Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Otimização no Custo para Processamento de *Big GeoSpatial Data* em Ambiente de Nuvem Computacional

João Bachiega Junior

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Orientadora
Prof.ª Dr.ª Aletéia Patrícia Favacho de Araújo

Coorientadora
Prof.ª Dr.ª Maristela Terto de Holanda

Brasília
2018

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenador: Prof. Dr. Bruno Luiggi Macchiavello Espinoza

Banca examinadora composta por:

Prof.ª Dr.ª Aletéia Patrícia Favacho de Araújo (Orientadora) — CIC/UnB
Prof.ª Dr.ª Maristela Terto de Holanda — CIC/UnB
Prof.ª Dr.ª Edna Dias Canedo — CIC/UnB
Prof. Dr. Angelo Roncalli Alencar Brayner — UFC

CIP — Catalogação Internacional na Publicação

Junior, João Bachiega.

Otimização no Custo para Processamento de *Big GeoSpatial Data* em Ambiente de Nuvem Computacional / João Bachiega Junior. Brasília : UnB, 2018.
65 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2018.

1. Big Geospatial Data, 2. Computação em Nuvem, 3. SpatialHadoop, 4. Dados Geográficos.

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Otimização no Custo para Processamento de *Big GeoSpatial Data* em Ambiente de Nuvem Computacional

João Bachiega Junior

Dissertação apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof.ª Dr.ª Aletéia Patrícia Favacho de Araújo (Orientadora)
CIC/UnB

Prof.ª Dr.ª Maristela Terto de Holanda
CIC/UnB

Prof.ª Dr.ª Edna Dias Canedo
CIC/UnB

Prof. Dr. Angelo Roncalli Alencar Brayner
UFC

Prof. Dr. Bruno Luiggi Macchiavello Espinoza
Coordenador do Mestrado em Informática

Brasília, 21 de Junho de 2018

Dedicatória

Ao meu filho, Gustavo, e à minha esposa, Cristiane, por terem suportado minha ausência para que este trabalho pudesse ser realizado.

Agradecimentos

À Deus, por ter me permitido chegar até aqui. As nossas silenciosas conversas foram fundamentais nos momentos de dificuldades.
Aos meus pais, João e Alzira, que através da simplicidade me transmitiram os maiores ensinamentos da vida.
À minha esposa, Cristiane, pelo incansável apoio.
Ao meu filho, Gustavo. Sua ingenuidade e amor não me permitiram desistir.
Às minhas orientadoras, Aletéia e Maristela, pela confiança.
Ao colega, Leornardo Moreira e aos demais colegas da Pós-Graduação, pela convivência e apoio.

Resumo

Os dados geográficos representam abstrações de entidades do Mundo real e podem ser obtidos de diversas formas. Além disso, eles possuem algumas propriedades que os diferenciam dos demais tipos de dados, tais como a estrutura complexa, a dinamicidade e o volume. Nos últimos anos, com o crescimento do volume dos dados geográficos, conceituado como *big geospatial data*, algumas ferramentas foram desenvolvidas para possibilitar o processamento eficiente desses dados, entre elas o *SpatialHadoop*, que é um *framework* incorporado ao *Hadoop*. A utilização da indexação correta dos dados baseado no conjunto de dados a ser processado, e também nas consultas e nas operações a serem realizadas é fundamental para que estas aplicações tenham o melhor desempenho. Por outro lado, como a tarifação em provedores públicos de computação em nuvem ocorre de acordo com o uso, é importante otimizar a execução das aplicações para evitar desperdício financeiro. Assim, este trabalho propõe a construção de uma Base de Conhecimento e de um Mecanismo de Inferência que buscam a otimização dos custos para o processamento de *big geospatial data* em provedores públicos de nuvem. Além disso, é apresentada uma comparação entre os serviços oferecidos pelos três principais provedores de nuvem pública para o processamento de grande volume de dados. Os testes executados demonstraram que a utilização das regras geradas pelo Mecanismo de Inferência e a escolha do provedor de menor custo são capazes de otimizar os custos totais de processamento em até 71%.

Palavras-chave: Big Geospatial Data, Computação em Nuvem, SpatialHadoop, Dados Geográficos.

Abstract

Spatial Data represents abstractions of real-world entities and can be obtained in various ways. They have properties that differentiate them from other types of data, such as a complex structure and dynamism. In recent years with the increasing volume of spatial data, referred to as big geospatial data, some tools have been developed to process this data efficiently, such as SpatialHadoop, a framework incorporated into Hadoop. The use of appropriate data indices based on the dataset to be processed, queries and operations to be performed is essential for the optimal performance of these applications. In particular, since public cloud providers charge based on the resources used, it is imperative to optimize application execution to avoid unnecessary expense. This paper proposes the construction of a Knowledge Base and an Inference Engine that seek to minimize the cost of processing big geospatial data in public cloud providers. In addition, a comparison of the services offered by three public cloud providers for large-volume data processing is presented. The tests performed demonstrate that the use of rules generated by the Inference Engine and the choice of the lowest-cost provider can reduce the total processing cost by up to 71%.

Keywords: Big Geospatial Data, Cloud Computing, SpatialHadoop, Spatial Data

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Os dados geográficos representam abstrações de entidades do Mundo real, tais como estradas, edifícios, lagos, florestas e países, por meio da localização geográfica e das características naturais ou artificiais, fenômenos e limites da Terra [9]. Este tipo de dado pode ser obtido de diversas formas, incluindo *raster*, vetor, ponto, linha, polígono, texto, vídeo, registros de banco de dados, etc [9]. Além disso, eles possuem algumas propriedades que os diferenciam dos demais tipos de dados, tais como a estrutura complexa, a dinamicidade e o volume [5]. Atrelado a isso, houve um crescimento assintótico de dados geográficos gerados e disponibilizados nos últimos anos, provenientes de diferentes fontes, como por exemplo *smartphones*, telescópios espaciais e satélites [10].

Dessa forma, surgiu o conceito de *Big GeoSpatial Data* [6], que incentivou a necessidade do desenvolvimento de aplicações capazes de processar de maneira otimizada uma grande quantidade de dados, e também a disponibilização de poder computacional capaz de suprir as necessidades geradas por estas aplicações.

Assim, existem diversas ferramentas disponíveis para o processamento de grandes volumes de dados, sendo o *Apache Hadoop* [11] um dos mais difundidos. Ele utiliza um modelo de programação conhecido como *MapReduce*, que divide um grande problema em pedaços menores e os executa de forma paralela e distribuída. Para conseguir ser eficiente também para o processamento de grandes volumes de dados geográficos, os conceitos de *Hadoop* foram adaptados para diversas ferramentas. Entre essas ferramentas, destaca-se o *SpatialHadoop* [8], que é um *framework* que implementa funcionalidades espaciais no núcleo do *Hadoop*, tornando-se eficiente no processamento de consultas e de operações espaciais.

Além disso, para que essas aplicações tenham um bom desempenho, uma tarefa importante é a indexação do conjunto de dados a ser processado, que é uma forma de organizar um conjunto de dados de maneira hierárquica e ordenada [12]. No entanto, existem diferenças entre os métodos de indexação, fazendo com que a escolha do índice mais adequado ao conjunto de dados, às consultas e às operações a serem executadas, também tenha papel fundamental na eficiência do processamento realizado.

Com relação à infraestrutura para suportar estas aplicações, a computação em nuvem tem se mostrado bastante aderente à esse cenário, pois é um modelo que possibilita o acesso sob demanda a um conjunto de recursos computacionais, com alto poder de processamento e de armazenamento [13]. Quando comparada à computação tradicional, ou seja, construída de forma dedicada para uma organização, a computação em nuvem

oferece diversas vantagens relacionadas à confiabilidade do serviço e também ao custo. Isto se deve ao fato do usuário evitar o investimento inicial na aquisição de equipamentos para o processamento da aplicação, e também não ter a preocupação do equipamento adquirido ser suficiente para manter o desempenho da aplicação em períodos de pico, uma vez que os recursos computacionais podem ser adquiridos e liberados rapidamente, de acordo com a necessidade exigida pela aplicação. Além disso, os recursos utilizados são tarifados de acordo com o uso, em forma de serviço, assim como já ocorre com serviços básicos como água, energia e telefone [14].

No entanto, existem diferenças entre os diversos provedores públicos de nuvem com relação à tarifação e aos serviços oferecidos para o processamento de grandes volumes de dados, gerando pesquisas como as apresentadas por Buyya *et al.* [15], que visam entender o custo-benefício de mover aplicativos com uso intensivo de recursos de ambientes locais para nuvem pública. Para garantir a escolha da oferta de menor custo, foi realizada uma análise dos serviços oferecidos para este fim pelos três principais provedores públicos de computação em nuvem – Amazon AWS, Google Cloud e Microsoft Azure.

Assim sendo, com a combinação dos três fatores – utilização do serviço para processamento de *Big Geospatial Data* em provedor de nuvem pública adequado, indexação do conjunto de dados que otimize as consultas e as operações a serem realizadas e, quando possível, o redimensionamento dos recursos computacionais – este trabalho propõe uma Base de Conhecimento e um Mecanismo de Inferência para otimizar o custo financeiro para o processamento de *big geospatial data* em provedores públicos de nuvem.

1.1 Motivação

O modelo de computação em nuvem vem sendo cada vez mais utilizado para a execução de aplicações e serviços que demandam alto poder de processamento e de armazenamento. Mais recentemente os provedores públicos de nuvem passaram a ofertar serviços específicos para o processamento de grandes volumes de dados, com suporte às diversas ferramentas voltadas para este tipo de aplicação, entre elas o *Hadoop*, facilitando, assim, a configuração da infraestrutura, o redimensionamento do *cluster* e o gerenciamento das aplicações.

Por outro lado, as aplicações para processamento de *big geospatial data* possuem algumas características específicas que devem ser consideradas para a alocação adequada da infraestrutura que as suportará, garantindo, desta forma, bom desempenho. Entre estas características está a indexação do conjunto de dados, pois se trata de uma tarefa onerosa computacionalmente e com impacto direto no desempenho das consultas e das operações a serem realizadas pelas aplicações.

Os trabalhos avaliados, mostraram que o ambiente de nuvem é bastante indicado para o processamento de *big geospatial data*, no entanto, não houve, até o momento, um estudo que focasse na otimização do custo neste ambiente, especificamente, para este tipo de aplicação.

Assim, diante da lacuna notada na literatura, este trabalho tem como motivação a necessidade de otimizar os custos financeiros para o processamento de *big geospatial data* alocados em provedores públicos de nuvem computacional.

1.2 Problema

A utilização de provedores públicos de computação em nuvem é tarifada de acordo com a utilização dos recursos computacionais – pelo segundo do processamento ou pelos *bytes* transferidos, por exemplo. Desta forma, a otimização no uso dos recursos computacionais alocados é fundamental para a redução dos custos.

As aplicações que realizam consultas e operações com dados geográficos, por sua vez, possuem características específicas, como por exemplo a necessidade de indexação do conjunto de dados e que, se não consideradas, podem impactar no desempenho da aplicação e, desta forma, gerar desperdício financeiro.

Assim, nota-se que um dos problemas nessa área é não tratar de maneira diferenciada as aplicações de acordo com as indexações a serem aplicadas às bases de dados, bem como às consultas e às operações a serem executadas. Com isso, os recursos computacionais são utilizados de forma não otimizada, resultando em gastos que poderiam ser evitados se estes fatores fossem considerados na alocação dos recursos.

1.3 Objetivos

O objetivo geral deste trabalho é otimizar o custo financeiro para o processamento de *big geospatial data* em provedores públicos de computação em nuvem. Para tanto, faz-se necessário cumprir os seguintes objetivos específicos:

1. Analisar o desempenho de diversas indexações de acordo com o conjunto de dados a ser processado, bem como das consultas e das operações a serem realizadas;
2. Avaliar o desempenho das operações e das consultas de acordo com a capacidade dos recursos computacionais;
3. Construir uma Base de Conhecimento e um Mecanismo de Inferência capazes de indicar a indexação mais adequada e a possibilidade do redimensionamento da infraestrutura;
4. Validar que a aplicação das regras do Mecanismo de Inferência garantam à otimização no custo financeiro para o processamento de *big geospatial data* em ambiente de nuvem computacional pública.

1.4 Metodologia de Pesquisa

Para atingir os objetivos propostos nesta pesquisa, foram definidas três fases a saber:

- **Fase 1:** Revisão bibliográfica, a fim de identificar os trabalhos já produzidos sobre o tema de pesquisa, e assim contribuir para o estado da arte;
- **Fase 2:** Construção da Base de Conhecimento e do Mecanismo de Inferência, executando diversas operações que sejam capazes de criar uma base consistente para a formalização da proposta, de maneira qualitativa e quantitativa. Para isso, foram considerados apenas provedores públicos de computação em nuvem, não sendo foco desta pesquisa outros modelos de implantação;

- **Fase 3:** Realização de testes, através de uma abordagem quantitativa, para validação da proposta de otimização de custo para processamento de *big geospatial data*, considerando o escopo dos cenários propostos.

A execução de cada uma das fases de maneira consistente são suficientes para garantir o atingimento do objetivo deste trabalho que é a otimização do custo financeiro para o processamento de *big geospatial data* em provedores públicos de computação em nuvem.

1.5 Estrutura do Trabalho

Este trabalho está dividido, além deste capítulo, em mais sete capítulos. O Capítulo 2 apresentará os conceitos sobre dados geográficos e *big geospatial data*. Os conceitos sobre computação em nuvem, os serviços e as ferramentas disponíveis para o processamento de *big geospatial data* serão apresentados no Capítulo 3. O Capítulo 4 apresenta os trabalhos relacionados ao tema desta dissertação. Os detalhes para a construção da Base de Conhecimento e do Mecanismo de Inferência propostos são apresentados no Capítulo 5. O Capítulo 6 traz a avaliação dos resultados obtidos com a utilização das regras do Mecanismo de Inferência e, por fim, o Capítulo 7 faz as considerações finais sobre esta dissertação e destaca alguns trabalhos futuros para a continuidade desta pesquisa.

Capítulo 2

Dados Geográficos

Neste capítulo são apresentados os principais conceitos relacionados aos dados geográficos, as principais indexações e as operações realizadas com estes dados. Além desses conceitos, este capítulo apresenta a definição do termo *Big Geospatial Data*.

2.1 Definição

Dado geográfico é aquele que se distingue essencialmente pelo componente espacial, que associa a cada entidade ou fenômeno uma localização na Terra, traduzida por sistema geodésico de referência, em dado instante ou período de tempo, podendo ser derivado, entre outras fontes, das tecnologias de levantamento, inclusive as associadas a sistemas globais de posicionamento apoiados por satélites, bem como de mapeamento ou de sensoriamento remoto [16].

As estruturas de dados utilizadas em bancos de dados geográficos podem ser divididas em duas grandes classes [1]: estruturas vetoriais e estruturas matriciais. As estruturas vetoriais são utilizadas para representarem as coordenadas das fronteiras de cada entidade geográfica, por meio de três formas básicas, as quais são pontos, linhas e áreas (ou polígonos), definidas por suas coordenadas cartesianas, conforme apresentado na Figura 2.1.

Por outro lado, as estruturas matriciais usam uma grade regular sobre a qual se representa, célula a célula, o elemento que está sendo especificado. A cada célula atribui-se um código referente ao atributo estudado, de tal forma que o computador saiba a que

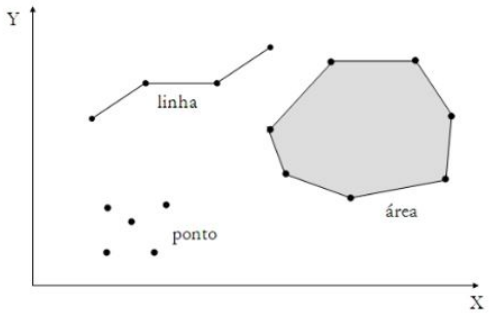

Figura 2.1: Formas Básicas de Representação Geográfica, reproduzida de [1].

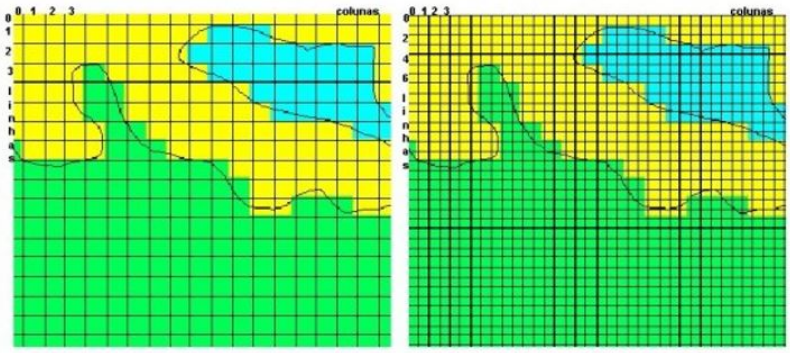

Figura 2.2: Representação Matricial, reproduzido de [1].

elemento ou objeto pertence determinada célula. Nesta representação o espaço é visto como uma matriz *P(m, n)*, composta de *m* colunas e *n* linhas, sendo que cada célula possui um número de linha, um número de coluna e um valor correspondente ao atributo estudado, e cada célula é individualmente acessada pelas suas coordenadas.

A representação matricial supõe que o espaço pode ser tratado como uma superfície plana, na qual cada célula está associada a uma porção do terreno. A resolução espacial do sistema é dada pela relação entre o tamanho da célula no mapa (ou documento), e a área por ela coberta no terreno [1]. A Figura 2.2 ilustra a representação matricial em duas diferentes resoluções espaciais. Note que as células da imagem à esquerda são maiores do que as da imagem à direita, o que significa que a segunda tem uma melhor resolução espacial.

2.2 *Big Geospatial Data*

O termo *big data* é usado, principalmente, para descrever enormes conjuntos de dados, em sua maioria não estruturados. Além disso, o *big data* também traz oportunidades ao descobrir novos valores, auxiliando na compreensão dos dados e gerando novos desafios, como por exemplo, para organizar e gerenciar esses conjuntos de dados com eficiência [17].

Diante disso, o termo *big data* é considerado um conceito abstrato [17], pois além do grande volume de dados, ele também engloba outros atributos, tais como a variedade e a velocidade, diferenciando-se da simples referência de "dados muito grandes". Assim, embora o conceito de *big data* seja amplamente utilizado, ainda há grande discussão sobre sua definição.

Em 2010, a Comunidade Apache Hadoop [18], definiu o termo *big data* como os conjuntos de dados que não podem ser capturados, gerenciados e processados por computadores comuns dentro de um escopo aceitável.

O *International Data Corporation* (IDC) publicou, em 2011, um relatório que definiu *big data* como uma nova geração de tecnologias e arquiteturas projetadas para extrair valor economicamente de grandes volumes de uma ampla variedade de dados, permitindo a captura, a descoberta e a análise de alta velocidade [19].

Para o *National Institute of Standards and Technology* (NIST) [20], *big data* significa os dados cujo volume de dados, velocidade de aquisição ou representação de dados limita a capacidade de usar banco de dados relacionais tradicionais para conduzir análises efetivas,

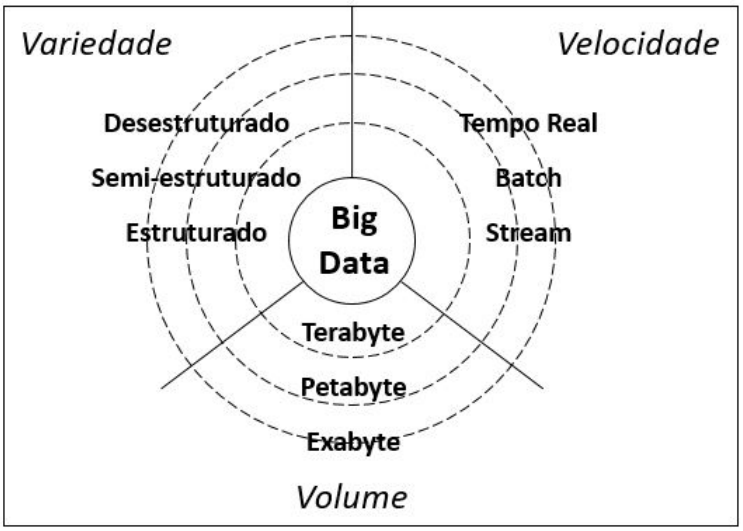

Figura 2.3: Características Essenciais de *Big Data*, reproduzida de [2].

ou os dados que devem ser efetivamente processados por tecnologias que exijam maior capacidade computacional, capazes de realizar o processamento em um menor tempo.

Por fim, de acordo com o Instituto Gartner [21], o termo *big data* refere-se ao "grande volume, alta velocidade e/ou grande variedade de informações que exigem formas inovadoras de processamento que permitem uma melhor compreensão, tomada de decisão e automação de processos". Esta definição, baseada em 3V's (Volume, Velocidade e Variedade), conforme apresentado na Figura 2.3 é considerada a mais tradicional para *big data* e foi utilizada, ao longo do tempo, por diversos outros autores [2, 22, 23].

A Variedade indica que, diferentemente dos sistemas tradicionais, nos quais os dados estão todos estruturados, para *big data*, os dados podem ser recebidos de forma semi-estruturada ou não estruturada. A característica da Velocidade refere-se às diversas formas de processamento e, por fim, a característica de Volume refere-se aos grandes volumes de dados que devem ser processados.

No entanto, com a evolução das tecnologias para a obtenção e para o processamento de dados, novos "Vs" foram atribuídos à definição de *big data*. Kaur *et al.* [24], adicionaram a característica da Variabilidade, referindo-se às constantes mudanças nas informações coletadas. Ward e Barker [25], e também Lovalekar [26], indicam a inclusão do termo Veracidade, uma vez que os dados devem ser confiáveis.

Da mesma maneira que ocorreu para os demais tipos de dados, nos últimos anos, houve um alto crescimento nos montantes de dados espaciais produzidos por vários dispositivos, como *smartphones*, dispositivos móveis, telescópios espaciais, dispositivos médicos, entre outros [10]. Estes dados são coletados a uma velocidade cada vez mais rápida e com maiores resoluções, excedendo a capacidade dos sistemas computacionais atuais, com um crescimento estimado de 20% a cada ano [27]. Por exemplo, os telescópios espaciais geram até 150 GB de espaço semanal de dados, dispositivos médicos produzem imagens espaciais a uma taxa de 50 PB por ano, um arquivo da NASA de satélite com imagens da Terra tem mais de 500 TB e é aumentada diariamente por 25 GB, enquanto existem 10 milhões de *tweets* etiquetados geograficamente emitidos do Twitter todos os dias [10, 27].

Além disso, dispositivos de GPS e *smartphones* geram uma enorme quantidade de

dados de localização e, frequentemente, estes dados de localização pontuais precisam ser associados a dados de infraestrutura urbana, como redes rodoviárias e zonas administrativas, para melhor entender os padrões de mobilidade humana e, posteriormente, melhorar o planejamento urbano, o controle de tráfego e a manutenção da infraestrutura. Como exemplo, mais de 13.000 táxis na cidade de Nova York têm gerado meio milhão de viagens de táxi por dia, cada uma com locais de coleta e devolução de passageriso, além de registros de data e hora. Isso equivale a quase 170 milhões de viagens de táxi anualmente, transportando mais de 300 milhões de passageiros [28].

Muitos dos conjuntos de dados científicos também têm um componente geoespacial e precisam se alinhar às zonas ecológicas e administrativas globais e regionais para facilitar a investigação científica e a descoberta. Por exemplo, o *Global Biodiversity Information Facility* possui mais de 400 milhões de registros de ocorrência de espécies e muitos deles estão associados a um par (latitude e longitude). É essencial mapear os registros de ocorrência das várias regiões ecológicas para entender os padrões de biodiversidade e fazer planos de conservação [28].

Desta forma, o termo *Big Geospatial Data* é um paradigma emergente para a enorme quantidade de informações geográficas e espaciais que se tornam disponíveis a cada segundo em todo o Mundo [29].

2.3 Métodos de Acesso Multidimensionais

O processamento de operações espaciais é fortemente influenciado e melhorado pelo uso de estruturas de dados e algoritmos de pesquisa conhecidos como Métodos de Acesso Multidimensionais (MAM) [12]. Estes métodos são projetados para atuarem como um caminho otimizado aos dados espaciais com base em um conjunto definido de predicados sobre os atributos. Neste sentido, o espaço indexado é organizado de tal forma que, por exemplo, a recuperação dos objetos espaciais contidos em uma área particular requeira apenas o acesso a objetos próximos a esta área, em oposição a análise do conjunto completo de objetos armazenados em disco.

De fato, normalmente, uma consulta espacial envolve apenas uma pequena parcela do banco de dados. Neste caso, percorrer todo o banco pode ser bastante ineficiente. Portanto, um plano de execução para uma consulta tipicamente começa com uma fase de filtragem, que identifica quais objetos podem satisfazer a qualificação da consulta. Esta fase depende da existência de índices espaciais, em geral definidos sobre aproximações das geometrias dos objetos [4].

O desempenho de um MAM varia significativamente em função, por exemplo, das características dos dados espaciais e da escolha do tipo de consulta espacial. Em outras palavras, a relação de desempenho entre um conjunto de MAM pode não ser a mesma quando são considerados diferentes conjuntos de dados, diferentes tipos de consultas e operações espaciais [12].

Um MAM é voltado, principalmente, à indexação direta de pontos e de áreas, sendo que seu principal objetivo é propiciar uma rápida obtenção dos objetos espaciais que satisfazem a um determinado critério. Desta forma, a utilização da indexação minimiza o conjunto de objetos espaciais pesquisados, sendo que este subconjunto, na maioria dos casos, é menor do que o total de objetos espaciais armazenados em um arquivo de dados [4].

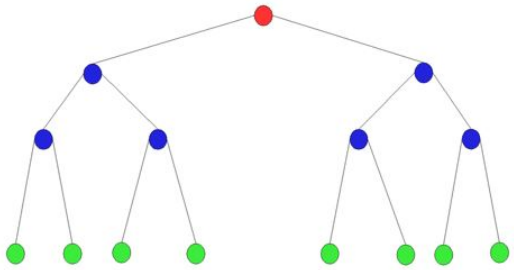

Figura 2.4: Árvore Binária Balanceada.

2.3.1 Principais Índices Espaciais

Ao longo do tempo, diversas pesquisas vêm sendo realizadas no intuito de melhorar as formas de indexação dos dados espaciais, que é uma forma de organizar um conjunto de dados de maneira hierárquica e ordenada, por meio do uso de árvores de pesquisa, como por exemplo a pesquisa apresentada por Bentley [30]. As mais simples e conhecidas são as árvores binárias, tendo como propriedades em comum o balanceamento, ou seja, todos os caminhos, desde a raiz (o nodo do topo da árvore) até as folhas (nodos finais na árvore), possuem o mesmo comprimento, conforme ilustrado na Figura 2.4.

Segundo Gaede e Gunther [5], devido à necessidade de indexar dados espaciais, com o decorrer do tempo, diversas estruturas de dados foram desenvolvidas e aperfeiçoadas para organizarem o espaço de forma diferente, sendo as principais: *KD-Tree* [30], *Grid File* [31], *R-Tree* [32], *Hilbert R-Tree* [33] e a *R+-Tree* [34]. Dessa forma, é importante ressaltar que estas estruturas diferem entre si no tipo de objeto que pode ser armazenado e no conceito base utilizado para a organização.

Todavia, embora diversas estruturas de indexação espacial tenham sido propostas, a escolha do melhor índice é uma tarefa complicada, pois cada um se comporta de uma maneira diferente. Pode-se ter, por exemplo, uma estrutura eficiente em consultas sobre pontos, mas ineficiente em relação a polígonos [12].

Nesta dissertação de mestrado serão utilizados os métodos de indexação de objetos espaciais *Grid File* [31], *R-Tree* [32] e *R+-Tree* [34] e, portanto, serão explorados em mais detalhes nas próximas seções.

2.3.2 *Grid File*

Uma das técnicas mais utilizadas para facilitar a pesquisa em dados convencionais é o *hashing* [31]. Este método consiste em criar uma série de *buckets*, que recebem os identificadores dos itens que se quer pesquisar [4]. O *Grid File* particiona o espaço de dados de uma forma que reflete a distribuição de dados em um determinado conjunto. O método é projetado para garantir que qualquer consulta de ponto pode ser respondida em no máximo dois acessos ao disco [3].

O *Grid File* depende de um diretório para identificar a página de dados que contém o ponto desejado. A entrada de diretório da grade identifica a página na qual o ponto desejado é armazenado, se o ponto estiver na base de dados. Para isso, o *Grid File* divide o espaço em regiões retangulares usando linhas que são paralelas aos eixos. Assim,

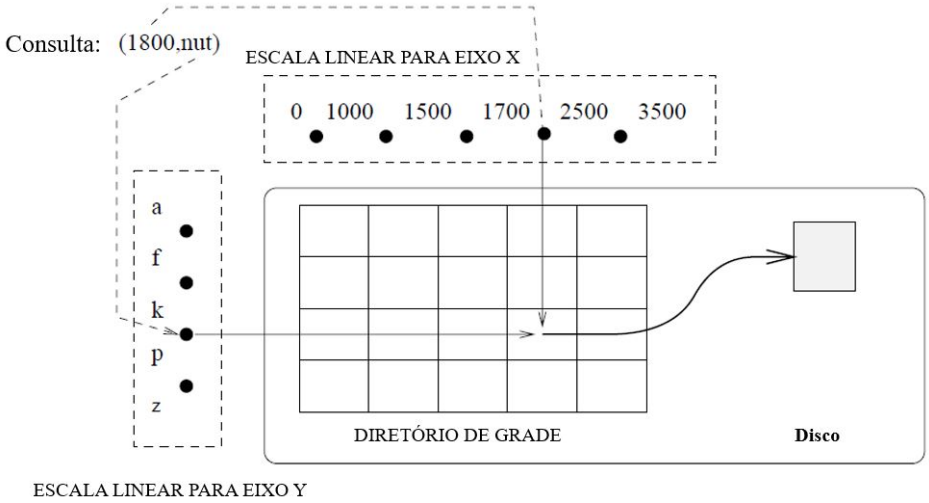

Figura 2.5: Pesquisa em um *Grid File*, reproduzido de [3].

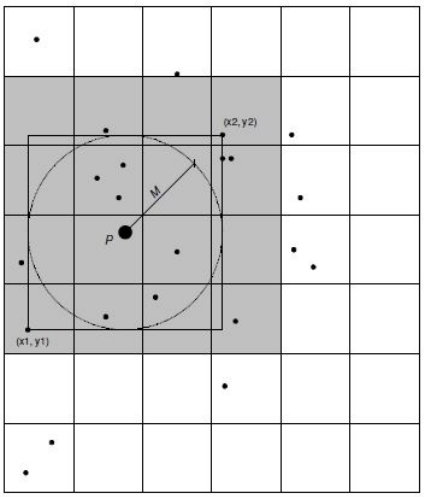

Figura 2.6: Localização de Pontos Próximos, reproduzido de [4].

podemos descrever um particionamento de arquivo de grade, especificando os pontos nos quais cada eixo é "cortado". Se o eixo X é cortado nos segmentos i e o eixo Y é cortado em segmentos j, temos um total de partições i x j. O diretório de grade é uma matriz i por j com uma entrada por partição. Esta descrição é mantida em um *array* chamado escala linear e existe uma escala linear por eixo. Caso o diretório de grade seja muito grande para ficar alocado na memória principal, ele é armazenado no disco, conforme apresentado na Figura 2.5.

Alguns tipos de pesquisas podem ser feitas com muitas facilidades utilizando *grid*, como por exemplo consulta por K vizinhos mais próximos (KNN), conforme apresentada na Figura 2.6 [4].

Todavia, a principal limitação deste método, superada pelos métodos que serão apresentados a seguir, é o fato de somente ser eficiente com pontos, ou com objetos que caibam inteiramente em um quadrado. Não é um método adequado para lidar com linhas ou polígonos, mas é bastante eficiente e simples de implementar para o tratamento de dados do

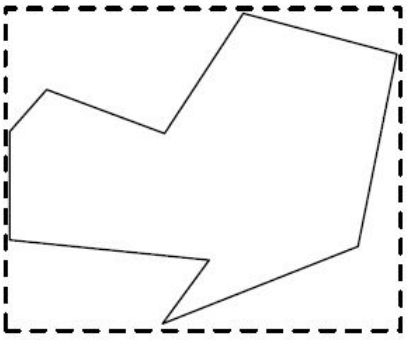

Figura 2.7: Retângulo Envolvente Mínimo, reproduzido de [4].

tipo ponto [4].

2.3.3 *R-Tree*

As limitações encontradas nos mecanismos de indexação, que não possibilitavam pesquisas no espaço multidimensional, tais como a B-Tree [5], motivaram a criação de uma nova estrutura de indexação multidimensional, chamada *R-Tree* [32], que foi considerada um marco no desenvolvimento de estruturas de índices espaciais.

A *R-Tree* não busca organizar exatamente o contorno ou a forma gráfica do objeto, e sim o seu retângulo envolvente mínimo, ou seja, o *Minimum Bounding Rectangle* (MBR). Este retângulo é formado a partir da observação dos limites geométricos mínimo e máximo do contorno do objeto, e é expresso pelas coordenadas dos seus pontos inferior esquerdo e superior direito, conforme apresentado na Figura 2.7. No caso de objetos pontuais, estes extremos vão coincidir com as coordenadas do próprio objeto (portanto, formando um retângulo nulo), mas isto não compromete o método [4].

A indexação *R-Tree* possui uma estrutura hierárquica dinâmica, utilizando os MBRs para organizar objetos espaciais de maneira que os objetos espacialmente próximos fiquem armazenados próximos uns dos outros, promovendo assim, uma redução no espaço de busca a cada nível de árvore [32]. A *R-Tree* é uma árvore balanceada por altura com ponteiros para os objetos espaciais nos nós folhas. A Figura 2.8 ilustra os MBRs (1, 2, 3 e 4) agrupando os objetos espaciais (A, B), (C, D, E), (F, G) e (H, I, J) conforme a sua localização. Uma indexação *R-Tree* possui uma estrutura de dados dividida em dois tipos de nós, os quais são [4]:

- Nós folhas: armazenam informações sobre os objetos indexados na forma (I, id), sendo:
  - I–MBR: que contorna o objeto indexado;
  - Id: identificador do objeto que é uma referência a um endereço de memória que possui os dados do objeto.
- Nós internos: que contém informações da forma (I, p), sendo:
  - I–MBR: que cobre todos os nós dos níveis inferiores;
  - p: ponteiro para um nó de nível inferior.

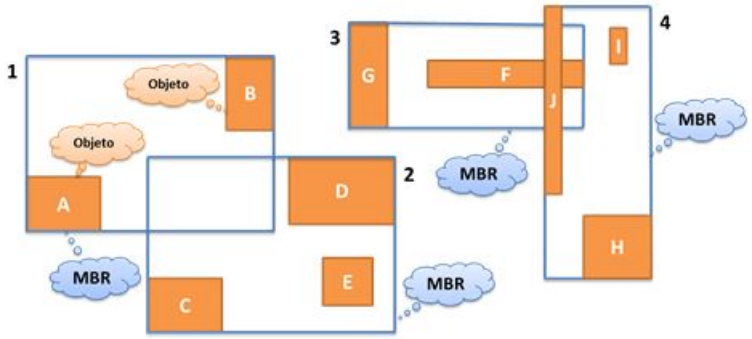

Figura 2.8: MBR Agrupando Objetos, adaptado de [4].

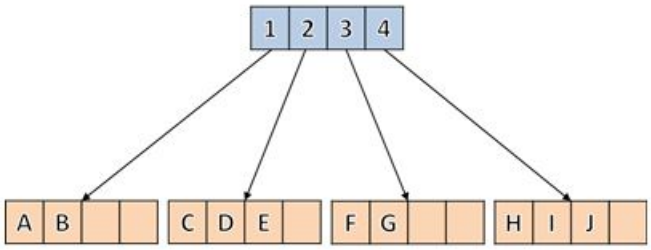

Figura 2.9: Estrutura de Indexação *R-Tree*, adaptado de [5].

Assim, cada nó é armazenado em uma página de disco cuja capacidade máxima é M, e a capacidade mínima é $m < M/2$ de entradas, e devem satisfazer as seguintes propriedades:

- Cada nó folha contém entre m e M elementos, a menos que ele seja uma raiz;
- Para cada elemento em um nó folha, I é o menor MBR que o contém espacialmente;
- Cada nó interno tem entre m e M elemento, a menos que ele seja o nó raiz;
- O nó raiz tem, pelo menos, dois filhos, a menos que ele seja um nó folha;
- Todos os nós folhas estão no mesmo nível da estrutura.

Um problema com as árvores-R é que a ordem de inserção dos objetos interfere na forma final da árvore e, portanto, vai interferir também com o resultado das operações de subdivisão dos nós para manter o balanceamento. Existem algumas técnicas para tentar otimizar este comportamento da *R-Tree*, mas sempre com algum custo adicional em termos de processamento [4].

A Figura 2.9 ilustra a estrutura hierárquica da *R-Tree*, apresentando um nó raiz (1, 2, 3, 4) e nós folhas (A, B), (C, D, E), (F, G) e (H, I, J). A principal função da *R-Tree* é reduzir o espaço de consulta, descartando os objetos que não fazem parte do predicado espacial selecionado. Um grande benefício proporcionado pela *R-Tree* é que sua estrutura permite que vários nós que não fazem parte do critério de busca sejam descartados, diminuindo o acesso a disco. Porém, a sobreposição de nós internos pode fazer com que a busca percorra vários caminhos do nó raiz até os nós folhas.

2.3.4 *R+-Tree*

Conforme apresentado na seção anterior, como os MBRs na *R-Tree* podem estar sobrepostos, quando um determinado objeto espacial é procurado, pode-se ter que verificar mais de um MBR até que o objeto seja encontrado, degradando o tempo da busca.

Devido a esse problema, desde a publicação da *R-Tree* [32], variações foram propostas, sendo a *R+-Tree* [34] uma delas. O foco central é a melhoria do tempo de busca, evitando a verificação de nós, devido ao problema de intersecção de MBR. Desta forma, essa nova variação adiciona uma restrição na representação dos MBRs, ou seja, nenhuma interseção pode ocorrer entre dois MBRs. Com esta restrição é possível melhorar o desempenho da busca. Portanto, um mesmo objeto espacial pode estar contido em mais de um MBR, facilitando a consulta.

Dessa forma, a *R+-Tree* apresenta um ganho de desempenho devido à redução de acessos ao disco, principalmente, em consultas envolvendo pontos, sobre a *R-Tree* [34]. No entanto, para garantir que os retângulos não se interseccionem, a *R+-Tree* força algumas divisões de MBR que seriam desnecessárias, o que causa um maior número de nós e, consequentemente, maior espaço ocupado do índice devido aos espaços não utilizados nos nós.

2.4 Operações e Consultas Geográficas

As operações e as consultas geográficas são funções que utilizam os atributos espaciais e não espaciais das entidades gráficas armazenadas na base de dados espaciais, e buscam fazer simulações (modelos) sobre os fenômenos do Mundo real, seus aspectos ou parâmetros [35].

Neste cenário, diversas são as operações e as consultas disponíveis para serem executadas com dados espaciais. Considerando as consultas e as operações apresentadas no trabalho de Eldawy *et al.* [36], para o desenvolvimento desta pesquisa, sete operações espaciais são utilizadas:

- ***Range Query* (Consulta por Faixa):** sendo R um conjunto de dados espaciais e A um intervalo de entrada, a consulta retorna o conjunto de registros em R que se sobrepõe com faixa A [8], veja a Figura 2.10(a);
- **kNN - *k-Nearest Neighbors* (k vizinhos mais próximos):** esta consulta considera um conjunto de pontos espaciais R, um ponto de consulta P e um inteiro k como entrada, e retorna os k pontos mais próximos de P em R [8], conforme mostrado na Figura 2.10(b);
- ***Farthest Pair* (Par mais longe):** dado um conjunto de pontos P, o par mais distante é o par de pontos que tem a maior distância euclidiana entre eles [36], veja a Figura 2.10(c);
- ***Closest Pair* (Par mais próximo):** dado um conjunto de pontos P, o par mais próximo é o par de pontos que tem a menor distância euclidiana entre eles [36], como mostrado na Figura 2.10(d);
- ***Union* (União):** a união de um conjunto S de polígonos é o conjunto de todos os pontos que tocam em, pelo menos, um dos polígonos em S, no qual apenas

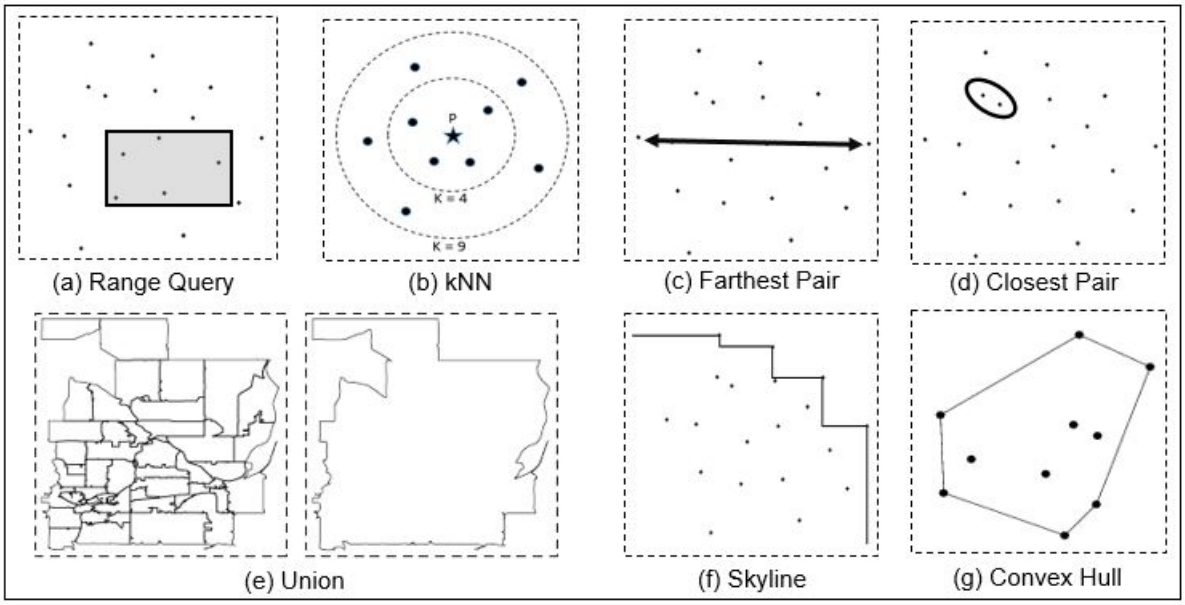

Figura 2.10: Operações com Dados Espaciais.

o perímetro de todos os pontos é mantido, enquanto os segmentos internos são removidos [36], veja a Figura 2.10(e);

- ***Skyline* (Pontos Máximos):** considerando um conjunto de pontos P, é dito que o ponto $p_i \in P$ domina o ponto $p_j \in P$ se cada uma das coordenadas de p_i é maior ou igual a coordenada correspondente de p_j, com estrita desigualdade em pelo menos uma dimensão, o *skyline* de P consiste dos pontos de P que não são dominados por nenhum outro ponto de P [36], conforme apresentado na Figura 2.10(f);
- ***ConvexHull* (Menor Polígono Convexo):** tendo um conjunto de pontos P, o *convexhull* é o menor polígono convexo que contém todos os pontos em P [36], veja a Figura 2.10(g).

Conforme apresentado neste capítulo, os dados geográficos possuem características que requerem métodos de processamento específicos para a execução de operações e de consultas de maneira eficiente.

Neste cenário, o próximo capítulo apresentará a computação em nuvem uma vez que ela possui características que suprem as necessidades requisitadas por aplicações que trabalham com estes tipos de dados, principalmente, para aquelas que processam *big geospatial data*, dada à capacidade computacional exigida.

Capítulo 3

Computação em Nuvem

Neste capítulo é apresentada a definição de computação em nuvem, bem como suas características essenciais, modelos de serviços e modelos de implantação. Também são apresentados os principais provedores públicos de computação em nuvem e as ofertas disponibilizadas por estes provedores de *big geospatial data* como serviço.

3.1 Definição

Diante de um cenário crescente de demanda por recursos computacionais para a execução de aplicações que requerem cada vez mais capacidade de processamento e de armazenamento, surge o conceito de computação em nuvem. De acordo com o *National Institute of Standards and Technology* (NIST) [37], definição mais utilizada na literatura, a computação em nuvem é um modelo que possibilita acesso, de modo conveniente e sob demanda, a um conjunto de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente adquiridos e liberados com mínimo esforço gerencial ou interação com o provedor de serviços.

Outros autores, tais como Fox *et al.* [14], citam que na computação em nuvem os recursos de tecnologia são fornecidos como serviços, permitindo aos usuários acessarem os serviços sob demanda, conforme sua necessidade e, sem precisar ter conhecimento sobre a tecnologia utilizada, fornecendo as características de disponibilidade e de escalabilidade.

Para Vaquero *et al.* [38], a computação em nuvem pode ser definida como um grande conjunto de recursos virtualizados e compartilhados (*hardware*, plataformas de desenvolvimento ou software) que podem ser facilmente acessados e utilizados. Ainda de acordo com este autor, esses recursos podem ser dinamicamente reconfigurados para se ajustarem a uma carga variável de trabalho, permitindo uso otimizado dos mesmos. Este conjunto de recursos é tipicamente explorado por um modelo de pagamento por utilização chamado de *pay-per-use* no qual as garantias são oferecidas pelo provedor de infraestrutura por meio de contratos de serviço personalizados, chamados de *Service Level Agreement* (SLA).

Além disso, NIST [37] descreve que o modelo de computação em nuvem é composto por cinco características essenciais, três modelos de serviços e quatro modelos de implantação, que são descritos nas próximas seções.

3.1.1 Características Essenciais

A computação em nuvem apresenta cinco características essenciais [37]:

- Serviço sob demanda: capacidade de provisionar os serviços de forma direta, sem a necessidade de interação do usuário com o provedor do serviço;
- Acesso amplo a rede: capacidade de acesso aos serviços por meio de diversos dispositivos;
- Agrupamento de recursos: capacidade de agrupamento dos serviços de forma que o mesmo possa atender a diferentes demandas de usuários;
- Elasticidade: capacidade de adquirir e configurar os recursos automaticamente de forma rápida, conforme a sua demanda;
- Serviço medido: capacidade de medição e de controle dos serviços oferecidos, conforme a necessidade do usuário, fazendo com que o mesmo pague somente pelo serviço que usou.

Dessa forma, é importante ressaltar que as principais vantagens da computação em nuvem são a disponibilidade de alto poder computacional e de armazenamento de acordo com a demanda de cada usuário, além do pagamento apenas dos recursos utilizados, o que é fundamental para atingir os objetivos propostos nesta dissertação. Nesse sentido, a característica de elasticidade tem papel fundamental no uso eficiente dos recursos, por isso, será melhor detalhada.

A elasticidade está relacionada ao fato dos recursos computacionais oferecidos pelos provedores de nuvem serem rapidamente alocados e liberados, de acordo com a demanda, passando a ideia de que os recursos são ilimitados e estarão disponíveis a qualquer momento [37]. Assim, Herbst *et al.* [39] definem a elasticidade como o grau no qual um sistema é capaz de se adaptar às mudanças de carga de trabalho através do provisionamento e do desprovisionamento de recursos. Nesse contexto, a elasticidade pode ser obtida de três formas [40, 41, 42]:

- *Scale-out* ou Horizontal: elasticidade obtida ao adicionar novas instâncias (máquinas virtuais) do mesmo modelo ao *cluster*. A redução de recursos, neste modelo, é chamada de *Scale-in*;
- *Scale-up* ou Vertical: neste caso a elasticidade ocorre ao ser adicionado mais recursos computacionais – por exemplo CPU – na instância que está em uso. A redução de recursos é conhecida por *Scale-down*;
- Híbrida: quando as duas formas anteriores (*Scale-out* e *Scale-up*) são utilizadas em conjunto.

Além desta classificação, baseada no tipo da elasticidade, outras classificações também são propostas. Entre elas, Coutinho *et al.* [43] definem uma classificação baseada na política de acionamento, que pode ser manual ou automática, e acrescenta que a elasticidade automática pode ocorrer de maneira proativa – quando a elasticidade é agendada, por exemplo; ou preditiva – quando o aumento ou a redução de recursos é baseado em alguma condição como o aumento de uso de CPU ou memória, por exemplo.

Desta forma, para evitar desperdício de recursos computacionais e, consequentemente, financeiro, é importante que as métricas utilizadas para a elasticidade de recursos sejam atribuídas de acordo com a carga e com o fluxo de trabalho a ser executado, garantindo assim o uso otimizado dos recursos alocados.

3.1.2 Modelos de Serviços

De acordo com NIST [37], a computação em nuvem pode ser entregue em três modelos de serviços:

- Software como Serviço (SaaS – *Software as a Service*): este modelo proporciona que os usuários tenham acesso aos serviços de software, por meio de diversos dispositivos, em qualquer lugar e a qualquer momento, por meio da Internet. Alguns exemplos deste modelo de serviço são: Microsoft Office 365, Google Apps e Dropbox;
- Plataforma como Serviço (PaaS – *Platform as a Service*): este modelo proporciona que os usuário tenham acesso aos serviços de plataforma, fornecendo ao usuário um ambiente de desenvolvimento de aplicações. Alguns desses serviços são: Heroky, Google App Engine e AWS Beanstalk;
- Infraestrutura como Serviço (IaaS – *Infrastructure as a Service*): este modelo garante que os usuários tenham acesso aos serviços de infraestrutura que dará suporte para o SaaS e o PaaS, proporcionando acesso aos recursos, tais como servidores, rede e armazenamento, baseado em técnicas de virtualização de recursos de computação. Assim, são exemplos de IaaS, Amazon AWS, Google Cloud e Microsoft Azure.

3.1.3 Modelos de Implantação

A computação em nuvem oferece quatro modelos de implantação para os seus serviços, sendo eles [37]:

- Privado: o serviço de infraestrutura é disponibilizado especificamente para um público com total restrição de acesso e, exclusivamente, operado pelo mesmo;
- Público: a infraestrutura é provisionada para atender ao público em geral. Ela pode ser provida por empresas, pelo governo ou por centros acadêmicos;
- Comunitário: o serviço de infraestrutura é compartilhado para um público específico, com interesses comuns;
- Híbrido: a infraestrutura é composta por, pelo menos, dois dos modelos citados anteriormente (privada, comunitária e pública). A nuvem permanece como uma única plataforma, permitindo a troca de dados e aplicações.

  Além destes modelos apresentados pelo NIST [37], existe também a implantação de maneira federada. Uma federação de nuvens é um sistema cooperativo de dois ou mais provedores de computação em nuvem. Esta cooperação é interessante por quebrar a dependência de um único provedor, melhorando assim a disponibilidade de serviço da federação, reduzindo custos, e também aumentando o número de combinações de instâncias [44].

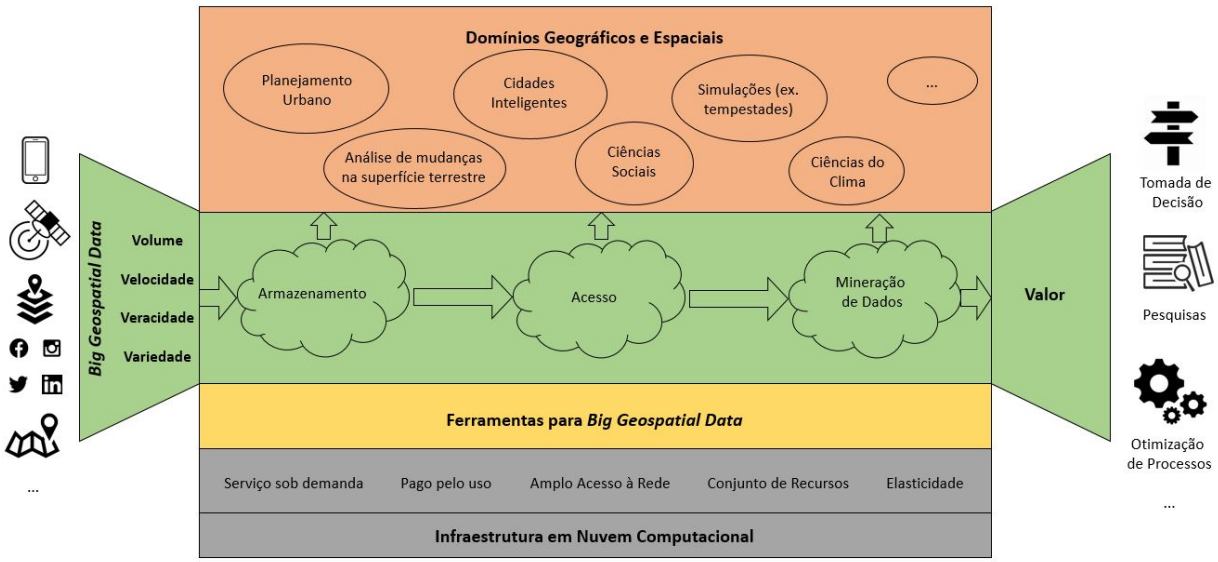

Figura 3.1: Utilização da Computação em Nuvem para o Processamento de *Big Geospatial Data*, adaptado de [6].

Assim sendo, nota-se que a computação em nuvem é um modelo de entrega de poder computacional em forma de serviço que possui características que favorecem o processamento de grandes volumes de dados, tais como a elasticidade para o provisionamento de recursos; o alto poder computacional obtidos através do compartilhamento de recursos; o amplo acesso que permite uma rápida comunicação; a obtenção de recursos de acordo com a demanda; e, por fim, a tarifação baseada apenas nos recursos que foram utilizados.

Por todas estas características, alguns autores como Li *et al.* [45] e Yang *et al.* [6], afirmam que a computação em nuvem é bastante aderente ao processamento de *Big Geospatial Data*, definindo o termo *Spatial Cloud Computing* para o ambiente em que a infraestrutura da computação em nuvem é utilizada em prol da geração de valor para domínios geográficos. Assim, como pode ser observado na Figura 3.1, tendo como insumo um grande volume de dados obtidos das mais variadas fontes, as ferramentas para o processamento de *big geospatial data* se apoiam na infraestrutura da nuvem computacional para armazenamento, acesso e mineração dos dados, com o objetivo de atingir resultados que agreguem valor às organizações, favorecendo a tomada de decisão e a otimização de processos. Desta forma, a Figura 3.1 sintetiza a ideia acerca do termo *Spatial Cloud Computing*.

Neste contexto, como o foco deste trabalho é a otimização no custo para o processamento de *Big Geospatial Data*, são analisados provedores públicos de computação em nuvem, uma vez que neste modelo a tarifação ocorre de forma mais detalhada e clara, conforme apresentado na próxima seção.

3.2 Provedores Públicos de Computação em Nuvem

De acordo com Hofer e Karagiannis [46], os provedores públicos, geralmente, oferecem dois tipos de planos para alocação de recursos: sob demanda e reservados. No plano sob demanda a tarifação é baseada no conceito de "*pay-per-use*", ou seja, o cliente é tarifado por aquilo que realmente utilizar (exemplo, pagar por 2 horas de uso), podendo aumentar e reduzir os recursos alocados conforme a necessidade da aplicação. Já o plano baseado em recursos reservados, tanto a alocação dos recursos quanto a tarifação ocorre por um período maior (por exemplo, pagar por ter os recursos disponíveis por 1 ano). Assim, embora o plano baseado em recursos reservados seja mais econômico do que o plano sob demanda, não é adequado para aplicações que possuem grande variação de processamento e/ou armazenamento, pois a não utilização dos recursos alocados resulta em um desperdício financeiro, e uma demanda abaixo do necessário, podendo causar impactos no desempenho da aplicação.

Além disso, alguns provedores oferecem uma forma de pagamento e obtenção de recursos denominada *spot*, para a aquisição de serviços, exclusivamente, na modalidade *IaaS*. Nesta modalidade, os clientes do provedor fazem lances em recursos e os utilizam sempre que a oferta excede o preço à vista atual, no entanto, os recursos podem ser desalocados quando o preço à vista exceder o lance atual, ou seja, o provedor necessita da capacidade dos recursos alocados por este modelo [47].

Por fim, estes provedores públicos, com o objetivo de facilitar o acesso e a utilização de seus serviços, oferecem serviços específicos para determinadas soluções, tais como bancos de dados, inteligência artificial, computação cognitiva, entre outros. Entre estes serviços, existe também aqueles oferecidos para o processamento de grandes volumes de dados, dada a complexidade que um ambiente para este tipo de aplicação exige, conforme será apresentado na Seção 3.2.1.

Em pesquisa divulgada em Julho de 2017 pelo Gartner Group [48], os três principais provedores públicos de computação em nuvem são: Amazon AWS, Google Cloud e Microsoft Azure. Entre elas, segundo o Gartner Group, a líder é a Amazon AWS, que teve início em 2006 e possui a maior participação no mercado. Em segundo lugar em relação a participação no mercado está Microsoft Azure, que iniciou suas atividades em 2008. Por fim, a Google Cloud tem uma participação mais discreta no mercado, mas é importante considerar que ela iniciou suas ofertas de serviços de nuvem apenas em 2013.

Um ponto comum entre esses três provedores está na abrangência de suas ofertas e a necessidade de constante evolução de seus *datacenters* para permitir o atendimento à crescente demanda [7]. A Figura 3.2 apresenta uma comparação entre as localidades em que cada um destes três provedores possuem *datacenters* instalados. Conforme apresentado, a maior concentração dos *datacenters* está localizada na América do Norte e na Ásia, com predomínio da quantidade de instalações da Microsoft Azure sobre os demais.

3.2.1 *Big Geospatial Data* como Serviço

Entre os provedores públicos, conforme já dito, é prática comum ofertar serviços específicos para determinado produto, facilitando a configuração por parte dos usuários e otimizando os custos. No caso dos serviços oferecidos para o processamento de *Big Data*,

Figura 3.2: *Datacenters* dos provedores Amazon AWS, Google Cloud e Microsoft Azure [7].

e, consequentemente, para *Big Geospatial Data*, o conceito principal está relacionado ao provisionamento e ao redimensionamento do *cluster*, e às plataformas suportadas.

Nesse contexto, um *cluster* é um agrupamento de recursos composto por um servidor principal, responsável pelo gerenciamento da aplicação, e um ou mais servidores secundários, responsáveis pela execução das tarefas da aplicação. O serviço oferecido pela Microsoft Azure é denominado HDInsight e é apresentado como um serviço de análise totalmente gerenciado, completo e de fonte aberta. O provedor Google Cloud oferece o serviço denominado DataProc, sendo ofertado como uma plataforma completa e eficiente para processamento de dados, análise e aprendizado de máquina. Por fim, o provedor Amazon AWS disponibiliza o serviço chamado Elastic MapReduce (EMR), ofertado com o objetivo de processar *big data* com segurança e confiabilidade, incluindo análise de *log*, indexação da web, transformações de dados, aprendizado de máquina, análise financeira, simulação científica e Bioinformática.

A Tabela 3.1 apresenta uma análise comparativa entre os serviços para *Big Data* oferecidos por esses três provedores de computação em nuvem. Conforme pode ser observado na Tabela 3.1, apenas o serviço HDInsight do Microsoft Azure faz a tarifação baseada em minutos, enquanto os demais provedores já precificam seus serviços pelos segundos utilizados pelo *cluster*.

Ainda em relação à análise comparativa realizada com esses provedores, é importante destacar que para o provisionamento de um *cluster*, diversas são as configurações de máquinas virtuais disponíveis nos provedores. Quanto maior a capacidade de processamento (CPU e memória), mais elevado é o preço cobrado por eles. Assim, a Tabela 3.2 apresenta uma comparação entre a máquina virtual com menos recursos e a máquina com mais recursos disponíveis para configuração do *cluster*, bem como seu preço por hora de uso, em Janeiro de 2018.

Os valores apresentados na Tabela 3.2 referem-se ao custo da instância no serviço específico para processamento de grandes volumes de dados de cada provedor. O preço da menor máquina do provedor Google, por exemplo, é composto de $ 1,90 pela hora da

Tabela 3.1: Serviços Oferecidos pelos Provedores de Nuvem Pública para Processamento de *Big Data*.

Provedor	**Microsoft Azure**	**Google Cloud**	**Amazon AWS**
Nome do Serviço	HDInsight	Dataproc	Elastic MapReduce (EMR)
Plataformas Suportadas	Hadoop, Spark, Hive, HBase, Storm, Kafka e R	Hadoop, Spark, Pig e Hive	Hadoop, Spark, HBase, Presto e Flink
Unidade de Tarifação	Minutos	Segundos	Segundos

Tabela 3.2: Comparação de Preço por Hora entre a Menor e a Maior Máquina Virtual Disponível para Provisionamento de um *Cluster* nos Principais Provedores Públicos.

Provedor	**Amazon**		**Azure**		**Google**	
vCPUs	2	64	1	16	1	64
Memória (Gb)	3,75	488	1,75	112	3,75	416
Preço da Instância	$ 0,35	$ 14,89	$ 0,366	$ 5,14	$ 1,90	$ 15,47
Preço do Serviço	$ 0,15	$ 0,945	$ 0,254	$ 4,96	$ 0,04	$ 2,24
Custo Total	$ 0,50	$ 15,83	$ 0,62	$ 10,10	$ 1,94	$ 17,71

instância mais $ 0,04 pelo serviço Dataproc. Assim, como pode ser observado na Tabela 3.2, há uma significativa variação nos preços, por exemplo, a menor instância disponível no provedor Google Cloud oferece apenas 1 vCPU e ainda assim é 533% mais cara que a oferecida pelo provedor Amazon AWS, que, oferece 2 vCPUs.

Com relação ao serviço específico para o processamento de grandes volumes de dados, o maior valor é cobrado pelo Microsoft Azure, com uma variação de 635% em relação ao Google Cloud para a menor instância, e de 525% em relação ao Amazon AWS para a instância mais robusta.

Outra característica importante para estes serviços específicos para *Big Data* está relacionado ao armazenamento dos dados. É essencial que o acesso, a leitura e a escrita sejam feitos de forma eficiente para não causarem impactos no desempenho da aplicação. Cada provedor recomenda a utilização de serviços próprios de armazenamento, no entanto, um ponto comum entre os três provedores avaliados, é que todos utilizam um *storage* externo ao *cluster*, e não os discos internos das instâncias virtuais. Isto garante a persistência do dado, mesmo com o redimensionamento ou até mesmo com a exclusão do *cluster*. Contudo, é importante ressaltar que como o armazenamento não é o foco deste trabalho, ele não será considerado para a elaboração dos custos no decorrer da pesquisa.

As ferramentas, atualmente, disponíveis para o processamento de *big geospatial data* são variações das ferramentas utilizadas para o processamento de *big data* e, portanto, são capazes de aproveitar os serviços oferecidos pelos provedores públicos para este fim, uma vez que utilizam as mesmas características para provisionamento de infraestrutura.

Assim sendo, nos últimos anos, diversas aplicações foram desenvolvidas para otimizar o processamento destes tipos de dados, entre elas:

- *SpatialHadoop* [49]: um *framework* que está incorporado no *Hadoop*, ou seja, implementa as funcionalidades espaciais no interior do núcleo do *Hadoop*, tornando-se bastante eficiente no processamento de consultas espaciais;
- *GeoSpark* [50]: um *framework* que estende as *RDDs* tradicionais do *Spark* para formar as *Spatial Resilient Distributed Dataset* (*SRDD*) e particiona de forma eficiente os elementos de dados entre as instâncias;
- *LocationSpark* [51]: assim como o *GeoSpark*, esta ferramenta também é baseada em *Spark*, sendo desenvolvida como uma biblioteca para executar apenas operações de consulta por faixa, k-vizinhos mais próximos e junções espaciais.

A Tabela 3.3 faz uma comparação das operações disponíveis em cada uma das ferramentas pesquisadas.

Tabela 3.3: Operações Disponíveis nas Ferramentas Pesquisadas.

Operação	***SpatialHadoop***	***GeoSpark***	***LocationSpark***
Range Query	✓	✓	✓
KNN	✓	✓	✓
ConvexHull	✓	✓	
Union	✓	✓	
Skyline	✓	✓	
Farthest Pair	✓		
Closest Pair	✓		

Como pode ser visto na Tabela 3.3, a ferramenta que apresentou a maior abrangência foi o *SpatialHadoop*. Dessa forma, essa foi a ferramenta escolhida para ser utilizada na condução dos testes abordados por este trabalho, por este motivo ela será apresentada em detalhes na próxima seção.

3.2.2 *SpatialHadoop*

O *SpatialHadoop* é um framework completo para o *Hadoop* ser capaz de processar com eficiência dados espaciais, por meio de uma linguagem simples e de alto nível para facilitar a realização das operações e das consultas espaciais.

Uma visão geral da arquitetura do *SpatialHadoop* é apresentada na Figura 3.3. Um *cluster SpatialHadoop* contém um nó mestre que divide um trabalho *MapReduce* em tarefas menores, realizadas por nós escravos. O núcleo de *SpatialHadoop* consiste em quatro camadas [8], que são camada de linguagem, camada de operações, camada *MapReduce* e camada de armazenamento, descritas em detalhes a seguir:

3.2.3 Camada de Linguagem

Para que o *Hadoop* se tornasse popular e possível de ser utilizado por usuários sem conhecimento avançado para a realização de consultas, algumas linguagens *SQL-like* foram

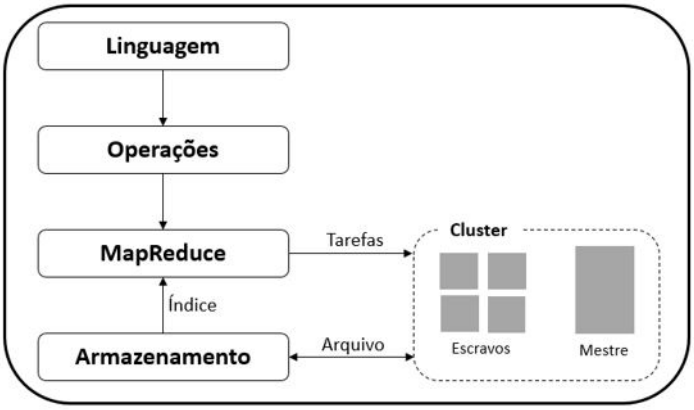

Figura 3.3: Visão Geral das Camadas do *SpatialHadoop*, adaptado de [8].

disponibilizadas, tais como *Pig Latin* [52] e *HiveQL* [53]. Estas linguagens permitiam o uso de funções simples tais como filtro (*filter*), classificação (*sort*) e junção (*join*). Para operações complexas era necessário a combinação destas funções.

No entanto, essas linguagens eram insuficientes para o tratamento de grandes volumes de dados geográficos e espaciais, dificultando o processamento das consultas e das operações. Para isso, foi desenvolvido o *Pigeon* [37], uma linguagem *SQL-like* baseada em *Pig-Latin*, que suporta os tipos de dados padrões do *Open Geospatial Consortiums* (OGC) [9], tais como ponto, linha e polígono, e também possui quatro categorias de funções espaciais, a saber:

- Funções Espaciais Básicas: são utilizadas para obter informações básicas sobre um objeto espacial, tais como o comprimento ou a área;
- Predicados Espaciais: estas funções retornam um valor booleano sobre a relação entre objetos, por exemplo, a função *touches* retorna se o objeto A encosta no objeto B;
- Análises Espaciais: executa algumas transformações espaciais na entrada dos objetos, como calcular o centroide ou a intersecção. Estas funções são normalmente utilizadas para realizar transformações de registos de entrada para produzir a resposta final;
- Funções Agregadoras: retornam valores resultantes da agregação de diversos objetos espaciais, por exemplo, a função *convexhull* retorna um polígono que representa o menor convexo que englobe todos os objetos de entrada.

O *Pigeon* [54] está em conformidade com o padrão *OGC* que é utilizado por Sistemas Gerenciadores de Banco de Dados (SGDB) comerciais e também de código aberto.

3.2.4 Camada de Operações

A camada de operações encapsula a implementação de diversas operações espaciais que utilizam os índices espaciais, e os novos componentes na camada de *MapReduce* [8]. Esta camada é composta por:

- Operações Básicas: entre as diversas operações espaciais disponíveis, três delas foram escolhidas como operações básicas pelo *SpatialHadoop* devido à grande utilização. Estas operações são consulta por faixa (*Range Query*), K-vizinhos mais próximos (KNN) e Junção Espacial (*Spatial Join*);
- CG-Hadoop: um conjunto de algoritmos apresentados por Eldawy *et al.* [36] para operações geográficas que utilizam da indexação do *SpatialHadoop* para obter um bom desempenho. Entre essas operações estão a união de polígonos, *skyline*, *convexhull*, par mais longe e par mais próximo;
- Mineração de Dados Espaciais: operações desenvolvidas para *SpatialHadoop* utilizando técnicas de mineração de dados.

Os índices espaciais introduzidos na camada de armazenamento, juntamente com os novos componentes na camada *MapReduce* possibilitam a realização eficiente de várias operações espaciais em *SpatialHadoop*. Outras operações espaciais podem ser adicionadas à camada de operações utilizando uma abordagem semelhante à execução de operações espaciais básicas [8].

3.2.5 Camada *MapReduce*

Segundo Mokbel e Eldawy [8], a camada de *MapReduce* no *Hadoop* tradicional é projetada para trabalhar com arquivos *heap* não indexados como entrada. No entanto, as operações espaciais em *SpatialHadoop* recebem arquivos indexados espacialmente como entrada, o que requer um tratamento diferente. Além disso, algumas operações espaciais, por exemplo junção espacial, são operações binárias que tomam dois arquivos como entrada.

Para ser capaz de lidar com arquivos indexados espacialmente, o *SpatialHadoop* introduz dois novos componentes na camada de *MapReduce* - *SpatialFileSplitter* e *SpatialRecordReader* - que exploram os índices globais e locais, respectivamente, para o acesso aos dados com mais eficiência [8].

3.2.6 Camada de Armazenamento

Segundo Mokbel e Eldawy [8], na camada de armazenamento, o *SpatialHadoop* adiciona índices espaciais bem adaptados ao paradigma MapReduce. Esses índices superam uma limitação do Hadoop, que provê suporte apenas para arquivos não indexados do tipo *heap*. O *SpatialHadoop* suporta os índices *Grid*, *R-Tree* e *R+-Tree*, que já foram detalhados na Seção 2.3.1.

Existem dois desafios que impedem índices espaciais de serem usados com o *Hadoop* tradicional [8, 55], pois os índices tradicionais são projetados para o paradigma de programação procedural, enquanto o *Hadoop* utiliza programação *MapReduce*. Além disso, os índices tradicionais são projetados para sistemas de arquivos locais, enquanto o *Hadoop* usa o *Hadoop Distributed File System* (HDFS) [56], que é limitado em arquivos que podem ser escritos apenas uma vez e depois não podem ser alterados.

Para superar esses desafios, o *SpatialHadoop* organiza o seu índice em dois níveis: indexação global, que particiona os dados através de nós no *cluster*; e indexação local, que organiza os dados em cada nó escravo [8, 55].

A separação dos índices globais e locais é realizada pelo paradigma de programação *MapReduce*, na qual o índice global é utilizado para preparar o trabalho *MapReduce* e os índices locais são usados pelas funções de processamento de *map* [49].

Uma vez descrito o *SpatialHadoop*, que será a ferramenta utilizada para o desenvolvimento deste trabalho, o próximo capítulo apresenta os trabalhos relacionados aos demais temas desta pesquisa, tais como a comparação de desempenho das indexações de dados geográficos, a comparação dos serviços oferecidos pelos provedores públicos de nuvem computacional e a otimização de custo no processamento de aplicações para *Big Geospatial Data*.

Capítulo 4

Trabalhos Relacionados

Este trabalho apresenta uma Base de Conhecimento e um Mecanismo de Inferência que garantam a otimização do custo para o processamento de *big geospatial data* em provedores públicos de nuvem. Para isto, a pesquisa bibliográfica realizada levou em consideração três pontos:

- A comparação de desempenho na indexação de dados espaciais;
- A comparação entre os serviços oferecidos pelos provedores de nuvem;
- A otimização do uso de recursos computacionais para processamento de grandes volumes de dados.

Assim, a revisão da literatura referente a cada um destes aspectos é apresentada nas próximas seções.

4.1 Comparação de Desempenho na Indexação de Dados Espaciais

A comparação de desempenho na indexação de dados espaciais é tema recorrente na literatura. Normalmente, a cada vez que um novo índice é proposto, ele é comparado com um ou mais índices de conhecimento geral. Em 1990, Smith e Gao [57] fizeram uma comparação de desempenho das indexações *Grid*, *R-Tree*, *R+-Tree* e *zkdB-Tree* para operações de inclusão, exclusão e pesquisa para diversos tipos de dados espaciais. A indexação *Grid* foi a mais performática para as operações de inclusão e exclusão, mas apresentou um desempenho insatisfatória para as operações de pesquisa. As indexações *R-Tree* e *R+-Tree* não tiveram um bom desempenho para inclusão e exclusão, porém foram bem superiores nas pesquisas. Por fim, segundo os autores, embora a indexação *R+-Tree* tenha sido a de melhor desempenho, ela não se mostrou como a melhor opção dado a grande necessidade de espaço em disco requerida por esta indexação.

Oii *et al.* [58] apresentaram uma taxonomia sobre índices espaciais, entre eles o *Grid*, o *R-Tree* e o *R+-Tree*, que são os índices utilizados neste trabalho. Após a realização de diversos testes com os índices analisados, os autores concluiram que independente do método utilizado, o desempenho da indexação é influenciado pelo número de objetos espaciais por unidade de espaço; pelo tamanho dos objetos; e pelo tamanho da base de dados.

Entre os testes realizados pelos autores, foi apresentada uma comparação de desempenho entre as indexações *R+-Tree* e *R-Tree* para as relações topológicas de encontro (*meet*), contorno (*overlap*), inserção (*inside*), cobertura (*covers*), contém ((*contains*) e disjunção (*disjoint*), utilizando bases de dados com 10.000 objetos, com diferentes tamanhos de MBRs e 100 consultas. Para MBRs pequenos e médios, a indexação *R+-tree* teve desempenho superior à *R-tree*. No entanto, para MBRs grandes, o *R+-Tree* é menos eficiente devido ao nível adicional causado pelas duplicações que são características desta indexação, indicando que o *R+-Tree* não é indicado para bases que contenham alta densidade de objetos.

Em 1998, o trabalho apresentado por Gaede e Gunther [5] sobre métodos de acessos multidimensionais, tornou-se uma referência sobre a indexação de conjuntos de dados espaciais. Embora o foco do trabalho não fosse a comparação entre as diversas indexações disponíveis naquela época, os autores destacaram uma seção exclusivamente para relacionar as análises e as comparações realizadas até então pela academia. Dentre estas comparações, foi citada a realizada por Greene [59], que apontou a indexação *R+-Tree* mais performática que a *R-Tree* para conjuntos de dados que possuam menos sobreposições de retângulos.

Teotônio [60] apresentou uma comparação de desempenho dos índices *R-Tree*, *Grid* e *Curvas de Hilbert* para consultas espaciais em bancos de dados geográficos relacionais, utilizando as ferramentas *PostgreSQL*, com extensão espacial *PostGIS* e *MySQL*. Os testes foram realizados em uma única máquina, sem qualquer variação de configuração dos recursos computacionais. A maior das três bases de dados utilizadas continha pouco mais de 1 milhão de registros, sendo composta basicamente por polígonos. Após a realização dos testes, os autores concluiram que a indexação *Grid* é mais performática para conjuntos de objetos nos quais há muitas sobreposições entre os MBRs dos objetos. Já para os conjuntos de objetos que apresentam poucas sobreposições, a *R-tree* foi mais eficiente.

Ainda com relação à comparação de desempenho entre os índices espaciais, no entanto, de maneira mais específica para o processamento de *Big Geospatial Data*, Eldawy *et al.* [55] apresentaram uma comparação entre algumas técnicas de indexação por meio do *SpatialHadoop*, utilizando as operações *Range Query* e *Spatial Join* em cinco bases de dados distintas. Os testes foram executados em ambiente de nuvem alocado no provedor Amazon AWS, sendo que o *cluster* foi composto de 5 a 35 nodos, todos com a mesma configuração de hardware. Entre as conclusões, os autores indicam que embora o *SpatialHadoop* seja escalável, o tempo para indexação não é reduzido de forma proporcional ao incremento no tamanho do *cluster*.

4.2 Comparação entre os Serviços Oferecidos pelos Provedores de Nuvem

Com relação à comparação entre provedores de nuvem, dado o dinamismo e a oferta constante de novos produtos que é realizada por estes provedores, este é um tema também bastante abordado por diversos autores. Em Ang Li *et al.* [45], quatro provedores públicos de nuvem foram comparados (Amazon AWS, Microsoft Azure, Google AppEngine, e Rackspace CloudServers) em quatro funcionalidades: elasticidade, persistência de armazenamento, conexão *intra-cloud* e amplo acesso. Neste cenário, foi apresentada uma

ferramenta chamada CloudCmp, que tem o objetivo de comparar tanto o desempenho quanto o custo de provedores de nuvem e que, nos diversos casos de estudo apresentados, mostraram uma grande diferença entre os custos de cada provedor. Os testes foram realizados com diversas variações da ferramenta, e eles mostraram-se consistentes e a ferramenta demonstrou possuir boa capacidade para *benchmark* de serviços de nuvem. No entanto, à época da realização deste trabalho, ainda eram poucos os provedores que ofereciam serviços voltados ao processamento de *Big Data*, e por isso este tipo de serviço não foi comparado.

A utilização do ambiente de nuvem computacional para aplicações de *Big Data* foi abordado por Ji *et al.* [61] e Kambatla *et al.* [62], focando principalmente em aplicações implementadas em *Hadoop*. Entre os desafios apresentados por Ji *et al.* [61] para melhorar o desempenho do processamento de grandes volumes de dados em ambiente de nuvem através de aplicações *Hadoop* estão: o custo e o tempo necessário para a transferência de dados; a otimização nas iterações dos *jobs* para evitar o desperdício de processamento; a necessidade de processamento em tempo real para algumas aplicações; e, por fim, a necessidade de otimizações na realização de consultas do tipo *join*. Os autores indicaram ainda que, mesmo a computação em nuvem tendo um alto poder computacional, é necessário que as aplicações sejam capazes de simplificar o processamento das informações – indexando as bases de dados a serem processadas, por exemplo – para evitar desperdícios de recursos.

Kambatla *et al.* [62] também propuseram uma otimização no provisionamento de recursos para a execução de *jobs Hadoop* em ambiente de nuvem utilizando o provedor Amazon AWS. Para isto, os autores utilizaram ajustes em parâmetros do *Hadoop*, como por exemplo o número de réplicas dos dados, o número de tarefas *map/reduce* executadas em paralelo etc, para estimarem os recursos adequados para a execução da aplicação, mantendo seu desempenho e evitando o desperdício de recursos. Todavia, embora tenha sido indicado que esta otimização era independente do provedor de nuvem a ser utilizado, não foi realizado nenhum estudo que comprovasse a efetividade da proposta em provedores diferentes.

O mesmo objetivo teve o trabalho apresentado por Zhao *et al.* [63], que propôs um algoritmo denominado *Profit Optimization Resource Scheduling* para a otimização dos recursos alocados em provedores de nuvem para o processamento de *Analytics-as-a-Service* (AaaS). Os testes para a demonstração da aplicabilidade deste algoritmo foram executados no provedor CloudSim, mostrando-se eficaz na otimização da utilização dos recursos. Não houve, no entanto, apresentação de resultados executados em outros provedores de nuvem e também nenhuma indicação dos custos efetivamente reduzidos.

4.3 Otimização do Uso de Recursos Computacionais para Processamento de Grandes Volumes de Dados

Com relação à otimização no uso de recursos computacionais em nuvem para aplicações que processam grandes volumes de dados, Pramila [64] apresentou uma ferramenta comercial chamada Qubole, que garantia um bom desempenho para processamento de aplicações *Hadoop* com o mínimo custo, utilizando a infraestrutura do provedor público

Amazon AWS. As razões para o uso do *Qubole*, ao invés de utilizar as ferramentas nativas para escalonamento do provedor foi motivada pelo fato do escalonamento depender de ações do usuário; por problemas no uso do *HDFS*, inclusive com perda de dados nos *datanodes*; e, por desperdício de recursos e falta de interface que facilite a interação com o usuário. No entanto, não foram apresentados quaisquer resultados de testes comparando a ferramenta *Qubole* com as ferramentas nativas oferecidas pelo provedor, bem como não foi indicado que os problemas encontrados no provedor *Amazon AWS* também ocorriam em outros provedores públicos de nuvem.

De maneira mais relacionada ao processamento de *Big Geospatial Data*, Yang *et al.* [6] apresentaram a utilização da computação em nuvem para auxiliar no processamento de grandes volumes de dados geográficos. Os benefícios da plataforma de nuvem, tais como elasticidade, medição do serviço, serviço sob demanda e pagamento pelo uso (*pay-per-use*) foram evidenciados através de quatro casos de uso, os quais foram estudos climáticos, mineração de conhecimento, análise de mudanças na superfície terrestre e simulação de tempestades. A computação em nuvem se mostrou bastante aderente e adequada para a realização de todos os testes gerados pelos casos de uso propostos. De qualquer forma, não houve preocupação dos autores em relação aos custos gerados pela execução dos testes, focando apenas no poder computacional oferecido pela plataforma.

Por fim, Krämer e Senner [29] apresentaram um sistema modular e flexível, permitindo o uso de diversas plataformas de processamento, tais como o *SpatialHadoop* e *SpatialSpark*. Um dos desafios relatados por Krämer e Senner [29] para a implantação do sistema em ambiente de nuvem está relacionado ao provisionamento e ao gerenciamento dos recursos de infraestrutura. A solução encontrada foi a utilização de uma ferramenta comercial chamada Ansible, que executa *scripts* para provisionamento de recursos em diversos ambientes de nuvem. Ainda sobre o trabalho de Krämer e Senner [29], os autores não apresentaram qualquer teste que demonstrasse a eficiência na alocação dos recursos computacionais em provedores de nuvem.

A Tabela 4.1 apresenta uma comparação entre os trabalhos apresentados nesta seção em relação aos itens relacionados ao escopo do tema de pesquisa desta dissertação.

Assim, por meio da análise dos trabalhos citados nesta revisão da literatura, é possível identificar que, embora o ambiente de nuvem seja bastante indicado para o processamento de *big geospatial data*, não houve, até o momento, um estudo que focasse na otimização dos recursos computacionais neste ambiente, especificamente, para este tipo de aplicação.

Nos trabalhos avaliados nota-se a escassez da utilização da análise da indexação à ser utilizada para organizar o conjunto de dados para, desta forma, obter melhor desempenho na execução de consultas e de operações. Além disso, a utilização do redimensionamento do *cluster* não foi explorada como maneira de otimizar os custos destas aplicações em nenhum trabalho.

Dessa forma, para contribuir para o estado da arte, este trabalho propõe uma Base de Conhecimento e um Mecanismo de Inferência para otimização no custo para processamento de *big geospatial data* em provedores de nuvem. Para isto, é necessário indicar a indexação mais adequada para o conjunto de dados a ser processado e também considerar as operações e as consultas a serem realizadas. Além disso, quando possível, é recomendado utilizar o redimensionamento dos recursos computacionais, minimizando, desta forma, o custo total para o processamento.

Tabela 4.1: Trabalhos Relacionados ao Tema desta Pesquisa.

Paper	**Plataforma em Nuvem**	**Ferramentas**	**Comparação de Indexações**	**Análise de Custo**	**Redimensionamento de Recursos**
[5]			✓		
[6]	✓				
[64]	✓	Hadoop			✓
[29]	✓	SpatialHadoop e SpatialSpark		✓	
[45]	✓	CloudCmp		✓	
[55]	✓	SpatialHadoop	✓		✓
[57]			✓		
[58]			✓		
[60]		PostGIS e MySQL	✓		
[61]	✓	Hadoop		✓	
[62]	✓	Hadoop		✓	✓
[63]	✓			✓	✓

Capítulo 5

Proposta para Otimização no Custo para Processamento de *Big GeoSpatial Data*

A utilização do poder computacional disponibilizado pelos provedores de nuvem é bastante conveniente para diversos tipos de aplicações, inclusive para o armazenamento e o processamento de grandes volumes de dados geográficos. No entanto, conforme apresentado nos capítulos anteriores, dada a tarifação do tipo *"pay-per-use"* dos provedores públicos de computação em nuvem, para a otimização dos custos envolvidos no processamento deste tipo de aplicação, faz-se necessário o uso dos recursos da maneira mais racional possível. Diante disto, o presente trabalho apresenta uma Base de Conhecimento e um Mecanismo de Inferência para a otimização no custo para o processamento de *Big Geospatial Data* alocados em provedores públicos de nuvem.

5.1 Base de Conhecimento

Uma Base de Conhecimento é um conjunto de informações sobre um determinado domínio que são necessárias para auxiliar um Mecanismo de Inferência na tomada de decisões [65]. Especificamente no caso deste trabalho, as informações contidas na Base de Conhecimento devem ser capazes de indicar a melhor indexação a ser aplicada à base de dados, bem como a indicação de redimensionamento do *cluster* para a execução das consultas e das operações, com o objetivo de otimizar o custo para o processamento em ambiente de nuvem computacional.

Assim sendo, para a construção da base de conhecimento foi executada uma série determinada de consultas e de operações espaciais, nos serviços específicos para o processamento de grandes volumes de dados disponíveis nos provedores Amazon AWS, Microsoft Azure e Google Cloud, conforme apresentados na Seção 3.2.1, utilizando o *SpatialHadoop* como ferramenta. Os detalhes sobre as bases de dados e as indexações utilizadas, das operações, das consultas avaliadas e das configurações de *clusters*, são apresentados nas próximas seções.

5.1.1 Bases de Dados

Para a construção da Base de Conhecimento foram utilizadas 33 bases de dados, sendo deste total 16 bases extraídas a partir do US Census Bureau TIGER e 17 bases extraídas do Open Street Map. Com esta variedade foi possível realizar testes com bases de quantidades de registros variados (a menor com 56 registros e a maior com 2,7 bilhões de registros), e com diferentes formas geográficas (pontos, linhas e polígonos). A Tabela 5.1 traz alguns exemplos de bases de dados utilizadas. A relação completa das bases de dados utilizadas está disponível em `http://spatialhadoop.cs.umn.edu/datasets.html`.

Tabela 5.1: Exemplos de Conjuntos de Dados Utilizados nos Testes.

Fonte	**Conteúdo**	**Forma**	**Qtde. Registros**	**Tamanho em Disco**
TIGER	Hidrografia	Linhas	5,8 milhões	6 GB
TIGER	Estradas de Ferro	Linhas	20 milhões	8 GB
TIGER	Estados (EUA)	Polígonos	56	17 MB
OSM	Pontos no planeta	Pontos	2,7 bilhões	96 GB
OSM	Objetos no planeta	Polígonos	263 milhões	92 GB
OSM	Estradas	Polígonos	59 milhões	21 GB

Com o objetivo de agrupar os resultados e facilitar a análise das informações obtidas, as bases de dados foram classificadas de acordo com a forma (Polígono, Linha e Ponto) e com a quantidade de registros contidos nelas (Pequena, Média e Grande), conforme apresentado na Tabela 5.2. Por não ter encontrado na literatura qualquer classificação similar, os valores foram atribuídos baseados na média das bases de dados avaliadas neste trabalho.

Tabela 5.2: Classificação das Bases de Dados.

Nome	**Forma**	**Qtde. Registros**
Poli_P	Polígonos	$\leq$ 1 milhão
Poli_M	Polígonos	$>$ 1 milhão e $\leq$ 100 milhões
Poli_G	Polígonos	$>$ 100 milhões
Linha_P	Linhas	$\leq$ 1 milhão
Linha_M	Linhas	$>$ 1 milhão e $\leq$ 100 milhões
Linha_G	Linhas	$>$ 100 milhões
Ponto_P	Pontos	$\leq$ 1 milhão
Ponto_M	Pontos	$>$ 1 milhão e $\leq$ 100 milhões
Ponto_G	Pontos	$>$ 100 milhões

5.1.2 Estruturas e Índices Utilizados

O *SpatialHadoop*, que foi a ferramenta utilizada na construção da base de conhecimento, tem suporte para as indexações *Grid*, *R-Tree* e *R+-Tree*. Portanto, os testes realizados levaram em consideração estes três tipos de indexação.

Conforme já demonstrado nos artigos publicados em decorrência desta pesquisa [66, 67, 68], a indexação é a tarefa mais onerosa para o processamento de *Big Geospatial Data*, portanto, é fundamental observar o comportamento de cada tipo de base de dados,

tanto pelo seu tamanho quanto pelo seu conteúdo, buscando a indexação mais adequada. Também é importante observar quais são as indexações com melhor desempenho em relação ao tempo e ao custo para cada operação e consulta geográfica.

5.1.3 Operações e Consultas

As operações e as consultas geográficas são as tarefas que retornarão algum resultado para o requisitante, tendo, portanto, papel fundamental na geração de valor para a aplicação.

Para a construção da base de conhecimento foram analisados os desempenhos das seguintes consultas e operações geográficas: *Union*, *Skyline*, *ConvexHull*, *Range Query*, *KNN*, *Closest Pair* e *Farthest Pair*, já apresentadas na Seção 2.4. Estas consultas e operações foram escolhidas por serem operações bastante utilizadas por ferramentas de processamento de dados geográficos [36] e, também, por abrangerem operações com todas as três formas predominantes de um dado geográfico, ou seja, ponto, linha e polígono.

Neste sentido, espera-se determinar o desempenho das indexações para cada uma destas operações e consultas, bem como avaliar o impacto dos recursos utilizados e o tempo necessário para o processamento.

5.1.4 Infraestrutura

Os *clusters* para a realização dos testes foram alocados nos provedores Amazon AWS, Microsoft Azure e Google Cloud, utilizando os serviços específicos oferecidos por estes provedores para o processamento de grandes volumes de dados, conforme apresentados na Seção 3.2.1.

Desta forma, foi possível comparar o desempenho da aplicação, apontar os pontos fortes e os pontos fracos de cada um destes provedores, bem como mensurar o custo que é o foco deste trabalho.

Assim, embora os provedores ofereçam diversos tipos de instâncias, variando a quantidade de *virtual Central Processing Unit* (vCPUs) e memória, para a realização dos testes foram padronizadas configurações para o *masternode* que deveria ter 8 vCPUs e 28GB de memória; e para cada *datanode*, que foi especificado com 4 vCPUs e 14GB de memória. O custo apresentado na Tabela 5.3 tem como referência o mês de Janeiro/2018.

Tabela 5.3: Preços Considerados para a Realização dos Testes.

Instância	**Amazon AWS**	**Microsoft Azure**	**Google Cloud**
Master Node	$ 0,42	$ 1,24	$ 0,28
Data Node	$ 0,21	$ 0,62	$ 0,14

O provedor Microsoft Azure, através do serviço Azure HDInsight, permite apenas a criação de *clusters* Hadoop utilizando, no mínimo, 2 *masternodes* [69], o que faz com que o seu custo seja o mais alto entre os provedores analisados.

5.1.5 Conhecimentos Obtidos

Considerando todos os parâmetros apresentados nos tópicos anteriores – bases de dados, indexações, operações, consultas e infraestrutura – foram realizados diversos testes com

o objetivo de construir a Base de Conhecimento. Os resultados apresentados a seguir representam uma média de, pelo menos, três execuções, realizadas nos três provedores avaliados – Azure, AWS e Google.

Os resultados obtidos com as execuções dos testes demonstram que:

- o aumento de instâncias no *cluster* reduz o tempo de processamento, mas aumenta o custo total;
- a criação do índice *Grid* é mais performática para bases que tenham apenas pontos;
- a criação do índice *R-Tree* é mais performática para bases com linhas e polígonos;
- a indexação *R+-Tree* é melhor para *Union*, *Skyline* e *ConvexHull*;
- a indexação *Grid* é melhor para *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair*;
- o tempo para as operações *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair* é pequeno após a indexação.

Assim, para melhor entendimento dos itens elencados, cada um será explicado nos tópicos abaixo.

a) O aumento de instâncias no *cluster* reduz o tempo de processamento, mas aumenta o custo total

Uma das características da computação em nuvem está na capacidade de expandir e retrair o poder de processamento dinamicamente. Uma das opções para isto, chamada de elasticidade horizontal, trata-se da adição de novas instâncias ao conjunto de recursos já em uso [14].

A Tabela 5.4 e a Figura 5.1 traz o tempo (em minutos) e o custo consumidos para a indexação R-Tree de uma base classificada como Poli_G, realizado no provedor Azure. Não foi considerado, neste cenário, o tempo para o provisionamento do *cluster*.

Os testes realizados demonstraram que, embora a adição de novas instâncias do tipo *datanodes* no *cluster SpatialHadoop* seja capaz de reduzir o tempo total de processamento, o aumento de custo gerado pela inclusão de uma nova instância não é deduzido pelo tempo reduzido, gerando, portanto, aumento no custo total.

Tabela 5.4: Tempo (em minutos) e Custo para Indexação com Incremento no Número de *Datanodes*.

	Indexações					
	Grid		**R-Tree**		**R+-Tree**	
Datanodes	**Tempo**	**Custo**	**Tempo**	**Custo**	**Tempo**	**Custo**
6	51	$ 5,29	47	$ 4,87	49	$ 5,08
9	47	$ 6,33	42	$ 5,66	45	$ 6,06
12	42	$ 6,97	38	$ 6,30	40	$ 6,63

Atualmente, todos os serviços para o processamento de grandes volumes de dados avaliados nos três provedores, permitem apenas a utilização da elasticidade horizontal de

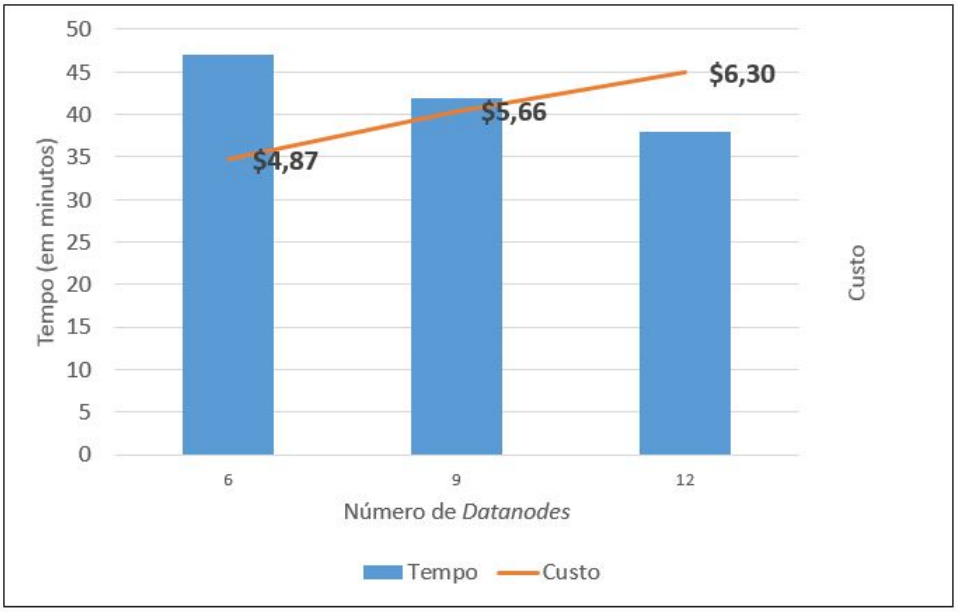

Figura 5.1: Tempo e Custo com Elasticidade Horizontal, para a Indexação R-Tree.

recursos, ou seja, o redimensionamento do *cluster* é restrito à inclusão ou à exclusão de instâncias.

Desta forma, a elasticidade vertical de recursos, que ocorre quando há incremento dos recursos computacionais da instância atual (por exemplo, o aumento de memória), mas sem adicionar novas máquinas ao *cluster*, ainda não está disponível nos serviços avaliados.

b) a criação do índice *Grid* tem o melhor desempenho para bases que tenham apenas pontos

Considerando todas as bases de dados analisadas, independente de seus tamanhos, é possível afirmar que a tarefa de criação do índice *Grid* teve o melhor desempenho para aquelas que possuiam apenas pontos, podendo esta diferença chegar até 13% quando comparada com a indexação *R-Tree*, e 9% para a indexação *R+-Tree*, veja a Tabela 5.5.

Tabela 5.5: Tempo (em segundos) para a Criação do Índice das Bases de Dados.

Base	*Datanodes*	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	1	1969	1814	1896
Poli_M	2	2050	1886	1975
Poli_G	6	3031	2789	2920
Ponto_P	2	1780	2023	1948
Ponto_M	5	2940	3322	3205
Ponto_G	7	4752	5369	5179
Linha_P	1	1843	1467	1588
Linha_M	2	2256	1797	1943
Linha_G	4	3849	3063	3317

Assim, conforme apresentado na Seção 2.3.1, a superioridade da indexação *Grid* sobre as demais para o processamento de bases de dados que contenham apenas pontos deve-se ao fato de serem objetos pequenos, capazes de serem armazenados em um *bucket*.

c) a criação do índice *R-Tree* tem o melhor desempenho para bases com linhas e polígonos

Para as bases de dados que possuem polígonos e linhas, independente do tamanho destas bases, a tarefa de criação do índice com melhor desempenho foi a *R-Tree*. Para polígonos, o desempenho da indexação *R-Tree* é 8% superior quando comparado a indexação *Grid*, e 4% superior quando comparado com a indexação *R+-Tree*. Já para as bases de dados com linhas, esta diferença sobe para 25% quando comparado com a indexação *Grid*, e para 8% quando comparado com a indexação *R-+Tree*.

Esta superioridade está relacionada ao fato de serem usados os MBRs, conforme apresentado na Seção 2.3.1. Além disso, por serem formas mais complexas, a indexação *Grid* não é capaz de armazená-las em um único *bucket*, tendo portanto, um desempenho mais onerosa para estas formas.

d) A indexação *R+-Tree* é melhor para *Union*, *Skyline* e *ConvexHull*

Considerando o desempenho para a execução das operações geográficas testadas, as bases indexadas com *R+-Tree* tiveram melhor desempenho para a execução das operações *Union*, *Skyline* e *ConvexHull*, conforme demonstrado na Figura 5.2. É importante ressaltar que a operação *Union* é realizada apenas com bases que sejam compostas por polígonos, enquanto as operações *Skyline* e *ConvexHull* são executadas com bases que contém pontos.

A busca otimizada pelo *R+-Tree* em relação às demais indexações é responsável por esta superioridade, uma vez que ele consegue realizar a consulta com menos acesso ao disco, conforme apresentado na Seção 2.3.1.

e) A indexação *Grid* é melhor para *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair*

Diante dos diversos testes realizados, foi possível observar que a indexação *Grid* teve melhor desempenho para a execução das consultas *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair*, conforme apresentado na Figura 5.3.

Esta superioridade está relacionada ao fato destas operações serem executadas em bases de dados compostas principalmente por pontos e, conforme apresentado na Seção 2.3.2, o *Grid* ser bastante eficiente para consutlas e operações nas quais os objetos caibam inteiramente em um quadrado da grade.

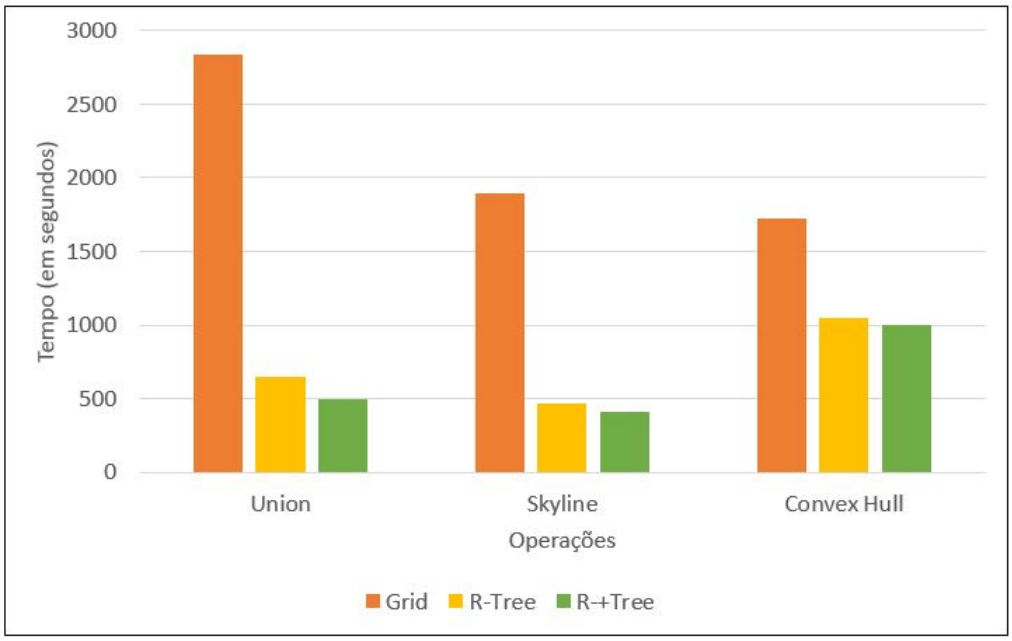

Figura 5.2: Comparação de Indexação para Operações *Union*, *Skyline* e *ConvelHull*.

Figura 5.3: Comparação de Indexação para Consultas *Range Query*, *KNN*, *Closest Pair* e *Farthest Pair*.

f) O tempo para as operações *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair* é pequeno após a indexação

Por fim, foi possível observar que a indexação é a tarefa que consome mais tempo entre todas as demais operações e consultas analisadas. Uma vez indexada a base de dados, as operações *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair* são realizadas em pequenos tempos. Apenas as operações *union*, *skyline* e *convexhull* requerem maior tempo de processamento, conforme demostrado nas Tabelas de 5.6 a 5.12.

Tabela 5.6: Tempo Médio (em segundos) para Execução de *Range Query*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	3	5	4
Poli_M	3	9	6
Poli_G	13	25	23
Ponto_P	4	7	7
Ponto_M	4	9	12
Ponto_G	7	14	13
Linha_P	2	8	5
Linha_M	6	7	8
Linha_G	5	12	11

Tabela 5.7: Tempo Médio (em segundos) para Execução de *KNN*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	3	7	7
Poli_M	5	14	12
Poli_G	14	35	27
Ponto_P	2	5	5
Ponto_M	5	6	8
Ponto_G	9	21	19
Linha_P	3	4	4
Linha_M	4	6	8
Linha_G	6	6	7

Desta forma, é possível afirmar que, dependendo das operações a serem executadas após a indexação da base, é possível redimensionar o *cluster*, reduzindo a quantidade de *datanodes*, por exemplo, possibilitando assim, a redução dos custos para o processamento destas consultas e operações.

E isto é uma grande contribuição deste trabalho, uma vez que os trabalhos já apresentados sobre o processamento de *big geospatial data* mantinham todas as instâncias do *cluster* ativas até a conclusão de todas as tarefas.

5.2 Mecanismo de Inferência

Um mecanismo de inferência é um algoritmo capaz de elaborar as conclusões a partir das variáveis fornecidas pelo usuário, e das informações que estão armazenadas na Base de

Tabela 5.8: Tempo Médio (em segundos) para Execução de *Closest Pair*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	3	3	4
Poli_M	6	8	8
Poli_G	9	11	14
Ponto_P	3	5	5
Ponto_M	3	5	4
Ponto_G	5	5	8
Linha_P	2	4	5
Linha_M	3	4	5
Linha_G	7	12	14

Tabela 5.9: Tempo Médio (em segundos) para Execução de *Farthest Pair*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	4	4	7
Poli_M	6	13	9
Poli_G	10	23	17
Ponto_P	2	5	5
Ponto_M	3	5	5
Ponto_G	6	11	10
Linha_P	2	5	7
Linha_M	4	7	6
Linha_G	4	8	7

Tabela 5.10: Tempo Médio (em segundos) para Execução de *Union*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Poli_P	2460	598	456
Poli_M	2841	636	486
Poli_G	2953	648	492

Tabela 5.11: Tempo Médio (em segundos) para Execução de *Skyline*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Ponto_P	1627	425	396
Ponto_M	1840	471	412
Ponto_G	1896	488	425

Tabela 5.12: Tempo Médio (em segundos) para Execução de *ConvexHull*.

Base	*Grid*	*R-Tree*	*R+-Tree*
Ponto_P	1566	1030	980
Ponto_M	1598	1048	996
Ponto_G	1722	1125	1050

Conhecimento [65]. Para o desenvolvimento do Mecanismo de Inferência desta pesquisa serão consideradas as seguintes variáveis:

- *forma*: contém a informação do tipo de forma predominante na base de dados (pontos, linhas ou polígonos);
- *operações*: contém as operações e as consultas a serem executadas com a base de dados;
- *index_ sugerida*: resultado na aplicação das regras do mecanismo de inferência. Indicará a indexação sugerida a ser utilizada, baseada no conteúdo da base de dados e também nas operações e consultas a serem realizadas;
- *redimensionamento*: é um campo booleano, contendo a indicação para o redimensionamento do *cluster* após a execução da tarefa de indexação.

Também foram elaboradas quatro regras que têm o objetivo de otimizar o custo ou o tempo de processamento de operações e consultas geográficas. As regras definidas são:

Regra 1

1: **Se** forma == pontos $\wedge$ operações ! = (union $\vee$ skyline $\vee$ convexhull) **Então**
2: index_sugerida $\leftarrow$ grid
3: redimensionamento $\leftarrow$ verdadeiro

Regra 2

1: **Se** forma == pontos $\wedge$ operações == (union $\vee$ skyline $\vee$ convexhull) **Então**
2: index_sugerida $\leftarrow$ r+tree
3: redimensionamento $\leftarrow$ falso

Regra 3

1: **Se** forma != pontos $\wedge$ operações ! = (union $\vee$ skyline $\vee$ convexhull) **Então**
2: index_sugerida $\leftarrow$ r-tree
3: redimensionamento $\leftarrow$ verdadeiro

Regra 4

1: **Se** forma != pontos $\wedge$ operações == (union $\vee$ skyline $\vee$ convexhull) **Então**
2: index_sugerida $\leftarrow$ r+tree
3: redimensionamento $\leftarrow$ falso

A Base de Conhecimento e o Mecanismo de Inferência propostos neste capítulo objetivam a otimização no custo financeiro para o processamento de *Big Geospatial Data* quando executados em provedores públicos de computação em nuvem. O escopo da Base de Conhecimento é limitado às bases de dados com dados geográficos do tipo vetorial (pontos, linhas e polígonos), às indexações executadas, aos serviços oferecidos pelos provedores avaliados, à ferramenta *SpatialHadoop*.

No Capítulo 6 será apresentada uma avaliação com o objetivo de demonstrar a eficiência, e garantir que os objetivos deste trabalho tenham sido atingidos.

Capítulo 6

Avaliação de Resultados

Para a avaliação da Base de Conhecimento e do Mecanismo de Inferência propostos, foram realizados diversos testes cujos resultados são reportados neste capítulo. O objetivo é mostrar que a utilização de ambos resulta em reduções nos custos para o processamento de operações e de consultas espaciais para *big geospatial data* em provedores públicos de nuvem computacional.

6.1 Ambiente de Testes

Os testes foram executados nos três provedores públicos de computação em nuvem avaliados nesta pesquisa – Amazon AWS, Microsoft Azure e Google Cloud. Em todos os provedores foram utilizadas, inicialmente, uma instância para o *masternode* tendo 8 vCPUs e 28GB de memória; e três instâncias como *datanode*, sendo essas compostas com 4 vCPUs e 14GB de memória. O custo das instâncias foi apresentado na Tabela 5.3.

Também foram utilizadas três bases de dados, todas extraídas aleatoriamente do *Open Street Maps*, e diferentes das utilizadas na construção da base de conhecimento. As bases usadas também foram capazes de representar todas as formas geográficas (pontos, linhas e polígonos), com quantidades de registros variadas, conforme Tabela 6.1.

Além disso, foram definidas, três sequências de operações e consultas a serem realizadas com as bases de dados, que são apresentadas na Tabela 6.2. Elas foram elaboradas com o objetivo de executarem operações e consultas mais leves em relação ao poder de processamento exigido (Sequência 1), mais complexas (Sequência 2) e a uma combinação de ambos os tipos (Sequência 3).

O fluxo esperado para a execução dos testes é apresentado na Figura 6.1. Ele inicia-se com o provisionamento do *cluster* no provedor de nuvem, ou seja, a alocação das instâncias. Em seguida, baseado no resultado gerado pela aplicação das regras do Mecanismo de Inferência, a base de dados é indexada com o índice de melhor desempenho. Caso seja indicado, pelas regras aplicadas, o redimensionamento do *cluster* é aplicado. Na sequência, são realizadas as consultas e as operações, os resultados são salvos e o *cluster* é, então, encerrado. Desta forma, este fluxo compreende todo o período em que ocorre a tarifação por parte dos provedores, ou seja, do provisionamento até o encerramento do *cluster*.

É importante ressaltar que os resultados apresentados nesta seção representam a média de, no mínimo, três execuções de cada uma das sequências em cada um dos provedores avaliados.

Tabela 6.1: Conjuntos de Dados Utilizados na Avaliação.

Nome	**Forma**	**Qtde. Registros**
Base 1	Linhas	15 milhões
Base 2	Polígonos	200 milhões
Base 3	Pontos	3 bilhões

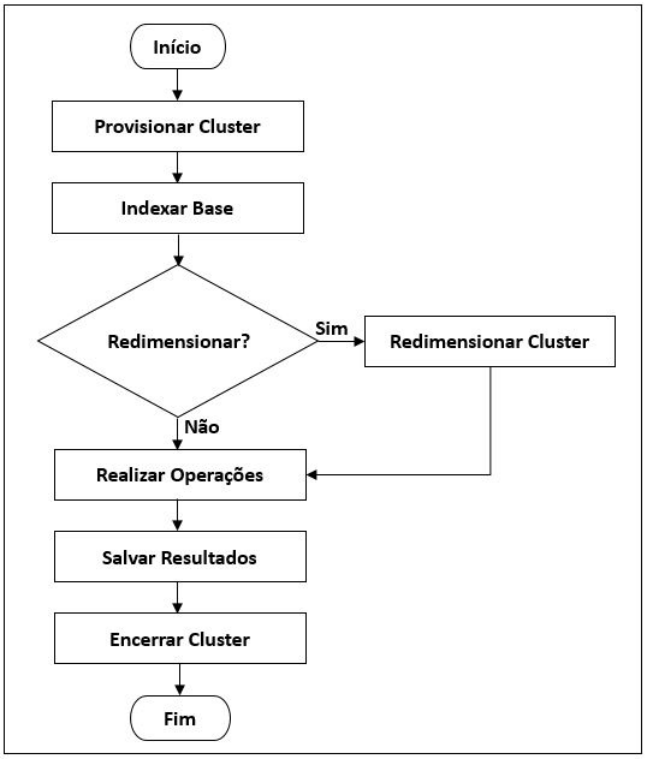

Figura 6.1: Sequência de Execução Esperada para a Aplicação.

Tabela 6.2: Sequências de Consultas e Operações.

Sequência 1	**Sequência 2**	**Sequência 3**
KNN (100x)	Union	KNN (100x)
Range Query (100x)	Skyline	Range Query (100x)
Closest Pair	Convex Hull	Closest Pair
Farthest Pair		Farthest Pair
		Union
		Skyline
		Convex Hull

A ordem das operações em cada uma das sequências foi definida de maneira aleatória, sem considerar a possibilidade de aproveitamento de leitura e escrita em memória em memória de atributos utilizados pelas consultas e operações já realizadas.

6.2 Resultados

A Base de Conhecimento e o Mecanismo de Inferência propostos neste trabalho foram utilizados para executar as consultas e as operações constantes nas sequências apresentadas na Tabela 6.2. Os resultados apresentados na Tabela 6.3 indicam as sugestões de indexação e de redimensionamento do *cluster* que foram obtidas, após a aplicação das regras que compõem o Mecanismo de Inferência.

Tabela 6.3: Indexações e Redimensionamento Sugeridas.

Sequência de Operações	**Base**	**Índice Sugerido**	**Redimensionar?**
Sequência 1	Base 1	*R-Tree*	Sim
	Base 2	*R-Tree*	Sim
	Base 3	*Grid*	Sim
Sequência 2	Base 1	*R+-Tree*	Não
	Base 2	*R+-Tree*	Não
	Base 3	*R+-Tree*	Não
Sequência 3	Base 1	*R+-Tree*	Não
	Base 2	*R+-Tree*	Não
	Base 3	*R+-Tree*	Não

Considerando que a Sequência 1 é a única que é composta unicamente por operações menos onerosas computacionalmente, para todas as bases utilizadas, independente da forma que a compõe, é sugerido o redimensionamento do *cluster*, objetivando a redução do custo total. Neste caso, o *cluster* foi redimensionado para apenas 1 *masternode* e 1 *datanode* para a execução das consultas e das operações. Para as sequências 2 e 3 há predominância da aplicação da indexação *R+-Tree*, que é a mais indicada para o processamento das operações mais onerosas, tais como *Union*, *Skyline* e *Convex Hull*, por isso o redimensionamento não foi necessário.

6.2.1 Comparação Entre os Provedores

De acordo com as indicações apresentadas na Tabela 6.3, foi realizada uma comparação em relação ao custo para a execução das operações e das consultas nos três provedores avaliados. Para tanto, a Tabela 6.4 apresenta o tempo e o custo total para execução de cada uma das sequências, para as três bases de dados, em cada provedor. É importante ressaltar que, neste caso, está incluído o tempo necessário para o provisionamento, o redimensionamento (quando for o caso) e a exclusão do *cluster*, além da indexação da base de dados.

Entre os provedores avaliados, o provedor Google Cloud apresentou a melhor relação custo/benefício, embora não seja o mais performático. O melhor desempenho em relação ao tempo de processamento foi obtida pelo provedor Microsoft Azure, porém com o custo sendo o mais elevado.

A Figura 6.2 apresenta um comparativo entre o tempo de processamento e o custo obtido para o processamento da Sequência 1 para a Base 1 nos provedores avaliados. Neste cenário, embora a execução do processamento tenha sido 3% mais rápida no provedor Azure quando comparado ao Google Cloud, utilizando 4.293 segundos, o custo total foi mais de 430% mais caro, totalizando $ 221,81.

Tabela 6.4: Tempos (em segundos) e Custos Obtidos na Execução das Sequências nos Provedores Selecionados.

Provedor	Sequência	Base 1		Base 2		Base 3	
		Tempo	Custo	Tempo	Custo	Tempo	Custo
AWS	1	4.660	$ 69,25	9.309	$ 98,92	8.754	$ 115,65
	2	5.145	$ 90,04	5.221	$ 91,37	7.013	$ 122,73
	3	6.965	$ 121,89	10.252	$ 179,41	10.231	$ 179,04
Azure	1	4.293	$ 185,50	9.123	$ 282,44	8.512	$ 328,93
	2	4.994	$ 258,02	4.985	$ 257,56	6.789	$ 350,77
	3	6.850	$ 353,92	9.439	$ 487,68	9.744	$ 503,44
Google	1	4.420	$ 43,37	9.455	$ 67,65	9.013	$ 80,12
	2	5.150	$ 60,08	5.474	$ 63,86	7.522	$ 87,76
	3	7.063	$ 82,40	10.751	$ 125,43	10.401	$ 121,35

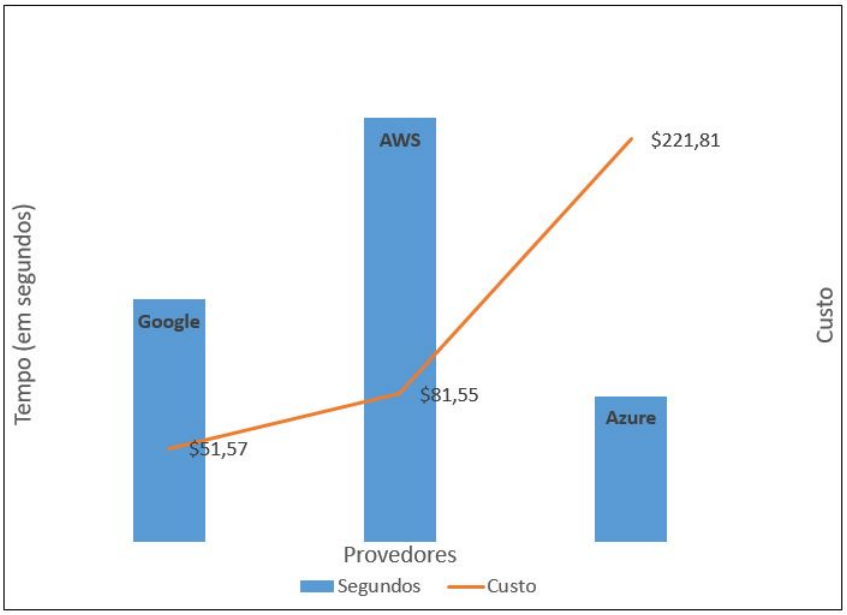

Figura 6.2: Tempo e Custo nos Provedores para Base 1 e Sequência 1.

A Figura 6.3 faz a comparação entre o tempo e o custo para processamento das consultas e operações da Sequência 3, que é a mais complexa de todas, para a Base 3, que é a base de dados com maior número de registros, com predominância da forma "ponto". Embora, neste caso, o tempo para processamento no provedor Google tenha sido 1,7% maior em relação ao obtido pelo AWS, o custo total no provedor Google foi 32,2% menor em relação ao custo apresentado pelo AWS, e 75,9% em relação ao custo do Azure.

6.2.2 Eficiência da Base de Conhecimento e do Mecanismo de Inferência

Para validar a eficiência da Base de Conhecimento e do Mecanismo de Inferência, o mesmo cenário de testes (instâncias, bases de dados e sequências de operações) foi executado nas mesmas condições e cenários, porém, sem levar em consideração a indicação de indexação

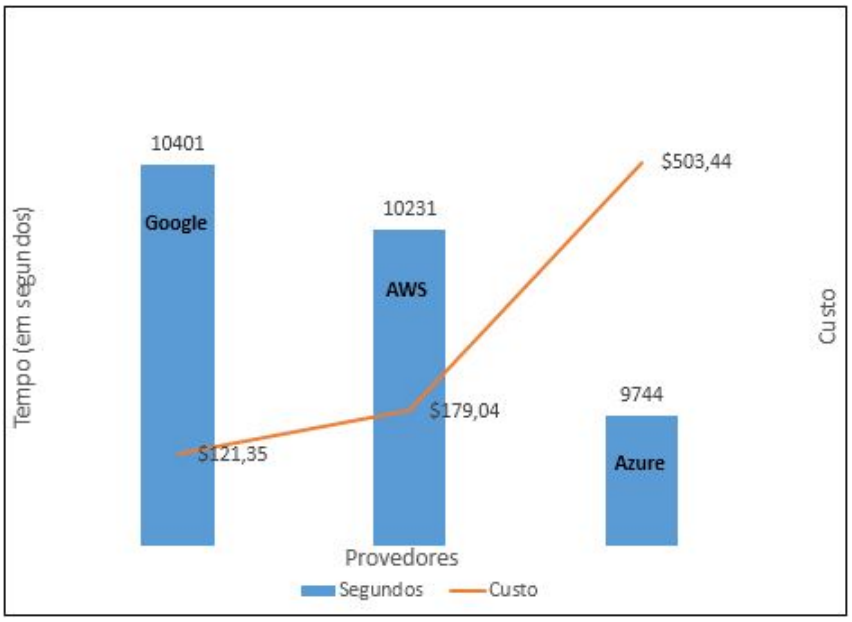

Figura 6.3: Tempo e Custo nos Provedores para Base 3 e Sequência 3.

que seria oferecida pelo seu uso. A Tabela 6.5 apresenta as indexações que foram utilizadas. Além disso, é importante ressaltar que para nenhuma das sequências foi utilizado o redimensionamento do *cluster*.

Tabela 6.5: Indexações Utilizadas para Validação.

Sequência de Operações	**Base**	**Índice**
Sequência 1	Base 1	*Grid*
	Base 2	*R+-Tree*
	Base 3	*R-Tree*
Sequência 2	Base 1	*Grid*
	Base 2	*R-Tree*
	Base 3	*Grid*
Sequência 3	Base 1	*Grid*
	Base 2	*R-Tree*
	Base 3	*Grid*

Os tempos e os custos resultantes da utilização destes novos parâmetros, na execução das sequências, são apresentados na Tabela 6.6. Todos os tempos e, consequentemente, os custos são superiores quando comparados à Tabela 6.4.

A comparação das Tabelas 6.4 e 6.6 permite ainda observar que, a indexação tem influência direta no desempenho das consultas e das operações realizadas. Note, por exemplo, o tempo de execução da Sequência 3 para a Base 3, que é composta por pontos. Considerando apenas a composição da base de dados, a indexação sugerida seria a *Grid*. No entanto, esta indexação não é performática para as operações *union*, *skyline* e *convexhull*, que estão presentes na sequência, e, portanto, a indexação mais adequada é a *R+-Tree*. Desta forma, utilizar a indexação *Grid* ao invés da *R+-Tree* aumenta o custo e o tempo total de execução em aproximadamente 2%.

Tabela 6.6: Tempos e Custos Obtidos na Execução das Sequências com Novos Parâmetros.

Provedor	Sequência	Base 1		Base 2		Base 3	
		Tempo	Custo	Tempo	Custo	Tempo	Custo
AWS	1	5.284	$ 92,47	9.560	$ 167,30	9.311	$ 162,94
	2	7.058	$ 123,52	5.358	$ 93,77	8.796	$ 153,93
	3	8.878	$ 155,37	10.389	$ 181,81	10.414	$ 182,25
Azure	1	5.123	$ 264,69	9.379	$ 484,58	9.128	$ 471,61
	2	6.852	$ 354,02	4.987	$ 257,66	8.477	$ 437,98
	3	8.411	$ 434,57	10.103	$ 521,99	9.983	$ 515,79
Google	1	5.389	$ 62,87	9.720	$ 113,40	9.354	$ 109,13
	2	7.199	$ 83,99	5.791	$ 67,56	8.912	$103,97
	3	8.893	$ 103,75	10.925	$ 127,46	10.423	$ 121,60

Figura 6.4: Comparação do Uso da Base de Conhecimento e do Mecanismo de Inferência.

Ainda considerando as informações da Tabela 6.6, também é possível notar que houve um aumento no custo até mesmo das sequências em que não ocorreram mais o redimensionamento do *cluster*, e que, portanto, deveriam ser executadas em tempos menores. Isto porque embora o tempo das operações e das consultas tenha sido reduzido, o tempo para indexação foi maior e, além disso, o custo do *cluster* também foi maior durante a execução das consultas e das operações, pois não houve otimização no uso dos recursos alocados, conforme apresentado na Figura 6.4.

A Figura 6.5 apresenta uma comparação do tempo (em segundos) para a execução de todas as sequências para a Base 3 no provedor Microsoft Azure, quando utilizada a sugestão de indexação gerada pela aplicação das regras do Mecanismo de Inferência, e quando elas não são utilizadas. Assim como ocorreu em todos os outros provedores, foi possível notar um aumento no tempo necessário para a execução das operações.

Pelos testes realizados é possível concluir que a Base de Conhecimento e o Mecanismo de Inferência propostos otimizam o custo para o processamento de consultas e de operações geográficas em *Big Geospatial Data* quando realizadas em provedores públicos de nuvem.

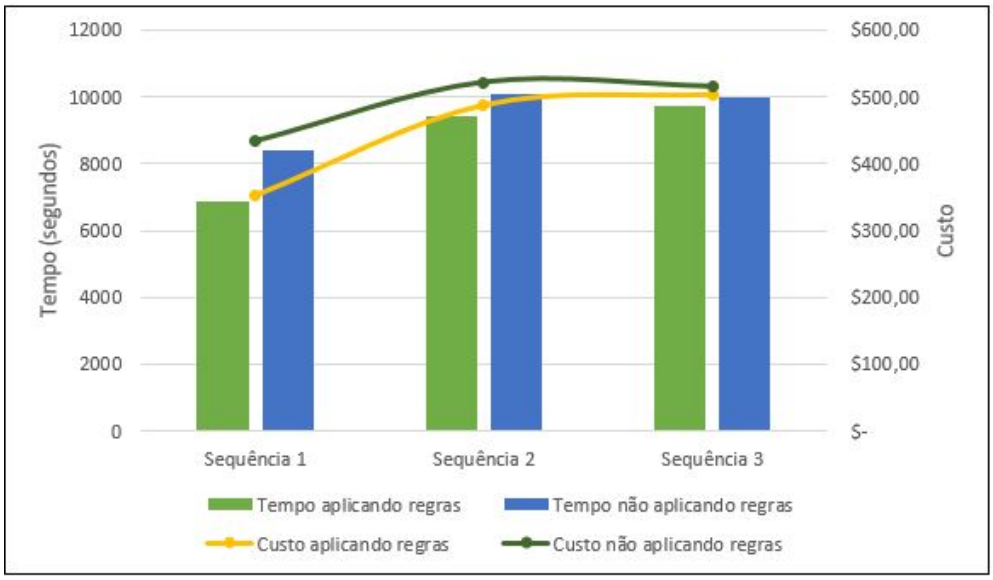

Figura 6.5: Comparação do Uso das Regras.

A utilização da indexação sugerida pelas regras do Mecanismo de Inferência atrelada a utilização do redimensionamento do *cluster*, quando possível, são capazes de gerar uma redução de até 39% em relação ao tempo, e 71% em relação ao custo total.

6.3 Resultados em Publicações

Durante o desenvolvimento deste trabalho foram publicados quatro artigos científicos produzidos pelo autor sobre o tema, os quais foram:

- O artigo com título *"A Cost-Efficient Method for Big Geospatial Data on Public Cloud Providers"*, foi publicado na 9ª Conferência Internacional sobre Sistemas de Informação Geográfica Avançada, Aplicações e Serviços (GeoProcessing 2017) [66];
- O artigo com título *"Cost Optimization on Public Cloud Provider for Big Geospatial Data: A Case Study Using Open Street Map"*, foi apresentado na 7ª Conferência Internacional em Computação em Nuvem (Closer 2017) [67];
- O artigo com o título "Comparação de Desempenho na Indexação de *Big Geospatial Data* em Ambiente de Nuvem Computacional", foi apresentado no *XVIII Brazilian Symposium on GeoInformatics - GeoInfo 2017* [68];
- O artigo com o título *"An Architecture for Cost Optimization in the Processing of Big Geospatial Data in Public Cloud Providers"*, será apresentado em Julho de 2018, no *7th IEEE International Congress on Big Data - BigData Congress 2018*.

Capítulo 7

Conclusões

Este trabalho apresentou uma Base de Conhecimento e um Mecanismo de Inferência para otimização no custo financeiro para o processamento de *big geospatial data* em provedores públicos de computação em nuvem.

A revisão da literatura mostrou que os trabalhos realizados até o presente momento não tiveram foco no critério custo/benefício das aplicações desenvolvidas para o processamento de *big geospatial data*, ficando restritos à comparação de indexações, à comparação entre os provedores ou à comparação no desempenho de aplicações, sempre de maneira isolada, diferenciando-se deste trabalho.

Conforme observado na literatura sobre o tema, a tarefa de indexação é fundamental para garantir um bom desempenho à aplicação, mas também é uma tarefa bastante onerosa e que exige alto poder computacional. Assim sendo, uma Base de Conhecimento e um Mecanismo de Inferência foram desenvolvidos considerando uma série de testes realizados, dos quais foi possível extrair informações que são capazes de indicar a indexação mais adequada de acordo com os parâmetros definidos pelo usuário, tais como a forma predominante (ponto, linha ou polígono) no conteúdo da base de dados a ser processada, e também a sequência de consultas e de operações a serem executadas.

Os testes realizados para a construção da Base de Conhecimento demonstraram que: o aumento de instâncias no *cluster* reduz o tempo de processamento, mas aumenta o custo total; instâncias com mais recursos computacionais reduzem o tempo de processamento, mas aumentam o custo total; a indexação *Grid* é mais performática para bases que tenham apenas pontos; a indexação *R-Tree* é mais performática para bases com linhas e polígonos; a indexação *R+-Tree* é melhor para *Union*, *Skyline* e *ConvexHull*; a indexação *Grid* é melhor para *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair*; e, o tempo para as operações *KNN*, *Range Query*, *Closest Pair* e *Farthest Pair* é pequeno após a indexação.

Outro parâmetro indicado pelas regras do Mecanismo de Inferência está relacionado ao redimensionamento da infraestrutura que suporta a aplicação, buscando reduzir o número de instâncias ativas para reduzir ainda mais o custo total de processamento.

Além disso, também foi apresentada uma comparação entre os principais provedores públicos de computação em nuvem que são Amazon AWS, Microsoft Azure e Google Cloud. Estes provedores disponibilizam serviços específicos para o processamento de *big data* que facilitam a configuração e o gerenciamento da infraestrutura. O provedor Google Cloud, por meio do serviço Dataproc, foi o que obteve a melhor relação custo/benefício.

Para validar a proposta foram criados cenários de testes que indicaram que as regras criadas no Mecanismo de Inferência são capazes de otimizar os custos financeiros totais para processamento de *big geospatial data* em até 71%.

Como trabalho futuro é sugerida a utilização de outras ferramentas atualmente disponíveis para o processamento de *big geospatial data*, além do *SpatialHadoop* que foi utilizado nesta pesquisa, tais como o *GeoSpark* [50] e o *LocationSpark* [51], bem como a avaliação em outros provedores ou até mesmo em regiões geograficamente diferentes de um mesmo provedor. Também é sugerida a análise de desempenho utilizando outras consultas e operações, tais como a junção espacial (*spatial join*). Outra melhoria pode ser feita com a análise pró-ativa do conjunto de dados, a fim de obter a informação sobre a forma predominante na base de dados. Além disso, podem ser utilizadas bases de dados que contenham os três tipos de dados vetorias (pontos, linhas e polígonos) em um único arquivo.

Referências

[1] Gilberto Câmara. Representação computacional de dados geográficos. *Banco de dados geográficos. Curitiba: Mundogeo*, pages 11–52, 2005. viii, 5, 6

[2] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *International Conference on Collaboration Technologies and Systems (CTS), 2013*, pages 42–47. IEEE, 2013. viii, 7

[3] Raghu Ramakrishnan and Johannes Gehrke. *Database management systems*. McGraw Hill, 2000. viii, 9, 10

[4] Clodoveu Davis Jr and Gilberto Queiroz. Métodos de acesso para dados espaciais. *MundoGEO Curitiba, 2005*, pages 213–231. viii, 8, 9, 10, 11, 12

[5] Volker Gaede and Oliver Günther. Multidimensional access methods. *ACM Computing Surveys (CSUR)*, 30(2):170–231, 1998. viii, 1, 9, 11, 12, 27, 30

[6] Chaowei Yang, Manzhu Yu, Fei Hu, Yongyao Jiang, and Yun Li. Utilizing cloud computing to address big geospatial data challenges. *Computers, Environment and Urban Systems*, 61:120–128, 2017. viii, 1, 18, 29, 30

[7] Rajeesh Menoth. Comparativo geográfico entre Azure, AWS e Google Cloud, 2017. Disponível em: https://social.technet.microsoft.com/wiki/pt-br/contents/articles/37376.comparativo-geografico-entre-azure-aws-e-google-cloud.aspx. Acessado em: 12-02-2018. viii, 19, 20

[8] Ahmed Eldawy and Mohamed F Mokbel. Spatialhadoop: A mapreduce framework for spatial data. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, pages 1352–1363. IEEE, 2015. viii, 1, 13, 22, 23, 24

[9] Eric van Rees. Open geospatial consortium (OGC). *GeoInformatics Journal*, 16(8):28, 2013. 1, 23

[10] Ahmed Eldawy and Mohamed F Mokbel. The era of big spatial data. In *31st IEEE International Conference on Data Engineering Workshops (ICDEW), 2015*, pages 42–49. IEEE, 2015. 1, 7

[11] Tom White. *Hadoop: The definitive guide*. "O'Reilly Media, Inc.", 2012. 1

[12] Ricardo Rodrigues Ciferri and Ana Carolina Salgado. Análise da eficiência de métodos de acesso espaciais em termos da distribuição espacial dos dados. In *Workshop Brasileiro de Geoinformática*, volume 3, pages 95–101, 2001. 1, 8, 9

[13] Blesson Varghese and Rajkumar Buyya. Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems*, 79:849–861, 2018. 1

[14] Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, and Ion Stoica. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28(13):2009, 2009. 2, 15, 34

[15] Marco AS Netto, Rodrigo N Calheiros, Eduardo R Rodrigues, Renato LF Cunha, and Rajkumar Buyya. Hpc cloud for scientific and business applications: taxonomy, vision, and research challenges. *ACM Computing Surveys (CSUR)*, 51(1):8, 2018. 2

[16] Brasil, Brasília. *Decreto Nº 6.666*, 27 de Novembro de 2008. Presidência da República. 5

[17] Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile networks and applications Journal*, 19(2):171–209, 2014. 6

[18] Hadoop Online Tutorial. Formula to Calculate HDFS Nodes Storage, 2017. Available at http://hadooptutorial.info/formula-to-calculate-hdfs-nodes-storage. 6

[19] John Gantz and David Reinsel. Extracting value from chaos. *IDC iview*, 1142(2011):1–12, 2011. 6

[20] NIST Big Data Public Working Group et al. Nist big data interoperability framework. *Special Publication*, pages 1500–6, 2015. 6

[21] Gartner Inc. What is big data?, Glossary IT, 2013. Disponível em https://www.gartner.com/it-glossary/big-data. Acessado em: 13-08-2017. 7

[22] Bill Gerhardt, Kate Griffin, and Roland Klemann. Unlocking value in the fragmented world of big data analytics. *Cisco Internet Business Solutions Group*, 2012. 7

[23] Sachchidanand Singh and Nirmala Singh. Big data analytics. *2012 International Conference on Communication Information & Computing Technology (ICCICT)*, 1, 2012. 7

[24] Navroop Kaur and Sandeep K Sood. Efficient resource management system based on 4vs of big data streams. *Big Data Research*, 2017. 7

[25] JS Ward and A Barker. Undefined by data: a survey of big data definitions. Ithaca: Cornell university library; 2013. 7

[26] Sampada Lovalekar. Big data: an emerging trend in future. *IJCSIT International Journal of Computer Science and Information Technologies*, 5(1), 2014. 7

[27] Jae-Gil Lee and Minseo Kang. Geospatial big data: challenges and opportunities. *Big Data Research*, 2(2):74–81, 2015. 7

[28] Simin You, Jianting Zhang, and Le Gruenwald. Large-scale spatial join query processing in cloud. In *Data Engineering Workshops (ICDEW), 2015 31st IEEE International Conference on*, pages 34–41. IEEE, 2015. 8

[29] Michel Krämer and Ivo Senner. A modular software architecture for processing of big geospatial data in the cloud. *Computers & Graphics*, 49:69–81, 2015. 8, 29, 30

[30] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975. 9

[31] Jürg Nievergelt, Hans Hinterberger, and Kenneth C Sevcik. The grid file: An adaptable, symmetric multikey file structure. *ACM Transactions on Database Systems (TODS)*, 9(1):38–71, 1984. 9

[32] Antonin Guttman. *R-trees: A dynamic index structure for spatial searching*, volume 14. Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, 1984. 9, 11, 13

[33] Christos Faloutsos and Ibrahim Kamel. Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension. In *Proceedings of the thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 4–13. ACM, 1994. 9

[34] T Sellis and N Roussopoulos. C. faloutsos; the R+–tree: A dynamic index for multi-dimensional objects. In *Proceedings of 13th Very Large Data Bases Conference (VLDB), Brighton, England*, pages 507–518, 1987. 9, 13

[35] Gilberto Câmara and J. S. de Medeiros. Operações de análise geográfica. *Sistema de informações geográficas. Brasília: Embrapa*, pages 67–91, 2003. 13

[36] Ahmed Eldawy, Yuan Li, Mohamed F Mokbel, and Ravi Janardan. Cg_hadoop: computational geometry in mapreduce. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 294–303. ACM, 2013. 13, 14, 24, 33

[37] Peter Mell and Tim Grance. The NIST definition of cloud computing. *Communications of the ACM*, 53(6):50, 2010. 15, 16, 17

[38] Luis M. Vaquero, Luis Rodero-Merino, and Rajkumar Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41(1):45–52, 2011. 15

[39] Nikolas Roman Herbst, Samuel Kounev, and Ralf H Reussner. Elasticity in cloud computing: What it is, and what it is not. In *Proceedings of 10th International Conference on Autonomic Computing*, pages 23–27, 2013. 16

[40] Kai Hwang, Xiaoying Bai, Yue Shi, Muyang Li, Wen-Guang Chen, and Yongwei Wu. Cloud performance modeling with benchmark evaluation of elastic scaling strategies. *IEEE Transactions on Parallel and Distributed Systems*, 27(1):130–143, 2016. 16

[41] Kai Hwang, Yue Shi, and Xiaoying Bai. Scale-out vs. scale-up techniques for cloud performance and productivity. In *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, pages 763–768. IEEE, 2014. 16

[42] Huanhuan Xiong, Frank Fowley, Claus Pahl, and Niall Moran. Scalable architectures for platform-as-a-service clouds: Performance and cost analysis. In *European Conference on Software Architecture*, pages 226–233. Springer, 2014. 16

[43] Emanuel Ferreira Coutinho, Flávio Rubens de Carvalho Sousa, Paulo Antonio Leal Rego, Danielo Gonçalves Gomes, and José Neuman de Souza. Elasticity in cloud computing: a survey. *annals of telecommunications-annales des télécommunications*, 70(7-8):289–309, 2015. 16

[44] Antonio Celesti, Francesco Tusa, Massimo Villari, and Antonio Puliafito. How to enhance cloud architectures to enable cross-federation. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 337–345. IEEE, 2010. 17

[45] Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. Cloudcmp: comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 1–14. ACM, 2010. 18, 27, 30

[46] CN Höfer and Georgios Karagiannis. Cloud computing services: taxonomy and comparison. *Journal of Internet Services and Applications*, 2(2):81–94, 2011. 19

[47] Zheng Li, He Zhang, Liam O'Brien, Shu Jiang, You Zhou, Maria Kihl, and Rajiv Ranjan. Spot pricing in the cloud ecosystem: A comparative investigation. *Journal of Systems and Software*, 114:1–19, 2016. 19

[48] John McArthur Raj Bala, Arun Chandrasekaran. Magic Quadrant for Public Cloud Storage Services, Worldwide, 2017. Disponível em: https://www.gartner.com/doc/reprints?id=1-2IH2LGIct=150626st=sb. Acessado em: 24-07-2017. 19

[49] Ahmed Eldawy. Spatialhadoop: towards flexible and scalable spatial processing using mapreduce. In *Proceedings of the 2014 SIGMOD PhD symposium*, pages 46–50. ACM, 2014. 22, 25

[50] Jia Yu, Jinxuan Wu, and Mohamed Sarwat. Geospark: A cluster computing framework for processing large-scale spatial data. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 70. ACM, 2015. 22, 49

[51] Mingjie Tang, Yongyang Yu, Qutaibah M Malluhi, Mourad Ouzzani, and Walid G Aref. Locationspark: a distributed in-memory data management system for big spatial data. *Proceedings of the Very Large Data Bases Conference (VLDB) Endowment*, 9(13):1565–1568, 2016. 22, 49

[52] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. Pig latin: a not-so-foreign language for data processing. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1099–1110. ACM, 2008. 23

[53] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. Hive: a warehousing solution over a map-reduce framework. *Proceedings of the Very Large Data Bases Conference (VLDB) Endowment*, 2(2):1626–1629, 2009. 23

[54] Ahmed Eldawy and Mohamed F Mokbel. Pigeon: A spatial mapreduce language. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, pages 1242–1245. IEEE, 2014. 23

[55] Ahmed Eldawy, Louai Alarabi, and Mohamed F Mokbel. Spatial partitioning techniques in Spatialhadoop. *Proceedings of the Very Large Data Bases Conference (VLDB) Endowment*, 8(12):1602–1605, 2015. 24, 27, 30

[56] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The Hadoop distributed file system. In *Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on*, pages 1–10. IEEE, 2010. 24

[57] TR Smith and P Gao. Experimental performance evaluations on spatial access methods. In *Proc. 4th Intl. Symposium on Spatial Data Handling, Zürich*, pages 991–1002, 1990. 26, 30

[58] B Ooi, Ron Sacks-Davis, and Jiawei Han. Indexing in spatial databases. *Technical Papers.*, 1993. Disponível em: https://docs.microsoft.com/en-in/azure/hdinsight/hdinsight-component-versioning. Acessado em: 20-01-2018. 26, 30

[59] Diane Greene. An implementation and performance analysis of spatial data access methods. In *Proceedings. Fifth International Conference on Data Engineering, 1989.*, pages 606–615. IEEE, 1989. 27

[60] Frederico Augusto Bedê Teotônio. Comparação do desempenho dos índices R-tree, grades fixas, e curvas de hilbert para consultas espaciais em bancos de dados geograficos. *Instituto Nacional de Pesquisas Espaciais-INPE, SP, Brazil*, 2008. 27, 30

[61] Changqing Ji, Yu Li, Wenming Qiu, Uchechukwu Awada, and Keqiu Li. Big data processing in cloud computing environments. In *12th International Symposium on Pervasive Systems, Algorithms and Networks (ISPAN), 2012*, pages 17–23. IEEE, 2012. 28, 30

[62] Karthik Kambatla, Abhinav Pathak, and Himabindu Pucha. Towards optimizing Hadoop provisioning in the cloud. *HotCloud*, 9:12, 2009. 28, 30

[63] Yali Zhao, Rodrigo N Calheiros, James Bailey, and Richard Sinnott. SLA-based profit optimization for resource management of big data analytics-as-a-service platforms in cloud computing environments. In *IEEE International Conference on Big Data (Big Data), 2016*, pages 432–441. IEEE, 2016. 28, 30

[64] Pramila Joshi. Cloud architecture for big data. *International Journal Of Engineering And Computer Science*, 4(06), 2015. 28, 30

[65] Dennis Merritt. *Building expert systems in Prolog.* Springer Science & Business Media, 2012. 31, 39

[66] Joao Bachiega, Marco Reis, Aleteia Araujo, and Maristela Holanda. A cost-efficient method for big geospatial data on public cloud providers. *The Ninth International Conference on Advanced Geographic Information Systems, Applications, and Services*, pages 25–31, 2017. 32, 47

[67] Joao Bachiega, Marco Reis, Aleteia Araujo, and Maristela Holanda. Cost optimization on public cloud provider for big geospatial data. *Proceedings of the 7th International Conference on Cloud Computing and Services Science*, pages 54–62, 2017. 32, 47

[68] Joao Bachiega, Marco Reis, Maristela Holanda, and Aleteia Araujo. Comparação de desempenho na indexação de big geospatial data em ambiente de nuvem computacional. *Proceedings XVIII Brazilian Symposium on GeoInformatics*, pages 134–139, 2017. 32, 47

[69] Microsoft Azure. Hadoop components and versions available with HDInsight, 2017. Disponível em: https://docs.microsoft.com/en-in/azure/hdinsight/hdinsight-component-versioning. Acessado em: 2017-11-12. 33