UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Analisando Sistemas Analíticos Espaciais Baseados em Hadoop e Spark: Uma Perspectiva de Usuário

João Pedro de Carvalho Castro

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

João Pedro de Carvalho Castro

Analisando Sistemas Analíticos Espaciais Baseados em Hadoop e Spark: Uma Perspectiva de Usuário

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Cristina Dutra de Aguiar Ciferri

USP – São Carlos
Dezembro de 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Castro, João Pedro de Carvalho
C355a Analisando Sistemas Analíticos Espaciais Baseados em Hadoop e Spark: Uma Perspectiva de Usuário / João Pedro de Carvalho Castro; orientadora Cristina Dutra de Aguiar Ciferri. -- São Carlos, 2019.
91 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2019.

1. Sistemas Analíticos Espaciais. 2. Hadoop. 3. Spark. 4. Comparação Centrada no Usuário. 5. Big Spatial Data. I. Ciferri, Cristina Dutra de Aguiar, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

João Pedro de Carvalho Castro

Analyzing Spatial Analytics Systems Based on Hadoop and Spark: A User Perspective

Dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP – in accordance with the requirements of the Computer and Mathematical Sciences Graduate Program, for the degree of Master in Science. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Cristina Dutra de Aguiar Ciferri

USP – São Carlos
December 2019

Este trabalho é dedicado aos pesquisadores do Brasil,
que continuam firmes neste cenário de desvalorização da ciência,
pois acreditam que ela é o caminho para a construção de um país melhor.

AGRADECIMENTOS

Os agradecimentos principais são direcionados aos meus pais, Vânia Aparecida de Sousa Borges e Giovanni de Castro Borges, que sempre trataram minha educação como prioridade. Sem o constante apoio e incentivo que vocês me deram eu com certeza não chegaria até aqui. Sou e serei eternamente grato por todos os sacrifícios, por toda a dedicação e por toda a paciência que vocês tiveram comigo. Eu tenho muito orgulho de ser filho de vocês e muita admiração pelos pais que tenho. Obrigado por tudo. Amo muito vocês!

Agradeço também à minha namorada, Sara Elizabeth da Silveira, que me acompanhou durante boa parte da minha trajetória no mestrado. Obrigado por acreditar no meu potencial, por sempre me ouvir e por me aguentar nos dias de estresse. Seu apoio foi fundamental para que eu chegasse até aqui. Você é uma pessoa especial e única no mundo, tenho muita sorte e orgulho de ser seu namorado. Te amo muito!

Agradeço à professora Cristina Dutra de Aguiar Ciferri, não apenas pela orientação de qualidade, mas também por acreditar que poderíamos construir um bom trabalho. Também manifesto minha gratidão ao professor Anderson Chaves Carniel, que me ajudou bastante na execução da pesquisa. Ademais, agradeço ao professor Sidgley Camargo de Andrade por fornecer um dos conjuntos de dados utilizados nos estudos de caso. Com certeza este trabalho não teria o mesmo nível de qualidade sem a ajuda de vocês. Muito obrigado!

Direciono os próximos agradecimentos aos meus amigos de Patos de Minas, meus amigos de Itajubá, meus primos, meus colegas das repúblicas Tibilisku e Aero, meus colegas de laboratório do Grupo de Bases de Dados e Imagens do ICMC-USP e meus colegas de trabalho do Centro de Computação da UFMG. Não vou citar o nome de todos para não correr o risco de esquecer de alguém, porém saibam que o apoio de vocês contribuiu consideravelmente para a conclusão deste trabalho.

Agradeço também aos demais professores e funcionários da USP pela educação de qualidade fornecida aos alunos da universidade. Por fim, o presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001. Portanto, manifesto minha gratidão à CAPES por prover o incentivo financeiro necessário para concluí-lo.

“Nós só podemos ver um pouco do futuro,
mas o suficiente para perceber que há muito a fazer.”
(Alan Turing)

RESUMO

CASTRO, J. P. C. **Analisando Sistemas Analíticos Espaciais Baseados em Hadoop e Spark: Uma Perspectiva de Usuário**. 2019. 91 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Sistemas Analíticos Espaciais (SAEs) representam uma nova tecnologia capaz de gerenciar um grande volume de dados espaciais por meio da utilização de *frameworks* de processamento paralelo e distribuído de dados, tais como o Hadoop e o Spark. Um número crescente de SAEs tem sido proposto na literatura, fato que evidencia a necessidade de se realizar análises comparativas entre esses sistemas. No entanto, as comparações disponíveis no estado da arte fornecem apenas uma visão centrada no desempenho dos SAEs. Ou seja, no melhor do conhecimento do autor do presente trabalho, não existem abordagens na literatura que realizem comparações entre SAEs com base em uma visão centrada no usuário, ou seja, comparações que visam ajudar os usuários a entender como as características dos SAEs são úteis para atender aos requisitos específicos de suas aplicações espaciais. No presente trabalho, essa lacuna na literatura é preenchida. Uma comparação dos seguintes SAEs baseados em Hadoop e Spark é fornecida, utilizando como base a perspectiva de seus usuários: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, GeoMesa Spark, SIMBA, LocationSpark, STARK, Magellan, SparkGIS e Elcano. Essa comparação é realizada de acordo com um amplo conjunto de critérios relacionados às características gerais desses sistemas, aos aspectos de manipulação de dados espaciais e aos aspectos inerentes ao ambiente distribuído. Com base nessa comparação, um conjunto de diretrizes é introduzido a fim de ajudar os usuários no processo de escolha de um SAE apropriado. Dois estudos de caso baseados em aplicações do mundo real também são descritos para ilustrar a aplicabilidade dessas diretrizes. Por fim, também são realizadas discussões sobre tendências cronológicas relacionadas aos SAEs e sobre as limitações que esses sistemas devem suprir a fim de aprimorar a experiência do usuário.

Palavras-chave: Sistemas Analíticos Espaciais, Hadoop, Spark, Comparação Centrada no Usuário, Big Spatial Data.

ABSTRACT

CASTRO, J. P. C. **Analyzing Spatial Analytics Systems Based on Hadoop and Spark: A User Perspective**. 2019. 91 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Spatial Analytics Systems (SAEs) represent a new technology capable of managing a huge volume of spatial data by using distributed data processing frameworks such as Hadoop and Spark. An increasing number of SAEs have been proposed in the literature, requiring a comparison among them. However, comparisons available in the literature only provide a system-centric view of SAEs, which is essentially based on performance evaluations. Thus, there is a lack of comparisons based on the user-centric view, i.e., comparisons that help users to understand how the characteristics of SAEs are useful to meet the specific requirements of their spatial applications. In this work, we fill this gap in the literature. We provide a user-centric comparison of the following SAEs based on Hadoop and Spark: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, GeoMesa Spark, SIMBA, LocationSpark, STARK, Magellan, SparkGIS, and Elcano. This comparison is performed by using an extensive set of criteria related to the general characteristics of these systems, to the aspects of spatial data handling, and to the aspects inherent to the distributed environment. Based on this comparison, we introduce a set of guidelines in order to help users to choose an appropriate SAE. We also describe two case studies based on real-world applications in order to illustrate the use of these guidelines. Finally, we also discuss chronological tendencies related to SAEs and limitations that SAEs should address to improve user experience.

Keywords: Spatial Analytics Systems, Hadoop, Spark, User-Centric Comparison, Big Spatial Data.

LISTA DE ABREVIATURAS E SIGLAS

CDRC	*Customer Data Research Centre*
CSV	*Comma Separated Values*
ESP-VT	*Ecosystem Services Partnership Visualization Tool*
GDELT	*Global Data of Events, Language and Tone*
GEOINFO	*Brazilian Symposium on Geoinformatics*
GeoJSON	*Geographic JavaScript Object Notation*
GISQF	*Geographic Information System Querying Framework*
GPU	*Graphics Processing Unit*
HDF	*Hierarchical Data Format*
HDFS	*Hadoop Distributed File System*
JTS	Java *Topology Suite*
KNN	K-*Nearest Neighbours*
MBR	*Minimum Bounding Rectangle*
NetCDF	*Network Common Data Form*
OGC	*Open Geospatial Consortium*
OSM-XML	*Open Street Maps Extensible Markup Language*
PAIRS	*Physical Analytics Integrated Repository and Services*
RDD	*Dataset* Distribuído e Resiliente
SAE	Sistema Analítico Espacial
SQL	*Structured Query Language*
TSV	*Tab Separated Values*
WKB	*Well-Known Binary*
WKT	*Well-Known Text*

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

1.1 Considerações Iniciais

Neste capítulo é descrita a introdução do presente trabalho, o qual possui como objetivo o desenvolvimento de uma análise comparativa que englobe os diferentes sistemas analíticos espaciais existentes na literatura sob a perspectiva de seus usuários. Na seção 1.2 é feita a contextualização do trabalho. Na seção 1.3 é destacada a lacuna existente na literatura que motiva a execução deste trabalho. Na seção 1.4 são detalhados seus objetivos, assim como a hipótese na qual os mesmos se baseiam. Por fim, na seção 1.5 é descrita a estrutura do presente trabalho.

1.2 Contextualização

A análise de dados espaciais é uma questão primordial para empresas que utilizam localizações geográficas para tomar decisões estratégicas e aprimorar a experiência do usuário. Essas empresas possuem uma enorme vantagem sobre suas concorrentes, sendo capazes de reagir rapidamente a possíveis mudanças em seu ambiente de negócio. Atualmente, o volume de dados espaciais está em constante crescimento, principalmente devido à grande variedade de aplicações que coletam esses dados, tais como aplicativos móveis e sistemas relacionados à internet das coisas (BAIG *et al.*, 2017). Portanto, uma demanda por novas tecnologias capazes de gerenciar grandes volumes de dados espaciais se torna evidente.

Sistemas Analíticos Espaciais (SAEs) surgiram como uma possível solução para essa demanda. Esses sistemas fornecem funcionalidades especializadas para processar e indexar grandes volumes de dados espaciais por meio da utilização de *frameworks* de processamento paralelo e distribuído de dados (GARCÍA-GARCÍA *et al.*, 2017). Na literatura, os *frameworks* Hadoop (APACHE, 2019a) e Spark (APACHE, 2019d) se destacam, uma vez que a maioria dos SAEs existentes são baseados nos mesmos (PANDEY *et al.*, 2018). Para processar dados de forma eficiente em ambientes de computação paralela e distribuída (ou seja, *clusters* de computadores

ou ambientes de computação em nuvem), o Hadoop utiliza um modelo de programação genérico composto por operações *map* e *reduce*, denominado MapReduce (DEAN; GHEMAWAT, 2008). Já o *framework* Spark disponibiliza uma série de operações para permitir o processamento de dados na memória principal desses ambientes, utilizando os conceitos de *datasets* distribuídos e resilientes (ZAHARIA *et al.*, 2010) e de *DataFrames* (ARMBRUST *et al.*, 2015). SAEs são desenvolvidos como extensões desses *frameworks*, fato que os permite herdar suas características e vantagens a fim de fornecer funcionalidades estendidas para lidar com dados espaciais.

1.3 Motivação

Diversos SAEs baseados em Hadoop e Spark foram propostos na literatura. Dentre os sistemas baseados em Hadoop, é possível destacar: Hadoop-GIS (AJI *et al.*, 2013; HADOOP-GIS, 2019) e SpatialHadoop (ELDAWY; MOKBEL, 2013; ELDAWY; MOKBEL, 2015; ELDAWY; ALARABI; MOKBEL, 2015; ELDAWY; MOKBEL; JONATHAN, 2016; SPATIALHADOOP, 2019). Já os sistemas baseados em Spark pertencentes ao estado da arte são: SpatialSpark (YOU; ZHANG; GRUENWALD, 2015; SPATIALSPARK, 2019), GeoSpark (YU; WU; SARWAT, 2015; YU; ZHANG; SARWAT, 2018; YU; ZHANG; SARWAT, 2019; GEOSPARK, 2019), GeoMesa Spark (HUGHES *et al.*, 2015; GEOMESA, 2019), Simba (XIE *et al.*, 2016; SIMBA, 2019), LocationSpark (TANG *et al.*, 2016; LOCATIONSPARK, 2019), STARK (HAGEDORN; GÖTZE; SATTLER, 2017b; STARK, 2019), Magellan (MAGELLAN, 2019), SparkGIS (BAIG *et al.*, 2017) e Elcano (ENGÉLINUS; BADARD, 2018).

Cada um dos SAEs supracitados possui características intrínsecas e funcionalidades distintas para lidar com dados espaciais vetoriais em ambientes computacionais paralelos e distribuídos. Além de suas características gerais, esses sistemas também integram duas perspectivas distintas: (i) características inerentes à manipulação de dados espaciais (GÜTING, 1994; OGC, 2019); e (ii) aspectos inerentes ao ambiente distribuído (PANDEY *et al.*, 2018). Sendo assim, os usuários que projetam, desenvolvem e implementam aplicações espaciais para organizações enfrentam o desafio de escolher um sistema em detrimento de outros.

De fato, os argumentos por trás da escolha de um determinado SAE dependem do objetivo da aplicação a ser desenvolvida. Existem aplicações espaciais cujo objetivo principal reside no processamento de consultas espaciais *ad-hoc*, como a descrita em Wiemann, Karrasch e Bernard (2018). Para essas aplicações, o sistema escolhido deve fornecer suporte para o processamento de diferentes tipos de operações espaciais, como relacionamentos topológicos (por exemplo, *contains*, *inside* e *meet*), operações geométricas de conjunto (por exemplo, união e interseção) e operações numéricas (por exemplo, cálculos de área e distância). Outras aplicações espaciais requerem interoperabilidade entre diferentes sistemas, como a descrita em Lee e Reichardt (2005); assim, a existência de diferentes representações (por exemplo, textual e binária) para objetos espaciais (por exemplo, pontos, linhas e polígonos) é um requisito. Além disso, existem

aplicações que requerem respostas rápidas para consultas espaciais específicas (PANDEY *et al.*, 2018). Nesse caso, o sistema escolhido deve fornecer, por exemplo, índices projetados especificamente para responder a essas consultas com eficiência.

Portanto, considerando a ampla variedade de aplicações espaciais e o fato de que cada SAE possui características e funcionalidades distintas, a necessidade de compará-los sob a perspectiva de seus usuários se torna evidente. Afinal, é necessário que o usuário entenda as características de cada SAE para ser capaz de selecionar os sistemas que atendam aos requisitos específicos de suas aplicações espaciais. Porém, a revisão da literatura realizada no presente trabalho evidencia que os estudos existentes no estado da arte se concentram apenas na comparação experimental dos SAEs, adotando uma visão centrada em desempenho (Capítulo 3). Ou seja, no melhor do conhecimento do autor do presente trabalho, não existem abordagens na literatura que apresentem análises comparativas centradas no usuário para esses sistemas. Essa lacuna identificada na literatura caracteriza a motivação deste trabalho de mestrado.

1.4 Objetivo e Contribuições

O presente trabalho de mestrado possui como objetivo realizar uma análise comparativa qualitativa de SAEs baseados em Hadoop e Spark sob a perspectiva de seus usuários. São analisados os seguintes sistemas: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, GeoMesaSpark, Simba, LocationSpark, STARK, Magellan, SparkGIS e Elcano. Com base nesse objetivo, define-se a seguinte hipótese:

Hipótese. *É possível assistir o usuário, por meio de uma análise comparativa das principais características dos SAEs, no processo de escolha dos sistemas que atendam aos requisitos específicos de aplicações espaciais distintas.*

Como consequência da investigação da hipótese supradescrita, o presente trabalho apresenta as seguintes contribuições:

- Comparação de SAEs baseados em Hadoop e Spark utilizando um conjunto extenso de critérios relacionados às suas características gerais, às características referentes à manipulação de dados espaciais e aos aspectos inerentes ao ambiente distribuído.
- Proposta de um conjunto de diretrizes centradas no usuário, baseadas em tal comparação, a fim de ajudá-lo a identificar os SAEs que atendem aos requisitos específicos de suas aplicações espaciais.
- Descrição de dois estudos de caso, um que utiliza o GeoSpark e outro que adota o SpatialHadoop, a fim de ilustrar a empregabilidade das diretrizes propostas.

- Discussão das tendências cronológicas relacionadas aos SAEs e identificação de limitações que esses sistemas devem tratar a fim de aprimorar a experiência do usuário.

Ademais, a produção dos seguintes artigos científicos também pode ser considerada como uma contribuição oriunda do presente trabalho de mestrado:

- Artigo intitulado "***A User-Centric View of Distributed Spatial Data Management Systems***" (CASTRO; CARNIEL; CIFERRI, 2018), publicado no XIX *Brazilian Symposium on Geoinformatics* (GEOINFO), 2018. O conteúdo desse artigo refere-se a uma versão inicial do presente trabalho.
- Artigo intitulado "***Analyzing Spatial Analytics Systems Based on Hadoop and Spark: A User Perspective***", submetido para a revista científica *Computers & Geosciences*. O conteúdo desse artigo representa uma síntese do presente trabalho.

1.5 Estrutura do Trabalho

Além do presente capítulo introdutório, este trabalho possui mais seis capítulos, estruturados da seguinte forma:

- No Capítulo 2 são descritos os conceitos necessários para a compreensão do trabalho. Esses conceitos englobam as perspectivas de manipulação de dados espaciais e de ambientes de computação paralela e distribuída.
- No Capítulo 3 é realizada uma revisão da literatura. São descritas abordagens correlatas ao presente trabalho, a fim de contextualizar seu respectivo tema de pesquisa.
- No Capítulo 4 é introduzida a comparação de SAEs baseados em Hadoop e Spark sob a perspectiva de seus usuários. Cada característica abordada é discutida no decorrer do capítulo, sendo também detalhados os critérios empregados para a escolha dos SAEs e as fontes de dados utilizadas para a extração de suas características.
- No Capítulo 5 são apresentadas diretrizes centradas no usuário, as quais visam enriquecer o auxílio provido ao mesmo durante o processo de seleção de um SAE.
- No Capítulo 6 são descritos dois estudos de caso a fim de ilustrar a aplicabilidade das diretrizes propostas.
- No Capítulo 7 são descritas as conclusões do trabalho, incluindo discussões acerca das tendências cronológicas dos SAEs e suas limitações, assim como as contribuições do presente trabalho e os possíveis trabalhos futuros.

CAPÍTULO

2

FUNDAMENTAÇÃO TEÓRICA

2.1 Considerações Iniciais

Neste capítulo são descritos os conceitos necessários para a compreensão do presente trabalho. Esses conceitos estão divididos em duas perspectivas: (i) manipulação de dados espaciais, descrita na seção 2.2; e (ii) ambientes de computação paralela e distribuída, descrita na seção 2.3. Por fim, na seção 2.4 são feitas as considerações finais do capítulo.

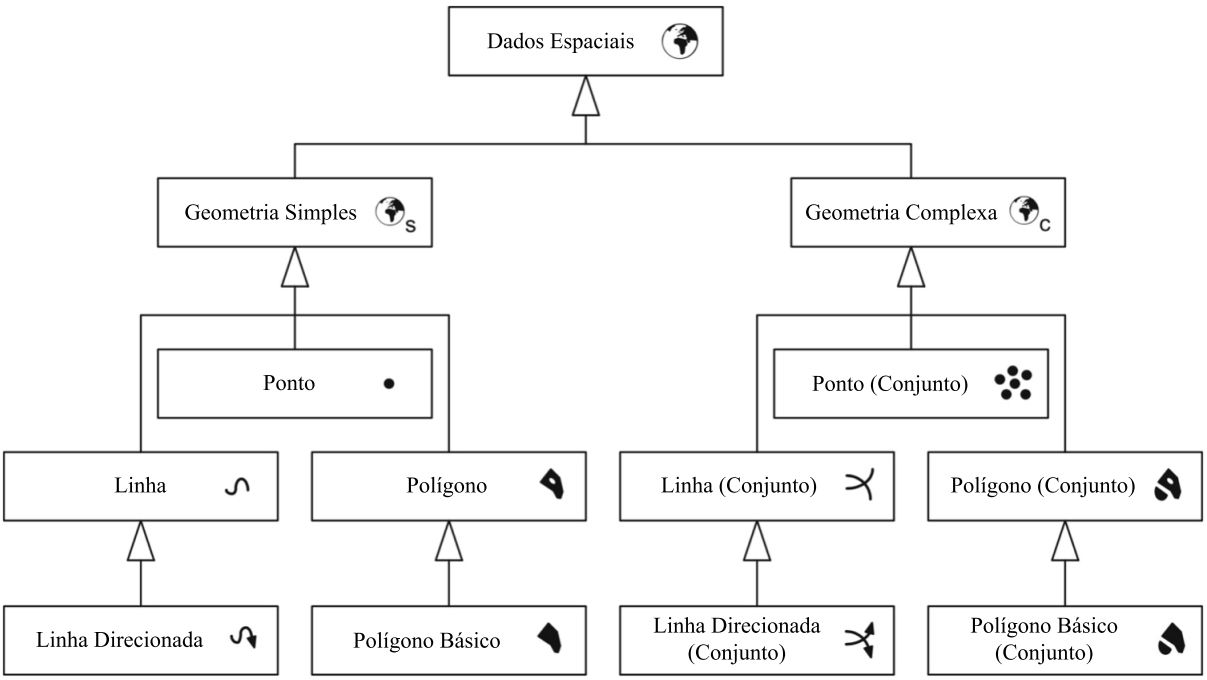
2.2 Manipulação de Dados Espaciais

Dados espaciais, também tratados na literatura como dados geográficos, dados georreferenciados ou dados geoespaciais, são componentes que representam a geometria de objetos espaciais (GÜTING, 1994). Existem diversos tipos de dados espaciais descritos na literatura que podem ser utilizados para este tipo de representação, a depender das características do objeto espacial a ser representado. Uma cidade, por exemplo, pode ser representada tanto por um dado espacial do tipo ponto quanto por um dado espacial do tipo polígono.

Os dados espaciais podem ser categorizados como sendo de geometria simples ou complexa (GÜTING, 1994; VAISMAN; ZIMÁNYI, 2014). Dados espaciais de geometria simples incluem: (i) pontos, representando geometrias zero-dimensionais que denotam uma localização única no espaço; (ii) linhas, representando geometrias unidimensionais que denotam uma curva contínua em um plano, composta por um conjunto de pontos interligados; e (iii) polígonos (ou regiões), representando geometrias bidimensionais, que denotam um conjunto de pontos localizados dentro de uma fronteira formada por linhas. Já a categoria de dados espaciais de geometria complexa existe para representar dados espaciais de geometria simples na forma de conjuntos. Sendo assim, é possível afirmar que um dado espacial de geometria complexa é composto por um ou mais conjuntos homogêneos de pontos, linhas ou polígonos.

A Figura 1 ilustra essa categorização. Nessa, é possível observar que podem existir subtipos para as categorias supracitadas: linhas podem ter uma orientação, se tornando linhas direcionadas; e polígonos podem não ter espaços vazios em seu interior, se tornando polígonos básicos.

Figura 1 – Categorização dos tipos de dados espaciais. Fonte: Adaptada de Vaisman e Zimányi (2014).



Dados espaciais podem ser armazenados tanto de forma discreta, por meio do modelo vetorial, quanto de forma contínua, por meio do modelo *raster* (VAISMAN; ZIMÁNYI, 2014). Enquanto o modelo vetorial é utilizado para modelar conceitos humanos de geografia, tais como regiões ou construções, o modelo *raster* é utilizado para modelar fenômenos geográficos que estão em constante mudança no espaço, tais como altitude e temperatura. No modelo vetorial, um ponto é armazenado como um conjunto de coordenadas, tais como (x, y) ou (x, y, z), a depender da quantidade de dimensões existentes no espaço onde está localizado. Já os demais tipos de dados são representados por uma estrutura de dados (por exemplo, um vetor) que contém os pontos que os formam. No presente trabalho, o modelo vetorial é adotado para a representação de dados espaciais. Dessa forma, o modelo *raster* não é detalhado na presente seção, tampouco abordado na comparação centrada no usuário introduzida no Capítulo 4.

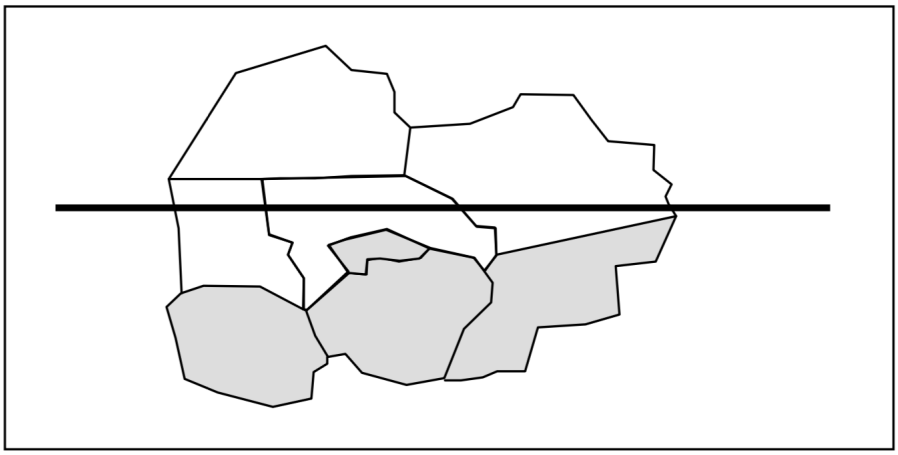
O restante desta seção está organizado da seguinte forma. Na seção 2.2.1 são detalhados os tipos de relacionamentos existentes entre dados espaciais. Em seguida, na seção 2.2.2 são descritos os diversos tipos de consultas espaciais que empregam esses relacionamentos. Por fim, na seção 2.2.3 é descrito como a indexação espacial pode ser utilizada para otimizar o processamento dessas consultas.

2.2.1 Relacionamentos Espaciais

Duas instâncias de dados espaciais podem se relacionar por meio de relacionamentos métricos, direcionais e topológicos. Relacionamentos métricos são baseados na distância entre esses dados no espaço onde eles estão localizados, sendo que a distância entre dados espaciais de geometria não-zero-dimensional é definida pela distância entre seus pontos mais próximos. A distância entre dois pontos no espaço é calculada por meio de funções de distância, tais como a função de distância Euclidiana e a função de distância Manhattan (GAEDE; GÜNTHER, 1998).

Relacionamentos direcionais, por sua vez, são definidos a partir da direção geográfica de um objeto espacial em relação a uma referência (PEUQUET; CI-XIANG, 1987). Essa direção geográfica pode assumir os seguintes valores (CIFERRI; SALGADO, 2002): (i) a norte de; (ii) a sul de; (iii) a leste de; (iv) a oeste de; (v) a nordeste de; (vi) a sudeste de; (vii) a noroeste de; e (viii) a sudoeste de. A Figura 2 ilustra um exemplo desse tipo de relacionamento, no qual são retornados todos os polígonos localizados completamente ao sul de uma linha.

Figura 2 – Exemplo de relacionamento direcional considerando a direção geográfica "a sul de". Fonte: Ciferri e Salgado (2002).



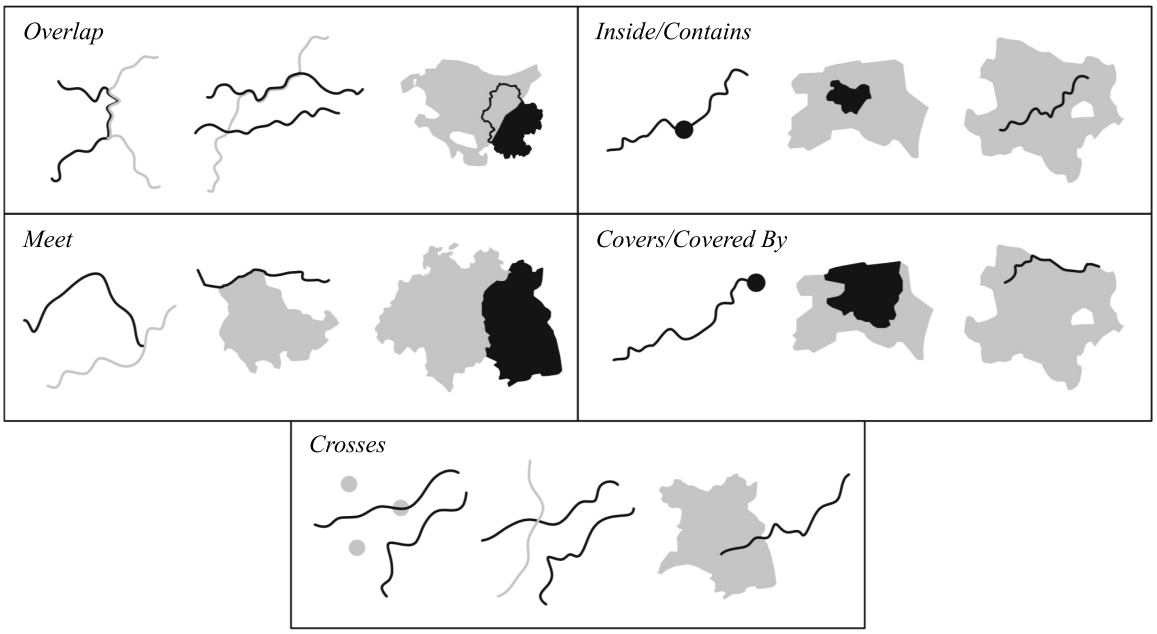
A definição de um relacionamento topológico é dependente dos conceitos de exterior, interior e fronteira de um dado espacial (VAISMAN; ZIMÁNYI, 2014). Enquanto o exterior deste dado é composto por todos os pontos do espaço que não pertencem ao dado, seu interior é composto por todos os pontos que pertencem ao dado, porém que não fazem parte de sua fronteira. O conceito de fronteira de um dado espacial é dependente do tipo que este assume. Enquanto pontos possuem uma fronteira vazia, linhas possuem sua fronteira definida pelos pontos localizados em suas extremidades. Já a fronteira de um polígono é determinada tanto pelas linhas que o cercam quanto pelas linhas que cercam os espaços vazios em seu interior.

Um relacionamento topológico ocorre quando o interior, o exterior e/ou a fronteira de duas instâncias de dados espaciais compartilham, mesmo que não totalmente, uma localização no espaço (OGC, 2019). O relacionamento *equal*, por exemplo, ocorre quando dois dados espaciais são definidos exatamente pelo mesmo conjunto de pontos. Dessa forma, quando dois dados espaciais satisfazem esse relacionamento, pode-se dizer que tais dados possuem os mesmos

interior, exterior e fronteira. Outros dois relacionamentos topológicos que merecem destaque são os relacionamentos *intersects* e *disjoint*. Esses relacionamentos são considerados inversos, de forma que a ocorrência de um impede a ocorrência do outro. O relacionamento *disjoint* é satisfeito quando um dado espacial compartilha somente a localização de seu exterior com outro. Em todos os outros casos – ou seja, quando um dado espacial compartilha a localização de seu interior e/ou de sua fronteira com outro – um relacionamento *intersects* é satisfeito.

Utilizando como base o conceito do relacionamento *intersects*, diversos outros relacionamentos topológicos podem ser definidos (VAISMAN; ZIMÁNYI, 2014). O relacionamento *overlap*, por exemplo, ocorre quando uma geometria intersecta tanto o interior quanto o exterior de outra geometria. Ademais, os relacionamentos *inside* e *contains* são satisfeitos quando um objeto espacial *a* intersecta somente o interior de um objeto espacial *b*. Dessa forma, o objeto *a* satisfaz o relacionamento *inside* em relação ao objeto *b*, enquanto o objeto *b* satisfaz o relacionamento *contains* em relação ao objeto *a*. Os relacionamentos *covered by* e *covers* funcionam de forma similar, porém com o objeto *a* também podendo intersectar a fronteira do objeto *b*. O relacionamento *meet*, por sua vez, ocorre quando as fronteiras de dois objetos espaciais se intersectam, mas seus interiores não. Por fim, o relacionamento *crosses* é atendido quando duas geometrias se intersectam e a dimensão desta intersecção é menor do que a maior dimensão dentre as geometrias envolvidas. Exemplos desses relacionamentos são ilustrados na Figura 3.

Figura 3 – Exemplos de diferentes relacionamentos topológicos. Os dois objetos envolvidos no relacionamento estão representados nas cores preta e cinza, respectivamente. Fonte: Adaptada de Vaisman e Zimányi (2014).



2.2.2 Consultas Espaciais

Consultas espaciais podem ser descritas como sendo aquelas nas quais ao menos um dentre seus predicados envolve um relacionamento espacial. De acordo o tipo de relacionamento espacial sendo verificado pelo predicado, uma consulta espacial pode ser classificada em diferentes tipos (GAEDE; GÜNTHER, 1998; CIFERRI; SALGADO, 2002; JACOX; SAMET, 2007):

- *Exact match query*, a qual visa retornar todos os dados espaciais que possuam a mesma geometria do dado espacial utilizado como critério de consulta.
- *Point query*, a qual possui o objetivo de retornar todos os dados espaciais que sobrepõem o ponto empregado no predicado da consulta.
- *Range query*, a qual retorna todos os dados espaciais que possuam pelo menos um ponto em comum com o polígono fornecido como janela de consulta no predicado, que deve ser obrigatoriamente convexo.
- *Region query*, também conhecida na literatura como *intersection query*, a qual possui as mesmas características da *range query*, porém não exige que o polígono utilizado como janela de consulta seja convexo.
- *Enclosure query*, a qual possui como objetivo o retorno de todos os dados espaciais que contenham a janela de consulta fornecida no predicado.
- *Containment query*, a qual visa realizar o oposto de uma *enclosure query*, retornando todos os dados espaciais que estão contidos na janela de consulta especificada.
- *Adjacency query*, a qual visa retornar todos os dados espaciais que satisfaçam o relacionamento topológico *meet* com o dado espacial especificado no predicado da consulta.
- *Nearest neighbours query*, a qual visa retornar todos os dados espaciais que se encontram dentro de uma distância determinada no predicado de consulta, tendo como referência um dado espacial também especificado nesse predicado.
- K-*nearest neighbours* (KNN) *query*, a qual visa retornar os k dados espaciais mais próximos de uma determinada referência, que também é um dado espacial especificado no predicado da consulta.
- *Spatial join query*, a qual se difere de uma consulta com junção convencional devido ao envolvimento de atributos espaciais em sua cláusula de junção. Essa cláusula pode ser definida por meio de um relacionamento topológico, métrico, ou direcional entre esses atributos. Dessa forma, a consulta retorna todos pares de registros de ambos os conjuntos envolvidos na junção cujos atributos espaciais satisfaçam tal relacionamento.

Além dessas consultas espaciais, três outras operações são frequentemente implementadas pelos SAEs revisados no presente trabalho (BÖHM; KREBS, 2004; ELDAWY *et al.*, 2013):

- KNN *join query*, a qual considera dois conjuntos de dados espaciais, retornando, para cada elemento do primeiro conjunto, seus k vizinhos mais próximos no segundo conjunto.
- *Convex hull query*, a qual retorna o menor polígono convexo capaz de conter todas as geometrias de um conjunto de dados espaciais.
- *Skyline query*, a qual visa retornar todos os pontos de referência que não são dominados por nenhum outro ponto de seu conjunto. Essa ocorrência de não dominação acontece quando nenhum outro ponto do conjunto possui ambas as coordenadas x e y maiores que as do ponto de referência ao mesmo tempo.

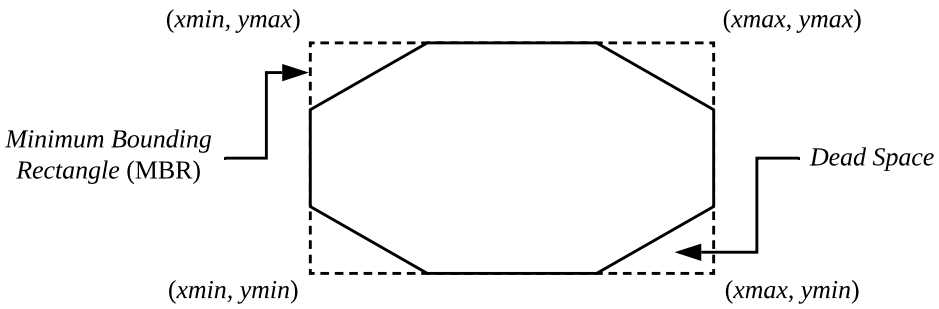
2.2.3 Indexação Espacial

Para que a execução eficiente de consultas espaciais seja possível, é necessário que o banco de dados espacial possua um mecanismo de indexação responsável por agilizar a recuperação de dados de acordo com suas características espaciais. Porém, mecanismos tradicionais de indexação não são adaptados para lidar com dados espaciais, visto que estes estão localizados em um espaço multidimensional e podem possuir geometrias não-zero-dimensionais (GUTTMAN, 1984). Nesse contexto, os índices espaciais – métodos de acesso multidimensionais que visam suprir tais necessidades – foram desenvolvidos (GAEDE; GÜNTHER, 1998). Com o intuito de ilustrar a maneira pela qual esses índices podem otimizar o processamento de consultas espaciais, as estruturas de indexação R-*Tree* e R*-*Tree* são detalhadas na presente seção. Essas estruturas foram escolhidas devido à sua ampla utilização em diversos trabalhos na literatura (CARNIEL; CIFERRI; CIFERRI, 2017a).

A R-*Tree*, proposta por Guttman (1984), é um método de acesso multidimensional que indexa dados espaciais com base no conceito de *Minimum Bounding Rectangles* (MBRs). O MBR de um dado espacial pode ser definido como sendo o menor retângulo possível capaz de conter sua geometria. Como este retângulo é definido por meio de quatro coordenadas (*xmin, ymin, xmax, ymax*), o espaço de armazenamento ocupado pelo mesmo é sempre menor ou igual àquele ocupado pela geometria do dado espacial sendo indexado. Porém, a utilização desta representação pode acarretar na existência de *dead space*, ou seja, uma área vazia entre a geometria do dado espacial e a fronteira de seu MBR, conforme ilustrado na Figura 4.

A estrutura da R-*Tree* é baseada na construção de uma hierarquia utilizando dois tipos de nó: (i) nós internos, que possuem entradas que armazenam tanto um MBR que engloba todos os MBR armazenados em seu nó filho quanto um ponteiro para este nó; e (ii) nós folha, que possuem entradas que armazenam tanto o MBR de um dado espacial quanto um ponteiro para este dado. A R-*Tree* é mantida balanceada, de forma que todos seus nós folha encontram-se no

Figura 4 – *Minimum Bounding Rectangle* (MBR). Fonte: Elaborada pelo autor.



mesmo nível. Cada nó possui limites mínimo e máximo de entradas, com exceção do nó raiz, que pode possuir uma quantidade de entradas menor do que o limite mínimo. Esses limites podem ser ajustados de acordo com o desempenho desejado, desde que o limite mínimo seja menor ou igual à metade do limite máximo (GUTTMAN, 1984).

Como a hierarquia da R-*Tree* é baseada somente nos MBR dos dados espaciais sendo indexados, a ocorrência de intersecções entre os MBR de um mesmo nível é comum. Nesse tipo de situação, é dito que os nós estão indexando a mesma porção do espaço. A ocorrência deste tipo de situação é minimizada graças ao algoritmo de inserção da R-*Tree*, responsável por alocar novas entradas em nós folha a fim de minimizar a abrangência das entradas existentes em nós internos (SIQUEIRA *et al.*, 2012).

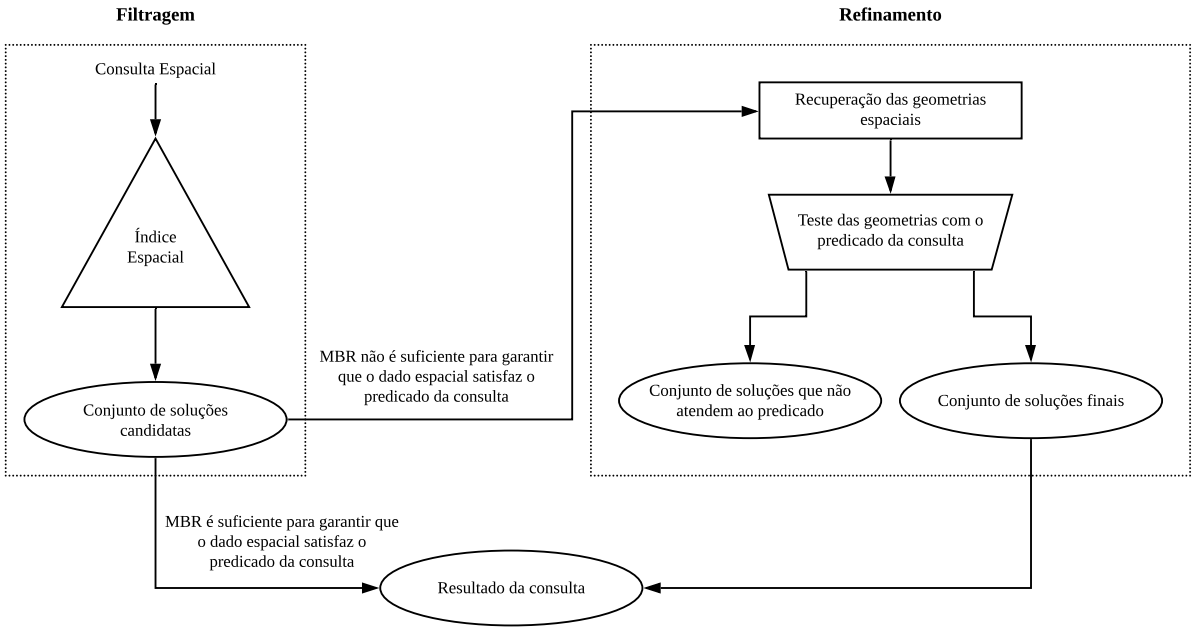
A R*-*Tree*, proposta por Beckmann *et al.* (1990), modifica a R-*Tree* no que diz respeito ao seu algoritmo de inserção. Esse algoritmo é aprimorado visando levar em consideração não apenas o critério de minimização da abrangência, como é feito na R-*Tree*, como também os critérios de: (i) sobreposição, que visa reduzir a área de intersecção entre os MBRs; (ii) margem, que é utilizado para minimizar o perímetro dos MBRs; e (iii) armazenamento, que é utilizado para maximizar a taxa de ocupação dos nós da árvore. A análise de todos estes critérios durante a inserção permite que a R*-*Tree* tenha maior probabilidade de possuir um melhor particionamento de espaço em relação à R-*Tree*, fato que acarreta em um melhor desempenho na execução de consultas (SIQUEIRA *et al.*, 2012).

Como tanto a R-*Tree* quanto a R*-*Tree* não armazenam a geometria real do dado espacial sendo indexado, esses índices são frequentemente incapazes de processar uma consulta espacial por si só, pois a janela de consulta fornecida no predicado espacial pode se relacionar somente com o *dead space* do MBR armazenado no índice (GAEDE; GÜNTHER, 1998). Logo, processar uma consulta utilizando indexação espacial envolve, frequentemente, a execução de duas fases, conforme ilustrado na Figura 5: (i) fase de filtragem; e (ii) fase de refinamento.

A fase de filtragem se refere à construção de um conjunto de soluções candidatas para a consulta, utilizando somente o conteúdo do índice – ou seja, os MBRs – na verificação do predicado espacial. Para cada solução candidata obtida nessa fase, é necessário determinar se o MBR é suficiente para garantir que o dado espacial satisfaz esse predicado. Em caso positivo, o

dado espacial é adicionado diretamente ao resultado da consulta. Porém, em caso negativo, a execução de uma fase de refinamento se torna necessária. Nessa etapa, a geometria real do dado espacial deve ser recuperada do banco de dados e testada com o predicado da consulta. Por fim, caso o predicado seja verdadeiro, é dito que o dado espacial faz parte do conjunto de soluções finais, sendo, portanto, adicionado ao resultado da consulta (GAEDE; GÜNTHER, 1998).

Figura 5 – Fases de filtragem e de refinamento. Fonte: Adaptada de Gaede e Günther (1998).



2.3 Ambientes de Computação Paralela e Distribuída

Recentemente, com a internet se tornando cada vez mais acessível e com a popularização crescente de tecnologias de armazenamento de baixo custo, um grande volume de dados, estruturados ou não, tem sido coletado (LI; MANOHARAN, 2013). Essa coleta traz consigo problemas relacionados tanto ao armazenamento desse grande volume de dados quanto à necessidade de gerenciamento de uma quantidade considerável de requisições de leitura e escrita sem que uma latência notável seja observada (HECHT; JABLONSKI, 2011). Na literatura, tanto ambientes de computação em nuvem quanto ambientes de *clusters* de computadores são amplamente utilizados para solucionar tais problemas. Esses ambientes de computação paralela e distribuída são capazes de prover um grande poder de armazenamento e de processamento, sendo, portanto, ideais para manipular um grande volume de dados (CHEN; MAO; LIU, 2014).

Porém, é notável a existência de uma certa complexidade relacionada à realização dessa manipulação em ambientes de computação paralela e distribuída. Esses ambientes exigem que diversos aspectos sejam considerados para que o processamento de dados ocorra de forma otimizada, o que muitas vezes reflete em tarefas não triviais. Nesse contexto, *frameworks* de processamento paralelo e distribuído de dados, tais como o Hadoop (APACHE, 2019a) e o Spark (APACHE, 2019d), surgem para simplificar a interação do usuário com tais ambientes.

Além disso, esses *frameworks* têm sido estendidos para que consigam executar consultas com predicados espaciais de forma eficiente. Essas extensões são conhecidas na literatura como sistemas analíticos espaciais (PANDEY *et al.*, 2018).

O restante desta Seção está organizado da seguinte forma. Na seção 2.3.1 são descritos conceitos relacionados aos ambientes de *clusters* de computadores e de computação em nuvem. Na seção 2.3.2 são detalhados os *frameworks* Hadoop e Spark. Por fim, na seção 2.3.3 são definidos conceitos acerca de sistemas analíticos espaciais.

2.3.1 Clusters de Computadores e Computação em Nuvem

O uso do modelo de *clusters* de computadores se popularizou em meados da década de 90, graças aos altos custos envolvidos na utilização de supercomputadores para solucionar tarefas computacionalmente intensivas. Permitindo a utilização de *hardware* básico e a adição ou remoção de componentes sempre que necessário, esse modelo é capaz de acompanhar evoluções tecnológicas sem gerar grandes impactos financeiros (HWANG; DONGARRA; FOX, 2013). Essa flexibilidade está intrinsicamente ligada à forma como os *clusters* de computadores são organizados. Esses ambientes são compostos por uma coleção de computadores dispostos de forma paralela e distribuída, porém interconectados por meio do uso de redes de alta velocidade. Essa conexão é realizada com o intuito de permitir que o *cluster* atenda a pelo menos uma dentre três necessidades: (i) computação intensiva; (ii) alta disponibilidade; e (iii) balanceamento de carga (SADASHIV; KUMAR, 2011).

Clusters que visam atender à necessidade de computação intensiva possuem como objetivo principal a utilização conjunta de seus nós com o intuito de realizar o processamento coletivo de uma única tarefa computacional complexa. Nesse tipo de *cluster*, os nós devem, idealmente: (i) ser majoritariamente homogêneos, ou seja, possuir a mesma arquitetura de processadores e o mesmo sistema operacional; (ii) compartilhar uma rede dedicada; e (iii) estar acoplados de forma próxima (HWANG; DONGARRA; FOX, 2013).

Ademais, *clusters* que visam atender à necessidade de alta disponibilidade possuem como objetivo principal a manutenção do funcionamento de suas aplicações no caso de eventuais falhas. É possível atingir esse feito graças à utilização de nós redundantes no *cluster*, fazendo com que caso o funcionamento de um destes seja comprometido, outro possa substituí-lo sem prejudicar a execução de suas aplicações (SADASHIV; KUMAR, 2011).

Por fim, *clusters* que visam atender à necessidade de balanceamento de carga possuem como objetivo principal dividir suas tarefas entre seus nós, almejando obter uma melhor utilização de seus recursos computacionais e um melhor desempenho na execução dessas tarefas. Em outras palavras, esse tipo de *cluster* se comporta como um único computador virtual, recebendo tarefas como uma única máquina e repartindo-as entre seus nós, que, por sua vez, agem como seus componentes (HWANG; DONGARRA; FOX, 2013).

Apesar da flexibilidade provida por ambientes de *clusters* de computadores ao solucionar diferentes necessidades, sua utilização pode exigir do usuário esforços consideráveis. De acordo com Sadashiv e Kumar (2011), aplicações desenvolvidas para esses ambientes precisam de incluir em seu código rotinas para lidar com a manipulação dos componentes do *cluster*, tais como a divisão de tarefas e o gerenciamento da comunicação entre seus nós. Dessa forma, a carência de um paradigma que permita uma interação simplificada entre o usuário e ambientes de computação paralela e distribuída se torna evidente. O conceito de computação em nuvem (ou *cloud computing*) surge para preencher essa lacuna, permitindo que o usuário terceirize esforços e custos envolvidos no gerenciamento de tal interação (ARMBRUST *et al.*, 2009).

A computação em nuvem pode ser definida como sendo um modelo para permitir o acesso por rede de forma conveniente, ubíqua e sob demanda a um conjunto configurável de recursos computacionais apresentados na forma de serviços. Nesse acesso, tanto o esforço de gerenciamento por parte do usuário quanto sua interação com o provedor de tais serviços são mínimos, já que estes são providos e liberados de forma rápida e automatizada (MELL; GRANCE, 2011). Além disso, graças à característica de mensuração do serviço presente nesse modelo, o usuário paga somente pelos recursos que esteja efetivamente utilizando, evitando cobranças por recursos ociosos. Isso ocorre devido ao fato de que o provedor de serviços utiliza métricas para medir o uso de tais recursos, realizando suas cobranças com base nestas medições (BHOWMIK, 2017).

2.3.2 Frameworks Hadoop e Spark

A manipulação de um grande volume de dados, tanto em ambientes de *clusters* de computadores quanto em ambientes de computação em nuvem, pode ser realizada por meio do auxílio de *frameworks* de processamento paralelo e distribuído de dados. A estratégia utilizada para a realização desse processamento é variável de acordo com o *framework* sendo adotado. No presente trabalho, tem-se como objetivo analisar as estratégias referentes aos *frameworks* Hadoop (APACHE, 2019a) e Spark (APACHE, 2019d), as quais são descritas a seguir.

O Hadoop é um *framework* de processamento paralelo e distribuído de dados que implementa o modelo de programação MapReduce (DEAN; GHEMAWAT, 2008). Dessa forma, seus usuários precisam se preocupar apenas com o desenvolvimento de funções *map* e *reduce*, uma vez que todos os detalhes relativos ao ambiente paralelo e distribuído são gerenciados pelo *framework*, tais como o uso eficiente da rede e dos discos. A função *map* é responsável por processar os dados de entrada, transformando-os em saídas intermediárias na forma de pares do tipo chave-valor. Posteriormente, a função *reduce* processa as saídas intermediárias, integrando todos os valores associados a uma mesma chave para que eles formem um único resultado (DERBEKO *et al.*, 2016).

As funções *map* e *reduce* são processadas no ambiente paralelo e distribuído por meio de um processo mestre e de vários processos executores, cujo funcionamento ocorre em diferentes

nós. Enquanto o processo mestre é responsável por criar e designar tarefas de execução das funções *map* e *reduce* para processos executores que estejam ociosos, os processos executores são responsáveis por executar essas tarefas em paralelo (DERBEKO *et al.*, 2016).

Apesar da ampla utilização do *framework* Hadoop na literatura, este possui algumas limitações. Essas se tornam evidentes principalmente em aplicações que processam dados de forma iterativa, ou seja, aplicações que requerem a realização de várias operações no mesmo conjunto de dados. Cada operação executada no *framework* Hadoop exige a leitura de dados de entrada, seu processamento e sua escrita no disco. Dessa forma, uma leitura é realizada no disco sempre que uma operação precisar consumir a saída de uma operação anterior, fato que gera uma sobrecarga no caso de um processamento iterativo (SHI *et al.*, 2015).

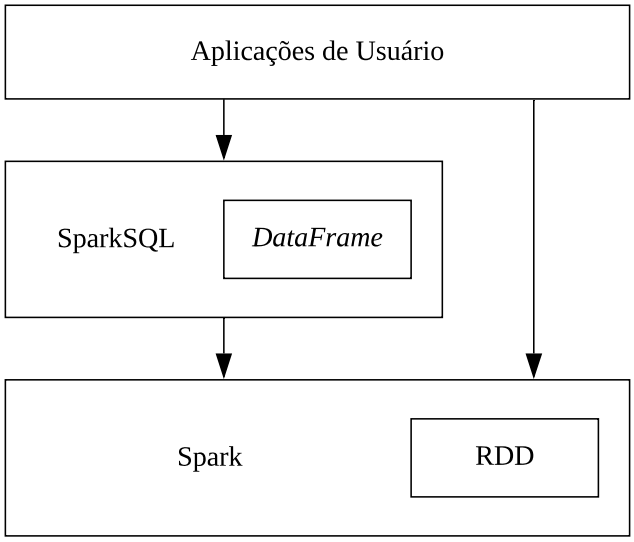
Com o intuito de prover uma solução viável para aplicações que realizem esse tipo de processamento, o *framework* Spark foi proposto. Esse *framework* trabalha de forma similar ao Hadoop, utilizando um processo mestre, denominado *driver*, para controlar o fluxo da aplicação e distribuir várias operações em paralelo, cuja execução é realizada por processos executores (ZAHARIA *et al.*, 2010). A principal diferença entre os dois *frameworks* reside no fato de que o Spark emprega abstrações denominadas *datasets* distribuídos e resilientes (RDDs) (LI *et al.*, 2017). Essas abstrações podem ser definidas como sendo coleções de blocos de dados distribuídos capazes de serem reconstruídos caso a partição onde estão armazenados venha a ser perdida. Os RDDs permitem aos programadores o armazenamento de saídas intermediárias em memória principal, evitando uma grande quantidade de acessos a disco quando se deseja reutilizar essas saídas em futuras operações, como ocorre no Hadoop.

O *framework* Spark permite que diversas operações sejam executadas sobre os RDDs, tais como (ZAHARIA *et al.*, 2010): (i) *parallelize*; (ii) *flatMap*; (iii) *reduce*; (iv) *collect*; (v) *foreach*; (vi) *cache*; e (vii) *save*. A operação *parallelize* divide uma coleção (que pode ser um vetor de dados, por exemplo) em diversas fatias que são enviadas para vários nós distintos para serem processadas. Já a operação *flatMap* tem a mesma semântica da operação *map* do *framework* Hadoop, transformando um *dataset* de elementos de um dado tipo em um *dataset* de elementos de outro tipo por meio de uma função definida pelo usuário. A operação *reduce* também é similar àquela presente no *framework* Hadoop, combinando elementos do *dataset* por meio do uso de uma função associativa para gerar um determinado resultado. Ademais, enquanto a operação *collect* envia todos os dados do dataset para o processo driver, a operação *foreach* executa uma função fornecida pelo usuário para cada elemento do *dataset*. Por fim, as operações *cache* e *save* se referem à persistência dos RDDs: enquanto a primeira garante que o *dataset* seja mantido em memória principal após seu uso, a segunda o persiste no banco de dados.

O vasto conjunto de operações nativas dos RDDs garante uma flexibilidade significativa para o usuário durante o desenvolvimento de suas aplicações. Porém, utilizá-los na implementação de consultas relacionais pode se provar uma tarefa onerosa, uma vez que os RDDs não são capazes de interpretar de forma nativa linguagens declarativas tais como a *Structured Query*

Language (SQL). Nesse contexto, o SparkSQL (ARMBRUST *et al.*, 2015) foi proposto, permitindo a execução de consultas SQL no *framework* Spark. Para tal, o SparkSQL disponibiliza uma abstração denominada *DataFrame*, a qual se assemelha a uma tabela de um banco de dados relacional. Sendo assim, um *DataFrame* pode ser definido como sendo uma coleção distribuída de tuplas com um esquema homogêneo. Essa estrutura permite que o *DataFrame* seja interpretado como sendo um RDD de tuplas, fato que torna possível sua manipulação tanto por meio de consultas SQL quanto por meio das operações nativas dos RDDs. Essa flexibilidade é evidenciada na Figura 6, a qual ilustra a comunicação entre as aplicações de usuário, o SparkSQL e o Spark.

Figura 6 – Comunicação entre as aplicações de usuário, o SparkSQL e o Spark. Fonte: Adaptada de Armbrust *et al.* (2015).

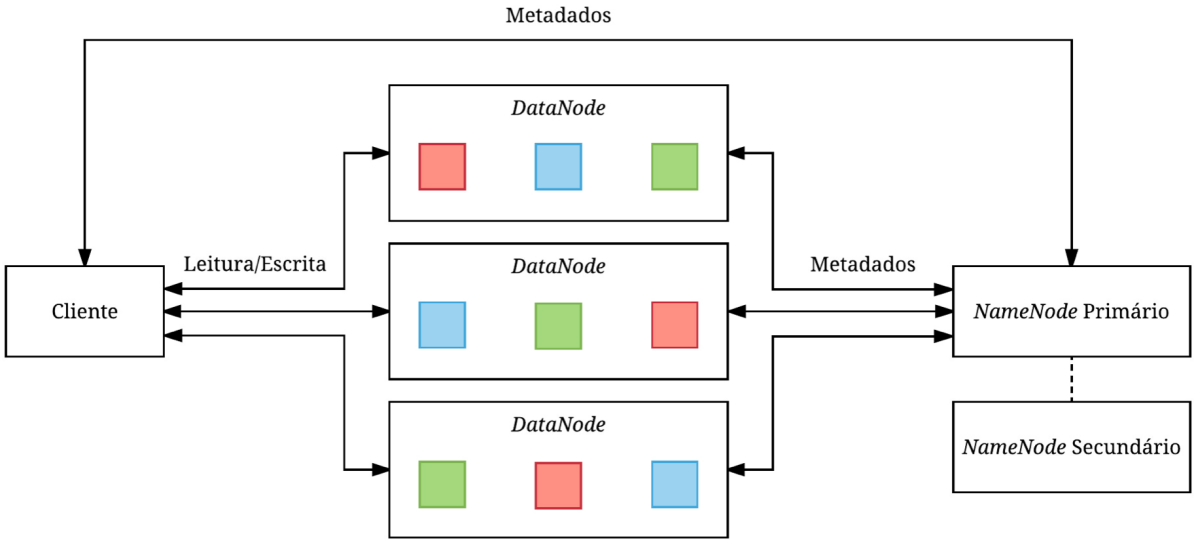


Tanto o *framework* Hadoop quanto o *framework* Spark utilizam o Hadoop *Distributed File System* (HDFS) como sistema de arquivos distribuídos padrão (MORAIS, 2015). O HDFS foi desenvolvido com o intuito de armazenar conjuntos de dados muito volumosos de forma confiável, permitindo que aplicações cliente os acessem de uma forma ágil. Essa agilidade está intrinsicamente relacionada ao fato de que o HDFS mantém réplicas de seus arquivos de dados repartidas entre os nós do ambiente paralelo e distribuído, aumentando a possibilidade de uma aplicação estar sendo executada em um nó próximo àquele que contém o arquivo de dados que esta precisa acessar (SHVACHKO *et al.*, 2010).

A arquitetura do HDFS inclui dois tipos de nó: (i) o *NameNode*, cujo papel reside na manutenção de metadados sobre os dados armazenados; e (ii) os *DataNodes*, que são responsáveis pelo armazenamento dos arquivos de dados em si. Uma aplicação cliente se comunica inicialmente com o *NameNode*, a fim de analisar os metadados armazenados para descobrir em qual *DataNode* os arquivos de dados que são manipulados estão localizados. Vale ressaltar que o HDFS também mantém um *NameNode* secundário, que funciona como um *backup* do *NameNode* primário caso este venha a falhar (SHVACHKO *et al.*, 2010).

Na Figura 7 é ilustrada a arquitetura do HDFS. Nessa, cada arquivo de dados é representado por meio de um quadrado de cor distinta, com suas réplicas sendo representadas por quadrados da mesma cor. Por padrão, o HDFS armazena três réplicas de cada arquivo de dados (SHVACHKO *et al.*, 2010). É possível observar a comunicação da aplicação cliente com o *NameNode* para a obtenção dos metadados e com os DataNodes para a realização das operações de leitura e escrita nos arquivos de dados.

Figura 7 – Arquitetura do HDFS. Fonte: Elaborada pelo autor.



2.3.3 Sistemas Analíticos Espaciais

Embora os *frameworks* Hadoop e Spark apresentem uma eficiência considerável para processar dados convencionais (por exemplo, dados alfanuméricos), a mesma situação não ocorre quando o objeto deste processamento envolve dados espaciais. Uma vez que esses *frameworks* não disponibilizam suporte nativo para esse tipo de dados, limitações relacionadas à sua manipulação podem ser observadas. Por exemplo, mecanismos de indexação que permitem acessos seletivos a regiões específicas de objetos espaciais não são fornecidos, impedindo o desenvolvimento de algoritmos eficientes para processar consultas espaciais (GARCÍA-GARCÍA *et al.*, 2017).

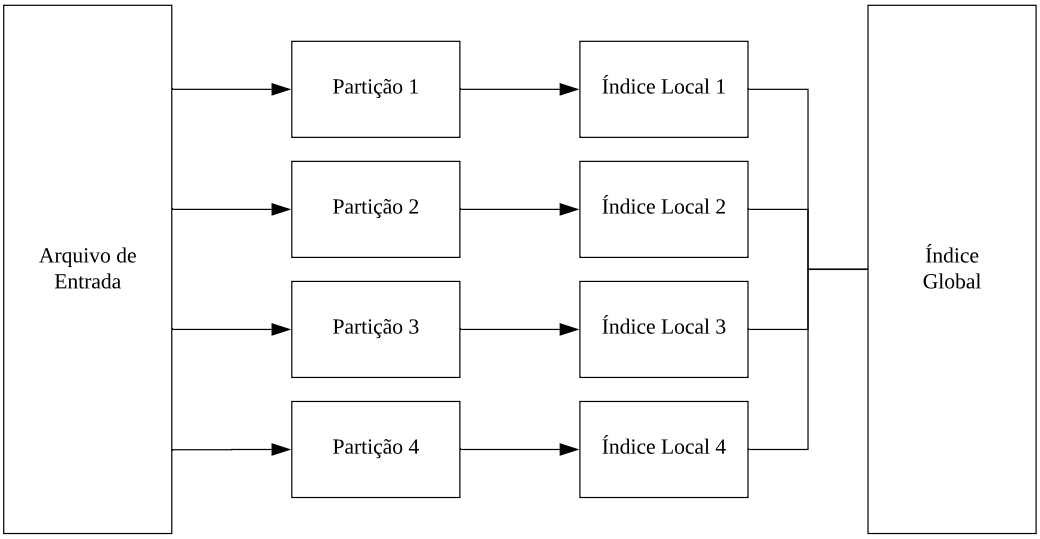
SAEs surgiram para superar essa limitação (PANDEY *et al.*, 2018). Esses sistemas incorporam o conceito de manipulação de dados espaciais vetoriais nos *frameworks* Hadoop e Spark, fornecendo técnicas otimizadas para o processamento de consultas espaciais. Os SAEs também adicionam uma camada que simplifica a interação do usuário com esse processamento, transformando a manipulação de dados espaciais em um ambiente paralelo e distribuído em uma tarefa menos onerosa. Dentre as otimizações disponibilizadas pelos SAEs, as técnicas de particionamento espacial e de indexação espacial distribuída são detalhadas no presente trabalho.

Técnicas de particionamento espacial são empregadas por SAEs com o intuito de particionar um conjunto de dados utilizando como critério suas características espaciais. Dessa

forma, objetos espaciais cuja localização geográfica seja próxima são alocados no mesmo nó do ambiente paralelo e distribuído. Por meio da utilização desse tipo de particionamento, o processamento de consultas espaciais pode ser otimizado ao tirar proveito da localização física de tais objetos (YU; ZHANG; SARWAT, 2019).

Uma vez que um SAE tenha realizado o particionamento de um conjunto de dados espaciais entre os diferentes nós de um ambiente paralelo e distribuído, estruturas de indexação espacial distribuída podem ser construídas. Na maioria dos casos, dois tipos de estruturas são criadas (GARCÍA-GARCÍA *et al.*, 2017): (i) um índice global, localizado no nó mestre, apontando para cada partição de dados; e (ii) múltiplos índices locais, cada um localizado em uma partição de dados, apontando para os dados armazenados dentro da partição. O índice global, criado com a mesma estrutura de dados utilizada no particionamento espacial, é aplicado antes do processamento de uma consulta a fim de eliminar as partições desnecessárias para a obtenção dos resultados. Este fato resulta em uma diminuição dos custos atrelados à comunicação de rede durante tal processamento. Em seguida, os índices locais são empregados a fim de otimizar o processamento da consulta espacial, o qual é realizado de forma paralela em cada partição não eliminada. Na Figura 8 é ilustrado o conceito de indexação espacial distribuída, destacando a comunicação entre o índice global, os índices locais e cada partição de dados.

Figura 8 – Tipos de estrutura envolvidos na indexação espacial distribuída. Fonte: Adaptada de Pandey *et al.* (2018).



As técnicas de particionamento espacial empregadas pelos SAEs revisados no presente trabalho são baseadas nos seguintes algoritmos: *Region Quadtree* (FINKEL; BENTLEY, 1974), KD-*Tree* (BENTLEY, 1975), KDB-*Tree* (ROBINSON, 1981), *Grid File* (NIEVERGELT; HINTERBERGER; SEVCIK, 1984), *Quadtree* (SAMET, 1984), R-*Tree* (GUTTMAN, 1984), Z-*Curve* (ORENSTEIN; MERRETT, 1984), Voronoi *Diagram* (AURENHAMMER, 1991), *Sort Tile Recursive* (LEUTENEGGER; LOPEZ; EDGINGTON, 1997), Hilbert *Curve* (LAWDER; KING, 2001), *Strip/Boundary Optimized Strip* e *Binary Split* (VO; AJI; WANG, 2014)

e *Sort Tile Recursive+* (ELDAWY; ALARABI; MOKBEL, 2015). Alguns desses algoritmos também podem ser utilizados como técnicas para a construção de índices locais, tais como a R-*Tree*, a *Quadtree*, o *Grid File* e a Z-*Curve*. Ademais, os SAEs revisados também utilizam a R+-*Tree* (SELLIS; ROUSSOPOULOS; FALOUTSOS, 1987), a R*-*Tree* (BECKMANN *et al.*, 1990) e a XZ-*Curve* (BÖXHM; KLUMP; KRIEGEL, 1999) para gerar tais índices.

Como o detalhamento do funcionamento de cada um dos algoritmos supracitados não é necessário para a compreensão do presente trabalho, estes não são descritos em maior profundidade nesta seção. Além disso, discussões acerca de quais técnicas são implementadas por cada SAE são realizadas no Capítulo 4.

2.4 Considerações Finais

Neste capítulo foram descritos os conceitos necessários para a compreensão do presente trabalho. Para tal, foram consideradas as perspectivas de: (i) manipulação de dados espaciais, sendo descritos conceitos acerca da definição dos tipos de dados espaciais, relacionamentos espaciais, consultas espaciais e indexação espacial; e (ii) ambientes de computação paralela e distribuída, sendo discutidas as definições de *clusters* de computadores, de computação em nuvem, dos *frameworks* Hadoop e Spark e de sistemas analíticos espaciais.

No próximo capítulo, uma revisão da literatura é realizada. São descritas abordagens correlatas ao presente trabalho, a fim de contextualizar seu respectivo tema de pesquisa e evidenciar seus diferenciais perante o estado da arte.

CAPÍTULO

3

TRABALHOS CORRELATOS

3.1 Considerações Iniciais

Este capítulo resume a revisão do estado da arte realizada no presente trabalho de mestrado. Na seção 3.2 essa revisão é contextualizada, com os trabalhos correlatos sendo divididos em dois grupos. Enquanto a seção 3.3 contém a descrição dos trabalhos referentes ao primeiro grupo, as abordagens categorizadas no segundo grupo são detalhadas na seção 3.4. Por fim, na seção 3.5 são feitas as considerações finais do capítulo.

3.2 Contextualização

O presente trabalho propõe uma comparação de SAEs baseados em Hadoop e Spark sob a perspectiva de usuários que projetam, desenvolvem e implementam aplicações espaciais para organizações. Essa comparação leva em consideração as seguintes características: (i) características gerais dos SAEs; (ii) características relacionadas à manipulação distribuída de dados espaciais, as quais consideram as definições propostas por Güting (1994) e os padrões definidos pela *Open Geospatial Consortium* (OGC) (OGC, 2019); e (iii) características inerentes ao ambiente distribuído. O intuito dessa comparação reside em auxiliar o usuário durante o processo de escolha do SAE mais adequado para o desenvolvimento de aplicações espaciais específicas. Para tal, o presente trabalho disponibiliza: (i) diretrizes cujo objetivo reside em guiar os usuários em tal escolha; (ii) estudos de caso que exemplificam a utilização prática dessas diretrizes; e (iii) uma análise das tendências cronológicas e das limitações dos SAEs revisados.

No melhor do conhecimento do autor deste trabalho, não existem abordagens na literatura que realizem comparações entre SAEs sob a perspectiva de seus usuários. Sendo assim, considera-se como trabalho correlato aquele que realiza algum tipo de comparação qualitativa entre esses sistemas. Ou seja, são consideradas abordagens que façam um levantamento, mesmo que pouco detalhado, das características de SAEs baseados nos *frameworks* Hadoop e Spark. A partir

dessa definição, os trabalhos correlatos aqui descritos podem ser categorizados em dois grupos distintos: (i) abordagens que introduzem SAEs ou extensões para SAEs; e (ii) abordagens que realizam comparações centradas em desempenho, ou seja, comparações de SAEs que visam mensurar o desempenho desses sistemas durante a execução de consultas espaciais.

3.3 Grupo 1: Sistemas Analíticos Espaciais e Extensões

Ao desenvolver um novo SAE, seus autores devem compará-lo com outros sistemas existentes no estado da arte a fim de evidenciar quais de suas características representam um diferencial em relação aos seus predecessores. Uma situação similar pode ser observada em trabalhos que visam propor extensões para SAEs: é necessário que uma comparação entre estes sistemas seja realizada, visto que precisa-se comprovar a efetiva necessidade da extensão sendo proposta. Dessa forma, essas abordagens fazem parte do primeiro grupo de trabalhos correlatos aqui descrito, uma vez que realizam algum tipo de comparação qualitativa entre SAEs.

Os primeiros trabalhos deste grupo são aqueles introduzidos em Aji *et al.* (2013) e Eldawy e Mokbel (2015). Esses trabalhos descrevem as características de dois SAEs: Hadoop-GIS e SpatialHadoop, respectivamente. Como esses são os primeiros SAEs baseados em Hadoop a surgir na literatura, seus autores comparam suas características com aquelas presentes em outros sistemas capazes de processar consultas espaciais de forma paralela e distribuída, tais como extensões espaciais para bancos de dados não relacionais. Já os trabalhos que descrevem SAEs baseados em Spark, por sua vez, comparam as características dos sistemas que introduzem com aquelas apresentadas por seus predecessores. Esses trabalhos descrevem os seguintes sistemas: (i) SpatialSpark (YOU; ZHANG; GRUENWALD, 2015); (ii) GeoSpark (YU; WU; SARWAT, 2015; YU; ZHANG; SARWAT, 2019); (iii) GeoMesa Spark (HUGHES *et al.*, 2015); (iv) Simba (XIE *et al.*, 2016); (v) LocationSpark (TANG *et al.*, 2016); (vi) STARK (HAGEDORN; GÖTZE; SATTLER, 2017b); (vii) SparkGIS (BAIG *et al.*, 2017); e (viii) Elcano (ENGÉLINUS; BADARD, 2018). O sistema Magellan não possui publicações associadas na literatura.

Diferentemente dos trabalhos supracitados, os quais visam propor novos SAEs, o trabalho de García-García *et al.* (2017) apresenta algoritmos que visam estender a capacidade de processamento de junções espaciais baseadas em distância nesses sistemas. A fim de fundamentar seu trabalho, os autores realizam uma comparação qualitativa entre os seguintes SAEs: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, Simba e LocationSpark. Uma comparação mais detalhada é realizada entre os sistemas SpatialHadoop e LocationSpark, uma vez que estes são utilizados pelos autores para a implementação de seus algoritmos.

As abordagens descritas neste grupo diferem em diversos aspectos daquela empregada no presente trabalho de mestrado. Primeiramente, as comparações realizadas pelas mesmas são pouco detalhadas, uma vez que não abordam de forma completa todas as características dos SAEs consideradas no presente trabalho. Ademais, os SAEs não são comparados por meio de

uma perspectiva centrada no usuário, ou seja, com o intuito de assessorá-lo na escolha do sistema ideal para a implementação de suas aplicações espaciais. Dessa forma, nem diretrizes centradas no usuário tampouco análises de tendências cronológicas e limitações dos SAEs são propostas.

3.4 Grupo 2: Comparações Centradas em Desempenho

O segundo grupo de trabalhos correlatos engloba abordagens que realizam comparações centradas em desempenho entre SAEs baseados em Hadoop e Spark. Trabalhos desse grupo possuem como objetivo principal a realização de uma análise quantitativa do desempenho de diferentes SAEs durante a execução de consultas espaciais. Porém, um levantamento das características desses sistemas também é frequentemente realizado, a fim de fundamentar a comparação centrada em desempenho sendo realizada.

O primeiro trabalho que compõe o presente grupo é o de Hagedorn, Götze e Sattler (2017a). Nesse trabalho, os autores realizam testes de desempenho envolvendo a execução de *range queries* e junções espaciais nos seguintes SAEs: SpatialHadoop, SpatialSpark, GeoSpark e STARK. Além disso, uma comparação qualitativa também é realizada, a qual envolve um levantamento de características não apenas desses sistemas, como também do Hadoop-GIS. Os SAEs são comparados em termos de seu suporte para particionamento e indexação espacial, suporte para linguagens de consulta e suporte para diferentes predicados espaciais em *range queries* e junções espaciais.

Uma comparação quantitativa mais ampla é apresentada no estudo de Pandey *et al.* (2018). Inicialmente, algumas características dos sistemas SpatialHadoop, Hadoop-GIS, SpatialSpark, GeoSpark, Simba, LocationSpark e Magellan são comparadas. Dentre essas, é possível destacar os tipos de dados espaciais presentes em cada sistema, as linguagens de consulta disponibilizadas, os tipos de consulta fornecidos e o suporte dos SAEs para diferentes técnicas de indexação e particionamento espacial. Em seguida, uma quantidade considerável de experimentos é realizada nos SAEs baseados em Spark. Esses experimentos consideram cinco tipos diferentes de consultas espaciais (*range query*, KNN *query*, junções espaciais envolvendo conjuntos com tipos de dados espaciais distintos, junções espaciais baseadas em distância e KNN *join query*) e quatro tipos diferentes de dados espaciais (pontos, linhas, retângulos e polígonos).

Por fim, o trabalho de Alam, Ray e Bhavsar (2018) também pode ser incluído no presente grupo. Esse estudo introduz um *benchmark* para comparações quantitativas entre SAEs, o qual é composto por uma carga de trabalho que envolve diversas operações de junção espacial, operações de *range queries* e outras operações, tais como *convex hull* e cálculo de MBRs. Esse *benchmark* é utilizado para a execução de testes de desempenho entre os SAEs SpatialHadoop e GeoSpark, avaliando seus respectivos tempos de execução para cada consulta apresentada na carga de trabalho. Ademais, uma comparação qualitativa também é realizada, a qual se baseia nos padrões definidos pela OGC para levantar características não apenas desses dois sistemas,

como também dos SAEs Hadoop-GIS, SpatialSpark, Simba, LocationSpark e STARK. Dentre as características comparadas, é possível destacar os tipos de dados espaciais presentes em cada sistema, as representações de objetos espaciais que estes são capazes de processar, as linguagens de consulta disponibilizadas, os tipos de consulta fornecidos e o suporte dos SAEs para diferentes técnicas de indexação e particionamento espacial.

Embora as abordagens revisadas neste grupo ofereçam uma análise comparativa de vários SAEs, estas são baseadas em uma visão centrada em desempenho, a qual possui como principal objetivo a comparação desses sistemas de forma quantitativa. Por outro lado, o presente trabalho revisa uma quantidade expressiva de SAEs com base em uma visão centrada no usuário, cujo objetivo reside em ajudá-lo a entender como as características desses sistemas são úteis para atender aos requisitos específicos de suas aplicações espaciais. Dessa forma, os trabalhos descritos no presente grupo não apresentam diretrizes centradas no usuário nem análises das tendências cronológicas e das limitações dos SAEs.

Outrossim, a maioria das comparações centradas em desempenho revisadas utiliza os artigos científicos dos SAEs como sua única fonte de informação. O presente trabalho, por outro lado, utiliza outras fontes de dados como base para a construção de sua análise comparativa, tais como a documentação adicional dos SAEs, o código fonte de sua última versão pública e também seus artigos científicos. Esse fato enriquece a análise comparativa de forma considerável. Ao analisar uma fonte mais ampla de informações, é possível observar que algumas funcionalidades descritas nos trabalhos científicos de alguns SAEs ainda não foram implementadas em sua última versão pública. Sendo assim, algumas diferenças no conteúdo das comparações descritas no presente trabalho e nos trabalhos correlatos podem ser notadas. Outro diferencial deste trabalho reside na inclusão dos SAEs GeoMesa Spark, SparkGIS e Elcano em sua análise comparativa.

3.5 Considerações Finais

Neste capítulo foram descritas abordagens existentes na literatura cujo tema pode ser considerado correlato àquele apresentado neste trabalho. Esse tema consiste na execução de uma comparação de SAEs baseados nos *frameworks* Hadoop e Spark sob a ótica de seus usuários. Como a revisão da literatura não foi capaz de encontrar trabalhos que analisam SAEs com base nessa perspectiva, foram considerados como trabalhos correlatos aqueles que realizam algum tipo de comparação qualitativa entre esses sistemas. A partir dessa definição, esses trabalhos foram categorizados em dois grupos distintos: (i) abordagens que introduzem SAEs ou extensões para SAEs; e (ii) abordagens que realizam comparações centradas em desempenho. Ambos os grupos tiveram seus trabalhos descritos com o foco nas comparações qualitativas realizadas, evidenciando suas limitações perante o presente trabalho.

No Quadro 1 são sintetizadas as diferenças existentes entre o trabalho realizado e os trabalhos correlatos. Nesse, é possível perceber que o presente trabalho se distingue dos demais

devido à perspectiva adotada em sua comparação. Ademais, também é possível identificar diferenciais deste trabalho no que diz respeito ao fornecimento de diretrizes centradas no usuário e de uma análise de tendências cronológicas e limitações dos SAEs. O Quadro 1 também evidencia o fato de que comparações centradas em desempenho estão fora do escopo do presente trabalho.

Quadro 1 – Síntese da revisão de trabalhos correlatos. Fonte: Elaborado pelo autor.

Aspecto Analisado	**Trabalhos do Grupo 1**	**Trabalhos do Grupo 2**	**Presente Trabalho**
Comparação centrada na perspectiva do usuário	✗	✗	✓
Execução de testes de desempenho	✓	✓	✗
Quantidade relevante de SAEs nas comparações	Somente em alguns trabalhos	✓	✓
Utilização dos padrões da OGC e das definições de Güting (1994) em comparações qualitativas	✗	Somente em Alam, Ray e Bhavsar (2018)	✓
Fornecimento de diretrizes centradas no usuário	✗	✗	✓
Análise de tendências cronológicas e de limitações dos SAEs	✗	✗	✓

O próximo capítulo apresenta a comparação centrada no usuário supracitada, com cada característica abordada sendo discutida no decorrer do capítulo. Também são detalhados os critérios empregados para a escolha dos SAEs e as fontes de dados utilizadas para a extração de suas características.

CAPÍTULO

4

COMPARAÇÃO CENTRADA NO USUÁRIO

4.1 Considerações Iniciais

Este capítulo apresenta uma comparação de diferentes SAEs sob a ótica de seus usuários. Na seção 4.2 essa comparação é contextualizada, sendo subdividida em três diferentes perspectivas. Na seção 4.3, a perspectiva de características gerais é abordada. Já as perspectivas de manipulação de dados espaciais e de aspectos inerentes ao ambiente distribuído são tratadas nas seções 4.4 e 4.5, respectivamente. Por fim, na seção 4.6 são feitas as considerações finais do capítulo.

4.2 Contextualização

Neste capítulo, os seguintes SAEs são comparados sob a ótica de usuários que projetam, desenvolvem e implementam aplicações espaciais para organizações: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, GeoMesa Spark, Simba, LocationSpark, STARK, Magellan, SparkGIS e Elcano. Os critérios utilizados para a escolha dos SAEs comparados são descritos a seguir. Foram analisados somente sistemas baseados nos *frameworks* Hadoop e Spark, uma vez que estes representam a maioria dos SAEs disponíveis na literatura (PANDEY *et al.*, 2018). Além disso, somente SAEs capazes de representar dados espaciais por meio do modelo vetorial foram selecionados. Ademais, propostas existentes no estado da arte que consistem em otimizações muito específicas de um único tipo de operação espacial não foram consideradas na presente análise comparativa, tais como otimizações específicas para o processamento de consultas baseadas em distância (GARCÍA-GARCÍA *et al.*, 2017).

A presente análise comparativa emprega uma metodologia de pesquisa cuja coleta de dados considera as seguintes fontes: (i) literatura científica que descreve os SAEs; (ii) documentações disponíveis para cada SAE, tais como *websites*, manuais de usuário e tutoriais; e (iii) código fonte do SAE, quando disponível. Com o uso de todas essas fontes, a análise comparativa é realizada considerando uma quantidade extensa de critérios, os quais englobam três diferentes perspectivas da ótica de um usuário: (i) características gerais; (ii) manipulação de

dados espaciais; e (iii) aspectos inerentes ao ambiente distribuído.

Para cada perspectiva, um conjunto de quadros é apresentado. Cada linha de um quadro representa um SAE, enquanto cada uma de suas colunas representa um critério inerente à perspectiva abordada. Nesses quadros, a expressão "não informado" é utilizada para indicar que nenhuma informação sobre o critério analisado está disponível no artigo científico do SAE e em sua documentação associada. Nesses casos, nenhuma investigação adicional é realizada devido ao fato do sistema ainda não possuir uma implementação pública.

4.3 Características Gerais

As características gerais dos SAEs comparadas nos Quadros 2 e 3 se referem a: ano de surgimento, *framework* base, versão do *framework* base, versão do SAE, licença, biblioteca espacial, linguagem de programação, linguagem de consulta, completude da documentação e suporte da comunidade. A motivação e o contexto relacionados a cada característica são discutidos no decorrer da seção.

Quadro 2 – Visão geral dos SAEs, considerando seu ano de surgimento, *framework* base, versão do *framework* base, versão do SAE e licença. Fonte: Elaborado pelo autor.

SAE	**Ano de Surgimento**	***Framework* Base**	**Versão do *Framework* Base**	**Versão do SAE**	**Licença**
Hadoop-GIS	2013	Apache Hadoop	Não informado	Não informado	GNU *General Public License* (Versão 2)
SpatialHadoop	2013	Apache Hadoop	Versão 2.7.2	Versão 2.4.2	*Apache License* (Versão 2.0)
SpatialSpark	2015	Apache Spark	Versão 2.0.2	Versão 1.0	*Apache License* (Versão 2.0)
GeoSpark	2015	Apache Spark	Versões 2.X or 1.X	Versão 1.2.0	*Apache License* (Versão 2.0)
GeoMesa Spark	2015	Apache Spark	Versões 2.2.X, 2.3.X, or 2.4.X	Versão 2.3.1	*Apache License* (Versão 2.0)
Simba	2016	Apache Spark	Versões 2.1.X	Versão *Standalone*	*Apache License* (Versão 2.0)
LocationSpark	2016	Apache Spark	Não informado	*The First Version*	Não informado
STARK	2017	Apache Spark	Não informado	Não informado	Não informado
Magellan	2017	Apache Spark	Versão 2.1 ou superior	Versão 1.0.5	*Apache License* (Versão 2.0)
SparkGIS	2017	Apache Spark	Não informado	Sem versões disponíveis	Não informado
Elcano	2018	Apache Spark	Não informado	Sem versões disponíveis	Não informado

Ano de Surgimento. No presente trabalho, o ano de surgimento de um SAE representa o ano de publicação de seu primeiro artigo científico. Uma exceção pode ser observada para o Magellan, o qual não possui nenhuma publicação associada. Para esse SAE específico, o ano de surgimento

representa o ano de publicação de sua primeira versão em seu respectivo repositório. Nos quadros descritos ao longo deste capítulo, o ano de surgimento é usado para organizar os SAEs em ordem cronológica, fato que permite ao usuário analisar a evolução histórica de suas funcionalidades.

***Framework* Base.** É importante que o usuário de um SAE possua conhecimento prévio acerca do *framework* base utilizado pelo sistema. Esses *frameworks* possuem configurações específicas que devem ser adequadamente ajustadas para que um SAE funcione de forma correta e também características intrínsecas que devem ser levadas em consideração ao definir uma consulta espacial (PANDEY *et al.*, 2018). No melhor do conhecimento do autor do presente trabalho, existem atualmente dois sistemas baseados em Hadoop e nove sistemas baseados em Spark na literatura, conforme evidenciado no Quadro 2. Essa diferença de quantidade pode ser justificada pelo fato de que o *framework* base de um SAE impacta o desempenho de aplicações espaciais. Esse impacto está relacionado aos custos de leitura e escrita de dados em memória secundária. Conforme detalhado na seção 2.3.2, enquanto sistemas baseados em Hadoop requerem que os dados intermediários de suas operações sejam armazenados em disco, sistemas baseados em Spark armazenam esses dados na memória principal por meio do uso de RDDs. Sendo assim, sistemas baseados em Spark normalmente possuem um desempenho superior quando comparados àqueles baseados em Hadoop (PANDEY *et al.*, 2018).

Versão do SAE e de seu *Framework* Base. Cada versão de um SAE é desenvolvida de acordo com uma versão específica de seu *framework* base. É importante que o usuário verifique essas versões antes do processo de instalação de um SAE a fim de evitar futuros empecilhos. O Quadro 2 revela que a maioria dos SAEs revisados especificam essas versões ou em um *website* dedicado ou em seus respectivos repositórios. Os sistemas Hadoop-GIS, LocationSpark e STARK são exceções, uma vez que estes não disponibilizam nenhuma informação relacionada ao versionamento de suas distribuições e/ou de seus *frameworks* base.

Licença. Durante o desenvolvimento de uma aplicação espacial, o usuário deve estar ciente da licença de seus componentes externos. Até mesmo componentes *open-source* podem possuir licenças protetivas, limitando o acesso do usuário no que diz respeito à sua modificação e distribuição (STEINIGER; BOCHER, 2009). No caso de aplicações baseadas em SAEs, o próprio SAE pode ser considerado como sendo seu principal componente. Sendo assim, o usuário deve observar cautelosamente se existem limitações atreladas à licença do SAE a ser empregado que possam afetar a distribuição da aplicação espacial a ser desenvolvida. O Quadro 2 evidencia que os SAEs revisados na presente pesquisa adotam ou a GNU *General Public License*, Versão 2 (GNU, 2019), ou a *Apache License*, Versão 2.0 (APACHE, 2019c).

Biblioteca Espacial. A biblioteca espacial empregada por um SAE impacta diretamente sua capacidade de manipulação de dados espaciais. Se tal biblioteca segue as especificações da OGC (2019), o usuário pode deduzir que o SAE em questão provê suporte para os tipos de dados espaciais, operações espaciais e predicados espaciais especificados pela OGC. De acordo com o

Quadro 3 – Comparação dos SAEs, considerando suas bibliotecas espaciais, linguagens de programação, linguagens de consulta, documentações e suporte da comunidade. Fonte: Elaborado pelo autor.

SAE	Biblioteca Espacial	Linguagem de Programação	Linguagem de Consulta	Completude da Documentação	Suporte da Comunidade
Hadoop-GIS	Implementação própria	C++ e Java	Baseada em HiveQL	Completa	Somente e-mail
SpatialHadoop	Implementação própria	Java	Baseada em Pigeon	Completa	E-mail, rastreamento de problemas
SpatialSpark	JTS	Scala	✗	Intermediária	Somente e-mail
GeoSpark	JTS	Java e Scala	Baseada em SQL	Completa	Twitter, e-mail, fórum de discussão, rastreamento de problemas, chat
GeoMesa Spark	JTS	Scala	Baseada em SQL	Completa	E-mail, rastreamento de problemas, *chat*
Simba	Implementação própria	Scala	Baseada em SQL	Incompleta	E-mail, rastreamento de problemas
LocationSpark	Implementação própria	Scala	✗	Incompleta	Somente e-mail
STARK	JTS	Java e Scala	Baseada em Piglet	Intermediária	E-mail, rastreamento de problemas
Magellan	Implementação própria	Scala	Baseada em SQL	Intermediária	Rastreamento de problemas, chat
SparkGIS	Não informado	Não informado	✗	Não existente	Somente e-mail
Elcano	JTS	Não informado	Baseada em SQL	Não existente	Somente e-mail

Quadro 3, os sistemas SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano utilizam a biblioteca *Java Topology Suite* (JTS) (JTS, 2019), a qual está em conformidade com os padrões da OGC. Por outro lado, o Hadoop-GIS, o SpatialHadoop, o Simba, o LocationSpark e o Magellan não empregam uma biblioteca espacial externa. Como alternativa, esses sistemas utilizam uma implementação própria para manipular dados espaciais.

Linguagem de Programação. A linguagem de programação utilizada no desenvolvimento de um SAE é um aspecto importante a ser considerado por seus usuários. É ideal que eles priorizem sistemas desenvolvidos em uma linguagem na qual tenham conhecimento prévio, uma vez que estudar o código fonte do SAE é essencial para compreender seu funcionamento e utilizá-lo de forma otimizada. Conforme evidenciado no Quadro 3, a linguagem Scala é a mais utilizada para o desenvolvimento dos SAEs, com os sistemas SpatialSpark, GeoMesa Spark, Simba, LocationSpark e Magellan sendo completamente desenvolvidos nesta linguagem. A linguagem Scala também é utilizada em conjunto com Java para o desenvolvimento dos sistemas

GeoSpark e STARK. O SpatialHadoop é o único SAE dentre os revisados no presente trabalho que é desenvolvido exclusivamente em Java, enquanto o Hadoop-GIS possui componentes desenvolvidos tanto em Java quanto em C++.

Linguagem de Consulta. Prover uma linguagem de consulta compreensível é algo que um SAE deve se atentar a fim de simplificar o processo de manipulação de objetos espaciais. SAEs que estendem linguagens de consulta amplamente conhecidas, tal como a linguagem SQL, contribuem para a redução da curva de aprendizado de seus usuários. Em outras palavras, esses sistemas permitem que o usuário se familiarize mais rapidamente com suas funcionalidades. O Quadro 3 revela que os sistemas GeoSpark, GeoMesa Spark, Simba, Magellan e Elcano se destacam ao estender o SparkSQL para prover suporte à execução de consultas espaciais. Ademais, o Hadoop-GIS, o SpatialHadoop e o STARK também estendem linguagens de consulta existentes, porém baseadas nos padrões HiveQL (THUSOO *et al.*, 2010), Pigeon (ELDAWY; MOKBEL, 2014) e Piglet (HAGEDORN; SATTLER, 2016), respectivamente. O restante dos SAEs não oferece nenhuma linguagem de consulta, fato que demanda um esforço adicional por parte de seus usuários durante a recuperação e a manipulação de objetos espaciais.

Completude da Documentação. Uma documentação completa, precisa e atual é um pré-requisito necessário para auxiliar de forma eficiente usuários que projetam, desenvolvem e implementam aplicações espaciais, especialmente quando os mesmos precisam lidar com sistemas pertencentes ao estado da arte. Sendo assim, a falta de uma documentação de qualidade pode impactar negativamente o desenvolvimento dessas aplicações, aumentando consideravelmente o tempo gasto para a execução do projeto. As documentações dos SAEs revisados podem ser classificadas em: (i) completa, quando o SAE disponibiliza uma documentação expressiva em seu *website* dedicado, contendo listas de funcionalidades, detalhamentos acerca dos procedimentos de instalação e tutoriais descrevendo a execução de operações espaciais; (ii) intermediária, quando nenhum *website* dedicado é fornecido, porém uma descrição detalhada das funcionalidades do SAE pode ser encontrada em seu respectivo repositório; (iii) incompleta, quando a documentação disponível para o SAE contém apenas uma pequena descrição desse sistema e exemplos de algumas de suas operações; e (vi) não existente. O Quadro 3 revela que os sistemas Hadoop-GIS, SpatialHadoop, GeoSpark e GeoMesa Spark se destacam em relação à completude de suas documentações.

Suporte da Comunidade. Ao desenvolver uma aplicação espacial baseada em um SAE, é comum que usuários enfrentem diversos desafios, tais como a falta de conhecimento acerca de uma de suas funcionalidades ou o surgimento de erros durante a execução da aplicação. Quando a complexidade do problema não é significativa, é possível que a documentação do SAE forneça todas as informações que o usuário necessita para solucioná-lo. Porém, até mesmo uma documentação completa pode não ser suficiente para auxiliar o usuário na solução de problemas de alta complexidade. Nesses casos, requisitar suporte dos desenvolvedores do SAE utilizado ou

de sua respectiva comunidade pode ser uma alternativa viável. Neste trabalho, são considerados diversos meios de suporte que podem ser disponibilizados por tais desenvolvedores e por tais comunidades, conforme evidenciado no Quadro 3: (i) e-mail; (ii) chat; (iii) redes sociais, tais como Facebook (FACEBOOK, 2019) e Twitter (TWITTER, 2019); (iv) fóruns de discussão; e (v) plataformas de rastreamento de problemas nas quais o usuário pode reportar falhas. Apenas canais de comunicação que apresentam alguma atividade a partir de 2018 são considerados neste trabalho. No caso do suporte via e-mail, foi verificado se os e-mails dos desenvolvedores podem ser localizados nos *websites* dos SAEs, em seus repositórios, ou em suas respectivas publicações.

4.4 Manipulação de Dados Espaciais

As características relacionadas à manipulação de dados espaciais comparadas nos Quadros de 4 a 8 se referem a: tipos de dados espaciais, representações de objetos espaciais, operações geométricas de conjunto, relacionamentos direcionais, relacionamentos métricos, relacionamentos topológicos, operações numéricas, operações atreladas aos tipos de dados, estruturas de indexação espacial e técnicas de junção espacial. Essas características são baseadas nas definições de bancos de dados espaciais propostas por Güting (1994) e nos padrões definidos pela OGC (2019). A motivação e o contexto relacionados a cada característica são discutidos no decorrer da seção.

Quadro 4 – Comparação dos SAEs, considerando o suporte provido aos tipos de dados espaciais e às representações de objetos espaciais. Fonte: Elaborado pelo autor.

	Tipos de Dados Espaciais							
	Simples			**Complexos**			**Representações**	
SAE	**Ponto**	**Linha**	**Polígono**	**Ponto**	**Linha**	**Polígono**	**Textuais**	**Binárias**
Hadoop-GIS	✓	✓	✓	✓	✓	✓	WKT	✗
SpatialHadoop	✓	✗	✓	✗	✗	✗	CSV	✗
SpatialSpark	✓	✓	✓	✓	✓	✓	WKT	✗
GeoSpark	✓	✓	✓	✓	✓	✓	CSV, TSV, WKT, GeoJSON	WKB, *Shapefile*, NetCDF / HDF
GeoMesa Spark	✓	✓	✓	✓	✓	✓	WKT	WKB
Simba	✓	✗	✗	✗	✗	✗	CSV	✗
LocationSpark	✓	✗	✗	✗	✗	✗	✗	✗
STARK	✓	✓	✓	✗	✗	✗	WKT	✗
Magellan	✓	✓	✓	✓	✗	✓	WKT, GeoJSON, OSM-XML	*Shapefile*
SparkGIS	✓	✓	✓	Não informado	Não informado	Não informado	Não informado	Não informado
Elcano	✓	✓	✓	✓	✓	✓	WKT	✗

Tipos de Dados Espaciais. Prover suporte para uma grande variedade de tipos de dados espaciais é fundamental para representar de forma apropriada fenômenos geográficos de diferentes dimensões, tais como pontos, linhas e polígonos (GÜTING, 1994). Polígonos também podem derivar objetos espaciais de tipos mais específicos, tais como retângulos, por exemplo. Ademais, conforme detalhado na seção 2.2, os tipos de dados espaciais podem ser categorizados em: (i) simples, cujos objetos espaciais possuem apenas um único componente; ou (ii) complexos, cujos objetos espaciais podem possuir múltiplos componentes, porém finitos. De acordo com o Quadro 4, os sistemas Hadoop-GIS, SpatialSpark, GeoSpark, GeoMesa Spark, Magellan e Elcano provêem suporte para tipos de dados espaciais simples e complexos. O restante dos SAEs, por outro lado, limitam seu escopo. Por exemplo, o STARK provê suporte somente para tipos de dados espaciais simples. Apesar desse sistema adotar a biblioteca espacial JTS, nenhuma função é disponibilizada para manipular dados espaciais complexos dentro de sua abstração de conjunto de dados. Outrossim, o Simba e o LocationSpark disponibilizam suporte somente para pontos e retângulos simples. Por fim, o SpatialHadoop permite que os usuários implementem seus próprios tipos de dados espaciais por meio de uma interface de extensão.

Representações de Objetos Espaciais. Objetos espaciais podem assumir representações distintas a fim de atender a diferentes propósitos, tais como a interoperabilidade entre aplicações, o carregamento desses objetos nos SAEs e o processo de visualização dos mesmos. Exemplos dessas representações incluem: (i) *Comma/Tab Separated Values* (CSV/TSV), representações textuais nas quais as coordenadas dos objetos espaciais são separadas por vírgulas ou por espaços em branco, respectivamente; (ii) *Geographic JavaScript Object Notation* (GeoJSON), representação textual útil para facilitar a visualização de objetos espaciais em aplicações *web* (BUTLER *et al.*, 2016); (iii) *Well-Known Text* (WKT) e *Well-Known Binary* (WKB), padrões bem conhecidos para representar objetos espaciais de forma textual ou de forma binária, respectivamente (OGC, 2019); (iv) *OpenStreetMaps Extensible Markup Language* (OSM-XML), representação textual utilizada em dados provenientes do *OpenStreetMaps* (HAKLAY; WEBER, 2008); e (v) *Network Common Data Form/Hierarchical Data Format* (NetCDF/HDF), representação binária utilizada majoritariamente para a reprodução de dados científicos (UNIDATA, 2019). Conforme demonstrado no Quadro 4, os SAEs GeoSpark, GeoMesa Spark e Magellan são os únicos dentre os revisados que possuem tanto representações textuais quanto representações binárias para a importação e exportação de objetos espaciais. O LocationSpark, por outro lado, não possui nenhum tipo de função encapsulada capaz de processar quaisquer tipos de representações, sobrecarregando o usuário com sua implementação.

Operações Geométricas de Conjunto. Este tipo de operação espacial engloba cálculos relacionados à intersecção geométrica, à união geométrica e à diferença geométrica entre dois objetos espaciais (GUTTMAN, 1984). A execução desses cálculos resulta em outro objeto espacial que pode ser utilizado em novas análises, fato que possibilita uma maior flexibilidade em tomadas de decisões estratégicas e, consequentemente, aprimora a experiência do usuá-

rio. O Quadro 5 evidencia que os sistemas SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano disponibilizam suporte para todos os tipos de operações geométricas de conjunto supracitados. Isso ocorre devido ao fato de que esses sistemas adotam a biblioteca espacial JTS, a qual provê suporte para essas operações de forma nativa. Adicionalmente, os SAEs Hadoop-GIS e SpatialHadoop provêem suporte para até duas operações, enquanto os sistemas Simba, LocationSpark e Magellan não implementam nenhuma dessas operações.

Quadro 5 – Comparação dos SAEs, considerando operações geométricas de conjunto, relacionamentos direcionais e relacionamentos métricos. Fonte: Elaborado pelo autor.

	Operações Geométricas de Conjunto				
SAE	**União**	**Intersecção**	**Diferença**	**Relacionamentos Direcionais**	**Relacionamentos Métricos**
Hadoop-GIS	✓	✓	✗	✗	KNN *query*, *range query* baseada em distância e junção espacial baseada em distância
SpatialHadoop	✓	✗	✗	✗	KNN *query*
SpatialSpark	✓	✓	✓	✗	*Range query* baseada em distância, junção espacial baseada em distância e funções programadas pelo usuário
GeoSpark	✓	✓	✓	✗	KNN *query*, junção espacial baseada em distância e funções programadas pelo usuário
GeoMesa Spark	✓	✓	✓	✗	Funções programadas pelo usuário
Simba	✗	✗	✗	✗	KNN *query*, KNN *join query* e junção espacial baseada em distância
LocationSpark	✗	✗	✗	✗	KNN *query*, KNN *join query* e junção espacial baseada em distância
STARK	✓	✓	✓	✗	KNN *query*, *range query* baseada em distância e funções programadas pelo usuário
Magellan	✗	✗	✗	✗	✗
SparkGIS	Não informado	Não informado	Não informado	Não informado	Ao menos na KNN *join query*
Elcano	✓	✓	✓	Não informado	Ao menos em funções programadas pelo usuário

Relacionamentos Direcionais. O relacionamento direcional entre dois objetos espaciais pode funcionar como uma poderosa restrição de pesquisa em um predicado espacial. Determinar se um objeto espacial está localizado ao norte ou ao sul em relação a uma determinada referência, por exemplo, pode gerar uma redução considerável em um espaço de busca (PEUQUET; CI-XIANG, 1987). Esse tipo de relacionamento também pode ser útil em aplicações cujo objetivo reside na

orientação geográfica de seus usuários. Apesar da aplicabilidade de relacionamentos direcionais em problemas do mundo real, nenhum dos SAEs revisados provê suporte para sua execução, conforme evidenciado no Quadro 5.

Relacionamentos Métricos. Nos SAEs, relacionamentos métricos são normalmente implementados a fim de filtrar de forma eficiente objetos espaciais em consultas baseadas em distância. O Quadro 5 revela que diversos SAEs possuem implementações para pelo menos uma das seguintes operações: (i) KNN *query*; (ii) KNN *join query*; (iii) junção espacial baseada em distância; (iv) *range query* baseada em distância; e (v) operações a serem usadas em funções implementadas pelo usuário, tais como as distâncias mínima e máxima entre dois objetos espaciais. Dentre todos os SAEs revisados, o Magellan é o único que não provê nenhuma implementação de relacionamentos métricos.

Quadro 6 – Comparação dos SAEs, considerando seu suporte para relacionamentos topológicos. Fonte: Elaborado pelo autor.

	Relacionamentos Topológicos							
SAE	**Disjoint**	**Overlap**	**Meet**	**Inside/ Contains**	**Equal**	**Covers/ Covered By**	**Intersects**	**Crosses**
Hadoop-GIS	✓[1, 2]	✓[1, 2]	✓[1, 2]	✓[1, 2]	✓[1, 2]	✗	✓[1, 2]	✓[1, 2]
SpatialHadoop	✗	✓[1, 2]	✗	✗	✗	✗	✗	✗
SpatialSpark	✓[3]	✓[2, 3]	✓[3]	✓[1, 2, 3]	✓[3]	✓[3]	✓[2, 3]	✓[3]
GeoSpark	✓[3]	✓[3]	✓[3]	✓[1, 2, 3]	✓[3]	✓[3]	✓[1, 2, 3]	✓[3]
GeoMesa Spark	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]
Simba	✗	✗	✗	✗	✗	✗	✓[1]	✗
LocationSpark	✗	✗	✗	✓[1, 2]	✗	✗	✓[1, 2]	✗
STARK	✓[3]	✓[3]	✓[3]	✓[1, 2, 3]	✓[3]	✓[1, 2, 3]	✓[1, 2, 3]	✓[3]
Magellan	✗	✗	✗	✓[1, 2]	✗	✗	✓[2]	✗
SparkGIS	Não informado	Não informado	Não informado	Não informado	Não informado	Não informado	Não informado	Não informado
Elcano	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]	✓[3]

[1]Em funções encapsuladas que executam *range queries*.
[2]Em funções encapsuladas que executam junções espaciais.
[3]Em funções programadas pelo usuário.

Relacionamentos Topológicos. Diversas consultas requeridas por aplicações espaciais fazem uso dos relacionamentos topológicos descritos na seção 2.2.1, tais como: (i) seleções que retornam um conjunto de objetos espaciais que satisfaçam um relacionamento topológico com um outro objeto espacial; (ii) *range queries* cujo retorno consiste em objetos espaciais que possuam um relacionamento topológico com uma janela de consulta retangular; e (iii) junções espaciais que associam diferentes conjuntos de objetos espaciais de acordo com um relacionamento topológico. Do ponto de vista do usuário, é importante que um SAE ofereça suporte nativo para uma quantidade expressiva de relacionamentos topológicos, a fim de garantir flexibilidade na definição de consultas em diferentes tipos de aplicações espaciais. Os sistemas Hadoop-GIS, SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano atendem a esse requisito, conforme

evidenciado no Quadro 6. Por outro lado, os SAEs SpatialHadoop, Simba, LocationSpark e Magellan não permitem a especificação de predicados topológicos em consultas espaciais *ad-hoc*. Ao invés disso, esses sistemas disponibilizam somente um subgrupo de consultas espaciais otimizadas que atendem a um conjunto predeterminado de relacionamentos topológicos. Por exemplo, o SpatialHadoop só permite que *range queries* e junções espaciais sejam executadas com o predicado *overlap*.

Quadro 7 – Comparação dos SAEs, considerando a disponibilidade de operações numéricas e de operações atreladas aos tipos de dados. Fonte: Elaborado pelo autor.

	Operações Numéricas			**Operações Atreladas aos Tipos de Dados**		
SAE	**Área de Polígono**	**Comprimento de Linha**	**Outros**	**Ponto**	**Linha**	**Polígono**
Hadoop-GIS	✓	✗	Coeficientes Jaccard e DICE	✗	✗	MBR
SpatialHadoop	✗	✗	✗	*Convex Hull*, *Skyline*	✗	MBR
SpatialSpark	✓	✓	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS
GeoSpark	✓	✓	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS
GeoMesa Spark	✓	✓	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS
Simba	✓[1]	✗	✗	✗	✗	✗
LocationSpark	✗	✗	✗	✗	✗	✗
STARK	✓	✓	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS
Magellan	✗	✗	✗	✗	✗	✗
SparkGIS	Não informado	Não informado	Não informado	Não informado	Não informado	Não informado
Elcano	✓	✓	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS	Operações da biblioteca JTS

[1] Apenas para retângulos.

Operações Numéricas. Esse tipo de operação retorna valores numéricos calculados a partir de propriedades geométricas dos objetos espaciais (GÜTING, 1994), tais como o comprimento de uma linha, a área de um polígono, o número de componentes de um objeto espacial complexo e o número de pontos que compõem um objeto espacial. Ademais, operações numéricas podem ser empregadas na execução de consultas espaciais baseadas em distância, conforme detalhado na categoria de relacionamentos métricos. De acordo com o Quadro 7, diversos SAEs oferecem pelo menos algum suporte para operações numéricas. Nos sistemas SpatialHadoop, LocationSpark e SparkGIS, esse tipo de operação é realizada somente dentro de funções encapsuladas que executam consultas espaciais. Por outro lado, os SAEs SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano possibilitam a obtenção direta dos resultados dos cálculos da área de polígonos

e do comprimento de linhas graças à adoção da biblioteca JTS. Ademais, outras operações numéricas – tal como o cálculo do número de pontos que compõem um objeto espacial – também são disponibilizadas por essa biblioteca. Já no sistema Hadoop-GIS é possível obter, além das áreas de polígonos distintos, os coeficientes Jaccard (JACCARD, 1901) e DICE (DICE, 1945), os quais mensuram a silimilaridade entre estas áreas. Por fim, o Magellan é o único SAE dentre os revisados que não provê nenhum tipo de suporte para esse tipo de operação.

Operações Atreladas aos Tipos de Dados. Um SAE pode fornecer operações cuja execução só pode ser realizada em objetos espaciais de um tipo específico. Por exemplo, a extração do MBR pode estar disponível somente para objetos do tipo polígono em certos sistemas. Outro exemplo pode ser descrito na obtenção da fronteira de um objeto espacial, operação que um SAE pode disponibilizar somente para objetos do tipo linha. Devido ao fato dos sistemas SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano serem baseados na biblioteca JTS, estes fornecem diversas operações atreladas aos tipos de dados, conforme evidenciado no Quadro 7. Exemplos incluem uma operação para verificar se um polígono é um retângulo e a obtenção do último ponto que compõe uma linha. Ademais, os sistemas Hadoop-GIS e SpatialHadoop fornecem suporte para pelo menos uma operação desse tipo, enquanto os SAEs Simba, LocationSpark e Magellan não implementam nenhuma dessas operações.

Quadro 8 – Comparação entre SAEs, considerando técnicas de junção espacial e estruturas de indexação espacial. Fonte: Elaborado pelo autor.

		Técnicas de Junção Espacial	
SAE	**Estruturas de Indexação Espacial**	**Sem Índices**	**Com Índices**
Hadoop-GIS	R*-*Tree*	✗	✓
SpatialHadoop	*Grid File*, R-*Tree*, R+-*Tree*	✓	✓
SpatialSpark	R-*Tree*	✗	✓
GeoSpark	R-*Tree*, *Quadtree*	✓	✓
GeoMesa Spark	Z-*Curve*, XZ-*Curve*	✗	✓
Simba	R-*Tree*, *Quadtree*	✓	✓
LocationSpark	*Grid File*, *Quadtree*, R-*Tree*	✗	✓
STARK	R-*Tree*	✗	✓
Magellan	Z-*Curve*	✗	✓
SparkGIS	R-*Tree*	✗	✓
Elcano	Não informado	Não informado	Não informado

Estruturas de Indexação Espacial. O uso de índices espaciais é uma das técnicas mais utilizadas para a acelerar o processamento de consultas espaciais (GAEDE; GÜNTHER, 1998). Diversos índices espaciais foram propostos na literatura, tais como as R-*Trees* e as *Quadtrees*. No geral, SAEs empregam índices espaciais com dois propósitos: (i) para processar consultas espaciais nos nós escravos; e (ii) para distribuir dados entre os nós escravos e possivelmente reduzir o número de partições acessadas durante o processamento de uma consulta espacial (seção 2.2.3). Devido às vantagens supracitadas, todos os SAEs revisados fornecem suporte para índices espaciais, conforme evidenciado no Quadro 8. Porém, somente os sistemas SpatialHadoop, GeoSpark, GeoMesa Spark, Simba e LocationSpark disponibilizam mais de uma estrutura

de indexação espacial. Os SAEs restantes são limitados em relação à flexibilidade de escolha de um índice espacial para atender às características específicas de uma aplicação.

Técnicas de Junção Espacial. Realizar a junção de dois conjuntos de dados espaciais é uma necessidade recorrente em aplicações espaciais. Conforme detalhado na seção 2.2.2, esse tipo de consulta difere de uma junção comum no que diz respeito à sua cláusula de junção, a qual inclui um predicado espacial. Dados dois conjuntos de dados espaciais, um exemplo desse tipo de consulta seria encontrar pares de objetos espaciais que intersectam entre si. Ademais, retornar pares de objetos espaciais que estão separados por uma certa distância também se enquadra como um exemplo. Junções espaciais são operações custosas, fato que leva à proposição de diversas técnicas otimizadas para sua execução. Essas técnicas podem exigir que pelo menos um dos conjuntos de dados envolvidos na junção esteja indexado, conforme evidenciado no Quadro 8. Diversos SAEs dentre os revisados fornecem suporte para a execução otimizada de junções espaciais com pelo menos um dos conjuntos de dados envolvidos na junção estando indexado. Por outro lado, somente os SAEs SpatialHadoop, GeoSpark e Simba provêem tal suporte quando nenhuma estrutura de indexação espacial é usada em nenhum dos conjuntos de dados envolvidos.

4.5 Aspectos Inerentes ao Ambiente Distribuído

Os aspectos inerentes ao ambiente distribuído comparados no Quadro 9 se referem a: abstração do conjunto de dados, paticionamento espacial, índices globais e locais e visualização espacial. A motivação e o contexto relacionados a cada característica são discutidos no decorrer da seção.

Abstração do Conjunto de Dados. *Frameworks* de processamento distribuído de dados podem manipular seu conjunto de dados por meio de abstrações. Essa estratégia contribui para a redução da dificuldade inerente ao processamento de consultas em ambientes distribuídos, uma vez que o usuário consegue realizar diversas operações sobre o conjunto de dados manipulando somente sua abstração (SHI *et al.*, 2015). Conforme detalhado na seção 2.3.2, os *frameworks* de processamento distribuído de dados implementam abstrações distintas: enquanto os usuários do Hadoop interagem com o conjunto de dados por meio de blocos do HDFS, no Spark os dados podem ser manipulados tanto via RDDs quanto via *Data Frames*. O Quadro 9 mostra que alguns SAEs implementam operações espaciais diretamente sobre as abstrações nativas de seus *frameworks* base, como é o caso dos sistemas Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoMesa Spark, Simba, Magellan e SparkGIS. Os SAEs restantes disponibilizam abstrações personalizadas, adicionando uma nova camada sobre o conjunto de dados espaciais a fim de fornecer ao usuário um conjunto de operações espaciais de forma encapsulada. Por exemplo, o STARK utiliza objetos do tipo ST*Object*, sendo cada instância composta por um objeto espacial e um componente temporal associado. Apesar das abstrações personalizadas simplificarem a interação do usuário com o conjunto de dados espaciais, é importante notar que seu uso pode acarretar em sobrecargas durante o processamento de consultas (BAIG *et al.*, 2017).

Quadro 9 – Comparação dos SAEs, considerando aspectos inerentes ao ambiente distribuído. Fonte: Elaborado pelo autor.

SAE	**Abstração do Conjunto de Dados**	**Particionamento Espacial**	**Índice Global**	**Índice Local**	**Visualização Espacial**
Hadoop-GIS	Bloco do HDFS	*Strip*, *Boundary Optimized Strip*, *Binary Split*, *Grid File*, *Quadtree*, Hilbert *Curve*, *Sort Tile Recursive*	✓	✓	Somente para as fronteiras das partições
SpatialHadoop	Bloco do HDFS	*Grid File*, *Quadtree*, *Sort Tile Recursive*, *Sort Tile Recursive*+, KD-*Tree*, Z-*Curve*, Hilbert *Curve*	✓	✓	Somente para objetos espaciais simples
SpatialSpark	Spark RDD	*Grid File*, *Binary Split*, *Sort Tile Recursive*	✗	✓	✗
GeoSpark	*Spatial*RDD, *Point*RDD, *LineString*RDD, *Polygon*RDD, SparkSQL *Data Frame*	KDB-*Tree*, *Quadtree*, R-*Tree*	✗	✓	Somente para objetos espaciais simples
GeoMesa Spark	Spark RDD, SparkSQL *Data Frame*	*Grid File*, R-*Tree*	✓	✓	✗
Simba	SparkSQL *Data Frame*	KD-*Tree*, *Quadtree*, *Sort Tile Recursive*, Voronoi *Diagram*	✓	✓	✗
LocationSpark	SpatialRDD	*Grid File*, *Region Quadtree*	✓	✓	✗
STARK	RDD de ST*Objects*	*Grid File*, *Binary Split*	✓	✓	Somente para pontos e polígonos simples
Magellan	SparkSQL *Data Frame*	✗	✗	✓	✗
SparkGIS	Spark RDD	*Grid File*, *Binary Split*, *Quadtree*, *Strip*, *Boundary Optimized Strip*, Hilbert *Curve*, *Sort Tile Recursive*	✓	✓	✗
Elcano	Não informado	Não informado	Não informado	Não informado	✗

Particionamento Espacial. O particionamento de dados entre os nós de um *cluster* é uma técnica frequentemente utilizada na computação paralela e distribuída. Com o uso dessa técnica, o processamento de consultas pode ter seu tempo reduzido ao tirar proveito da localização física dos dados. No contexto de dados espaciais, um SAE é capaz de utilizar as características espaciais de um conjunto de dados como critério para o particionamento, aprimorando, consequentemente, o desempenho de suas consultas espaciais. De acordo com o Quadro 9, diversos SAEs empregam técnicas de particionamento espacial, com os sistemas Hadoop-GIS, SpatialHadoop, Simba e SparkGIS disponibilizando uma quantidade considerável de algoritmos. Essa quantidade contribui para o enriquecimento da experiência do usuário, uma vez que garante flexibilidade na escolha de um algoritmo para atender às características específicas de uma aplicação espacial.

Índices Globais e Locais. O conceito de indexação espacial em ambientes paralelos e distribuídos está fortemente atrelado à ideia de particionamento espacial. Isso se deve ao fato de que os índices globais são criados com a mesma estrutura utilizada durante esse tipo de particionamento, conforme detalhado na seção 2.3.3. Porém, de acordo com o Quadro 9, nem todos os sistemas que disponibilizam algoritmos de particionamento espacial permitem o aproveitamento de suas estruturas como índices globais. Nesse caso, durante o processamento de consultas espaciais, os SAEs não eliminam as partições desnecessárias para a obtenção de seus resultados. Como alternativa, esses sistemas acessam todas as partições espaciais até que o resultado completo da consulta seja obtido, utilizando seus respectivos índices locais para a otimização do processamento. As estruturas de indexação espacial disponibilizadas por cada SAE para a criação de índices locais estão listadas na seção 4.4.

Visualização Espacial. A disponibilização de um módulo para visualizar os resultados da execução de uma consulta espacial contribui consideravelmente para o engrandecimento da experiência do usuário. Tal contribuição é decorrente do fato de que a interpretação dos resultados por parte de um usuário é facilitada quando estes resultados são visualizados em um mapa ao invés de em um arquivo de texto, permitindo que o processo de tomada de decisão seja mais ágil e preciso. Para que um sistema consiga prover uma visualização de objetos espaciais com completude, a geração de mapas de alta resolução é necessária. O Quadro 9 revela que somente os SAEs SpatialHadoop, GeoSpark e STARK disponibilizam suporte para essa funcionalidade. Porém, esse suporte possui algumas restrições. Enquanto os sistemas SpatialHadoop e GeoSpark permitem apenas a visualização de objetos espaciais simples, o módulo de visualização do STARK impõe uma restrição ainda maior, disponibilizando somente a visualização de pontos e polígonos simples. Ademais, o Hadoop-GIS fornece apenas um módulo básico que permite que seus usuários visualizem fronteiras de partições. Nesse sistema, uma fronteira de uma partição pode ser definida como sendo o MBR que contém todos os objetos espaciais localizados na partição. Dessa forma, sua visualização permite que o usuário compreenda como os dados estão organizados no ambiente distribuído. O restante dos SAEs revisados não disponibilizam nenhum módulo para a visualização distribuída de dados espaciais, sobrecarregando o usuário com a busca de *softwares* adicionais para este fim.

4.6 Considerações Finais

Neste capítulo, uma análise comparativa de diferentes SAEs foi introduzida. Essa análise foi desenvolvida sob o ponto de vista de seus usuários, vislumbrando três perspectivas que englobam diferentes critérios essenciais para a escolha desses sistemas: (i) características gerais; (ii) dados espaciais e operações relacionadas; e (iii) aspectos inerentes ao ambiente distribuído. Também foi apresentado um conjunto de quadros comparativos para cada perspectiva, com seus respectivos critérios sendo devidamente detalhados durante a comparação.

A partir dos resultados dessa análise, diversas diretrizes centradas no usuário foram desenvolvidas. Essas diretrizes, apresentadas no próximo capítulo, visam enriquecer o auxílio provido ao usuário durante o processo de seleção de um SAE.

CAPÍTULO

5

DIRETRIZES CENTRADAS NO USUÁRIO

5.1 Considerações Iniciais

Neste capítulo são propostas diretrizes centradas nos usuários que projetam, desenvolvem e implementam aplicações espaciais para organizações. Essas diretrizes são contextualizadas na seção 5.2 e descritas nas seções de 5.3 a 5.8. Por fim, na seção 5.9 são feitas as considerações finais do capítulo.

5.2 Contextualização

As diretrizes propostas no presente capítulo são projetadas com o intuito de auxiliar os usuários a identificar, com base na análise descrita no Capítulo 4, qual SAE é o mais apropriado para manipular um grande volume de dados espaciais em diferentes contextos. Como esses contextos são dependentes dos objetivos específicos de cada aplicação espacial, é comum que sua variabilidade seja considerável. Sendo assim, este capítulo apresenta uma categorização geral, a qual pode ser posteriormente especializada levando em consideração os requisitos intrínsecos de cada aplicação espacial. Ou seja, as diretrizes aqui apresentadas são baseadas nos principais recursos que aplicações que lidam com um grande volume de dados espaciais possam requerer.

Dessa forma, exemplos reais dessas aplicações são listados em cada uma das diretrizes propostas no presente capítulo, ilustrando contextos nos quais sua adoção seja propícia. Por fim, os SAEs que atendem às especificações de cada diretriz também são listados. O fato de um SAE não ser listado em uma determinada diretriz não implica que este não possua nenhuma relação com a mesma: é possível que alguma funcionalidade relacionada seja disponibilizada, porém com a exigência de maiores esforços de implementação por parte do usuário.

5.3 Diretriz 1: Foco em Consultas Espaciais *Ad-hoc*

Esta diretriz é baseada em aplicações que precisam processar consultas espaciais *ad-hoc*, ou seja, cujo formato não é predefinido. Para atender à Diretriz 1, um SAE deve fornecer suporte para uma variedade significativa de operações espaciais, tais como operações geométricas de conjunto, relacionamentos topológicos e operações numéricas. Caso contrário, a experiência do usuário durante a execução de uma consulta espacial *ad-hoc* pode ser prejudicada.

A utilidade da presente diretriz pode ser observada em diversas aplicações espaciais. Um exemplo é um sistema para monitoramento ambiental por meio da análise de dados espaciais distribuídos e heterogêneos (WIEMANN; KARRASCH; BERNARD, 2018), no qual a determinação das estruturas de lagos e rios é um exemplo de uma operação comum. Outro exemplo de sistema no qual a utilidade dessa diretriz pode ser evidenciada é o *Terrafly GeoCloud* (ZHANG *et al.*, 2015), uma aplicação *online* projetada para auxiliar usuários em suas análises por meio da execução de consultas espaciais de acordo com o contexto da base de dados espacial *TerraFly* (TERRAFLY, 2019). Ademais, o *Geographic Information System Querying Framework* (GISQF) (NAAMI; SEKER; KHAN, 2014) também pode ser descrito como sendo um exemplo capaz de ilustrar a aplicação da presente diretriz, uma vez que este sistema realiza diversas consultas analíticas no conjunto de dados *Global Data of Events, Language and Tone* (GDELT) (GDELT, 2019).

Com base nas análises descritas no Capítulo 4, pode-se concluir que os seguintes SAEs atendem à Diretriz 1: SpatialSpark, GeoSpark, GeoMesa Spark, STARK e Elcano. Já o Hadoop-GIS atende parcialmente a essa diretriz, uma vez que este sistema não provê suporte para as seguintes operações espaciais: (i) cálculo de comprimento de linha; (ii) cálculo da diferença entre dois objetos espaciais; e (iii) uso dos predicados topológicos *covers* e *covered by*.

5.4 Diretriz 2: Foco em Interoperabilidade

Esta diretriz é baseada em aplicações espaciais que precisam comunicar umas com as outras, tais como aquelas que integram dados espaciais heterogêneos a partir de diferentes provedores de informação. Para atender à Diretriz 2, um SAE deve prover suporte para todos os tipos de dados espaciais. Também é necessário que esse SAE forneça suporte para o intercâmbio de objetos espaciais nas formas textual e binária, visto que a representação de fenômenos espaciais pode se dar de forma distinta em diferentes fontes. Por exemplo, é possível que uma aplicação espacial represente um determinado fenômeno por meio de polígonos simples armazenados em formato textual, enquanto outra aplicação pode representá-lo usando polígonos complexos salvos em formato binário.

Um exemplo prático desse tipo de situação ocorre em uma aplicação desenvolvida para prover suporte para o planejamento urbano, na qual pontos de interesse de provedores de dados públicos e privados são integrados (SMARZARO; LIMA; DAVIS JÚNIOR, 2017).

Outro exemplo que pode ser destacado é o de um sistema que integra a localização geográfica de objetos em um ambiente de internet das coisas (MADAAN; AHAD; SASTRY, 2018). Ademais, a presente diretriz também pode ser utilizada por uma aplicação que integra dados espaciais para mapear fatores humanos e naturais em recifes de coral (WEDDING *et al.*, 2018).

Por meio das investigações descritas no Capítulo 4, é possível afirmar que os SAEs GeoSpark e GeoMesa Spark atendem completamente à Diretriz 2. Já os sistemas Hadoop-GIS, SpatialSpark e Elcano atendem parcialmente a essa diretriz, visto que estes fornecem suporte apenas para representações textuais de objetos espaciais. Um atendimento parcial também é observado para o Magellan: apesar deste sistema considerar representações textuais e binárias, nenhum suporte para objetos espaciais do tipo linha é disponibilizado.

5.5 Diretriz 3: Foco em Padrões Bem Conhecidos

Esta diretriz é baseada em aplicações espaciais que requerem o uso de conceitos bem conhecidos na literatura, tais como expressões e termos padronizados, linguagens de consulta e representações de objetos espaciais. A adoção desses padrões durante o desenvolvimento de uma aplicação espacial pode reduzir ambiguidades de forma significativa, tal como no caso de funções que possuem o mesmo nome, porém apresentam comportamentos distintos. Ademais, tal adoção pode reduzir a curva de aprendizado de um sistema, visto que os usuários que já estão familiarizados com padrões bem conhecidos vão encontrar menos barreiras durante o processo de aprendizado.

Para atender à Diretriz 3, um SAE deve empregar tais padrões em seu projeto. Também é necessário que o SAE satisfaça à Diretriz 2, uma vez que a interoperabilidade entre diferentes aplicações é uma consequência direta da adoção de padrões bem conhecidos. Um exemplo de uma situação em que essa diretriz pode ser empregada é a aplicação de padrões abertos para fornecer integração e interconexão entre sistemas na *web* (LEE; REICHARDT, 2005). Outros exemplos são um sistema de monitoramento ambiental que lida com a análise de dados espaciais heterogêneos e distribuídos (WIEMANN; KARRASCH; BERNARD, 2018) e a combinação de fontes de dados heterogêneas a fim de aumentar o nível de dados relacionados ao uso e à cobertura de terra na Europa (ROSINA *et al.*, 2018).

Com base nas análises descritas no Capítulo 4, os sistemas GeoSpark e GeoMesa Spark atendem à Diretriz 3. Os SAEs SpatialSpark, Simba, Magellan e Elcano também consideram diversos aspectos da presente diretriz, porém introduzem limitações relacionadas à falta de uma linguagem de consulta bem conhecida ou à falta de representações textuais e binárias bem conhecidas.

5.6 Diretriz 4: Foco em Visualização Espacial

Esta diretriz é baseada em aplicações que requerem a visualização de objetos espaciais de forma gráfica, permitindo que usuários enriqueçam o processo de tomada de decisão ao compartilhar descobertas por meio da plotagem de tais objetos em mapas reais. Para atender à Diretriz 4, um SAE deve disponibilizar um módulo de visualização que não imponha restrições relacionadas ao tipo de dado espacial sendo manipulado. Caso contrário, a experiência do usuário durante o processo de visualização pode ser comprometida.

Um exemplo de aplicação em que essa diretriz pode ser empregada é o Taghreed (MAGDY *et al.*, 2014), um sistema para consultar, analisar e visualizar *microblogs* georreferenciados. Esse tipo de módulo também é necessário para a visualização de dados de trânsito, visto que seu uso pode ocasionar uma melhora no entendimento de sistemas de transporte (CHEN; GUO; WANG, 2015). Outro exemplo é o *Ecosystem Services Partnership Visualization Tool* (ESP-VT) (DRAKOU *et al.*, 2015), uma ferramenta de visualização e compartilhamento de dados para mapas relacionados a ecossistemas.

No melhor do conhecimento do autor do presente trabalho, não existe um SAE dentre os revisados que atenda completamente à Diretriz 4. Os SAEs SpatialHadoop e GeoSpark atendem parcialmente a essa diretriz, visto que estes sistemas permitem a visualização de todos os tipos de objetos espaciais simples, porém não são capazes de visualizar objetos espaciais complexos.

5.7 Diretriz 5: Foco em Eficiência

Esta diretriz é baseada em aplicações que requerem o processamento eficiente de consultas espaciais. Apesar do termo eficiência ser frequentemente associado com reduções no tempo de processamento, um usuário pode considerar diversos outros critérios para mensurá-la. Por exemplo, a utilização de memória e o espaço ocupado por índices podem ser ferramentas poderosas para a comparação da eficiência de SAEs.

Para atender à Diretriz 5, um SAE deve fornecer algoritmos especializados e otimizados para o processamento de consultas espaciais, preferencialmente com o uso de índices. O SAE em questão também deve se sobressair nas comparações centradas em desempenho disponíveis na literatura, ou seja, em estudos cujo foco reside em contrastar a eficiência de diferentes SAEs no processamento de consultas espaciais.

No escopo dessas comparações, o trabalho de Pandey *et al.* (2018) pode ser considerado estado da arte, uma vez que seus resultados são baseados em um volume considerável de experimentos que envolvem uma quantidade expressiva de SAEs. Porém, como sistemas baseados no *framework* Hadoop não são considerados nesses experimentos, o trabalho de Eldawy e Mokbel (2015) pode ser considerado estado da arte para comparações cujo escopo considere apenas SAEs baseados em Hadoop.

Um exemplo da adoção da presente diretriz pode ser encontrado no desenvolvimento de algoritmos destinados a otimizar consultas de junção espacial baseadas em distância (GARCÍA-GARCÍA *et al.*, 2017). Outro exemplo refere-se ao algoritmo descrito em Du *et al.* (2017), o qual utiliza o Spark para processar junções espaciais multidirecionais de forma eficiente. Pode-se citar ainda como exemplo um algoritmo para otimizar o desempenho de junções espaciais baseadas na intersecção de polilinhas em *clusters* acelerados por *Graphics Processing Units* (GPUs) (YOU; ZHANG; GRUENWALD, 2016).

Desde que a realização de uma comparação centrada em desempenho está fora do escopo do presente trabalho, as conclusões apresentadas nesta diretriz se fundamentam nos resultados dos testes de desempenho descritos nos trabalhos de Pandey *et al.* (2018) e Eldawy e Mokbel (2015). Com base nas comparações descritas em Pandey *et al.* (2018), é possível afirmar que o GeoSpark é o SAE mais adequado para atender à Diretriz 5, uma vez que este oferece os melhores resultados gerais de desempenho. Entretanto, se o escopo da presente diretriz for limitado aos sistemas baseados em Hadoop, é possível afirmar que o SpatialHadoop cumpre esta diretriz com base nas descobertas descritas em Eldawy e Mokbel (2015).

5.8 Diretriz 6: Foco em Extensibilidade

Esta diretriz é baseada em aplicações espaciais que requerem meios para facilitar a implementação de extensões pelo usuário. Estender uma aplicação espacial pode ser um esforço necessário para solucionar um problema relacionado ao seu contexto, porém que não foi considerado previamente em sua implementação original. Caso a aplicação seja baseada em um SAE, suas características intrínsecas podem interferir no processo de extensão. Por exemplo, se o SAE for implementado em uma linguagem de programação na qual o usuário não possua domínio, estendê-lo pode se provar uma tarefa custosa. Outro empecilho pode ser observado caso o SAE possua alguma limitação legal em relação à sua modificação e à sua distribuição.

Sendo assim, para atender à Diretriz 6 um SAE deve possuir características que evitem a ocorrência desses empecilhos. Ou seja, o SAE deve: (i) ser desenvolvido em uma linguagem de programação com popularidade relevante (PYPL, 2019); (ii) adotar uma licença que permita sua modificação e redistribuição; (iii) ser baseado em uma biblioteca espacial capaz de prover suporte para diversos aspectos inerentes à manipulação de dados espaciais; (iv) disponibilizar uma documentação completa e atualizada; e (v) prover múltiplos meios de se obter suporte de sua comunidade.

Um exemplo de aplicação da presente diretriz pode ser encontrado nos sistemas *Routes4People* e Melhor Busão (ALIC *et al.*, 2018). Enquanto o *Routes4People* é uma aplicação baseada em Spark que disponibiliza informações relacionadas à melhor rota em um mapa considerando uma série de critérios, o Melhor Busão estende esta aplicação por meio da adição de uma funcionalidade que prevê o próximo ônibus que irá passar por uma área específica. Outro

exemplo é a implementação de uma extensão para executar consultas KNN no GISQF (NAAMI; SEKER; KHAN, 2014), um sistema cujas consultas retornam eventos associados aos objetos espaciais do conjunto de dados GDELT. Ademais, disponibilizar a execução de *range queries* baseadas em distância no SHAHEED (ELDAWY *et al.*, 2015), uma aplicação para consultar e visualizar dados de satélite do *Land Process Distributed Active Archive Center*, pode ser considerado como outro exemplo.

Por meio das investigações descritas no Capítulo 4, os SAEs GeoSpark e GeoMesa Spark são os únicos dentre os revisados que atendem completamente à Diretriz 6. O SpatialHadoop atende parcialmente a essa diretriz, apresentando apenas uma deficiência relacionada à não adoção de uma biblioteca espacial bem conhecida na literatura. Porém, esse sistema compensa parte das desvantagens associadas a tal deficiência por meio da disponibilização de uma interface que permite que seus usuários implementem seus próprios tipos de dados espaciais. Por fim, o STARK também considera diversos aspectos da presente diretriz, porém não a atende totalmente por não prover uma documentação completa.

5.9 Considerações Finais

Neste capítulo foram apresentadas diversas diretrizes centradas no usuário, desenvolvidas com o intuito de auxiliá-lo durante o processo de escolha de um SAE. Para tal, cada diretriz foi baseada em uma funcionalidade comum em aplicações espaciais reais, com sua aplicabilidade sendo exemplificada por meio da descrição de algumas dessas aplicações. No Quadro 10 é exibida a conformidade dos SAEs em relação às diretrizes propostas.

Quadro 10 – Conformidade dos SAEs com as diretrizes apresentadas. Fonte: Elaborado pelo autor.

SAE	Diretriz 1	Diretriz 2	Diretriz 3	Diretriz 4	Diretriz 5	Diretriz 6
Hadoop-GIS	Parcial	Parcial	✗	✗	✗	✗
SpatialHadoop	✗	✗	✗	Parcial	✓[1]	Parcial
SpatialSpark	✓	Parcial	Parcial	✗	✗	✗
GeoSpark	✓	✓	✓	Parcial	✓	✓
GeoMesa Spark	✓	✓	✓	✗	✗	✓
Simba	✗	✗	Parcial	✗	✗	✗
LocationSpark	✗	✗	✗	✗	✗	✗
STARK	✓	✗	✗	✗	✗	Parcial
Magellan	✗	Parcial	Parcial	✗	✗	✗
SparkGIS	✗	✗	✗	✗	✗	✗
Elcano	✓	Parcial	Parcial	✗	✗	✗

[1] Somente no escopo de SAEs baseados em Hadoop.

No próximo capítulo são ilustrados dois estudos de caso que descrevem requisitos de duas aplicações espaciais distintas, as quais são baseadas em situações e em conjuntos de dados provenientes do mundo real. Nesse contexto, as diretrizes propostas são utilizadas a fim de selecionar o SAE ideal para atender aos requisitos de tais aplicações.

CAPÍTULO

6

ESTUDOS DE CASO

6.1 Considerações Iniciais

Neste capítulo são apresentados dois estudos de caso, os quais descrevem aplicações cujo objetivo reside no processamento de conjuntos de dados provenientes do mundo real. Na seção 6.2 esses estudos de caso são contextualizados. Na seção 6.3 a primeira aplicação é introduzida, a qual é baseada em dados do *OpenStreetMaps*. Na seção 6.4 uma aplicação baseada em dados do Twitter é descrita. Por fim, na seção 6.5 são feitas as considerações finais do capítulo.

6.2 Contextualização

Os estudos de caso apresentados neste capítulo possuem como objetivo a ilustração da aplicabilidade das diretrizes propostas no Capítulo 5. Essa aplicabilidade está relacionada à utilização dessas diretrizes durante o processo de escolha do SAE mais apropriado para implementar uma determinada aplicação espacial. Dessa forma, cada estudo de caso descreve uma aplicação baseada em situações e em conjuntos de dados provenientes do mundo real. Para cada aplicação, os seguintes aspectos são detalhados: (i) seus requisitos; (ii) as diretrizes utilizadas para escolher o SAE ideal para sua implementação; e (iii) os procedimentos necessários para a carga de dados e para a execução de consultas espaciais no SAE selecionado.

Para a implementação dessas aplicações, um ambiente de *cluster* de computadores foi utilizado. Esse *cluster* é composto por um total de seis máquinas pertencentes à Profa. Dra. Cristina Dutra de Aguiar Ciferri, orientadora do presente trabalho. Essas máquinas dispõem, conjuntamente, de cerca de 64 *gigabytes* de memória principal e 4 *terabytes* de memória secundária. Além disso, estão instaladas no *cluster* as versões mais recentes dos SAEs selecionados nos estudos de caso e as versões compatíveis de seus respectivos *frameworks* base, conforme especificado na seção 4.3.

6.3 Aplicação Baseada em Dados do *OpenStreetMaps*

O *OpenStreetMaps* (HAKLAY; WEBER, 2008) faz parte de uma iniciativa colaborativa cujo objetivo reside em disponibilizar publicamente dados geográficos globais. Para esse fim, a *Open Database License* (ODBL, 2019) é adotada como sua licença padrão. Várias aplicações fazem uso de conjuntos de dados mantidos pelo *OpenStreetMaps*, tais como os mapas do *Customer Data Research Centre* (CDRC) (VIJ, 2016), os quais fazem parte de um sistema interativo para visualizar associações entre dados espaciais e dados relacionados a clientes.

Levando em consideração a real aplicabilidade do sistema de mapas do CDRC, uma aplicação baseada em objetos espaciais reais extraídos do *OpenStreetMaps* em 16 de janeiro de 2017 (CARNIEL; CIFERRI; CIFERRI, 2017b) é proposta. Tais objetos correspondem a 1.486.557 edifícios, 2.644.432 rodovias e 770.842 localidades únicas do Brasil, representados por polígonos, linhas e pontos, respectivamente. A aplicação proposta lida com esses objetos em consultas espaciais que analisam a situação da infraestrutura de diferentes locais, tais como fazendas, escolas e estradas.

Os usuários que projetam, desenvolvem e implementam a presente aplicação espacial devem considerar os seguintes requisitos para decidir qual SAE escolher. A aplicação deve gerenciar objetos espaciais representados por tipos de dados simples e complexos, os quais estão armazenados em arquivos CSV. Cada linha desses arquivos possui o formato (*geo*, *desc*), onde *geo* é a representação WKT do objeto espacial e *desc* é sua respectiva descrição. A aplicação também deve fornecer suporte para consultas espaciais *ad-hoc*, as quais podem conter operações de conjunto geométrico, relacionamentos topológicos e operações numéricas. Ademais, a aplicação deve fornecer bons resultados de desempenho. Por fim, uma informação importante a ser levada em consideração é que os usuários dessa aplicação possuem algum nível de conhecimento prévio acerca da linguagem SQL.

Os requisitos da presente aplicação indicam que as Diretrizes 1, 3 e 5 (Capítulo 5) devem ser levadas em consideração para a escolha do SAE a ser utilizado. Como resultado, os usuários devem escolher o GeoSpark como sendo o SAE mais adequado para implementar a aplicação, uma vez que este sistema atende aos requisitos supracitados. Em relação à Diretriz 5, os usuários devem aplicar os mesmos valores dos parâmetros descritos em Pandey *et al.* (2018) para garantir os melhores tempos de processamento para suas consultas espaciais. Esses parâmetros são: (i) a R-*Tree* como índice local; e (ii) a *Quadtree* como técnica de particionamento espacial. Como o GeoSpark adota esses parâmetros como padrão, nenhum esforço adicional de configuração é necessário por parte do usuário.

O restante desta seção está organizado da seguinte forma. Na seção 6.3.1 são descritos procedimentos relacionados ao carregamento de dados no GeoSpark. Por fim, na seção 6.3.2 são descritos exemplos de consultas espaciais para a aplicação.

6.3.1 Carregamento dos Dados

Inicialmente, os objetos espaciais contidos nos arquivos CSV são carregados no GeoSpark por meio do GeoSparkSQL. Como esse módulo é uma extensão do SparkSQL, os arquivos são carregados em uma estrutura denominada *DataFrame*, a qual se assemelha a uma tabela relacional. Em seguida, a coluna que armazena a representação textual dos dados espaciais deve ser transformada em uma coluna espacial. O GeoSparkSQL fornece uma função capaz de transformar uma representação WKT em um objeto espacial, o qual pode ser manipulado em consultas SQL. A consulta descrita no Código-fonte 1 é um exemplo de como a coluna que contém a representação WKT dos polígonos armazenados em *brazil_buildings* pode ser transformada em uma coluna espacial.

Código-fonte 1 – Carregamento dos dados no GeoSpark. Fonte: Elaborado pelo autor.

```
1: SELECT ST_GeomFromWKT(brazil_buildings.geo) AS geo,
2:        brazil_buildings.desc AS desc
3: FROM brazil_buildings
```

6.3.2 Consultas

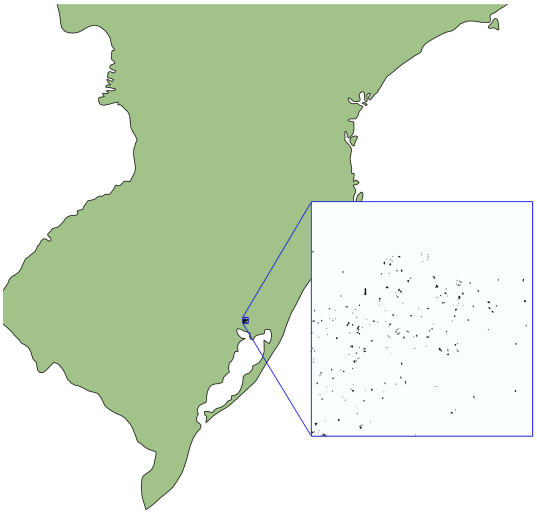
Assim que o carregamento de todos os objetos espaciais estiver concluído, os usuários podem executar consultas no *DataFrame*. São definidas três consultas espaciais como exemplo para a presente aplicação. Essas consultas são descritas nos Códigos-fonte de 2 a 4, sendo seus respectivos resultados ilustrados nas Figuras de 9 a 11. O *software* QGIS (QGIS, 2019) é utilizado para a visualização desses resultados, uma vez que o módulo GeoSparkViz (YU; ZHANG; SARWAT, 2018) não permite a visualização de objetos espaciais complexos. Ademais, uma aproximação é aplicada nas figuras a fim de melhor visualizar partes importantes dos resultados, visto que suas proporções não permitem sua exibição completa.

***Range query*.** A primeira consulta retorna todas as escolas localizadas dentro de uma janela de consulta QW, a qual é representada por um objeto de formato retangular correspondente a 0,001% da extensão total do Brasil. Os resultados da consulta, ilustrados na Figura 9, permitem que os usuários analisem a quantidade de escolas presente em uma determinada região. O comando que expressa a consulta é descrito no Código-fonte 2.

Código-fonte 2 – Execução da *range query* no GeoSpark. Fonte: Elaborado pelo autor.

```
1: SELECT brazil_buildings.geo
2: FROM brazil_buildings
3: WHERE ST_Contains(QW, brazil_buildings.geo)
4:       AND brazil_buildings.desc = 'school'
```

Figura 9 – Execução da *range query* no GeoSpark. Fonte: Elaborada pelo autor.

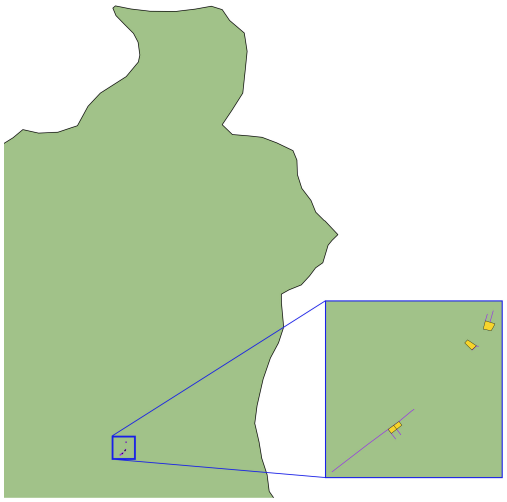


Junção espacial. A segunda consulta retorna todas as estradas irregulares utilizadas por veículos agrícolas que intersectam com uma construção. Sua utilidade está relacionada à investigação da necessidade de melhoria dessas estradas. Na Figura 10 são ilustrados os resultados do processamento da consulta. O comando que expressa a consulta é descrito no Código-fonte 3.

Código-fonte 3 – Execução da junção espacial no GeoSpark. Fonte: Elaborado pelo autor.

```
1: SELECT brazil_buildings.geo, brazil_highways.geo
2: FROM brazil_buildings, brazil_highways
3: WHERE ST_Intersects(brazil_buildings.geo, brazil_highways.geo)
4:       AND brazil_highways.desc = 'track'
```

Figura 10 – Execução da junção espacial no GeoSpark. Como o resultado completo engloba todo o território brasileiro, apenas uma fatia do mesmo é exibida. Fonte: Elaborada pelo autor.

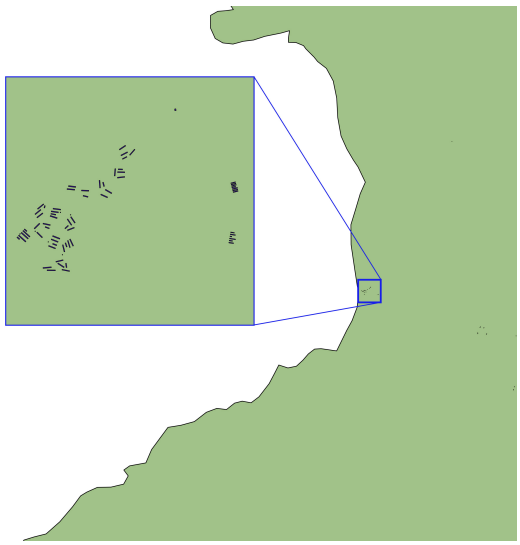


Manipulação geométrica de conjuntos. A terceira consulta retorna a área total de todas as fazendas do Brasil. É necessário primeiro computar a união geométrica de tais fazendas, ilustrada na Figura 11, para que então seja possível realizar o cálculo da área resultante. O comando que expressa a consulta é descrito no Código-fonte 4.

Código-fonte 4 – Execução da consulta de manipulação geométrica de conjuntos no GeoSpark. Fonte: Elaborado pelo autor.

```
1: SELECT ST_Area(ST_Union_Aggr(brazil_buildings.geo))
2: FROM brazil_buildings
3: WHERE brazil_buildings.desc = 'farm'
```

Figura 11 – Execução da consulta de manipulação geométrica de conjuntos no GeoSpark. Como o resultado completo engloba todo o território brasileiro, apenas uma fatia do mesmo é exibida. Fonte: Elaborada pelo autor.



6.4 Aplicação Baseada em Dados do Twitter

Vários estudos na literatura usam dados do Twitter (TWITTER, 2019) para identificar aspectos inerentes a localizações geográficas específicas, tais como os assuntos mais discutidos pelos usuários, ou até mesmo seus respectivos estilos de escrita (PAVALANATHAN; EISENSTEIN, 2015). Esses estudos podem se basear nas coordenadas geográficas anexadas a cada *tweet* (ou seja, a cada mensagem publicada no Twitter) ou na localidade informada no perfil do usuário. Nesse contexto, aplicações como o Taghreed (MAGDY *et al.*, 2014) emergem, simplificando para o usuário final o processo de consulta, análise e visualização de *tweets* com identificação geográfica.

No presente estudo de caso, uma aplicação similar é proposta, cujo objetivo reside na análise de um conjunto de *tweets* publicados em três cidades diferentes no período de 7 de

novembro de 2016 a 28 de fevereiro de 2017 (ANDRADE *et al.*, 2018). O número de *tweets* com identificação geográfica varia para cada cidade: (i) 891.367 *tweets* de São Paulo, Brasil; (ii) 849.026 *tweets* do Rio de Janeiro, Brasil; e (iii) 68.884 *tweets* de Medellín, Colômbia. A presente aplicação lida com tais *tweets* em consultas espaciais que analisam suas respectivas localizações geográficas em relação a diferentes referências, tais como localidades únicas ou regiões de uma determinada cidade.

Os requisitos para selecionar o SAE a ser empregado no desenvolvimento da presente aplicação são descritos a seguir. A aplicação deve ser capaz de gerenciar *tweets* representados por objetos espaciais do tipo ponto. Esses objetos estão armazenados em arquivos CSV, sendo que cada uma de suas linhas contém apenas as coordenadas X e Y do *tweet*, separadas por vírgula. Ademais, a aplicação requer o processamento eficiente de consultas espaciais para obter bons resultados de desempenho. Outrossim, essa aplicação necessita de ser atualizada constantemente, fato que pode exigir a implementação de extensões no SAE. Por fim, é considerado que o usuário possui apenas um ambiente Hadoop disponível, com vários sistemas legados em funcionamento que precisam se comunicar com a aplicação sendo proposta.

Levando em consideração os requisitos descritos, os usuários devem utilizar as diretrizes 5 e 6 (Capítulo 5) para selecionar o SAE mais apropriado para a implementação da presente aplicação. Ademais, os usuários também devem limitar sua escolha para que a mesma considere apenas sistemas baseados em Hadoop, devido às restrições de ambiente descritas nos requisitos da aplicação. Esses fatos levam o usuário a escolher o SpatialHadoop, uma vez que este sistema atende à maior parte dos critérios especificados nas diretrizes supracitadas.

O restante desta seção está organizado da seguinte forma. Na seção 6.4.1 são descritos procedimentos relacionados ao carregamento de dados no SpatialHadoop. Por fim, na seção 6.4.2 são descritos exemplos de consultas espaciais para a aplicação.

6.4.1 Carregamento dos Dados

O primeiro passo da presente aplicação consiste no carregamento no HDFS dos arquivos CSV que contêm os objetos espaciais a serem analisados. Essa tarefa pode ser realizada por meio da execução de uma sequência de comandos *put*, conforme descrito no Código-fonte 5. Cada comando *put* se refere ao carregamento dos dados de uma localidade considerada na aplicação.

Código-fonte 5 – Carregamento dos dados no SpatialHadoop. Fonte: Elaborado pelo autor.

```
1: hdfs dfs -put saopaulo.csv
2: hdfs dfs -put medellin.csv
3: hdfs dfs -put riodejaneiro.csv
```

Assim que os dados estiverem armazenados no HDFS, sua indexação pode ser realizada. Essa etapa, embora não seja obrigatória, é recomendada para acelerar significativamente o

processamento de consultas espaciais (seção 2.2.3). Para esse fim, é necessário utilizar o comando *index*. Nesse comando, o usuário deve especificar os seguintes parâmetros: (i) o arquivo de entrada; (ii) o diretório de saída no qual o índice será armazenado; (iii) o tipo de dado espacial dos objetos contidos no arquivo de entrada; e (iv) a estrutura de indexação espacial a ser criada. Todo o processo pode ser expresso por meio da sequência de comandos descrita no Código-fonte 6. Nesses comandos, a R-*Tree* é utilizada para indexar todos os conjuntos de dados. A escolha dessa estrutura é motivada pelos experimentos descritos em Eldawy e Mokbel (2015), os quais empregam a R-*Tree* para a indexação de conjuntos de dados compostos exclusivamente por objetos do tipo ponto.

Código-fonte 6 – Indexação dos dados no SpatialHadoop. Fonte: Elaborado pelo autor.

```
1: shadoop index saopaulo.csv sp-index/ shape:point sindex:rtree
2: shadoop index medellin.csv medellin-index/ shape:point
3:         sindex:rtree
4: shadoop index riodejaneiro.csv rj-index/ shape:point
5:         sindex:rtree
```

6.4.2 Consultas

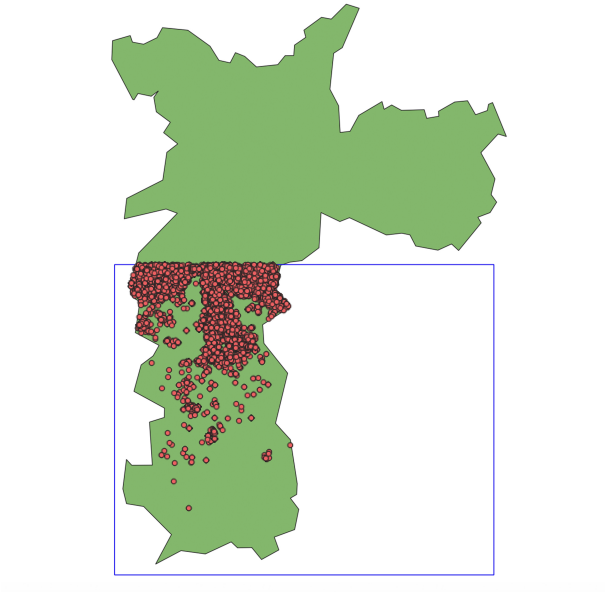
Após a conclusão do processo de carregamento dos objetos espaciais, os usuários podem executar consultas nos arquivos de dados. São definidas três consultas espaciais como exemplo para a presente aplicação. Essas consultas são descritas nos Códigos-fonte de 7 a 9, sendo seus respectivos resultados ilustrados nas Figuras de 12 a 14. O *software* QGIS (QGIS, 2019) é utilizado para a visualização desses resultados, uma vez que o módulo HadoopViz (ELDAWY; MOKBEL; JONATHAN, 2016) não disponibiliza meios amigáveis para que o usuário gere imagens contendo múltiplas camadas de objetos espaciais. Ademais, uma aproximação é aplicada na Figura 14 a fim de permitir uma melhor visualização dos resultados.

***Range query*.** A primeira consulta analisa a concentração de *tweets* em uma região específica. São retornados todos os *tweets* que se sobrepõem a uma janela de consulta que cobre a região sul da cidade de São Paulo (QW), conforme ilustrado na Figura 12. A consulta pode ser expressa com o comando *rangequery*. Nesse comando, o usuário deve especificar: (i) o arquivo de entrada; (ii) o arquivo de saída; (iii) o tipo de dado espacial dos objetos contidos no arquivo de entrada; e (iv) uma janela de consulta retangular. O comando resultante pode ser expresso conforme descrito no Código-fonte 7.

Código-fonte 7 – Execução da *range query* no SpatialHadoop. Fonte: Elaborado pelo autor.

```
1: shadoop rangequery saopaulo.csv result-sp.csv shape:point
2:         rect:QW
```

Figura 12 – Resultado da execução da *range query* no SpatialHadoop. Fonte: Elaborada pelo autor.

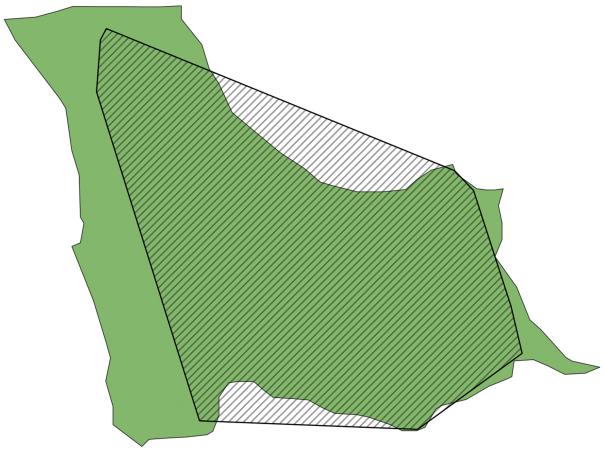


***Convex hull query*.** A segunda consulta retorna o menor polígono convexo (ou seja, o *convex hull*) que contém todos os *tweets* da cidade de Medellín. Esse tipo de consulta é útil para analisar em quais áreas da cidade nenhum *tweet* foi publicado, conforme evidenciado na Figura 13. O comando *convexhull* pode ser utilizado para expressar a consulta, sendo necessário que o usuário especifique o arquivo de entrada e o diretório de saída, assim como descrito no Código-fonte 8.

Código-fonte 8 – Execução da *convex hull query* no SpatialHadoop. Fonte: Elaborado pelo autor.

```
1: shadoop convexhull medellin.csv result-medellin/
```

Figura 13 – Resultado da execução da *convex hull query* no SpatialHadoop. Fonte: Elaborada pelo autor.

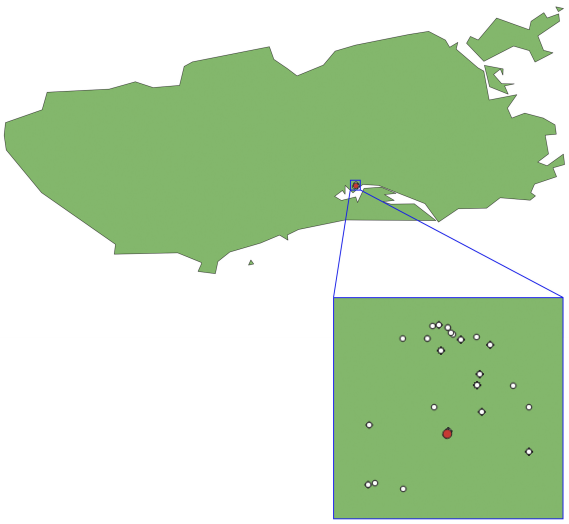


K-*nearest neighbours query*. A terceira consulta analisa a concentração de *tweets* em torno de uma importante atração turística. São retornados os 200 *tweets* cuja localização seja a mais próxima possível da Arena Olímpica do Rio de Janeiro (AO), assim como ilustrado na Figura 14. Para executar a consulta, o comando *knn* deve ser utilizado. Nesse comando, o usuário deve especificar: (i) o arquivo de entrada; (ii) o arquivo de saída; (iii) o tipo de dado espacial dos objetos contidos no arquivo de entrada; (iv) o ponto de referência a partir do qual os *tweets* são buscados; e (v) a quantidade de *tweets* a serem recuperados. O comando resultante pode ser expresso conforme descrito no Código-fonte 9.

Código-fonte 9 – Execução da k-*nearest neighbours query* no SpatialHadoop. Fonte: Elaborado pelo autor.

```
1: shadoop knn riodejaneiro.csv result-rj.csv shape:point
2:          point:AO k:200
```

Figura 14 – Resultado da execução da k-*nearest neighbours query* no SpatialHadoop. Fonte: Elaborada pelo autor.



6.5 Considerações Finais

No presente capítulo foram apresentados dois estudos de caso com o objetivo de exemplificar o emprego das diretrizes propostas no Capítulo 5. Duas aplicações foram descritas, utilizando como base contextos e conjuntos de dados provenientes do mundo real: (i) uma aplicação para analisar a situação da infraestrutura de diferentes locais, baseada em dados do *OpenStreetMaps*; e (ii) uma aplicação para análise da localização geográfica de publicações em relação a diferentes referências, baseada em dados do Twitter. Em seguida, diversas diretrizes foram utilizadas no processo de seleção do SAE ideal para cada aplicação. Por fim, um conjunto de consultas distinto

foi apresentado em cada estudo de caso, com seus respectivos comandos e resultados sendo devidamente ilustrados.

A análise dos estudos de caso descritos no presente capítulo demonstra a relevância das diretrizes propostas. Por meio de sua utilização, foi possível determinar o SAE ideal para duas aplicações com requisitos e conjuntos de dados distintos. Sendo assim, tais diretrizes representam uma solução robusta para usuários de SAEs, uma vez que facilitam e agilizam o processo de seleção desses sistemas para a implementação de diferentes tipos de aplicações espaciais.

No próximo capítulo são apresentadas as conclusões do trabalho desenvolvido, as quais englobam uma análise das tendências cronológicas e das limitações dos SAEs revisados. As contribuições do presente trabalho e os trabalhos futuros também são descritos.

CAPÍTULO

7

CONCLUSÃO

7.1 Considerações Iniciais

Este capítulo descreve as conclusões finais do presente trabalho. Na seção 7.2 está sumarizada a conclusão geral, a qual engloba uma análise de tendências cronológicas dos SAEs e a descrição de suas principais limitações. Na seção 7.3 são destacadas as contribuições obtidas pelo presente trabalho, enquanto na seção 7.4 são listadas as dificuldades encontradas durante seu desenvolvimento. Os artigos científicos produzidos durante o período de execução do trabalho são listados na seção 7.5. Por fim, na seção 7.6 são descritas possibilidades de trabalhos futuros.

7.2 Conclusão Geral

O presente trabalho tem como principal objetivo realizar uma comparação qualitativa de SAEs baseados em Hadoop e Spark sob a perspectiva dos usuários que projetam, desenvolvem e implementam aplicações espaciais para organizações. Essa comparação visa ajudá-los a compreender as características desses sistemas, a fim de torná-los capazes de escolher o sistema ideal para o desenvolvimento de suas aplicações espaciais. Em outras palavras, o presente trabalho foi motivado pela seguinte hipótese: "***É possível assistir o usuário, por meio de uma análise comparativa das principais características dos SAEs, no processo de escolha dos sistemas que atendam aos requisitos específicos de aplicações espaciais distintas***".

Para a investigação de tal hipótese, além do fornecimento da análise comparativa supradescrita, também foi proposto um conjunto de diretrizes centradas no usuário. Essas diretrizes, por sua vez, tiveram sua aplicabilidade comprovada por meio de dois estudos de caso baseados em contextos e em conjuntos de dados provenientes do mundo real. Por fim, essa investigação permitiu a identificação de uma série de tendências cronológicas e limitações dos SAEs baseados em Hadoop e Spark existentes na literatura. Essas tendências e limitações são discutidas nas seções 7.2.1 e 7.2.2, respectivamente.

7.2.1 Tendências Cronológicas dos Sistemas Analíticos Espaciais

Uma característica de um SAE pode ser considerada uma tendência cronológica quando é constantemente adotada por novos sistemas à medida que estes surgem. A análise dessas tendências possui uma importância significativa para usuários de SAEs, uma vez que permite a previsão das principais características dos sistemas que ainda não foram lançados. Além disso, os desenvolvedores de SAEs também podem se beneficiar desse conteúdo durante o processo de planejamento de novos recursos para esses sistemas. Diversas tendências cronológicas foram identificadas a partir dos SAEs revisados, as quais são descritas a seguir.

A primeira tendência está relacionada aos *frameworks* base empregados pelos SAEs. Após o surgimento do primeiro sistema baseado em Spark em 2015, nenhum outro SAE foi desenvolvido utilizando o Hadoop como base. Conforme discutido no Capítulo 4, essa ocorrência pode ser justificada devido ao desempenho superior normalmente apresentado pelo Spark, principalmente graças ao fato de seu processamento de dados ser realizado majoritariamente em memória principal. Além de se basearem no Spark, SAEs lançados em 2015 ou após este ano também tendem a ser desenvolvidos usando a linguagem de programação Scala, possivelmente devido ao aumento de sua popularidade na última década (PYPL, 2019).

As estratégias de indexação distribuída dos SAEs também estão associadas a uma tendência cronológica. Essa tendência pode ser observada no emprego da indexação global, a qual se tornou um paradigma para sistemas baseados em Spark após a introdução do GeoMesa Spark em 2015. Uma possível justificativa para essa ocorrência se baseia no fato de os índices globais estarem associados a uma redução em custos de comunicação de rede, conforme discutido no Capítulo 4. Essa redução, no entanto, não está necessariamente relacionada a ganhos de desempenho: é possível que um SAE que não empregue índices globais obtenha resultados superiores em comparações de desempenho. Esse é o caso do GeoSpark, conforme destacado na Diretriz 5 (Capítulo 5).

No contexto das técnicas de junção espacial dos SAEs, uma tendência adicional pode ser observada. Após o lançamento do LocationSpark em 2016, todos os SAEs subsequentes empregam apenas técnicas de junção espacial que exigem a indexação de pelo menos um dos conjuntos de dados envolvidos na junção. Considerando que as junções espaciais são operações custosas, esse favorecimento de técnicas dependentes de indexação pode ser associado aos ganhos de desempenho geralmente fornecidos pelo emprego de índices na execução de consultas espaciais, conforme descrito no Capítulo 2.

Outra tendência cronológica relevante está relacionada ao emprego de padrões bem conhecidos. Após o surgimento do GeoSpark em 2015, a maioria dos SAEs subsequentes adotam uma linguagem de consulta baseada em SQL. Uma tendência similar pode ser observada para bibliotecas espaciais bem conhecidas. Metade dos sucessores do SpatialSpark, os quais foram lançados em 2015 ou após este ano, emprega a biblioteca JTS, a qual é baseada nos padrões definidos pela OGC. De acordo com as descobertas descritas no Capítulo 4, a justificativa de

tais tendências é evidente: o emprego de padrões bem conhecidos geralmente gera uma redução na curva de aprendizado do usuário e aprimora a interoperabilidade entre aplicações e sistemas distintos.

7.2.2 O que falta em um Sistema Analítico Espacial?

Várias limitações podem ser observadas nos SAEs revisados, as quais podem comprometer a experiência do usuário em diferentes aspectos. Por exemplo, nenhum desses sistemas provê suporte para a execução de relacionamentos direcionais e para a visualização de objetos espaciais complexos. Por conseguinte, se um usuário necessitar de alguma dessas operações para o desenvolvimento de uma aplicação espacial, ele ficará sobrecarregado com a tarefa de procurar um *software* adicional para seu processamento.

O suporte para a visualização de objetos espaciais complexos não é a única limitação identificada em relação aos módulos de visualização dos SAEs. Problemas relacionados à experiência do usuário durante a manipulação desses módulos também foram identificados, uma vez que várias etapas e parâmetros podem ser necessários para a produção de uma única imagem. Ademais, as imagens produzidas são significativamente menos detalhadas do que aquelas geradas por outros *softwares* de visualização disponíveis no mercado. Imagens de baixa qualidade podem impactar negativamente a experiência do usuário, uma vez que geram uma diminuição na capacidade de percepção do mesmo durante o processo de tomada de decisão.

Por fim, a dificuldade de obtenção de suporte por meio dos canais de comunicação disponibilizados pelos desenvolvedores dos SAEs também pode ser listada como sendo uma limitação. Com exceção do GeoSpark e do GeoMesa Spark, todos os sistemas revisados no presente trabalho fornecem no máximo dois canais ativos de suporte. Além disso, os desenvolvedores podem levar vários dias para responder às perguntas dos usuários, fato que pode comprometer consideravelmente o processo de desenvolvimento.

7.3 Contribuições

No presente trabalho, diversas contribuições foram obtidas, as quais geraram um avanço significativo no estado da arte. Primeiro, uma análise comparativa dos seguintes SAEs baseados em Hadoop e Spark foi desenvolvida: Hadoop-GIS, SpatialHadoop, SpatialSpark, GeoSpark, GeoMesa Spark, Simba, LocationSpark, STARK, Magellan, SparkGIS e Elcano. Como essa análise foi realizada com base em uma perspectiva centrada no usuário, seu objetivo consiste em ajudar os usuários a entender como as características dos SAEs são úteis para atender aos requisitos específicos de suas aplicações espaciais. As comparações realizadas no presente trabalho foram fundamentadas por um extenso conjunto de critérios que integram três perspectivas diferentes.

A primeira dessas perspectivas se refere às características gerais dos SAEs, as quais englobam ano de surgimento, *framework* base, versão do *framework* base, versão do SAE, licença, biblioteca espacial, linguagem de programação, linguagem de consulta, completude da documentação e suporte da comunidade. A segunda perspectiva se baseia nas características referentes à manipulação de dados espaciais, as quais incluem suporte para tipos de dados espaciais, representações de objetos espaciais, operações geométricas de conjunto, relacionamentos direcionais, relacionamentos métricos, relacionamentos topológicos, operações numéricas, operações atreladas aos tipos de dados, estruturas de indexação espacial e técnicas de junção espacial. A última perspectiva se baseia nos aspectos inerentes aos sistemas distribuídos, os quais incluem abstração do conjunto de dados, paticionamento espacial, índices globais, índices locais e visualização espacial.

Com base nas comparações centradas no usuário, foi proposto um conjunto de diretrizes para ajudá-los a escolher um SAE apropriado para projetar, desenvolver e implementar suas aplicações espaciais. Como o contexto e os objetivos por trás do desenvolvimento dessas aplicações são muito variáveis, uma caracterização geral foi fornecida, a qual pode ser posteriormente especializada de acordo com os requisitos de cada aplicação. As diretrizes propostas concentram-se na execução de consultas espaciais *ad-hoc*, na interoperabilidade entre diferentes sistemas, no uso de padrões bem conhecidos, na necessidade de visualização de dados espaciais, no processamento eficiente de consultas espaciais e na extensibilidade dos SAEs.

Também foram apresentados dois estudos de caso, um utilizando o GeoSpark e outro empregando o SpatialHadoop, a fim de comprovar a aplicabilidade das diretrizes propostas. Esses estudos de caso descrevem aplicações cujos requisitos são baseados em problemas do mundo real e cujos conjuntos de dados são extraídos das fontes *OpenStreetMaps* e Twitter, respectivamente. Para cada estudo de caso, o carregamento e a indexação de dados foram exemplificados, a execução de três consultas espaciais de interesse foi descrita e a visualização dos resultados dessas consultas foi realizada.

Por fim, uma discussão das tendências cronológicas relacionadas a SAEs foi fornecida, assim como a identificação das limitações que esses sistemas devem abordar para melhorar a experiência do usuário. Com base no ano de surgimento dos SAEs, foram destacadas tendências relacionadas a *framework* base, linguagens de programação, estratégias de indexação distribuída, técnicas de junção espacial e emprego de padrões e bibliotecas espaciais bem conhecidos. Com relação às limitações, foram identificadas a falta de relacionamentos direcionais e da visualização de objetos espaciais complexos, a qualidade limitada dos módulos de visualização e a dificuldade de obtenção de suporte por parte do usuário.

7.4 Dificuldades Encontradas

As seguintes dificuldades foram encontradas durante o desenvolvimento deste trabalho:

- Levantamento dos dados necessários para a comparação centrada no usuário apresentada no Capítulo 4. Muitos dados não estavam disponíveis nas documentações dos sistemas, exigindo maiores investigações em suas respectivas implementações. Ademais, foram observados casos em que informações advindas de diferentes fontes se provaram conflitantes. Nesses casos, a informação extraída do código fonte do SAE foi priorizada.
- Instalação dos *frameworks* Hadoop e Spark em um ambiente de *cluster* de computadores. Foi necessário um estudo extenso da arquitetura desses *frameworks* – assim como diversas tentativas de configuração – para que sua execução se tornasse possível.
- Utilização dos módulos de visualização dos SAEs SpatialHadoop e GeoSpark para exibir os resultados das consultas desenvolvidas nos estudos de caso (Capítulo 6). Conforme descrito na seção 7.2.2, esses módulos possuem uma complexidade considerável, uma vez que podem demandar várias etapas e parâmetros para a produção de uma única imagem.

7.5 Divulgação de Resultados

Durante o desenvolvimento deste trabalho, os seguintes artigos científicos foram produzidos:

- Artigo intitulado "***A User-Centric View of Distributed Spatial Data Management Systems***" (CASTRO; CARNIEL; CIFERRI, 2018), publicado no XIX *Brazilian Symposium on Geoinformatics* (GEOINFO), 2018. O conteúdo desse artigo refere-se a uma versão inicial do presente trabalho.
- Artigo intitulado "***Analyzing Spatial Analytics Systems Based on Hadoop and Spark: A User Perspective***", submetido para a revista científica *Computers & Geosciences*. O conteúdo desse artigo representa uma síntese do presente trabalho.

7.6 Trabalhos Futuros

Trabalhos futuros relacionados ao presente trabalho incluem:

- Coletar dados acerca da aceitação das diretrizes propostas no Capítulo 5 por usuários que fazem uso de SAEs para projetar, desenvolver e implementar aplicações espaciais para organizações. Essa coleta pode ser realizada por meio da realização de entrevistas e da aplicação de questionários.

- Implementar extensões para SAEs capazes de prover soluções para as limitações evidenciadas na seção 7.2.2. Um exemplo é a implementação de um módulo que permita a execução de consultas espaciais que envolvam relacionamentos direcionais nesses sistemas.
- Selecionar um SAE, com base em sua popularidade no mercado e na literatura, a fim de implementar cada característica descrita no Capítulo 4 não suportada por este sistema.
- Realizar comparações centradas em desempenho assim que as primeiras versões públicas dos sistemas SparkGIS e Elcano forem lançadas, uma vez que estas não são consideradas no estudo de Pandey *et al.* (2018).
- Realizar comparações centradas no usuário que englobem SAEs baseados em Hadoop e Spark capazes de representar dados espaciais por meio do modelo *raster*, como o *Physical Analytics Integrated Repository and Services* (PAIRS) (KLEIN *et al.*, 2015).
- Realizar comparações centradas no usuário que englobem SAEs baseados em outros *frameworks* de processamento paralelo e distribuído de dados, como o Apache Ignite (APACHE, 2019b).

REFERÊNCIAS

AJI, A.; WANG, F.; VO, H.; LEE, R.; LIU, Q.; ZHANG, X.; SALTZ, J. H. Hadoop-GIS: A high performance spatial data warehousing system over MapReduce. **Proceedings of the VLDB Endowment**, v. 6, n. 11, p. 1009–1020, 2013. Citado nas páginas 20 e 40.

ALAM, M. M.; RAY, S.; BHAVSAR, V. C. A performance study of big spatial data systems. In: **Proceedings of the 7th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data**. New York, NY, USA: ACM, 2018. p. 1–9. Citado nas páginas 41 e 43.

ALIC, A. S.; ALMEIDA, J.; MEIRA JÚNIOR, W.; GUEDES, D.; SANTOS, W.; BLANQUER, I.; FIORE, S.; KOZIEVITCH, N. P.; ANDRADE, N.; BRAZ, T.; BRITO, A.; PIRES, C. E.; ANTUNES, N.; VIEIRA, M.; SILVA, P.; ARDAGNA, D.; FONSECA, K.; LEZZI, D.; ELIA, D.; MORAES, R.; BASSO, T.; CAVASSIN, W. H. GIS and Data: Three applications to enhance mobility. In: **Proceedings of the XIX Brazilian Symposium on Geoinformatics**. Campina Grande, PB, Brazil: MCTIC/INPE, 2018. p. 1–12. Citado na página 65.

ANDRADE, S. C.; DEGROSSI, L. C.; RESTREPO-ESTRADA, C.; DELBEM, A. C. B.; ALBUQUERQUE, J. P. Does keyword noise change over space and time? A case study of social media messages. In: **Proceedings of the XIX Brazilian Symposium on Geoinformatics**. Campina Grande, PB, Brazil: MCTIC/INPE, 2018. p. 116–121. Citado na página 72.

APACHE. **Apache Hadoop**. 2019. Disponível em: <https://hadoop.apache.org/>. Acesso em: 31 de julho de 2019. Citado nas páginas 19, 30 e 32.

____. **Apache Ignite**. 2019. Disponível em: <https://ignite.apache.org/>. Acesso em: 31 de julho de 2019. Citado na página 82.

____. **Apache License: Version 2.0**. 2019. Disponível em: <https://www.apache.org/licenses/LICENSE-2.0>. Acesso em: 31 de julho de 2019. Citado na página 47.

____. **Apache Spark**. 2019. Disponível em: <https://spark.apache.org/>. Acesso em: 31 de julho de 2019. Citado nas páginas 19, 30 e 32.

ARMBRUST, M.; FOX, A.; GRIFFITH, R.; JOSEPH, A. D.; KATZ, R. H.; KONWINSKI, A.; LEE, G.; PATTERSON, D. A.; RABKIN, A.; STOICA, I.; ZAHARIA, M. **Above the clouds: A Berkeley view of cloud computing**. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2009. Citado na página 32.

ARMBRUST, M.; XIN, R. S.; LIAN, C.; HUAI, Y.; LIU, D.; BRADLEY, J. K.; MENG, X.; KAFTAN, T.; FRANKLIN, M. J.; GHODSI, A.; ZAHARIA, M. Spark SQL: Relational data processing in Spark. In: **ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2015. p. 1383–1394. Citado nas páginas 20 e 34.

AURENHAMMER, F. Voronoi Diagrams – A survey of a fundamental geometric data structure. **ACM Computing Surveys**, v. 23, n. 3, p. 345–405, 1991. Citado na página 36.

BAIG, F.; VO, H.; KURÇ, T. M.; SALTZ, J. H.; WANG, F. SparkGIS: Resource aware efficient in-memory spatial query processing. In: **ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2017. p. 28:1–28:10. Citado nas páginas 19, 20, 40 e 56.

BECKMANN, N.; KRIEGEL, H.-P.; SCHNEIDER, R.; SEEGER, B. The R*-Tree: An efficient and robust access method for points and rectangles. In: **Proceedings of the ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1990. p. 322–331. Citado nas páginas 29 e 37.

BENTLEY, J. L. Multidimensional binary search trees used for associative searching. **Communications of the ACM**, v. 18, n. 9, p. 509–517, 1975. Citado na página 36.

BHOWMIK, S. **Cloud Computing**. Cambridge, UK: Cambridge University Press, 2017. Citado na página 32.

BÖHM, C.; KREBS, F. The K-Nearest Neighbour Join: Turbo charging the KDD process. **Knowledge and Information Systems**, v. 6, n. 6, p. 728–749, 2004. Citado na página 28.

BÖXHM, C.; KLUMP, G.; KRIEGEL, H.-P. XZ-Ordering: A space-filling curve for objects with spatial extension. In: **Advances in Spatial Databases**. Berlin/Heidelberg, Germany: Springer International Publishing, 1999. p. 75–90. Citado na página 37.

BUTLER, H.; DALY, M.; DOYLE, A.; GILLIES, S.; HAGEN, S.; SCHAUB, T. **The GeoJSON Format**. RFC 7946, 2016. 1–28 p. Citado na página 51.

CARNIEL, A. C.; CIFERRI, R. R.; CIFERRI, C. D. A. Analyzing the performance of spatial indices on hard disk drives and flash-based solid state drives. **Journal of Information and Data Management**, v. 8, n. 1, p. 34–49, 2017. Citado na página 28.

____. Spatial datasets for conducting experimental evaluations of spatial indices. In: **Satellite Events of the Brazilian Symposium on Databases - Dataset Showcase Workshop**. Uberlândia, MG, Brazil: SBC, 2017. p. 286–295. Citado na página 68.

CASTRO, J. P. C.; CARNIEL, A. C.; CIFERRI, C. D. A. A user-centric view of distributed spatial data management systems. In: **Proceedings of the XIX Brazilian Symposium on Geoinformatics**. Campina Grande, PB, Brazil: MCTIC/INPE, 2018. p. 80–91. Citado nas páginas 22 e 81.

CHEN, M.; MAO, S.; LIU, Y. Big Data: A survey. **Mobile Networks and Applications**, v. 19, n. 2, p. 171–209, 2014. Citado na página 30.

CHEN, W.; GUO, F.; WANG, F. A survey of traffic data visualization. **IEEE Transactions on Intelligent Transportation Systems**, v. 16, n. 6, p. 2970–2984, 2015. Citado na página 64.

CIFERRI, R. R.; SALGADO, A. C. B. **Análise da Influência do Fator Distribuição Espacial dos Dados no Desempenho de Métodos de Acesso Multidimensionais**. Tese (Doutorado) — Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Pernambuco, 2002. Citado nas páginas 25 e 27.

DEAN, J.; GHEMAWAT, S. MapReduce: Simplified data processing on large clusters. **Communications of the ACM**, v. 51, n. 1, p. 107–113, 2008. Citado nas páginas 20 e 32.

DERBEKO, P.; DOLEV, S.; GUDES, E.; SHARMA, S. Security and privacy aspects in MapReduce on clouds: A survey. **Computer Science Review**, v. 20, p. 1–28, 2016. Citado nas páginas 32 e 33.

DICE, L. R. Measures of the amount of ecologic association between species. **Ecology**, v. 26, n. 3, p. 297–302, 1945. Citado na página 55.

DRAKOU, E.; CROSSMAN, N.; WILLEMEN, L.; BURKHARD, B.; PALOMO, I.; MAES, J.; PEEDELL, S. A visualization and data-sharing tool for ecosystem service maps: Lessons learnt, challenges and the way forward. **Ecosystem Services**, v. 13, p. 134–140, 2015. Citado na página 64.

DU, Z.; ZHAO, X.; YE, X.; ZHOU, J.; ZHANG, F.; LIU, R. An effective high-performance multiway spatial join algorithm with Spark. **ISPRS International Journal of Geo-Information**, v. 6, n. 4, p. 96:1–96:13, 2017. Citado na página 65.

ELDAWY, A.; ALARABI, L.; MOKBEL, M. F. Spatial partitioning techniques in SpatialHadoop. **Proceedings of the VLDB Endowment**, v. 8, n. 12, p. 1602–1605, 2015. Citado nas páginas 20 e 37.

ELDAWY, A.; LI, Y.; MOKBEL, M. F.; JANARDAN, R. CG_Hadoop: Computational geometry in MapReduce. In: **Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2013. p. 294–303. Citado na página 28.

ELDAWY, A.; MOKBEL, M. F. A demonstration of SpatialHadoop: An efficient MapReduce framework for spatial data. **Proceedings of the VLDB Endowment**, v. 6, n. 12, p. 1230–1233, 2013. Citado na página 20.

____. Pigeon: A spatial MapReduce language. In: **International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 2014. p. 1242–1245. Citado na página 49.

____. SpatialHadoop: A MapReduce framework for spatial data. In: **International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 2015. p. 1352–1363. Citado nas páginas 20, 40, 64, 65 e 73.

ELDAWY, A.; MOKBEL, M. F.; ALHARTHI, S.; ALZAIDY, A.; TAREK, K.; GHANI, S. SHAHED: A MapReduce-based system for querying and visualizing spatio-temporal satellite data. In: **International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 2015. p. 1585–1596. Citado na página 66.

ELDAWY, A.; MOKBEL, M. F.; JONATHAN, C. HadoopViz: A MapReduce framework for extensible visualization of big spatial data. In: **International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 2016. p. 601–612. Citado nas páginas 20 e 73.

ENGÉLINUS, J.; BADARD, T. Elcano: A geospatial big data processing system based on SparkSQL. In: **International Conference on Geographical Information Systems Theory, Applications and Management**. Funchal, Madeira, Portugal: SCITEPRESS, 2018. p. 119–128. Citado nas páginas 20 e 40.

FACEBOOK. **Home Page**. 2019. Disponível em: <https://facebook.com/>. Acesso em: 31 de julho de 2019. Citado na página 50.

FINKEL, R. A.; BENTLEY, J. L. Quadtrees: A data structure for retrieval on composite keys. **Acta Informatica**, v. 4, n. 1, p. 1–9, 1974. Citado na página 36.

GAEDE, V.; GÜNTHER, O. Multidimensional access methods. **ACM Computing Surveys**, v. 30, n. 2, p. 170–231, 1998. Citado nas páginas 25, 27, 28, 29, 30 e 55.

GARCÍA-GARCÍA, F.; CORRAL, A.; IRIBARNE, L.; MAVROMMATIS, G.; VASSILAKOPOULOS, M. A comparison of distributed spatial data management systems for processing distance join queries. In: **European Conference on Advances in Databases and Information Systems**. Berlin/Heidelberg, Germany: Springer International Publishing, 2017. p. 214–228. Citado nas páginas 19, 35, 36, 40, 45 e 65.

GDELT. **Global Data of Events, Language and Tone**. 2019. Disponível em: <https://www.gdeltproject.org/data.html>. Acesso em: 31 de julho de 2019. Citado na página 62.

GEOMESA. **User Manual**. 2019. Disponível em: <https://www.geomesa.org/documentation/user/index.html>. Acesso em: 31 de julho de 2019. Citado na página 20.

GEOSPARK. **Home Page**. 2019. Disponível em: <http://geospark.datasyslab.org/>. Acesso em: 31 de julho de 2019. Citado na página 20.

GNU. **General Public License: Version 2**. 2019. Disponível em: <https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html>. Acesso em: 31 de julho de 2019. Citado na página 47.

GÜTING, R. H. An introduction to spatial database systems. **The VLDB Journal**, v. 3, n. 4, p. 357–399, 1994. Citado nas páginas 20, 23, 39, 43, 50, 51 e 54.

GUTTMAN, A. R-Trees: A dynamic index structure for spatial searching. In: **ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1984. p. 47–57. Citado nas páginas 28, 29, 36 e 51.

HADOOP-GIS. **Home Page**. 2019. Disponível em: <http://bmidb.cs.stonybrook.edu/hadoopgis/index>. Acesso em: 31 de julho de 2019. Citado na página 20.

HAGEDORN, S.; GÖTZE, P.; SATTLER, K. Big spatial data processing frameworks: Feature and performance evaluation. In: **International Conference on Extending Database Technology**. Venice, Italy: OpenProceedings, 2017. p. 490–493. Citado na página 41.

____. The STARK framework for spatio-temporal data analytics on Spark. In: **Datenbanksysteme für Business, Technologie und Web**. Stuttgart, Germany: Gesellschaft für Informatik, 2017. p. 123–142. Citado nas páginas 20 e 40.

HAGEDORN, S.; SATTLER, K. Piglet: Interactive and platform transparent analytics for RDF & dynamic data. In: **Proceedings of the 25th International Conference Companion on World Wide Web**. Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2016. p. 187–190. Citado na página 49.

HAKLAY, M.; WEBER, P. OpenStreetMap: User-generated street maps. **IEEE Pervasive Computing**, v. 7, n. 4, p. 12–18, 2008. Citado nas páginas 51 e 68.

HECHT, R.; JABLONSKI, S. NoSQL evaluation: A use case oriented survey. In: **International Conference on Cloud and Service Computing**. Piscataway, NJ, USA: IEEE, 2011. p. 336–341. Citado na página 30.

HUGHES, J. N.; ANNEX, A.; EICHELBERGER, C. N.; FOX, A.; HULBERT, A.; RONQUEST, M. Geomesa: A distributed architecture for spatio-temporal fusion. In: **SPIE Defense + Security**. Baltimore, MD, USA: SPIE, 2015. p. 94730F:1–94730F:12. Citado nas páginas 20 e 40.

HWANG, K.; DONGARRA, J.; FOX, G. C. **Distributed and cloud computing: From parallel processing to the Internet of Things**. Burlington, Massachusetts, USA: Morgan Kaufmann, 2013. Citado na página 31.

JACCARD, P. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. **Bull Soc Vaudoise Sci Nat**, v. 37, p. 547–579, 1901. Citado na página 55.

JACOX, E. H.; SAMET, H. Spatial join techniques. **ACM Transactions on Database Systems**, v. 32, n. 1, p. 7:1–7:44, 2007. Citado na página 27.

JTS. **Java Topology Suite**. 2019. Disponível em: <https://locationtech.github.io/jts/>. Acesso em: 31 de julho de 2019. Citado na página 48.

KLEIN, L. J.; MARIANNO, F. J.; ALBRECHT, C. M.; FREITAG, M.; LU, S.; HINDS, N.; SHAO, X.; RODRIGUEZ, S. B.; HAMANN, H. F. PAIRS: A scalable geo-spatial data analytics platform. In: **IEEE International Conference on Big Data**. Piscataway, NJ, USA: IEEE, 2015. p. 1290–1298. Citado na página 82.

LAWDER, J. K.; KING, P. J. H. Querying multi-dimensional data indexed using the Hilbert space-filling curve. **ACM SIGMOD Record**, v. 30, n. 1, p. 19–24, 2001. Citado na página 36.

LEE, K. B.; REICHARDT, M. E. Open standards for homeland security sensor networks. **IEEE Instrumentation Measurement Magazine**, v. 8, n. 5, p. 14–21, 2005. Citado nas páginas 20 e 63.

LEUTENEGGER, S. T.; LOPEZ, M. A.; EDGINGTON, J. STR: A simple and efficient algorithm for R-Tree packing. In: **Proceedings of the 13th International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 1997. p. 497–506. Citado na página 36.

LI, M.; TAN, J.; WANG, Y.; ZHANG, L.; SALAPURA, V. SparkBench: A Spark benchmarking suite characterizing large-scale in-memory data analytics. **Cluster Computing**, v. 20, n. 3, p. 2575–2589, 2017. Citado na página 33.

LI, Y.; MANOHARAN, S. A performance comparison of SQL and NoSQL databases. In: **IEEE Pacific Rim Conference on Communications, Computers and Signal Processing**. Piscataway, NJ, USA: IEEE, 2013. p. 15–19. Citado na página 30.

LOCATIONSPARK. **Home Page**. 2019. Disponível em: <https://github.com/purduedb/LocationSpark>. Acesso em: 31 de julho de 2019. Citado na página 20.

MADAAN, N.; AHAD, M. A.; SASTRY, S. M. Data integration in IoT ecosystem: Information linkage as a privacy threat. **Computer Law Security Review**, v. 34, n. 1, p. 125–133, 2018. Citado na página 63.

MAGDY, A.; ALARABI, L.; AL-HARTHI, S.; MUSLEH, M.; GHANEM, T. M.; GHANI, S.; MOKBEL, M. F. Taghreed: A system for querying, analyzing, and visualizing geotagged microblogs. In: **Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2014. p. 163–172. Citado nas páginas 64 e 71.

MAGELLAN. **Home Page**. 2019. Disponível em: <https://github.com/harsha2010/magellan>. Acesso em: 31 de julho de 2019. Citado na página 20.

MELL, P.; GRANCE, T. **The NIST definition of cloud computing**. Computer Security Division, Information Technology Laboratory, National Institution of Standards and Technology, 2011. Citado na página 32.

MORAIS, T. S. Survey on frameworks for distributed computing: Hadoop, Spark and Storm. In: **Proceedings of the 10th Doctoral Symposium in Informatics Engineering**. Porto, Portugal: Faculdade de Engenharia da Universidade do Porto, 2015. p. 95–105. Citado na página 34.

NAAMI, K. M. A.; SEKER, S.; KHAN, L. GISQF: An efficient spatial query processing system. In: **2014 IEEE 7th International Conference on Cloud Computing**. Piscataway, NJ, USA: IEEE, 2014. p. 681–688. Citado nas páginas 62 e 66.

NIEVERGELT, J.; HINTERBERGER, H.; SEVCIK, K. C. The grid file: An adaptable, symmetric multikey file structure. **ACM Transactions on Database Systems**, v. 9, n. 1, p. 38–71, 1984. Citado na página 36.

ODBL. **Open Data Commons: Open Database License**. 2019. Disponível em: <https://opendatacommons.org/licenses/odbl/>. Acesso em: 31 de julho de 2019. Citado na página 68.

OGC. **OpenGIS® Implementation Standard for Geographic Information - Simple Feature Access - Part 1: Common Architecture**. 2019. Disponível em: <http://www.opengeospatial.org/standards/sfa>. Acesso em: 31 de julho de 2019. Citado nas páginas 20, 25, 39, 47, 50 e 51.

ORENSTEIN, J. A.; MERRETT, T. H. A class of data structures for associative searching. In: **Proceedings of the 3rd ACM SIGACT-SIGMOD Symposium on Principles of Database Systems**. New York, NY, USA: ACM, 1984. p. 181–190. Citado na página 36.

PANDEY, V.; KIPF, A.; NEUMANN, T.; KEMPER, A. How good are modern spatial analytics systems? **Proceedings of the VLDB Endowment**, v. 11, n. 11, p. 1661–1673, 2018. Citado nas páginas 19, 20, 21, 31, 35, 36, 41, 45, 47, 64, 65, 68 e 82.

PAVALANATHAN, U.; EISENSTEIN, J. Confounds and consequences in geotagged Twitter data. In: **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**. Lisbon, Portugal: The Association for Computational Linguistics, 2015. p. 2138–2148. Citado na página 71.

PEUQUET, D. J.; CI-XIANG, Z. An algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. **Pattern Recognition**, v. 20, n. 1, p. 65–74, 1987. Citado nas páginas 25 e 52.

PYPL. **Popularity of Programming Language Index**. 2019. Disponível em: <http://pypl.github.io/PYPL.html>. Acesso em: 31 de julho de 2019. Citado nas páginas 65 e 78.

QGIS. **A Free and Open Source Geographic Information System**. 2019. Disponível em: <https://qgis.org/en/site/>. Acesso em: 31 de julho de 2019. Citado nas páginas 69 e 73.

ROBINSON, J. T. The K-D-B-Tree: A search structure for large multidimensional dynamic indexes. In: **Proceedings of the ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1981. p. 10–18. Citado na página 36.

ROSINA, K.; SILVA, F. B.; VIZCAINO, P.; HERRERA, M. M.; FREIRE, S.; SCHIAVINA, M. Increasing the detail of european land use/cover data by combining heterogeneous data sets. **International Journal of Digital Earth**, v. 0, n. 0, p. 1–25, 2018. Citado na página 63.

SADASHIV, N.; KUMAR, S. M. D. Cluster, grid and cloud computing: A detailed comparison. In: **6th International Conference on Computer Science Education**. Piscataway, NJ, USA: IEEE, 2011. p. 477–482. Citado nas páginas 31 e 32.

SAMET, H. The Quadtree and related hierarchical data structures. **ACM Computing Surveys**, v. 16, n. 2, p. 187–260, 1984. Citado na página 36.

SELLIS, T.; ROUSSOPOULOS, N.; FALOUTSOS, C. The R+-Tree: A dynamic index for multi-dimensional objects. In: **Proceedings of the 13th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann, 1987. p. 507–518. Citado na página 37.

SHI, J.; QIU, Y.; MINHAS, U. F.; JIAO, L.; WANG, C.; REINWALD, B.; ÖZCAN, F. Clash of the titans: MapReduce vs. Spark for large scale data analytics. **Proceedings of the VLDB Endowment**, v. 8, n. 13, p. 2110–2121, 2015. Citado nas páginas 33 e 56.

SHVACHKO, K.; KUANG, H.; RADIA, S.; CHANSLER, R. The Hadoop Distributed File System. In: **Proceedings of the IEEE 26th Symposium on Mass Storage Systems and Technologies**. Piscataway, NJ, USA: IEEE, 2010. p. 1–10. Citado nas páginas 34 e 35.

SIMBA. **Home Page**. 2019. Disponível em: <http://www.cs.utah.edu/~dongx/simba/>. Acesso em: 31 de julho de 2019. Citado na página 20.

SIQUEIRA, T. L. L.; CIFERRI, C. D. A.; TIMES, V. C.; CIFERRI, R. R. The SB-Index and the HSB-Index: Efficient indices for spatial data warehouses. **GeoInformatica**, v. 16, n. 1, p. 165–205, 2012. Citado na página 29.

SMARZARO, R.; LIMA, T. F. M.; DAVIS JÚNIOR, C. A. Could data from location-based social networks be used to support urban planning? In: **Proceedings of the 26th International Conference on World Wide Web Companion**. Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2017. p. 1463–1468. Citado na página 62.

SPATIALHADOOP. **Home Page**. 2019. Disponível em: <http://spatialhadoop.cs.umn.edu/>. Acesso em: 31 de julho de 2019. Citado na página 20.

SPATIALSPARK. **Home Page**. 2019. Disponível em: <https://github.com/syoummer/SpatialSpark>. Acesso em: 31 de julho de 2019. Citado na página 20.

STARK. **Home Page**. 2019. Disponível em: <https://github.com/dbis-ilm/stark>. Acesso em: 31 de julho de 2019. Citado na página 20.

STEINIGER, S.; BOCHER, E. An overview on current free and open source desktop GIS developments. **International Journal of Geographical Information Science**, v. 23, n. 10, p. 1345–1370, 2009. Citado na página 47.

TANG, M.; YU, Y.; MALLUHI, Q. M.; OUZZANI, M.; AREF, W. G. LocationSpark: A distributed in-memory data management system for big spatial data. **Proceedings of the VLDB Endowment**, v. 9, n. 13, p. 1565–1568, 2016. Citado nas páginas 20 e 40.

TERRAFLY. **Geospatial Big Data Platform and Solutions**. 2019. Disponível em: <http://terrafly.com/>. Acesso em: 31 de julho de 2019. Citado na página 62.

THUSOO, A.; SARMA, J. S.; JAIN, N.; SHAO, Z.; CHAKKA, P.; ZHANG, N.; ANTHONY, S.; LIU, H.; MURTHY, R. Hive – A petabyte scale data warehouse using Hadoop. In: **International Conference on Data Engineering**. Piscataway, NJ, USA: IEEE, 2010. p. 996–1005. Citado na página 49.

TWITTER. **Home Page**. 2019. Disponível em: <https://twitter.com/>. Acesso em: 31 de julho de 2019. Citado nas páginas 50 e 71.

UNIDATA. **NetCDF: Introduction and Overview**. 2019. Disponível em: <https://www.unidata.ucar.edu/software/netcdf/docs/index.html>. Acesso em: 31 de julho de 2019. Citado na página 51.

VAISMAN, A.; ZIMÁNYI, E. **Data Warehouse Systems: Design and Implementation**. 1st. ed. Berlin/Heidelberg, Germany: Springer International Publishing, 2014. Citado nas páginas 23, 24, 25 e 26.

VIJ, N. Introducing the consumer data research centre (CDRC). **Journal of Direct, Data and Digital Marketing Practice**, v. 17, n. 4, p. 232–235, 2016. Citado na página 68.

VO, H.; AJI, A.; WANG, F. SATO: A spatial data partitioning framework for scalable query processing. In: **Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2014. p. 545–548. Citado na página 36.

WEDDING, L. M.; LECKY, J.; GOVE, J. M.; WALECKA, H. R.; DONOVAN, M. K.; WILLIAMS, G. J.; JOUFFRAY, J.-B.; CROWDER, L. B.; ERICKSON, A.; FALINSKI, K.; FRIEDLANDER, A. M.; KAPPEL, C. V.; KITTINGER, J. N.; MCCOY, K.; NORSTRÖM, A.; NYSTRÖM, M.; OLESON, K. L. L.; STAMOULIS, K. A.; WHITE, C.; SELKOE, K. A. Advancing the integration of spatial data to map human and natural drivers on coral reefs. **PLOS ONE**, v. 13, n. 3, p. 1–29, 2018. Citado na página 63.

WIEMANN, S.; KARRASCH, P.; BERNARD, L. Ad-hoc combination and analysis of heterogeneous and distributed spatial data for environmental monitoring – design and prototype of a web-based solution. **International Journal Digital Earth**, v. 11, n. 1, p. 79–94, 2018. Citado nas páginas 20, 62 e 63.

XIE, D.; LI, F.; YAO, B.; LI, G.; ZHOU, L.; GUO, M. Simba: Efficient in-memory spatial analytics. In: **ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 2016. p. 1071–1085. Citado nas páginas 20 e 40.

YOU, S.; ZHANG, J.; GRUENWALD, L. Large-scale spatial join query processing in cloud. In: **International Conference on Data Engineering Workshops**. Piscataway, NJ, USA: IEEE, 2015. p. 34–41. Citado nas páginas 20 e 40.

____. High-performance polyline intersection based spatial join on GPU-accelerated clusters. In: **Proceedings of the 5th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data**. New York, NY, USA: ACM, 2016. p. 42–49. Citado na página 65.

YU, J.; WU, J.; SARWAT, M. GeoSpark: A cluster computing framework for processing large-scale spatial data. In: **ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2015. p. 70:1–70:4. Citado nas páginas 20 e 40.

YU, J.; ZHANG, Z.; SARWAT, M. GeoSparkViz: A scalable geospatial data visualization framework in the Apache Spark ecosystem. In: **Proceedings of the 30th International Conference on Scientific and Statistical Database Management**. New York, NY, USA: ACM, 2018. p. 15:1–15:12. Citado nas páginas 20 e 69.

____. Spatial data management in Apache Spark: The GeoSpark perspective and beyond. **GeoInformatica**, v. 23, n. 1, p. 37–78, 2019. Citado nas páginas 20, 36 e 40.

ZAHARIA, M.; CHOWDHURY, M.; FRANKLIN, M. J.; SHENKER, S.; STOICA, I. Spark: Cluster computing with working sets. In: **2nd USENIX Workshop on Hot Topics in Cloud Computing**. Berkeley, CA, USA: USENIX Association, 2010. Citado nas páginas 20 e 33.

ZHANG, M.; WANG, H.; LU, Y.; LI, T.; GUANG, Y.; LIU, C.; EDROSA, E.; LI, H.; RISHE, N. TerraFly GeoCloud: An online spatial data analysis and visualization system. **ACM Transactions on Intelligent Systems and Technology**, v. 6, n. 3, p. 34:1–34:24, 2015. Citado na página 62.


ICMC
USP
SÃO CARLOS