Universidade do Minho
Escola de Engenharia

Hugo Miguel Oliveira Torres

Benchmarking de Tecnologias de *Big Data* aplicadas à saúde/medicina

Dissertação de Mestrado

Mestrado em Engenharia e Gestão de Sistemas de Informação

Trabalho efetuado sob a orientação do(s)

Professor Doutor Manuel Filipe Vieira Torres dos Santos

Professor Doutor Carlos Filipe da Silva Portela

Outubro de 2017

DECLARAÇÃO

Nome: Hugo Miguel Oliveira Torres

Endereço eletrónico: a64862@alunos.uminho.pt **Telefone**: 917641479

Número do Cartão de Cidadão: 14175696

Título da dissertação: *Benchmarking* de Tecnologias de *Big Data* aplicadas à saúde/medicina

Orientador(es):

Professor Doutor Manuel Filipe Vieira Torres dos Santos

Professor Doutor Carlos Filipe da Silva Portela

Ano de conclusão: 2017

Designação do Mestrado: Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

Universidade do Minho, 03/01/2018

Assinatura: Hugo Miguel Oliveira Torres

AGRADECIMENTOS

Após ter sido concluída mais uma etapa da minha vida, é importante ser dado o devido reconhecimento àqueles fizeram parte deste percurso.

Ao Professor Doutor Manuel Filipe Vieira Torres dos Santos um obrigado pelos concelhos e pela oportunidade de trabalhar neste projeto.

Ao Professor Doutor Carlos Filipe da Silva Portela um especial agradecimento pela dedicação, paciência, entrega e conhecimento partilhado. A forma como se dedicou a este projeto como a tantos outros foi única e sinto-me sortudo por ter sido meu coorientador.

A todos os meus amigos, tanto os da infância como os que fiz durante o percurso académico, um eterno obrigado.

Tomás e Pedro, a vossa dedicação e empenho ao vosso trabalho e à vossa área inspiram-me a fazer mais por mim e pelos meus sonhos.

À minha namorada, um obrigado pelo incentivo, pela motivação e pela companhia nos bons e maus momentos.

Aos meus pais e à minha querida irmã, deixo uma palavra de agradecimento por tudo que fazem por mim, por todo o esforço, apoio incondicional, pelo incentivo e pelo investimento que fizeram na minha educação.

Resumo

Os avanços tecnológicos observados nas últimas décadas levaram a um aumento no volume e variedade dos dados gerados. Esses dados, quando armazenados, processados e analisados, podem fornecer novo conhecimento e uma maior perceção do negócio, o que pode ajudar as organizações a obter vantagem sobre os seus concorrentes. Está provado que *Big Data* está relacionado com um aumento na eficiência e eficácia em várias áreas. Embora muitos estudos tenham sido realizados com o intuito de provar o valor do *Big Data* na saúde/medicina, não existem muitos avanços efetuados na prática.

Pretende-se facilitar a adoção de tecnologias *Big Data* na medicina e em organizações ligadas à saúde. Vai ser discutido o potencial e os desafios na adoção de *Big Data*, comparando várias tecnologias *Big Data* (*Benchmarking*) que foram utilizadas ou projetadas para ser aplicadas na área da saúde.

Nesta dissertação foi realizada uma análise às tecnologias *Big Data* já existentes e aplicadas na área da saúde. Foi analisada uma solução *Big Data* desenvolvida para o projeto *INTCare*, uma solução baseada em *Hadoop* proposta para o hospital *Maharaja Yeshwatrao* situado na Índia e também uma solução que utiliza o *Apache Spark*. As três soluções mencionadas assentam em tecnologias *open source*. A solução *IBM PureData Solution For Healthcare Analytics* utilizada no *Seattle's Children's Hospital* e a solução *Cisco Connected Health Solutions and Services* fazem parte das soluções proprietárias analisadas.

O inicio deste documento apresenta uma descrição sucinta do contexto do projeto, qual a motivação e o objetivo deste trabalho. Posteriormente é apresentado o Estado da Arte onde são explicados os vários tópicos relacionados com o que foi feito e estudado. De seguida são apresentados os objetivos e as abordagens metodológicas utlizadas e de que forma foram aplicadas no desenvolvimento desta dissertação. Posteriormente são apresentadas várias ferramentas que integram as soluções encontradas. Na secção seguinte são apresentadas as várias experiências que comparam as soluções escolhidas para o *benchmarking*. Por fim, é apresentada a discussão dos resultados obtidos e sugestões que podem ser adotadas para reduzir o risco de adoção de tecnologias *Big Data* na área da saúde.

Palavras-Chave: *Big Data, Big Data Technologies, Big Data in Healthcare, Benchmarking*

ABSTRACT

The technological advances observed in the last decades led to an increase in the volume and variety of the generated data. This data, when collected, processed and analyzed, can provide new knowledge and deeper insights, which may help organizations in getting advantage over competitors. It is proven that Big Data is related to an increase in efficiency and effectiveness in many areas. Although many studies have been conducted trying to prove the value of Big Data in healthcare/medicine, few practical advances have been made. In this dissertation project, we intend to facilitate the adoption of Big Data technologies in medicine and healthcare organizations. The potential value and the challenges of the adoption of Big Data will be discussed by comparing several Big Data technologies (Benchmarking) used in or designed to be applied to healthcare.

In this dissertation, an analysis was made of the existing Big Data technologies applied in healthcare. We analyzed a Big Data solution developed for the INTCare project, a Hadoop-based solution proposed for the Maharaja Yeshwatrao hospital located in India and a solution that uses Apache Spark. The three solutions mentioned above are based on open source technology. The IBM PureData Solution for Healthcare Analytics solution used at Seattle's Children's Hospital and the Cisco Connected Health Solutions and Services solution are part of the proprietary solutions reviewed.

The beginning of this document presents a brief description of the project context, the motivation and the purpose of this work. Subsequently, the State of Art is presented where the various topics related to what was done and studied are explained. Following this topic, the objectives are presented also the research methodologies used and how they were applied in the development of this dissertation. Subsequently several tools are presented that integrate the solutions found. In the following section, various experiences are presented that compare the solutions chosen for benchmarking. Finally, the discussion of the results obtained and suggestions that can be adopted to reduce the risk of adoption of Big Data technologies in the health area are presented.

Keywords: Big Data, Big Data Technologies, Big Data in Healthcare, Benchmarking

ÍNDICE

Índice de Figuras

ÍNDICE DE TABELAS

Lista de Abreviaturas, Siglas e Acrónimos

HDFS - Hadoop Distributed File System

NIST - National Institute of Standards and Technology

EUA - Estados Unidos da América

ENR - Electronic Nursing Record

EHR - Electronic Health Record

GPS - Global Positioning System

PIB - Produto Interno Bruto

BI - Business Intelligence

APQC - American Productivity and Quality Center

SQL - Structured Query Language

UI - User Interface

API - Application Programming Interface

ML - Machine Learning

XML - eXtensible Markup Language

MPP - Massive Parallel Processing

CPU - Central Processing Unit

RAM - Random Access Memory

1. INTRODUÇÃO

1.1 Contextualização da Dissertação

A rápida evolução da tecnologia conduziu a novas fontes de dados e como consequência a um aumento no volume dos mesmos. Esta realidade veio alterar a dinâmica do mercado e passou a exigir das organizações uma maior capacidade para encontrarem novas formas de inovação para com isto se manterem na vanguarda. Foi assim que o *Big Data* começou a revelar o seu verdadeiro potencial em lidar com grandes volumes de dados provenientes de várias fontes e gerados a alta velocidade. Deste então, o *Big Data* tem evoluído pelo potencial e pelos resultados que demonstra e, quando bem implementado, traduz melhorias a nível do processo de tomada de decisão. Nos dias de hoje as tecnologias de *Big Data* estão presentes nas mais variadas áreas, estando presente no nosso quotidiano sem, por vezes, nos apercebemos da sua imensidão e da forma como influencia as nossas decisões. Temos o exemplo de organizações que utilizam o *Big Data* para perceber os clientes, descobrir as suas preferências, os seus comportamentos, entre outros, de modo a, através da informação recolhida, fornecer um serviço orientado ao cliente e obter assim a satisfação do mesmo. Atualmente, existem vários dispositivos que recolhem e armazenam a nossa informação, temos o exemplo dos *smartwatches* que registam a nossa pulsação cardíaca, os aparelhos médicos que registam várias informações sobre o nosso organismo, entre outros. De forma a retirar valor da informação recolhida é necessário armazenar, tratar e analisar a mesma.

O mercado de aplicações de *Big Data* é extenso e existem várias soluções ajustadas a diferentes necessidades.

Esta dissertação foca-se na pesquisa, seleção e posterior comparação de aplicações *Big Data* já implementadas na área da saúde/medicina, de modo a obter um conjunto de soluções devidamente testadas e comparadas.

1.2 Objetivos

Esta dissertação tem como objetivo responder à seguinte questão de investigação:

> *De que forma a aplicação de tecnologias de Big Data podem melhorar os serviços prestados por organizações na área da saúde?*

Para dar resposta a esta questão de investigação, o trabalho encontra-se dividido em três fases:

- A primeira fase coincidiu com a revisão de literatura e consistiu na pesquisa de informação relacionada com os temas *Big Data* e *Big Data* na saúde. Nesta fase foi possível perceber qual o estado da adoção de tecnologias *Big Data* na saúde e foram identificados os desafios e o potencial da implementação de tecnologias *Big Data* em hospitais/clínicas de saúde; foi também possível perceber a necessidade de avaliar projetos para reduzir o risco associado a investimentos em *Big Data*.
- A segunda fase coincidiu com a pesquisa de aplicações utilizadas ou projetadas para ser implementadas em hospitais/clínicas de saúde, aprovadas pela comunidade científica. Nesta fase efetuou-se o levantamento de aplicações utilizadas na área da saúde, posteriormente procedeu-se à análise das mesmas.
- A terceira fase focou-se numa investigação contínua de aplicações utilizadas na área da saúde e também na pesquisa de experiências efetuadas a aplicações semelhantes às escolhidas para comparação. Nesta fase foram definidos requisitos, escolhidas as aplicações e analisadas várias experiências aprovadas pela comunidade científica.

A análise do conceito de *Big Data* possibilitou um maior entendimento acerca do tema, mas também foi possível perceber a influência e as áreas de aplicação do mesmo.
Existem poucas soluções *Big Data* que demonstraram o seu valor na área da saúde/medicina, como será possível observar ao longo do documento. O objetivo desta dissertação é então facilitar a entrada do *Big Data* na área da saúde/medicina.

1.3 Estrutura do Documento

O presente documento tem como principais objetivos permitir ao leitor uma perceção aprofundada do problema em causa, assim como permiti-lhe compreender a abordagem para a sua resolução.

Para facilitar a compreensão do leitor, o documento foi organizado em seis capítulos:

- **Capítulo 1:** Introdução sobre o tema desta dissertação; esclarecendo em detalhe o problema;
- **Capítulo 2:** Neste capítulo serão apresentados os conceitos teóricos que podem auxiliar na resolução do problema. É pretendido também situar o leitor no contexto teórico e tecnológico desta dissertação;
- **Capítulo 3:** Este terceiro capítulo tem como propósito explicar as metodologias utilizadas para elaborar a pesquisa sobre este tema e como as mesmas foram aplicadas no desenvolvimento desta dissertação;
- **Capítulo 4:** Este capítulo apresenta e descreve os componentes de parte das arquiteturas encontradas;
- **Capítulo 5:** Este capítulo tem o propósito de apresentar as arquiteturas encontradas, quais os componentes e em que âmbito foram apresentadas à comunidade científica;
- **Capítulo 6:** Neste capítulo serão apresentadas duas experiências que fazem comparação entre as arquiteturas selecionadas;
- **Capítulo 7:** Neste capítulo será apresentada a análise e discussão dos resultados observados em ambas as experiências e também uma análise ao estado atual do *Big Data* na saúde;
- **Capítulo 8:** Por fim, o último capítulo consiste na apresentação das conclusões finais, apresentação dos contributos, análise de riscos e enumeração de perspetivas de trabalho futuro.

2. REVISÃO DE LITERATURA

Este capítulo reflete a pesquisa bibliográfica efetuada. O mesmo encontra-se dividido em diferentes subsecções, referentes à estratégia de pesquisa utilizada e aos principais conceitos a considerar.

2.1 Estratégia de Pesquisa Bibliográfica

A pesquisa foi efetuada sobre os seguintes conceitos: *Big Data, Big Data in Healthcare, Big Data Technologies in Healthcare, Big Data Analytics, Analytics in Healthcare, Big Data Technologies in Healthcare, Benchmarking*. Os artigos utilizados para auxiliar a elaboração deste documento foram selecionados, principalmente, tendo em consideração a data do artigo (excetuando alguns casos, os mesmos não têm mais de dez anos), o conteúdo do artigo/obra e de que forma de relaciona com o tema desta dissertação. As fontes de dados utilizadas para fazer a pesquisa foram as seguintes:

- *Science Direct;*
- *Google Scholar;*
- *Ieeexplore;*
- *Scopus;*
- *SpringerLink;*
- Diversos livros científicos relacionados com a área.

2.2 Big Data

Desde a invenção dos computadores que grandes volumes de dados são gerados a um ritmo surpreendente (Yaqoob et al., 2016). O excesso de dados na *internet*, provenientes de diferentes fontes, levam a uma sobrecarga de dados disponíveis para a sociedade (José & Ribeiro, 2014). Até 2003, 5 *exabytes* de dados foram criados pelo ser-humano; atualmente, essa quantidade é criada em apenas 2 dias (Sagiroglu & Sinanc, 2013).

Avanços tecnológicos a nível dos dispositivos móveis e de tecnologias, como a *internet* sem fios, criou uma sociedade dependente destes dispositivos e tecnologias, tornando-os essenciais para o nosso quotidiano. (C. Lima & Calazans, 2013). Em 2010, mais de 4 biliões de pessoas, cerca de 60% da população mundial, usavam telemóveis, das quais 12% possuíam *smartphones* (Manyika et al., 2011). Segundo Lima & Calazans (2013), a cada minuto, 571 novos *sites* são criados e 204.166.667 mensagens de correio eletrónico são enviadas. A rede móvel adquire 217 novos utilizadores e o *Google*

recebe mais de 2 milhões de pesquisas. O *Foursquare* regista 2.083 *check-ins*, o *YouTube* recebe 48 horas de vídeos novos, 684.478 conteúdos são publicados no *Facebook*, 3.600 fotos são partilhadas no *Instagram* e mais de 1.000 *tweets* são enviados pelo *Twitter*. A tendência é que todos estes números continuem a aumentar, à medida que o número de utilizadores da *internet* e a sua atividade online aumenta. Atualmente, está a ser gerada uma quantidade tão grande de dados que torna fisicamente impossível o seu armazenamento (Manyika et al., 2011).

As principais fontes de grandes volumes de dados são a *Internet of Things*, *self-quantified*, multimédia e redes sociais (Yaqoob et al., 2016).

A *Internet of Things* - internet das coisas - refere-se à interconexão em rede de objetos do quotidiano, que muitas vezes são equipados com inteligência ubíqua. A *internet of things* aumentará a ubiquidade da *internet* através da integração de objetos que utilizamos no quotidiano (Xia, Yang, Wang, & Vinel, 2012). Os dados da *internet of things* provêm de dispositivos *GPS (Global Positioning System*, Sistema de Posicionamento Global*)*, *smartphones*, alarmes, vários tipos de sensores, entre outros (Yaqoob et al., 2016).

Os dados *self-quantified* são gerados por indivíduos que monitorizam o seu comportamento, consistem em dados provenientes de pulseiras utilizadas para monitorizar movimentos, aparelhos utlizados para medir a tensão arterial, entre outros. Os dados multimédia são provenientes de várias fontes e têm diferentes formatos, tais como texto, imagens, vídeo, áudio, entre outros. Os dados das redes sociais são provenientes do *Facebook*, *Twitter*, *Instagram*, entre outras redes existentes. (Yaqoob et al., 2016).

O ambiente em que as organizações se inserem é moldado pela disponibilidade dos dados (Silva & Breternitz, 2013). As organizações recolhem grandes volumes de informação acerca dos seus clientes, fornecedores, operações e milhões de sensores conectados à rede que estão a ser incorporados no nosso dia-a-dia em dispositivos como *smartphones*, automóveis, entre outros (Manyika et al., 2011).

Segundo Zikopoulos et al. (2012), os negócios têm mais potencial para adquirir novo conhecimento, mais do que alguma vez foi possível, no entanto, à medida que os dados se acumulam, a percentagem de processamento de dados baixa rapidamente.

"You can't manage what you don't measure" (Brynjolfsson & McAfee, 2012) ("Não podes gerir o que não medires"). Em suma, devido ao *Big Data*, os gestores conseguem medir e, portanto, saber mais relativamente aos seus negócios e traduzir esse conhecimento em melhorias a nível do processo de tomada de decisão e de desempenho (Brynjolfsson & McAfee, 2012).

2.2.1 Definições e Conceitos

Existem várias definições de *Big Data*, dado que é um termo que recebe bastante atenção a nível global e é alvo de várias publicações. Apesar do seu mediatismo, ainda não existe uma definição comum, daí o propósito desta secção, onde serão apresentadas várias definições de diferentes autores.

O NIST Big Data Public Working Group (2015) afirma que *Big Data* é um termo utilizado para descrever a grande quantidade de dados a circular num Mundo cada vez mais ligado à rede, cada vez mais digital, carregado de sensores e orientado pela disponibilidade de informação.

Segundo Zikopoulos et al. (2012), *Big Data* aplica-se a informação que ultrapassa as capacidades de processamento e análise dos processos e ferramentas tradicionais.

Para Manyika et al. (2011), *Big Data* refere-se a *datasets* cujo tamanho excede a capacidade de captura, armazenamento, gestão e análise das ferramentas de bases de dados tradicionais.

Já Davenport (2011), refere *Big Data* como dados que têm características, tais como, o tamanho, a falta de estrutura e a velocidade a que os mesmos são gerados, que estão além da capacidade das ferramentas tradicionais.

Silva & Breternitz (2013), designam *Big Data* como um conjunto de tendências tecnológicas que possibilitam uma nova abordagem para o tratamento e compreensão de grandes volumes de dados para auxiliar o processo de tomada de decisão.

Por fim, Hashem et al. (2015) define *Big Data* como um conjunto de técnicas e tecnologias que requerem novas formas de integração para revelar valores escondidos de *datasets* diversos, complexos e de grande escala.

As várias definições acima apresentadas são bastante similares, mas alguns autores elaboram mais o conceito. As duas primeiras definições apresentadas por NIST Big Data Public Working Group (2015) e Zikopoulos et al. (2012) são bastante curtas e simples, referem um grande volume de dados e a falta de capacidade das ferramentas tradicionais para lidar com o *Big Data*; já a terceira, por Manyika et al. (2011) é uma junção das anteriores. A quarta definição apresentada por Davenport (2011) introduz novas características, menciona grandes volumes de dados pouco ou nada estruturados -não estruturados ou semiestruturados- gerados a grande velocidade. As restantes por Silva & Breternitz (2013) e Hashem et al. (2015) definem o *Big Data* como um conjunto de tecnologias que possibilitam a descoberta de novo conhecimento para auxiliar a decisão.

Pode então ser afirmado que o *Big Data* é algo mais que apenas um grande volume de dados, é um conjunto de tecnologias que permite capturar, armazenar, processar, analisar e retirar valor de forma a adquirir novo conhecimento de grandes volumes de dados provenientes de várias fontes que são gerados

a grandes velocidades. Tendo por base esta definição surgiram as características apelidadas de V's (Volume, Variedade, Velocidade) que vão ser apresentadas em mais detalhe no próximo tópico.

2.2.2 Características

Alguns autores supracitados, Davenport (2011) e Zikopoulos et al. (2012), afirmam que existe um conjunto de caraterísticas que definem o *Big Data*, denominadas de "V's".

Segundo Zikopoulos et al. (2012) as três características que definem o *Big Data*, são: Volume, Variedade e Velocidade. Já Hurwitz et al. (2013) afirmam que os três V's citados anteriormente são uma visão demasiado simplista do termo e propõe um quarto V, sendo ele a veracidade. Na literatura ainda surgem alguns autores que adicionam um quinto V, o valor (Taurion, 2013). No entanto, para Maçada et al. (2015), os três V's, Volume, Velocidade e Variedade são a melhor forma de caracterizar o *Big Data*, afirmando que Valor e Veracidade são resultados e não características.

Neste projeto vão ser apresentados os cinco V's do *Big Data*, dando maior ênfase aos três V's iniciais (Volume, Variedade, Velocidade) uma vez que, como mencionado por Maçada et al. (2015), Valor e Veracidade estão relacionados com a apresentação dos resultados.

a) Volume

O grande volume de dados a ser gerado cresce exponencialmente e provém das mais variadas fontes. Desde 2012, cerca de 2.5 *exabytes* de dados são gerados diariamente. Atualmente, as organizações são confrontadas com enormes quantidades de dados que rondam os *petabytes* (Brynjolfsson & McAfee, 2012).

O armazenamento desta enorme quantidade de dados possibilita às organizações a descoberta de novo conhecimento e padrões escondidos (Hashem et al., 2015).

Zikopoulos et al. (2012) afirma que, apesar do aumento constante no volume de dados, a percentagem de dados que as organizações são capazes de processar, compreender e analisar está a diminuir. Assim, esta enorme quantidade de dados é um desafio por si só, dado que as ferramentas tradicionais de bases de dados não têm a capacidade para capturar, armazenar e gerir a mesma (Manyika et al., 2011).

b) Variedade

Segundo Zikopoulos et al. (2012), a sociedade investe grande parte do seu tempo com dados estruturados (representam 20% do volume total dos dados gerados), que as ferramentas tradicionais conseguem processar, analisar e compreender. Mas o grande desafio está nos restantes 80% que, para além do grande volume e velocidade a que são gerados, são dados que se apresentam semiestruturados

ou não estruturados. Algumas organizações consideram mais importante a questão da falta de estrutura dos dados do que o volume dos mesmos (Davenport, 2011).

Quando se menciona variedade em *Big Data*, estão a ser referidas as várias fontes de dados, tais como redes sociais, diferentes tipos de sensores, dispositivos móveis, entre outros; e a referir vários tipos de dados - estruturados, semiestruturados e não estruturados (Brynjolfsson & McAfee, 2012). São exemplo disso, imagens, vídeos, *tweets*, leituras de sensores, áudio, publicações, entre muitos outros.

c) Velocidade

A velocidade a que os dados são gerados acompanha o ritmo de aumento do volume dos mesmos (Davenport, 2011). Atualmente, algumas organizações necessitam que a informação seja processada em tempo real ou quase em tempo real, ultrapassar este desafio pode garantir vantagem competitiva sobre os seus concorrentes (Brynjolfsson & McAfee, 2012). Essa vantagem pode traduzir-se em definir uma nova tendência, identificar um novo problema, ou até mesmo identificar e aproveitar novas oportunidades, por vezes, esta vantagem sobre os concorrentes consegue-se por apenas uma questão de segundos. Hoje em dia, muitos dos dados que são gerados têm um "prazo de validade", ou seja, são apenas relevantes para as organizações se forem analisados quase em tempo real (Zikopoulos et al., 2012).

d) Valor

Esta característica está relacionada com o valor económico dos dados. Muitas vezes as organizações têm ao seu dispor informação valiosa escondida na imensidão de dados que possuem, o desafio passa por identificar o que é ou não valioso (Dijcks, 2012).

e) Veracidade

Esta característica está relacionada com a qualidade dos dados. White (2012) afirma que a má qualidade dos dados, aquando da integração dos mesmos com outros dados ou informação, pode levar as organizações a fazer análises incorretas o que pode afetar o processo de tomada de decisão.

Segundo LaValle (2009), 1 em cada 3 gestores tomam decisões baseadas em informação incompleta ou na qual não confiam.

Após alguma análise da literatura, torna-se evidente que existe bastante potencial na informação, como por exemplo, a descoberta de padrões escondidos, possibilidade de adquirir novo conhecimento, vantagem competitiva, entre outros. Até à data era difícil a análise do *Big Data*, dado que a tecnologia era insuficiente e também bastante dispendiosa. Atualmente, a tecnologia evolui a um ritmo surpreendente e os preços decrescem gradualmente, tornando aquilo que era bastante difícil no passado a realidade do presente (Zikopoulos et al., 2012).

A análise de toda a informação disponível para uma organização (*Big Data*, informação tradicional) fornece uma melhor compreensão do seu negócio o que pode levar a um aumento de produtividade, a uma melhor posição no mercado face aos seus concorrentes e a uma maior inovação (Dijcks, 2012)

2.3 *Big Data* na Saúde

Atualmente, o *Big Data* está bem implementado em vários setores de atividade, mas o mesmo não se verifica para o setor da saúde (Groves, Kayyali, Knott, & Van Kuiken, 2013). A indústria da saúde gera enormes quantidades de dados, apesar da sua maioria estar armazenado em formato não digital. Atualmente, a tendência é digitalizar a maioria da informação (Raghupathi & Raghupathi, 2014).

Segundo Feldman et al. (2012), o aumento do volume de dados na indústria da saúde vem, não só da criação de novas formas de dados (imagens a três dimensões, leituras de sensores biométricos, entre outros) como também da digitalização dos dados já existentes (registos médicos, imagens de radiologia, dados de sequências de ADN, entre outros).

O *McKinsey Global Institute* efetuou um estudo com o intuito de perceber o potencial do *Big Data* em cinco áreas, sendo uma delas a área da saúde nos Estados Unidos da América (EUA) (Manyika et al., 2011). O setor da saúde nos EUA é um dos maiores setores da economia do país, representando cerca de 17% do produto interno bruto (PIB) e emprega cerca de 11% dos trabalhadores do país. Apesar da importância deste setor, ainda é possível verificar-se um atraso na adoção do *Big Data* face a outras indústrias (Manyika et al., 2011).

Dado o notório atraso na adoção do *Big Data* por parte da indústria da saúde, é necessário identificar os desafios e potenciais uso de *Big Data* nesta indústria.

2.3.1 Potencial do *Big Data* na saúde

Manyika et al. (2011) estimam que existe a oportunidade de gerar mais de 300 biliões de dólares americanos anualmente, sendo que, dois terços deste valor advêm de reduções nos custos com a saúde nacional. Os mesmos identificaram cinco categorias onde o *Big Data* pode ter um impacto positivo na área da saúde, iremos agora apresentar essas categorias:

a) Operações Clínicas

Manyika et al. (2011) estimam que, caso o *Big Data* seja bem implementado nesta categoria, existe potencial de reduzir os custos com a saúde em cerca de 165 biliões de dólares americanos.
Nesta categoria existem cinco maneiras/formas de obter valor com a implementação do *Big Data* (Manyika et al., 2011):

- Pesquisa baseada em resultados determina qual/quais os melhores tratamentos de acordo com o utente. Consiste em analisar dados de utentes e resultados de forma a comparar a eficácia de várias intervenções;
- Implementar sistemas de apoio a decisão clínica para aumentar a eficiência e qualidade das operações;
- Analisar e criar transparência em dados de procedimentos médicos. A transparência destes dados permite, por exemplo, identificar as fontes de desperdício dos processos clínicos e após uma análise, é possível otimizar esses processos;
- Monitorização remota de utentes, ou seja, coletar dados de utentes com doenças crónicas e analisar os dados de forma a determinar se os utentes estão a seguir o tratamento corretamente, isto permite otimizar a terapêutica e também perceber quais os melhores fármacos;
- Analisar os dados de cada utente de forma a identificar que utentes beneficiariam de cuidados proativos, como por exemplo, avaliar se um utente está propício a desenvolver um tipo de doença (por exemplo, diabetes) e se beneficiaria de medicina preventiva.

b) Preço

Estima-se que é possível criar 50 biliões de dólares americanos com a implementação do *Big Data* nesta área, sendo que, metade deste valor advém da redução de custos com a saúde. Exemplo de formas que permitem extrair valor do *Big Data* nesta categoria (Manyika et al., 2011):

- Implementação de sistemas automatizados - através de técnicas de *machine learning,* tais como redes neuronais para detetar fraudes e verificar o rigor dos pagamentos a seguradoras. As poupanças nesta área podem ser atingidas através da implementação de bases de dados de pagamentos e algoritmos treinados para detetar fraudes.

c) Departamento de Investigação e Desenvolvimento

Estima-se que seja possível criar mais de 100 biliões de dólares americanos e cerca de um quarto deste valor provém da redução de custos com a saúde. Alguns exemplos do potencial do *Big Data* são (Manyika et al., 2011):

- Agregação de dados de investigação para realizar modelos preditos para testar novos fármacos e determinar uma melhor alocação de recursos;
- Utilizar ferramentas estatísticas e algoritmos para melhorar o processo de ensaios clínicos e de recrutamento de utentes;
- Analisar dados de ensaios clínicos e registos de utentes para identificar, por exemplo, novas indicações e efeitos secundários de um fármaco;
- Análise de grandes volumes de dados (dados do genoma humano) para melhorar o setor de investigação de desenvolvimento e desenvolver medicina personalizada (examinar relações entre variações no genoma e predisposição para certas doenças e adequar o tratamento);
- Analisar padrões de doenças e tendências para modelar a procura futura, os custos e definir estratégias e investimentos no setor de investigação e desenvolvimento.

d) Novos modelos de negócio

A digitalização de informação na área da saúde está a criar novos modelos de negócios que complementam ou podem até substituir os existentes. Manyika et al. (2011) realçam o potencial de dois novos modelos de negócio:

- Agregar e analisar registos de utentes com o intuito de fornecer dados e serviços a outras organizações;

- Plataformas *online* e comunidades que permitam a partilha de conhecimento/experiências entre utentes e médicos.

e) Saúde Pública

O uso de *Big Data* pode melhorar a vigilância e resposta à saúde pública. Com o uso de uma base de dados de utentes e tratamentos a nível nacional, as organizações responsáveis pela saúde pública podem garantir uma rápida e coordenada resposta na deteção de doenças contagiosas e uma vigilância mais rigorosa na contenção de surtos infeciosos (Manyika et al., 2011).

O *Big Data* pode contribuir para um desenvolvimento mais rápido e mais eficaz de vacinas específicas, por exemplo, a escolha da estirpe de gripe. (Raghupathi & Raghupathi, 2014).

2.3.2 Desafios ao *Big Data* na saúde

Groves et al. (2013) identificou alguns desafios na adoção do *Big Data* na indústria da saúde nos EUA, sendo eles: resistência à mudança; pouco investimento em tecnologias de informação; privacidade dos dados; e interoperabilidade.

Já Salas-Veja et al. (2015) enumerou vários desafios ao uso do *Big Data* na saúde no ambiente da União Europeia, sendo eles: confidencialidade e segurança dos dados; acesso à informação; fiabilidade dos dados; interoperabilidade.

a) Resistência à mudança

Os médicos e profissionais de saúde estão habituados a tomar decisões com base no seu julgamento clínico e não em resultados de análises ao *Big Data* (Groves et al., 2013).

b) Pouco Investimento em tecnologias de informação

Muitos *stakeholders* da área da saúde têm receio de fazer investimentos em tecnologias de informação dada a incerteza do retorno. Apesar dos sistemas que possuem no momento serem funcionais, têm pouca capacidade de padronizar e consolidar dados (Groves et al., 2013).

Segundo Dias & Duarte (2015), existem poucas iniciativas que defendem o uso generalizado do *Big Data* em hospitais e clínicas.

c) Privacidade e segurança dos dados

Um dos maiores desafios a ultrapassar é, possivelmente, a questão da privacidade dos dados, dado que, a implementação do *Big Data* pode requerer a partilha de dados entre várias organizações o que por vezes é um fator desmotivador.

Segundo Salas-Vega et al. (2015), os utentes têm receio que a má interpretação da sua informação, em particular dados acerca da sua genética, os possa afetar negativamente, em termos de seguros de saúde e emprego.

Groves et al. (2013) afirma que a partilha de informação entre as várias organizações da indústria da saúde torna-se difícil, em parte, devido a preocupações com a confidencialidade dos dados.

d) Interoperabilidade

Muitas vezes a partilha é dificultada não só por questões como a privacidade, mas também por falta de interoperabilidade.

Segundo Groves et al. (2013), por vezes, numa mesma organização é possível verificar que informação importante fica retida num determinado departamento, dado que não existe interoperabilidade, o que dificulta a partilha de dados e a comunicação de resultados.

Para Salas-Vega et al. (2015), a interoperabilidade é crucial no armazenamento de informação médica, desenvolvimento de *interfaces* comuns, definir padrões de qualidade, entre outros. No entanto, é necessário estabelecer leis e reestruturar ou criar novos modelos de negócio. A interoperabilidade na União Europeia enfrenta complicações como a linguagem e os diferentes padrões clínicos.

e) Acesso à informação

Salas-Vega et al. (2015) afirma que, para o consumidor, a principal preocupação é o uso e controlo dos dados por parte de terceiros. As organizações, apesar de interessadas no armazenamento e uso de informação, estão também preocupadas com a divulgação de propriedade intelectual.

f) Fiabilidade dos dados

A introdução manual de dados de saúde em sistemas eletrónicos está sujeita ao erro humano. No entanto, o mesmo pode acontecer para sistemas regularizados que podem também ser tendenciosos no armazenamento e análise de dados (Salas-Vega et al., 2015)

Existem outros desafios que devem ser abordados e solucionados, mas que não se encaixam nas categorias acima, sendo eles (Dias & Duarte, 2015):

- Demasiada dependência de sistemas eletrónicos;
- A maioria dos defensores da utilização do *Big Data* na saúde (investigadores, organizações farmacêuticas, saúde publica e outras organizações governamentais) não prestam assistência diretamente ao utente.

Apesar dos desafios que necessitam de ser ultrapassados, existe bastante potencial para ser explorado. Mas antes de implementar *Big Data* numa organização é necessário perceber se essa organização está a lidar de facto com *Big Data.* No próximo tópico iremos discutir se um determinado projeto é, ou não um projeto *Big Data.*

2.4 Avaliar Projetos

Apesar do potencial que o *Big Data* apresenta, é necessário avaliar se um projeto é ou não *Big Data* e se há capacidade para lidar com o mesmo.

Segundo Portela et al. (2016), grande parte das organizações, principalmente as Europeias, não estão preparadas para *Big Data* numa perspetiva técnica e de negócio, e não veem a relevância de investir em *Big Data*. Um projeto *Big Data* pode ser visto como um investimento de risco. A organização deve ser consciente e saber quais as razões que levam um projeto a ser bem ou mal sucedido. Alguns projetos *Big Data* falham, pois, as organizações têm o foco na tecnologia e não nas oportunidades de negócio.

Lavastorm Analytics (2014) identificaram três razões para que levam ao fracasso de projetos *Big Data*, sendo elas:

- **a)** *Big Data* é tratado como um projeto com início e fim bem definidos e não como um exercício de exploração ágil;
- **b)** O retorno de investimento é um obstáculo muito grande na adoção de *Big Data*. É necessário haver flexibilidade e agilidade para cometer erros "baratos" e aprender com os mesmos.
- **c)** Em alguns casos, projetos *Big Data* não estão ligados a uma necessidade de negócio. São apenas considerados para fins científicos, sem objetivos de negócio ou métricas.

Portela et al. (2016), apresentaram uma *framework* para avaliar a complexidade de projetos *Big Data*. O objetivo desta *framework* é ajudar a definir o "problema" da organização, de acordo com a dimensão dos 3 V's (volume, velocidade e variedade) e os respetivos níveis de complexidade. Através de cálculos que vão ser apresentados mais à frente, esta *framework* permite posicionar o "problema" a ser analisado numa gama de problemas *Big Data*.

A *framework* assume a forma de uma matriz como se pode ver na Tabela 1. Esta matriz é constituída por níveis de complexidade e dimensões.

Níveis de complexidade (CL- *complexity level*) – Define a escala de complexidade para cada dimensão. Existem cinco níveis de complexidade CL1 até CL5, sendo que, CL1 é o menor nível de complexidade e CL5 o maior.

Dimensão – Uma arquitetura *Big Data* é definida/depende de três dimensões que estão relacionadas entre si, que são: volume, velocidade e variedade. A conjugação destas dimensões com os níveis de complexidade, torna possível definir um "problema" *Big Data*. O volume não refere o volume de dados existentes na organização, mas sim a quantidade de dados coletada atualmente.

Tabela 1 - BigDAF. Adaptado de (F. Portela et al., 2016).

Dimensão/CL	**CL1**	**CL2**	**CL3**	**CL4**	**CL5**
Volume	<1000GB	5TB-50TB	50TB-500TB	500TB-2000TB	>2PB
Velocidade	*Batch*	*Intra-day*	*Hourly-refresh*	*Real-time*	*Streaming*
Variedade	*Structured Data*	*Docs,XML,TXT,JSON*	*Web-log; sensors and device events*	*Image; social graph feeds; geospatial information*	*Video; Voice*

Portela et al. (2016), forneceram uma solução para obter uma medida absoluta para categorizar um projeto. A solução apresentada consiste na soma do produto entre cada dimensão e o valor do nível de complexidade como se pode ver na Tabela 2. Dado que existem dimensões mais importantes, a cada dimensão é atribuído um peso (sendo que o total é de 100%).

Tabela 2- Fórmula de cálculo do nível de complexidade. Adaptado de (F. Portela et al., 2016).

Dimensão/CL	**CL1(1)**	**CL2(2)**	**CL3(3)**	**CL4 (4)**	**CL5 (5)**
Volume (60)	CL=(WVolume*CLx)+(WVelocidade*CLx)+(WVariedade*CLx)				
Velocidade (10)					
Variedade (30)					

Portela et al. (2016) afirma que o volume é a dimensão de maior importância em projetos *Big Data* e portanto, na tabela proposta pelo mesmo, o volume assume o maior valor (60%). Afirma também que o volume é a característica mais desafiante. A variedade representa 30% dado que representa vários desafios para as organizações que têm que lidar com tipos de dados que nunca foram processados para retirar valor dos mesmos. Por fim, temos a velocidade com 10%, aparece com menor peso, dado que os desafios que esta característica apresenta podem ser resolvidos aumentando as capacidades de processamento, como por exemplo, investir em *hardware* e paralelismo.

Após a atribuição dos pesos é possível fazer o cálculo e avaliar o projeto. O resultado da equação permite enquadrar o projeto num dos quatro resultados possíveis, que são:

- [100-200] – *Traditional BI issue* (Problema *Business Intelligence* tradicional) – A organização enfrenta um problema que pode ser resolvido com investimentos em capacidade de armazenamento e/ou ferramentas de processamento de texto;
-]200-300] – *BI Issue near Big Data challenge* (Problema *Business Intelligence* perto de um desafio *Big Data*) – Define um problema que pode evoluir para um problema *Big Data*. Pode ser resolvido, temporariamente, através do uso de ferramentas de análise avançada;
-]300-400] – *Big Data Issue* (Problema *Big Data*) – Necessidade para investimentos numa arquitetura *Big Data*. Quando se atinge um problema *Big Data*, as suas características (volume, variedade e velocidade) não são um grande problema pois as ferramentas *Big Data* estão preparadas para lidar com elas;
-]400-500] – *Complex Big Data Issue* (Problema *Big Data* Complexo) – Neste caso, torna-se urgente investir num projeto *Big Data.* Tecnologias *Business Intelligence* não são capazes de lidar com uma elevada e contínua quantidade de dados. Algumas ferramentas *Big Data* podem ser incapazes de lidar com este tipo de problema dado que nem todas as ferramentas satisfazem em pleno as necessidades do cliente (F. Portela et al., 2016).

A avaliação de projetos ajuda as organizações a tomar melhores decisões no que diz respeito a investimentos e pode evitar insucessos. O objetivo da *framework* apresentada é ajudar as organizações a encontrar um equilíbrio entre a necessidade de acompanhar as tendências tecnológicas e do mercado e as suas necessidades reais, presentes e futuras (F. Portela et al., 2016).

3. ABORDAGEM METODOLÓGICA

Neste projeto de dissertação foram utilizadas duas metodologias, uma de investigação e uma prática. A metodologia de investigação escolhida foi a *Case Study*, para a parte prática foi utilizada a metodologia *Benchmarking*.

3.1 Descrição das Metodologias

Neste capítulo serão apresentadas e descritas as metodologias que foram utilizadas para a parte de investigação e para a parte prática.

3.1.1 *Case Study* (Estudo de Caso)

Esta abordagem metodológica consiste na análise de várias fontes de dados com o objetivo de fazer um estudo aprofundado sobre o tema escolhido. (Tellis, 1997)

Para Coutinho & Chaves (2002), o estudo de caso consiste em examinar o tema escolhido em detalhe e em profundidade, no seu contexto natural, reconhecendo a sua complexidade e recorrendo a todos os métodos que se revelem apropriados.

Yin, citado de Tellis (1997), apresentou quatro aplicações para um estudo de caso, sendo elas:

- Explicar relações causais complexas em intervenções da vida real;
- Descrever o contexto da vida real em que a intervenção ocorreu;
- Descrever a própria intervenção;
- Explorar as situações em que a intervenção, que está a ser avaliada, não tem um conjunto claro de resultados.

Para aplicar a metodologia, foram seguidas as quatro fases propostas por Yin, citado de Coutinho & Chaves (2002), que são:

- **Planear o protocolo de estudo de caso** – Determinar as competências necessárias para executar o estudo de caso e desenvolver e rever o protocolo;
- **Realizar o estudo de caso** – Recolher os dados e fazer entrevistas e/ou questionários, caso seja necessário;
- **Analisar as evidências do estudo de caso** – Analisar as evidências recolhidas na fase da fase anterior;

- **Desenvolver conclusões, recomendações, e as implicações com base nas evidências** – Esta fase é possivelmente a mais importante para o utilizador. É necessário que seja feita uma boa explicação do caso de estudo para que o mesmo não leve a conclusões erradas por parte do utilizador.

3.1.2 Benchmarking

Segundo Stapenhurst (2009), o *benchmarking* é um método de medir e melhorar uma organização comparando-a com os melhores.

Benchmarking é definido como o processo de identificar os padrões de excelência de produtos, serviços e processos e posteriormente fazer as alterações necessárias para atingir esses padrões (Singh & Grover, 2013).

A metodologia de *Benchmarking* escolhida para este projeto é a disponibilizada pela *American Productivity and Quality Center* (APQC). Esta organização é líder mundial na área do *Benchmarking* e ajuda organizações a melhorar os seus processos e desempenho para obtenção de melhores resultados (APQC, 2017).

A metodologia aplicada encontra-se dividida em quatro fases, que são:

- **Fase 1: Planeamento –** Esta fase consiste no planeamento do projeto, estabelecer o âmbito do projeto, desenvolver a abordagem para recolha de dados e requisitos, e definir os critérios;
- **Fase 2: Recolher –** Nesta fase pretende-se que sejam recolhidos os dados estabelecidos no planeamento;
- **Fase 3: Analisar –** Consiste no processamento e análise dos dados. Nesta fase é necessário validar a informação recolhida para identificar os níveis de desempenho, indicadores e modelos de referência;
- **Fase 4: Adaptar –** Esta fase final consiste em desenvolver um plano de ação para a mudança com o intuito de melhorar a organização.

A Figura 1 apresenta um conjunto de 7 atividades que podem ser integradas nas 4 fases acima descritas da seguinte forma:

- As atividades 1 (Estabelecer a necessidade de *Benchmarking*), 2 (Identificar as funções a ser comparadas), e 3 (Selecionar os pontos de referência que se pretendem atingir) integram a fase 1;

- A atividade 4 (Recolher e analisar os dados para identificar as falhas no desempenho, processos e práticas) faz parte da fase 2;
- A atividade 5 (Estabelecer os ideais/objetivos que se pretendem atingir para melhorar e ultrapassar os melhores) integra a fase 3;
- As atividades 6 (Identificar as falhas e idealizar uma forma de as implementar) e a atividade 7 (Implementar o plano para corrigir as falhas e monitorizar os resultados) configuram a fase 4, adaptar.

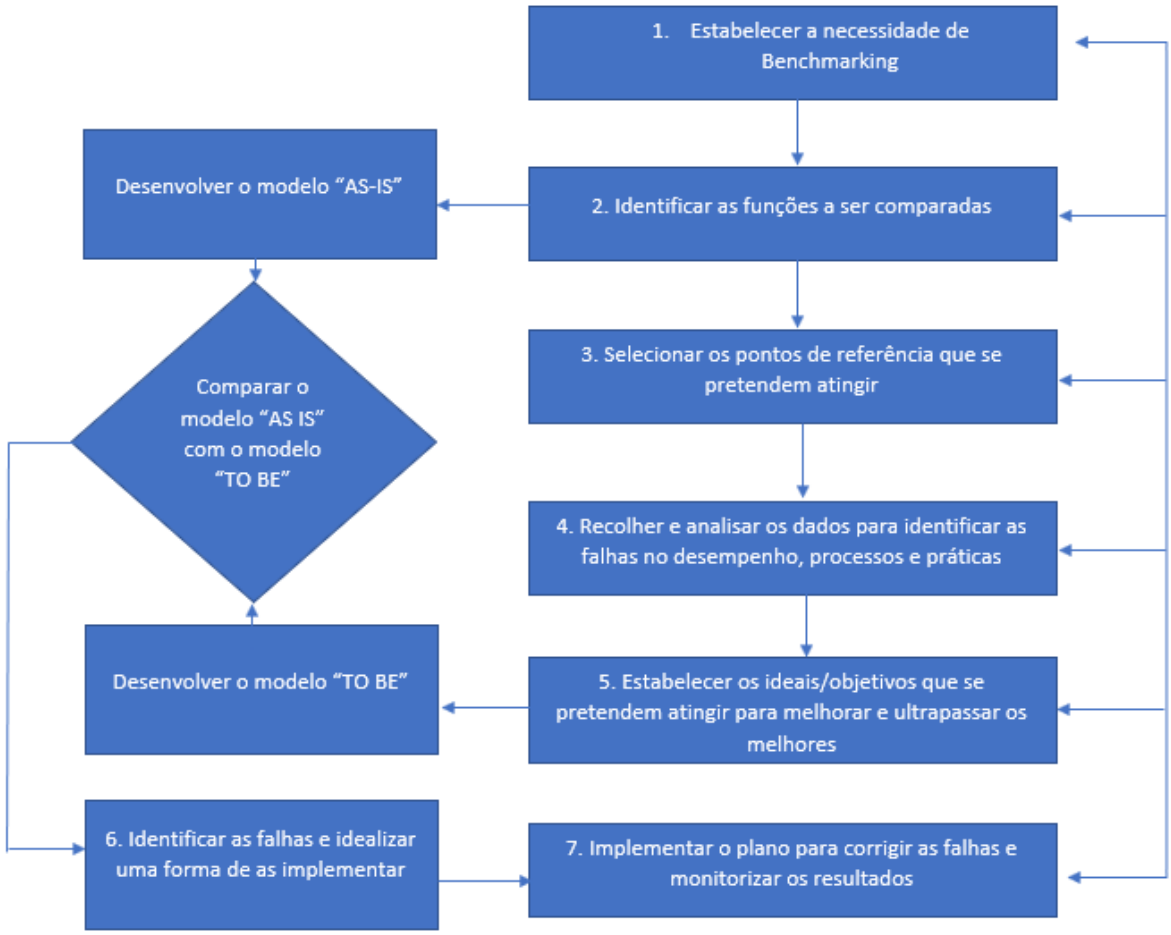

Figura 1 - Processo do Benchmarking. Adaptado de (Singh & Grover, 2013)

3.2 Aplicação da abordagem metodológica

3.2.1 Metodologia de Investigação – *Case Study*

Numa fase inicial e aplicando a metodologia de investigação foi elaborado um método de pesquisa que passou por identificar quais as fontes de dados a ser utilizadas (ver subcapítulo 2.1). Após identificadas as fontes de dados a utilizar, foi realizada a recolha de informação sobre os temas *Big Data* e *Big Data*

na saúde. Nesta fase foram analisados os desafios e potencial da implementação de *Big Data* na saúde e que soluções existiam atualmente. Após a pesquisa foram retirados todos os conceitos relevantes e toda a informação relativa às soluções encontradas.

A aplicação da metodologia *Case Study* passou pela análise de documentação científica, pesquisada em vários repositórios e de vários autores, com o intuito de explorar e desenvolver o tema *Big Data* na saúde, quais os riscos e quais os benefícios da utilização de tecnologias *Big Data* na saúde e também avaliar o estado da implementação de soluções *Big Data* na saúde.

3.2.2 Metodologia prática – *Benchmarking*

O objetivo consistiu em encontrar um conjunto de experiências que comparassem o desempenho de soluções *Big Data* semelhantes às soluções escolhidas para a fase de *benchmarking*.

Na fase 1 (Planear), foram definidas as seguintes palavras-chave para a pesquisa de experiências:

- *Comparison;*
- *Benchmarking;*
- *Performance comparison.*

As fontes de dados utilizadas para a pesquisa de experiências foram as seguintes:

- *Science Direct;*
- *Google Scholar;*
- *Ieeexplore;*

Para haver um maior rigor e credibilidade, a data da publicação das experiências não deveria exceder em mais de 5 anos a data da publicação dos artigos das soluções apresentadas e todas as experiências necessitam de aprovação da comunidade científica.

Posteriormente na fase 2 (Recolher) foram recolhidas as experiências existentes, com base na estratégia de pesquisa definida na fase 1. Aqui foram pesquisadas várias comparações e *benchmarkings* a soluções semelhantes às escolhidas para o *benchmarking*.

Na fase 3 (Analisar) foram analisadas as várias experiências encontradas. Dentro das encontradas, foram selecionadas as que mais se enquadravam com o tema desta dissertação e também com as aplicações escolhidas.

Por fim na fase 4 (Adaptar) foram identificadas as vantagens e desvantagens das soluções analisadas e também apresentadas um conjunto de sugestões que as organizações da área da saúde podem adotar

para reduzir o risco de investimentos em tecnologias *Big Data*. Esta fase termina com a implementação das sugestões.

4. FERRAMENTAS *BIG DATA*

Nesta secção vai ser apresentado um conjunto de ferramentas que integram as seguintes arquiteturas:

- Arquitetura Big Data para o projeto INTCare;
- Arquitetura que inclui *Apache Spark*;
- E Arquitetura *Maharaja Yeshwatrao*.

4.1.1 Apache Hadoop

Hadoop é um projeto *open source* da *Apache* escrito em *Java* baseado em computação distribuída. Uma das principais características do *Hadoop* é a redundância, como se pode ver na Figura 2, os blocos são armazenados e replicados pelos vários nós do *cluster*. Os dados são armazenados de forma redundante ao longo dos vários nós do *cluster* e o modelo de programação está preparado para lidar com falhas. Devido à redundância implementada, é possível distribuir os dados e a programação associada por todo o *cluster* (Zikopoulos et al., 2012).

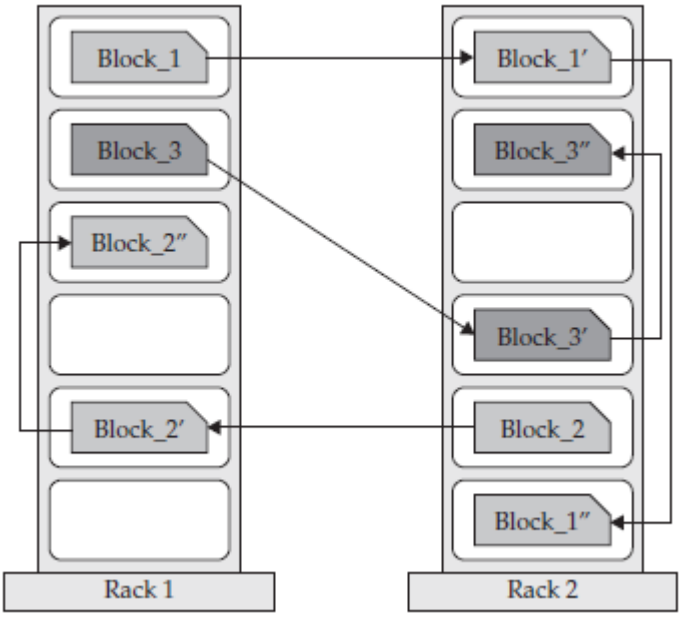

Figura 2- Exemplo de como os blocos de dados são escritos no HDFS. Retirado de (Zikopoulos et al., 2012).

Hurwitz et al. (2013) classifica o *Hadoop* como sendo capaz de se auto-regenerar. Dado que é implementado em *hardware* de comodidade, ou seja, componentes pouco dispendiosos e facilmente substituíveis, o *Hadoop* é capaz de detetar alterações, incluindo falhas, ajustar-se às mesmas e continuar a funcionar sem interrupções.

Segundo Hurwitz et al. (2013), o *Hadoop* tem dois componentes principais sendo eles o *Hadoop Distributed File System* (HDFS) e o *MapReduce*.

a) Hadoop Distributed File System (HDFS)

Quando o tamanho do *dataset* é maior que a capacidade de armazenamento de uma só máquina, torna-se necessário o particionamento do mesmo por várias máquinas. Sistemas de ficheiros que lidam com este tipo de problema são chamados de sistemas de ficheiros distribuídos (T. White, 2012).

O *Hadoop Distributed File System*, como o próprio nome indica, é um sistema de ficheiros distribuídos concebido para armazenar ficheiros por várias máquinas, para ser altamente tolerante a faltas e para ser implementado em *hardware* de baixo custo e facilmente substituível (Kala Karun & Chitharanjan, 2013).

Segundo Hurwitz et al. (2013), o HDFS não é o destino final dos ficheiros, mas sim um serviço que oferece as capacidades necessárias para lidar com grandes volumes de dados a grandes velocidades.

Os ficheiros em HDFS são particionados em blocos que são armazenados como unidades independentes. Esta abstração de blocos tem vários benefícios, dos quais, um ficheiro pode ser maior do que a capacidade de armazenamento de um qualquer disco rígido (singular) na rede um e o facto que nada especifica que, após o particionamento de um ficheiro em blocos, os vários blocos desse ficheiro sejam armazenados no mesmo disco rígido, o que permite tirar partido de todos os discos do *cluster*. Como foi explicado anteriormente, esta abstração de blocos facilita a redundância e permite aumentar a disponibilidade e a tolerância a faltas (T. White, 2012).

Este sistema de ficheiros possui dois tipos de nós, o "*NameNode*" e o "*DataNode*". Os "*DataNodes*" (existem vários) servem para armazenar os blocos enquanto o "*NameNode*" (apenas um) tem a responsabilidade de saber a localização dos blocos nos vários "*DataNodes*" que compõe um ficheiro (Hurwitz et al., 2013). A Figura 3 ilustra a arquitetura do HDFS.

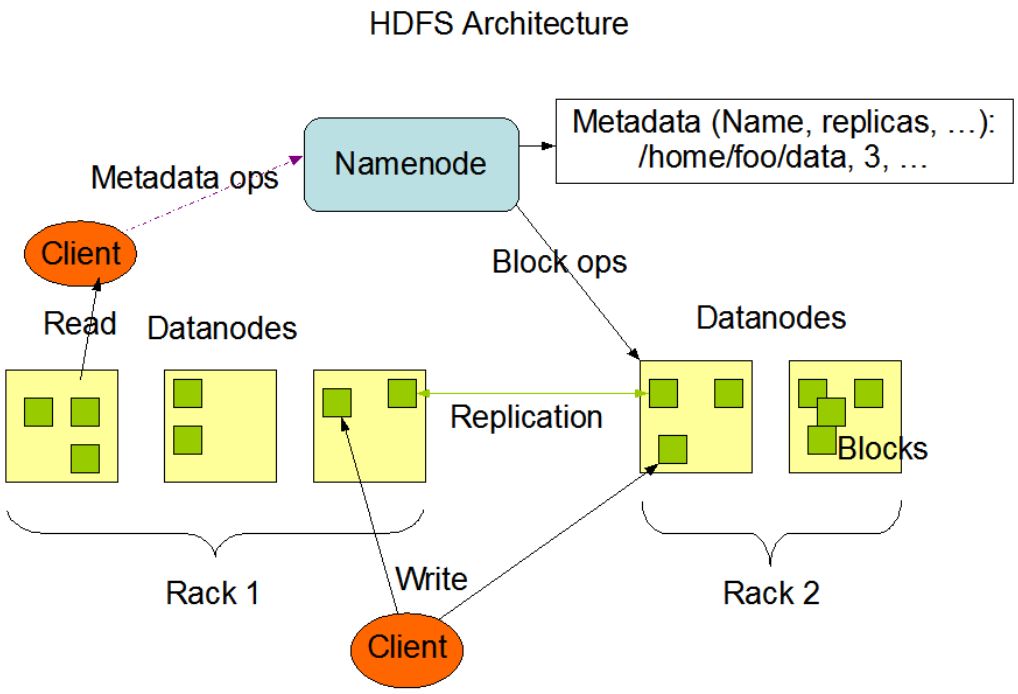

Figura 3- Arquitetura do HDFS. Retirado de (The Apache Software Foundation, 2013)

O "*NameNode*" é responsável por gerir o *namespace* (coleção de todos os ficheiros armazenados no *cluster*) do sistema de ficheiros; mantém a árvore do sistema de ficheiros e os metadados de todos os ficheiros e diretorias da árvore (T. White, 2012).

Os "*DataNodes*" são responsáveis por armazenar e fornecer os blocos quando são requisitados tanto pelos clientes como pelo "*NameNode*"; também reportam ao "*NameNode*" listas dos blocos que armazenam (T. White, 2012).

b) Hadoop MapReduce

MapReduce é um modelo de programação para processamento de dados que confere grande escalabilidade ao *cluster Hadoop*. Zikopoulos et al. (2012) afirma que o *MapReduce* é o coração do *Hadoop*.

MapReduce consiste em duas fases, a fase *map* e a fase *reduce*. A primeira fase, fase *map*, recebe um conjunto de dados e converte-o num outro conjunto de dados, onde os elementos individuais são particionados em pares *key-value* (chave-valor). A fase *reduce* recebe o *output* da fase *map* e combina esses pares em conjuntos de pares mais pequenos (Zikopoulos et al., 2012). A Figura 4 lustra o fluxo de dados em *MapReduce*. Como se pode observar, inicialmente temos as fontes de dados ("*Data Source*(*s*)") de onde são extraídos os dados ainda não processados ("*Raw Data*"), posteriormente ocorre a fase de mapeamento ("*Map*") e de redução ("*reduce*") explicadas acima.

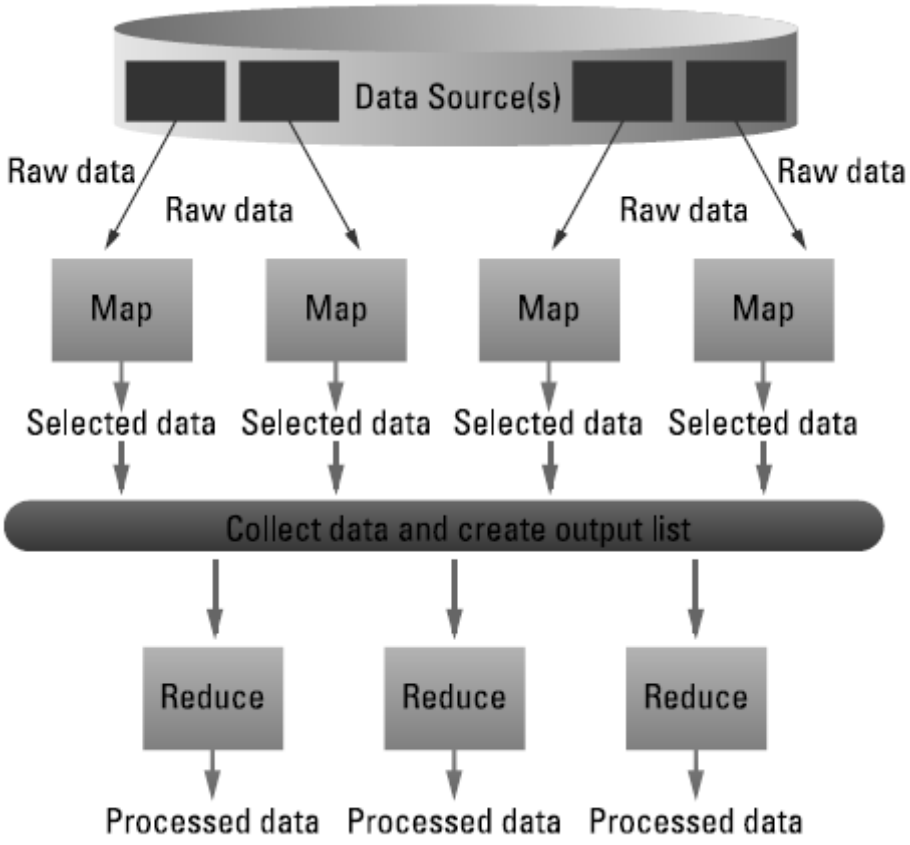

Figura 4 - Fluxo dos dados em MapReduce. Retirado de (Hurwitz et al., 2013)

c) Apache HadoopYARN (Yet Another Resource Negotiator)

O *YARN* é uma tecnologia de gestão de *cluster*. Esta tecnologia surgiu da necessidade de resolver algumas das limitações da primeira versão do *Hadoop* (*Hadoop 1.0*), como por exemplo, o *MapReduce* suportar, no máximo, 4000 nós; então, no *Hadoop 2.0* foi introduzido o *YARN* (Kulkami & Khandewal, 2014).

O objetivo do *YARN* é dividir as funcionalidades de gestão de recursos e de monitorização/ agendamento de tarefas (The Apache Software Foundation, 2016). A ideia consiste em ter um *ResourceManager* global, um *ApplicationMaster* por aplicação, um NodeManager por nó e um *Container* por aplicação a executar sobre um *NodeManager*. Na Figura 5, que representa a arquitetura do *YARN*, é possível observar a interação entre as várias entidades referidas.

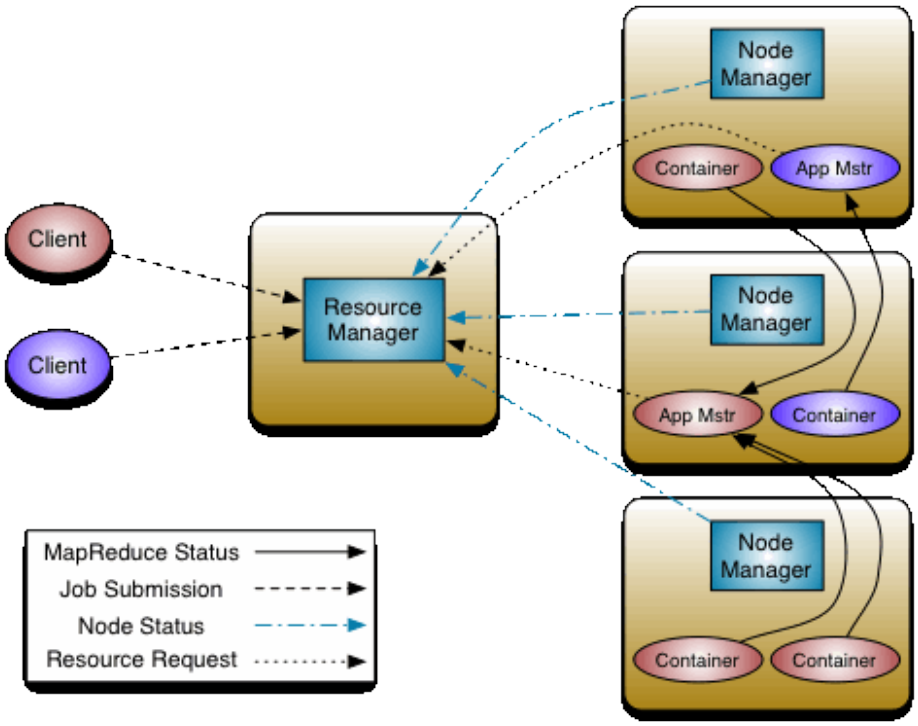

Figura 5 - Arquitetura do Apache Hadoop YARN. Retirado de (The Apache Software Foundation, 2016)

Os dois principais componentes do *ResourceManager* são (The Apache Software Foundation, 2016):

- ***Scheduler*** – responsável por alocar recursos às várias aplicações em execução, tendo em conta as restrições de capacidades de *queues*, limites de utilizador, entre outros. O *Scheduler* não efetua qualquer tipo de monitorização ou rastreamento de estado e também não oferece qualquer garantia em caso de tarefas falhadas. O agendamento de processos é feito tendo em conta os requisitos de recursos das aplicações;
- ***ApplicationsManager*** – responsável por aceitar submissões de tarefas, negociando o primeiro *Container* para executar o *ApplicationMaster* especifico de uma aplicação e fornece também o serviço para reiniciar o *Container* em caso de falha.

O *NodeManager* gere as dependências do *Container*, monitoriza a sua execução e fornece-lhe um conjunto de serviços (Vavilapalli et al., 2013).

4.1.2 Apache Hive

O *Apache Hive* é uma solução de *data warehousing* de código aberto que assenta sobre o *Hadoop*. Esta tecnologia suporta consultas expressas numa linguagem declarativa semelhante ao *SQL* (*Structured Query Language*), conhecida por *HiveQL*. As consultas são compiladas em tarefas *map-reduce*

executadas em *Hadoop*. As *scripts* de *map-reduce* podem ser incorporadas nas *queries HiveQL* (Thusoo et al., 2009).

Os dados no *Hive* estão organizados da seguinte forma (Thusoo et al., 2009):

- Tabelas – Cada tabela possui uma diretoria HDFS. Os dados de uma tabela são armazenados em arquivos do diretório correspondente;
- Partições – Cada tabela pode ter uma ou mais partições que determinam a distribuição dos dados nas subdiretorias da partição que corresponde à tabela;
- *Buckets* (Baldes) – Para cada partição, os dados podem ser divididos em baldes com base no *hash* de uma coluna de uma tabela. Cada balde é armazenado como um ficheiro no diretório da partição.

Na Figura 6 encontra-se representado os principais componentes do *Apache Hive* e as interações entre os mesmos.

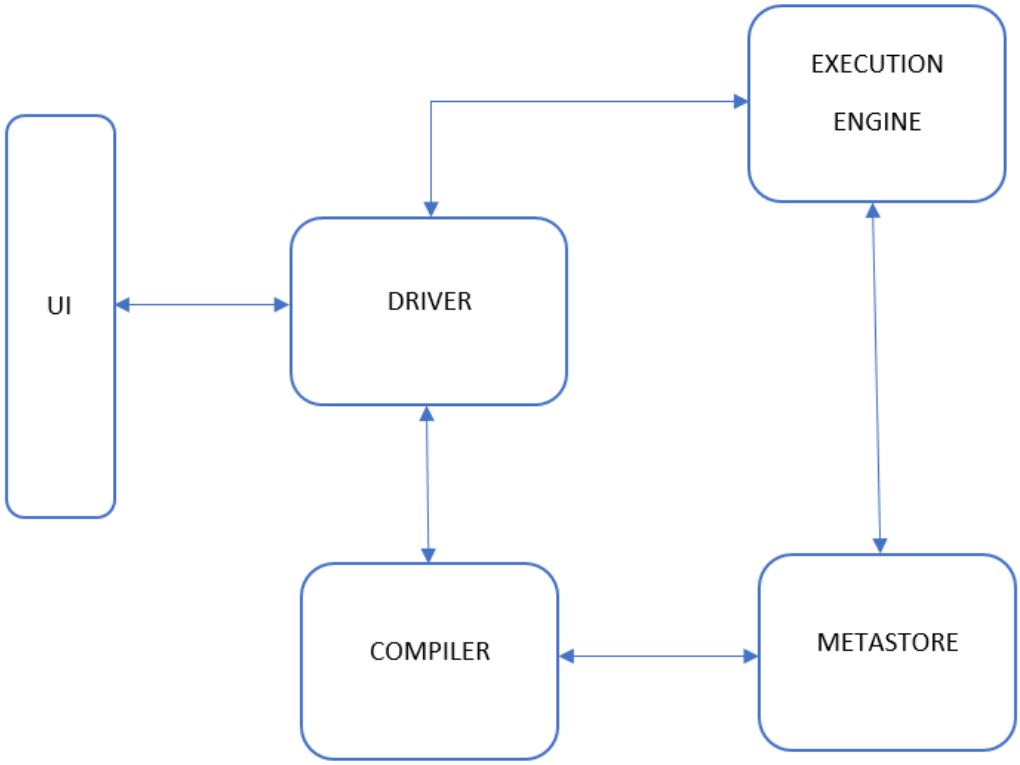

Figura 6 - Principais componentes do Apache Hive. Adaptado de (The Apache Software Foundation, 2015c)

Como se pode ver na Figura 6, os principais componentes do *Apache Hive* são (The Apache Software Foundation, 2015c):

- O componente UI (*user interface*), é a *interface* que permite aos utilizadores fazer consultas e outras operações ao sistema;

- O *Driver* é o componente que recebe as *queries*. Este componente manipula as sessões e fornece APIs de execução e de procura;
- O *Compiler* é o componente que analisa a *query*, faz a análise semântica nos diferentes *query blocks* e *query expressions* e gera um plano de execução com recurso aos metadados da tabela e da partição, cuja informação está armazenada no componente *metastore*;
- O *Metastore* é o componente que armazena toda a informação da estrutura das várias partições e tabelas do *data warehouse*, incluindo informações acerca da coluna e tipo de informação nela contida, os serializadores e desserializadores necessários para ler e escrever os dados e os ficheiros HDFS correspondentes, onde os dados são armazenados;
- O componente *Execution Engine* é responsável por executar o plano de execução criado pelo *Compiler*. Este componente gere as dependências entre as diferentes fases do plano e executa essas mesmas fases nos componentes do sistema apropriados.

4.1.3 Apache Spark

O *Apache Spark* é uma *framework* de processamento *batch* com capacidade para processamento *stream* (em fluxo) (The Apache Software Foundation, 2017i).

No que diz respeito à velocidade, o *Apache Spark* estende o *MapReduce*, o que permite um suporte mais eficiente de mais tipos de computação, como por exemplo, *interactive queries* (consultas interativas) e *stream processing* (processamento em tempo real ou quase tempo real). O *Spark* foi idealizado para ser capaz de processar grandes volumes de dados a grandes velocidades já que isto pode significar ter a análise de dados em tempo real, ou quase tempo real ou ter de esperar minutos ou horas. A redução de velocidade de processamento do *Spark* face ao *Hadoop* provém do facto de o *Spark* trabalhar maioritariamente em memória, reduzindo assim o tempo de operações de leitura e escrita nos discos. Os resultados intermédios do processamento dos dados é armazenado em memória (Yu, Gill, Dalal, Jha, & Shah, 2015).

As aplicações *Spark* executam como conjuntos de processos independes no *cluster*, são coordenados pelo objeto *SparkContext* no *main* do programa (chamado de *driver program*).

Para executar num *cluster*, o *SparkContext* pode-se conectar a vários tipos de *cluster manager* (ou o gestor de cluster autónomo da *Spark*, *Mesos* ou *YARN*), que alocam recursos entre as várias aplicações. Quando conectado, o *Spark* adquire processos que executam cálculos e armazenam dados para a aplicação - chamados *executors*. Posteriormente, envia o código da aplicação para os *executors* e no final, o *SparkContext* envia *tasks* (tarefas) para os *executors* executarem. Este exemplo ilustra o

funcionamento do *Spark* em *cluster* e pode ser observado na Figura 7 (The Apache Software Foundation, 2016b).

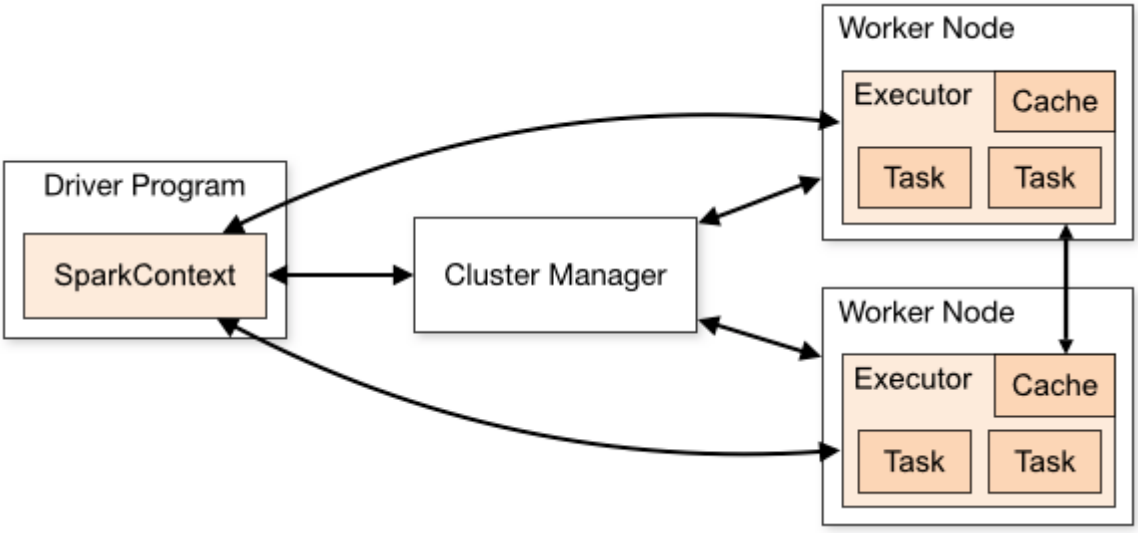

Figura 7- Arquitetura Spark. Retirado de (The Apache Software Foundation, 2016b)

O mecanismo de tolerância a falhas que o *Apache Spark* utiliza são *RDDs* (*Resilient Distributed Datasets*). Um *RDD* é uma coleção de objetos fracionada por várias máquinas que pode ser recuperado caso uma partição se perca (Zaharia, Chowdhury, Franklin, Shenker, & Stoica, 2010).

1) Componentes

Na Figura 8 estão representados os vários componentes do *Spark*.

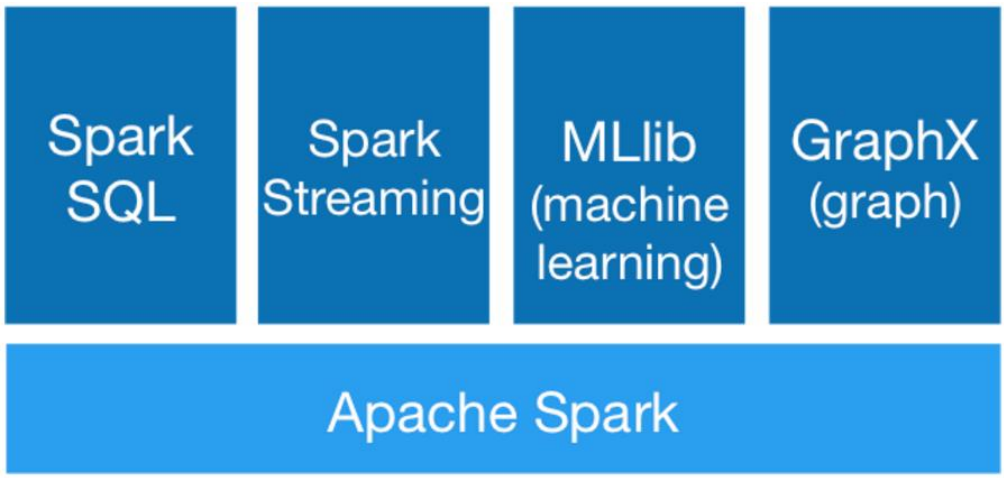

Figura 8 - Componentes Apache Spark. Retirado de (The Apache Software Foundation, 2017d)

a) Spark SQL

O *Spark SQL* é um módulo do *Apache Spark* para processamento de dados estruturados. A interação com o *Spark SQL* pode ser feita através de *SQL* e da *API (Application programming interface) Dataframe*. A utilização de *SQL* e da *API DataFrame* permitem não só aceder a dados de várias fontes - *Apche Hive, JSON, Apache Avro*, entre outras – mas também agregar dados destas.(The Apache Software Foundation, 2016c).

b) *Spark Streaming*

O *Spark Streaming* permite o processamento de um fluxo contínuo de dados (*streams* de dados) provenientes de várias fontes como se pode ver na Figura 9 (The Apache Software Foundation, 2016d).

Figura 9 - Interações do Spark Streaming. Retirado de (The Apache Software Foundation, 2016d)

A Figura 10 demonstra o funcionamento do *Spark Streaming*.

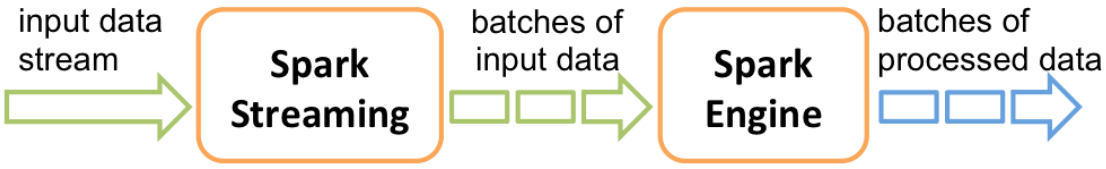

Figura 10 - Funcionamento do Spark Streaming. Retirado de (The Apache Software Foundation, 2016d)

O *Spark Streaming* após receber o fluxo de dados (*input*) divide os mesmos em lotes (*batch*) que são processados pelo *Spark*. O resultado deste processamento é um fluxo de dados em lotes (The Apache Software Foundation, 2016d).

c) MLlib

MLlib é a biblioteca de *machine learning (ML)* do *Apache Spark*. Esta biblioteca pode utilizar qualquer fonte de dados do ecossistema *Hadoop*, *HDFS, HBase,* entre outros (The Apache Software Foundation, 2017k).

d) GraphX

O *GraphX* é uma *framework* para processamento de grafos distribuídos que assenta sobre o *Apache Spark*. Fornece uma API para computação de grafos que pode modelar grafos definidos pelo utilizador (The Apache Software Foundation, 2017h).

4.1.4 Apache HBase

O *HBase*, um projeto de código aberto (*open source*) da *Apache*, é uma base de dados *NoSQL* distribuída, tolerante a faltas, altamente escalável, orientada em colunas e assenta sobre o HDFS (Vora, 2011). *NoSQL* é o termo utilizado para as base de dados que não suportam *SQL* como linguagem de acesso primário (The Apache Software Foundation, 2017f).
As soluções *NoSQL* estão divididas em grupos, sendo os principais: (Alexandre & Cavique, 2013):

- Armazenamento de pares chave/valor (*key/value);*
- Armazenamento de super colunas;
- Armazenamento de documentos, como *XML (eXtensible Markup Language) database* ou *MongoDB;*
- Armazenamento de grafos;
- Armazenamento orientado a objetos.

Os componentes do *HBase* são (The Apache Software Foundation, 2017g):

- **Tabela** – a informação é organizada em tabelas;
- **Linha** – uma linha no *HBase* consiste na chave da linha (*row key*) de uma ou mais colunas com valores associados às mesmas. As linhas são ordenadas alfabeticamente pela chave da linha com o objetivo de armazenar os dados de forma as linhas relacionadas estejam próximas umas das outras;
- **Coluna** – uma coluna no *HBase* consiste na família de coluna e no quantificador da coluna que são delimitadas pelo caractere ":";
- **Família de coluna** – os dados de cada linha são agrupados pela família de coluna. Estas têm de ser definidas aquando da criação das tabelas e não são facilmente modificáveis. Cada família de coluna possui um conjunto de propriedades de armazenamento;

- **Qualificador de coluna** – um qualificador de coluna é adicionado a uma família de coluna para fornecer um índice para um determinado dado. Dado o conteúdo de uma família de colunas, um qualificador de coluna pode ter os seguintes formatos: *content:html* e *content:pdf*. Apesar das famílias de colunas serem definidas no momento da criação da tabela, os qualificadores de colunas são mutáveis e podem diferir entre as linhas;
- **Célula** – uma célula é uma combinação da linha, da família de colunas e do qualificador de coluna e contém um valor e um registo da data/hora (*timestamp)* que representa a versão do valor;
- **Versão (*timestamp*)** – o registo data/hora é escrito para cada valor e é o identificador de uma determinada versão de um valor.

4.1.5 Apache Pig

O *Apache Pig* é uma plataforma que permite analisar grandes volumes de dados. Esta é constituída por dois componentes, um compilador que produz sequências de programas *Map-Reduce* e uma linguagem, *PigLatin* (The Apache Software Foundation, 2017n).

Uma das principais caraterísticas da linguagem *PigLatin* é o facto de simplificar a escrita de programas *MapReduce* (The Apache Software Foundation, 2017n).

4.1.6 Apache Zookeeper

O *Apache Zookeeper* é um serviço *open source* de coordenação distribuída projetado para aplicações distribuídas. O *Zookeeper* surgiu da necessidade de facilitar a implementação de serviços de coordenação, assim, este serviço liberta as aplicações distribuídas da responsabilidade de implementar serviços de coordenação de raiz (The Apache Software Foundation, 2016f).

4.1.7 Apache Chukwa

O *Apache Chukwa* é um sistema *open source* de recolha de registos (*logs*) concebido para monitorizar sistemas distribuídos de grande escala. Este sistema assenta sobre o HDFS e sobre a *framework MapReduce* herdando assim as mesmas propriedades de escalabilidade e robustez (The Apache Software Foundation, 2016e).

4.1.8 Apache Avro

O *Apache Avro* é um sistema de serialização de dados que fornece serviços de serialização e troca de dados para *Hadoop,* estes serviços podem ser usados em conjunto ou em separado.

O serviço de serialização permite que os programas possam serializar dados em ficheiros ou mensagens de forma eficiente. Este sistema armazena a informação que define os dados e os dados juntos num arquivo ou mensagem *Avro*, o que facilita a compreensão das informações armazenadas. A informação que define os dados é armazenada no formato *JSON* (*JavaScript Object Notation*) e os dados em si são armazenados em binário (The Apache Software Foundation, 2017a).

4.1.9 Apache Kafka

O *Apache Kafka* é uma plataforma distribuída de processamento *streaming*. Esta ferramenta age como um sistema de mensagens, é rápido, tolerante a falhas e escalável. O *Kafka* permite criar fluxos de dados *streaming* que permitem a troca de dados entre sistemas e aplicações, possibilitando também desenvolver aplicações que reagem, em tempo real, a dados *streaming* (The Apache Software Foundation, 2015a).

4.1.10 Apache Storm

O *Apache Storm* é um sistema distribuído de processamento de dados em tempo real (The Apache Software Foundation, 2015b). Segundo W. Liu & Park (2014) é possível utilizar o *Apache Storm* com qualquer linguagem de programação, sendo que as mais utilizadas são o *Java* e *Python*.

5. SOLUÇÕES BIG DATA NA SAÚDE

Nesta secção vai ser apresentado um conjunto de soluções *Big Data*. Para facilitar a adoção de tecnologias *Big Data* na saúde, todas soluções apresentadas foram aprovadas pela comunidade científica.

5.1 Arquitetura Big Data para o projeto INTCare

O *INTCare* é um projeto de investigação desenvolvido na Unidade de Cuidados Intensivos (UCI), do Centro Hospitalar do Porto, que, numa fase inicial, foi projetado para desenvolver um sistema inteligente para prever falhas de órgãos e os efeitos causados por estas nos utentes (F. Portela et al., 2014).

Segundo Portela et al. (2014), em 2009 tornou-se notória a quantidade excessiva de registos médicos em papel ou inseridos manualmente na base de dados. Após um conjunto de estudos efetuados com o objetivo de identificar quais as lacunas no sistema de informação da UCI, foi possível desenvolver uma nova solução baseada em sistemas inteligentes capazes de realizar tarefas automáticas como aquisição e processamento de dados (F. Portela et al., 2014).

Atualmente, o *INTCare* é um *Pervasive Intelligent Decision Support System* (*PIDSS*) que atua de forma automática e em tempo-real, com o propósito de fornecer nova informação às entidades responsáveis pela tomada de decisão na UCI, ou seja, médicos e enfermeiros (F. Portela et al., 2014).

O projeto *INTCare* utiliza um fluxo contínuo e em tempo real de recolha de dados, provenientes de vários sensores dispositivos de monitorização (Gonçalves, 2017) que citou (Portela, Santos, Silva, Machado, & Abelha, 2011) o que gera um volume de dados compreendido entre 50 a 500 *Terabytes* (Gonçalves, 2017) que citou (Lima, 2014).

Segundo Gonçalves (2017) aplicando a *framework BigDAF* apresentada no subcapítulo 2.4 obtém-se um nível de complexidade igual a 320 ((60*3) + (10*5) + (30*3)) o que classifica o projeto *INTCare* como um *Big Data Issue*.

Após identificada a necessidade da utilização de tecnologias *Big Data* no projeto *INTCare*, Gonçalves (2017) apresentou uma solução baseada no *Hadoop*, mais propriamente na distribuição *Hortonworks* como se pode ver na Figura 11. A *Hortonworks* é uma organização que desenvolve e suporta projetos *open source*, tais como o *Hadoop*, *Spark*, entre outros (Hortonworks Inc., 2017).

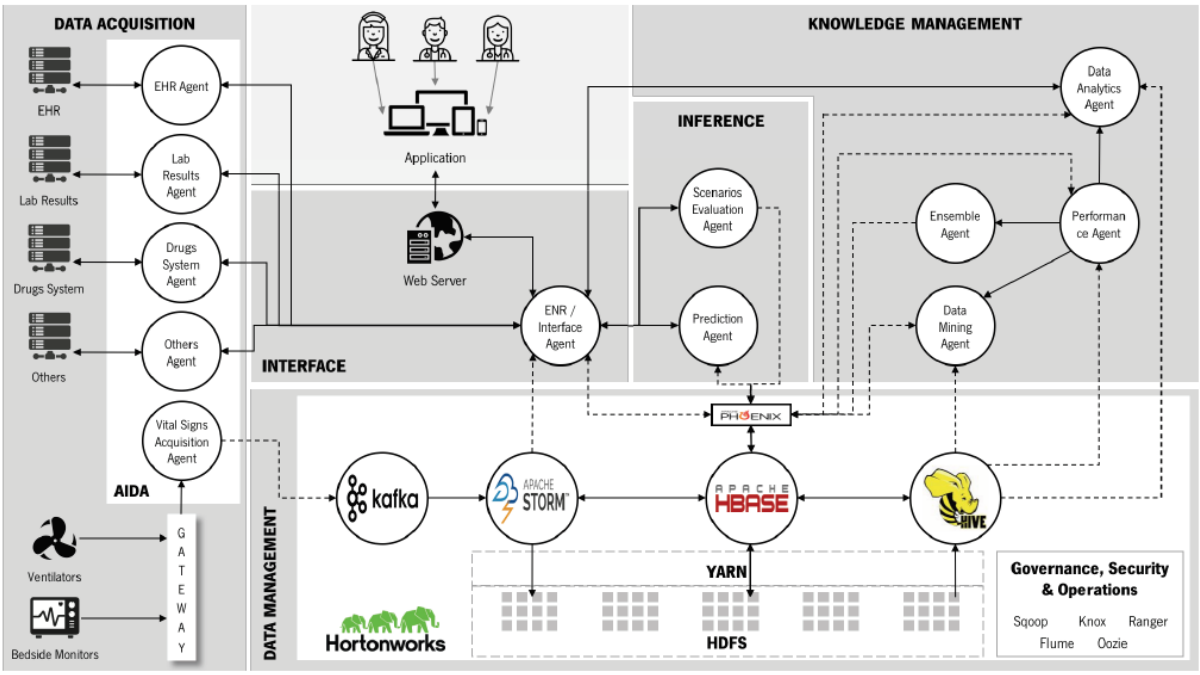

Figura 11 - Arquitetura Big Data para o projeto INTCare. Retirado de (Gonçalves, 2017)

Como se pode observar pela Figura 11, o subsistema *Data Management* conta com o *Apache Kafka* e *Apache Storm* para processamento de dados *streaming*. Para processamento de dados operacionais conta com o *Apache Phoenix e Apache HBase*. O processamento de dados analíticos é assegurado pelo *Apache Hive.* A segurança, administração e operações são asseguradas pelos seguintes subprojectos *Hadoop:*

- ***Apache Sqoop*** - ferramenta projetada para transferir enormes quantidades de dados entre o *Hadoop* e bases de dados relacionais (The Apache Software Foundation, 2017e)*;*
- ***Apache Knox*** - fornece mecanismos de segurança para que seja possível às organizações expandir, com confiança, o acesso de novos utilizadores ao ecossistema *Hadoop. O knox gateway* fornece um único ponto de acesso para todas as interaçõe*s REST (Representational State Transfer) e HTTP (Hypertext Transfer Protocol)* com o *cluster Hadoop* (The Apache Software Foundation, 2017l)*;*
- ***Apache Ranger*** - *framework* que fornece uma abordagem compreensiva a segurança de um *cluster Hadoop.* Disponibiliza uma plataforma centralizada para definir, e gerir as politicas de segurança do ecossistema *Hadoop* (The Apache Software Foundation, 2017c)*;*
- ***Apache Flume*** - serviço distribuído para recolher, agregar e mover para o *HDFS*, grandes quantidades de dados relativos a registos (*logs*) (The Apache Software Foundation, 2017m)*;*

- ***Apache Oozie*** – sistema de agendamento de processos do Hadoop (The Apache Software Foundation, 2017b).

A Tabela 3 auxilia a uma melhor compreensão da arquitetura do projeto *INTCare*.

Tabela 3 - Subsistemas da arquitetura INTCare. Adaptado de (Gonçalves, 2017)

Subsistema	**Funcionalidades**
Aquisição de Dados (*Data Acquisition*)	Este subsistema é responsável pela aquisição de dados provenientes de monitores de cabeceira (medidor de sinais vitais, ventiladores e outros), da Folha de Enfermagem Eletrónica (*ENR*), do Processo Clínico Eletrónico (*EHR*), do Sistema Farmacêutico (*Drug System*) e das Análises Clinicas (*Lab Results*).
Gestão de Conhecimento (*Knowledge Management*)	Composto por três agentes: *Data Mining*, *Performance* e *Ensemble*. O agente de *Data Mining* converte, em conhecimento, os dados registados no *data warehouse*, através da criação de modelos de previsão em tempo-real. A captação de dados estatísticos é realizada pelo agente *Performance*. Por último, o agente *Ensemble* combina vários modelos com o objetivo de melhorar o desempenho preditivo.
Inferência (*Inference*)	Constituído por três agentes, o *Prediction, Data Retrieval e Scenario Evaluation. O Prediction* é utilizado para responder a questões dos utilizadores, aplicando os modelos adequados contidos no *Knowledge Management*. A informação solicitada pelo *Interface* é processada pelo agente *Data Retrieval*. O agente *Scenario Evaluation* permite aos médicos criar e avaliar cenários hipotéticos.
Interface	Fornece aos médicos uma plataforma *web* para avaliar cenários e requisitarem prognósticos.

5.2 Arquitetura que inclui *Apache Spark*

Liu et al. (2015), propuseram uma arquitetura para uma ferramenta de processamento de *Big Data* na saúde que inclui o *Apache Spark,* como se pode ver na Figura 12.

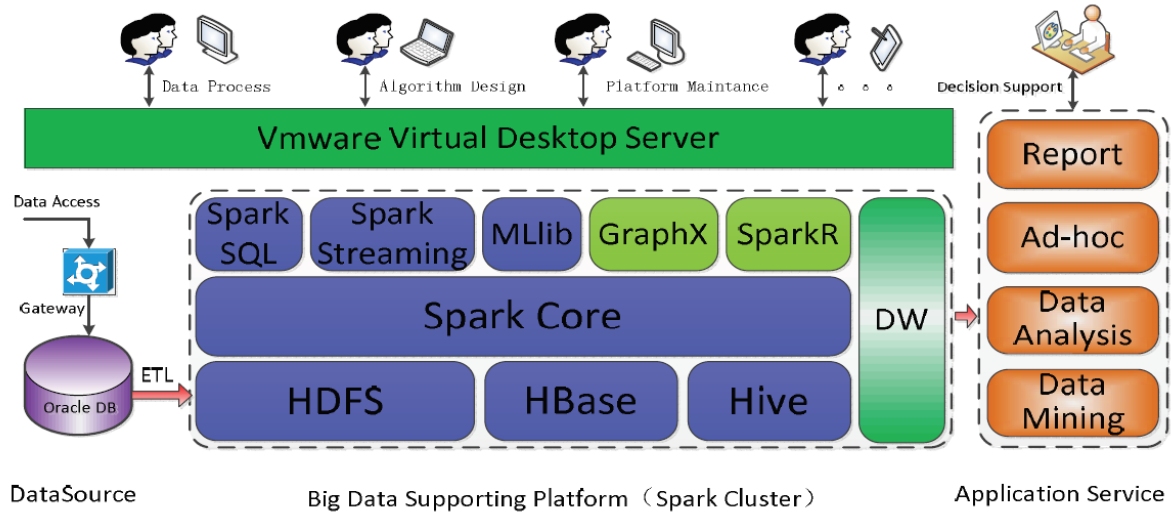

Figura 12 - Protótipo de um sistema de processamento de Big Data aplicado à área da saúde. Retirado de (Liu et al., 2015)

Segundo Liu et al. (2015) muitos projetos *Big Data* devem tomar como referência uma arquitetura que consista em quatro módulos – sistema de armazenamento distribuído, plataforma de computação distribuída, *application engine* ("motor da aplicação") e apresentação dos dados. O sistema de armazenamento distribuído é um sistema altamente tolerante a faltas, projetado para ser executado em *hardware* de comodidade, que fornece altas taxas de transferências a dados de aplicações, como *HDFS* e *HBase*. A plataforma de computação distribuída é um modelo de programação para processamento de grandes volumes de dados, neste caso a plataforma utilizada é o *Apache Spark*. O motor de aplicação é uma interface entre a aplicação e os dois módulos inferiores, neste caso, *Apache Hive*, *Spark SQL*, *Spark Streaming*, *MLib* e *GraphX*. A apresentação dos dados é a janela onde serão apresentados os resultados do processamento.

Os componentes da arquitetura apresentada na Figura 12 estão descritos na Tabela 4 divididos pelas respetivas camadas. Como se pode observar pela Tabela 4, a camada de "Armazenamento dos dados", é constituída pelo *Hadoop Distributed File System* explicado no subcapítulo 4.1.1, e também pelo *Apache HBase* explicado no subcapítulo 4.1.4. A camada "Processamento dos dados" contém o *Apache Spark* e *Spark Streaming* ambos apresentados no subcapítulo 4.1.3. O *Apache Hive* e *Spark Sql* apresentados nos subcapítulos 4.1.2 e 4.1.3 respetivamente, fazem parte da camada "Acesso aos dados". Por fim, a

camada "Analítica e *Business Intelligence*" contém as ferramentas *MLib, GraphX* e *SparkR* apresentadas no subcapítulo 4.1.3.

Tabela 4 - Componentes da Arquitetura que inclui o Apache Spark.

Camada	**Ferramenta**
Armazenamento dos dados	*HDFS*
	Apache HBase
Processamento dos dados	*Apache Spark*
	Spark Streaming
Acesso aos dados	*Apache Hive*
	Spark Sql
Analítica e *Business Intelligence*	*MLib*
	GraphX
	SparkR

5.3 Arquitetura *Maharaja Yeshwatrao*

Ojha & Mathur (2016) propuseram a arquitetura apresentada na Figura 13 para dar resposta às necessidades do hospital *Maharaja Yeshwatrao* situado em Indore na Índia que é considerado pelos autores como o maior hospital público da India central.

Ojha & Mathur (2016) afirmam que este hospital gera grandes volumes de dados baseando-se na quantidade de cidadãos que o frequenta diariamente. Com a implementação de *Big Data*, os autores pretendem melhorar a qualidade de vida dos utentes, principalmente dos mais carenciados, uma vez que, tempos de espera demorados têm um impacto negativo na população mais pobre pois obriga os cidadãos a perder dias de trabalho e consequentemente perder uma porção do seu salário. Para dar resposta aos longos tempos de espera, Ojha & Mathur (2016) pretendem armazenar os dados dos utentes como *Electronic Health Record* (*EHR*). Um *EHR* é um registo de saúde de um utente em formato digital, feito de forma sistemática. Os EHR podem conter informação demográfica, historial médico, histórico de medicação, alergias, análises, entre outros parâmetros. Com recurso ao *Big Data* e às

ferramentas de análise de dados, será possível armazenar os dados que o hospital *Maharaja Yeshwatrao* gera e, por conseguinte, os profissionais de saúde poderão encontrar novo conhecimento, padrões escondidos e tendências o que poderá resultar numa melhoria dos tratamentos, redução de readmissões, redução dos gastos, entre outros. Os componentes da arquitetura estão apresentados na Figura 13 divididos pelas respetivas camadas.

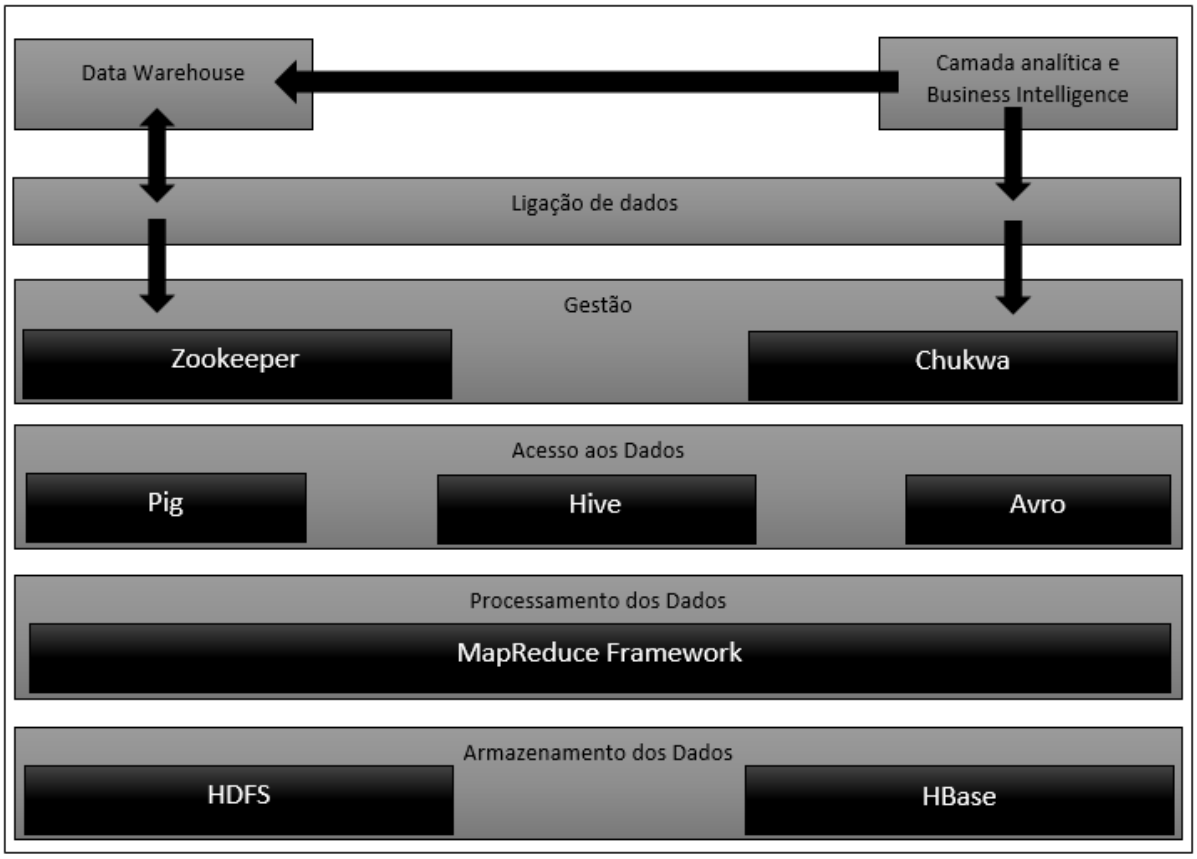

Figura 13- Descrição da arquitetura proposta para o hospital Maharaja Yeshwatrao. *Adaptado de* (Ojha & Mathur, 2016)

Como se pode observar na Figura 13, a camada de "Armazenamento dos dados", é composta pelo *HDFS* explicado no subcapítulo 4.1.1, e também pelo *Apache HBase* explicado no subcapítulo 4.1.4. A camada "Processamento dos dados" contém o *Hadoop MapReduce* apresentado no subcapítulo 4.1.1. O *Apache Hive, Apache Pig* e *Apache Avro* apresentados nos subcapítulos 4.1.2, 4.1.5 e 4.1.8 respetivamente, fazem parte da camada "Acesso aos dados". A camada "Gestão" é constituída pelo *Apache Zookeeper* apresentado no subcapítulo 4.1.6 e pelo *Apache Chukwa* explicado no subcapítulo 4.1.7. Por fim, Ojha & Mathur (2016) não apresentam ferramentas que permitam fazer a ligação entre a "camada analítica e *Business Intelligence* " e o *Data Warehouse* com a restante solução.

5.4 IBM PureData Solution For Healthcare Analytics

Esta arquitetura é constituída pela solução *IBM PureData Solution for Healthcare Analytics* que está a ser utilizada no *Seattle's Children's Hospital* para melhorar a capacidade de diagnóstico e assistência ao utente (Krishnan, 2016).

O *IBM PureData Solution for Healthcare Analytics* é uma solução desenvolvida pela IBM que integra várias tecnologias com o objetivo de dar resposta às necessidades *Big Data* de uma organização que atua na área da saúde.

Esta solução conta com os seguintes componentes:

- *IBM Cognos Business Intelligence* – Pacote de *Business Intelligence;*
- *IBM PureData System* – Sistema altamente escalável que conta com servidores, bases de dados, armazenamento, entre outros;
- *IBM Healthcare Provider Data Model* – Conjunto de modelos de dados e de modelos de soluções de negócio;
- *IBM InfoSphere Information Server for DataWarehouse* – Plataforma de integração de dados que suporta a captura, integração e transformação de grandes volumes de dados estruturados ou não estruturados (IBM, 2013).

A solução *IBM PureData System* vai ser apresentada com maior detalhe pois é o fator que diferencia esta solução das restantes apresentadas.

O *IBM PureData System* é um produto comercializado pela *IBM* que utiliza a tecnologia *Netezza.* O *IBM PureData System for Analytics* é um sistema escalável e de processamento altamente paralelo (*MPP – Massive parallel processing*). Este sistema integra bases de dados, servidores, armazenamento e capacidades de análise avançada num único sistema de fácil gestão (IBM, 2012).

Este sistema contém o seguinte *software* (IBM, 2016):

- Base de Dados - *IBM Netezza Platform Software (NPS) v7.2 (ou superior);*
- Sistema Operativo - *Red Hat Linux Advanced Server 6.5;*
- *IBM Fluid Query* 1.0 – Unifica o acesso aos dados no *data darehouse* (armazém de dados*).* Faz o encaminhamento da *query* (ou parte da *query*) para o armazenamento correto.
- APIs suportadas – *SQL, OLE DB, ODBC 3.5, JDBC 3.0 Type 4;*
- *SQL Standards – Compatível com SQL-92*, com extensões *SQL-99;*
- Linguagens de programação – *Java, Python, Open Source R, R3, Fortran, C/C++, Perl, Lua;*

O *hardware* presente neste sistema será apresentado na Tabela 5.

Tabela 5 - Componentes de um sistema IBM PureData System. Retirado de: (IBM, 2012).

Especificações	**Sistemas com uma prateleira**			**Sistemas com várias prateleiras**		
IBM PureData System for Analytics	*IBM PureData System for Analytics N3001-002*	*IBM PureData System for Analytics N3001-005*	*IBM PureData System for Analytics N3001-010*	*IBM PureData System for Analytics N3001-020*	*IBM PureData System for Analytics N3001-040*	*IBM PureData System for Analytics N3001-080*
Racks (Prateleiras)	*1*	*1*	*1*	*2*	*4*	*8*
Active S-Blades	*2*	*4*	*7*	*14*	*28*	*56*
CPU cores	*40*	*80*	*140*	*280*	*560*	*1,120*
FPGA cores	*32*	*64*	*112*	*224*	*448*	*896*
User data in TB (assumes 4X compression)	*32*	*96*	*192*	*384*	*768*	*1,536*
Power (Watts maximum)/ rack	*3,200*	*4,200*	*7,600*	*7,600*	*7,600*	*7,600*
Cooling - BTU/hour	*11,000*	*14,400*	*27,000*	*54,000*	*108,000*	*216,000*
Rack Weight Kg (Peso da prateleira)	*620*	*771*	*907*	*907*	*907*	*907*
Height/rack cm (Altura da prateleira)	*202*	*202*	*202*	*202*	*202*	*202*
Depth/rack cm (Profundidade da prateleira)	*110*	*110*	*110*	*110*	*110*	*110*
Width/rack cm (largura da prateleira)	*64.8*	*64.8*	*64.8*	*64.8*	*64.8*	*64.8*
Power	*200-240 V, 50Hz/60 Hz (Single phase), 24A*	*200-240 V, 50Hz/60 Hz (Single phase), 24A*	*200-240 V, 50Hz/60 Hz (Single phase), 2X 24A*	*200-240 V, 50Hz/60 Hz (Single phase), 2X 24A, per rack*	*200-240 V, 50Hz/60 Hz (Single phase), 2X 24A, per rack*	*200-240 V, 50Hz/60 Hz (Single phase), 2X 24A, per rack*

Snippet blades (S-Blades), como se pode observar na Tabela 5, são placas de processamento especializadas que combinam o poder de processamento do *CPU* (*Central Processing Unit*) de um servidor *blade* (um servidor *blade* combina componentes de um servidor, como processador, memória

e conexão à rede, numa placa de expansão), com a inteligência de análise da placa *IBM® Netezza® Database Accelerator* (IBM, 2014).

5.5 Cisco Connected Health Solutions and Services

A infraestrutura desenvolvida pela *Cisco Systems, Inc.*, integra vários serviços numa única solução que possa satisfazer "todas" as necessidades de uma organização na área da saúde.

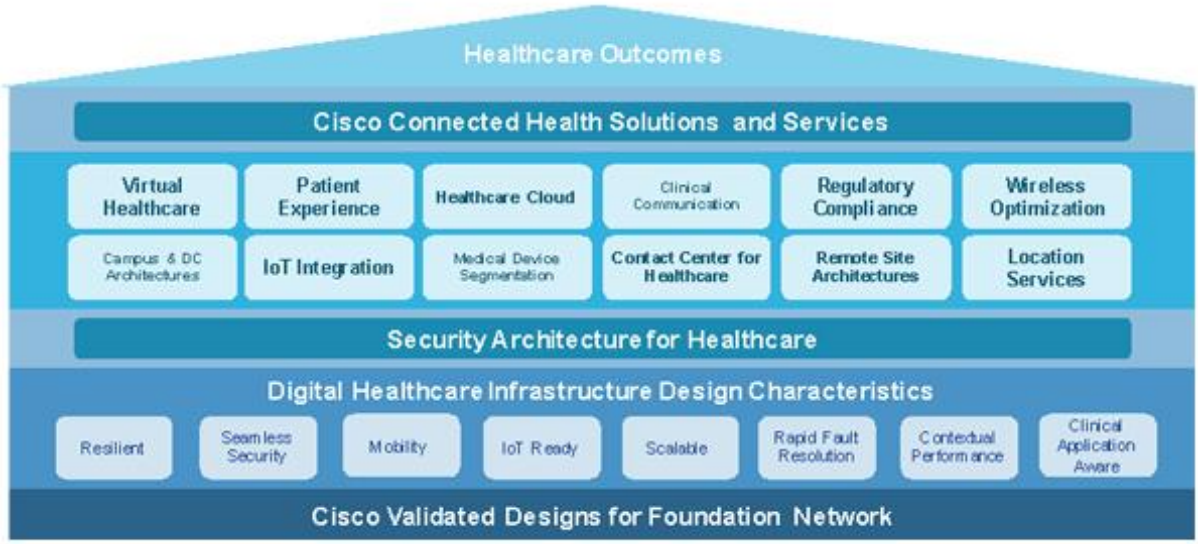

Figura 14- Cisco Connected Health Solutions and Services. Retirado de (© 2017 Cisco and/or its affiliates, 2016)

A Figura 14 apresenta todos os serviços e soluções presentes na solução *Cisco Connected Health Solutions and Services*.

A solução *Cisco Patient Connect* tem duas componentes, uma para internamento e outra para áreas comuns. A componente de internamento é constituída por uma aplicação que fornece conteúdo, informação e serviços personalizados (utilizando o monitor da televisão da sala do utente), os utentes podem aceder a serviços prestados pelo hospital como tradução de vídeo, entre outros. A componente desenvolvida para as áreas comuns pode utilizar o sinal digital disponível no hospital (ou centro de saúde) para fornecer informação para os utentes, funcionários ou fornecer serviços de entretenimento em salas de espera e áreas comuns (© 2017 Cisco and/or its affiliates, 2016).

A componente de internamento permite um atendimento personalizado ao utente, oferecendo-lhe (© 2017 Cisco and/or its affiliates, 2016):

- **Serviços de entretenimento:** fornecer serviços de qualidade como televisão, filmes, música, a fim de tornar a estadia do utente mais agradável;
- **Conteúdo educacional:** disponibilização de conteúdos educacionais relacionados com o problema atual do utente, com o objetivo de o capacitar para a sua doença melhorando os resultados esperados.;
- **Colaboração por vídeo:** diminuir o risco de infeção ao trocar as visitas presenciais à sala de internamento por contato multimédia. Facilitar a comunicação dos utentes com familiares;
- **Visão interativa do utente:** prestar mais cuidados personalizados, como fornecer conteúdo, mensagens e outras informações na sala do utente, sem necessidade de aumentar o *staff*;
- **Ferramentas clinicas e administrativas:** permitir aos profissionais de saúde aceder a informação dos utentes através do uso de dispositivos móveis. Possibilitando assim aceder ao estado do utente, ordenar tarefas, escrever relatórios, pedir análises, entre outros à distância.

A solução *Cisco Extended Care* é uma plataforma de colaboração que integra os componentes da infraestrutura de comunicação e outros dispositivos, simplifica fluxos de trabalho, e possibilita uma colaboração visual interativa.

Esta aplicação permite gerir consultas marcadas e não marcadas, ou seja, pode acompanhar as consultas programadas, mas também permite gerir visitar não urgentes para ajudar a diminuir a lotação nas emergências.

Na Tabela 6 apresentamos as características da solução *Cisco Extended Care*.

Tabela 6- Características da solução Cisco Extended Care. Retirado de (© 2017 Cisco and/or its affiliates, 2016).

Características	**Descrição**
Application server Software (Servidor de aplicações)	Fornece novos fluxos de trabalho de telessaúde e gere a conexão de colaboração *Cisco Extended Care* para utilizadores, recursos, sessões, entre outros.

Características	**Descrição**
Proxy	Permite que os utilizadores fora da rede da organização comuniquem e colaborem com os profissionais de saúde com segurança.

Virtual care rooms	Fornece uma "sala de internamento" virtual para aumentar o envolvimento remoto entre os utentes e os profissionais da área de saúde.
Integration Platform (Plataforma de integração)	Permite que soluções integradas sejam desenvolvidas. Suporta *API's REST* padrão para desenvolver soluções integradas.
Acesso em qualquer lugar com a *Cisco Unified Communication and Unified Mobility*	Permite a colaboração e um maior envolvimento dos profissionais de saúde com os utentes através do uso dos seus computadores, e dispositivos móveis.
Telehealth Collaboration Workflows (Fluxos de trabalho de colaboração telessaúde)	Acrescenta novos fluxos de trabalho de colaboração relacionados com a prestação de cuidados de saúde, como salas de espera virtuais e salas de cuidado virtuais.

No *site* oficial da *Cisco Systems, Inc.*, podemos observar que existem várias aplicações e os vários serviços que disponibilizam. Os serviços encontram-se divididos em 6 categorias, sendo elas:

1. Atendimento personalizado ao utente;
2. Assistência e colaboração remotas;
3. Simplificar os fluxos de trabalho clínicos;
4. Aumentar a eficiência no local de trabalho;
5. Conectar o departamento de Investigação e Desenvolvimento com a produção;
6. Habilitar segurança e conformidade.

Esta solução foi apresentada à comunidade científica por Nambiar et al.(2013).

6. BENCHMARKING

Neste capítulo vão ser comparadas as três arquiteturas selecionadas, mais concretamente, os componentes (da camada de "Processamento dos dados") das mesmas. As arquiteturas foram selecionadas com base no tipo da sua licença, neste caso, apenas as soluções *open source* são passiveis de ser comparadas. Este requisito foi definido para reduzir o valor dos investimentos em tecnologias *Big Data*.

Como se pode ver, a Tabela 7 apresenta todas as soluções selecionadas para a fase de *benchmarking*. A tabela apresenta as várias camadas e as ferramentas que constituem uma determinada camada.

Tabela 7 - Tabela comparativa entre as arquiteturas selecionadas.

Camada	**Arquitetura que inclui *Apache Spark***	**Arquitetura *Maharaja Yeshwatrao***	**Arquitetura Big Data para o projeto INTCare**
Armazenamento dos dados	*HDFS*	*HDFS*	*HDFS*
	Apache HBase	*Apache HBase*	*Apache HBase*
Processamento dos dados	*Apache Spark*	*Hadoop MapReduce*	*Apache Kafka*
	Spark Streaming		*Apache Storm*
			Apache Phoenix
Gestão	Sem Informação	*Apache Zookeeper*	*Apache Flume*
		Apache Chukwa	*Apache Oozie*
Acesso aos dados	*Apache Hive*	*Apache Hive*	*Apache Hive*
		Apache Pig	*Apache Sqoop*
		Apache Avro	
Camada analítica e *Business Intelligence*	*Spark Sql*	Sem Informação	Subsistema Gestão de Conhecimento do projeto *INTCare*
	MLib		
	GraphX		
	SparkR		
Segurança	Sem Informação	Sem Informação	*Apache Knox*
			Apache Ranger

Como se pode observar pela Tabela 7, na camada "Armazenamento dos dados", todas as soluções são constituídas pelas ferramentas *HDFS* e *Apache HBase*.

Na camada "Processamento dos dados" a Arquitetura que inclui *Apache Spark* é constituída pelo *Apache Spark* e *Spark Streaming*, a Arquitetura *Maharaja Yeshwatrao* é constituída pelo *Hadoop MapReduce,* e a Arquitetura Big Data para o projeto INTCare é constituída pelas ferramentas *Apache Kafka*, *Apache Storm* e *Apache Phoenix*.

Na camada "Gestão" a Arquitetura que inclui *Apache Spark* não apresenta qualquer ferramenta, a Arquitetura *Maharaja Yeshwatrao*é constituída pelo *Apache Zookeeper* e *Apache Chukwa*, a Arquitetura Big Data para o projeto INTCare possui as ferramentas *Apache Oozie* e *Apache Flume*.

A camada "Acesso aos dados" o *Apache Hive* está presente em todas as soluções, mas a Arquitetura *Maharaja Yeshwatrao*possui também o *Apache Pig* e *Apache Avro* e a Arquitetura Big Data para o projeto INTCare inclui também o *Apache Sqoop*.

A Arquitetura que inclui *Apache Spark* apresenta para a "Camada analítica e *Business Intelligence*" as ferramentas *Spark Sql*, *MLib*, *GraphX* e *SparkR* enquanto que a Arquitetura Big Data para o projeto INTCare conta com o subsistema gestão de conhecimento desenvolvido para o projeto INTCare e a Arquitetura *Maharaja Yeshwatrao*não apresenta qualquer ferramenta.

Por fim, na camada "Segurança" apenas a Arquitetura Big Data para o projeto INTCare apresenta ferramentas, que são o *Apache Knox* e *Apache Ranger*.

6.1 Comparação entre Hadoop MapReduce e Apache Spark

Neste subcapítulo vai ser apresentado, numa primeira fase, as diferenças entre as duas *fameworks* de processamento de dados que são o *MapReduce* e *Apache Spark*. Posteriormente vão ser apresentadas duas experiências que comparam o desempenho das duas *frameworks* em vários cenários.

Na Tabela 8 é possível observar as principais diferenças entre o *MapReduce* e *Apache Spark*.

Tabela 8 - Diferenças entre Hadoop MapReduce e Apache Spark. Retirado de (Verma, Mansuri, & Jain, 2016).

Hadoop MapReduce	***Apache Spark***
Armazena os dados no disco	Armazena os dados em memória. Os dados são primeiramente armazenados em memória e depois processados
Computação baseada na memória do disco, uso parcial da memória *RAM* (*Random Access Memory*)	Computação baseada na memória *RAM*, uso parcial da memória do disco
A tolerância a faltas é conseguida através de replicação	A tolerância a faltas é conseguida através de RDDs
Difícil de processar e analisar dados em tempo real	Pode ser utilizado para modificar dados em tempo real
Ineficiente para aplicações que necessitam de reutilizar constantemente o mesmo conjunto de dados	Guarda o conjunto de dados em memória para uma reutilização eficiente

Shi et al. (2015) realizaram uma experiência para comparar o desempenho entre as duas *frameworks*. A experiência, que vai ser apresentada em detalhe, consistiu na execução de várias cargas de trabalho (*workloads*) que simulassem vários cenários que se aproximam da utilização real destas *frameworks*.
Para a experiência utilizaram o seguinte *hardware*:

- 4 servidores;
- Cada nó possuía 32 CPU cores a 2,9 GHz, 9 discos de 7200 RPM (Rotações por Minuto) com 1 TB (*TeraByte)* de armazenamento e 190 GB de memória *RAM*, e possuía o *Red Hat Enterprise Linux* 64 *bits*;
- Os nós estavam conectados por um *switch* com uma velocidade de 1Gbps (*Gigabit per second* – *gigabit* por segundo).

Ambas as *frameworks* foram executadas sobre a versão 1.7.0 do *Java*. A versão 2.4.0 do *Hadoop* foi utilizada para executar o *MapReduce*. O *Apache Spark,* versão 1.3.0, foi implementado sobre a versão 2.4.0 do *HDFS*.
As cargas de trabalho que escolheram foram as seguintes:

- *Word Count;*
- *Sort;*
- *K-means.*

Utilizaram o exemplo do programa *WordCount* incluído tanto no *MapReduce* como no *Apache Spark* para a experiência com o *WordCount*. O *WordCount* é uma aplicação que conta o número de ocorrências de uma determinada palavra de dado um *input* (The Apache Software Foundation, 2017j). Para gerar o *input* foi utilizado o *Hadoop random text writer*.

Tabela 9 - Resultados da carga de trabalho WordCount. Adaptado de (Shi et al., 2015)

Plataforma	***Spark***	***MapReduce***	***Spark***	***MapReduce***	***Spark***	***MapReduce***
Tamanho do *input* (GB)	1	1	40	40	200	200
Tempo (Segundos)	30	64	70	180	232	630

Como se pode ver na Tabela 9, o *Apache Spark* obteve um melhor desempenho na execução do *WordCount*. Para um *input* de 1GB o *Apache Spark* foi 34s mais rápido, para 40GB foi 110s mais rápido e por fim, para 200GB a diferença foi maior, sendo que o *Apache Spark* foi 398s mais rápido a executar a tarefa.

Para a experiência com o *Sort* utilizaram o *TeraSort* no *MapReduce* e implementaram a função *sortByKey* no *Apache Spark*. O *TeraSort* mede quanto tempo o *MapReduce* demora a ordenar dados distribuídos de forma aleatória num determinado sistema. Para gerar o *input* foi utilizado o *gensort*, programa utilizado para gerar *inputs* para *benchmarks* (Nyberg, 2011).

Como se pode ver pela Tabela 10 para um *input* de 1GB, o *Apache Spark* executou a tarefa em menos tempo com uma diferença de 3s face ao *MapReduce*, mas o mesmo não se verifica para um *input* de 100 e 500 GB, onde o *MapReduce* executou a tarefa com uma diferença de 1.5m e 20m respetivamente.

Tabela 10 - Resultados da carga de trabalho Sort. Adaptado de (Shi et al., 2015)

Plataforma	***Spark***	***MapReduce***	***Spark***	***MapReduce***	***Spark***	***MapReduce***
Tamanho do *input* (GB)	1	1	100	100	500	500
Tempo	32s	35s	4.8m	3.3m	44m	24m

O *k-means* é um algoritmo de *clustering* que particiona *N* observações em *K clusters* onde cada observação pertence ao *cluster* com a média mais próxima. Este algoritmo é utilizado para *data mining* e descoberta de conhecimento. Os dados de treino foram gerados pela ferramenta *Hi-Bench*. Para executar a carga de trabalho em questão utilizaram o *Mahout k-means* para o *MapReduce* e o pacote que contém um exemplo do programa *k-means* para o *Apache Spark*. O *Apache Mahout* é um projeto *open source* da *Apache* utilizado para criar algoritmos de *machine learning* escaláveis.

Tabela 11 - Resultados da carga de trabalho k-means. Adaptado de (Shi et al., 2015)

Plataforma	***Spark***	***MapReduce***	***Spark***	***MapReduce***	***Spark***	***MapReduce***
Tamanho do *input* (GB)	1	1	100	100	500	500
Tempo da 1ª iteração	13s	20s	1.6m	2.3m	8.4m	9.4m
Tempo das iterações subsequentes	3s	20s	26s	2.3m	2.1m	10.6m

A Tabela 11 apresenta os resultados da execução da carga de trabalho *k-means*. Como se pode observar, tanto para a primeira como para as iterações subsequentes, o *Apache Spark* apresenta menores tempos de execução, de salientar o facto que a diferença em tempos se acentuar nas iterações seguintes devido ao mecanismo de *caching* mencionado no capítulo 4.1.3.

Gu & Li (2014) realizaram uma experiência para comparar a performance do *Hadoop MapReduce* e do *Apache Spark* em realizar tarefas iterativas. Para a realização da experiência utilizaram a seguinte configuração:

- 8 servidores (1 designado *Master* e os restantes *slaves*);
- Cada nó tinha 4 *CPU (Central Processing Unit) cores* e 3 GB (GigaByte) de memória;
- Cada nó possuía o Ubuntu 12.04.2 (GNU/Linux 3.5.0-28-generic x86 64) e estavam todos conectados por um *switch* (modelo *H3C S5100*) com uma velocidade de 100Mbps (*Megabits per second - megabit* por segundo);

A versão do *Hadoop* utilizada foi a 0.20.205.0 e para o *Spark* foi utilizada a versão 0.6.1. O *PageRank* foi o algoritmo escolhido para a experiência; é um algoritmo de grafos que avalia a qualidade de um nó (vértice) consoante o número de grafos ligados a ele e a qualidade dos mesmos, ou seja, um nó tem boa qualidade se possuir uma grande quantidade de nós de boa qualidade ligados a ele. Foram utilizados 10 *datasets*, dos quais 5 eram reais e 5 foram gerados pelo *Kronecker graph generator*. Os *datasets* vão ser apresentados em mais detalhe na Tabela 12.

Tabela 12 - Datasets utilizados na experiência realizada por Gu & Li (2014) . Retirado de Gu & Li (2014).

Nome	**Tamanho do Ficheiro**	**Nós**	**Arestas**	**Descrição**
Wiki-Vote	1.0 MB	7115	103.689	Rede de quem-vota-em-quem da *Wikipedia*
soc-Slashdot0902	10.8 MB	82.168	948.464	Dados da rede social *Slashdot* de fevereiro de 2009
web-Google	71.9 MB	875.713	5.105.039	Grafo da *web* da *Google*
cit-Patents	267.5 MB	3.774.768	16.518.948	Rede de citações de patentes dos Estados Unidos
Twitter	1.3 GB	11.316.811	85.331.845	Rede do *Twitter* de quem segue quem
kronecker19	40 MB	416.962	3.206.497	Grafo gerado pelo *Kronecker generator* com 19 iterações.

Nome	**Tamanho do Ficheiro**	**Nós**	**Arestas**	**Descrição**
Kronecker20	89 MB	833.566	7.054.294	Grafo gerado pelo *Kronecker generator* com 20 iterações.
Kronecker21	209 MB	1.665.554	15.519.448	Grafo gerado pelo *Kronecker generator* com 21 iterações.
Kronecker22	479 MB	3.330.326	34.142.787	Grafo gerado pelo *Kronecker generator* com 22 iterações.
Kronecker23	1.1 GB	6.654.956	75.114.133	Grafo gerado pelo *Kronecker generator* com 23 iterações.

Os tempos de execução variam consoante o tamanho do *dataset*. Para *datasets* pequenos, como por exemplo o *wiki-vote* e *soc-Slashdot0902*, o *Apache Spark* obteve melhores tempos que o *MapReduce*. Para estes casos, o *Apache Spark* foi entre 25 a 40 vezes mais rápido a executar as tarefas. Para os *datasets* com tamanho compreendido entre 40MB e 89 MB (*web-Google, knonecker19* e *kronecker20*) o *Apache Spark* continua a apresentar melhores tempos, mas a diferença de velocidades é menor que o caso anterior. Para esta situação o *Apache Spark* foi cerca de 10-15 vezes mais rápido que o *MapReduce*. Para os *datasets* cit-Patents e *kronecker22* o *Apache Spark* voltou a ser superior, mas a diferença voltou a ser menor que nos casos anteriores, neste caso foi entre 3 a 5 vezes mais rápido que o *MapReduce*. Quando o tamanho do *dataset* é superior a 1GB, o *MapReduce* superou o *Apache Spark*, sendo que, para o caso do *dataset Twitter*, o *Apache Spark* falhou durante a execução da tarefa enquanto que o *MapReduce* continuou até ao final. A Figura 15 apresenta dois gráficos com os tempos de execução das tarefas dos vários *datasets* em ambas as *frameworks*.

PageRank was the algorithm chosen for the experiment

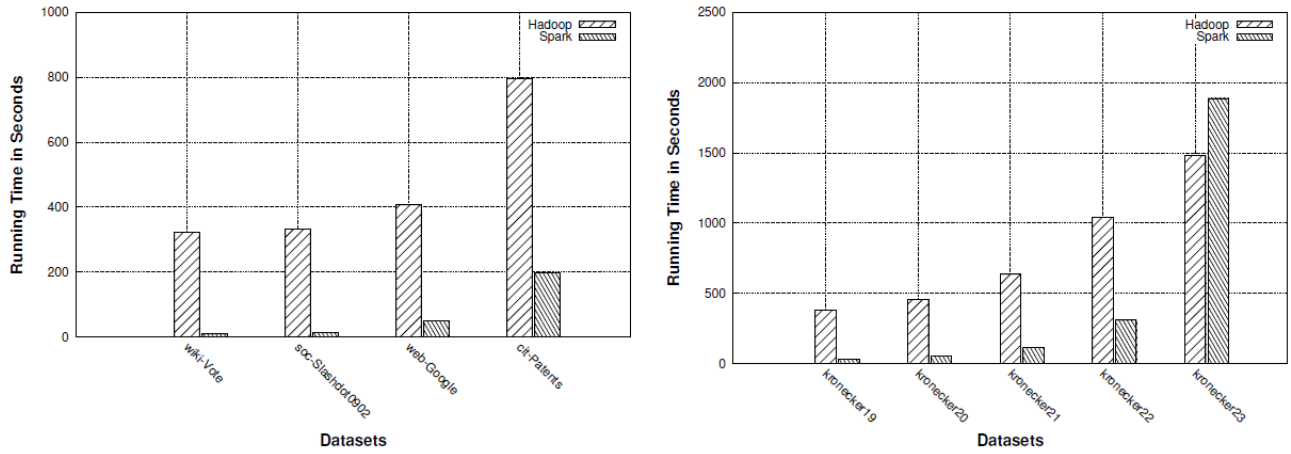

Figura 15 - Resultados da execução do algoritmo PageRank. Retirado de Gu & Li (2014)

Para ambos os gráficos, o eixo vertical está representado o tempo de execução da tarefa em segundos com valores compreendidos entre 0 e 1000, no eixo horizontal do gráfico à esquerda da estão representados os vários *datasets* pela seguinte ordem, da esquerda para a direita, *wiki-Vote*, *soc-Slashdot0902*, *web-Google* e *cit-Patients*; no eixo horizontal do gráfico à direita da Figura 15 estão representados os restantes *datasets* pela seguinte ordem, da esquerda para a direita, *kronecker19, kronecker20, kronecker21, kronecker22* e *kronecker23*.

6.2 Comparação entre Apache Spark e Apache Storm

Neste subcapítulo vai ser apresentada uma experiência que comparou o *Apache Spark* ao *Apache Storm* para perceber qual apresentava melhor desempenho em processamento de dados *streaming*.

A experiência realizada por Lu et al. (2014) consistiu na execução de 7 cargas de trabalho com o intuito de simular vários cenários que se aproximam da utilização real destas *frameworks*.

Para garantir uma maior aproximação de aplicações do mundo real, Lu et al. (2014) escolheram gerar o fluxos de dados tendo em conta os dois principais cenários de dados *streaming*, que são o processamento em tempo real de web log e a monitorização de tráfego de rede. Os *datasets* utilizados foram os seguintes:

- *AOL Search Data* – registo de pesquisas web de utilizadores reais. Os dados reais consistem em 20 milhões de pesquisas de 650.000 utilizadores, no entanto, para esta experiência os autores reduziram o *dataset*;

- *CAIDA Anonymized Internet Traces Dataset* – este *dataset* consiste em informações estatísticas de monitorização de tráfego *web*. Estes dados são coletados uma vez por mês e o tráfego corresponde a 1 hora de utilização da rede.

A Tabela 13 apresenta os *datasets* utlizados em mais detalhe.

Tabela 13 - Datasets utilizados para a comparação entre Apache Spark e Apache Storm. Adaptado de (Lu et al., 2014)

Dataset	**Tipo de dados**	**Número de registos**	**Média de tamanho de um registo**
AOL Search Data	Texto	100.000	60 *Bytes*
CAIDA Anonymized Internet Traces Dataset	Numérico	11.942	200 *Bytes*

As cargas de trabalho utilizadas foram (Lu et al., 2014):

- *Identity* – apenas lê o *input*, não faz qualquer tipo de operação sobre o mesmo*;*
- *Sample* – a amostra utilizada no *input* vai de encontro à probabilidade especificada*;*
- *Projection* - extrai um campo do *input*. É utlizado como um passo de várias *workloads*, tais como, *DistinctCount* e *Statistics;*
- *Grep* – verifica se o *input* contém uma determinada *string;*
- *Wordcount* – apresentado no subcapítulo 6.1*;*
- *DistinctCount* – numa primeira fase esta carga de trabalho extrai um campo de um determinado registo, posteriormente coloca-o num *set* com todas as palavras distintas e exibe o tamanho do *set*, que corresponde ao número de palavras distintas*;*
- *Statistics* – calcula o máximo, mínimo, somatório e média de um determinado campo do *input.*

Tabela 14 - Escalas de dados utilizados na comparação entre Apache Spark e Apache Storm. Adaptado de (Lu et al., 2014)

Número de registos	**Descrição utilizada**
5 milhões de registos	*Baseline*
10 milhões de registos	2X
20 milhões de registos	4X
50 milhões de registos	10X

Segundo Lu et al. (2014), para avaliar o desempenho de ferramentas de dados *streaming* é necessário que os dados sejam todos enviados para o sistema de mensagens para que, desta forma, as ferramentas possam obter os dados o mais rápido possível. As métricas são coletadas durante a execução das cargas

de trabalho. Para esta experiência foram utilizadas quatro escalas de dados como se pode observar pela Tabela 14.

As métricas utilizadas para avaliar o desempenho foram (Lu et al., 2014):

- *Throughput*: média da quantidade de registos processados por segundo;
- Latência: média dos intervalos desde a chegada de cada registo até ao final de processamento do mesmo.

Para esta experiência foram utilizados dois *clusters* com 12 nós no total, ou seja, 6 nós cada *cluster*. Um *cluster* serviu para executar o *Apache Spark* e o *Apache Storm* em que, 1 nó servia como *Master*, no caso do *Apache Spark*, ou *Nimbus*, no caso do *Apache Storm*, os restantes 5 nós serviam como s*laves* no caso do *Apache Spark*, ou *supervisors*, no caso do *Apache Storm*. No *cluster* restante, um nó servia para executar o *Apache Zookeeper* que fornecia serviços para o *Apache Kafka* e *Apache Storm* e os restantes 5 nós servia para executar o *Apache Kafka*.

Os nós possuíam diferentes configurações (ver Tabela 15), sendo que, o *cluster* que serviu para executar o *Apache Spark* e *Apache Storm* possuía 3 nós do tipo1 e 3 nós do tipo2. Os 6 nós *cluster* que serviram para executar o *Apache Kafka* e o *Apache Zookeeper* eram do tipo3. A velocidade máxima da ligação *ethernet*, que ligava todos os nós, era de 1000*Mb/s*.

Tabela 15 - Configuração dos nós utilizados na experiência. Adaptado de (Lu et al., 2014)

Nome do nó	**CPU**	**Memória RAM**
Tipo1	Xeon X5570@2.93GHz 16cores	48GB
Tipo2	Xeon E5-2680@2.70GHz 16cores	64GB
Tipo3	Xeon E5-2660@2.20GHz 32cores	192GB

Para a comparação foram utilizadas a versão *0.9.0-incubating* do *Apache Spark* e a versão *0.9.1-incubating* do *Apache Storm*.

No que diz respeito à execução da experiência, para o *Apache Spark*, todas as escalas de dados (Tabela 14) são apresentadas, enquanto que, para o *Apache Storm*, a escala de dados 10X foi omitida devido a elevados tempos de execução.

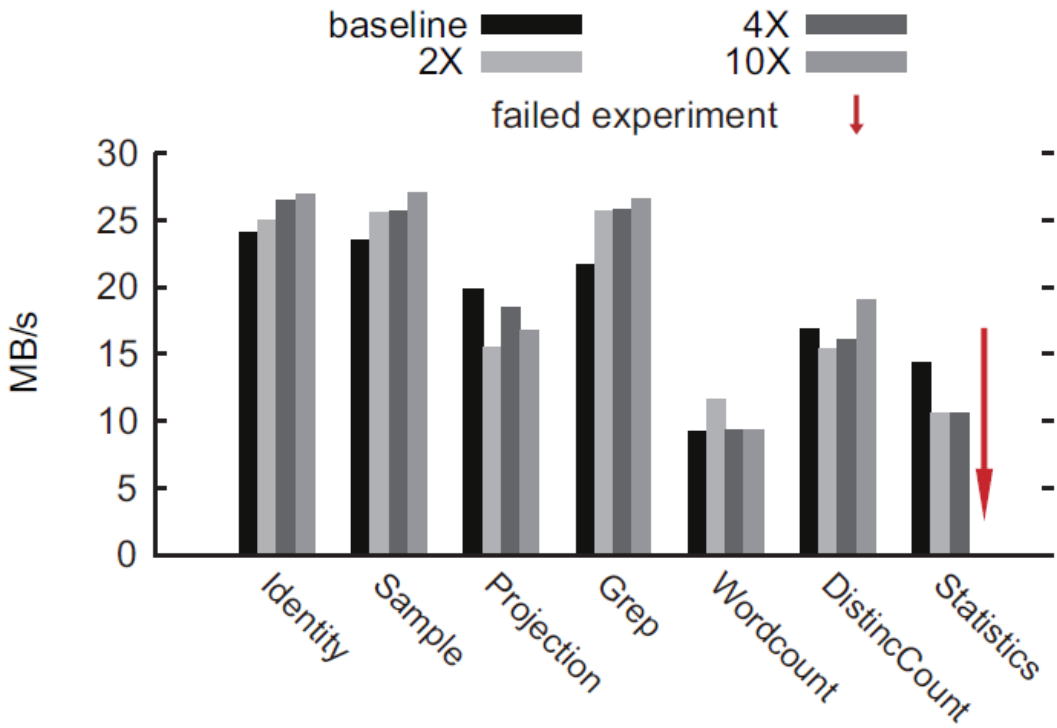

Figura 16 - Resultados da métrica throughput do Apache Spark Streaming. Retirado (Lu et al., 2014)

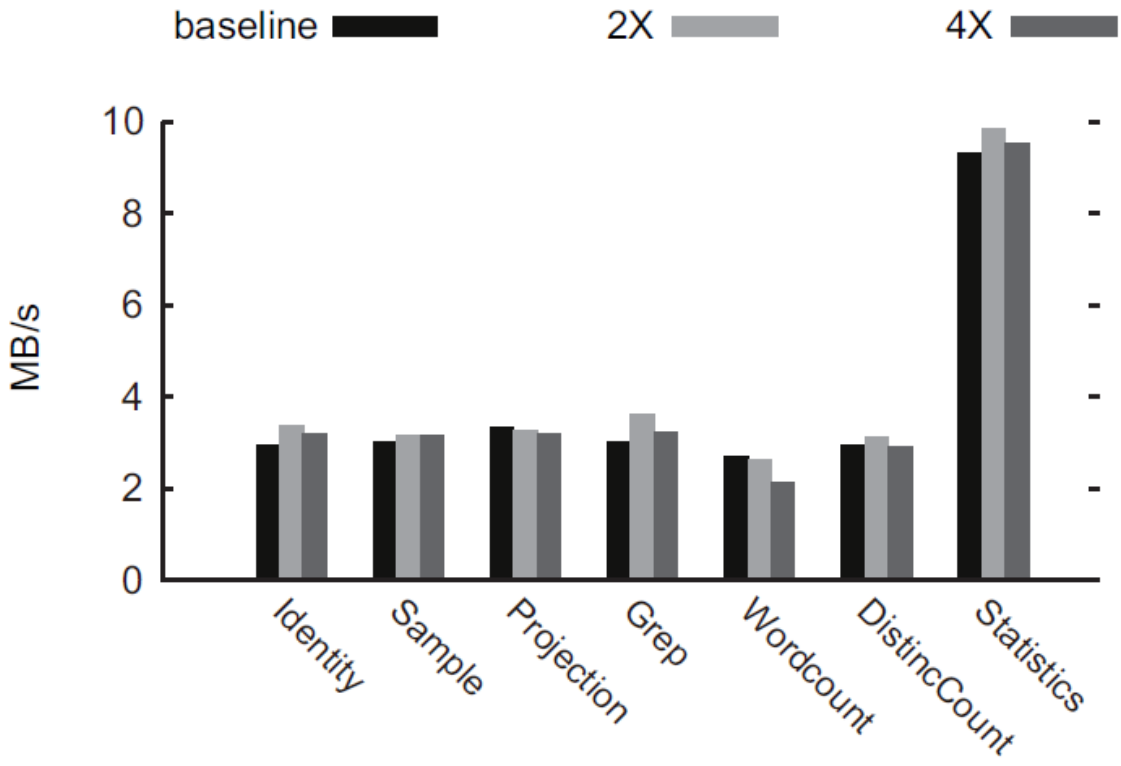

Figura 17 - Resultados da métrica throughput do Apache Storm. Retirado (Lu et al., 2014)

As figuras, Figura 16 e Figura 17, no eixo horizontal apresentam as várias cargas de trabalho mencionadas neste subcapítulo, no eixo vertical apresenta o valor de *throughput* em MB/s (*MegaBytes per second – MegaBytes* por segundo). As várias cores que estão representadas são referentes às várias escalas dos dados (Tabela 14), a seta vermelha com sentido descendente representa uma falha na experiência.

Como se pode observar, os valores de *throughput* do *Apache Spark Streaming* são cerca de 4.5 vezes maiores que os valores obtidos pelo *Apache Storm*. A diferença de valores é notória para cargas de trabalho menos complexas. Para a carga de trabalho *Satistics*, os valores são bastante próximos (Lu et al., 2014).

Lu et al. (2014) notaram também que os valores de *throughput* do *Apache Spark Streaming* variam bastante entre as várias cargas de trabalho enquanto que para o *Apache Storm* os valores são constantes, excetuando os valores da carga de trabalho *Statistics*. Pela análise das figuras (Figura 16 e Figura 17) os autores concluíram que o tamanho dos registos influencia os valores de *throughput* especialmente no *Apache Storm*.

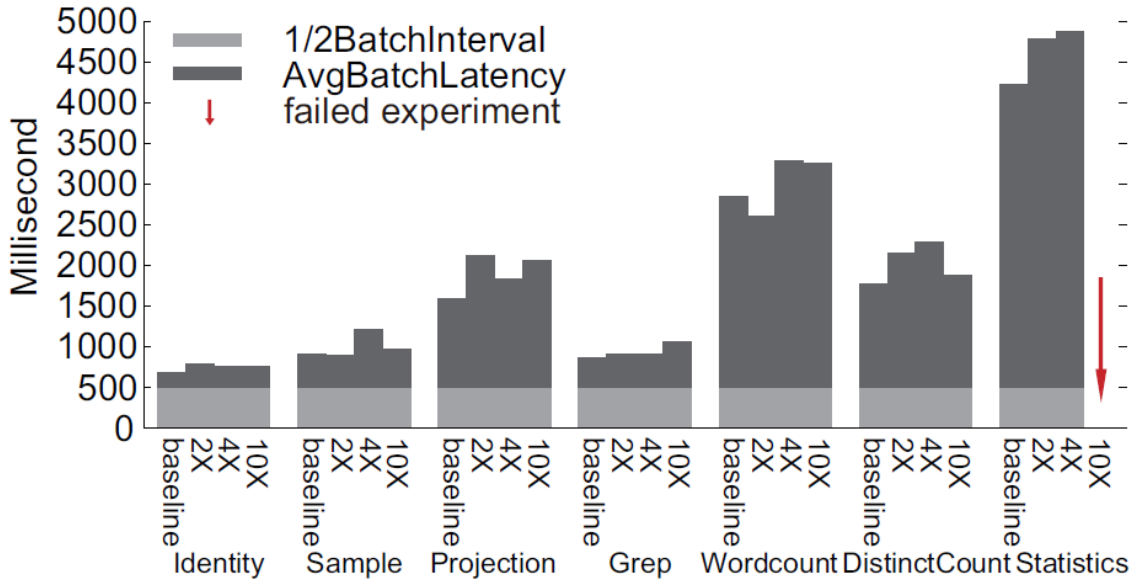

Figura 18 - Resultados da métrica latência do Apache Spark Streaming. Retirado (Lu et al., 2014))

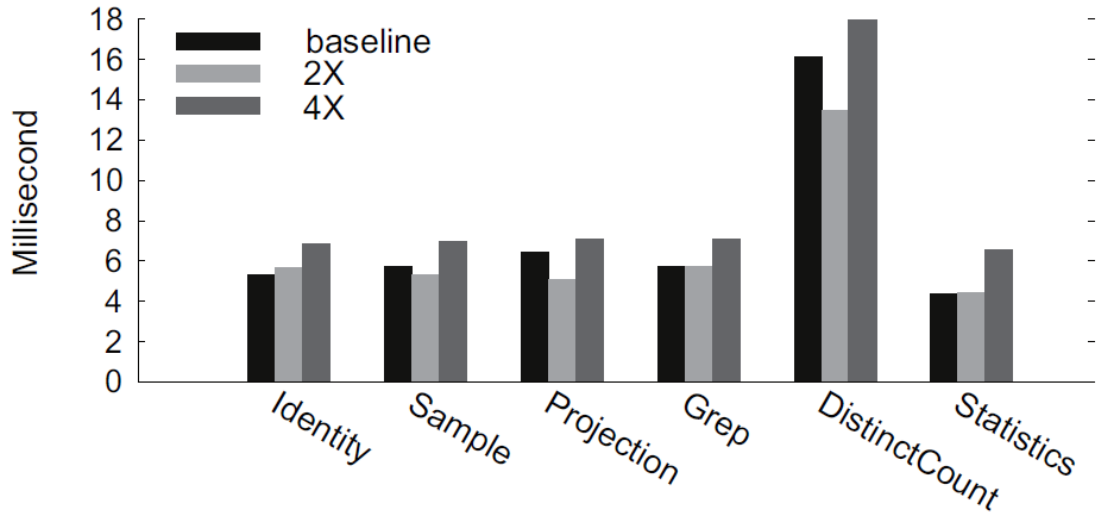

Figura 19 - Resultados da métrica latência do Apache Storm. Retirado (Lu et al., 2014))

A Figura 18 apresenta no eixo horizontal as várias cargas de trabalho e as diferentes escalas de dados (Tabela 14); no eixo vertical apresenta os valores de latência em milissegundos (ms). A cor representada pela legenda "*1/2BatchInterval*" representa o valor definido pelos autores da experiência como intervalo de tempo para conclusão do processamento de um determinado *batch*. A cor representada pela legenda "*AvgBatchLatency*" representa o valor médio do valor de latência de processamento de um *batch*. A seta vermelha com sentido descente representa uma falha na experiência.

A Figura 19, no eixo horizontal apresenta as várias cargas de trabalho mencionadas neste subcapítulo, no eixo vertical apresenta o valor de latência em milissegundos. As várias cores que estão representadas são referentes às várias escalas dos dados (Tabela 14).

Como se pode observar pelas figuras (Figura 18 e Figura 19) os valores de latência do *Apache Storm* são bastante inferiores aos valores de latencia do *Apache Spark Streaming*, enquanto que para o *Apache Storm,* a escala de valores de latência é inferior a 20 milisegundos, para o *Apache Spark Streaming* os valores estão à escala de 1000 milissegundos. No entanto, para a carga de trabalho *WordCount*, os valores de latência do *Apache Storm* são 4 vezes superiores aos valores obtidos pelo *Apache Spark* como se pode observar na Tabela 16 (Lu et al., 2014).

Tabela 16 - Valores de latência do Apache Storm na execução do WordCount. Adaptado de *(Lu et al., 2014)*

Carga de trabalho	**Baseline (milissegundos)**	**2X (milissegundos)**	**4X (milissegundos)**
WordCount	166.827	1900.146	12019.256

Lu et al. (2014) salientaram que os valores de latência do *Apache Storm* aumentam drasticamente dependendo da complexidade da carga de trabalho, o mesmo não se verifica para o *Apache Spark*.

Após a análise das experiências que comparam o *Hadoop MapReduce* e o *Apache Spark*, é possível concluir que o *Hadoop MapReduce* está mais preparado para lidar com *datasets* com tamanhos superiores a 1GB (tendo em conta a configuração dos *clusters* utilizados nas experiências). Também se pode concluir que, o *Apache Spark* é mais rápido no processamento dos dados, excetuando casos em que o tamanho do *dataset* obriga a ativação do mecanismo de substituição de RDD's o que resulta numa perda de desempenho.

Após uma análise à experiência que compara o *Apache Spark Streaming* ao *Apache Storm* é possível concluir que o *Apache Spark Streaming* possui melhores valores no que diz respeito à métrica *throughput* face ao *Apache Storm* mas o mesmo não se verifica nos valores de latência, onde o *Apache Storm* apresenta melhores valores, excetuando os valores obtidos na execução da carga de trabalho *WordCount*. No que diz respeito à capacidade de lidar com os dados, o *Apache Storm* apresenta piores resultados em comparação com o *Apache Spark Streaming*, sendo que Lu et al. (2014) escolheram omitir os valores do *Apache Storm* para a escala 10X (Tabela 14) pois eram bastante elevados.

7. Análise e Discussão de Resultados

Neste capítulo será feita uma análise ao estado da adoção de ferramentas *Big Data* em hospitais/clinicas de saúde aprovadas pela comunidade científica e também uma análise detalhada às experiências que comparam os componentes (da camada de "Processamento dos dados") das soluções escolhidas para o *benchmarking*.

Com base no estudo efetuado é possível concluir que não existe muita documentação científica acerca da implementação de tecnologias *Big Data* em hospitais/clínicas de saúde. Conclui-se que a aprovação da comunidade científica pode ajudar a ultrapassar alguns dos desafios que se apresentam à adoção de tecnologias *Big Data* na área da saúde.

As experiências analisadas no subcapítulo 6.1 demonstram que o *Apache Spark* apresenta um melhor desempenho face ao *Hadoop MapReduce* na maioria dos testes efetuados.

A experiência efetuada por Shi et al. (2015) demonstrou um melhor desempenho do *Apache Spark* face ao *MapReduce* para as cargas de trabalho *WordCount* e *k-means*. O mesmo não se verificou para a da carga de trabalho *Sort*, na qual o *MapReduce* teve melhores resultados. No entanto, Shi et al. (2015) apresentam como causa para o melhor desempenho do *MapReduce* a configuração do *cluster*, mais propriamente a velocidade da rede que conecta os vários nós. Shi et al. (2015) referenciam o concurso *Daytona Gray Sort* de 2014 onde o *Apache Spark* obteve um melhor desempenho face ao *MapReduce* e onde a velocidade da rede era de 10Gbps face aos 1Gbps da experiência mencionada.

Na experiência conduzida por Gu & Li (2014) o *Apache Spark* apresenta um melhor desempenho face ao *MapReduce* para *datasets* com tamanho inferior a 1GB, mas o mesmo não se verifica para *datasets* com tamanho superior a 1GB. Os autores da experiência atribuem esta perda de desempenho à falta de memória *RAM* do *cluster*, e completam dizendo que para tamanhos superiores a 1GB não existe memória suficiente para os *RDDs* recentemente criados e por consequência o *Apache Spark* ativa um mecanismo de substituição de *RDD*, o que leva a uma perda de desempenho gradual ou até mesmo falha como se pôde verificar para o caso do *dataset Twitter*. Concluímos então que a configuração do *cluster* tem um papel fundamental para o desempenho do *Apache Spark*.

No que diz respeito à experiência apresentada no subcapítulo 6.2 é possível concluir que o *Apache Spark Streaming* é mais robusto no sentido em que apenas não conseguiu processar a carga de trabalho *Statistic* para a escala de dados 10X (caso que se verifica em ambas as métricas), enquanto que os valores do *Apache Storm* para a escala 10X de todas as cargas de trabalho foram omitidos devido aos

elevados tempos de processamento Lu et al. (2014). No que diz respeito ao *throughput*, o *Apache Spark Streaming* apresenta valores superiores para todas as cargas de trabalho, ou seja, processa mais dados por segundo em comparação com o *Apache Storm*. No que diz respeito aos valores de latência, o *Apache Storm* apresenta um melhor desempenho face *Apache Spark Streaming* para todos os cenários excetuando a carga de trabalho *WordCount* onde os valores foram aproximadamente 4 vezes superiores aos obtidos pelo *Apache Spark Streaming*.

Perante os resultados obtidos nas experiências analisadas, é possível concluir que:

- A Arquitetura Big Data para o projeto INTCare é a mais adequada para lidar com dados *streaming*. Esta solução conjuga o *Apache Kafka* e o *Apache Storm* para lidar com dados provenientes de monitores de cabeceira (sinais vitais, ventilação e outros) (Gonçalves, 2017);
- A Arquitetura *Maharaja Yeshwatrao* é a mais indicada para lidar com grandes volumes de dados, apesar do *Hadoop MapReduce* apresentar um pior desempenho face ao *Apache Spark* em grande parte dos testes apresentados no subcapítulo 6.1, o mesmo revelou-se mais capaz de lidar com grandes volumes de dados;
- A Arquitetura que inclui *Apache Spark* é uma solução híbrida pois revelou-se capaz de lidar com dados *streaming* e *batch*. No entanto, como foi referido anteriormente, o desempenho do *Apache Spark* está muito dependente da configuração do *cluster*.

Apesar de não ter sido possível fazer uma comparação direta de desempenho entre todas as soluções escolhidas para o *benchmarking*, conclui-se que a aplicação mais adequada para uma organização na área da saúde é a Arquitetura Big Data para o projeto INTCare. Esta solução apresenta em detalhe todos os componentes e como os mesmos vão interagir com o sistema onde se inserem e, mais importante, é a única solução que apresenta componentes na camada "Segurança" como se pode observar na Tabela 7. Dado que a segurança é um dos desafios à implementação de *Big Data* na saúde, considera-se necessário a integração de ferramentas que assegurem a segurança dos dados e do sistema no geral.

8. Conclusão

Neste capítulo serão abordados os aspetos finais desta dissertação será descrito o trabalho realizado, as principais conclusões retiradas, algumas limitações que surgiram durante o desenvolvimento deste projeto e aspetos a serem abordados futuramente.

8.1 Síntese do trabalho efetuado

O desenvolvimento desta dissertação dividiu-se em três fases. A primeira coincide com a revisão de literatura e consistiu na pesquisa de informação relacionada com o tema *Big Data*, *Big Data* na saúde e identificar o potencial e os desafios à implementação de tecnologias *Big Data* em hospitais/clínicas de saúde. A segunda componente coincide com a pesquisa de aplicações utilizadas ou projetadas para ser implementadas em hospitais/clínicas de saúde aprovadas pela comunidade científica. Na terceira componente definiram-se os critérios para filtrar as aplicações a ser comparadas e foi realizada uma pesquisa de informação relativa a comparações efetuadas a aplicações similares às arquiteturas selecionadas para o *benchmarking*.

Durante a fase de revisão de literatura foi possível perceber qual o estado da implementação de *Big Data* na área da saúde, o potencial e os principais desafios. Nesta fase foi também analisada uma publicação de Portela, Lima, & Santos (2016) que fala sobre a importância de avaliar projetos para saber se existe ou não a necessidade de utilizar tecnologias *Big Data*, a avaliação de projetos ajuda as organizações a tomar melhores decisões no que diz respeito a investimentos em tecnologias e pode evitar insucessos.

Durante a segunda fase foi feita uma pesquisa a várias aplicações *Big Data*, posteriormente foi feita uma filtragem para perceber se existia documentação científica que comprovasse a utilização ou possível implementação das aplicações analisadas em hospitais/clínicas de saúde.

Na terceira fase foi feita a comparação das tecnologias com base no tipo da sua licença, para o nosso caso, apenas as tecnologias *open source* foram comparadas. Este critério foi definido com o intuito de facilitar a adoção de tecnologias *Big Data* por parte das instituições de saúde.

8.2 Contributos

Terminado o projeto, é possível afirmar que foram cumpridos todos os objetivos propostos inicialmente, sendo que um dos objetivos foi reajustado no decorrer desta dissertação. Inicialmente foi proposto realizar testes às aplicações selecionadas para o *benchmarking*, que posteriormente teve que ser

reajustado devido a restrições de tempo. Apesar do reajustamento, os resultados obtidos consideram-se positivos.

Os resultados obtidos nesta dissertação revelaram-se positivos, dado que foi possível cumprir os objetivos definidos:

- A revisão de literatura permitiu perceber qual o estado da adoção de tecnologias *Big Data* na área saúde. Foram identificados os desafios e também o potencial da implementação de tecnologias *Big Data* em hospitais/clínicas de saúde*;*
- A pesquisa de aplicações utilizadas ou projetadas para ser implementadas em hospitais/clínicas de saúde revelou-se a tarefa mais desafiante de toda a dissertação devido à escassez de literatura relativa à implementação de soluções *Big Data* na saúde;
- A pesquisa de experiências efetuadas a aplicações semelhantes às escolhidas para comparação permitiu avaliar o desempenho das aplicações em vários cenários. Desta forma, foi possível perceber os pontos fortes e fracos das soluções escolhidas.

A análise de aplicações revelou a variedade de soluções a ser exploradas, ou seja, não existe uma solução ideal que consiga satisfazer todas as necessidades.

Por fim, é possível concluir que a implementação de tecnologias *Big Data* na área da saúde pode introduzir melhorias significativas, ajudando a reduzir custos, melhorar tratamentos, diminuir taxas de readmissão, melhorar a monitorização dos utentes, melhorar a saúde pública, entre outros; com isto respondeu-se à questão de investigação definida inicialmente. Torna-se evidente que o *Big Data* na saúde é uma aposta para o futuro.

8.3 Análise de Riscos

A Tabela 17 apresenta alguns dos riscos identificados inicialmente e as ações de mitigação para reduzir o impacto dos mesmos.

Tabela 17- Lista de riscos.

Nº	Descrição	Grau	Ação de mitigação	Impacto	Verificado
1	Ausência de informação necessária relativa ao tema do projeto de dissertação	3	Prolongamento do tempo estabelecido no plano de trabalhos para a pesquisa de aplicações *Big Data* utilizadas na área da saúde.	Médio	Sim

Nº	Descrição	Grau	Ação de mitigação	Impacto	Verificado
2	Atrasos no decorrer do projeto	5	Foram efetuadas alterações ao plano de trabalhos e foi feito um reajustamento dos objetivos.	Muito alto	Sim
3	Dificuldades em coordenar as unidades curriculares em falta com as tarefas relativas à dissertação.	5	Reajustamento dos objetivos, de forma a garantir resultados relevantes para o estudo.	Muito alto	Sim

8.4 Trabalho Futuro

Após a conclusão deste projeto, existem algumas vertentes a serem exploradas no futuro, destacando-se entre elas:

- a exploração de ferramentas similares às apresentadas que não tenham sido ainda apresentadas à comunidade científica;
- realizar testes práticos às ferramentas apresentadas com dados reais;
- a realização de um artigo cientifico relativo ao estudo efetuado nesta dissertação.

REFERÊNCIAS BIBLIOGRÁFICAS

© 2017 Cisco and/or its affiliates. (2016). *Cisco Patient Connect*.

Alexandre, J., & Cavique, L. (2013). NoSQL no Suporte à Análise de Grande Volume de Dados. *Revista de Ciências Da Computação, 2013, nº8*, *d*, 37–48.

APQC. (2017). Benchmarking Methodology. Retrieved from https://www.apqc.org/benchmarking-methodology

Brynjolfsson, E., & McAfee, A. (2012). Big Data: The Management Revolution. https://doi.org/2024194500

Coutinho, C. P., & Chaves, J. H. (2002). O estudo de caso na investigação em Tecnologia Educativa em Portugal. *Revista Portuguesa de Educação*, *15*(1), 221–243. https://doi.org/10.1371/journal.pcbi.1000106

Davenport, T. (2011). *Big data at Work*. Retrieved from https://www.systemonline.cz/clanky/big-data.htm

Dias, J. A., & Duarte, P. (2015). BIG DATA OPPORTUNITIES IN HEALTHCARE. HOW CAN MEDICAL AFFAIRS CONTRIBUTE? AS OPORTUNIDADES DOS BIG DATA NOS CUIDADOS DE SAÚDE. QUE CONTRIBUTO PODEM DAR OS ASSUNTOS MÉDICOS?, 230–236.

Dijcks, J. (2012). Oracle: Big data for the enterprise. *Oracle White Paper*, (June), 16. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Oracle+:+Big+Data+for+the+Enterprise#0

Feldman, B., Martin, E. M., & Skotnes, T. (2012). Big Data in Healthcare - Hype and Hope. *Dr.Bonnie 360 Degree (Business Development for Digital Health)*, *2013*(1), 122–125. Retrieved from http://www.riss.kr/link?id=A99883549

Gonçalves, A. J. P. (2017). *Arquitetura de Big Data em tempo-real para unidades de cuidados intensivos*. Universidade do Minho.

Groves, P., Kayyali, B., Knott, D., & Van Kuiken, S. (2013). The “big data” revolution in healthcare: accelerating value and innovation. *McKinsey Global Institute*, (January), 1–22. Retrieved from http://www.images-et-reseaux.com/sites/default/files/medias/blog/2013/12/mckinsey_131204_-_the_big_data_revolution_in_healthcare.pdf

Gu, L., & Li, H. (2014). Memory or time: Performance evaluation for iterative operation on hadoop and

spark. *Proceedings - 2013 IEEE International Conference on High Performance Computing and Communications, HPCC 2013 and 2013 IEEE International Conference on Embedded and Ubiquitous Computing, EUC 2013*, 721–727. https://doi.org/10.1109/HPCC.and.EUC.2013.106

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, *47*, 98–115. https://doi.org/10.1016/j.is.2014.07.006

Hortonworks Inc. (2017). ABOUT HORTONWORKS. Retrieved from https://hortonworks.com/about-us/

Hurwitz, J., Nugent, A., Halper, D. F., & Kaufman, M. (2013). *Big Data For Dummies*.

IBM. (2012). IBM PureData System for Analytics N3001.

IBM. (2013). IBM PureData Solution for Healthcare Analytics.

IBM. (2014). Snippet blades (S-Blades). Retrieved from https://www.ibm.com/support/knowledgecenter/en/SSULQD_7.2.0/com.ibm.nz.gsg.doc/c_getstrt_s_blades.html

IBM. (2016). IBM Fluid Query 1 . 7, 1–8.

José, C., & Ribeiro, S. (2014). Big Data : os novos desafios para o profissional da informação, *1*(1), 96–105.

Kala Karun, A., & Chitharanjan, K. (2013). A review on hadoop - HDFS infrastructure extensions. *2013 IEEE Conference on Information and Communication Technologies, ICT 2013*, (Ict), 132–137. https://doi.org/10.1109/CICT.2013.6558077

Krishnan, S. M. (2016). Application of analytics to big data in healthcare. *Proceedings - 32nd Southern Biomedical Engineering Conference, SBEC 2016*, 156–157. https://doi.org/10.1109/SBEC.2016.88

Kulkami, A. P., & Khandewal, M. (2014). Survey on Hadoop and Introduction to YARN. *International Journal of Emerging Technology and Advanced Engineering*, *4*(5), 82–87. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.639.2875&rank=1

LaValle, S. (2009). Business Analytics and Optimization for the Intelligent Enterprise. *Business*, 1–20. https://doi.org/10.1108/17410400510571446

Lavastorm Analytics. (2014). Why Most Big Data Projects Fail - Learning from Common Mistakes to Transform Big Data into Insights, 1–7.

Lima, C., & Calazans, J. (2013). Pegadas Digitais: "Big Data" e informação estratégica sobre o consumidor.

Lima, L. C. B. de. (2014). *Big data for data analysis in financial industry*. Universidade do Minho. Retrieved

from http://repositorium.sdum.uminho.pt/handle/1822/34919

Liu, W., Li, Q., Cai, Y., Li, Y., & Li, X. (2015). A Prototype of Healthcare Big Data Processing System Based on Spark, (Bmei), 516–520.

Liu, W., & Park, E. K. (2014). Big data as an e-health service. *2014 International Conference on Computing, Networking and Communications, ICNC 2014*, 982–988. https://doi.org/10.1109/ICCNC.2014.6785471

Lu, R., Wu, G., Xie, B., & Hu, J. (2014). Stream bench: Towards benchmarking modern distributed stream computing frameworks. *Proceedings - 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing, UCC 2014*, 69–78. https://doi.org/10.1109/UCC.2014.15

Maçada, A. C. G., Brinkhues, R. A., & Júnior, J. C. F. (2015). Big data e as capacidades de gestão da informação, (July), 6–9.

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Hung Byers, A. (2011). Big data: The next frontier for innovation, competition, and productivity. *McKinsey Global Institute*, (June), 156. https://doi.org/10.1080/01443610903114527

Nambiar, R., Sethi, A., Bhardwaj, R., & Vargheeseh, R. (2013). A Look at Challenges and Opportunities of Big Data Analytics in Healthcare, 17–22.

NIST Big Data Public Working Group. (2015). *NIST Special Publication 1500-4 DRAFT NIST Big Data Interoperability Framework : Volume 4 , Security and Privacy DRAFT NIST Big Data Interoperability Framework : Volume 4 , Security and Privacy* (Vol. 4). https://doi.org/10.6028/NIST.SP.1500-5

Nyberg, C. (2011). gensort Data Generator. Retrieved from http://www.ordinal.com/gensort.html

Ojha, M., & Mathur, K. (2016). Proposed application of big data analytics in healthcare at Maharaja Yeshwantrao Hospital. *2016 3rd MEC International Conference on Big Data and Smart City, ICBDSC 2016*, 40–46. https://doi.org/10.1109/ICBDSC.2016.7460340

Portela, C. F., Santos, M. F., Silva, Á., Machado, J., & Abelha, A. (2011). Enabling a pervasive approach for intelligent decision support in critical health care. *Communications in Computer and Information Science*, *221 CCIS*(PART 3), 233–243. https://doi.org/10.1007/978-3-642-24352-3_25

Portela, F., Lima, L., & Santos, M. F. (2016). Why Big Data? Towards a Project Assessment Framework. *Procedia Computer Science*, *58*(WoTBD), 604–609. https://doi.org/10.1016/j.procs.2016.09.094

Portela, F., Santos, M. F., Machado, J., Abelha, A., Silva, Á., & Rua, F. (2014). *Pervasive and Intelligent Decision Support in Intensive Medicine – The Complete Picture*.

Raghupathi, W., & Raghupathi, V. (2014). Big data analytics in healthcare: promise and potential. *Health*

Information Science and Systems, *2*, 3. https://doi.org/10.1186/2047-2501-2-3

Sagiroglu, S., & Sinanc, D. (2013). Big data: A review. *2013 International Conference on Collaboration Technologies and Systems (CTS)*, 42–47. https://doi.org/10.1109/CTS.2013.6567202

Salas-Vega, S., Haimann, A., & Mossialos, E. (2015). Big data and healthcare: Challenges and opportunities for coordinated policy development in the EU. *Health Systems & Reform*, *8604*(September), 00–00. https://doi.org/10.1080/23288604.2015.1091538

Shi, J., Qiu, Y., Farooq Minhas, U., Jiao, L., Wang, C., Reinwald, B., & Ozcan, F. (2015). Clash of the Titans: MapReduce vs. Spark for Large Scale Data Analytics, (3), 2110–2121. Retrieved from http://delivery.acm.org/10.1145/2840000/2831365/p2110-shi.pdf?ip=200.133.1.60&id=2831365&acc=ACTIVE SERVICE&key=344E943C9DC262BB.7073CFED1EAC8731.4D4702B0C3E38B35.4D4702B0C3E38B35&CFID=975951557&CFTOKEN=63119937&__acm__=1503529947_e42901080d97089c0937e7f4

Silva, L. A., & Breternitz, V. J. (2013). Big Data : Um Novo Conceito Gerando Oportunidades E. *Revista RETC*, 106–113. https://doi.org/http://revista-fatecjd.com.br/retc/index.php/RETC/article/view/74/pdf

Singh, B., & Grover, S. (2013). An Overview of Benchmarking Process : The Continuous Improvement Tool, *1*(July), 80–83.

Stapenhurst, T. (2009). *The Benchmarking Book: A How-to-Guide to Best Practice for Managers and Practitioners*. *The Benchmarking Book*. https://doi.org/10.1016/B978-0-7506-8905-2.00010-5

Tellis, W. M. (1997). The Qualitative Report Application of a Case Study Methodology Application of a Case Study Methodology. *The Qualitative Report*, *3*(33), 1–19. https://doi.org/3.3

The Apache Software Foundation. (2013). HDFS Architecture Guide. Retrieved from https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

The Apache Software Foundation. (2015a). Apache Kafka. Retrieved from https://kafka.apache.org/intro

The Apache Software Foundation. (2015b). Apache Storm. Retrieved from http://storm.apache.org/

The Apache Software Foundation. (2015c). Hive design and architecture. Retrieved from https://cwiki.apache.org/confluence/display/Hive/Design

The Apache Software Foundation. (2016). Apache Hadoop YARN. Retrieved from https://hadoop.apache.org/docs/r2.7.2/hadoop-yarn/hadoop-yarn-site/YARN.html

The Apache Software Foundation. (2016). Cluster Mode Overview. Retrieved from http://spark.apache.org/docs/latest/cluster-overview.html

The Apache Software Foundation. (2016). Spark SQL, DataFrames and Datasets Guide. Retrieved from https://spark.apache.org/docs/latest/sql-programming-guide.html

The Apache Software Foundation. (2016). Spark Streaming Programming Guide. Retrieved from https://spark.apache.org/docs/latest/streaming-programming-guide.html

The Apache Software Foundation. (2016). Welcome to Apache Chukwa! Retrieved from http://chukwa.apache.org/

The Apache Software Foundation. (2016). ZooKeeper: A Distributed Coordination Service for Distributed Applications. Retrieved from https://zookeeper.apache.org/doc/trunk/zookeeperOver.html

The Apache Software Foundation. (2017a). Apache Avro™ 1.8.2 Documentation. Retrieved from https://avro.apache.org/docs/current/

The Apache Software Foundation. (2017b). Apache Oozie Workflow Scheduler for Hadoop. Retrieved from http://oozie.apache.org/

The Apache Software Foundation. (2017c). Apache Ranger. Retrieved from https://ranger.apache.org/

The Apache Software Foundation. (2017d). Apache Spark. Retrieved from https://spark.apache.org/

The Apache Software Foundation. (2017e). Apache Sqoop. Retrieved from http://sqoop.apache.org/

The Apache Software Foundation. (2017f). Architecture Overview. Retrieved from http://hbase.apache.org/book.html#arch.overview

The Apache Software Foundation. (2017g). Data Model. Retrieved from http://hbase.apache.org/book.html#datamodel

The Apache Software Foundation. (2017h). GraphX Programming Guide. Retrieved from https://spark.apache.org/docs/latest/graphx-programming-guide.html

The Apache Software Foundation. (2017i). Lightning-fast cluster computing. Retrieved from http://spark.apache.org/

The Apache Software Foundation. (2017j). MapReduce Tutorial. Retrieved from https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

The Apache Software Foundation. (2017k). MLlib. Retrieved from https://spark.apache.org/mllib/

The Apache Software Foundation. (2017l). REST API and Application Gateway for the Apache Hadoop Ecosystem. Retrieved from https://knox.apache.org/

The Apache Software Foundation. (2017m). Welcome to Apache Flume.

The Apache Software Foundation. (2017n). Welcome to Apache Pig! Retrieved from https://pig.apache.org/

Thusoo, A., Sarma, J. Sen, Jain, N., Shao, Z., Chakka, P., Anthony, S., ... Murthy, R. (2009). Hive - A Warehousing Solution Over a Map-Reduce Framework. *Sort*, *2*, 1626–1629. https://doi.org/10.1109/ICDE.2010.5447738

Vavilapalli, V. K., Murthy, A., Douglas, C., Agarwal, S., Konar, M., Evans, R., ... Reed, B. (2013). Apache Hadoop YARN : Yet Another Resource Negotiator.

Verma, A., Mansuri, A. H., & Jain, N. (2016). Big data management processing with Hadoop MapReduce and spark technology: A comparison. *2016 Symposium on Colossal Data Analysis and Networking, CDAN 2016*. https://doi.org/10.1109/CDAN.2016.7570891

Vora, M. N. (2011). Hadoop-HBase for large-scale data. *Proceedings of 2011 International Conference on Computer Science and Network Technology, ICCSNT 2011*, *1*, 601–605. https://doi.org/10.1109/ICCSNT.2011.6182030

White, M. (2012). Digital workplaces. *Business Information Review*, *29*(4), 205–214. https://doi.org/10.1177/0266382112470412

White, T. (2012). Hadoop: The definitive guide 4th Edition. *Online*, *54*, 258. https://doi.org/citeulike-article-id:4882841

Xia, F., Yang, L. T., Wang, L., & Vinel, A. (2012). Internet of Things, 1101–1102. https://doi.org/10.1002/dac

Yaqoob, I., Hashem, I. A. T., Gani, A., Mokhtar, S., Ahmed, E., Anuar, N. B., & Vasilakos, A. V. (2016). Big data: From beginning to future. *International Journal of Information Management*, *36*(6), 1231–1247. https://doi.org/10.1016/j.ijinfomgt.2016.07.009

Yu, W. D., Gill, J. S., Dalal, M., Jha, P., & Shah, S. (2015). BIG DATA APPROACH IN HEALTHCARE USED FOR INTELLIGENT DESIGN - Software As A Service, (Bmei), 516–520.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark : Cluster Computing with Working Sets. *HotCloud'10 Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 10. https://doi.org/10.1007/s00256-009-0861-0

Zikopoulos, P., Eaton, C., DeRoos, D., Deutsch, T., & Lapis, G. (2012). *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*.