foi desenvolvido neste trabalho. Nela é possível que o usuário escolha o filtro para a busca de informações na base de dados (região, estado e cidade) através da interação com o mapa, tornando possível que o usuário visualize as informações de acordo com o filtro escolhido de forma simples e intuitiva.

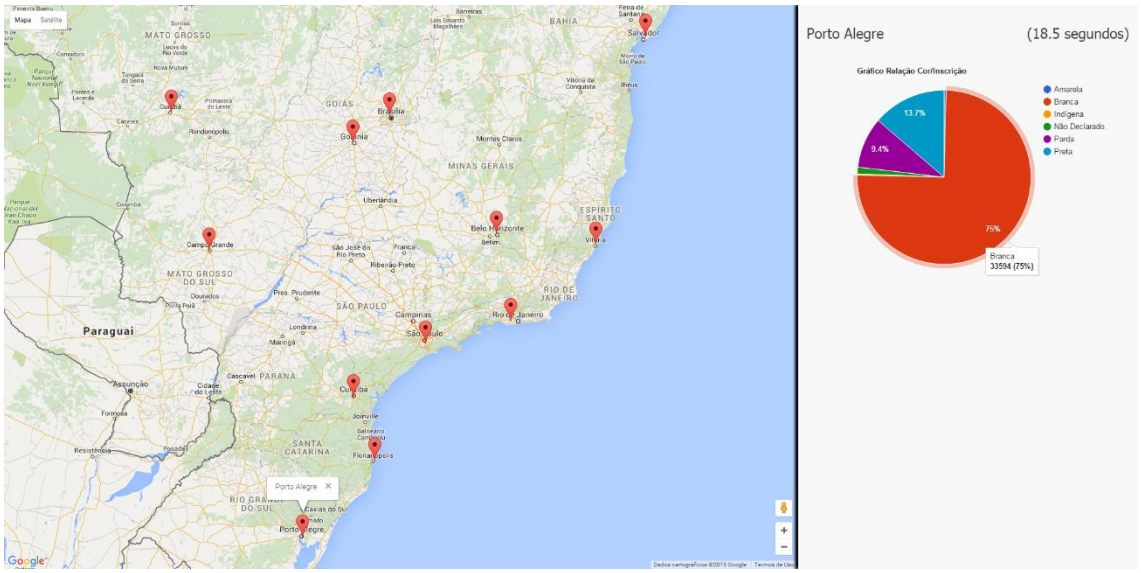

Figura 4. Interface do Software

7 Problemas Encontrados

Durante o desenvolvimento deste trabalho, foram encontrados alguns problemas que ocasionaram atrasos no desenvolvimento, tais como a falta de memória ao executar determinados procedimentos e/ou problemas relacionados a velocidade em que os

procedimentos executados levavam para serem concluídos (devido a limitação do ambiente de testes utilizado). Posteriormente e, provavelmente, o maior problema encontrado não possui relação com desempenho apresentado pelo *framework* e sim com estabilidade do sistema. No início do trabalho foi utilizado a versão 2.2 do *sandbox* da distribuição Hortonworks, porém durante algum tempo aconteceram problemas com os serviços básicos do Hadoop como, por exemplo, com o YARN, e também de conexão com o *host*, fazendo com que o sistema fosse totalmente reconfigurado, atrasando o desenvolvimento. Este problema foi solucionado após ser feita a instalação da versão mais recente da distribuição, a 2.3.

8 Conclusão

A quantidade de dados aumenta constantemente dificultando cada vez mais o processamento, sendo assim, o Hadoop é um *framework* que tem o objetivo de fazer com que esses dados sejam processados de maneira fácil e em tempo hábil. Desta maneira, este trabalho permitiu, além de estudar esta tecnologia, mostrar de que maneira o *framework* contribui para ambientes de *Big Data*. Para uma melhor observação de seu funcionamento foi desenvolvido um software a fim de validar este trabalho e mostrar como o Hadoop interage com as outras tecnologias.

Ao final deste trabalho de estudo realizado, foi possível observar que o Hadoop é um *framework* com diversas possibilidades, visto que é disponibilizado uma grande quantidade de mecanismos que auxiliam na sua utilização. Porém não foi possível utilizar todo seu potencial de processamento, devido ser um sistema para ambientes distribuídos e o estudo foi realizado utilizando apenas uma máquina. No entanto, mesmo em um ambiente limitado, foi possível perceber um ganho mínimo de desempenho.

Contudo foi possível observar alguns outros benefícios que o *framework* oferece, alguns deles são: a rapidez e a facilidade de importação da base de dados relacional, relativamente grande, para o Hadoop, uma interface completa onde podem ser realizadas as configurações do sistema de forma interativa, a fácil incorporação do *framework* com diferentes linguagens de programação, um ambiente para monitoramento do sistema que apresenta informações sobre a utilização recursos (e diversas outras informações), em tempo real, tornando o gerenciamento do sistema mais fácil.

Como trabalhos futuros pretende-se analisar o *framewok* Hadoop, porém desta vez dando deixando de lado como é o funcionamento do sistema e sim verificar o desempenho na execução das tarefas. Para isto, ao contrário de utilizar um ambiente virtualizado, pretende-se utilizar um *cluster*, de pelo menos dois nós para que seja possível fazer uma projeção de desempenho no caso de adicionar mais máquinas. Também é de objeto de estudo futuro a utilização do Apache Spark, visando aumentar o desempenho do sistema e dar maior possibilidade de utilização do *framework* em outras áreas da computação e não somente banco de dados como, por exemplo, aprendizado de máquina.

Referências

Alecrim E. (2015) O que é Big Data?. http://www.infowester.com/big-data.php. Acesso em Abril de 2015.

Andrade, T. (2012) MapReduce – Conceitos e Aplicações

Apache Hadoop (2015a) *What is* Apache Hadoop. http://hadoop.apache.org. Acesso em Março de 2015.

Apache Hadoop (2015b) Hadoop MapReduce NextGen Apache (YARN). https://hadoop.apache.org/docs/current/hadoop-YARN/hadoop-YARN-site/yarn.html. Acesso em Novembro de 2015.

Apache HBase (2015) *What* HBase *Does*. http://br.hortonworks.com/hadoop/hbase/. Acesso em Novembro de 2015.

Apache HCatalog (2015) HCatalog. http://br.hortonworks.com/hadoop/hive/#section_4. Acesso em Novembro de 2015.

Apache Spark (2015) What Apache Spark Does. http://br.hortonworks.com/hadoop/spark/. Acesso em Novembro de 2015.

Chatalov A., Rufino R. (2014) Big Data: Utilizando a Internet para tomada de Decisões

CIO (2014) Hadoop: Nove fornecedores que você deveria conhecer. http://cio.com.br/tecnologia/2014/07/02/hadoop-nove-fornecedores-que-voce-deveria-conhecer/. Acesso em Maio de 2015.

Cloudera (2015) Cloudera *Manager*. http://www.cloudera.com/content/cloudera/en/products-and-services/cloudera-enterprise/cloudera-manager.html. Acesso em Junho de 2015.

Dantas, R. e Barreto, M. (2013) Análise comparativa de ambientes e linguagens para computação intensiva de dados na nuvem

Dean, J. e Ghemawat, S. (2008) MapReduce: *Simplified Data Processing on Large Clusters* ACM, 51(1):107–113

Gasparotto M. H. (2014) Hadoop MapReduce: Introdução a *Big Data*. http://www.devmedia.com.br/hadoop-mapreduce-introducao-a-big-data/30034. Acesso em Abril de 2015.

Google *Developers* (2013) API Javascript do Google *Maps* v3. https://developers.google.com/maps/documentation/javascript/. Acesso em Maio de 2015.

Google *Developers* (2015) API Javascript do Google *Charts*. https://developers.google.com/chart/. Acesso em Outubro de 2015.

Gualtieri M. e Yuhanna N. (2014) *The* Forrester *Wave: Big Data* Hadoop *Solutions*, Q1 2014

Hortonworks (2015) MapReduce. http://br.hortonworks.com/hadoop/mapreduce/. Acesso em Junho de 2015.

Hue (2014) *How to configure Hue for your* Hadoop *cluster*. http://gethue.com/how-to-configure-hue-in-your-hadoop-cluster/. Acesso em Novembro de 2015.

Ianni, V. (2013) Big Data: Algumas definições e, sim, serve também para o pequeno negócio. http://www.devmedia.com.br/big-data-algumas-definicoes-e-sim-serve-tambem-para-o-pequeno-negocio/27527. Acesso em Novembro de 2015.

IBM (2013) *Big Data*: Expectativas, benefícios e barreiras. https://www.ibm.com/developerworks/community/blogs/ctaurion/entry/big_data_expectativas_beneficios_e_barreiras. Acesso em Maio de 2015.

Oliveira, D., Boeres C., Porto, F. (2014) Análise de Estratégias de Acesso a Grandes Volumes de Dados.

Portal Brasileiro de Dados Abertos (PBDA, 2015) Microdados do Exame Nacional do Ensino Médio – ENEM. http://dados.gov.br/dataset/microdados-do-exame-nacional-do-ensino-medio-enem. Acesso em Maio de 2015.

PostgreSQL (2015) *About*. http://www.postgresql.org/about/. Acesso em Maio de 2015

Schneider, R. D. (2012) Hadoop *for Dummies*

Lockwood G. (2015) Conceptual Overview of Map-Reduce and Hadoop. http://www.glennklockwood.com/data-intensive/hadoop/overview.html. Acesso em Novembro de 2015.

VMware (2015) *Why Choose* VMware. https://www.vmware.com/br/why-choose-vmware. Acesso em Junho de 2015

Volpato T., Rufino R., Dias J. (2013) Big Data – Transformando Dados em Decisões

YARN (2015) *What* YARN *Does*. http://br.hortonworks.com/hadoop/YARN/. Acesso em Novembro de 2015.