

UNIVERSIDADE FEDERAL DO PARANÁ
DEPARTAMENTO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM INFORMÁTICA

Método de Representação
Espacial de Clustering

Dissertação de Mestrado

Aluno: Luciane Cristina Pinheiro
Orientadora: Maria Salete Marcon Gomes Vaz

Curitiba, Abril de 2006

LUCIANE CRISTINA PINHEIRO

Método de Representação Espacial de Clustering

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Setor de Ciências Exatas da Universidade Federal do Paraná como requisito parcial para a obtenção do grau de Mestre.

Orientação Dr^a Maria Salete Marcon Gomes Vaz.

**CURITIBA
ABRIL, 2006**

*Para aqueles que me acompanharam de perto ou de longe, me apoiando e
torcendo em toda jornada.*

AGRADECIMENTO

O mestrado trouxe para mim não apenas conhecimentos, mas também experiências para a vida. Agradeço as pessoas que de uma forma ou de outra me ajudaram nesta jornada:

A Salete por ter acreditado, incentivado e me apoiado. A você obrigada pela orientação e amizade.

Ao Departamento de Informática pela oportunidade, em especial a Jucélia pelo carinho, aos professores Alexandre I. Direne, Marcos S. Sunye e Laura Sanches com suas idéias e esclarecimentos. Jucélia pelo carinho.

Ao Simepar pela ajuda financeira, estrutura e a motivação.

Aos meus colegas Marco Jusevicius, Leonardo Calvetti, Rafael Cabral, Marcelo Andrioni, Helder Nocko, Anderson Araújo, Carlos Meneghette, Luis Fernando G., Izabeli Langner, Lílían S., Rozana, Sheila R. P., Cícero Z. e Carlos Vasconcellos pelo incentivo e a ajuda tirando dúvidas e oferecendo idéias. Fábio Sato pelo apoio e sugestões. Daniel Cordeiro e Anderson Tadashi pela grande ajuda e paciência.

A todos meus amigos que se preocuparam e torceram.

Àqueles que se foram desta vida, mas deixaram um eterno carinho, o apoio, as orações e um abraço amigo.

Ao Leonardo pelo incentivo, amizade, paciência, força, companheirismo e amor incondicional.

Não posso deixar de agradecer a toda minha família, em especial meus pais por ter me dado as primeiras lições e ter me possibilitado voar. Para minha irmã Angela, pelo incentivo, conselhos e ajuda financeira, bem como Silvana pela compreensão e colaboração e ao meu irmão Luis Eduardo pela prontidão.

Pelos caminhos, pessoas e lugares em que me colocaste, agradeço principalmente a Deus, me iluminando, dando força e fazendo evoluir.

A todos meus agradecimentos.

***“Saber que se sabe o que se sabe, e
saber que não sabe o que não sabe;
eis a verdadeira ciência”.***

Confúcio

***“O poder nasce do querer. Sempre
que o homem aplicar a veemência e
perseverante energia de sua alma a
um fim, vencerá os obstáculos, e, se
não atingir o alvo, fará pelo menos
coisas admiráveis”.***

José de Alencar

SUMÁRIO

Resumo	i
Abstract.....	ii
1. Introdução	1
2. Abordagem de Clustering.....	4
2.1. Conceituação	4
2.2. Componentes de Função de <i>Cluster</i>	5
2.3. Estágios de <i>Clustering</i>	6
2.4. Representação de Clustering	8
2.5. Algoritmos de Clustering.....	10
2.5.1. Clustering hierárquico	10
2.5.2. Clustering Particionado	10
2.5.2.1. K-Médias ou K-Means	11
2.5.2.2. Algoritmo QT Clustering.....	12
2.5.2.3. Mapas Auto-Organizáveis (SOMs)	13
2.5.3. Clustering baseado em grafos.....	14
2.6. Considerações Finais	15
3. Banco de Dados Espaciais.....	17
3.1. Dados Espaciais	18
3.2. Objetos Espaciais.....	24
3.3. Métodos Espaciais	30
3.4. Alguns SGBDs e o Enfoque Espacial.....	33
3.4.1. Oracle e Oracle Spatial	34
3.4.2. PostgreSQL e a Extensão PostGIS	35
3.4.3. MySql	37
3.5. Considerações Finais	38
4. Método de Representação Espacial de Clustering para Banco de Dados.....	40
4.1. Descargas Atmosféricas	43
4.2. Banco de Dados Espaciais no Apoio de Consultas sobre Raios.....	46
4.3. Metodologia Utilizada	50
4.4. Método de Representação Espacial de Clustering.....	53
4.4.1. Experimentos para a Criação do Método	55
4.4.1.1. Experimento 1- Aplicação de Algoritmos da Inteligência Artificial através do Weka.....	55
4.4.1.2. Experimento 2 – Funções OGIS	58
4.4.2. O Método Desenvolvido.....	60
4.4.3. Implementação.....	68
4.4.4. Experimentos	73
4.4.4.1. Experimento com 30 pontos	73
4.4.4.2. Experimento com 500 pontos	79
4.4.4.3. Experimento com 1000 pontos	83
4.4.4.4. Análise Comparativa	85
4.4.5. Conclusões Gerais sobre o Método Desenvolvido	91
5. Conclusões e Expectativas de Trabalhos Futuros.....	94
Referências Bibliográficas.....	98
APÊNDICE 1	110
Experimento com 30 pontos -vizinhos	110
APÊNDICE 2	111

Experimento 1: Dados de Entrada	111
APÊNDICE 3	115
Experimento nº 1 pelo Método de Representação de Clustering: Dados de Entrada e Clustering associado	115
APÊNDICE 4	116
Experimento nº 1 pelo Método de Representação de Clustering: Passos realizados	116
APÊNDICE 5	119
Resultado do Método de <i>Clustering</i> Espacial para 32 quilômetros.....	119
APÊNDICE 6	121
Experimento 1: Resultado da utilização do Weka :.....	121
K-Means	121
APÊNDICE 7	122
Experimento 1: Resultado da utilização do Weka : EM.....	122
ANEXO 1	123

LISTA DE ABREVIATURAS OU SIGLAS

API	Application Programming Interface
ARFF	Attribute-Relation File Format
ASCII	American Standard Code for Information Interchange
EM	Expectation Maximization
GIST	Generalized Search Tree
GMT	Greenwich Mean Time
IA	Inteligência Artificial
IBGE	Instituto Brasileiro de Geografia e Estatística
ICC	Iterative Conductance Cutting
ODBC	Open Database Connectivity
OGC	Open Gis Consortium
QT	Quality Threshold
QGIS	QuantumGis
MCL	Markov Clustering
PSQL	Postgres Query Language
RINDAT	Rede Integrada Nacional de Descargas Atmosféricas
SAD/69	South American Datum 1969
SFSSQL	Simple Features Specification for SQL
SIG	Sistemas de Informações Geográfica
SGBD	Sistema Gerenciador de Banco de Dados
SGBDE	Sistema Gerenciador de Banco de Dados Espaciais
SGBD-R	Sistemas de Gerenciamento de Dados Relacionais
SGBD-OR	Sistema Gerenciador de Banco de Dados Objeto/Relaciona
SOMs	Mapas Auto-Organizáveis
SQL	Structured Query Language
SQL/MM	Structured Query Language Multi Media
SRID	Identificação do Sistema de Referência Espacial
UTM	Universal Transverse Mercator

ÍNDICE DAS FIGURAS

Figura 2.1: Estágio de Clustering	6
Figura 2.2 Representações de Cluster.....	9
Figura 3.1: Componentes Básicos de umSIG	18
Figura 3.2: Arquitetura de Sistemas de Informação Geográfica	19
Figura 3.3 Hieraquia de Classe Geométrica	25
Figura 3.4: Objeto Espacial doTipo Point	27
Figura 3.5: Objeto Espacial doTipo Linestring	27
Figura 3.6: Objeto Espacial doTipo Multilinestring.....	28
Figura 3.7: Objeto Espacial doTipo Polygon	29
Figura 3.8: Objeto Espacial doTipo Multipolygon.....	29
Figura 3.9: Elementos Geométricos doOracle Spatial (Queiroz,2003)	35
Figura 4.1: Primeiro Acompanhamento de Descargas Atmosféricas.....	40
Figura 4.2: Segundo Acompanhamento de Descargas Atmosféricas.....	41
Figura 4.3: Terceiro Acompanhamento de Descargas Atmosféricas	42
Figura 4.4: Consulta Por Raios Dentro de umMunicípio	47
Figura 4.5: Estado doParaná e A Incidência de Descargas Atmosféricas em 06-07-2003 14:00 Horas Às 20:00 Horas (Horário Gmt -2)	52
Figura 4.6:Clusters Gerados pela Utilização doK-Means com o Software Weka.....	57
Figura 4.7: Clusters Formados pelo Experimento 02	59
Figura 4.8: Dados Iniciais.....	61
Figura 4.9: Pontos Expandidos Segundo Distância Informada pelo Usuário.....	61
Figura 4.10: Área de Vizinhança doPonto A.....	62
Figura 4.11 Ponto com Mais Vizinhos	63
Figura 4.12: Formação de umCluster	65
Figura 4.13: Cluster 1	66
Figura 4.14: Visualização doMétodo de Clustering Para Sgbde	67
Figura 4.15: Teste de Sobreposição de Geometrias	70
Figura 4.16: Teste com Poucos Pontos – Pontos Avaliados.....	74
Figura 4.17: Teste com Poucos Pontos – 8 Quilômetros de Vizinhança.....	75
Figura 4.18: Teste com Poucos Pontos – 16 Quilômetros de Vizinhança.....	77
Figura 4.19: Teste com Poucos Pontos – 32 Quilômetros de Vizinhança.....	78
Figura 4.20: Diferença Entre Limiares de Vizinhança	79
Figura 4.21: Descargas Atmosféricas emDois Momentos	80
Figura 4.22: Clusters Vistos como Polígonos	82
Figura 4.23: Teste 1 com Mil Pontos – 8 Quilômetros de Vizinhança	84
Figura 4.24: Clustering Formado Para 30 Pontos Aplicando Weka K-Means.....	86
Figura 4.25: Clustering Formado Para 30 Pontos Aplicando Weka Em	87
Figura 4.26: Teste Utilizando o Weka.....	89
Figura 4.27: Teste Utilizando o Método de Clustering Espacial.....	90

ÍNDICE DAS TABELAS

Tabela 1: Exemplo da Matriz de Distância	11
Tabela 2: Vizinhos.....	63
Tabela 3: Primeira Iteração	64
Tabela 4: Fase Atual de Clustering	65
Tabela 6: Resultado do Teste com Poucos Pontos	74
Tabela 7: Resultado com Limiar de Vizinhança de 16 Km.....	76
Tabela 8: Teste com Vizinhança de 32 Km.....	78
Tabela 9: Clustering Momento 1	81
Tabela 10: Clustering Momento 2	81
Tabela 11 : Teste com Mil Pontos	84
Tabela 12: Resultado Teste Utilizando Weka- K-Means	85
Tabela 13: Resultado Teste Utilizando Weka- EM.....	87
Tabela 14: Teste com o Algoritmo K-Means	88
Tabela 15: Teste com o Método de Representação Espacial de Clustering	89

RESUMO

Os Sistemas Gerenciadores de Banco de Dados Espaciais (SGBDE) estão crescendo de acordo com as necessidades do mercado. Esta evolução está presente desde a inclusão dos primeiros aspectos para armazenamento de dados na forma espacial, até funções mais complexas tais como achar o envoltório de um conjunto de pontos e a distância entre um ponto e uma linha. Apesar da crescente evolução, muitas funcionalidades ainda estão por ser inseridas nestes SGBDE. Uma das funcionalidades em estudo é aquela que detecta aglomerado ou agrupamentos (clusters) de dados espaciais. *Clustering* de dados é uma tarefa não trivial, que está sendo estudada há anos em diversas áreas, visando descobrir padrões em dados. Esta dissertação propõe um método que viabiliza esta nova funcionalidade através da detecção de cluster. O método criado tem como entrada dados do tipo ponto em um determinado período de tempo. As saídas são pontos agrupados que estão dispostos espacialmente e temporalmente. O método visou criar os primeiros passos para o desenvolvimento de uma função de cluster. Este método terá aplicações nas diversas áreas, como a meteorologia, biologia ou qualquer área em que se almeja descobrir padrões de dados baseados em similaridade.

METHOD OF SPATIAL DATA CLUSTERING

ABSTRACT

The Spatial Data Base Management Systems (SDBMS) are growing in accordance with the necessities of the market. This evolution is present since the inclusion of the first aspects for storage of data in the space form, until functions more complex such as to find convex hull of a collection of points and distance between a point and a line. Despite the increasing evolution, much functionality still is for being inserted in these SGBDE. One of the functionalities in study is that one that detects accumulations or groupings (clusters) of space data.. *Clustering* of data is a not trivial task, that is being studied has years in diverse areas, aiming at to discover standards not waited in data. This dissertation considers a method that makes possible this new functionality through the detention of cluster. The method bred has as input given of the type point in one determined period of time. The output are grouped points that are made use space and secularly. The method aimed at to create the first steps for the development of a function of cluster. This method will have applications in the diverse areas, as meteorology, biology or any area where if it longs for to discover standards of data based on similarity.

1. INTRODUÇÃO

A indagação do homem com sua localização espacial, bem como o registro de suas atividades é uma preocupação registrada desde o início dos tempos. Com o passar dos anos a tecnologia foi se desenvolvendo para se adaptar as necessidades da humanidade.

A área de banco de dados está passando por mudanças induzidas por uma nova geração de SGBD's – Sistema Gerenciador de Banco de Dados, voltados para Sistemas de Informações Geográficas – SIG. Esses sistemas são utilizados para representar e analisar computacionalmente informações localizadas no espaço (POSTGRESQL, 2005; FERREIRA *et al*, 2002) e têm aplicações em diversas áreas.

Um exemplo de aplicação desta nova geração de SGBD é a gestão de informações sobre descargas atmosféricas, ou como são mais conhecidas: raios. A localização destas descargas atmosféricas pode ser inserida espacialmente dentro de um SGBDE – SGBD Espacial, como tipo ponto, facilitando assim, diversas consultas utilizando operações topológicas tal como união e intersecção. O tratamento dessas operações, dentro de um banco de dados convencional não é possível, visto que estes tipos de operações só se aplicam ao formato de objetos espaciais.

E para a gestão de informações sobre descargas elétricas, faltam ferramentas automatizadas para: a) representação do desenho de elipse de probabilidade (ou elipse de segurança) da incidência de descargas elétricas (cálculo realizado pelos sensores de descargas elétricas) (BENETI, 2002). A representação da elipse deve ser flexível em relação ao número de vértices, isto é, pode-se definir o número de vértices. Este parâmetro pode ser informado pelo usuário e assim, difere do tipo do objeto círculo, que nada mais é que uma elipse de excentricidade 0; b) representação de quadrantes dentro de uma circunferência, (por exemplo, redondezas de uma refinaria), aonde é possível definir, por exemplo, os pontos cardeais e colaterais; e c) caracterizar aglomeração (“*clustering*”) de certos tipos de dados, tal como aglomeração de descargas elétricas dentro de uma tempestade.

Com o aumento de dados armazenados nos banco de dados aumenta a necessidade por métodos de análise para fazer uso da informação contida implicitamente nos dados. Uma das tarefas primárias para esta gestão de dados é a análise de clusters (conjuntos)

para entender o agrupamento ou a estrutura natural em um conjunto de dados (ANKERST *et al.*, 1999).

A análise de clusters envolve várias áreas, em especial dentro da meteorologia, onde a descoberta de núcleos de descargas elétricas em tempestades é de valia para o acompanhamento da trajetória de uma tempestade e para o entendimento do contexto climático dentro de uma região: “A clusterização serve para facilitar a visualização de descargas elétricas, sendo uma importante informação auxiliar sobre a região afetada pela parte mais severa da tempestade, devido a sua estrutura eletrodinâmica” (POLITI *et al.*, 2004).

Na literatura de banco de dados pesquisada não foram encontradas funções e/ou métodos, em especial, algoritmos de cluster. As funções de *clustering* similares foram descritas na Matemática, Medicina, Processamento de Imagens, Geologia e na Inteligência Artificial. Esta carência na ciência se deve ao fato de que o tratamento de cluster não é uma tarefa trivial em qualquer uma das áreas. Ressalta-se ainda que algoritmos de *clustering* e classificação de argumentos geralmente são específicos para determinadas aplicações.

Os atuais SGBDE's, tal como o *Oracle Spatial* (Oracle) e *PostGis* (PostgreSQL) ainda não implementaram uma função de cluster. A motivação para pesquisar *clustering* dentro de banco de dados parte da necessidade de detectar dados espaciais com características muito próximas. Essas características podem ser vistas como um cálculo estatístico, no qual o desvio padrão é mínimo, isto é, estatisticamente os dados têm particularidades próximas, como, por exemplo, distribuição temporal, cores iguais, classificação e agrupamento de dados iguais. Metodologias que tratam cluster de dados são contempladas visando o agrupamento visual dos dados, o ponto central do cluster ou a análise de dispersão espacial de pontos.

Tendo em vista o exposto, nesta dissertação aborda-se a construção de um método para banco de dados cuja finalidade é gerar os primeiros passos para a criação de uma função de detecção de cluster, que possibilite aglomerar dados do tipo espacial *Ponto*. Para isso, foram realizados testes na tentativa de descobrir padrões nas localizações onde ocorreram descargas elétricas (raios).

Os dados utilizados para esses testes foram obtidos da triangularização das informações provindas dos sensores de descargas atmosféricas. Os sensores informam a origem da região onde há possibilidade de ter ocorrido uma descarga nuvem-solo. Esta região tem o formato de uma elipse - elipse de probabilidade - na qual a interseção, do

eixo maior e menor, forma o ponto central, onde existe a maior probabilidade de ter ocorrido o raio. O ponto central da elipse, onde ocorreu esta perturbação elétrica, foi caracterizada espacialmente como POINT (ponto). Os testes foram realizados dentro da área de cobertura dos sensores de descargas atmosféricas.

Vale ressaltar ainda que o estudo de caso feito para os dados de descargas elétricas inseridos dentro de um gerenciador de banco de dados espacial é pouco utilizado no Brasil, devido à dificuldade de se obter os dados. Geralmente, esses dados são informações valiosas dentro de institutos meteorológicos e, portanto não são facilmente disponíveis.

Neste trabalho utilizou-se os dados e equipamentos do Instituto Tecnológico do Paraná - Simepar – para a realização desta pesquisa. Para a apresentação do método de representação espacial de cluster esta dissertação foi organizada em 5 capítulos, além deste introdutório. No Capítulo 2 a abordagem de *clustering* é descrita. No Capítulo 3 discute-se banco de dados espaciais, bem como os dados, tipos e métodos espaciais. E alguns SGBDs, com enfoque espacial, são descritos. No Capítulo 4 é apresentado o método para representação de *clustering* desenvolvido. No Capítulo 5 são abordadas as conclusões bem como a expectativa de trabalhos futuros.

2. ABORDAGEM DE CLUSTERING

2.1. Conceituação

Um cluster pode ser visto como um conjunto de dados de modo que os elementos deste conjunto possuam características em comum. *Clustering* de dados é uma técnica de classificação de padrões (observações, itens de dados ou vetor de características) dentro de clusters. O objetivo principal de *clustering* é separar objetos ou observações em classes naturais de forma que os elementos pertencentes a um mesmo grupo tenham um alto grau de semelhança ou similaridade, enquanto que, quaisquer elementos pertencentes a grupos distintos, tenham pouca semelhança entre si (ANDEBERG, 1973 *apud* JAIN *et al* 1999; NEVES *et al* 2001).

Uma descrição genérica do objetivo do *Clustering* seria o de maximizar a homogeneidade dentro de cada cluster enquanto se maximiza a heterogeneidade entre clusters. (HRUSCHKA e EBECKEN (2001), CARLANTONIO, 2001).

Alguns termos possuem significado similar com *clustering*, tais como agrupamento, conglomerados, análise de cluster, classificação automática, taxonomia numérica e análise tipológica. *Clustering* é uma técnica para análise estatística de dados, que engloba métodos utilizados para descobrir estruturas em um conjunto complexo de dados, na qual são usados muitos campos e reconhecimento de padrões, análises que envolvem mineração de dados, agrupamento, tomada de decisões, segmentação de imagem e padrões de classificação. Intuitivamente, padrões válidos para *clustering* são mais similares entre si, ao invés de um padrão pertencente a um cluster diferente. (JAIN *et al*, 1999; WIKIPEDIA, 2005).

Podemos definir ainda: “O problema de Clustering é descrito como: recebido um conjunto de dados, de objetos, tentar agrupá-los de forma que os elementos que compõem cada grupo sejam mais parecidos entre si do que parecidos com os elementos dos outros grupos. É colocar os iguais (ou quase) juntos num mesmo grupo e os desiguais em grupos distintos” (CARLANTÔNIO, 2001).

Clustering, aplicado em Inteligência Artificial, é seguido por um estágio onde a árvore de decisão ou o conjunto de regras é inferido para cada instância que o cluster no qual é pertencente (WITTEN, 2000). Já em banco de dados espaciais (Capítulo 3), a “*clusterização*” para o tipo de dado *point* foi escolhida visto que é o tipo espacial mais

simples existente, e como *linestring* e *polygon*, podem ser ditos como tipos essenciais. Vale ressaltar que o tipo *linestring* é resultado da interpolação de vários *points* e o tipo *polygon*, vários *linestrings* (POSTGIS, 2005). Um exemplo de aplicação do tipo *point* pode ser dito em latitude e longitude como a incidência de uma descarga elétrica, localização das sedes de um município, entre outros.

2.2. Componentes de Função de *Cluster*

Os componentes que devem estar presentes em uma função que represente cluster são os seguintes (JAIN E DUBES, 1988; CARLANTONIO, 2001; KAESTNER, 2006):

i) Representação de padrão: extração de características da seleção. Refere-se ao número de classes, o número de padrões disponíveis. Deve ter uma alta similaridade intraclasse e baixa similaridade interclasses. Deve fornecer resultados de fácil interpretação e utilizáveis na descrição dos clusters, sendo assim, de representação simples. No caso de incidências de descargas atmosféricas, o tipo *point* é único.

ii) Definição de padrão da medida de proximidade: apropriada para o domínio dos dados. Geralmente, a distância entre um par de padrões. “A qualidade do resultado de um processo de *clustering* depende da medida de proximidade, do método utilizado e de sua implementação. A qualidade de um processo de *clustering* também deve ser avaliada pela sua habilidade de descobrir alguns ou todos os padrões escondidos” (KAESTNER, 2006). A distância Euclidiana, geralmente, reflete a similaridade entre os padrões (DIDAY e SIMON, 1976 *apud* JAIN *et al* 1999) “As medidas de similaridade fornecem valores numéricos que exprimem a ‘distância’ entre dois objetos. Quanto menor o valor desta, mais semelhantes será os objetos e estes deverão ficar no mesmo cluster. De outro modo, quanto maior a ‘distância’, menos similares serão os objetos e, em consequência, eles deverão estar em grupos distintos”.(CARLANTONIO, 2001)

Para esta pesquisa o domínio espacial dos dados é o Brasil. Valores de aproximadamente 8 km, 16 km e 32 km entre os pontos foram utilizados para poder avaliar o impacto, em relação ao desempenho nos testes realizados, da consulta dentro de espaços maiores. O valor ideal da distância entre os pontos será calculado de acordo com o algoritmo de agrupamento da área de Inteligência Artificial escolhido (abordados

no Capítulo 2), ou definido como valores de entrada pelo usuário. Neste trabalho foram estudados vários algoritmos de *clustering* conhecidos na literatura e foi elaborado um método próprio com base na essência desses algoritmos. Alguns destes algoritmos serão apresentados na Seção 2.5.

2.3. Estágios de *Clustering*

Alguns dos estágios de *clustering* são os que seguem:

- i) **Classificação ou agrupamento:** pode ser desempenhada de muitas formas. A saída de aglomeração de dados pode ser complicada (uma porção de dados dentro de grupos) ou *fuzzy* (onde cada padrão tem um grau de diferença de outros membros em cada grupo). Os formatos deverão ser os mais diversificados possíveis (redondos, alongados, etc.).
- ii) **Abstração de dados (se necessário):** é o processo de extração de uma simples e compacta representação de um conjunto de dados. A classificação pode ser “filtrada” pela medição de uma característica, por exemplo, a distribuição temporal da descarga elétrica.
- iii) **Número adequado de clusters:** o método deve ser capaz de encontrar o número natural de clusters. Muitos métodos utilizam um valor de referência, definidos *à priori*.
- iv) **Garantia de validação (se necessário):** a análise da validade dos dados agrupados. O cluster deve ser robusto na presença de ruídos (que podem ser dados perdidos ou errados), de maneira que a qualidade dos clusters obtidos não seja afetada. Esta análise poderá ser feita através de software de visualização de dados.

A Figura 2.1 ilustra a sequência típica dos três primeiros passos descritos anteriormente.

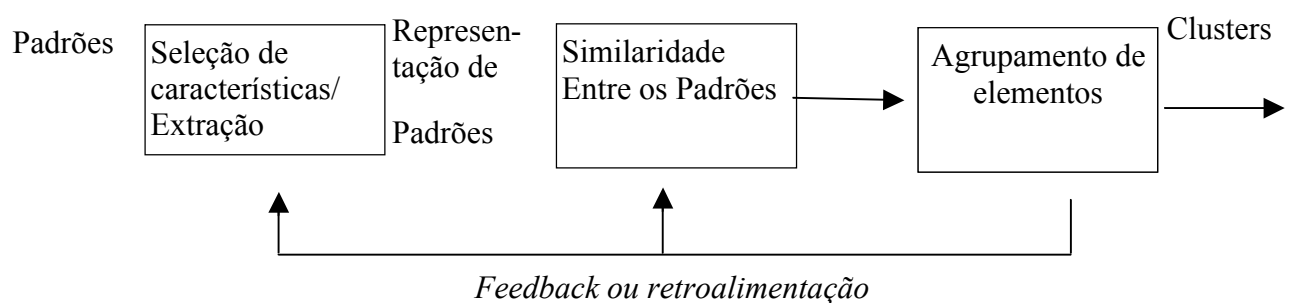


FIGURA 2.1: ESTÁGIO DE CLUSTERING
Adaptado (JAIN et al 1999)

A Figura 2.1 ilustra os estágios de *clustering* onde o processo de identificação do subconjunto de características mais efetivas é extraído. A proximidade de padrões pode ser um processo automático de reconhecimento. O caminho de retorno (*feedback*) entre os estágios de *clustering*. É onde o processo de agrupamento deve efetivamente extrair a similaridade computacionalmente.

O agrupamento, não somente de dados de descargas elétricas, mas quaisquer dados do tipo espacial *point*, ajuda a descobrir padrões no decorrer do tempo. A análise espaço-temporal das ocorrências de descargas elétricas exige uma forma de representação conveniente das mesmas, o que é agravado pela alta frequência de descargas elétricas em uma tempestade, que gera um volume de dados elevado.

Por exemplo, deseja-se saber se um determinado ponto está contido um cluster. Se o mesmo está, em qual cluster está presente? Qual o comportamento desse cluster com o tempo? Qual a distância do ponto central deste cluster com os demais clusters?

A detecção de cluster espacial “ pode ser feita a partir da análise da distribuição de pontos (*point pattern*), onde se avalia se a distância entre os pontos de ocorrência de eventos é ou não aleatória, ou entre áreas, onde se diagnostica se a frequência na ocorrência de eventos em áreas apresenta distribuição condicionada pela posição espacial das regiões estudadas’ (CARVALHO, 1997).

As metodologias de representação espaço-temporal, geralmente, tratam estas questões embasadas em imagens. Algumas metodologias utilizadas na representação espaço-temporal das descargas elétricas são as que seguem (SILVERMAN, 1990):

i) **Paintball**: baseia-se na plotagem de cada uma das instâncias de descargas elétricas num dado instante. Possui muitas falhas de acordo com intervalo de tempo, devido à natureza aleatória de cada evento.

ii) **Agrupamento em grade**: divide-se o espaço onde as descargas elétricas estão distribuídas numa grade retangular de tamanho fixo, e para cada célula, em determinado período de tempo, é calculado o número de descargas ocorridas. A grade pode deixar de representar adequadamente um núcleo convectivo.

iii) **Clustering**: todos os algoritmos de *clustering* têm como objetivo agrupar, em classes, elementos com características comuns, no caso de descargas elétricas (intraclasse) e minimizar a similaridade entre elementos de classes distintas (interclasse). Os clusters possuem formatos irregulares (poligonais), e

isso dificulta o acompanhamento da evolução temporal desses núcleos de descargas elétricas.

As técnicas de *clustering* são aplicadas quando não há classificação prévia, de preferência quando as instâncias estão sendo divididas dentro de grupos naturais. Estas aglomerações, presumidamente, refletem algum mecanismo que trabalha no domínio do qual as instâncias são desenhadas (WITTEN, 2000; GAMA, 2006).

Com o agrupamento destes dados, via banco de dados com extensão espacial, é possível acompanhar tempestades através das funções de geometria, como por exemplo, cidades afetadas por núcleos de atividade elétrica, focos de calor, doenças em uma região, entre outros.

Existem diferentes modos, no qual o resultado de *clustering* pode ser expresso. Os grupos que são identificados podem ser exclusivos, no qual uma instância pertence a somente um grupo.

Dentro de uma base de dados, a classificação destes padrões de dados tem relevância em estudos estatísticos. Em uma base de dados com abordagem espacial, esta relevância se mostra em mais de uma forma: a assimilação de valores através do cálculo de distância entre os objetos espaciais e o cálculo de vizinhança.

Dentre as vantagens da representação de *clustering* espacial estão possibilidades de manipulação e análise independente de qualquer unidade de área pré-fixada. Geralmente, as representações tradicionais de cluster fixam determinadas áreas e a partir dessas os clusters são calculados. Nesta representação, os dados podem ser colhidos de acordo com características temporais, e não apenas espaciais.

Este método, a princípio, foca uma aplicação de cluster para dados do tipo ponto. O tipo ponto representa os aspectos centrais das descargas atmosféricas, podendo ser disseminada para várias outras aplicações, tal como, por exemplo, aglomeração de determinados tipos de comércio por região (onde cada comércio possui coordenada do tipo latitude, longitude), doenças em uma região, ou a aglomeração de um determinado tipo de vegetação.

2.4. Representação de Clustering

Na Figura 2.2 são apresentadas formas de representação de clusters: a) divisão por barras; b) diagrama de Venn; c) matriz de valores; e d) dendograma.

Alguns algoritmos de *clustering* permitem uma instância pertencer a mais de um agrupamento. Neste caso, para cada instância há uma probabilidade, ou grau de membros, onde é permitido a cada objeto, pertencer a dois clusters, e, portanto, ter instâncias dentro de duas dimensões, sobrepondo subconjuntos. Esta sobreposição pode ser representada como um Diagrama de *Venn* (Figura 2.2b) (WITTEN, 2000).

O dendograma (Figura 2.2d) pode ser formado através de uma abordagem aglomerativa (*bottom-up*) ou divisiva – de partição (*top-down*). A primeira abordagem parte das folhas superiores para a raiz, onde cada objeto forma seu próprio *cluster*. Em cada etapa é calculada a distância. Os dois *clusters* com distância mínima são agrupados e as distâncias são atualizadas até que todos os objetos estejam num único *cluster* ou que uma condição de término ocorra. (AGRAWAL *et al.*, 1998; NG e HAN; HAN e KAMBER, 2001; CARLANTONIO,2001). A segunda abordagem parte da raiz para as folhas. “Em cada etapa, um *cluster* é escolhido e dividido em dois *clusters* menores. Este processo continua até que tenhamos *n clusters* ou até que uma condição de término aconteça” (NG e HAN, 1994; HAN e KAMBER, 2001).

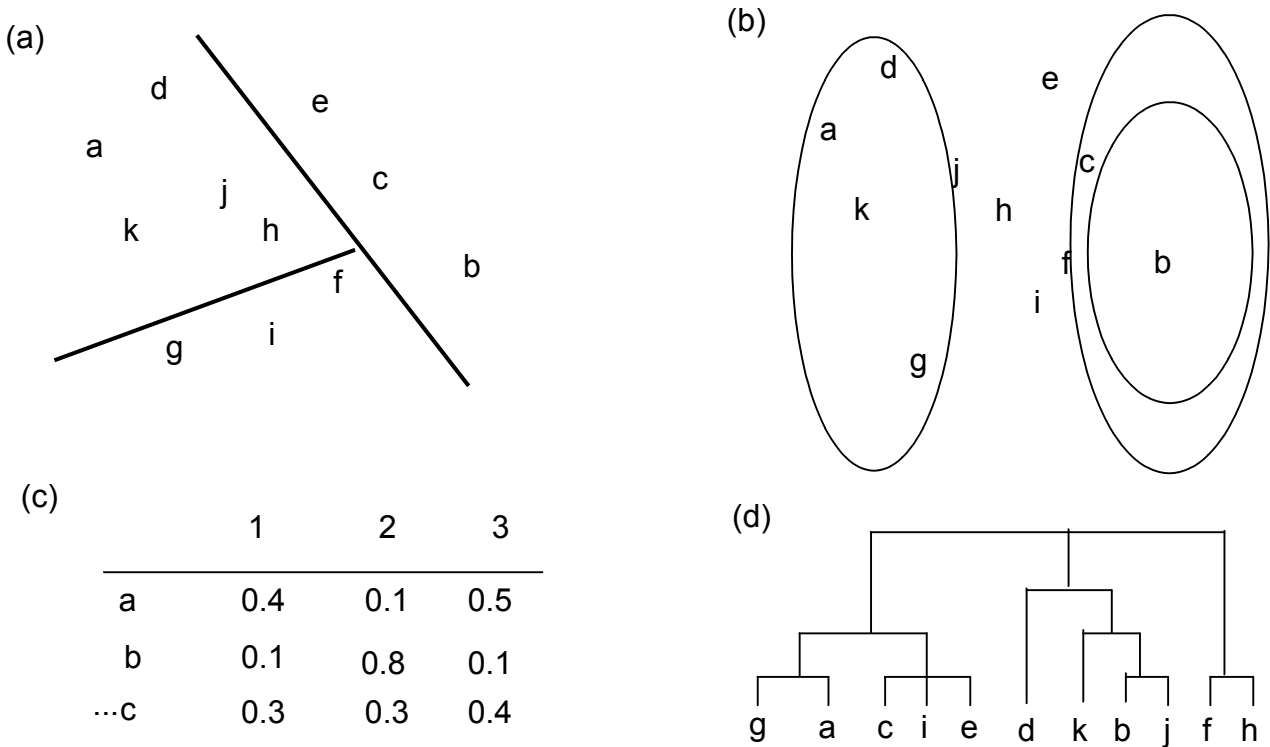


FIGURA 2.2 REPRESENTAÇÕES DE CLUSTER
 Fonte:(BARANAUSKAS, 2003)

A seção seguinte levanta alguns algoritmos de *clustering* existentes na literatura.

2.5. Algoritmos de Clustering

Os *clusters* são classificações em diagramas, que mostram como as instâncias estão dispostas dentro de aglomerações. Os algoritmos de *clustering* servem para, dado um conjunto de objetos, colocar os objetos em grupos baseados na similaridade entre eles. Estes algoritmos são utilizados para encontrar padrões nos dados.

Os algoritmos de *clustering* tradicionais podem ser do tipo *hierárquico* e *particionado*. Nas seções seguintes serão descritos alguns algoritmos desses tipos.

2.5.1. Clustering hierárquico

O algoritmo de *Clustering* hierárquico produz uma ordem para os padrões de agrupamento, criando uma decomposição hierárquica da base de dados. Este cria uma árvore, chamada de dendograma, na qual as folhas são os objetos e os nós internos revelam a estrutura de similaridade dos pontos. O *clustering* hierárquico decide os pares de *clusters* mais próximos que serão agrupados ou em qual *cluster* maior será separado; respeitando a medida de similaridade entre eles.

Os passos deste algoritmo são: (i) Iniciar com n clusters, sendo todos os pontos colocados em seus próprios clusters; (ii) Calcular as dissimilaridades entre os objetos. (iii) Enquanto há mais de um *cluster*, agrupe o par de *clusters* mais próximos. O comportamento do algoritmo depende de como o *par de clusters mais próximo* é definido”. (BARANAUSKAS,2003; NEVES et al 2001).

Algumas características do método de clustering hierárquico: (a) esse algoritmo não reflete as maneiras distintas em que padrões podem ser similares; (b) não refaz o que foi feito previamente, isto é, no nível anterior; (c) a estrutura sempre é uma árvore; (d) padrões só podem ser agrupados baseando-se em decisões locais, as quais, uma vez tomadas, não podem ser re-avaliadas; (e) a complexidade é de $O(n^2)$, onde n é o número inicial de objetos; (f) essas técnicas não são eficazes no caso do aparecimento de ruído, isto é, dados errados, desconhecidos ou perdidos; e (g) o usuário interfere no resultado decidindo onde separar a árvore em grupos.

2.5.2. Clustering Particionado

Nos algoritmos do tipo particionado, são determinadas todas as aglomerações dentro de uma aglomeração, até que não seja possível esta divisão e sua avaliação ser feita por algum critério. Geralmente, o critério global envolve a minimização de alguma

medida de dissimilaridade dentro de cada *cluster*, enquanto maximiza a dissimilaridade entre *clusters* diferentes (RAYMOND E HAN, 1994; KASKI, 1997).

2.5.2.1. K-Médias ou K-Means

Esta técnica difere do agrupamento hierárquico de várias maneiras, como por exemplo, no *clustering* hierárquico, um passo decisivo é a seleção da medida de distância. No K-Médias particular, não há hierarquias, pois os dados são particionados. O resultado é apenas a pertinência final de cada padrão aos aglomerados. O número de aglomerados permitido (k) tem que ser definido *a priori*. Esse método é simples e razoavelmente efetivo. O centro final do *cluster* não representa o mínimo global, podendo ter diferentes *clusters* finais devido às escolhas diferentes do centróide do *cluster*.

Um grande número de variações do procedimento K-Médias tem sido desenvolvido. Um *clustering* hierárquico pode ser produzido pela aplicação do algoritmo com $k = 2$ para todo o conjunto de dados. A partir daí repete-se, recursivamente dentro de cada *cluster*, até não haver mais partições. Outras variações se concentram na velocidade das aglomerações: o algoritmo básico pode ter melhor desempenho por ter um número substancial de iterações, cada uma encontrando a distância do centro de k *cluster*, para cada instância, para determinar o mais próximo (SOUTO 2004, WITTEN, 2000). SOUTO (2004) ilustra as etapas deste algoritmo:

- **Etapla 1:** Suponha que os padrões de expressão estejam posicionados em um espaço bi-dimensional baseados na matriz de distância. Esta distribuição espacial segue uma matriz (Tabela 1) que contém o ponto e a distância desse ponto com todos os outros pontos.

	Ponto A	Ponto B	Ponto C
Ponto A (x1, y1)	-	1	3
Ponto B (x2, y2)	1	-	2
Ponto C (x3,y3)	3	2	-
...	...	...	...

Tabela 1: Exemplo da matriz de distância

- **Etapla 2:** Os primeiros k centros dos aglomerados são escolhidos aleatoriamente. Supondo um $k = 3$, a semente que gera este número aleatório pode ser informada pelo usuário, ou não. Estes k pontos escolhidos serão ditos o centro de cada *cluster*.

- **Etapa 3:** Cada ponto é atribuído ao aglomerado associado com o centro mais próximo. Depois de escolhidos os pontos centrais de cada cluster, a partir da matriz de distância, verifica-se para cada ponto, qual o centro mais próximo. Para cada centro são atribuídos n valores de pontos próximos.
- **Etapa 4:** Minimiza-se a soma dos valores inter-aglomerados dos quadrados das distâncias da média do aglomerado. O centróide é movido, computando um novo representante (centro) do aglomerado, em um espaço bi-dimensional baseado na matriz de distância. Para cada centróide e seus pontos próximos, as distâncias são recalculadas. Através desta nova matriz de distância, recalcula o seu centróide que será o novo representante do *cluster*.
- **Etapa 5:** As etapas 3 e 4 são repetidas com os novos representantes e são executadas até que mudanças não mais ocorram.

Este algoritmo apresenta as características de (GAMA, 2006, BARANAUSKAS, 2003): a) Partição: constrói uma partição de um conjunto de dados com n observações em k grupos distintos; b) Usado quando o número de aglomerados é conhecido *a priori* ou vários valores devem ser tentados até encontrar o “melhor”; c) Não - determinístico: inicializações aleatórias dos centros; e d) Aglomerados esféricos. Os dados devem ser numéricos e o cálculo de similaridade é através da distância Euclidiana (há um variante chamado algoritmo *K-medians* que aborda esse problema). O algoritmo K-Médias interage sobre o conjunto de dados até que a convergência seja alcançada.

A dificuldade deste método é não conseguir determinar se há realmente apenas k aglomerados distintos representados nos dados. “Na prática converge num número finito de passos”. O algoritmo K-médias é sensível a pontos que não pertencem a clusters (*outliers*), sendo assim, pode distorcer a posição do centróide e deteriorar o *cluster*. Além dos pontos expostos, o algoritmo trabalha melhor com dados que contêm *clusters* esféricos; *clusters* com outro tipo de geometria podem não ser encontrados. Como é um método exaustivo, ele enumera todas as possíveis partições, possuindo complexidade exponencial.

2.5.2.2. Algoritmo QT Clustering

O QT (*Quality Threshold*) Clustering (HEYER et al, 1999) é um outro algoritmo do método de particionamento de dados. Este algoritmo visa um “limiar de qualidade” entre os *clusters* e o número de *clusters* não é definido *a priori* porém requer mais capacidade computacional que o *K-means*.

O algoritmo segue as seguintes fases (WIKIPEDIA, 2005; SILICONGENETICS, 2003): (i) um objeto é escolhido aleatoriamente dentro de uma lista de objetos. (ii) o algoritmo determina quais objetos tem maior similaridade com o objeto escolhido no Passo (i); (iii) o usuário escolhe o diâmetro máximo para cada *cluster*. Além do diâmetro, o usuário pode informar também o número mínimo de similaridade entre os objetos e o tamanho máximo que um cluster deve ter; (iv) o *cluster* candidato é escolhido para cada ponto, incluindo o ponto mais próximo, bem como o próximo mais próximo, e assim por diante até o diâmetro do *cluster* não passar este limiar. (v) o *cluster* candidato com mais pontos é salvo como primeiro *cluster* válido, e todos os outros pontos do *cluster* são removidos para uso posterior.

O algoritmo é recursivo, repetindo o processo com um conjunto reduzido de dados. “A medida de distância entre os pontos e do grupo de pontos é computada usando ligação completa (*complete linkage*), que corresponde a distância máxima de um ponto para todos os outros pontos pertencentes ao grupo”. A complexidade, portanto, é exponencial.

Este algoritmo utiliza muito recurso computacional, isto é, utiliza muito processador, mas possui como vantagens: (i) não precisar antecipar o número de clusters que devem ser retornados; (ii) somente os clusters dentro de um padrão (limiar) definido pelo usuário são obtidos; e (iii) não é sensível a ordem de entrada dos dados.

2.5.2.3. Mapas Auto-Organizáveis (SOMs)

Os mapas auto-organizáveis foram propostos por Teuvo Kohonen (KOHONEN, 2001) para trabalhar com a identificação de padrões utilizando Redes Neurais. Nesses mapas, os *clusters* são organizados num mapa retangular bidimensional, onde os *clusters* semelhantes estão espacialmente correlacionados. Cada *cluster* é definido por um vetor de pesos com a mesma dimensão do vetor de entrada. Cada *cluster* está ligado aos seus vizinhos (GAMA, 2006).

O algoritmo de mapas auto-organizáveis é similar ao K-Médias, onde o número de aglomerados permitido tem que ser definido a priori, no entanto, diferentemente dos aglomerados gerados pelo K-Médias - nos SOMs há uma ordenação topológica dos aglomerados.

Os passos deste algoritmo são os que seguem (SOUTO, 2004; ASSIS, 2005):

- **Passo 1.** Achar o neurônio $i(x)$ cujo peso P esteja mais perto (em termos de distância Euclidiana) do vetor de entrada x apresentado. Este neurônio é chamado de *neurônio vencedor*;
- **Passo 2:** Como foi visto, o padrão P de pesos é selecionado aleatoriamente. Ajustar os pesos P do neurônio vencedor e de todos os que se encontram dentro de certa vizinhança centrada nele. Este ajuste deve ser feito de forma que estes vetores de peso (w_j) fiquem cada vez mais perto (vetorialmente) do centro de gravidade da classe de padrões que representam. Esta vizinhança do neurônio vencedor é na verdade uma função da distância entre ele e os demais neurônios do mapa de características topológico, e varia dinamicamente durante o aprendizado. Para cada nodo calcula-se a distância d de cada *cluster*; e
- **Passo 3:** Repetir até que critério de parada (convergência) seja satisfeito, isto é, até que não haja mais mudanças sensíveis.

Este algoritmo apresenta características de partição, descritas anteriormente e é usado quando o número de aglomerados é conhecido a priori. É não determinístico, pois utiliza inicialização aleatória dos pesos. A ordenação dos aglomerados é topológica, pois a cada iteração os pesos são reordenados. “Os SOMs tem demonstrado um bom desempenho no primeiro passo da classificação, que é a extração de características. No segundo passo, a classificação propriamente dita, onde são atribuídas classes aos dados de entrada” (ASSIS,2005)

As dificuldades apresentadas neste algoritmo são: resultado depende de muitos parâmetros, inicialização de pesos, taxa de aprendizado, taxa de vizinhança, número de iterações. A inicialização dos pesos é feita de forma aleatória o que pode acarretar pesos com valores inadequados. Há um custo para o processamento dos pesos, onde para cada iteração, um novo aprendizado dos pesos deverá ser feito. O cálculo para a descoberta e vizinhança depende da interação com o usuário, bem como o número máximo de iterações que este algoritmo deverá realizar.

2.5.3. Clustering baseado em grafos

O método de clustering apresentado nesta seção é baseado na teoria de grafos e consiste em: dado um grafo $G = (V, E)$ onde V é o conjunto de vértices (os próprios objetos) e E o conjunto de arestas (grau de similaridade), dividir os vértices de V em

subconjuntos disjuntos, ou *clusters* onde cada objeto conexo é um conjunto alcançável através dos demais (DIAS e OCHI; BARANAUSKAS, 2003).

Um algoritmo de *clustering* baseado neste método incorpora a idéia de desempenhar um percurso aleatório no grafo, identificando subgrafos mais densamente conectados. Destes algoritmos, destacam-se três: *Markov Clustering* (MCL) e *Iterative Conductance Cutting* (ICC). No primeiro, enfatiza iterativamente a conectividade intracluster sobre conectividade intercluster. No segundo algoritmo, a porção inicial baseada na distância intracluster é repetidamente redefinida. Mais detalhes sobre estes algoritmos pode ser visto em BRANDES (BRANDES *et al*, 2003).

O *Clustering* baseado em grafos necessita ser executado várias vezes, além disso, não é claro quando se deve parar a execução do algoritmo. Como dados espaciais possuem um custo computacional diferenciado (devido suas particularidades espaciais), a execução deste método é inviável.

2.6. Considerações Finais

Neste capítulo abordou-se a conceituação de *clustering* e algoritmos de *clustering* conhecidos na literatura, sendo que estes são classificados mais tradicionalmente como *clustering* hierárquico e particionado. Além dos algoritmos destas classificações foram abordados os algoritmos de mapa auto-organizáveis e *clustering* utilizando grafos.

No *clustering* hierárquico, as distâncias entre todos os objetos são calculadas. A raiz do dendograma formado por esta abordagem é o par de pontos com a distância menor entre eles. Nos próximos passos a distância é recalculada e o agrupamento é formado segundo a ordem de uma árvore.

No *clustering* particionado do método K-médias, os clusters são formados em domínios numéricos, por particionamento em instâncias dentro de clusters disjuntos. Como o número de clusters deve ser definido antecipadamente, este método é mais específico para calcular poucos clusters. Outra desvantagem deste método é que todos os pontos são considerados pertencentes a um cluster, até mesmo os que não possuem uma grande similaridade.

No algoritmo QT um limiar de distância, entre os pontos, é informado. Depois de descoberto os objetos pertencentes a um cluster, estes são desconsiderados da iteração recursiva, diminuindo o desempenho. Apesar disso, este método depende muito recurso computacional, dependendo de quantas iterações forem feitas. No *clustering* utilizando

Mapas Auto-Organizáveis, os agrupamentos dependem de variáveis informadas pelo usuário. Já no *clustering* baseado em grafos, a medida de similaridade considera as conexões entre os vértices de um grafo orientado. O cálculo do peso entre os vértices e as várias iterações deste método deixa este algoritmo com baixo desempenho.

3. BANCO DE DADOS ESPACIAIS

Os Sistemas Gerenciadores de Banco de Dados (SGBD) convencionais não possuem dados para descrever objetos espaciais, como a geometria e as funções específicas para poder garantir a integridade e manipulação destes objetos. Os dados que um Sistema de Informação Geográfica (GIS ou SIG) manipulam representam objetos e fenômenos, cuja descrição da localização geográfica e sua forma de representação são as principais características utilizadas em sua análise.

“Para responder às consultas espaciais o banco de dados deve possibilitar a ordenação e o referenciamento dos dados, de maneira a recuperá-los agilmente. No caso de dados geográficos, o ordenamento deve estar baseado em chaves espaciais e significa que, diferentemente dos sistemas de gerenciamento de banco de dados convencionais, a ordenação e o referenciamento estão baseados no espaço ocupado pelos dados. As técnicas que possibilitam este ordenamento são conhecidas como métodos de índices espaciais”. (COSTA, 2003).

“Um Sistema de Bancos de Dados Espacial é um sistema de bancos de dados com capacidades adicionais que permitem representar, consultar e manipular dados cuja localização espacial é uma das principais características”. (QUEIROZ, 2003).

Todas essas informações podem ser extraídas de uma forma mais eficiente, se forem armazenadas dentro de um banco de dados espacial bem modelado, que levante várias informações tais como imagens de satélite, radar, polígono que delimita as cidades, estados, sensores de descargas atmosféricas ou qualquer outra informação que possa ser armazenada espacialmente.

Através da geometria de formação de raios (ponto central e a elipse de probabilidade de incidência de descargas atmosféricas), é muito mais simples e perceptível analisar dados, pois estes poderão não apenas ser consultados, mas também ser visualizados espacialmente em aplicativos específicos.

Com o armazenamento da geometria em forma de ponto, visualmente pode-se perceber a presença de acumulações de raios em certos períodos de tempo, em certas regiões. Visualmente isto é possível, mas, a operação para determinar a quantidade destas acumulações dentro de uma área especificada, atualmente não foi realizada.

A arquitetura de banco de dados geográfico e os dados espaciais serão discutidos na sequência.

3.1. Dados Espaciais

O Sistema Gerenciador de Banco de Dados (SGBD) evoluiu de acordo com as exigências que foram surgindo (SCHNEIDER, 1997). As maiores concentrações de pesquisa em banco de dado têm enfoque nas aplicações não convencionais. Entre as aplicações não convencionais se destaca o Sistema de Informação Geográfico - SIG, (QUEIROZ, 2004).

Pode-se ainda, apontar mais algumas definições de SIG (INPE, 2005): (i) Um conjunto manual ou computacional de procedimentos utilizados para armazenar e manipular dados georeferenciados (ARONOFF, 1989); (ii) Conjunto poderoso de ferramentas para coletar, armazenar, recuperar, transformar e visualizar dados sobre o mundo real. Os SIGs são capazes de gerenciar dados geográficos, ou seja, dados que descrevem entes do mundo real em termos de sua posição (em relação a um sistema de referência espacial), geometria, relacionamentos espaciais (relações topológicas) e atributos (BURROUGH, 1986); (iii) “Um sistema de suporte à decisão que integra dados referenciados espacialmente num ambiente de respostas a problemas” (COWEN, 1988); (iv) Um banco de dados indexados espacialmente, sobre o qual opera um conjunto de procedimentos para responder a consultas sobre entidades espaciais (SMITH *et al.*, 1987); (v) Os SIGs são sistemas utilizados para a representação computacional e análise de informações localizadas no espaço utilizado na meteorologia, astronomia e estudos ambientais (YATES, 1999).

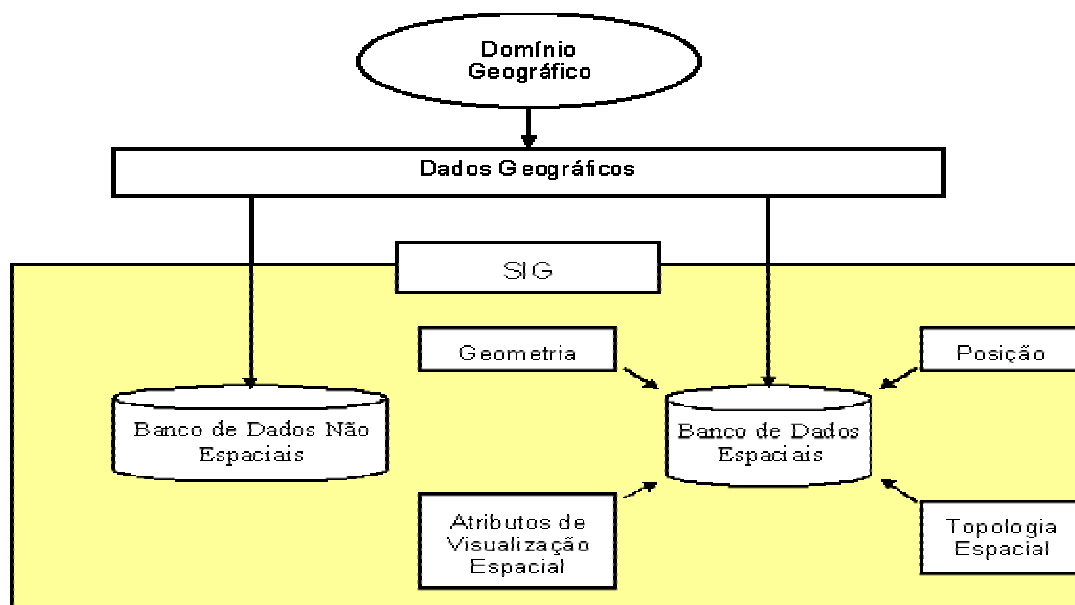


FIGURA 3.1: COMPONENTES BÁSICOS DE UM SIG
(Fonte: RAFAELI NETO; 2000)

A Figura 3.1 acima ilustra que o SIG abrange os dados espaciais provenientes da abstração do *domínio geográfico*. Desta abstração são extraídos *dados geográficos* que podem ser subdivididos em um *Banco de Dados Não Espacial* e *Banco de Dados Espacial*.

O Banco de Dados Não Espacial é formado por uma base de dados da forma convencional, isto é, não georeferenciada (sem dados da localização espacial como a latitude e longitude). Exemplos destes atributos do Banco de Dados Não Espacial são: nome de uma cidade, código, etc.

O Banco de Dados Espacial é formado por uma base espacial, isto é, a inserção de objetos geométricos (utilizados para representar os dados espaciais e está ligado ao objetivo da aplicação. Exemplos: ponto, polígono, etc.), atributos de visualização espacial (projeção), posição (georeferência de cada ponto) e funções de topologia espacial (relacionamento espacial mantém inalterados após distorções no mapa, tais como a conectividade e adjacência).

Estas definições de SIG refletem cada uma à sua maneira, a multiplicidade de usos e visões possíveis desta tecnologia e apontam para uma perspectiva interdisciplinar de sua utilização.

A principal diferença entre estes sistemas e os demais está na sua capacidade em armazenar e gerenciar dados espaciais e seus atributos (BURROUGH, 1986).

A Figura 3.2 indica o relacionamento dos principais componentes da arquitetura de um SIG.

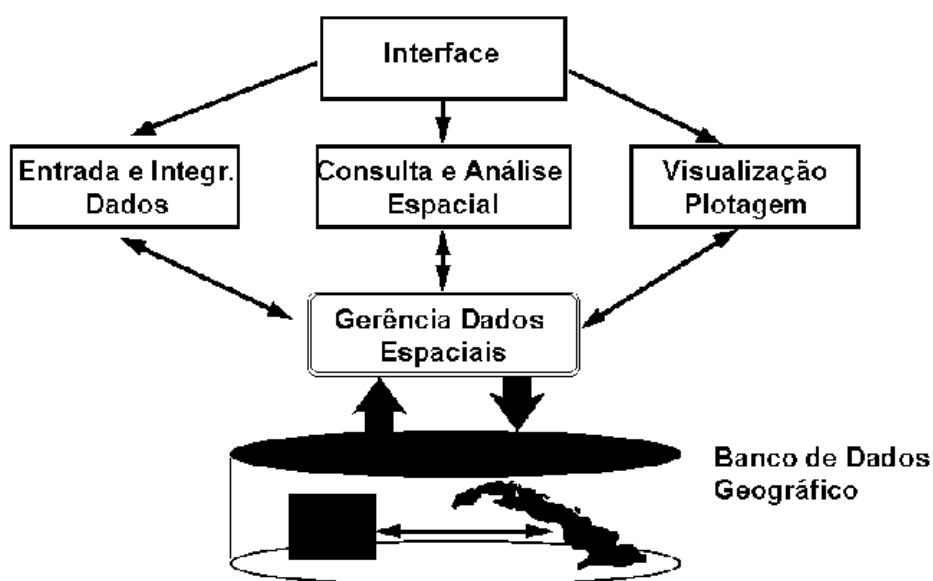


FIGURA 3.2: ARQUITETURA DE SISTEMAS DE INFORMAÇÃO GEOGRÁFICA
(Fonte: DPI-INPE, 2006; CÂMARA *et al*, 2003)

O SIG é formado por um Banco de Dados Geográfico (com dados espaciais e não-espaciais). Este Banco de Dados tem um componente que gerencia estes dados espaciais e faz a integração com a manipulação de dados tal como fazer uma consulta SQL, para extrair dados de forma não convencional. “O Processamento da Consulta Espacial consiste na interpretação (sintática e semântica) de uma consulta em linguagem de alto nível para instruções de baixo nível, gerando-se um plano estratégico de execução e diretrizes para o processamento”. (SILVA, 2002)

A gerência destes dados espaciais também deve manipular corretamente a entrada e a alteração de dados, bem como a integração com outras tabelas do banco de dados.

Outro componente desta arquitetura se refere a um subsistema de visualização, que oferece facilidades para apresentar os diferentes tipos de dados espaciais e suas representações.

O componente de interface tem por finalidade permitir a integração, manipulação e visualização desses dados.

Das aplicações SIG de interface e visualização pode-se citar:

i) Terra View, um aplicativo construído sobre a biblioteca de geoprocessamento TerraLib que manipula dados vetoriais (pontos, linhas e polígonos) e matriciais (grades e imagens), ambos armazenados em SGBD relacionais ou geo-relacionais, incluindo PostgreSQL, MySQL e Oracle (TERRA VIEW,2005);

ii) UMN MapServer: o servidor de mapas, gráficos é uma API para o desenvolvimento de aplicações SIG para a internet. *MapServer* utilizado para gerenciamento e análise de dados geo-espaciais, processamento de imagens e visualização (MAPSERVER,2005; GEOLIVRE,2006).

iii) Open 3D GIS: “Solução para SIG 3D com base em padrões abertos”. Utilizada as extensões 3D do PostGIS para armazenar a geometria. O formato X3D é utilizado como padrão para visualização” (GEOLIVRE,2006);

iv) GRASS (*Geographic Resource Analysis Support System*) (GRASS, 2006) é um software SIG livre com suporte a dados *raster*, topológicos, vetorial, processamento de imagens.

v) Spring: O SPRING é um SIG com funções de processamento de imagens, análise espacial, modelagem numérica de terreno e consulta a bancos de dados espaciais (GEOLIVRE, 2006);

vi) QuantumGis (*QGIS 2005*) é um software visualizador de dados geográficos que pode ser executado no Linux, Unix, Mac OSX (Apple ®), e Windows (Microsoft ®). O QGIS tem suporte a dados vetoriais, *raster* e formato de banco de dados. Possui poucos recursos para tratamento dos dados (vetoriais ou matriciais), mas permite acesso a uma grande variedade de dados vetoriais através da biblioteca OGR; e

vii) Thuban: Visualizador para dados de SIG escrito em Python. O Thuban possui uma interface amigável e alguns recursos úteis.

Cada sistema, em função de seus objetivos e necessidades, implementa estes componentes de forma distinta, mas todos os subsistemas citados estão presentes num SIG.

Os Bancos de Dados tiveram que ser adaptados para manipular dados associados a uma localização (de tempo e espaço) na superfície da terra. Este tipo de dado tende a ser mais complexo dos outros tipos e dados, devido à variedade de informações que são representadas: “Geralmente esta complexidade se deve ao formato digital imposto pelo próprio software que não manipula certas funções geométricas”. (BORGORNY, 1999).

O dado espacial, armazenado dentro de um banco de dados relacional, é referenciado dentro de uma coluna. Os dados espaciais são dados relacionados a objetos que ocupam espaço como tráfego, habitantes de uma cidade e clima. Estes tipos de objetos podem ser pontos, linhas e polígonos. A forma de armazenamento aponta algumas dificuldades, tal como a garantia de integridade entre a geometria e sua parte descritiva.

A manipulação de objetos complexos com linguagem de consulta e a extensão do tipo básico de SQL para objetos complexos (como é o caso de manipulações geométricas) surgiram os SGBD-OR (Sistema Gerenciador de Banco de Dados Objeto/Relacional).

Destes Bancos de Dados pode-se citar ainda algumas características particulares: funções como intersecção, disjunção, união e proximidade; uma linguagem de consulta estendida para manipulação de tipos de dados espaciais. Esse tipo de SGBD estende mecanismos de indexação sobre tipos espaciais.

Os mecanismos de indexação servem para acelerar o processamento de consultas espaciais através da organização de seus registros “(..) de tal maneira que objetos que estão mais próximos um dos outros no espaço correspondam a registros próximos um

do outro no índice.”- (QUEIROZ, 2003). São exemplos destes mecanismos de indexação: **R-Tree**, **QuadTree** e o mais utilizado, **GiST** (HELLERSTEIN *et al*, 1995).

O método de indexação **R-Tree** é uma extensão da árvore B para o espaço multidimensional, onde os objetos são aproximados através de seus retângulos envolventes mínimos (GUTTMAN, 1984).

O **QuadTree** subdivide o espaço em quadrantes. A árvore é formada por nós que possuem quatro descendentes que representam os quatro quadrantes nos quais o nó original foi subdividido. Os quadrantes serão divididos até que não são mais subdivididos, e então são armazenados em nós folha. Esse método pode ser aplicado a dados no formato matricial, pontos e objetos mais complexos tais como polígonos (SAMET, 1990; QUEIROZ, 2003).

O sistema de indexação espacial chamado *Generalized Search Tree (GiST)* é uma forma generalizada de uma indexação **R-Tree** (GUTTMAN, 1984). O **GiST** provê uma possibilidade de criar tipo de dados personalizados - “customizados” - com métodos de acesso indexados e a linguagem de busca extensível para domínio específico de consultas.

Formalmente, **GiST** é definido como uma estrutura de árvore balanceada com nodos contendo **(p, ptr)** pares, onde **p** é um predicado (atributo), que é usado como uma chave de busca, e **ptr** é o ponteiro para o dado em um nó folha ou um ponteiro para outro nó da árvore, para um nó não folha. **GiST**, ainda possui as seguintes propriedades: a) Para cada nó contém entradas de índices **Min** e **Max**, a não ser que este seja o nó raiz; b) Para cada entrada de índice **(p, ptr)** no nó folha, **p** é verdade quando instanciado com valores da tupla indicada; c) Para cada entrada do índice **(p, ptr)** em um nó não folha, **p** é verdade quando instanciado com valores de alguma tupla que pode ser alcançada por **ptr**; d) A raiz tem pelo menos dois filhos, a menos que este seja uma folha; e) Todas as folhas aparecem no mesmo nível. (GIST, 2006).

Um sistema de banco de dados espaço-temporal é capaz de arquivar e recuperar dados provenientes de diferentes fontes, tais como sensores, mapas temáticos, dados censitários e de cadastro urbano e rural, imagens de satélite e modelos numéricos de terreno.

Também se observa em um banco de dados estendido (CÂMARA, 2002):

i) Atributo topológico: a adjacência ou inclusão de informação como, por exemplo, relação com clima;

ii) Atributo não-espacial: também chamado de atributos alfanuméricos, que são conjuntos de atributos usados para descrever o dado geográfico de estudo, como por exemplo, nome e população de um país, pico de corrente de uma descarga atmosférica;

iii) Atributo geográfico: trata da posição (latitude/longitude), área, perímetro, polígono de fronteiras, como por exemplo, localização de estações meteorológicas, localização das descargas atmosféricas.

Queiroz (QUEIROZ, 2005; CAMARA, 2002:) ainda aponta algumas situações típicas em que as funcionalidades destes Gerenciadores de Banco de Dados utilizados nos SIGs: (i) seleção por apontamento, onde um usuário seleciona um determinado objeto através da interface gráfica ,por exemplo, descobrir as estações meteorológicas dentro de uma cidade; (ii) consultas espaciais que envolvem restrições espaciais entre os objetos, como determinar os municípios vizinhos de uma determinada cidade, consulta sobre a ocorrência de descargas atmosféricas em uma determinada região, distância entre cidades; (iii) criação de mapas de distância (*buffer zones*) ao redor dos objetos para determinar regiões de influência; (iv) sobreposição de polígonos para extração de informações, como por exemplo, a porcentagem da área de um município coberta por um determinado tipo de vegetação.

Estas funcionalidades se diferenciam no desempenho de resposta de consultas e possuem características mais complexas na sintaxe das consultas (*PSQL*), que fogem das linguagens padrão mais facilmente perceptível. Um exemplo de consulta *PSQL* é descrito a seguir:

```
SELECT MUNICÍPIO, ATEXT(M.THE_GEOM)
FROM MUNICÍPIO AS M, ESTADO AS UF
WHERE M.THE_GEOM && UF.THE_GEOM AND
        CONTAINS (UF.THE_GEOM,M.THE_GEOM.) AND
        UF.NOME = 'PARANA'
```

A consulta anterior possui características mais complexas em relação às consultas SQL convencionais. Aqui se percebe o uso do operador de sobreposição de geometrias

&& e a função espacial *contains*, onde uma geometria está contida dentro da outra. Neste caso, o município está contido na geometria do Estado do Paraná.

Todas as situações podem ser utilizadas para descobrir mais precisamente aonde ocorreram as descargas atmosféricas. Para isso, objetos específicos, por exemplo, o tipo ponto, é utilizado para todas as manipulações. A localização exata da espacialização do ponto central da descarga (ponto) pode ser feita com a combinação da espacialização dos municípios (polígono). O cruzamento destas informações de município e descarga atmosférica, e as funções de topologia da extensão espacial mostram o município onde aconteceu a incidência.

Os tipos geométricos, bem como alguns métodos espaciais que podem ser utilizados no exemplo anterior foram abordados na Seção 4.3. A listagem de todas as atuais funções implementadas no *PostGis* e *Oracle Spatial*, definidas pelo *Open Gis Consortium* (OGC, 2006), se encontra na página oficial do *Postgresql/PostGis* (RAMSEY, 2005) e da *Oracle/Oracle Spatial* (ORACLE SPATIAL, 2006).

Os objetos espaciais utilizados para estas manipulações são abordados na seção seguinte.

3.2. Objetos Espaciais

O Consórcio OpenGIS (OGC, 2005 e *OpenGIS Consortium*, 1999) –*Open Geodata Interoperability Specification* é uma organização dedicada à especificação de sistemas de geoprocessamento abertos. Todas essas extensões são baseadas nas especificações da Norma *SQL/MM Spatial* (ISO/IEC, 1999), cujas especificações são baseadas na SFSSQL (*Simple Features Specification for SQL*) do OpenGIS (OGIS, 1995): “O propósito da sua especificação SQL é definir um esquema SQL padrão que suporte o armazenamento, recuperação, consulta e atualização de coleções de dados geo-espaciais via ODBC (*Open Database Connectivity*) API” (CORTEZ, 2002). Em termos práticos, “a ODBC permite que uma aplicação acesse simultaneamente uma variedade de SGBDs, relacionais e não relacionais, situados em locais diferentes, através de uma única API. Além disso, ela padroniza as mensagens de erro, a interface de conexão com as fontes de dados, os tipos de dados e, até mesmo, a linguagem SQL.” (MARTINS, 1996)

O Consórcio OpenGis também padroniza construções topológicas avançadas tais como *Coverage*, *Surface* e *Network* de forma genérica *Coverage*, ou cobertura, inclui imagens, mapas, campos. *Surface*, ou superfície é tratada com um objeto geométrico de duas dimensões com um limite exterior representado por um polígono plano. *Network* é formado por uma quantia finita de pontos que formam os nós de uma rede. É formado por um conjunto de linhas que iniciam e terminam em nós distintos desta rede (BÜCHNER, 2002).

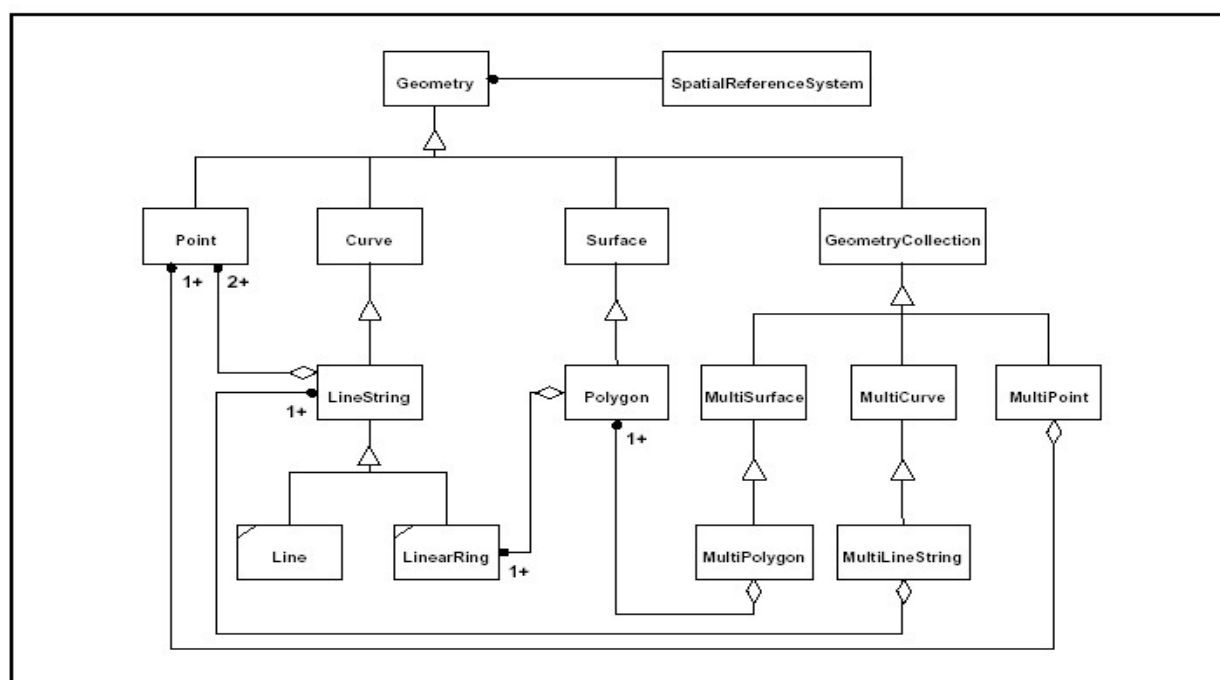


FIGURA 3.3 HIERAQUIA DE CLASSE GEOMÉTRICA
(Fonte: OGIS, 1999)

A Figura 3.3 é baseada na extensão do Modelo Geométrico especificado pelo *OpenGis* (OGIS, 1999), onde a geometria (*Geometry*) é a classe raiz da hierarquia. Para cada geometria há vários métodos que retornam informações.

Curve representam objetos de uma dimensão, geralmente armazenada como uma sequência de pontos, e tem a subclasse ***LineString***, com subclasses ***Line*** e ***LinearRing***. *Surface* é criado para objetos bidimensionais e tem a subclasse ***Polygon***. *GeometryCollection* tem classes de coleção com zero, uma e duas dimensões chamadas ***MultiPoint***, ***MultiLineString*** e ***MultiPolygon*** para modelagem geométrica correspondente a coleções de *Points*, *LineStrings* e *Polygons*, respectivamente.

A ***MultiCurve*** e ***MultiSurface*** são introduzidas como superclasses abstratas que generalizam a interface de coleção para tratar *Curves* e *Surfaces*. ***Geometry***, ***Curve***,

Surface, *MultiCurve* e *MultiSurface* são definidas como classes não instanciáveis. Elas definem em conjunto de métodos comuns para suas subclasses e são incluídos por razões de extensibilidade. *Point*, *LineString*, *Polygon*, *GeometryCollection*, *MultiPoint*, *MultiLineString*, *MultiPolygon* são classes instanciáveis (MySQL, 2006).

A *Geometry* é uma classe abstrata, isto é, não instanciável. Todas as classes geométricas instanciáveis são definidas para que as instâncias válidas da classe *geometry* sejam topologicamente fechadas (isto é, todas as geometrias definidas incluem seus limites).

A geometria tem o significado original provindo do ramo da matemática. Outro significado para esta palavra provém da cartografia, se referindo aos recursos geométricos para mapear o mundo. A geometria é definida como um ponto, ou um agregado de pontos representando uma localização de um objeto (MySQL, 2006).

A geo-referência adota um sistema de coordenadas (*Sistema de Referência Espacial*) que pode ser usado para expressar a posição do objeto com relação à superfície da terra. O uso de latitude e longitude recai como forma de expressar o espaço.

Os dados coletados geralmente estão em diferentes modelos de representação da Terra, isto é, sistemas de projeção, em consequência disso, suas posições podem divergir e os sistemas de informações devem prover as operações matemáticas necessárias para o tratamento destas questões e a integração de dos resultados em um único sistema de projeção através da conversão de dados (COSTA, 2003).

Para que todos os objetos relacionados sejam referenciados da mesma forma, é utilizado um *SRID* (Identificador Espacial). O *SRID* utiliza um ponto inicial *DATUM*, que é um conjunto de valores que podem servir como referência ou base de cálculo para outros valores, como medição (ISO/IEC, 1999), para que as projeções geométricas (ângulo que permite melhor visualização de um mapa) sejam realizadas.

Tipo *Point* (Ponto)

O ponto é uma geometria que possui dimensão zero e representa uma posição simples no sistema de coordenadas. Um ponto tem um valor da coordenada x e outro valor da coordenada y.

Exemplo de dados que podem ser inseridos dentro de um banco de dados espacial é a geometria em forma de ponto, que pode ser bidimensional ou até mesmo

tridimensional. A Figura 3.4 ilustra um objeto do tipo espacial: POINT (0 0) com coordenadas X e Y :

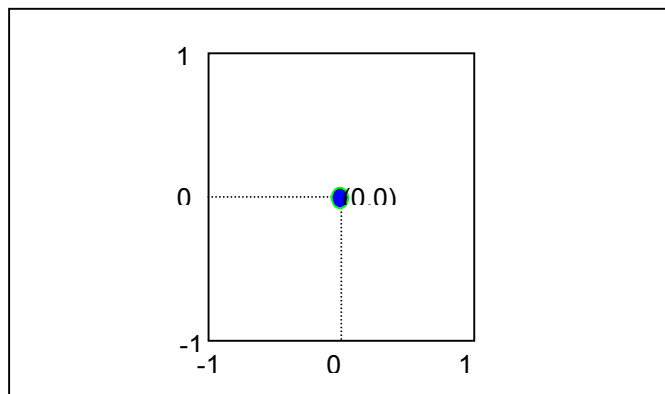


FIGURA 3.4: OBJETO ESPACIAL DO TIPO POINT

Fonte: (BÜCHNER, 2002)

O tipo de objeto *point* pode ser utilizado para armazenar a posição relacionada a altitude, latitude e longitude de um ponto. A localização de uma descarga elétrica (tema deste estudo), localização das sedes de um município, localização de uma estação meteorológica são alguns exemplos.

Tipo *Linestring* (Linha)

Uma Linha é uma geometria que possui dimensão 1 e é formada por um conjunto de pontos. O tipo de dados *Linestring* pode ser visto como conjunto de linhas (com início x e final y):LINESTRING (x y, x y,x y). A Figura 3.5 ilustra o tipo LINESTRING (0 0,1 1, 1 2).

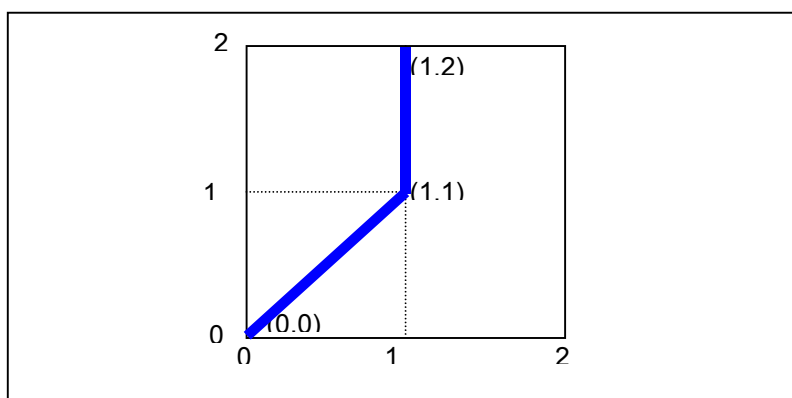


FIGURA 3.5: OBJETO ESPACIAL DO TIPO LINESTRING

Fonte: (BÜCHNER, 2002)

Este tipo pode ser utilizado para “especializar” (representar) as estações de distribuição e transmissão de energia elétrica. Este tipo de informação é interessante

para descobrir se raios ocorreram perto destas estações, facilitando assim laudos técnicos.

Tipo *Multilinestring* (Multi-linhas)

O tipo *multilinestring*, como o próprio nome descreve, se trata de muitos *linestrings*. Este tipo de objeto também pode ser obtido realizando uma operação de *Collect* para *linestring*. A sintaxe é *MULTILINESTRING*: ((*linestring* 1),(*linestring* n)[,]...) A Figura 3.9 exemplifica este tipo. Para *MULTILINESTRING*: ((0 0 0,1 1 0,1 2 1),(2 3 1,3 2 1,5 4 1))

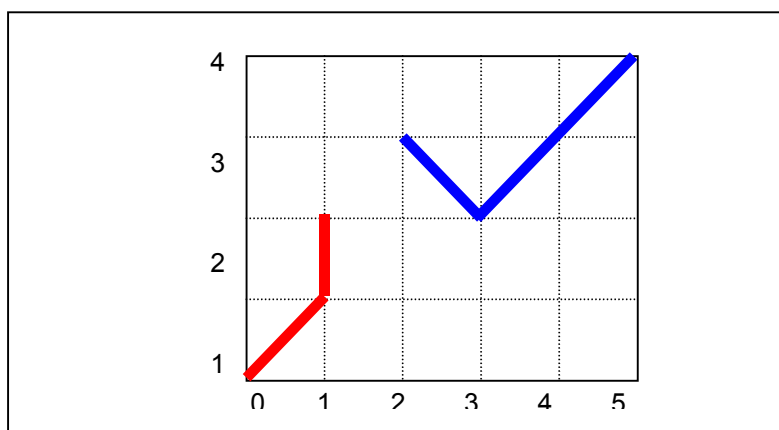


FIGURA 3.6: OBJETO ESPACIAL DO TIPO MULTILINESTRING

Fonte: (BÜCHNER, 2002)

Este tipo pode ser utilizado para descrever a geometria que delimita municípios, ou o envoltório de vários pontos, densidade de descargas elétricas, por exemplo.

Tipo *Polygon* (Polígono)

O objeto polígono (*Polygon*) possui coordenadas X,Y,Z que é formado pelo limite exterior e o limite interior. A sintaxe é: *POLYGON* (X Y Z, X Y Z). A Figura 3.7 ilustra este tipo com as seguintes coordenadas:

POLYGON ((0 0 0,4 0 0,4 4 0,0 4 0,0 0 0),(1 1 0,2 1 0,2 2 0,1 2 0,1 1 0));

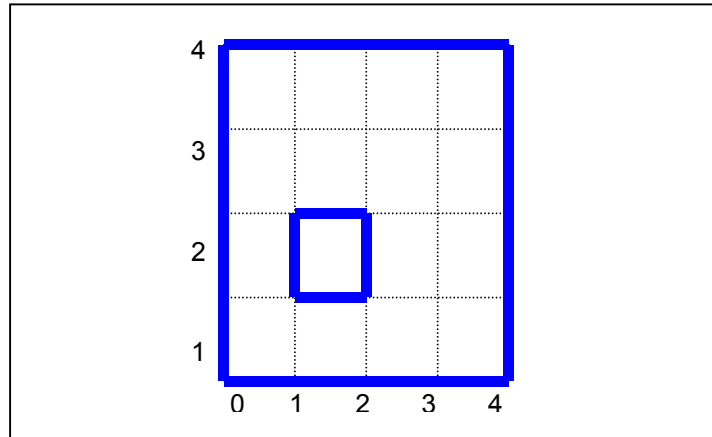


FIGURA 3.7: OBJETO ESPACIAL DO TIPO POLYGON
 Fonte: (BÜCHNER, 2002)

Este tipo é utilizado para limitar uma grande área de descargas elétricas. Este tipo é o formado pela função envelope do *Postgis*. A função envelope expande todas as áreas de uma geometria, formando um retângulo envolvente.

Tipo *Multipolygon* (Multi-polígono)

O tipo *multipolygon* é utilizado para descrever muitos polígonos. A exemplo do tipo *multilinestring*, *multipolygon* também pode ser resultante da função *collect* para polígonos. A Figura 3.8 mostra como é visualizado o tipo *multipolygon*:
`MULTIPOLYGON(((0 0 0,4 0 0,4 4 0,0 4 0,0 0 0)),(1 1 0,2 1 0,2 2 0,1 2 0,1 1 0)),((-1 -1 0,-1 -2 0,-2 -2 0,-2 -1 0,-1 -1 0)));`

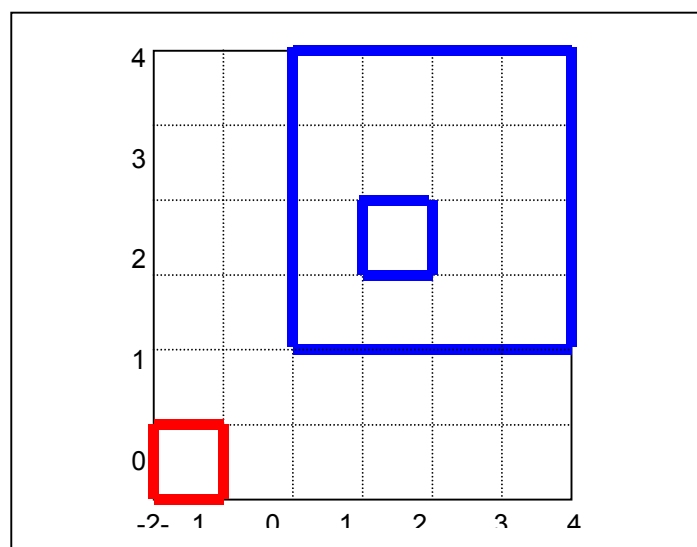


FIGURA 3.8: OBJETO ESPACIAL DO TIPO MULTIPOLYGON
 Fonte: (BÜCHNER, 2002)

Um outro tipo de objeto é especificado dentro do OpenGIS (OGIS,1995): o ***Geometrycollection***, que, como o próprio nome diz, é a coleção de geometrias, por exemplo, ***Linestring*** e ***Multipolygon***.

Os objetos espaciais possuem características únicas, nas quais alguns métodos podem ser aplicados. Os principais métodos aplicados a estes objetos são descritos na sequência.

3.3. Métodos Espaciais

Os principais métodos espaciais que manipulam as geometrias são descritos a seguir (CORTEZ, 2002):

1) ***Dimension(): integer***

A Função *Dimension()* retorna a dimensão herdada de um dado objeto geométrico.

2) ***GeometryType():String***

A Função *GeometryType()* retorna o nome do subtipo instanciável da Geometria, da qual uma dada instância de Geometria é membro. O nome do subtipo instanciável da Geometria é retornado como *string*.

3) ***Boundary():Geometry***

A Função *Boundary()* retorna os limites de uma dada Geometria.

4) ***SRID():Integer***

A Função *SRID()* retorna a Identificação do Sistema de Referência Espacial para uma dada Geometria.

5) ***IsEmpty(): Integer***

A Função *IsEmpty()* retorna 1 (Verdadeiro) se a geometria está vazia.

6) ***AsText():String***

A Função *AsText()* retorna a geometria como uma representação textual bem formada.

7) ***AsBinary():Binary***

A Função *AsBinary()* retorna a geometria como uma representação binária bem formada.

Os métodos para representação topológica são os que seguem.

1) ***Equals (another Geometry: Geometry): Integer***

A Função *Equals* retorna 1 (Verdadeiro) se uma dada geometria for espacialmente igual a uma outra geometria, ou seja, representada pelo mesmo tipo geométrico.

2) ***Disjoint (another Geometry:Geometry): Integer***

A Função *Disjoint* retorna 1 (Verdadeiro) se uma dada Geometria for espacialmente disjunta de uma outra Geometria.

3) ***Crosses(another Geometry:Geometry)***

A Função ***Crosses*** retorna **1** (Verdadeiro) se uma geometria cruza espacialmente outra geometria. Retorna **NULL** se a geometria é um ***Polygon*** ou um ***MultiPolygon***, ou se a outra geometria é do tipo ***Point*** ou ***MultiPoint***. Senão **0** é retornado.

O termo cruza espacialmente denota uma relação espacial entre duas geometrias que têm as seguintes propriedades: a) As duas geometrias se interceptam; b) A interseção resulta em uma geometria que tem uma dimensão que é menor que a dimensão máxima das duas geometrias dadas; C) A interseção não é igual a nenhuma das duas geometrias dadas.

4) ***Intersects (another Geometry:Geometry):Integer***

A Função ***Intersects*** retorna **1** (Verdadeiro) se uma dada geometria intersectar espacialmente uma outra Geometria.

5) ***Touches (another Geometry:Geometry):Integer***

A Função ***Touches*** retorna **1** (Verdadeiro) se uma dada geometria toca espacialmente uma outra Geometria. Duas geometrias se tocam espacialmente se o interior de ambas geometrias não se interceptam, mas o limite de uma delas intercepta o limite ou o interior das geometrias.

6) *Within (another Geometry:Geometry): Integer*

A Função ***Within*** retorna **1** (Verdadeiro) se uma dada geometria está contida espacialmente na outra Geometria.

7) *Contains (another Geometry:Geometry):Integer*

A Função ***Contains*** retorna **1** (Verdadeiro) se uma dada geometria contém espacialmente uma outra Geometria.

8) *Overlaps (another Geometry:Geometry): Integer*

A Função ***Overlaps*** retorna **1** (Verdadeiro) se uma dada geometria sobrepõe espacialmente uma outra Geometria.

9) *ConvexHull(geometry)*

A Função ***ConvexHull*** retorna uma geometria que representa o envoltório (“casca”) convexa desta geometria.

10) *Collect*

A Função ***Collect*** retorna a reunião de diversas geometrias, sendo elas do mesmo tipo ou não. A função agrupa todas as geometrias informadas na função.

Os métodos que manipulam geometrias servem para retornar, inserir o valor de um atributo de geometria. No método criado nesta dissertação foi utilizado ***AsText*** (como texto) para retornar a geometria no formato texto.

Os métodos de representação topologia são aqueles utilizados para a realização de operações com conjuntos. O método de detecção de cluster utiliza ***Contains***, e como aplicação, ou variação do método, pode-se utilizar ***Overlap***, ***ConvexHull***, ***Collect***, entre outros.

Para a validação do método de detecção de *clustering* uma modelagem de banco de dados foi formulada, bem como a aplicação diferenciada de Banco de Dados convencionais e as consultas espaciais sobre estas diversas informações referentes a raios.

Uma aplicação na tentativa para a criação deste método foi feita em Java e foram utilizadas bibliotecas de algoritmos de *clustering* já testadas no Sistema Weka, da Universidade de Waikato. Outra tentativa para a criação do método foi feita com as próprias funções presentes dentro de SGBD com extensão espacial, tal como as funções de *distance_sphere*, *buffer*, *makeline*. O método em si também utilizou as funções presentes da extensão, porém com metodologia diferente. A metodologia empregada no método pesquisa a distância da vizinhança, e não a distância de cada ponto, além disso, as funções espaciais empregadas foram *contains* e *buffer*.

O método de *clustering* desenvolvido para este tipo de Banco de Dados também poderá ser utilizado para responder questões sobre padrões de incidência de raios, através da determinação da área onde mais ocorrem, levando em consideração aspectos espaciais, de forma organizada e ágil. Com estas informações armazenadas de forma espacial, haverá mais argumentos para apoiar esta decisão. As funções, a metodologia, bem como as conclusões são abordadas no Capítulo 4. Na Seção 3.4. avalia alguns SGBDs com extensão utilizando as particularidades espaciais.

3.4. Alguns SGBDs e o Enfoque Espacial

Os SGBDs de uma forma geral evoluíram da família dos Sistemas de Gerenciamento de Dados Relacionais (SGBD-R) (DATE, 2000), passando pelos Sistemas Objeto-Relacionais (SGBD-OR) (SILBERSCHATZ *et al*, 1999).

Atualmente, existem basicamente três extensões comerciais disponíveis no mercado para tratar de dados geográficos no formato vetorial: (1) *Oracle Spatial* (RAVADA E SHARMA, 1999), (2) *IBM DB2 Spatial Extender* (IBM, 2002) e (3) *Informix Spatial e Geodetic Datablade* (IBM, 2003).

No universo do software de código fonte aberto e gratuito existem dois projetos para a criação de extensões espaciais, um baseado no SGBD *PostgreSQL* (STONEBRAKER *et al*, 1990) chamado de *PostGIS* (RAMSEY, 2004) e outro baseado no *MySQL* (YARGER *et al*, 1999). A seguir, são apresentadas as características e funcionalidades das extensões da *Oracle*, do *PostgreSQL* e do *MySQL*.

3.4.1. Oracle e Oracle Spatial

O *Oracle Spatial* (MURRAY, 2003; ORACLE SPATIAL, 2006) é uma extensão espacial do SGBD *Oracle*, e os dados são organizados conforme o modelo objeto-relacional, que permite definir novos tipos de dados. Esta extensão contém um conjunto de funcionalidades e procedimentos que permitem armazenar, acessar e analisar dados espaciais em um banco de dados *Oracle*. O Oracle Spatial está disponível na distribuição *Oracle Enterprise Edition*.

Segundo Queiroz (QUEIROZ, 2003), O Oracle Spatial é formado pelos seguintes componentes: (i) Implementa o Modelo Objeto-Relacional através de um modelo próprio de dados chamado MDSYS que define a forma de armazenamento, a sintaxe e semântica dos tipos espaciais suportados. (ii) Mecanismo de indexação espacial. (iii) Um conjunto de operadores e funções para representar consultas, junção espacial e outras operações de análise espacial. (iv) Aplicativos administrativos.

Silva (SILVA, 2002) aponta características da arquitetura de Banco de Dados Integrado: a) O *Oracle Spatial* possui uma forte ligação entre dados espaciais e não espaciais. b) Herda todos os recursos de banco de dados, tais como segurança de acesso ao dado, backup, recuperação da informação, proteção contra falhas, entre outros. A consulta à base de dados envolve recuperação de ambos os tipos de dados e um único processo é disparado, de modo a aumentar a eficiência da consulta e evita problemas como a perda da integridade referencial do Banco de Dados.

Pode-se citar algumas características deste SGBDE (ORACLE SPATIAL OPTION, 2006): (i) padrão sql de acesso para todas as funções e operadores; (ii) tipo objeto espacial armazenado de acordo com tipo geométrico e referência linear; (iii) Operadores espaciais e funções incluindo funções de agregação (e.g. unions); (iv) indexação ***R-Tree*** e ***QuadTree***. Suporte e particionamento de índice espacial. (v) sistema de referência linear. (vi) ferramentas de integração de dados heterogêneos (fusão), incluindo gerenciamento de projeção e transformação de coordenadas; (vii) integração com Oracle9i *Application Server MapViewer* (visualizador do Oracle®).

Seu modelo de dados consiste em uma estrutura hierárquica de elementos, geometrias e camadas de informação. O *Oracle Spatial* suporta três tipos primitivos de geometrias (ponto, linha e polígono) e uma coleção de geometrias derivadas destes tipos primitivos, tal como arcos circulares, círculos, linhas compostas e polígonos compostos (SILVA, 2002).

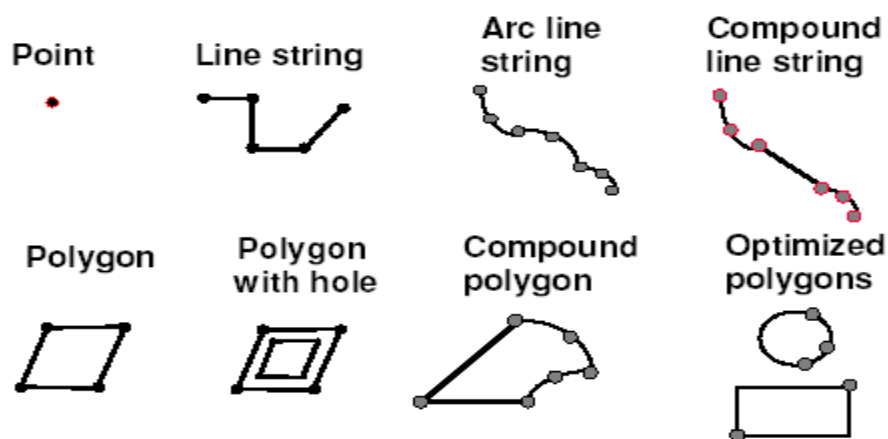


FIGURA 3.9: ELEMENTOS GEOMÉTRICOS DO ORACLE SPATIAL (QUEIROZ,2003)

No nível mais baixo desse modelo têm-se os elementos que são tipos geométricos primitivos (Figura 3.9): Point, Polygon, Polygon with hole, Line string, Arc line String, Compound Line string, Compound Polygon e Optimized Polygons. Todos estes tipos geométricos estão em conformidade com o padrão proposto pelo Open Gis.

O Oracle está em conformidade com o SFSSQL, embora possua discordância em relação à nomenclatura original das funções e procedimentos. Isso se deve, provavelmente, por razões históricas e de compatibilidade, o que limita a interoperabilidade com as funções propostas pelo padrão da OGC.

3.4.2. PostgreSQL e a Extensão PostGIS

O *PostgreSQL* (POSTGRES,2005), foi desenvolvido através de pesquisas em Banco de Dados da Universidade da Califórnia em Berkeley. O predecessor *Ingres* “criou” a tecnologia relacional; o *Postgres* (*Post-Ingres*) focava a tecnologia de objetos. Projetos independentes acrescentaram o suporte a SQL, gerando o *PostgreSQL*.

Um grupo de estudantes do professor Michael Stonebraker (STONEBRAKER *et al*, 1990) de Berkeley assumiu a evolução da versão SQL para atender às demandas do meio empresarial. Implementação completa dos padrões ANSI para o SQL, como *selects* aninhados/correlacionados, integridade referencial, visões, gatilhos (*triggers*), extensões orientadas a objeto, como tipos abstratos de dados, herança e atributos multivalorados (LOZANO, 2002).

Entre as características do *PostgreSQL* destacam-se (DOUGLAS,2003; LOZANO, 2002):

- a) Objeto-relacional, cada tabela define uma classe, que possui herança entre tabelas e as funções e operadores são polimórficos.
- b) Sintaxes padrão *SQL92* e *SQL99*, com código aberto, integridade referencial (suporte relacionamento de chaves estrangeiras e chaves primárias, bem como *triggers*, linguagens procedurais como *PL/PGSQL*, *TCL*, *PERL*).
- c) Múltiplos-clientes *API* – aplicações cliente de muitas linguagens (*C*, *C++*, *ODBC*, *PERL*, *PHP*, *TCL/TK* e *PYTHON*, *JAVA*). Dá suporte a criação de funções em Linguagem *PLJAVA* (consulta para programas Java), inseridos em formato *SQLJ*, e extensão com a criação de objetos em *C*.
- d) Uso intensivo de memória compartilhada e semáforos – pode ser necessário ajustar parâmetros do *kernel* para melhorar o desempenho ou para atender a grandes cargas de trabalhos;
- e) Tabelas e índices correspondem a arquivos físicos e bancos de dados correspondem aos diretórios. Vários diretórios diferentes podem ser configurados para hospedar (arquivos de) tabelas e índices;
- f) Extensibilidade – uma das mais importantes características é que ele pode se estender, como por exemplo, tipos dados espaciais – GIS.

O *PostgreSQL* é suportado para servidores e clientes: sistemas *Unix-like GNU/Linux* (GNU,2006), *FreeBSD* (FREE BSD,2006), *Solaris* (Sun ®), *AIX* (IBM ®), *HP-UX* (HP®), e versões experimentais para *Windows NT/2000/XP/2003* (*Cygwin*)(*Microsoft* ®), *MacOS X* (Apple Computer Inc) e *Netware* (Novell®). O *PostgreSQL* também possui *Drivers ODBC*, *OLEDDB* e *dbExpress* nativos . Driver *JDBC* totalmente Java.

Algumas características de SGBDEs foram desenvolvidas para resolver problemas que os SGBDs convencionais não resolviam. Estas características tão particulares não dependem de chaves estrangeiras como atributos das tabelas para serem executadas. Os SGBDs convencionais não possuem, por exemplo, funções que descobrem se um ponto está contido em uma geometria ou não (*truly-inside*) e não possuem funções para de cálculos como a distância entre dois pontos (*distance*). De uma forma geral, os SGBDs convencionais não são capazes de manipular informações geométricas (BÜCHNER, 2002).

O *PostGis* é um módulo de extensão do SGBD *PostgreSQL*, desenvolvida e mantida desde 1986 pela empresa canadense *Refractions Research Inc*, especializada em consultoria em SIG e banco de dados e em, desenvolvimento de softwares para integração de dados (BUCHNER,2002, RAMSEY, 2005) como alternativa ao *Oracle Spatial*.(ORACLE SPATIAL, 2006). Essa extensão permite armazenar e gerenciar objetos espaciais, dar suporte para índices e trabalha com diversas funções para geometria. A indexação espacial é feita através de uma *R-Tree* nativa ou através do mecanismo de indexação *GiST* (HELLERSTEIN *et al*, 1995).

3.4.3. MySql

Outro SGBD que entra no mercado de *software* livre para SIG como resposta às extensões comerciais atuais é o *MySQL*. Este SGBD pode ser distribuído tanto com licença comercial quanto com uma licença integrada baseada na GNU/GPL. Em abril de 2003, a sua extensão espacial foi disponibilizada (versão 4.1.0).

Atualmente, a extensão espacial desse SGBD encontra-se em construção, onde as características de geração, armazenamento e análise de dados espaciais estão disponíveis apenas para tabelas *MyISAM* (MySQL, 2006). O seu projeto segue as especificações da *SFSQL*, já explanadas anteriormente. Os únicos recursos disponibilizados até o momento são os tipos de dados espaciais semelhantes aos fornecidos pelo *PostGis* (*Point*, *LineString*, *Polygon*, *MultiLineString*, *MultiPolygon*, *MultiPoint* e *GeometryCollection*). Além disso, o mecanismo de indexação utiliza Árvore R com separação quadrática para indexar colunas espaciais (através de *R-Tree*).

Os grupos de funções, de acordo com a operação que eles realizam dentro do MySQL são(Manual de Referência do MySQL,2006): a) Funções que convertem geometrias entre vários formatos, como por exemplo, *line* para *multiline*. b) Funções que fornecem acesso a propriedades qualitativas ou quantitativas de uma geometria c) Funções que descrevem relações entre duas geometrias.d) Funções que criam novas geometrias de outras existentes. e) Funções de análise espacial que podem ser usados em muitos contextos, tais como: qualquer programa SQL interativo, como *MySql* ou *MySQLCC*. f) Aplicativos escritos em qualquer linguagem suportando uma API do cliente MySQL. Os grupos *a,b,c* e *d* descritos acima também foram contemplados pelo Oracle e *Postgresql*.

Entre as características ausentes nessa extensão e que constam na especificação do *OGIS* encontram-se as *views* de metadados das colunas geométricas, funções de criação e remoção das colunas geométricas e operadores espaciais.

Ainda não foi implementado, segundo as especificações do *OpenGis*, funções especiais para adição “*add*” e remoção “*drop*” de colunas com tipo geométrico (eg. *AddGeometryColumn()* e *DropGeometryColumn()*). O *MySQL* faz a remoção da forma convencional, isto é, com uso de *Alter Table*, *Create Index* e *Drop Index*.

Essa extensão também não fornece nenhum recurso para conversão entre sistemas de coordenadas e Funções que Testam Relacionamentos Espaciais Entre Geometrias (como *contains*, *crosses*). Estas funções que não foram implementadas no *MySQL* devem aparecer em distribuições a serem lançadas no futuro.

Essa extensão apresenta duas desvantagens em relação ao *PostGis*. Primeiro, as colunas geométricas devem ser criadas com a restrição de não conter valores nulos (*NOT NULL*) para que possam ser indexadas através do índice espacial. Segundo, a extensão só está disponível para o tipo de tabela *MySQL*, o que significa a ausência de recursos de transação.

Embora o *MySQL* ter seguido as especificações do *SFSQL*, a extensão espacial apresenta algumas limitações (MELO e GUERRA, 2005). As principais são: i) só está disponível para o tipo de tabela *MyISAM* que é própria do *MySQL*. Neste caso não existem mecanismos de controle de transação e de integridade referencial, o que limita a sua aplicabilidade. ii) apenas um número reduzido de produtos na área de geoprocessamento faz interface com o *MySQL*. Isto delimita o desenvolvimento de sistemas.

3.5. Considerações Finais

Os tipos espaciais básicos de SGBDE definidos pelo consórcio *OpenGis*- *Point*, *MultiPoint*, *Linestring*, *MultiLinestring*, *Polygon* e *MultiPolygon* foram implementados nas três extensões. Até a versão mais atual, do *PostGis* 8.1.3 (*PostGis*, 2006), bem como o *MySQL* não apresentam o tipo *Curve* e *Multicurve*, implementados pelo *Oracle*.

O *Oracle Spatial* é uma extensão espacial disponível na distribuição *Oracle Enterprise Edition*. Quanto a SGBDs de código aberto, o *PostGis* se mostra mais completo que o *MySQL*. O *PostGis* se destaca na tabela de metadados das colunas geométricas (tabela *geometry_columns*), com suporte a softwares livres como *Geotools*,

e *Toolkit Java*, *Mapserver*. É o único que segue padrão o especificado pelo *OpenGis*. O *PostGis* e o *Oracle* possuem características como suporte *MapServer*, *JDBC* que o *MySQL* ainda não implementou.

O *MySQL* está numa fase muito inicial da implementação, faltando definições importantes especificadas pelo *OpenGis*, tal como funções *AddGeometryColumn()* e *DropGeometryColumn()*. Estas funções validam a adição e remoção de colunas geométricas, e que diferenciam dos tipos nativos de um banco de dados convencional, como, por exemplo, tipo *Box*. Os tipos nativos não possuem suporte às funções espaciais de uma extensão.

Outra especificação importante que o *MySQL* ainda não implementou é a especificação da identificação espacial (*SRID*), sendo assim, todos os objetos estão considerados no mesmo sistema de coordenadas, e sem um *Datum* como base para projeções.

Em relação a utilização, o *MySQL* não possui formato externo para migração e *upgrade* de dados espaciais, isto é, os arquivos de dados são carregados apenas com declarações *insert*, não possuindo *loader*. Sendo assim, as operações de *insert* possuem baixa performance (PAGE, 2004). *Oracle Spatial* e *PostGis* possuem esta utilidade (*SQL Loader* e *Shp2Pgsq*l).

4. MÉTODO DE REPRESENTAÇÃO ESPACIAL DE CLUSTERING PARA BANCO DE DADOS

O método de detecção de cluster, criado para dados espaciais, busca identificar aglomerados de pontos dentro de uma distância definida. Em todo resultado de algoritmo de cluster, os dados aglomerados geralmente possuem características próximas ou em comum. Estas características podem ser vistas, por exemplo, a proximidade espacial - significando um padrão entre estes dados.

O método busca criar a viabilidade, isto é, há uma forma de detectar *clusters* dentro de banco de dados. O método é apenas um primeiro passo objetivando a inserção de uma nova funcionalidade para os SGBDEs.

Um método de detecção de cluster pode ser mais bem compreendido através de um exemplo: dentro da meteorologia deseja-se saber se num evento de tempestade elétrica se há núcleos de descargas atmosféricas presente.

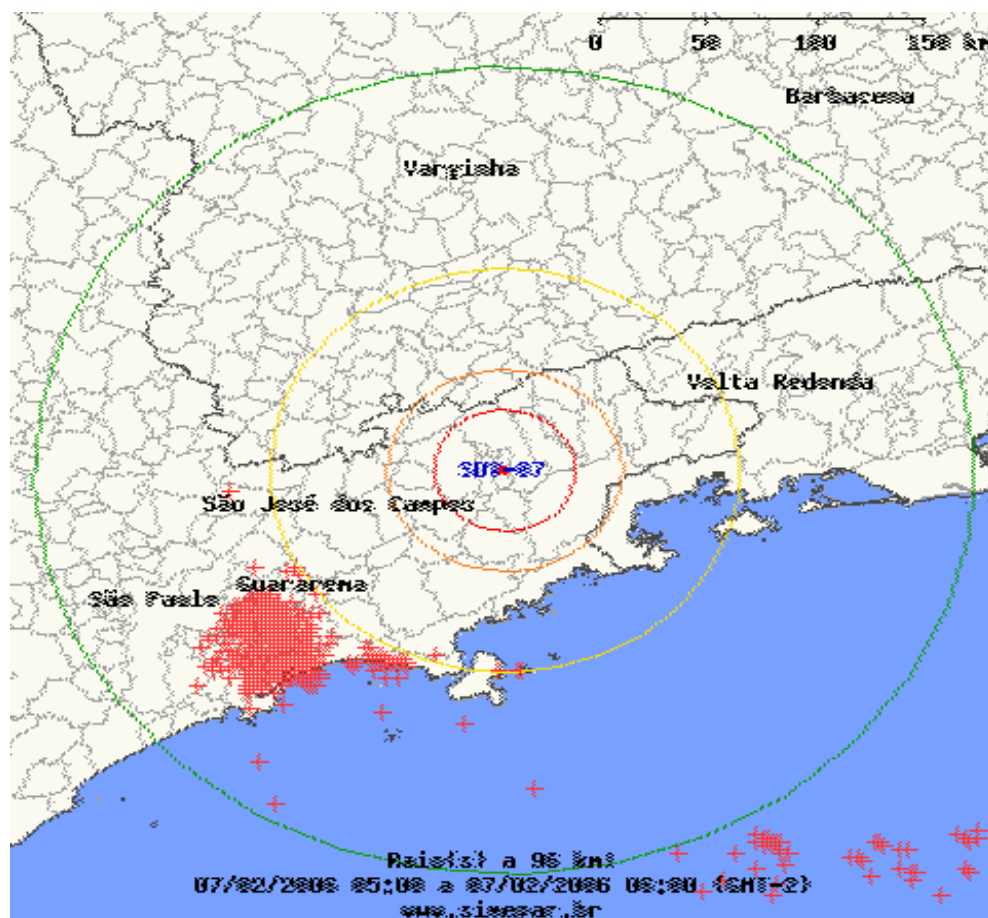


FIGURA 4.1: PRIMEIRO ACOMPANHAMENTO DE DESCARGAS ATMOSFÉRICAS
Fonte: (SIMEPAR, 2006)

A Figura 4.1 ilustra o monitoramento de tempestades, a partir de um ponto central, neste caso uma refinaria na região de São Paulo. Cada círculo ao redor deste ponto central possui 30, 50, 100 e 200 km. Todas as informações plotadas nesta figuras foram geradas, acessando o SGBDE PostGIS, através do Mapserver (MAPSERVER, 2005). A mostra todas as descargas atmosféricas (em formato de cruz) que estão espacialmente representadas como tipo ponto no intervalo das 05:00 às 06:00 h (GMT) do dia 07 de fevereiro de 2006. Nesta imagem percebe-se uma grande concentração de descargas atmosféricas na região de Guararema e outras menores no mar.

Deseja-se saber o comportamento deste evento num segundo instante de tempo, por exemplo, uma hora depois da observação deste evento:

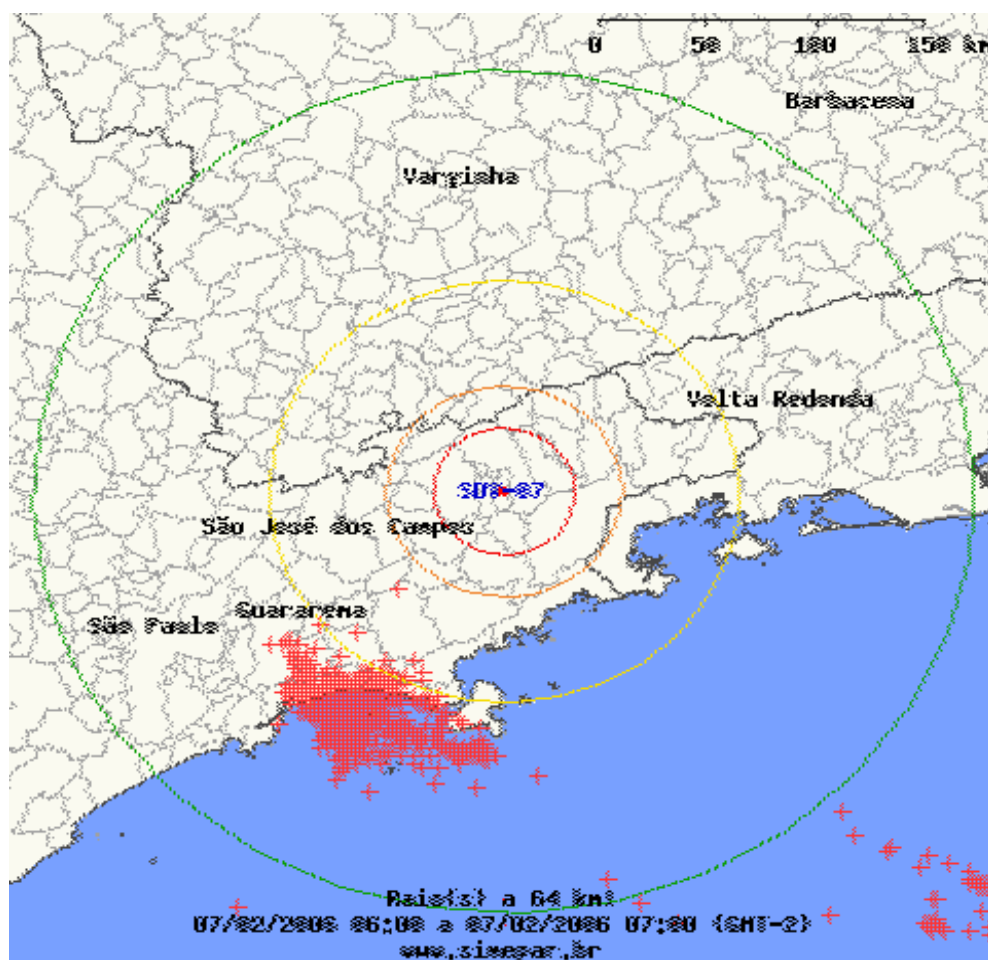


FIGURA 4.2: SEGUNDO ACOMPANHAMENTO DE DESCARGAS ATMOSFÉRICAS
Fonte: (SIMEPAR, 2006).

A Figura 4.2 ilustra a quantidade de descargas atmosféricas na hora imediatamente posterior da Figura 4.1 (06:00 h até às 07:00). Nessa Figura, pode-se perceber o comportamento de uma tempestade deslocando-se para o mar. Observa-se ainda a forma irregular destes núcleos de descargas atmosféricas (aglomerado maior).

Para se obter uma visão mais ampla deste evento, foi observada também a incidência de descargas atmosféricas na hora imediatamente seguinte da Figura 4.2:

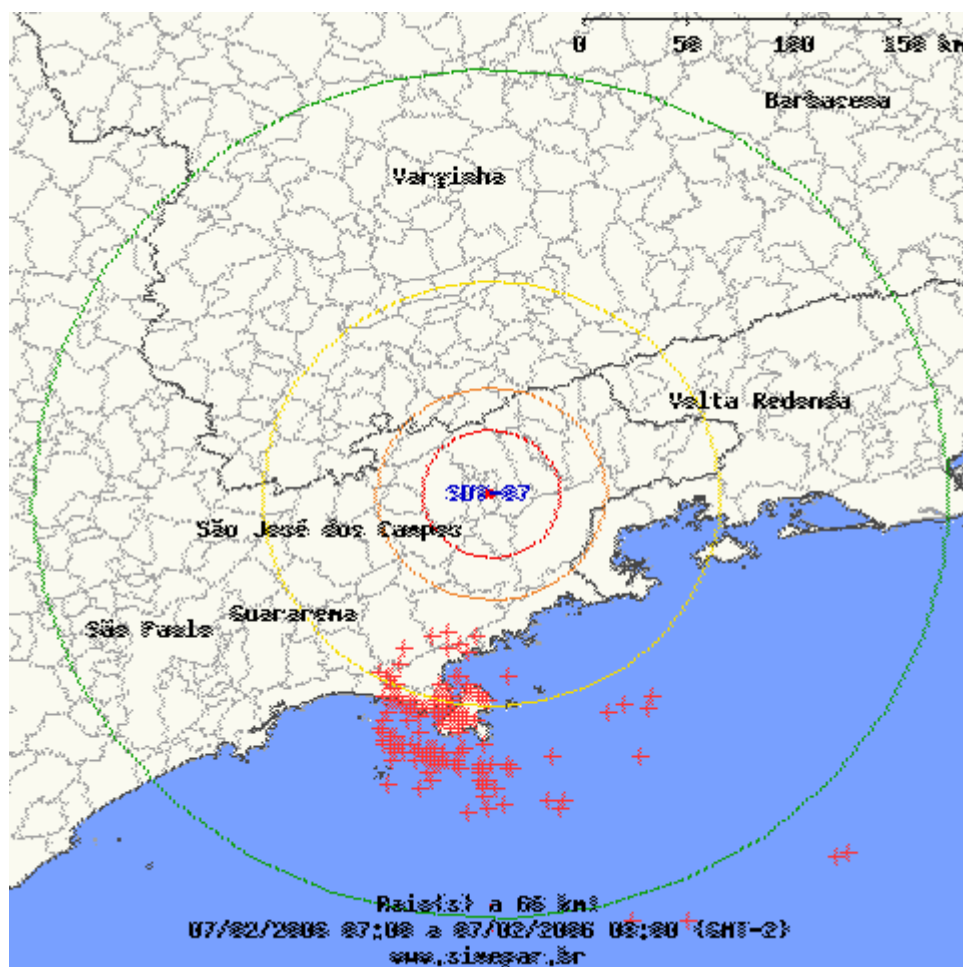


FIGURA 4.3: TERCEIRO ACOMPANHAMENTO DE DESCARGAS ATMOSFÉRICAS
Fonte: (SIMEPAR, 2006).

A Figura 4.3 (SIMEPAR, 2006), das 07:00 às 08:00h GMT -ilustra como estes núcleos podem se desfazer após um período de tempo. As descargas atmosféricas nos três tempos são distintas, porém mostra um comportamento das atividades de uma tempestade elétrica.

Através da aglomeração de dados via banco de dados, outras análises e interpretações podem ser feitas. Por exemplo, se estas descargas em três tempos distintos tratam da mesma tempestade? Qual a diferença do trânsito destes núcleos? Quais os setores (30, 50, 100 ou 200 km) mais afetados e quais as topologias e regiões afetadas? Dado um ponto qualquer, a qual aglomerado ele pertence? Qual o ponto central do núcleo destes aglomerados? Qual a distância com outros clusters?

Após o tratamento do método aqui apresentado, os pontos poderão ser unidos e transformados em polígonos de funções da própria extensão, por exemplo, as funções *convex hull* ou *buffer* (vide Seção 3.3). O tratamento destes pontos como polígono pode gerar outros tipos de indagações, por exemplo: Qual localidade ao cluster está se sobrepondo? Quais as localidades que fizeram parte deste trajeto? Qual o tamanho? Diminuiu ou aumentou? A tempestade se dissipou e ou se intensificou?

Para resolver todas estas questões, no nível computacional, foi realizado um estudo com algoritmos de *clustering* adaptados para Banco de Dados. Para isso foi escolhida a extensão do Banco de Dados *Open Source PostgreSQL*, o *PostGIS*, visto que: i) é único de código aberto que pode ter a sua estrutura alterada; ii) o estudo de viabilidade da extensão para *PostgreSQL/PostGis* é uma forma de contribuição ao crescente movimento Software Livre; iii) o *PostgreSQL* tem crescido nas comunidades científicas, devido aos bons resultados de desempenho e estabilidade (RAMSEY,2004); iv) possui uma lista de discussão ativa na internet; v) o *PostGIS* possui funções espaciais que se adaptam aos propósitos do método; e vi) o *PostGIS* possui conectividade com softwares de visualização de resultados, como por exemplo, *QuantumGis* e *MapServer*.

Após a análise deste método poderá ser gerado arquivo do tipo *shape* no qual poderão ser realizadas comparações com imagens de satélite ou imagens obtidas por radar com existência de tempestades.

Outra aplicação para este método pode ser visto como aglomerado de casos de doenças dentro de uma região, aglomerados de focos de calor dentro de um Estado, aglomerados de municípios, entre outros exemplos. A exigência para a busca de cluster é a passagem de uma chave primária, ou atributos que identifiquem os pontos, e o ponto em formato espacial.

Uma breve abordagem sobre o tema alvo (descargas atmosféricas) será feita, bem como as idéias principais na tentativa de responder a viabilidade, bem como métodos, hipóteses e paradigmas.

4.1. Descargas Atmosféricas

É importante entender o mecanismo básico de funcionamento do alvo de estudo: a descarga atmosférica. A descarga atmosférica, ou como é mais conhecida- raio- é um fenômeno da natureza que causa muitos danos. “Atualmente, cerca de 700 milhões de

raios por ano lampejam pelo globo terrestre. Provocam em média duas mil mortes e pelo menos 30% da falta de energia elétrica, entre outros inúmeros prejuízos”. (PINTO Jr *et al*, 2000).

A aglomeração dos dados, em princípio, tem foco na detecção do padrão de descargas atmosféricas que atingiram determinados alvos, por exemplo, uma região. A determinação de padrões de ocorrência de descargas atmosféricas tem aplicação imediata na identificação de áreas alvo para a companhia de energia elétrica, para os quais aperfeiçoamentos na proteção contra descargas atmosféricas trariam melhorias no desempenho das linhas de transmissão e distribuição. (BENETI, 2002):

“É internacionalmente reconhecido que as descargas elétricas atmosféricas é a principal causa dos desligamentos não programados nos sistemas de transmissão e distribuição de energia elétrica, representando entre 40% a 60% desses desligamentos, e com valores ainda maiores quando há alta incidência de descargas em terrenos de alta resistividade. Além disso, o curto-circuito causado pelas descargas elétricas, ainda que eliminados pelas proteções, causam quedas de tensão que são um dos tipos de defeitos na qualidade de energia fornecida que mais causam problemas aos consumidores industriais com processos controlados eletronicamente. Então, uma melhor compreensão do fenômeno é fundamental para reduzir o impacto das descargas atmosféricas nos sistemas de transmissão e distribuição”.

Assim, como objetivo inicial de pesquisa foi definido a identificação da distribuição espacial, como o formato que o agrupamento das descargas elétricas possui (mais alongado, mais redondo), e finalmente, a relação temporal das descargas atmosféricas.

Estas descargas elétricas estão associadas a células convectivas existentes nas nuvens *Cumulonimbus*, que podem estar isoladas ou constituir um aglomerado. O processo de formação dessas nuvens depende da instabilidade atmosférica e das condições termodinâmicas e dinâmicas predominantes. Dentro das nuvens, cargas intensas desenvolvem-se a partir da colisão de diferentes tipos de partículas como cristais de gelo e granizo, atingindo às vezes carga elétrica. (POLITI, 2004).

As descargas atmosféricas se classificam quanto a forma de ocorrência e configuração, em nuvem-solo, solo-nuvem, nuvem-nuvem, intranuvem, horizontais e para a estratosfera (BENETI *et al*, 2000). Nesta pesquisa serão utilizados dados do Brasil, o qual se registra descargas nuvem-solo, devidas o tipo de equipamento de detecção predominante.

A caracterização dos sistemas convectivos por meio de dados de descargas é uma informação auxiliar sobre a região afetada pela parte mais severa da tempestade devido à sua estrutura eletrodinâmica (POLITI, 2004). Esta informação auxiliar armazenada em forma de *clustering* irá facilitar pesquisas como a relação dos mesmos com os tipos de tempestades.

A formação de núcleo de nuvens, geralmente, está relacionada com tempestades. Onde há tempestade, há descargas elétricas. Isto se deve às perturbações, no campo eletromagnético, produzidas por este tipo de fenômeno.

A detecção da densidade de incidência de raios em forma de *clustering* poderá auxiliar na comparação deste agrupamento de raios com imagens de satélites, que mostram de forma ampla, isto é, como um todo numa tempestade.

A robustez de um banco de dados espacial para este tipo de situação é indicada. Com a integração de todos os dados necessários, a previsão de descarga elétrica mais eficaz, é possível fazer uma relação de todas estas informações. Além disso, através de métodos e da geometria de formação de raios, é muito mais simples e perceptível analisar dados, pois estes poderão não apenas ser consultados, mas também serem visualizados espacialmente em aplicativos específicos.

Na forma convencional de armazenamento de dados, a busca por certas características, tais como a distância entre incidência de descargas elétricas e linhas de distribuição de energia, município onde ocorreu a descarga e o tipo de forma dos agrupamentos dos mesmos é dificultado. No *ArcView/ArcGis* (ESRI, 1996), software de geoprocessamento, o cálculo de distância entre pontos, e a proximidade das descargas com linhas de transmissão é possível através de funções matemáticas inseridas por este software. Além do trabalho de exportar estes dados para este software, como por exemplo, conversão de dados do tipo *shape* para tabelas de informações. Além disso, não é possível a gerência da integridade, da não duplicidade dos dados, visto que o *ArcGIS* é para manipulação de *shapes* e um gerenciador de dados.

O agrupamento de descargas elétricas em formato numérico não é transparente para o usuário. Este agrupamento é perceptível quando estes dados são transformados em imagens. Se os dados estiverem dentro do banco de dados geo-espaciais, estes cálculos serão possíveis, sem precisar exportar ou visualizar em outros softwares. A função de distância entre pontos, a função que verifica se um ponto pertence a um polígono (como é o caso de municípios), são funções implementadas pelo *PostGis* e o *Oracle Spatial*.

Como foi descrito anteriormente, nesses SGBDs não foi encontrado uma função que implementasse detecção de *clustering*. Como este tipo de função se mostra necessária na aplicação de detecção da densidade de descargas elétricas. Esta dissertação analisa e sugere a criação de uma função de detecção de cluster através de um método desenvolvido para este fim.

4.2. Banco de Dados Espaciais no Apoio de Consultas sobre Raios

A previsão do tempo e os laudos são artefatos essencialmente baseados em informação e conhecimento, portanto, suas tecnologias, métodos e ferramentas avançam rapidamente, pois falar sobre tempo é tratar de ambiente dinâmico e flexível. Esta essência e necessidade de correção de informações, em um sistema gerenciador de banco de dados que possua ferramentas que cooperam nesta precisão aumentam o grau potencial de resultados precisos.

Uma aplicação diferenciada de um banco de dados convencional é uma consulta visando determinar as localizações exatas de certos objetos. Com os dados na forma convencional, dados aproximados são encontrados. Um exemplo disso é a consulta por raios dentro de cada município. Em um sistema convencional é necessário inserir na consulta a latitude, longitude de abrangência de cada município que deseja consultar. Com dados do tipo espacial, esta precisão pode ser conseguida, pois considera valores geométricos, e não apenas valores alfa-numéricos.

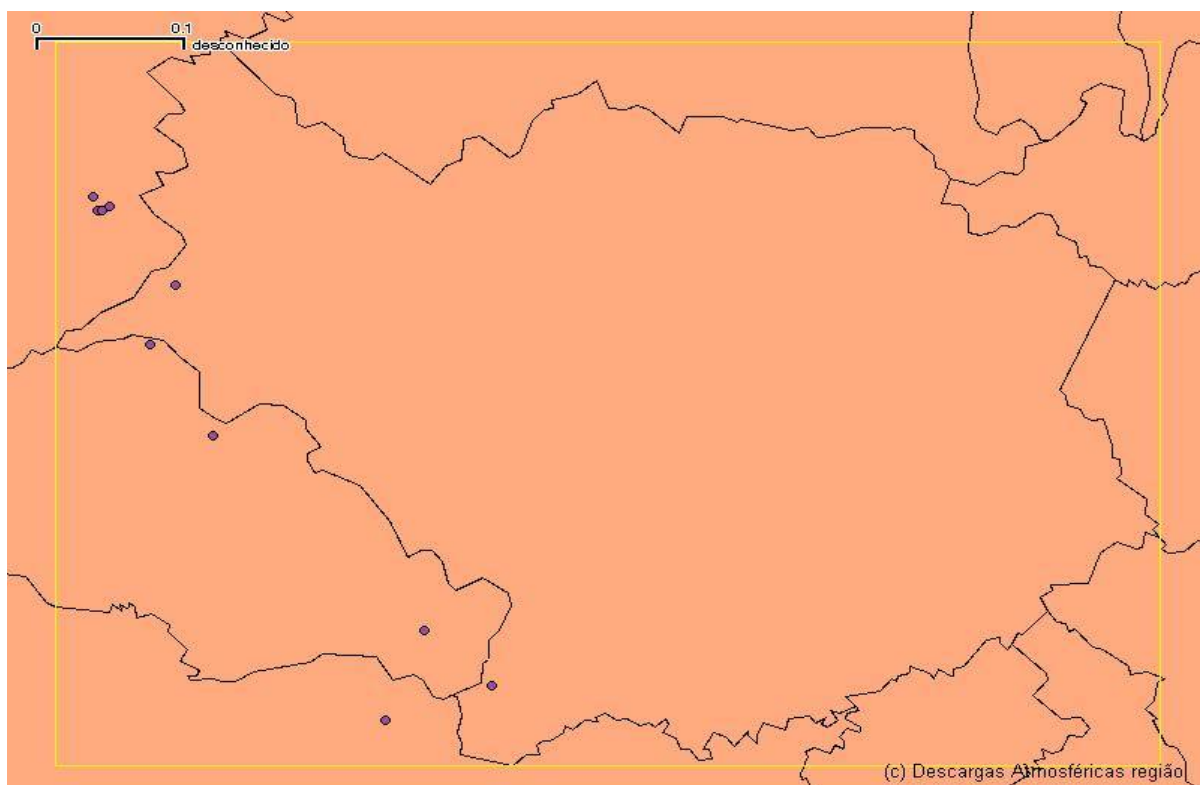


FIGURA 4.4: CONSULTA POR RAIOS DENTRO DE UM MUNICÍPIO

A Figura 4.4 ilustra a geometria do município de Castro-PR, bem como a distribuição espacial de algumas descargas atmosféricas. Pela consulta SQL tradicionais têm:

```
Select count(*)
From raio
Where RAIDATAHORA >= '2005-06-15 14:00:00'
And RAIDATAHORA <= '2005-06-15 15:59:59'
And raipicocorrente>= -10
And rainumsensor>=4
And RAILONGITUDE >= -50.2462692260742
And RAILONGITUDE <= -49.4987144470215
And RAILATITUDE >= -25.0723686218262
And RAILATITUDE <= -24.5410842895508
Order by 1;
```

Essa consulta por descargas atmosféricas retorna todas as descargas atmosféricas, informando um período de tempo nas dimensões de latitude e longitude que o usuário indica (área do retângulo em amarelo). Neste exemplo o resultado da contagem de raios é 10.

A mesma consulta utilizando a extensão espacial *PostGIS* é a que se segue:

```

Select count(*)
From raio, municipio
Where RAIDATAHORA >= '2005-06-15 14:00:00'
And RAIDATAHORA <= '2005-06-16 15:59:59'
And raipicocorrente >= -10
And rainumsensor >= 4
And the_centrogeom && the_geom
And munnome= upper('Castro')
And contains(the_geom,the_centrogeom)
Order by 1;

```

Na segunda consulta é necessário indicar o nome do município presente na tabela município, que contém a geometria que limita os municípios (coluna the_geom). Nesta consulta pode-se observar o uso do operador && - de sobreposição de geometria e a função *contains*. O teste verificado aqui é: verifique se o quadrado envolvente de uma geometria está sobreposto sobre outra geometria. Se retornar *true* (verdade) verifique se o raio (the_centrogeom) está totalmente contido dentro da geometria de municípios. O resultado desta consulta SQL é mais precisa que a consulta anterior e corresponde a 2. Neste simples exemplo verificamos que 8 pontos foram contados a mais, mas num dia de tempestade, com a presença de muitas descargas atmosféricas, aqui representadas como ponto, o erro pode ser maior.

Com o armazenamento deste tipo de dados, podemos consultar dentro de um banco de dados e encontrar onde houve menos acontecimentos de raios. Esta consulta se justifica, por exemplo, para a construção de torres de transmissão.

Outro motivo de armazenar estes dados em um Banco de Dados Espacial é a utilização de ferramentas que aperfeiçoam certos processos, tal como interpolação de valores e outros processos complexos que exigem conhecimento de meteorologia. A maioria dessas atividades não é feita só por uma pessoa, requer a distribuição de conhecimento e a atuação de grupos. O uso desta tecnologia de informação auxilia no processo de consulta e atualização contínua dos dados.

A escolha de uma modelagem que melhor represente estes objetos espaciais dentro do banco de dados é uma das maiores preocupações, tal como observa Câmara (CÂMARA *et al*,2000):” ... o processo de modelagem é precedido de uma fase de análise exploratória, associada à representação visual dos dados sob forma de gráficos e mapas e a identificação de padrões de dependência espacial no fenômeno de estudo”.

Na modelagem, os primeiros dados que podem ser inseridos são a latitude e a longitude da ocorrência de uma descarga atmosférica. Esses dados ficam armazenados

no banco de dados expressos geometricamente em formato ponto (*point*). Da forma convencional, com apenas o valor de localização da descarga, uma consulta seria escrita considerando latitude e longitude maiores, menores ou iguais a determinado valor. De forma espacial, esta latitude deixa de ser apenas um valor, e se torna um objeto geométrico, podendo sofrer consultas topológicas, tais como distância, sobreposição a outra geometria, entre outros.

É interessante comparar estes dados com matriz de dados de radar transformados em imagens. A partir desta comparação é possível determinar os núcleos de chuva, isto é, determinadas formações de nuvens com certas temperaturas, gotas de chuva propícias a chuviscos ou as tempestades que geralmente estão relacionadas aos raios. Estes dados numéricos são imperceptíveis para usuários leigos no assunto. Somente com transformação desta matriz em imagens e com auxílio de orientações técnicas de meteorologistas é possível entender estes focos de nuvens e o que elas representam.

Um dos pontos significativos relativos as descargas é que não há padrão de onde e quando as descargas acontecem. É impossível colocar campos específicos para cada hora certa onde há incidência de uma descarga atmosférica - estes são os fatores que interferem neste processo.

Um outro ponto é a cultura na utilização deste novo tipo de SGBD, ou seja, são poucas as pessoas que estão acostumadas com este tipo de dados dentro de um SGBD. É um tipo de SGBD ainda pouco utilizado, devido à especificidade dos dados que podem ser utilizados.

São diversas as áreas que necessitam saber as condições do tempo, mas não são todos que possuem os dados para manipulá-los. Como exemplo, o cálculo da aproximação de nuvens sobre determinada região. Com estes dados em um gerenciador de banco de dados convencional, o cálculo seria feito através de uma área aproximada que envolveria toda uma cidade. Como foi exemplificada, esta área geralmente se trata de retângulos, não levando em consideração que uma cidade não possui uma geometria com apenas retas, mas sim muitas curvas que não possuem funções específicas para tal cálculo.

Com a indexação destes agrupamentos de descargas atmosféricas (clusters) e o armazenamento em uma base de dados, é possível fazer pesquisas, buscas e extrair diversas informações que hoje são feitas manualmente. Com a geometria dos municípios e a posição exata de diversos dados ambientais, além de pesquisas na área meteorológica, estatística e mineração de dados serão feitos com mais precisão. Isto se

deve porque os dados são testados para cada ponto e não para um aglomerado de pontos. Além deste argumento, o ponto antes era testado para a latitude e longitude aproximada de um município. Com o armazenamento destes municípios respeitando cada curva da geometria que forma o município, este teste, além de mais rápido, pode ser feito até mesmo na fronteira entre cidades.

4.3. Metodologia Utilizada

Diversos testes para a realização do método de representação de *clustering* para banco de dados espaciais foram realizados na tentativa de trazer aglomerações de dados do tipo ponto. Estes testes tiveram como objetivo trazer as aglomerações de maneira que a resposta corresponda com a realidade específica da aplicação aqui abordada (descargas atmosféricas).

Em relação ao hardware utilizado, os testes foram realizados utilizando uma máquina servidora Intel(R) Xeon(TM) CPU 2.00GHz, com 4Gb de memória RAM, com o Sistema Operacional *Linux*.

Nesta máquina foi necessário utilizar um SGBD que desse suporte a uma extensão para tipos de dados espaciais. Devidos aos aspectos já mencionados como os bons resultados relatados pela comunidade acadêmica, além de ser *open source*, foi escolhido o SGBD *PostgreSQL* 8.1.3 com a extensão *PostGis* 1.1.1 (últimas versões lançadas).

Também, foram utilizadas as bibliotecas Geos 2.1.1 e Proj4 4.4.9, que são requisitos para a instalação do *PostGIS* (RAMSEY, 2005). A biblioteca GEOS é usada para fornecer testes em geometria (*Touches()*, *Contains()*, *Intersects()*) e operações (*Buffer()*, *GeomUnion()*, *Difference()*) com *PostGIS*. GEOS (GEOS, 2006) A biblioteca Proj4 é usada para prover suporte a reprojeções de coordenadas dentro do *PostGIS*. (PROJ4, 2006).

Este trabalho utilizou, em um dos métodos, as bibliotecas do software Weka (WEKA, 2005; WITTEN, 2000) que inclui algoritmos de *clustering*, *weka.clusterers*. Estes operam de modo parecido com métodos de classificação em *weka.classifiers*. As opções de linha de comando são divididas em opções de esquemas genéricos e específicos tais como entrar com um arquivo desejado no qual se encontra treinado, um arquivo de teste, arquivo de saída do modelo, entre outros.

Além destas bibliotecas, classes do software Weka *clusterers* foram utilizadas. Ferramentas para visualização dos dados inseridos tal como o *QuantumGis*

(QGIS,2005) e MapServer (MAPSERVER,2005) foram utilizadas, visto que possuem suporte *PostGis*.

Os dados de descargas elétricas foram cedidos pelo RINDAT (Rede Integrada Nacional de Descargas Atmosféricas) no qual o Paraná faz parte através do Simepar (SIMEPAR, 2006) e incluem 25 parâmetros para cada ocorrência de descargas nuvem-solo, incluindo tempo, localização (latitude/longitude), pico de corrente, polaridade e multiplicidade. As características mais relevantes nesta pesquisa foram a interação espaço-tempo, portanto, data, hora, nanosegundo, posição das coordenadas latitude e longitude.

Os parâmetros utilizados estão em coordenadas geográficas bidimensionais, isto é, num plano x, y. As coordenadas também poderão ser facilmente transformadas para coordenadas *UTM* (*Universal Transverse Mercator*). Coordenadas UTM são métricas referentes a cada uma das 60 zonas UTM, isto é, “projeção cartográfica cilíndrica do esferóide terrestre em 60 cilindros secantes à superfície da Terra ao longo de meridianos em zonas múltiplas de 6 graus de longitude e estendendo-se de 80 graus de latitude Sul a 84 graus de latitude norte”. (UNB, 2006).

Como metodologia para a validação do método foram utilizados dados extraídos de uma base de dados espacial histórica dentro da região de abrangência dos sensores da RINDAT. Entre estes dados históricos foi escolhido um evento bastante estudado na área meteorológica, como foi o caso de uma tempestade que passou pelo Estado do Paraná, neste caso o dia 06 de Julho de 2003 das 14:00 h às 20:00 h (horário GMT). Salienta-se também que outros períodos de tempo também foram utilizados visando testes mais abrangentes.

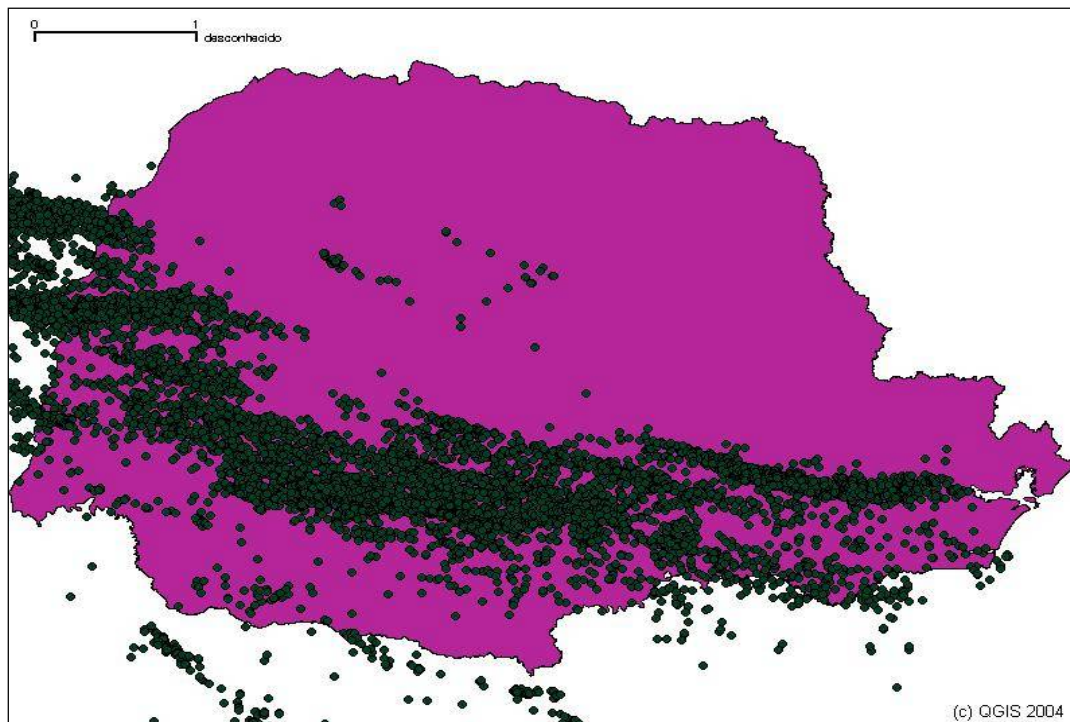


FIGURA 4.5: ESTADO DO PARANÁ E A INCIDÊNCIA DE DESCARGAS ATMOSFÉRICAS EM 06-07-2003 14:00 HORAS ÀS 20:00 HORAS (HORÁRIO GMT -2)

A Figura 4.5 acima ilustra a incidência do aspecto meteorológico na data de 06 de Julho de 2003. Em cerca de 6 horas, foram registradas pelos sensores 15.573 descargas atmosféricas nuvens-solo. Diferentes pontos no tempo, no período mencionando, foram utilizados como teste.

Também, foram utilizados os dados do Instituto Brasileiro de Geografia e Estatística (IBGE), tais como cidades, geometria, população, para ilustrar as regiões mais atingidas por descargas elétricas e também, para servir como ponto de referência ao leitor. Um exemplo destes dados do IBGE “especializados” dentro do Banco de Dados foi visto na Figura 4.5 (contorno do Estado do Paraná).

Como programação de apoio utilizou-se a Linguagem Java, devido suas particularidades de implementação com *drivers* de suporte *PostGis*, além de trabalhar de forma multi-plataforma.

Os métodos testados aqui mostram um estudo preliminar da viabilidade da criação de uma função para este tipo de problema. Uma nova função leva tempo para ser consolidada, pois muitos testes devem ser realizados na prática. Um estudo mais

aprofundado pode mostrar o quão complicado é a criação de funcionalidades dentro de um código feito por várias pessoas.

4.4. Método de Representação Espacial de Clustering

O método de *clustering* de dados espaciais se originou da observação do problema de agrupar dados espacialmente próximos, detectando automaticamente os *n* agrupamentos possíveis dentro de um conjunto de dados.

Na literatura foram encontradas ferramentas que aplicam algoritmos provindos da Inteligência Artificial ou da Computação Gráfica cuja finalidade é classificar e visualizar os dados com similaridades de distância.

Devido às particularidades dos objetos espaciais, estas técnicas de *clustering* não solucionaram um problema típico da meteorologia, que é detectar aglomerados de descargas elétricas no decorrer de um período de tempo. Sabido destas aglomerações, era necessário saber o tamanho, núcleo, os pontos e a quantidade de pontos associados a estas aglomerações. Com estas informações outras pesquisas poderiam ser efetuadas.

A partir deste problema foi desenvolvido um método que resolvesse estes questionamentos, podendo ser estendido a outros propósitos fora da área da meteorologia. A análise estatística do comportamento das descargas elétricas exige uma forma de representação conveniente das mesmas, o que é agravada pela alta frequência da ocorrência de descargas elétricas em uma tempestade, que gera um volume de dados elevado (POLITI, 2004). Esta representação será facilitada com a inserção espacial de alguns aspectos importantes no estudo deste fenômeno, tal como a direção da incidência de descargas elétricas, e a detecção da geometria formada pelos raios.

Esta análise mais aprimorada permitida com a manipulação geométrica viabiliza uma visão mais ampla dos fenômenos ocorridos dentro de uma região, além de um ajuste operacional do alerta. O método tem como parâmetros de entrada os pontos dentro de um espaço temporal (data e hora da descarga) e uma distância máxima que delimita a vizinha para cada ponto. Como resultado deste método será a obtenção dos *n* pontos que representam *clusters*.

O método de detecção de *clustering* auxiliará na obtenção de áreas de incidência de muitos raios em curto período de tempo, podendo assim, comparar estes dados com tempestades, correlacionando informações enviadas através de um radar meteorológico.

Com a representação de cluster dentro de um SGBDE é possível achar núcleos de descargas elétricas através do cálculo do centróide de cada cluster. Depois de detectados todos os possíveis clusters de um conjunto de dados espaciais do tipo ponto, também se podem alinhar diferentes visões do mesmo núcleo de raios, isto é, através do centro (centróide) destes núcleos é possível saber o caminho por onde uma tempestade está se deslocando. Sabendo este deslocamento é possível obter uma imagem final com o percurso da mesma. Esta forma de tratar as descargas atmosféricas, levando em consideração as características que uma tempestade possui, os meteorologistas vão obter mais uma forma de extração de conhecimento. Vale ressaltar novamente que esta aplicação do método é apenas um exemplo da contribuição feita pelo uso do método apresentado neste trabalho.

Para poder alinhar duas geometrias, deve haver dois casos imediatamente parecidos (com distância temporal próxima) e também geometrias com áreas suficientes de sobreposição, para que estas não se tornem distorcidas, podendo comparar assim, núcleos de nuvens semelhantes e próximos entre si, o que pode levar ao erro.

Outra particularidade que objetivou a descoberta de núcleos através do método de *clustering*, é que pode se automatizar a extração de valores de metadados, isto é dados que descrevem dados, visto que “os metadados visam manter a organização interna dos dados geo-espaciais” (PINHEIRO *et al*, 2005). Um tipo de metadados interessante para descargas elétricas atmosféricas é núcleo. Este núcleo de nuvens se refere aos tipos de formação que foram criados pela incidência de raios, tais como redondos, chatos, largos, etc. É possível procurar dentro de uma base de dados inteira e depois converter em imagens, que podem ilustrar tipos de formação de tempestades.

Após a validação do método, o mesmo poderá gerar uma função. Essa função poderá ser adicionada dentro da extensão espacial. No futuro, quando isso for feito, será necessário criar um pacote (*.jar*), configurar o ***classpath*** no diretório do *PostgreSQL*. Dentro do *PostgreSQL*, criar a função configurando para apontar para o ***classpath***, podendo assim chamar a função dentro da linguagem ***psql*** do Postgresql.

A seção a seguir visa mostrar dois dos caminhos que foram percorridos na busca do método de representação de *clustering* mais adequado, e finalmente o método criado que resolve o problema de agrupamento de descargas atmosféricas e os testes que o validam.

4.4.1. Experimentos para a Criação do Método

Dentro do domínio de aplicação, foram elaborados testes para mono-usuários para identificar *clusters* para dados espaciais pontuais. Um destes testes teve como base a aplicação de algoritmos de clustering já estudados na literatura. Aqui será abordada a aplicação do K-Médias, através do software Weka, desenvolvido para a área de Inteligência Artificial. Outra tentativa foi realizada utilizando as funções de uma extensão espacial (*PostGis*).

A seguir serão apresentados os experimentos e os resultados obtidos através deles: (a) Aplicação de algoritmos bastante utilizados pela Inteligência Artificial; (b) Aplicação de funcionalidades do *OGIS* para geração de polígono.

4.4.1.1. Experimento 1- Aplicação de Algoritmos da Inteligência Artificial através do Weka

A aplicação das classes do Weka foi realizada para validação dos argumentos (método) apresentados neste trabalho, no sentido de comparação. O Weka (WEKA, 2005) é muito utilizado na área da Inteligência Artificial, e possui funcionalidades de classificação e *clustering*.

Para o experimento, foi utilizado o algoritmo K-Médias, com valor de k igual a 3. O parâmetro K deve ser obrigatoriamente passado como parâmetro pelo usuário. Este K se refere a quantia de partições que o algoritmo deve retornar. O valor semente, isto é, valor inicial que gera valores aleatórios para o teste, não foi especificado, deixando o algoritmo escolher o melhor (estatisticamente) possível. Cem descargas atmosféricas foram escolhidas aleatoriamente para este teste.

O experimento segue: (1º.) escolha de todos os pontos em uma determinada data, bem como a data, hora, nanosegundo (ns) da incidência da descarga atmosférica; e (2º.) crie um arquivo de leitura reconhecida pelo Weka, o ARFF (*Attribute-Relation File Format*), que possui o seguinte formato:

```

% 1. Título: Base de Dados de Teste para Clustering no PostGis
%
% 2. Recursos:
%   (a) Criador: Luciane C. Pinheiro
%   (b) luciane@simepar.br/ luciane@inf.ufpr.br
%   (c) Data: Janeiro de 2006
%
@RELATION raiotestepdistance
@RELATION Timestamps
@ATTRIBUTE timestamp          DATE "yyyy-MM-dd HH:mm:ss"
@ATTRIBUTE araidatahora       DATE
@ATTRIBUTE arainanosegundo    NUMERIC
@ATTRIBUTE arailatitude       NUMERIC
@ATTRIBUTE arailongitude     NUMERIC
@ATTRIBUTE araimultiplicidade NUMERIC
@ATTRIBUTE araipicocorrente   NUMERIC

@DATA

```

ARFF é um arquivo texto formato ASCII que descreve uma lista de instâncias, compartilhado um conjunto de atributos (WITTEN, 2002). O arquivo ARFF com os dados do teste pode ser verificado no Apêndice 2.

- 1) Passe o arquivo arff para o Weka através de linha de comando:

```

java -cp "weka.jar" weka.clusterers.SimpleKMeans -N
3 -p 0 -t data/raioscluster.arff > /tmp/teste.txt

```

O Algoritmo *SimpleKMeans* executa 10 vezes o algoritmo *K-Means* ou K-Médias e determina um caso considerado estatisticamente bom. O parâmetro *-N* indica o número de K clusters. *-P* indica predict, isto é, como resultado traz o número da linha do ponto em questão e o valor de cluster para esse ponto. O argumento “data/raioscluster.arff” indica o local e o arquivo no formato arff que deverá ser pesquisado. Finalmente, o resultado é redirecionado para um arquivo chamado teste.txt no diretório /tmp.

- 2) Através do arquivo de saída do Weka, atribua para cada ponto o valor de cluster obtido. Transforme latitude e longitude para o formato espacial ponto.

A Figura 4.6 demonstra o resultado deste experimento. A Figura 4.6 gerada através do software *QuantumGis* (QGIS, 2005) mostra como cluster 1 os pontos em verde, como cluster 2 os pontos em amarelo e como cluster 3 os pontos em roxo. Verificamos que há pontos que estão sob outro cluster, como visto nos clusters 2 e 3. O

cluster 3 se demonstra de forma muito espalhada, devido as estatísticas utilizadas aqui utilizaram os aspectos temporais e de pico de corrente. O aspecto temporal (data, hora e nanosegundo da descarga atmosférica) foi utilizado como chave-primária e mais um elemento teve que ser utilizado nesta técnica: o pico de corrente.

As aglomerações que resultaram estes clusters demonstram apenas grupos estatisticamente próximos ou visualmente próximos, porém na média revelaram características sem sentido relevante (neste caso pico de corrente que possui valores muitos discrepantes). De uma forma geral, para *clustering* utilizando métodos estatísticos presumem a independência amostral, por exemplo, aplicando um método estatístico de *regressão linear* em dados espaciais, os resultados obtidos serão falseados para o teste de ajuste do modelo. Isso também é válido para os intervalos de confiança dos coeficientes do modelo e das previsões (BAILEY e GATRELL, 1995; ANSELIN e BAO, 1997; NETER *et al.* 1989; NEVES *et al* 2001).

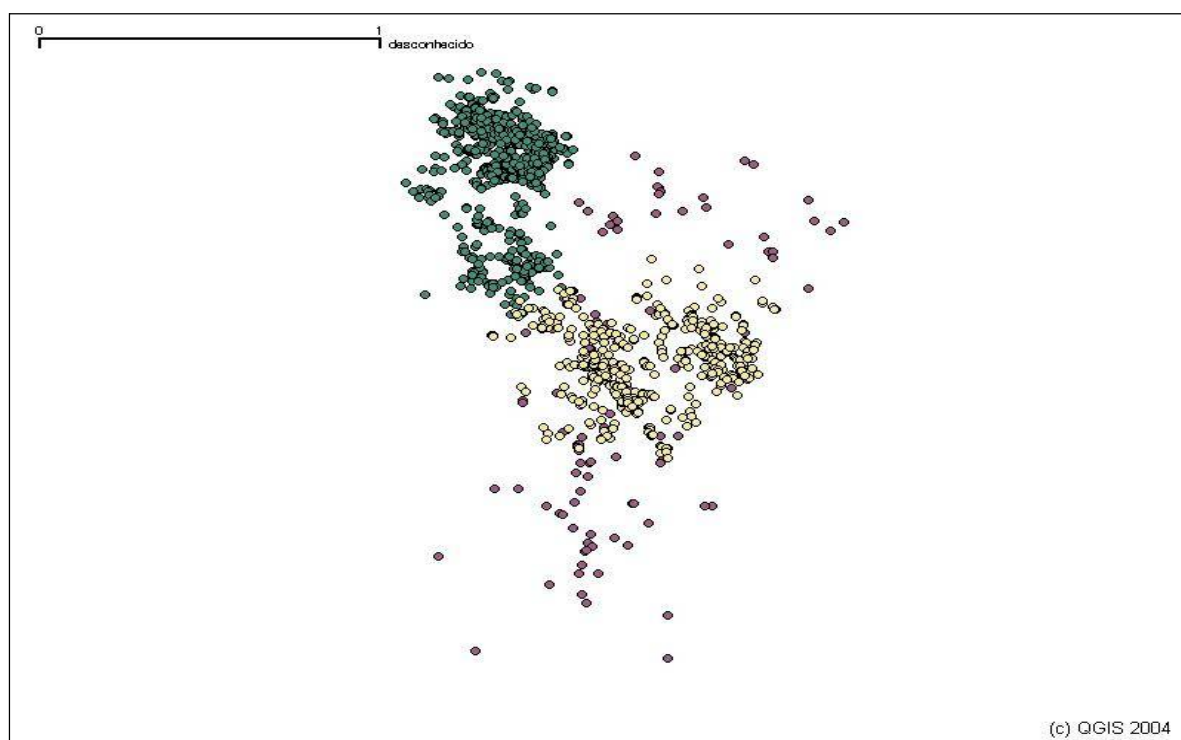


FIGURA 4.6:CLUSTERS GERADOS PELA UTILIZAÇÃO DO K-MEANS COM O SOFTWARE WEKA

O experimento não obteve resultado esperado para os dados espaciais, pois para descargas atmosféricas, os dados são aleatórios, isto é, não há um padrão definido da incidência dos mesmos.

Como ponto positivo observa-se que pela quantidade de pontos utilizados nos testes, este algoritmo apresentou um bom desempenho na consulta realizada, já que foi utilizado dado do tipo alfanumérico. E como ponto negativo pode-se concluir que atributos utilizados no teste, como pico de corrente, possuem variância muito alta, o que possui uma estatística que não reflete a veracidade dos dados. Outro ponto negativo da aplicação para este tipo de dados é a interação com o usuário, que informa o valor de clusters desejados ou o valor semente. Este tipo de interação fornece resultados diferentes para cada valor informado pelo usuário.

Este experimento permitiu avaliar se o uso de técnicas utilizado pela literatura, inicialmente com dados alfanuméricos, teria o mesmo efeito com dados espaciais. Devido os resultados obtidos, uma outra tentativa foi elaborada utilizando as funcionalidades individuais de uma extensão espacial definidas pelo OGIS. Este outro método é visto na Seção 4.4.1.2.

4.4.1.2. Experimento 2 – Funções OGIS

Foram realizadas diversas tentativas de representar ou determinar *cluster* através do uso de técnicas das funcionalidades já implementadas dentro do PostGis, seguindo o Padrão do OGC. Para todos os casos foi utilizada a função *distance_sphere*. Esta função de distância só pode ser calculada para dados do tipo ponto e possui uma precisão maior do que a distância ponto a ponto, pois se leva em consideração a curvatura terrestre.

É apresentada, a seguir, uma entre as abordagens de criação de *cluster*, com formato de polígono. O experimento envolve o que segue:

- i) Para cada par de pontos, ache a distância mínima, através da função *distance_sphere*.
- ii) Ache a matriz de distâncias e ordene da distância mínima até a distância máxima.
- iii) Ligue cada ponto com o seu vizinho mais próximo através da interpolação de pontos da função *makeline*. A ligação dos pontos mais próximos foi feita na tentativa de achar o parâmetro que separa estes pontos em grupos naturais.
- iv) Repita estes passos até que a distância mínima seja menor ou igual a média das distâncias mínimas.
- v) Dado um parâmetro de distância mínima, isto é, com interferência do usuário, defina um envoltório ao redor do conjunto destas linhas. Para isso foram utilizadas as

funções: *buffer* – que cria o envoltório de uma geometria segundo uma distância e *collect* – que transforma dados com relação espacial, em uma coleção de dados, neste caso uma coleção de linhas (*linestring* – *multilinestring*).

vi) Através da comparação dos pontos iniciais com os polígonos criados, separe os pontos em conjuntos.

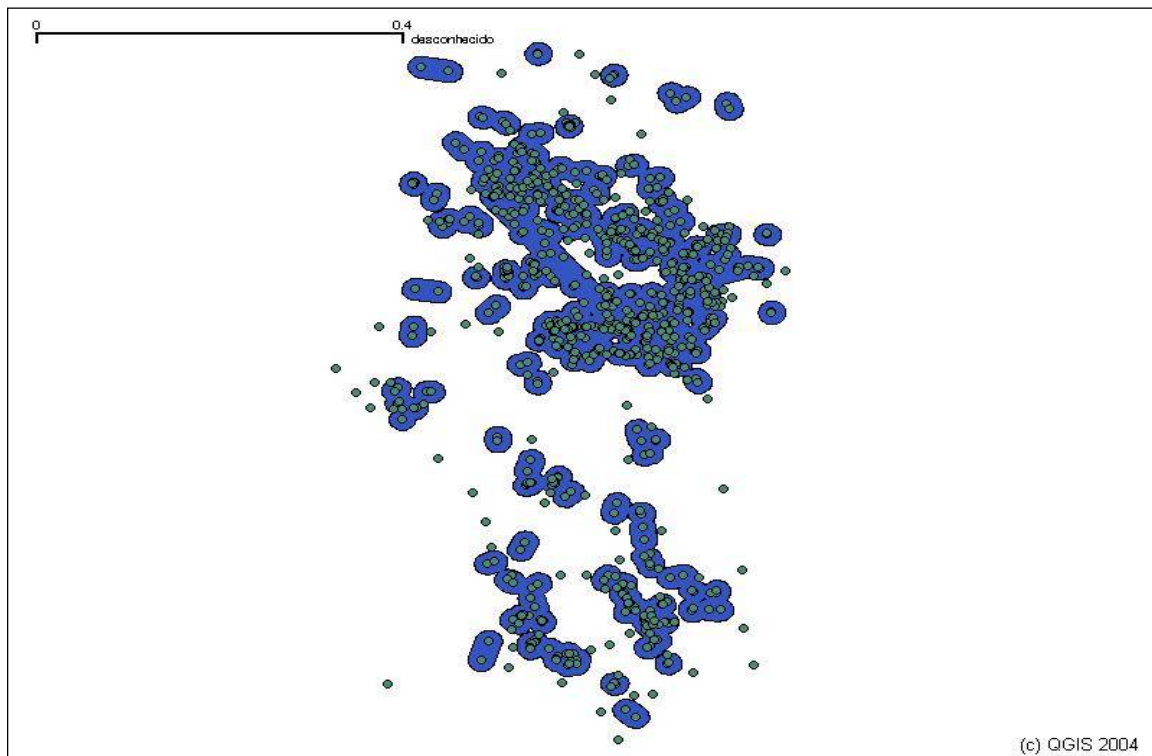


FIGURA 4.7: CLUSTERS FORMADOS PELO EXPERIMENTO 02

A criação de cluster para pontos em determinados casos obteve um bom resultado visual. Em outros casos a formação de cluster é errônea, visto que: a) precisa da interferência do usuário para determinar um valor ótimo que separa os pontos em grupos. b) um valor muito alto, passado pelo usuário, unindo com a concorrência de memória compartilhada do SGBD pode ter saídas com desempenho muito lento. c) critério no procedimento do Passo 2 e no Passo 3 a união é feita através da ordem especificada. A junção dos pares de pontos com distâncias mínimas pode estar distante dos outros pares com distâncias mínimas, logo a ligação dos pares pode se formar de modo incorreto. Sem a ordenação das distâncias mínimas até as máximas percebem-se mais clusters, visto que as ligações são realizadas entre os pontos, de forma isolada. Neste caso a ordenação foi feita pela ordem cronológica.

Nesta abordagem pode-se perceber a essência dos métodos do vizinho mais próximo e por grafos, com corte nas ligações acima da média.

Esta seção demonstrou, de forma exemplificativa, duas tentativas de resolver o problema de *clustering* de dados espaciais. Pode-se dizer que as técnicas de utilização da medida de proximidade entre os pontos, bem como o desvio padrão (estatísticas do Weka) entre as datas e horários das descargas, trouxe o ponto e o cluster no qual está associado de uma forma não representativa. No contexto de descargas atmosféricas os clusters não trouxeram significado que represente um núcleo de descargas.

Após a análise dos testes que não obtiveram resultados satisfatórios, outro experimento foi necessário. A determinação de *clusters* de forma semi-automática, isto é, com a intervenção mínima do usuário foi realizada. Na Seção seguinte é apresentado um método para a resolução do problema.

4.4.2. O Método Desenvolvido

Tomar como exemplo uma casa, e deseja-se saber quais são seus vizinhos. O critério que delimita o que é vizinho desta casa pode ser uma cerca, o bairro ou uma distância máxima desta casa com as outras casas. Neste último critério, todas as casas englobadas dentro de uma distância *n* entre elas são ditas vizinhas.

O método de representação espacial de *clustering* que será apresentado se baseia em descobrir o quão próximo deve estar um ponto de outro para que ele possa ser dito vizinho. Esta medida que limita a vizinhança entre os pontos deverá ser obtida com a ajuda do usuário. A partir dos pontos que possuem vizinhos em comum é feita uma iteração até que sejam encontradas aglomerações (clusters).

O algoritmo que representa o método em si é descrito em 7 (sete) passos descritos a seguir:

Passo 1: Selecionar os pontos desejados

Selecione todos os pontos dentro de um determinado período de tempo. Este passo pode ser substituído passando como parâmetro de entrada os pontos desejados em forma de uma tabela, visão. O uso de índice nesta tabela é sugerido para a otimização do processo. A Figura 4.8 exemplifica o Passo 1.

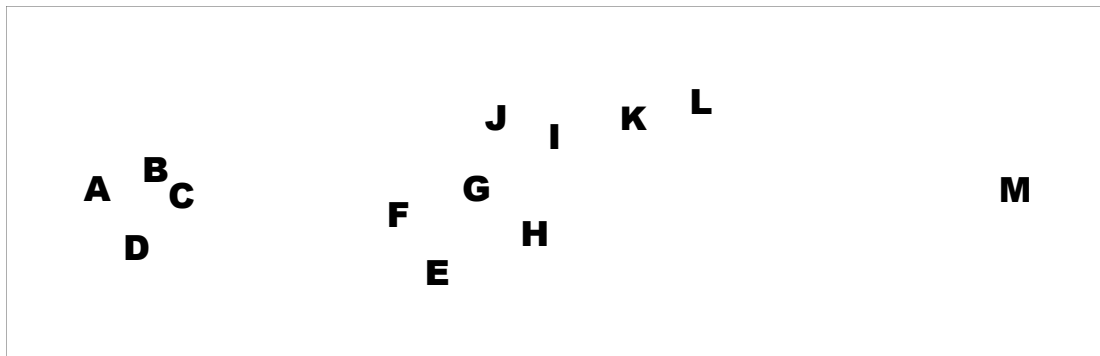


FIGURA 4.8: DADOS INICIAIS

Sejam os dados (A,B,C,D,E,F,G,H,I,J,K,L e M) selecionados pelo passo 1, pontos dispostos espacialmente em um período de tempo específico. Neste passo pode se observar a captura das características dos pontos, tal como data, hora e nanosegundo, bem como a geometria espacial do objeto (*point*)

Perceba que visualmente existem 3 grupos de dados distintos, mas para o banco de dados estes grupos não são perceptíveis, sem que haja um método para descobri-los.

Passo 2: Calcular todos os vizinhos.

Para calcular os vizinhos são necessários os Passos 2 e 3. Para cada ponto expanda sua área uniformemente para todas as direções, segundo um critério de distância n , com interação do usuário.

A Figura 4.9 mostra como são representados os pontos expandidos para descobrir a área de abrangência de cada ponto.

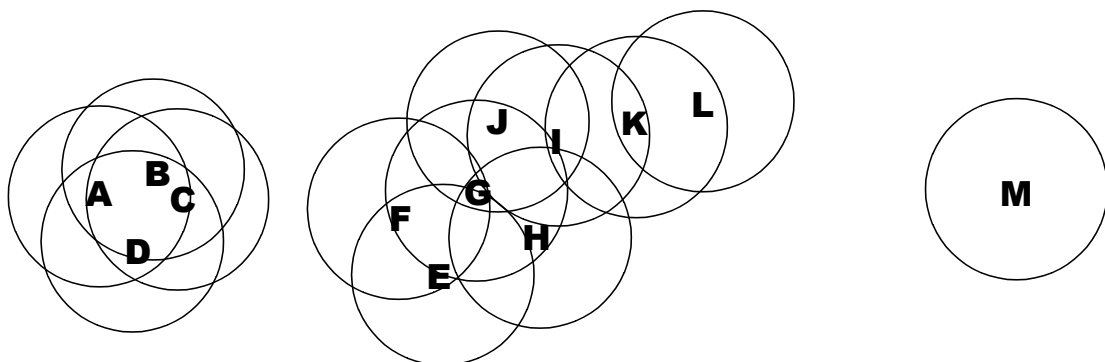


FIGURA 4.9: PONTOS EXPANDIDOS SEGUNDO DISTÂNCIA INFORMADA PELO USUÁRIO

Aqui se pode observar a utilização da função espacial *buffer*, que expande uma geometria segundo uma distância. O buffer pode ser visto como um polígono que

envolve uma determinada geometria. No caso do tipo ponto, a expansão (*buffer*) se apresenta na forma de um círculo.

Passo 3: Pontos contidos na vizinhança

Para cada ponto, calcule quantos pontos estão contidos dentro da distância n de vizinhança, isto é, para cada ponto encontre os seus vizinhos e a quantidade deles. Perceba que neste caso foi utilizado como teste a função *contains*, pois a função *distance* para cada ponto possui um custo de processamento maior. A Figura 4.10 abaixo usa como exemplo a detecção dos vizinhos do Ponto A.

Ponto A:

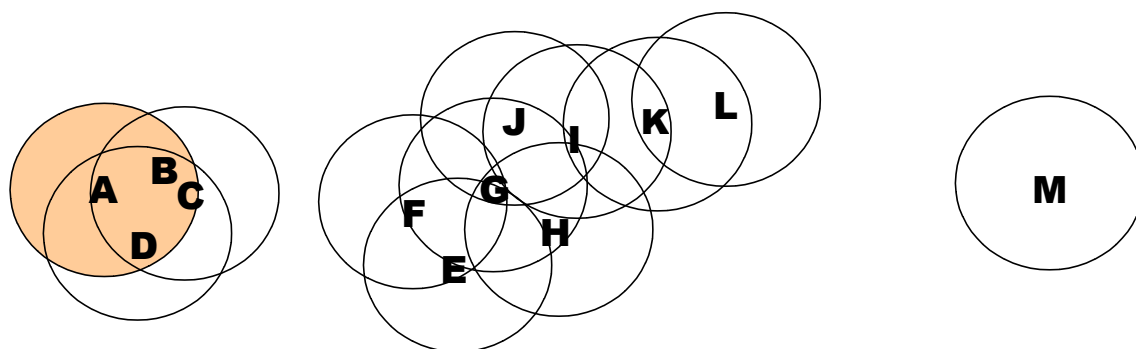


FIGURA 4.10: ÁREA DE VIZINHANÇA DO PONTO A

O Passo 3 observa a utilização da função espacial *contains*, que testa se um objeto espacial está contido dentro de outro objeto espacial.

Neste passo é testado se um ponto está contido dentro de um polígono. Pode-se dar como exemplo o buffer do ponto A, que está contém os pontos B, C, D. Com isso pode-se dizer que os vizinhos de A em uma dada distância n são três pontos: B, C, D.

Com a obtenção dos vizinhos para cada ponto, pode-se atribuir a cada ponto um valor de cluster. Inicialmente todos os pontos constituem seu próprio cluster, sendo assim atribuiremos a ele um valor em comum: 0. A iteração deste método para cada ponto do exemplo na Figura 4.9 pode ser vista na tabela a seguir:

Ponto	Pontos Vizinhos	Quantidade de Vizinhos	Cluster
A	BCD	3	0
B	ACD	3	0
C	ABD	3	0
D	ABC	3	0
E	FGH	3	0
F	GE	2	0
G	EFHIJ	5	0
H	GI	2	0
I	G,JK	3	0
J	IG	2	0
K	IL	2	0
L	K	1	0
M	-	0	0

Tabela 2: Vizinhos

Enquanto houver pontos vizinhos não visitados repita os passos a seguir:

Passo 4: Marcar o ponto com mais vizinhos

Para o ponto que obtiver a maior quantidade de vizinhos na área de abrangência marque-o como cluster n+1. Para os seus vizinhos marque-os também os pontos como cluster +1.

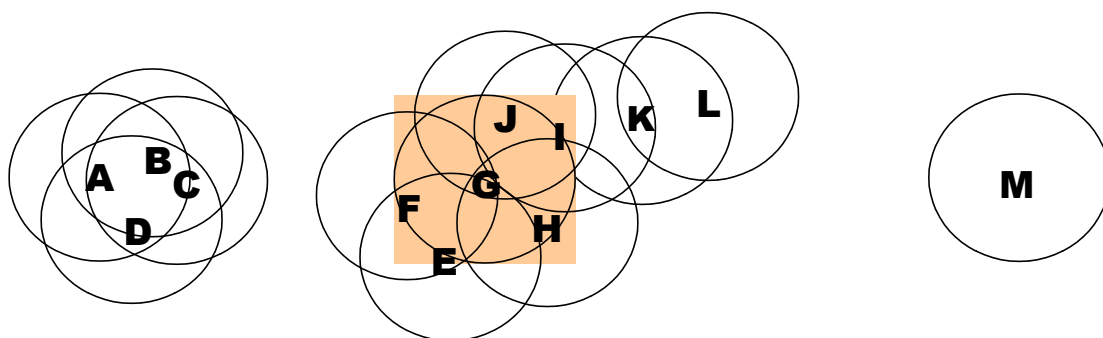


FIGURA 4.11 PONTO COM MAIS VIZINHOS

O Passo 4 descreve que para o ponto que possuir mais vizinhos (G, com 5 vizinhos), marque-o como *cluster* 1. Capture os pontos vizinhos de G (E, F, H, I e J.) e marque-os também como *cluster* 1.

Ponto	Pontos Vizinhos	Quantidade de Vizinhos	Cluster
A	BCD	3	0
B	ACD	3	0
C	ABD	3	0
D	ABC	3	0
E	FGH	3	1
F	GE	2	1
G	EFHIJ	5	1
H	GI	2	1
I	G,JK	3	1
J	IG	2	1
K	IL	2	0
L	K	1	0
M	-	0	0

Tabela 3: Primeira iteração

A Tabela 3 mostra como estão os pontos nesta fase. G pode ser visto como o nó pai, que possui filhos E, F,H, I e J.

Passo 5: Repetir para os vizinhos

Repita para cada ponto, de cada vizinho, com cluster marcado como 0 de maneira que todos os vizinhos sejam capturados recursivamente. Aqui se pode repetir a analogia com grafos: repetir até que todos os nós filho sejam folhas. O critério de parada na escolha dos vizinhos é o desligamento espacial dos mesmos, situação em que todos os pontos que possuir pelo menos um vizinho em comum está englobado. A escolha do ponto que possui mais vizinho sempre é feita para um único o ponto. No caso de empate, escolhe-se o vizinho que possui menor data e horário (desempate pelo critério cronológico).

Podemos verificar que até este passo, o cluster se forma da seguinte maneira (Figura 4.11):

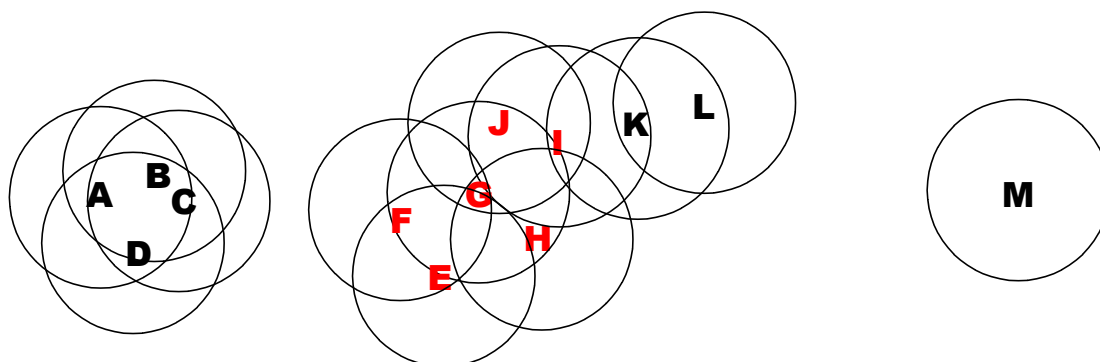


FIGURA 4.12: FORMAÇÃO DE UM CLUSTER

Para este passo, o exemplo é o que se segue: para todos os pontos marcados com 1 (E,F,G,H,I e J.) determinam-se os seus vizinhos (os vizinhos de E são F,G,H; os vizinhos de F são G,E e assim por diante).

Os pontos E, F, G, H, I, J estão marcados como *cluster* 1. Para cada vizinho (E,F,G,H,I,J,K), verifique se algum deles está marcado como 0. Se está, marque como pertencente a este cluster n+1 (neste momento cluster vale 1). Até aqui somente o ponto K não foi marcado como 1, então o marque como pertencente a este cluster.

	<i>Ponto</i>	<i>Pontos Vizinhos</i>	<i>Cluster</i>
	A	BCD	0
	B	ACD	0
	C	ABD	0
	D	ABC	0
1º teste	E	FGH	1
2º teste	F	GE	1
Ponto Inicial	G	EFHIJ	1
3º teste	H	GI	1
4º teste	I	G,JK	1
5º teste	J	IG	1
	K	IL	1
	L	K	0
	M	-	0

Tabela 4: Fase atual de clustering

A Tabela 4 apresenta a fase atual do método. No Passo 5 deve-se para repetir o processo até que todos os vizinhos dos pontos marcados como cluster =1 sejam visitados. Como K foi o último ponto a ser marcado como 1, verifique seus vizinhos (I,L). O ponto I não é verificado novamente, já que faz parte do cluster 1. O ponto L está marcado como 0, então o marque como cluster 1. Recursivamente, L é verificado. Como entre seus vizinhos (aqui apenas é K) já está marcado como 1 pare.

Como foi demonstrado, este passo termina quando todos os pontos com pelo menos um vizinho em comum são marcados com número de cluster corrente. Se não houver pontos dentro desta condição, passe para a próxima etapa.

Passo 6: Incrementar cluster

Depois de obtido o critério de parada, informe que todos os dados englobados formam o cluster n . Incremente o valor de cluster para $n + 1$. No exemplo dado cluster valerá 2.

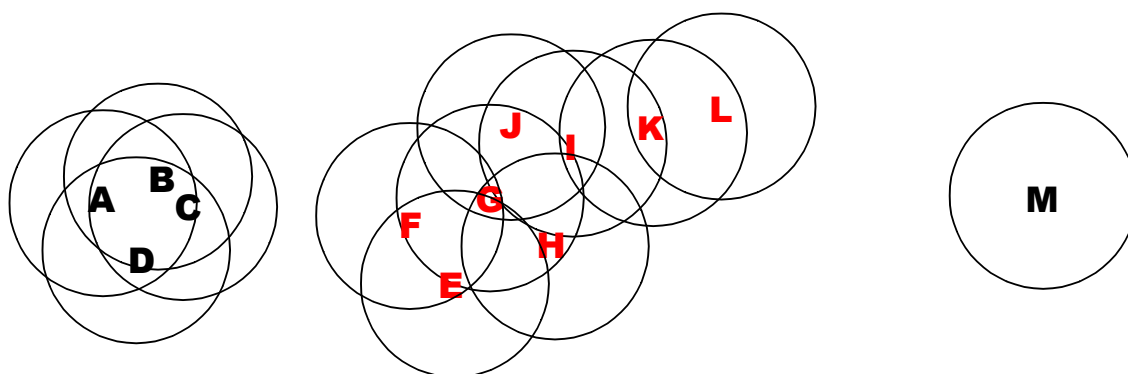


FIGURA 4.13: CLUSTER 1

O cluster 1 então é formado pelos pontos marcados com 1 (E,F,G,H,I,J,K e L). Como o critério de parada do Passo 6 foi obtido, isto é, houve desligamento espacial verificado pela vizinhança, o valor de cluster é incrementado. O critério de parada do método como um todo é verificado no Passo 7.

Passo 7: Terminar quando todos os pontos estiverem marcados diferentemente de 0

Enquanto houver pontos marcados como 0, retorne ao Passo 4 e siga até o Passo 7. Se todos os pontos que possuíam pelo menos um vizinho foram verificados, então o método acaba e o usuário pode verificar o resultado. Portanto, no exemplo dado, para o usuário o resultado final do algoritmo é apresentado na Tabela 5.

Ponto	Cluster
A	2
B	2
C	2
D	2
E	1
F	1
G	1
H	1
I	1
J	1
K	1
L	1
M	3

Tabela 5: Resultado final do clustering

Os dados da tabela acima podem ser vistos distribuídos espacialmente na ilustração a seguir:

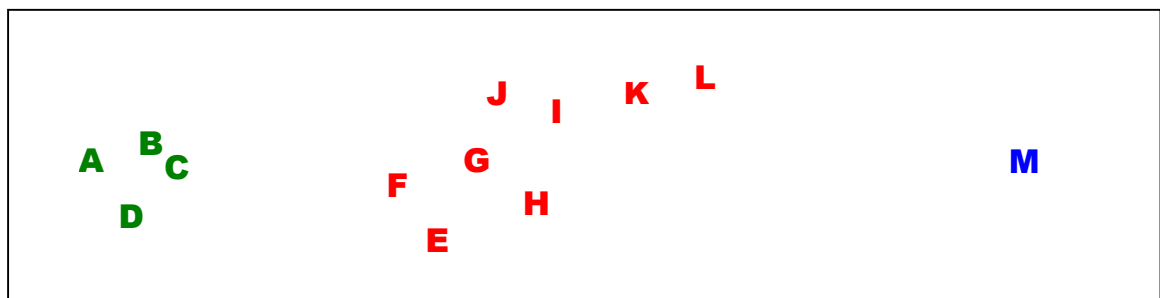


FIGURA 4.14: VISUALIZAÇÃO DO MÉTODO DE *CLUSTERING* PARA SGBDE

Este método tem como entrada os dados de um período x , e uma distância n que define até onde um ponto pertence a certa vizinhança. Analogamente podemos dizer que a distância e os pontos são como uma cidade. A definição da distância pode ser vista como tão longe estão os vizinhos, esta medida pode ser tida como limites de bairros e os pontos podem ser considerados os lotes. Enquanto um lote tiver vizinho rotule o mesmo como pertencente ao mesmo aglomerado, até que não ajam vizinhos em comum. A forma que o método pode ser implementado via consultas SQL será discutida na Seção 4.4.3.

4.4.3. Implementação

Como foi visto, o método se baseia em definir a distância de vizinhança e englobar os pontos com vizinhos em comum. Pode-se assim, descrever sintaticamente o algoritmo da seguinte forma:

```
selecione os pontos desejados dentro da base de dados
para cada ponto expanda em uma distância n
para cada ponto crie uma coluna “cluster” e atribua inicialmente o valor 0
repita
    selecione o ponto que possuir mais vizinhos
        atribua a este ponto o valor de cluster  $n+1$ 
    selecione todos os vizinhos do ponto que possui mais vizinhos e
    marcados como 0
        atribua aos vizinhos do ponto o valor de  $cluster(cluster\_atual)$ 
    repita
        selecione os pontos que estão marcados como cluster 0
        e que possui como vizinho um ponto marcado como cluster
            para cada ponto selecionado atribua o valor de cluster
        até todos os vizinhos do ponto estejam diferente de cluster = 0
    até todos os pontos (vizinhos) estejam num cluster diferente de cluster = 0
```

O algoritmo acima foi implementado em forma de consultas SQL, de maneira que uma tabela com os vizinhos de cada ponto seja criada. O uso de uma tabela com o ponto e os vizinhos do mesmo é encorajado visando a redução do custo de processamento. Esta tabela de vizinhos também pode ser indexada para otimizar a consulta. Apesar destes pontos positivos, deve-se ressaltar que para cada ponto o tamanho desta tabela pelo menos dobra de tamanho, já que além de armazenar os n pontos, os n vizinhos também serão armazenados. Além disso, poderá haver a duplicação de vizinhos, isto é, haverá vizinhos em comum. Esta etapa de criação da tabela de vizinhos é a que despende mais tempo dentro deste método.

A geração da tabela de vizinhos (*tabelavizinhos*) deverá calcular a área de abrangência da vizinhança para cada ponto. Após expandir cada ponto é testado se um outro ponto central está contido dentro desta área. O exemplo utiliza os atributos data,

hora, nanosegundo, ponto central da descarga atmosférica. O algoritmo visa o uso de uma chave primária, neste caso a data, hora (*raidatahora*) e nanosegundo (*rainanosegundo*). Aqui se optou concatenar (“||”) as duas colunas para distinguir a coluna de chave-primária.

O Sistema Geodésico Brasileiro utilizado é o SAD-69, que na tabela de SRID do PostGis possui código 4618. Como existem distorções na projeção, as coordenadas do ponto devem estar preferencialmente em UTM. Se não estiverem, as coordenadas de expansão são transformadas para coordenadas UTM, já que este tipo de coordenada possui característica de medição em metros, não havendo assim problemas de projeção incorreta. O código da coordenada UTM que melhor se adapta para a área do Paraná é o 22 Sul, que corresponde no *Postgis* o código 29192. As linhas correspondentes aos códigos 4618 e 29192 da tabela *spatial_ref_sys* com a descrição do *Datum* e das configurações da biblioteca PROJ4 podem ser encontradas no Anexo 1.

A consulta para a criação da tabela de vizinhos é descrita na sequência:

```
---Cria tabela de vizinhos
create table tabelavizinhos as
select distinct(b.raidatahora||' '||b.rainanosegundo) as
datavizinho,
  a.raidatahora as dataponto ,
  a.rainanosegundo as nanoponto,
  b.the_centrogeom
from tabelateste as a,
(
select
transform(buffer(transform(the_centrogeom,29192),8000),4618)
as buf,
      raidatahora,
      rainanosegundo,
      the_centrogeom
from tabelateste
) as b
where buf && a.the_centrogeom
and contains(buf,a.the_centrogeom)
and (b.raidatahora||' '||b.rainanosegundo) <>
(a.raidatahora||' '||a.rainanosegundo);
```

A *tabelavizinhos* possui, portanto: *datavizinho*: text; *dataponto*: timestamp without time zone ; *nanoponto*: numeric; *the_centrogeom*: geometry POINT.

A criação da tabela utilizou as funções de *buffer* (expande uma geometria segundo uma distância *n*). Como teste foi utilizado a distância de 8.000 metros ou 8 quilômetros (km). Primeiramente o ponto é transformado para coordenada UTM, para facilitar e aprimorar a medição da expansão do ponto. Como trabalhamos como coordenadas geográficas é necessário a reprojeção para o código 4618. Aqui utilizamos

a função **transform**. Observação: foi optado utilizar a transformação de coordenadas para demonstrar como pode ser utilizada esta funcionalidade e para verificar a performance (tempo de resposta) do método neste teste.

Após as devidas transformações da geometria deve-se testar se um ponto está contido na área de expansão (*buf*). Uma técnica de utilização de indexação espacial utiliza primeiramente o operador **&&** que significa: “o *bounding box* de uma geometria está sobreposta a outra geometria”. O *bounding box* de uma geometria pode ser visto um retângulo envolvente da geometria, utilizando os valores limitantes da geometria:

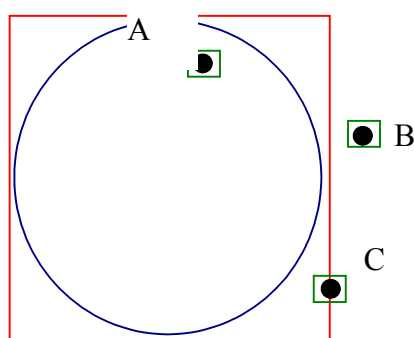


FIGURA 4.15: TESTE DE SOBREPOSIÇÃO DE GEOMETRIAS

A Figura 4.15 ilustra o teste do *bounding box* (em vermelho) para o *buffer* (em azul), com o *bounding box* (em verde) de um outro ponto (em preto).

Primeiramente é testado se os limites de uma geometria estão sobrepondo os limites de outra geometria. Os pontos que retornarem como verdade para esta condição continuam no teste (no exemplo são os pontos A e C). Se não retornarem verdade, estes serão excluídos do restante da consulta (no exemplo é o ponto B). Este teste filtra muitos pontos desnecessários, e assim diminui o conjunto de dados envolvidos na consulta SQL.

Após este teste, a consulta verifica se um ponto está contido dentro da figura através da função **contains**, onde a condição geometria *a.the_centrogeom* está contida em *buf* deve ser verdadeira. No exemplo, somente o ponto A é obtido como resultado desta consulta.

Perceba que na criação da tabela de vizinhos, alguns pontos que estão isolados, isto é, sem vizinhos, não serão testados como cluster. Isto se deve ao fato de que o mesmo forme seu próprio cluster. Neste caso, se houver o caso de não ter vizinho, não serão utilizados neste passo.

A tabela de vizinhos ainda está incompleta. Falta criar uma coluna para indicar qual cluster o ponto pertence:

```
alter table tabelavizinhos add column cluster integer;
```

Inicializa o atributo com o valor 0.

```
update tabelavizinhos set cluster=0;
```

O método desenvolvido nesta dissertação indica que nesta fase deverá ser feito um *looping* até que nenhum ponto possua o valor de cluster como 0. Uma variável auxiliar (*cluster_atual*) será utilizada para indicar o valor corrente de cluster. Aqui a variável “*cluster_atual*” tem valor 0.

O próximo passo captura apenas o ponto que possuir o maior número de vizinhos. O ponto com maior número de vizinhos significa que os pontos estão bem próximos. Utilizando este ponto, já no primeiro teste a maioria dos pontos é capturada, podendo até mesmo, capturar todos os pontos (neste caso seria um único grande cluster).

Uma subconsulta simples conta todos os vizinhos, e os ordena de forma decrescente pelo contador, depois pela data e hora, e depois pelo nanosegundo da incidência do raio. Um corte (*limit* 1) captura apenas o valor que possui o maior número de vizinhos. A sintaxe do Postgresql *limit* é muito utilizada e aqui facilita esse processo em caso de empate no número maior de vizinhos.

Após a consulta, altera-se o valor da variável auxiliar *cluster_atual* para *cluster_atual* +1. Esta variável serve para guardar o valor que irá modificar o atributo cluster da tabela tabelavizinhos, sendo, portanto *cluster* = *cluster_atual*. O código do primeiro passo é o seguinte:

```
cluster_atual:= cluster_atual + 1;
```

```
update tabelavizinhos set cluster= cluster_atual
where dataponto||' '||nanoponto in
(
  select dataponto||' '||nanoponto from
  (
    select count(datavizinho),dataponto,nanoponto
    from tabelavizinhos
    where cluster=0
    group by dataponto,nanoponto
    order by 1,2,3 desc
    limit 1
  ) as t
);
```

A consulta acima opta em buscar na *tabela_vizinhos* apenas os valores marcados

como cluster 0, ou seja, os que não foram “visitados” pela consulta. Esta restrição evita também que pontos que foram visitados e marcados como cluster, possam aparecer sobrepondo outro cluster.

Após achar e marcar o ponto que possuir mais vizinhos como *clusters*, os seus vizinhos também deverão ser marcados como *cluster*, já que eles são pontos dentro da mesma vizinhança.

```
---altera a coluna cluster para os vizinhos do ponto com mais vizinho

update tabelavizinhos set cluster= cluster_atual
where dataponto||' '||nanoponto in
(
    select datavizinho
    from tmpvizinhosa
    where cluster=cluster_atual
)
and cluster=0;
```

Aqui também se pode observar que há probabilidade de um ponto vizinho ser o segundo colocado na lista de pontos com mais vizinhos, logo, vários pontos serão englobados na primeira iteração. Outra questão deste passo é que, um ponto pode ter vários vizinhos e alguns destes vizinhos não terem pontos em comum com o primeiro ponto. Esta questão foi ilustrada na Figura 4.11, onde o ponto G possui vários vizinhos (EFHIJ) e um destes vizinhos, I, tem relação com J que por sua vez tem relação com L. J e L estão dentro do mesmo contexto de G, porém não são vizinhos diretamente.

Para resolver casos como este, o passo seguinte deve ser repetido até se encontrar o ponto em que todos os pontos espacialmente relacionados estejam marcados dentro do mesmo cluster:

```
---repita até return=0 atribuindo o valor do cluster_atual em
cluster para os pontos que estão sem cluster (=0) e aparecem
como vizinho em algum ponto já marcado com este valor de
cluster (valor cluster_atual)

update tabelavizinhos set cluster= cluster_atual
where dataponto||' '||nanoponto in
(
    select dataponto||' '||nanoponto
    from tabelavizinhos
    where cluster=0
    and dataponto||' '||nanoponto in
    (
        select datavizinho
        from tmpvizinhosa
        where cluster=cluster_atual
    )
);
```

A consulta busca os pontos na tabela de vizinhos que não foram visitados, verificando se algum ponto (dataponto) é retornado como vizinho (datavizinho). Aqui verifica duas tabelas temporárias. No primeiro momento (o mais interno) traz os vizinhos dos pontos marcados com o valor de *cluster_atual*. No segundo momento, uma subconsulta traz todos os pontos não visitados. A alteração é feita somente para os pontos que foram extraídos segundo este critério.

Se toda esta consulta retornar algum valor como “*update*” continue, senão repita a consulta. Retornando “*update 0*” volte ao Passo 1 até que não exista valor 0 para o atributo *cluster*.

Após estes passos, obtemos uma *tabelavizinhos* com os valores de *cluster* para cada ponto. É aconselhado também criar uma coluna *cluster* na tabela original (tabela teste). Esta tabela possui os valores não duplicados de cada ponto e também os pontos que não possuem vizinhos. Um simples *update* pega para cada ponto distinto os valores de *cluster* da *tabelavizinhos* e altera na *tabelateste*. Os pontos com valor nulo são os que pertencem ao seu próprio *cluster*.

Após esta modificação na tabela original, podemos: a) agrupar os pontos por *cluster*; b) unir os pontos de cada *cluster* e transforma-los em polígonos; c) unir os pontos por cada *cluster* e descobrir a coordenada que representa o centróide do *cluster*; d) unir os pontos e calcular o tamanho do *cluster* (da tempestade, por exemplo); e) calcular de forma simplificada a distância entre os pontos por *cluster*, entre outras possibilidades.

4.4.4. Experimentos

Nesta seção mostra-se alguns dos experimentos realizados com a aplicação do método desenvolvido e os apresentados nesta dissertação.

4.4.4.1. Experimento com 30 pontos

O primeiro experimento mostra o comportamento do método para uma quantidade limitada de pontos. Trinta pontos foram escolhidos para este teste, sendo que o período utilizado foi de 06-07-2003 às 14:00:24 e 677244300 ns até 06-07-2003 às 14:03:42 e 157879600 ns.

Os dados de entrada para este teste para data e hora estão no formato *yyyy-mm-dd hh24:mi:ss* e a geometria no formato ponto (POINT) com dimensão 2. Os dados

utilizados como teste se encontram no Apêndice 3. O método de representação espacial de *clustering* para este conjunto de dados foi testado com 3 limiares de vizinhança diferentes, escolhidos aleatoriamente: 8, 16 e 32 quilômetros. Para o primeiro experimento será utilizado como distância de vizinhança 8 km.

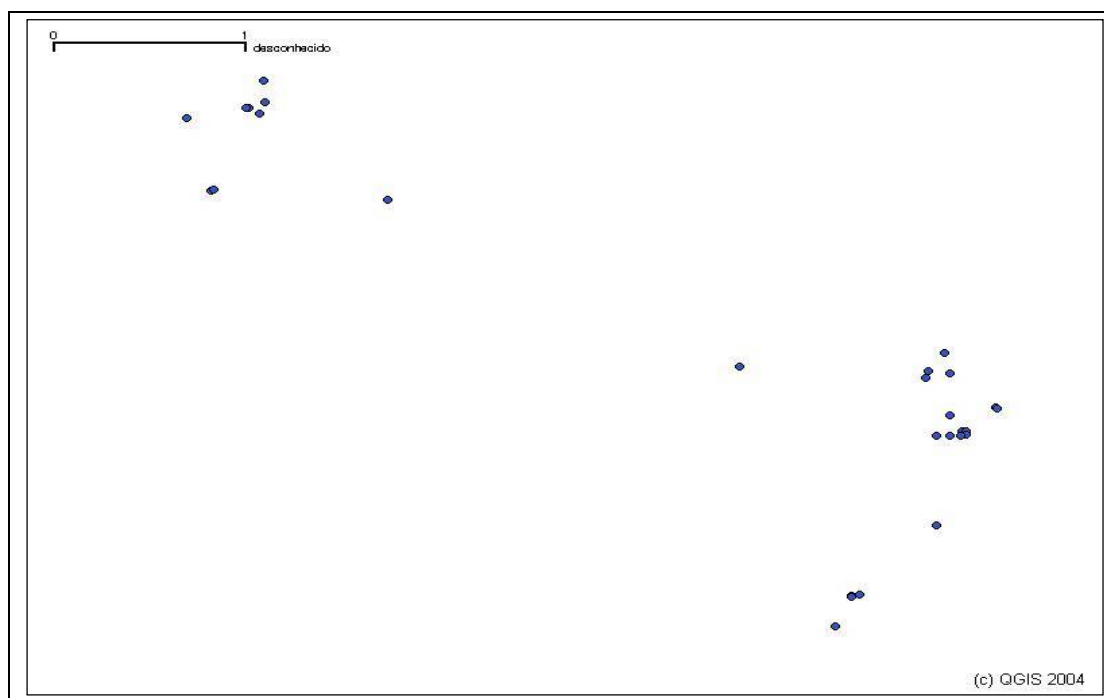


FIGURA 4.16: TESTE COM POUCOS PONTOS – PONTOS AVALIADOS

Na Figura 4.16 podemos visualizar a distribuição espacial dos pontos que possuem vizinhos, isto é, representa apenas os pontos cuja distância entre os objetos é de até 8 kms. Dos 30 pontos de entrada 21 pontos se encaixam nesta situação. Estes 21 pontos se encontram no Apêndice 1.

O resultado deste teste pode ser verificado na Tabela 6, que contém a indicação do número de elementos para cada cluster. Todos os passos realizados no teste para 8 km se encontram no Apêndice 4.

<i>Cluster</i>	<i>Elementos de cada aglomerado</i>
1	7 (vermelho)
2	5 (azul)
3	3 (rosa)
4	2 (roxo)
5	2 (verde)
6	2 (laranja)

Tabela 6: Resultado do teste com poucos pontos

Pela Tabela 6 observa-se que o método encontrou 6 *clusters* para os valores que foram informados. O número de elementos (pontos) para cada cluster é informado na segunda coluna. Para visualizar o resultado final deste experimento foi utilizado o software *QuantumGis* (QGIS,2005), para isso foram escolhidas, aleatoriamente, cores associadas a cada cluster . O resultado pode ser visualizado na Figura 4.17, abaixo:

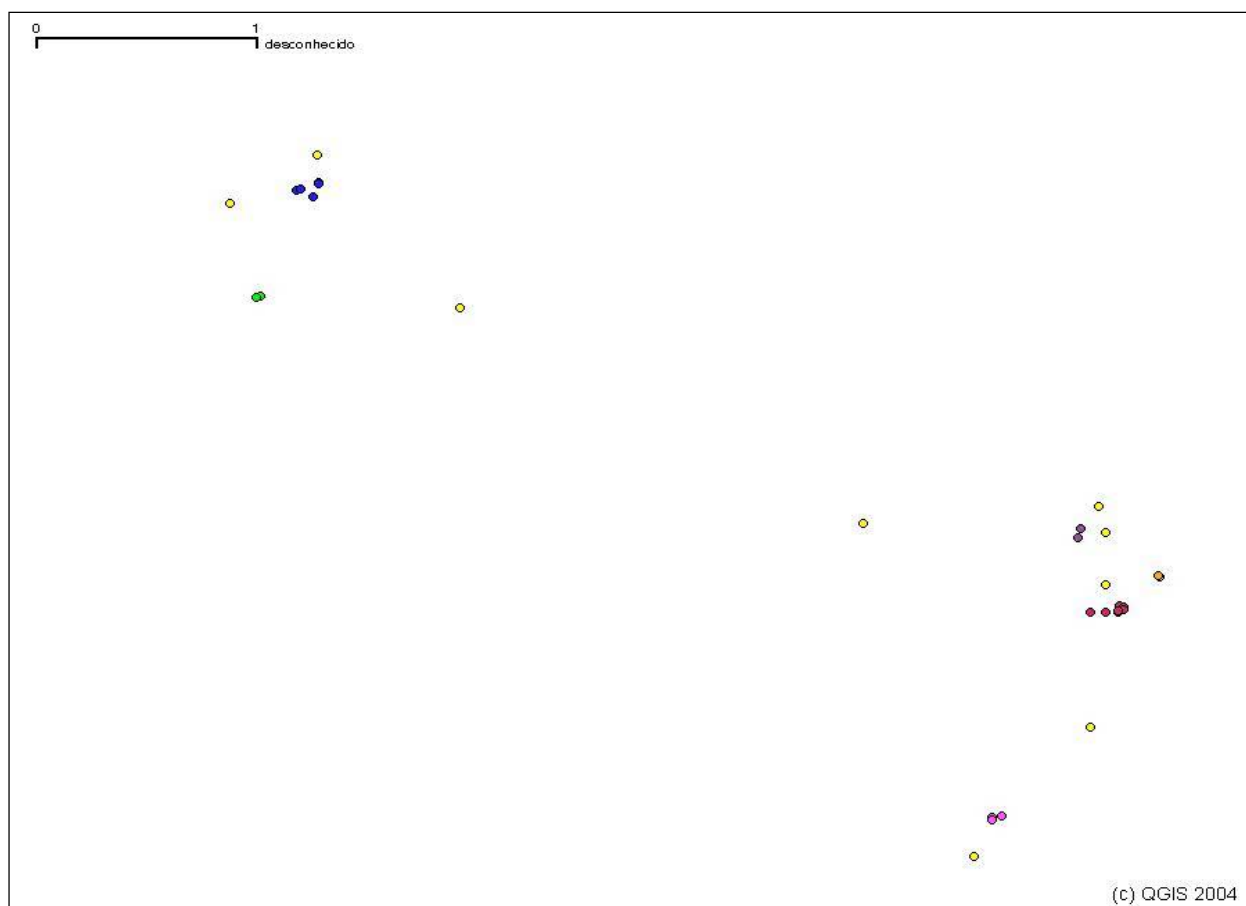


FIGURA 4.17: TESTE COM POUCOS PONTOS – 8 QUILÔMETROS DE VIZINHANÇA

A Figura 4.17 ilustra o resultado final deste teste: cluster 1 representado como vermelho, cluster 2: representado como os pontos em azul, cluster 3 representado como rosa, cluster 4 representado como roxo, cluster 5 representado em verde e cluster 6 em laranja. Os pontos em amarelo são objetos que não passaram por nenhuma iteração, isto é, não se encaixaram dentro do limiar de vizinhança de 8 km.

No exemplo observamos a ordenação dos clusters por ordem da quantidade de vizinhos. O cluster maior foi rotulado como 1, o segundo maior em 2, o terceiro maior em 3. Clusters que empataram no número de vizinhos foram rotulados com os números 4 e 5 e 6 com 2 pontos cada. O empate no número de vizinhos obedeceu a uma ordem

cronológica de desempate. Os demais pontos (em amarelo) são formados por *clusters* com um elemento (eles mesmos), sendo que estes pontos não entraram nas iterações do método, pois não tinham vizinhos dentro da área de abrangência. Este limiar de qualidade proposto pelo método permite filtrar somente os elementos espaciais desejados, diminuindo assim o tempo de resposta (desempenho) para o usuário.

O primeiro teste, com um limiar de vizinhança relativamente pequeno, obteve o resultado desejado: os pontos próximos espacialmente, numa distância máxima de 8 km entre si foram agrupados de forma satisfatória, sem intervenção a priori do número de clusters de saída.

Deseja-se saber o comportamento dos mesmos dados utilizados no primeiro teste, dentro de um limiar de segurança de 16 e 32 km.

O resultado deste segundo teste, considerando distância máxima de vizinhança 16 km pode ser observado na tabela a seguir:

Cluster	Elementos de cada aglomerado
1	8 (azul)
2	6 (rosa)
3	4 (roxo)
4	3 (verde claro)
5	2 (verde escuro)
6	2 (vermelho)

Tabela 7: Resultado com limiar de vizinhança de 16 km

A Tabela 7 demonstra o número de clusters, a quantidade de elementos em cada cluster com os mesmos valores de entrada do experimento para 8 km. O resultado deste teste pode ser observado na Figura 4.18:

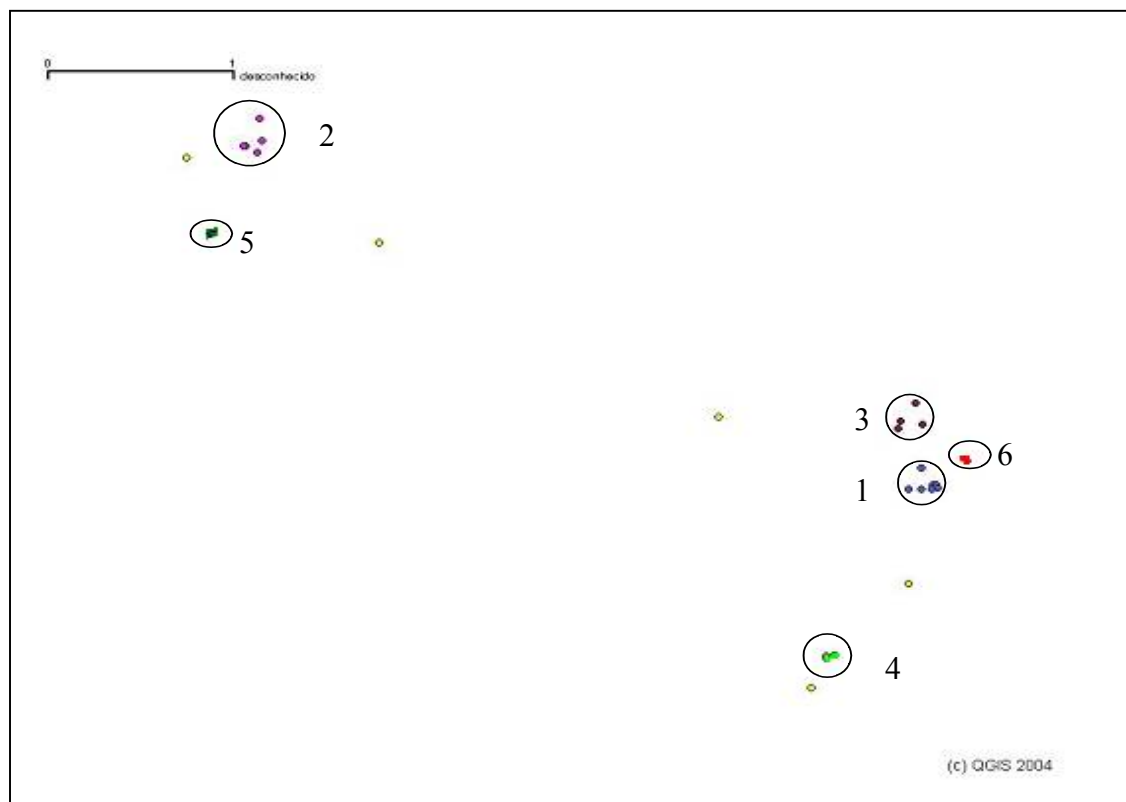


FIGURA 4.18: TESTE COM POUCOS PONTOS – 16 QUILOMETROS DE VIZINHANÇA

Do teste realizado com vizinhança menor de 8 km foi obtido 6 clusters distintos, mesmo valor de cluster obtido no texto com limiar de 16 km. A posição dos clusters foi a mesma, porém, com 16 km, 25 pontos foram englobados, contra 21 pontos englobados no teste com 8 km. Neste simples teste, observa-se que a medida de tolerância informada pelo usuário interfere diretamente no resultado do cluster.

Deve-se ressaltar que 8 e 16 km são valores que obtiveram um bom caso de comparação neste teste, sendo que qualquer valor inteiro positivo pode ser inserido neste contexto. A flexibilidade de poder informar a medida de vizinhança pelo usuário, pode ser avaliada como um ponto positivo, pois a análise de cluster pode variar dependendo do objetivo buscado. Por exemplo, dependendo dos dados, uma medida de 8km é suficiente determinar para o núcleo de uma tempestade elétrica. Em outros casos, por exemplo, focos de calor ao redor de uma indústria madeireira, 16 km é um bom valor.

Para mais uma análise comparativa, o valor de vizinhança de 32 km foi testado. O seguinte resultado foi obtido:

Cluster	Elementos de cada aglomerado
1	14 (azul)
2	6 (verde)
3	4 (azul claro)
4	2 (laranja)

Tabela 8: Teste com vizinhança de 32 kms

A visualização do resultado do teste feito para 32 km é apresentada na Figura a seguir:

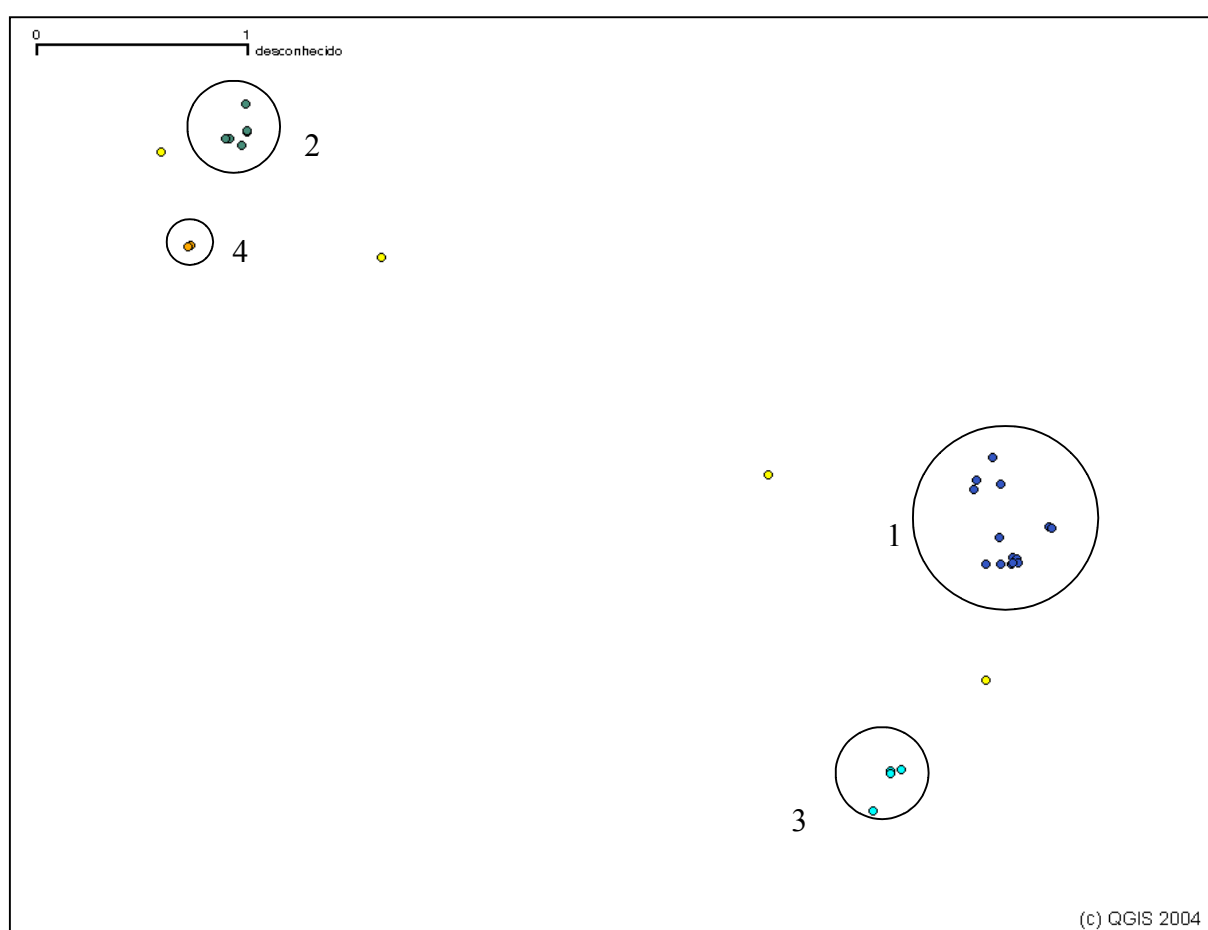


FIGURA 4.19: TESTE COM POUCOS PONTOS – 32 QUILÔMETROS DE VIZINHANÇA

Na Figura 4.19 observou-se 26 pontos que formam clusters e 4 pontos que não se encaixaram no limiar de 32 quilômetros de vizinhança. O resultado deste teste pode ser verificado no Apêndice 5. Um experimento comparando este resultado com o método aplicando o K-Means do Weka foi realizado e será apresentado na Seção 4.4.4.4.

A diferença entre as medidas utilizadas para vizinhança máxima, neste experimento, pode ser observada na Figura 4.20, onde os pontos em amarelo representam descargas atmosféricas, em roxo a expansão do ponto pelo limiar de 8 quilômetros, em verde 16 e 32 quilômetros.

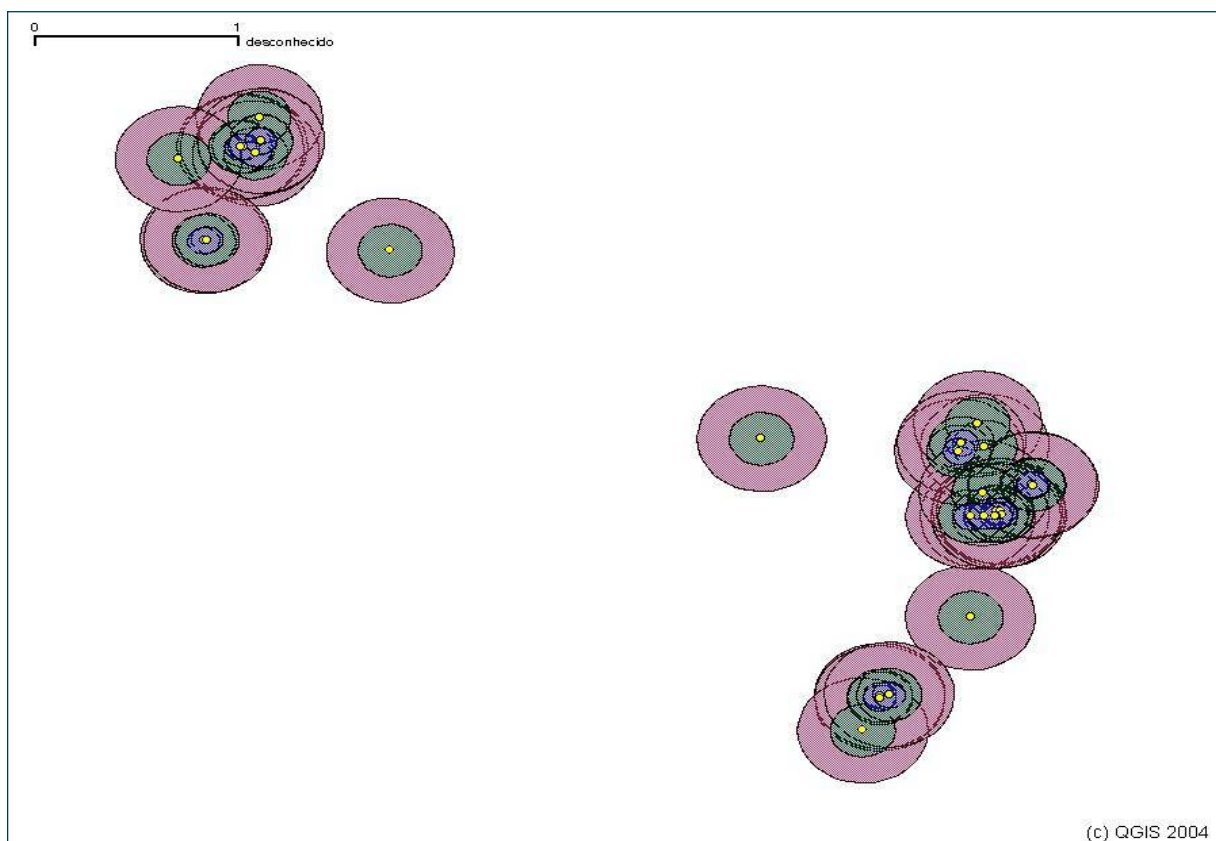


FIGURA 4.20: DIFERENÇA ENTRE LIMIARES DE VIZINHANÇA

A avaliação deste teste verificou a importância da interação do usuário para a qualidade dos resultados. Todos os resultados apresentaram agrupamentos na forma correta, o que varia é medida de vizinhança. Dependendo da aplicação almejada 8 quilômetros é a resposta mais correta, em outros 16, 32, 80 ou 100 quilômetros. Neste exemplo específico, 32 km se demonstrou de forma mais agradável visualmente.

4.4.4.2. Experimento com 500 pontos

Neste experimento testou-se o agrupamento utilizando 500 pontos e observar algumas aplicações práticas destes clusters formados. Os testes foram realizados em dois tempos: sendo o segundo teste na hora imediatamente posterior do primeiro grupo

testado. A data dos testes foram o dia 06 de julho de 2003 início 14:52:42 e 52108200 ns até às 15:09:57 e 649189100 ns para o primeiro teste, e 15:09:58 e 5755500 ns até 15:22:38 e 211900100 ns. Podemos verificar a disposição dos pontos nos dois momentos na ilustração a seguir:

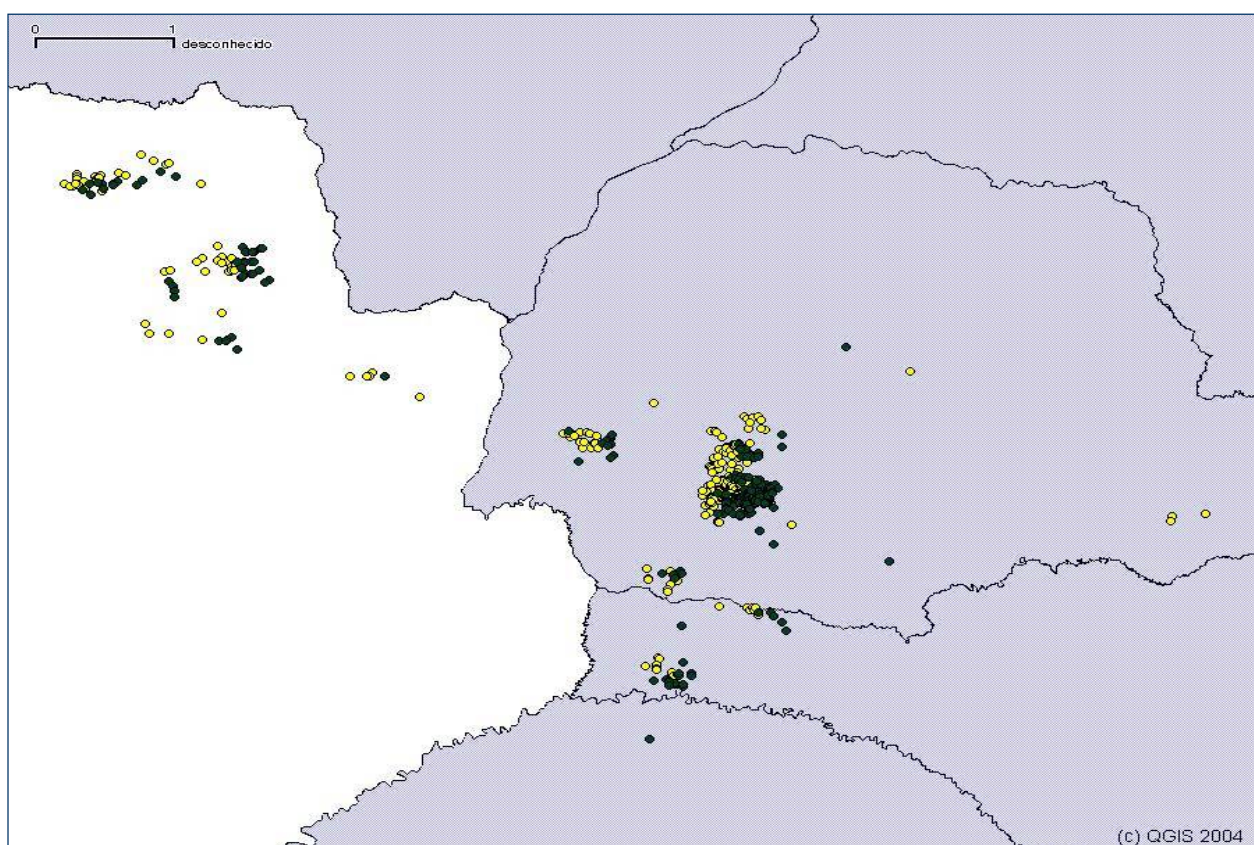


FIGURA 4.21: DESCARGAS ATMOSFÉRICAS EM DOIS MOMENTOS

Este é um exemplo típico que podemos visualizar os pontos correspondendo a descargas atmosféricas em dois momentos: em amarelo o primeiro grupo e em verde o segundo grupo. Até este momento, através de uma consulta sql, não é possível distinguir quais são os pontos que foram os aglomerados, qual a diferença entre o aglomerado 1 do primeiro instante com o aglomerado 1 do segundo instante, qual a distância de deslocamento entre estes clusters. Aplicando o método obtém-se:

Cluster	Elementos de cada aglomerado
1	383
2	23
3	28

Cluster	Elementos de cada aglomerado
4	19
5	15
6	9
7	8
8	4
9	5
10	3

Tabela 9: Clustering momento 1

No primeiro momento foram encontrados 10 clusters, sendo que 497 pontos se encaixaram em algum agrupamento. Neste teste pode-se verificar a grande aglomeração de informações no cluster número 1. A área calculada (com métodos do Postgis) para o primeiro cluster é de 424.91 km.

Para o segundo momento o resultado foi o seguinte:

Cluster	Elementos de cada aglomerado
1	409
2	21
3	19
4	13
5	10
6	6
7	5
8	5
9	4

Tabela 10: Clustering momento 2

No segundo momento foram encontrados 9 clusters, nos quais 492 pontos se encaixaram, verificando novamente a grande aglomeração de informações no cluster número 1, porém, a área calculada para o primeiro cluster é de 291.60 km.

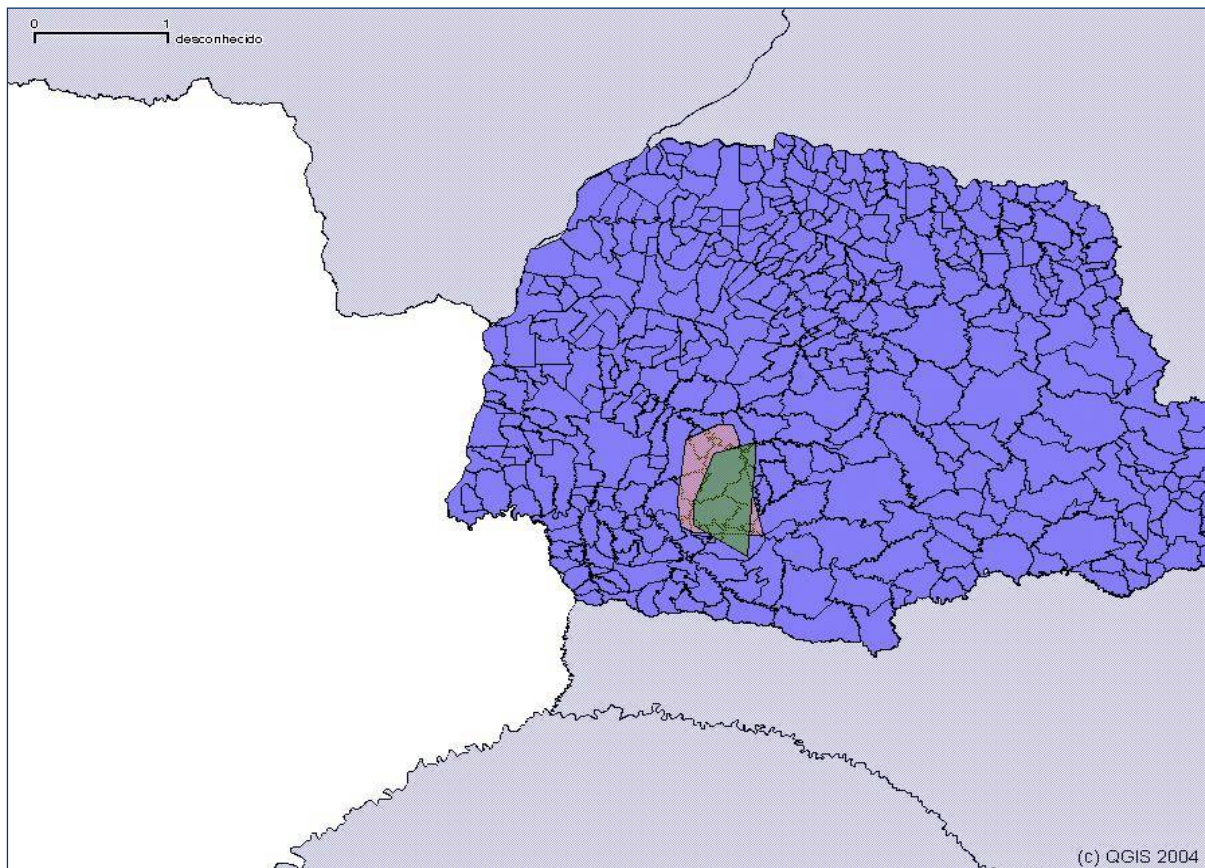


FIGURA 4.22: CLUSTERS VISTOS COMO POLÍGONOS

Apesar da diminuição da área aonde os *clusters* maiores foram localizados, nos dois momentos, verifica-se que se trata do mesmo, visto que a deslocação dos aglomerados se apresenta de forma constante. Esta informação pode ser verificada através da função ***convexhull*** na Figura 4.22. A Figura ilustra o agrupamento convexo dos pontos dos clusters maiores formados pelos testes acima. Em rosa está representando os pontos do cluster 1 do primeiro teste e em verde os pontos do cluster 1 do segundo teste.

Com estes dados, comparando com o mapa dos limites municipais do Paraná - também armazenado no SGBDE, podemos verificar que: (a) A distância do ponto central do cluster 1 no primeiro tempo até o cluster 1 do segundo tempo é de 12.733 km. (b) As cidades atingidas pelo cluster 1 no primeiro teste são: Altamira do Paraná, Cândói, Chopinzinho, Diamante do Sul, Espigão Alto do Iguaçu, Guaraniaçu, Laranjal, Laranjeiras do Sul, Marquinho, Nova Laranjeiras, Palmital, Porto Barreiro, Quedas do

Iguaçu, Rio Bonito do Iguaçu, São João, São Jorge d'Oeste, Saudade do Iguaçu, Sulina e Virmond.c) As cidades atingidas pelo cluster 1, pelo segundo teste, são: Candói, Chopinzinho, Espigão Alto do Iguaçu, Guaraniasçu, Laranjal, Laranjeiras do Sul, Marquinho, Nova Laranjeiras, Palmital, Porto Barreiro, Rio Bonito do Iguaçu, Saudade do Iguaçu, Sulina. d) O acompanhamento de mais fases desta tempestade elétrica descreve o perfil que a mesma toma no decorrer do tempo.

Este experimento tentou demonstrar uma situação prática, que o método de detecção e representação de *clustering* pode ser utilizado. Além disso, este teste tentou demonstrar o comportamento do método com um número grande de pontos, aqui representado por descargas atmosféricas. A performance na detecção de *clustering* nestes testes se demonstrou na faixa de aproximadamente 2 minutos, já no caso de transformar os pontos com cluster em comum, o processo foi mais demorado, na faixa dos 6 minutos. Esta perda de desempenho se justifica pelo uso de funções de união de geometrias para achar os pontos mais limitantes, isto é, achar o envoltório convexo de um conjunto de objetos com características em comum.

4.4.4.3. Experimento com 1000 pontos

Este experimento visou um caso de teste aonde houvesse uma grande correlação entre os pontos, isto é, muitos vizinhos em comum. Além disso, este experimento visou verificar a performance do método para um caso com mais pontos envolvidos. Aqui neste teste foram utilizados 1000 pontos, sendo o período da consulta: 06 julho de 2003 às 14h03min: 42 e 157879600 ns até 06 de julho de 2003 às 14:52:42 e 356951542 ns. Neste teste percebeu-se que dos 1000 pontos distintos foram obtidos 52928 vizinhos, isto é, uma média de 52 vizinhos por ponto. O tempo de execução deste algoritmo de 941053.220 ns ou 15 minutos. Esta performance é baixa, mas comparando com a quantidade de pontos e distância de aproximação dos objetos baixa (muitos vizinhos) e utilizando funções pesadas (como *buffer*) da extensão *Postgis*, este tempo é considerável.

A Tabela 11 apresenta o resultado deste teste:

Cluster	Elementos de cada aglomerado
1	773 (vinho)
2	63 (verde escuro)
3	35 (vermelho)
4	13 (azul marinho)
5	20 (azul claro)
6	12 (laranja)
7	13 (roxo)
8	6 (verde)
9	6 (magenta)
10	4 (amarelo)
11	7 (rosa)
12	5 (marrom)
13	3 (verde claro)
14	2
15	2
16	2
17	2
18	2

Tabela 11 : Teste com mil pontos

A Figura 4.23 ilustra os primeiros 13 *clusters* formados pelo método.

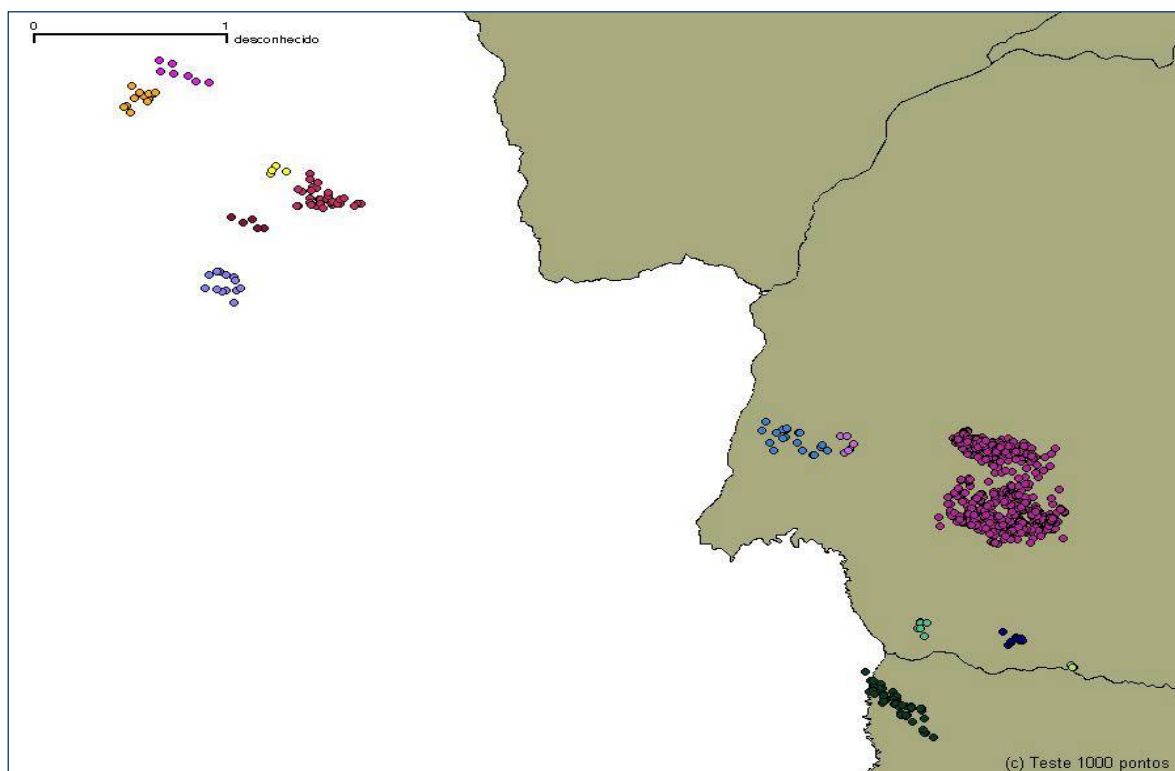


FIGURA 4.23: TESTE 1 COM MIL PONTOS – 8 QUILOMETROS DE VIZINHANÇA

Pode-se observar que a Figura 4.23 está referenciada com o contorno dos Estados do Brasil. Nesta ilustração observamos a aplicação do método descobrindo regiões afetadas com uma grande quantidade de descargas atmosféricas. Neste caso, percebe-se sobre a região de Cascavel uma grande quantidade de descargas atmosféricas sobre a região de Cascavel - PR.

4.4.4.4. Análise Comparativa

Foram realizados dois experimentos utilizando o método descrito nesta dissertação comparando o resultado com a aplicação do algoritmo K-Médias e EM do Weka.

Experimento 1

Para o primeiro experimento, foi tomado como base o teste realizado com 30 pontos, para 32 km de vizinhança. A visualização deste teste pode ser observada na Figura 4.19. Com os mesmos dados utilizados neste teste, foram realizados os procedimentos com o Weka utilizando o algoritmo K-Means e posteriormente utilizando o algoritmo EM.

Como no algoritmo K-Means deve ser informado o número de clusters *a priori*, utilizará o valor 4 (N), que foi o número de clusters obtido pelo método apresentado. A linha de comando executada para este teste foi a seguinte:

```
java -cp "weka.jar" weka.clusterers.SimpleKMeans -N 4
-p 0 -t testeComparativo.arff
```

O parâmetro **-p** significa predict e **-t** significa o arquivo “.arff” que será utilizado no teste. O resultado para o teste aplicando K-Médias do Weka foi o seguinte:

Cluster	Elementos de cada aglomerado
1	9 (azul)
2	4 (verde)
3	6 (azul claro)
4	11 (laranja)

Tabela 12: Resultado teste utilizando Weka- K-Means

A Tabela 12 demonstrou o número de elementos para cluster, bem como a cor associada para a visualização na Figura 4.24 abaixo:

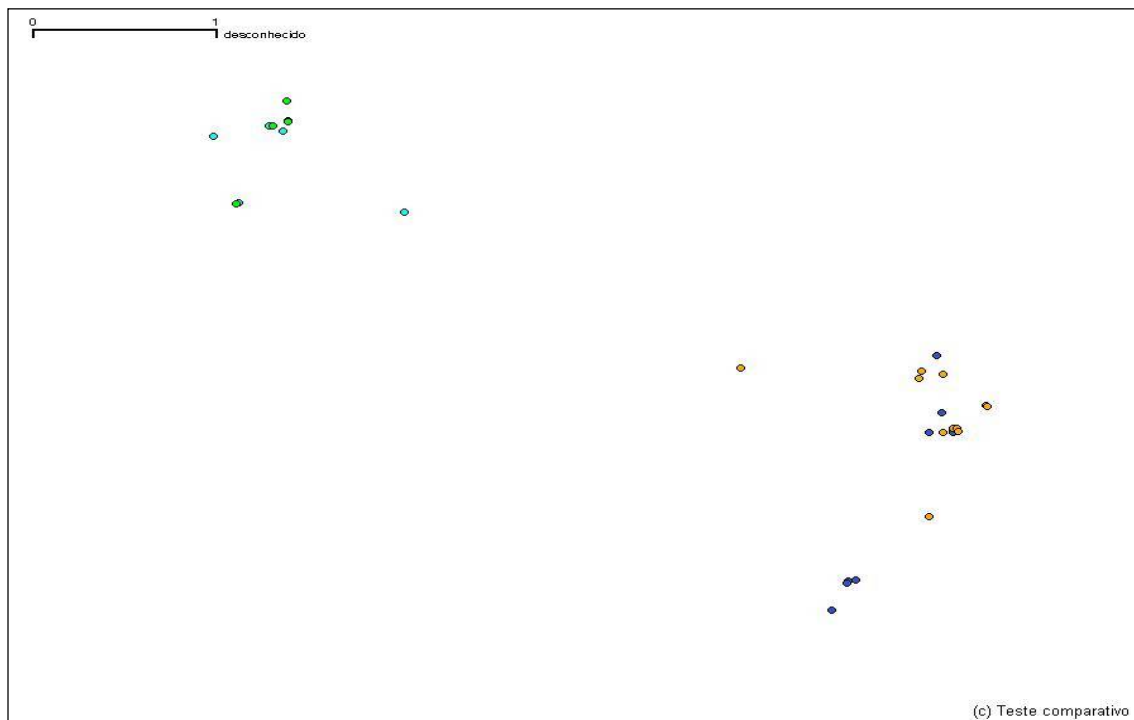


FIGURA 4.24: CLUSTERING FORMADO PARA 30 PONTOS APLICANDO WEKA K-MEANS

A comparação entre as Figuras 4.19 (método) e 4.24 (Weka - K-Médias) demonstra a coerência do método desenvolvido. Em ambas as imagens, os pontos referentes ao cluster 1 foi atribuído a cor azul, para o cluster 2 verde, cluster 3 azul claro, cluster 4 laranja.

Neste mesmo experimento, foi utilizado o algoritmo EM, do Weka. Basicamente, este método de *clustering* mede probabilisticamente os membros dos clusters, baseado em um ou mais distribuições de probabilidade. A meta do algoritmo é maximizar a probabilidade da distribuição dentro do cluster final. (CLUSTER ANALYSIS, 2004). O algoritmo EM (*Expectation Maximization*) de *clustering* é descrito mais detalhadamente por Witten e Frank (WITTEN, 2000). Para o algoritmo EM a seguinte linha de comando foi executada:

```
java -cp "weka.jar" weka.clusterers.EM -N 4 -p 0 -t  
/tmp/teste.arff > /tmp/raioesteste.txt
```

Para este teste, o seguinte resultado foi obtido:

Cluster	Elementos de cada aglomerado
1	9 (azul)
2	2 (verde)
3	11 (azul claro)
4	8 (laranja)

Tabela 13: Resultado teste utilizando Weka- EM

A Tabela 13 demonstrou o número de elementos para cluster, bem como a cor associada para a visualização na Figura 4.25 abaixo:

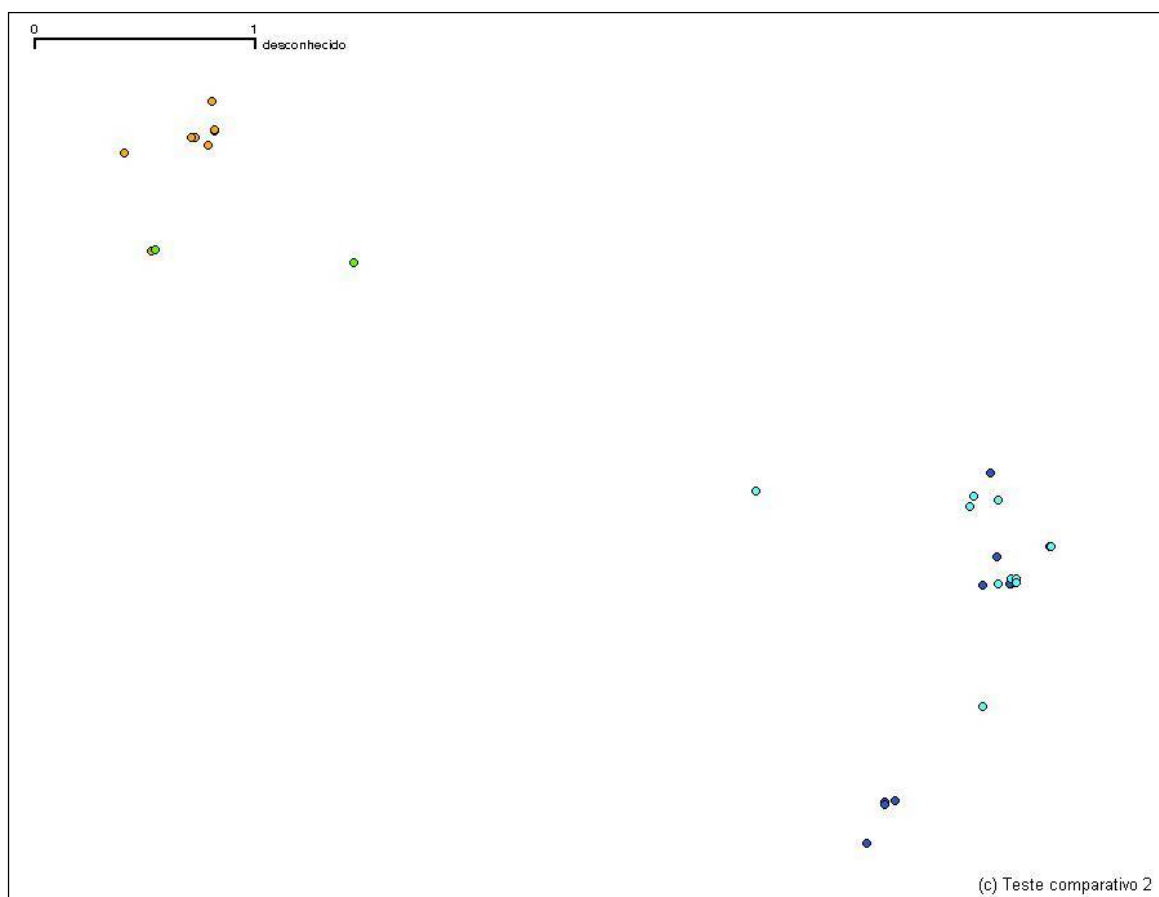


FIGURA 4.25: CLUSTERING FORMADO PARA 30 PONTOS APLICANDO WEKA EM

A comparação entre as Figuras 4.19 (método) e 4.25 (Weka -EM) demonstra mais uma vez a coerência do método desenvolvido. Assim, como foi feito no teste anterior, em ambas as imagens os pontos referentes ao cluster 1 foi atribuído a cor azul, para o *cluster 2* verde, *cluster 3* azul claro, *cluster 4* laranja.

Entre os 3 testes realizados neste experimento, o método de clustering espacial demonstrou um agrupamento visivelmente mais eficiente que os algoritmos K-Means e Weka no qual foram atribuídos valores de K a priori. Os dados e os resultados do *clustering* realizado pelo Weka se encontram nos apêndices 5, 6 e 7.

Experimento 2

Para o segundo experimento, foram utilizados dados de 15 minutos do dia 15 de junho de 2005 (04:30:30 até 04:44:57) e foram comparados os métodos utilizando o algoritmo K-Médias pelo Weka, e os mesmos dados para o método apresentado nesta dissertação.

Para o primeiro teste deste segundo experimento foi configurado o valor de K= 8 clusters. Como o K-Médias considera todos os pontos, mesmo os que não pertencem a clusters o resultado foi o seguinte:

Cluster	Elementos de cada aglomerado
1	73 (roxo)
2	117 (vermelho)
3	110 (azul)
4	111 (amarelo)
5	56 (rosa)
6	70 (verde escuro)
7	9 (verde claro)
8	15 (magenta)

Tabela 14: Teste com o algoritmo K-Means

A Tabela 14 demonstrou o número de elementos para cluster, bem como a cor associada para a visualização na Figura 4.26 abaixo:

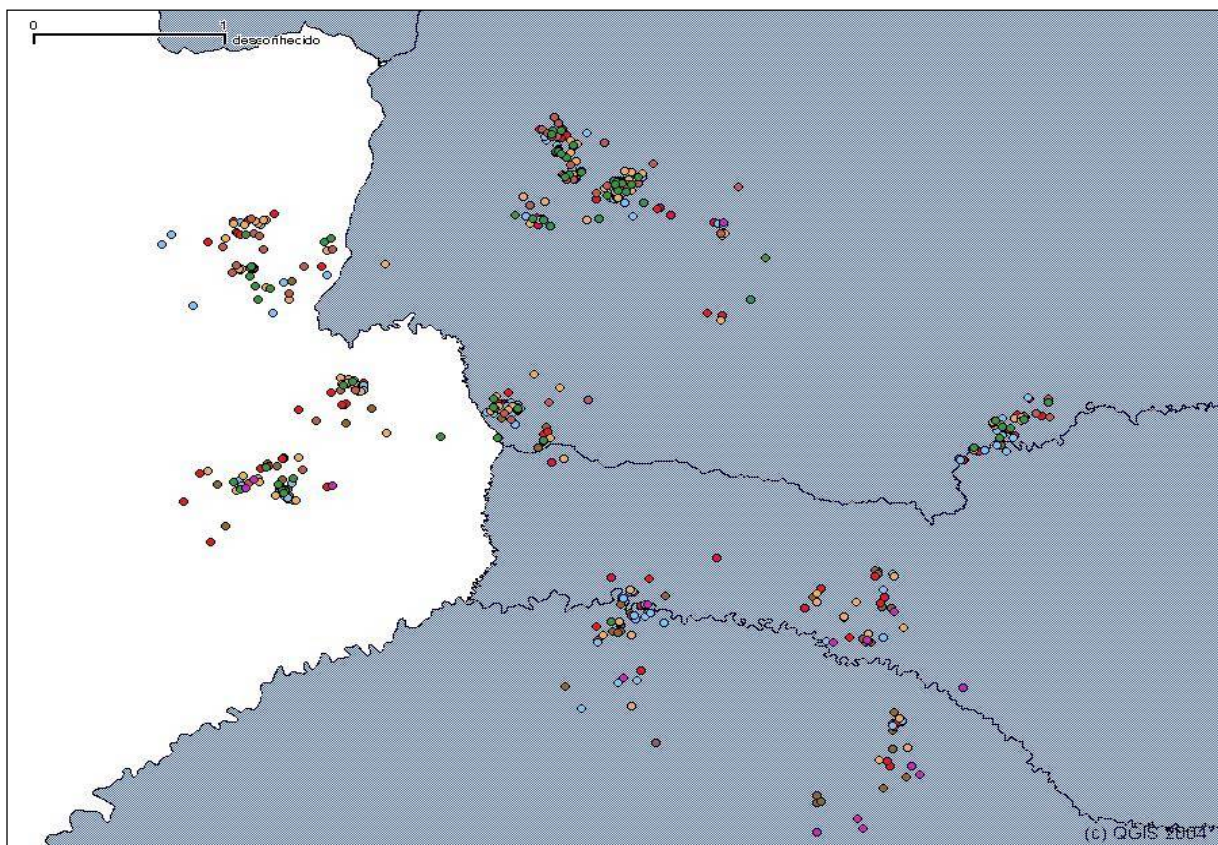


FIGURA 4.26: TESTE UTILIZANDO O WEKA

A Figura 4.26 não apresentou um conjunto pequeno de *clusters* que se formou no mar. Para o segundo teste, utilizando o método apresentado nesta dissertação, foi utilizado um limiar de 32 km indicando a vizinha máxima entre os pontos. Todos os pontos dentro deste limiar foram considerados vizinhos. Nenhum valor de quantos aglomerados desejados foi informado pelo usuário, deixando o método decidir a melhor quantidade de grupos ou *clusters*. Para este teste obtivemos o seguinte resultado:

Cluster	Elementos de cada aglomerado
1	116 amarelo
2	37 rosa magenta
3	81 verde claro
4	36 azul
5	29 verde piscina
6	21 verde oliva
7	18 vinho
8	7 (não aparece na figura porque estão no mar)

Tabela 15: Teste com o método de representação espacial de clustering

Podemos visualizar o resultado do teste na Figura 4.27:

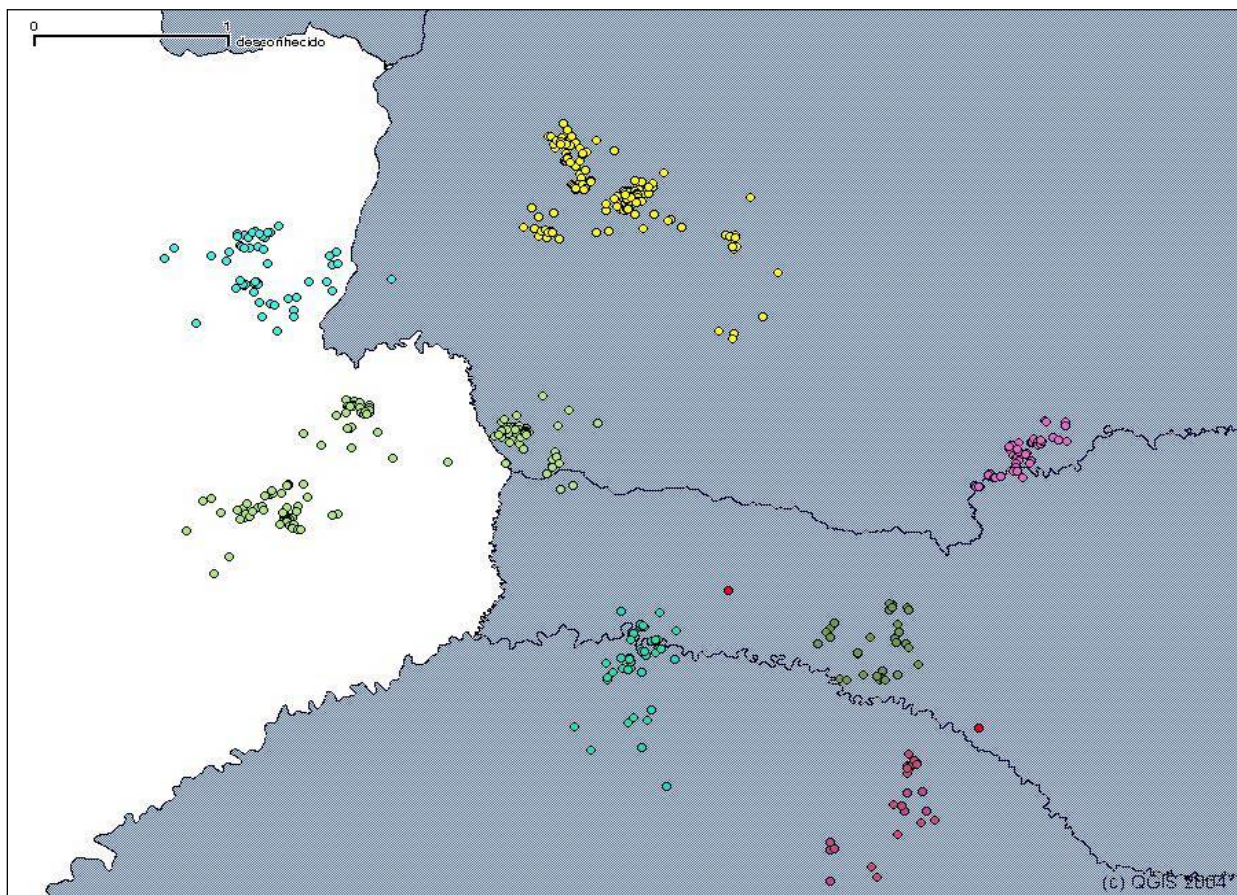


FIGURA 4.27: TESTE UTILIZANDO O MÉTODO DE CLUSTERING ESPACIAL

Os pontos em vermelho denotam pontos que pertencem a nenhum cluster. O teste demonstrou que o Weka classificou os elementos utilizando características de pico de corrente (que possui grande variância) e os dados temporais das descargas atmosféricas. Estas informações foram agrupadas segundo critério estatístico e do algoritmo K-Médias. O teste revelou clusters formados de forma dispersa e de entendimento confuso.

O experimento utilizando o método descrito neste trabalho além de agrupar os dados naturalmente, isto é, sem interferência do usuário em relação ao número de *clusters*, foi utilizado um limiar sugerido. Este limiar é flexível e depende da aplicação no qual o usuário deseja desempenhar o *clustering*.

Com relação ao custo do processamento, ambos os testes obtiveram tempo de resposta (desempenho) similar.

Além do exposto acima, o método apresenta visualmente um resultado esperado, isto é, os pontos mais próximos estão agrupados corretamente e a distância entre os clusters é maior que intracluster. O experimento também salientou pontos que

pertencem a nenhum grupo, pois possui distância incompatível com os demais *clusters*. Estes pontos são calculados pelo K-Médias ou pelo EM e podem interferir negativamente no resultado final.

Observação: Outros estudos realizados abordando *clustering* e utilizando a mesma base de descargas atmosféricas da RINDAT (SIMEPAR, 2006), neste período de tempo, podem ser observados no trabalho de Vasconcellos (VASCONCELLOS *et al*, 2006). O trabalho citado anteriormente utilizou o pacote do SQL Server 2005 Beta, o Microsoft Business Intelligence.

4.4.5. Conclusões Gerais sobre o Método Desenvolvido

O método sugerido obteve o resultado desejado. O algoritmo apresentado neste trabalho obteve sucesso em todos os testes realizados. Faltam ainda neste algoritmo, mais estudos no sentido de diminuir o tempo de resposta (desempenho) das consultas. Além disso, este método contemplou o problema identificado pela dissertação, isto é, dado uma relação tempo–espaço encontre para cada ponto o cluster ao qual pertence se desejar agrupe estes pontos e transforme-os em polígono, linhas, ou qualquer outro objeto espacial.

A dissertação se baseou em demonstrar o método que resolve o clássico problema de *clustering* através de técnicas e funcionalidades do *Postgis*. Através deste método uma aplicação para automatizar os passos definidos pode ser realizada. Depois da automatização este método pode ser inserido num *SGBD* com extensão espacial.

O *SGBD* utilizado foi o *PostGreSQL* com *Postgis*. O *PostGis* tem bom desempenho na execução das consultas espaciais aliado a algumas técnicas de otimização de consultas e indexação espacial. Esses últimos obtiveram um peso importante nas consultas em tabelas muito grandes.

Esta metodologia visou abordar todos os requisitos de *clustering* citados por KAESTNER (KAESTNER, 2006) que são: a escalabilidade e medida de qualidade. Estes requisitos foram utilizados para validar esta metodologia. Quanto ao requisito de escalabilidade: (i) habilidade de tratar diferentes tipos de atributos; (ii) Descoberta de clusters de formas arbitrárias; (iii) Requisitos mínimos de conhecimento de domínio para determinar parâmetros de entrada; (iv) Habilidade para tratar ruído e desvios; (v) Insensibilidade à ordem de entrada dos registros; (vi) Alta dimensionalidade; (vii) Incorporação de restrições específicas dos usuários; (viii) Interpretabilidade e utilidade.

Quanto ao requisito de medida de qualidade: (i) Métrica de similaridade/dissimilaridade: expressa em termos de função de distância $d(i, j)$; (ii) Existe uma função de “qualidade” que é uma medida da “adequação” de um cluster; (iii) Existem definições de funções de distância que são diferentes para variáveis intervalares, booleanas, categóricas e proporções; (iv) Pesos devem ser associados às variáveis baseados na aplicação e na semântica dos dados; (v) É difícil definir “suficientemente similar”, pois tipicamente esta avaliação é subjetiva.

Pode-se concluir ainda, que o método de *clustering* para banco de dados espaciais sugeridos nesta dissertação possui características:

- a) Característica de busca: Busca exaustiva do melhor conjunto de dados, onde a procura realizada na tabela de vizinhos para o ponto que possui mais elementos associados é captado para os pontos que não foram visitados.
- b) Interferência do usuário: O algoritmo encontra clusters automaticamente, isto é, não é necessário informar a quantidade de clusters desejados. É necessário apenas um parâmetro de distância de agrupamento sugerido pelo usuário. Por outro lado, a interferência do usuário pode retornar clusters mais flexíveis, isto é, depende do propósito utilizado na busca e área que o usuário deseja abordar. Dependendo do caso uma distância entre os elementos de 10 km é a que se adapta melhor, em outros casos uma distância de 30 km, por exemplo, é a mais indicada.
- c) Resultados: Obteve boas soluções nos casos utilizados como teste quando o número de pontos é menor que 1000 e limiar de distância entre os pontos não seja muito grande (acima de 100 km, por exemplo). O conjunto de objetos se demonstra homogêneo.
- d) Categorização do método. O método se baseia na distância expandida ao redor de cada ponto e na vizinhança entre os pontos.
- e) Medida de qualidade: A definição da qualidade entre os grupos depende da aplicação e das características subjetivas destes pontos. No estudo de caso realizado, os clusters possuem elevada semelhança entre os elementos do mesmo grupo, baixa semelhança entre os grupos diferentes.
- f) Métricas para a definição de semelhança: geralmente a medida realizada para detectar a semelhança entre os objetos é a soma da diferença da distância entre os pontos, como por exemplo, a diferença entre as distâncias de um ponto A e um ponto B deve ser zero (GAMA, 2002)

A distância entre os clusters é realizada pelo centróide de cada agrupamento com todos os seus vizinhos. O método possui características de particionamento, como é visto no algoritmo K-Médias. O método também apresenta características do algoritmo utilizando grafos, no qual o ponto que possui mais vizinho pode ser visto como a raiz da árvore. A tabela que possui todos os vizinhos representa os nós filhos. O método possui características algoritmo QT em relação ao limiar de qualidade. No algoritmo QT limiar significa o ponto cuja distância máxima deve ser inferior.

No método faz parte da área de abrangência de vizinhança, isto é, dentro deste limiar, todos os pontos são considerados vizinhos. Além disso, o algoritmo QT escolhe aleatoriamente um objeto de uma lista de objetos para então verificar todos os pontos dentro da distância mínima estipulada e assinalar como cluster. O QT também se difere, pois se pode informar o número mínimo de similaridade e máximo de tamanho de cluster, além de a distância máxima ser sempre recalculada. O método buscou a interferência mínima do usuário, naturalmente organizando os pontos em clusters.

Os métodos de *clustering* também têm sido utilizados em análise exploratória de dados e em procedimentos de regionalização. Nestes, os algoritmos de *clustering* com restrição de contigüidade espacial são utilizados com o objetivo de agrupar áreas homogêneas em regiões contíguas (OPENSHAW e WYMER, 1995; ASSUNÇÃO, 2004; NEVES *et al* 2001).

Esta função de detecção de *clustering* para Banco de Dados *Open Source* será de importância, principalmente, para Institutos de Pesquisa meteorológicos que geralmente estão ligados ao setor público.

5. CONCLUSÕES E EXPECTATIVAS DE TRABALHOS FUTUROS

Os Bancos de Dados Espaciais surgiram da evolução e necessidade de representar dados com características geográficas. Para o tratamento deste tipo particular de informação, foram criados objetos, funções e índices específicos através de extensões de Banco de Dados.

Ao observar estas características nas extensões de Banco de Dados Espaciais percebeu-se que o problema de tratar agrupamentos de dados espaciais (*Spatial Data Clustering*) ainda não havia sido explorado. A partir desta constatação, nesta dissertação buscou-se propor um método de detecção e representação de aglomerados do tipo espacial Ponto. Este é o primeiro passo para criação desta nova funcionalidade dentro dos SGBDE's.

Esta dissertação teve como escopo de trabalho criar as técnicas para a solução deste problema dentro da área da meteorologia através de um método de *clustering*. O estudo foi realizado para descargas atmosféricas como dados espacialmente e temporalmente próximos. O fenômeno de descarga atmosférica pode ser ilustrado como um caso em que estes tipos de dados se encaixam, visto que o *clustering* auxilia em estudos sobre o formato, tamanho e deslocamento de tempestades elétricas, entre outras pesquisas.

Outra motivação para a criação de um método é que estes tipos de dados, fornecidos diariamente por sensores, poderiam ser mais bem aproveitados devido à riqueza de conhecimento que se pode obter dentro do banco de dados. *Clustering* é uma técnica útil na exploração de dados, cujos bons resultados dependem da métrica utilizada para a detecção de similaridade dos dados. Se houver um padrão nos dados detectados por cluster, poderá permitir integrar informações destas relações com outras técnicas mais específicas dos meteorologistas. Além disso, será mais uma forma de apoio a decisão na emissão de alertas na refinaria, podendo, na prática, salvar vidas.

Para chegar ao resultado almejado, estudos sobre extensões espaciais de Gerenciadores Banco de Dados foi realizado. Destas extensões, foram levantados aspectos inerentes ao *Oracle Spatial*, *MySQL Espacial* e *Postgis*. Do perfil extraído

sobre as características e funcionalidades destas extensões foi realizada uma análise comparativa.

Levando em consideração o software livre e os aspectos contemplados pelo Consórcio *Open Gis*, o *Postgresql* e sua extensão *Postgis* foram alvo de experiências para o desenvolvimento do método de representação espacial de *clustering*. O *Postgis* possui versões lançadas muito rapidamente, o que demonstra a preocupação da comunidade no melhoramento contínuo da extensão, de modo que procura oferecer cada vez mais confiança às soluções inspiradas nos padrões do Consórcio *Open Gis*, que por sua vez estão sendo cada vez mais consolidadas na comunidade científica e nas empresas.

Estudos sobre algoritmos de *clustering* abordados dentro da comunidade científica também foram levantados. Dos vários algoritmos presentes na literatura, quatro algoritmos com características distintas foram escolhidos para exemplificar o problema.

Dentre os algoritmos estudados, nenhum deles pôde ser totalmente adaptado para Banco de Dados. Podemos citar alguns pontos inerentes da falta de implementação para Banco de Dados: a) projetos de banco de dados livres e comerciais não se propuseram a implementar tal algoritmo; b) necessidade abrangente de tal função, onde a aplicação para dados meteorológicos do tipo ponto é apenas um problema; c) O cálculo de similaridade de diferentes tipos de dados é uma tarefa difícil, pois é dependente da representação de dados, como cita (BARANAUSKAS, 2003); e c) a comunidade dedicada na implementação de extensão possui preocupações com outros tipos de funcionalidades, tais como os descritos no Padrão *OpenGIS*. (OGIS, 2005).

Características de cada algoritmo serviram como base no desenvolvimento do método. O método busca por limiar de expansão de vizinhança, parâmetro definido *a priori*. Com a medida de distância máxima entre os pontos, são descobertos quantos e quais são os vizinhos para todos os pontos. “A importância da consideração dos relacionamentos de vizinhança, está no fato que para processos espaciais, a medida associada a um local, ou objeto, tende a ser similar às medidas em locais próximos”.(NEVES *et al* 2001). Este fato é conhecido como a *1ª lei da geografia*, especificada por Topler (ZEITOUNE *et al*. 2001).

Pelo método apresentado, o primeiro ponto selecionado é o que possui a maior quantidade de vizinhos, visto que neste primeiro passo, grande parte dos pontos (ou todos) pode ser capturada, diminuindo a busca entre os outros pontos. Por esta característica podemos definir o método desenvolvido como método de heurística de

partição. O ponto com mais vizinhos e seus vizinhos são rotulados como sendo pertencentes ao primeiro cluster. Entre pontos rotulados seus vizinhos também são rotulados e assim sucessivamente até que não aja vizinho em comum. Este método termina quando todos os pontos que estejam relacionados espacialmente com pelo menos um vizinho estejam rotulado.

Como caso de teste foi utilizada aglomeração de descargas atmosféricas em determinados períodos de tempo. A diversidade das atividades elétricas e as mudanças climáticas precisam de ferramentas para um estudo mais aprofundado sobre estas descargas, reorganizando estes dados num banco de dados. O método contribui com este objetivo, mas pode ser adaptado a qualquer outra área onde é necessário o agrupamento de dados espaciais do tipo *Ponto*.

Esta dissertação mostrou alguns algoritmos de *clustering* contemplados na literatura, as vantagens e desvantagens destes e as diferenças e a dificuldades de adaptação a *clustering* de dados espaciais dentro de um SGBDE de Código aberto, como o *PostgreSQL*. Além dos pontos explicitados, esta dissertação apresentou um método para representar estes clusters de forma coerente para o usuário. Esta tarefa de representar clusters em nenhuma área é um processo trivial, e por isso, em nenhum dos SGBDs estudados foi implementado, sendo eles de código aberto, ou não.

O método desenvolvido mostrou que é possível aglomerar dados do tipo espacial *Ponto*, através da tecnologia presente. Podemos destacar os pontos: (i) O método revela não apenas quantos clusters um conjunto de dados possui, mas bem como quais os pontos pertencentes a cada agrupamento, facilitando a identificação do padrão dos mesmos; (ii) O método possui alta similaridade intracluster e baixa similaridade intercluster; (iii) O número de clusters não é definido *a priori*, o que representa clusters coerentes com a medida de similaridade; (iv) O usuário define um limiar de qualidade, através da passagem de um valor de vizinha máxima. Este valor depende das necessidades do usuário; (v) A qualidade do resultado depende da definição de atributos não espaciais, como tempo, bem como o limite de vizinhança entre os pontos; (vi) O método se mostrou compatível em relação a desempenho com algoritmo K-Means através de testes realizados com o Weka, utilizando até 2000 pontos distintos. Em comparação a resultados obtidos, o método proposto demonstrou clusters mais bem formados, isto é, clusters mais representativos sob sua similaridade. Quanto a pontos com muitos vizinhos, isto é, pontos muito próximos e o limite de vizinhança alto, o desempenho do método se demonstrou mais lento computacionalmente, isto é, utilizou

muitos recursos de processador; (vii) O método se mostra flexível para objetos espaciais do tipo polígono, pois a mudança do teste topológico *contains* para o tipo ponto, pode ser facilmente adaptado para o teste topológico *intersects*; (viii) O agrupamento dos objetos tipo ponto facilitam operações de extração do centróide de um cluster, bem como a determinação do envoltório convexo, buffer, cálculo de distância, sobreposição sob outras geometrias, área do cluster, cálculo de dispersão temporal do cluster, entre outros.

Destacamos como trabalhos futuros mais estudos sobre a otimização da consulta para muitos pontos, bem como a otimização do método para *n* colunas de chave primária; automatização do método através da implementação em linguagem Java e a inserção da função num SGBD com extensão espacial.

REFERÊNCIAS BIBLIOGRÁFICAS

AGGARWAL, C. C.; PROCOPIUC C.; WOLF, J., L., YU,S.;PARK, J.S. *Fast Algorithms for Projected Clustering*, In: Proceedings of the ACM SIGMOD Conference on Management of Data, pp. 61-72, Philadelphia, EUA, 1999.

AMARAL, S.;MONTEIRO, A.M.V.;CÂMARA,G.; QUINTANILHA J.A., *Interpoladores Espaciais para Geração de Superfícies de Densidade Populacional na Amazônia Brasileira: problemas e perspectivas* . IV Simpósio Brasileiro de Geoinformática, IV, 2002

ANDEBERG, M.R. *Cluster analysis for applications*. New York: Academic Press, 1973.

ANKERST, M.; BREUNIG, M.; M., KRIEGEL; H. P., *et al.* “OPTICS: Ordering Points to Identify the Clustering Structure”, In: *Proceedings of the ACM SIGMOD Conference on Management of Data*, pp. 49-60, Philadelphia, EUA, 1999.

ANSELIN, L.; BAO, S. *Exploratory spatial data analysis linking SpaceStat and ArcView*. In: FISCHER, M.M.; GETIS, A., Eds., *Recent Developments in Spatial Analysis: Spatial Statistics, Behavioural Modelling, and Computational Intelligence*. Berlin: Springer, 1997, p. 35-59.

ARONOFF, S., *Geographic Information Systems*. WDL Publications, Canadá 1989.

ASSIS, V.C.D. *Uso de Mapas Auto-Organizáveis de Kohonen em Classificação de Padrões*. [2005]. Disponível em
<<http://twiki.im.ufba.br/pub/MAT054/SemestreArtigos20052/ArtigoKohonen.doc>>.

ASSUNÇÃO, R.M.; LAGE, J.P.; AREIS, E.; SILVA, P.L.N. *Análise de conglomerados espaciais via árvore geradora mínima*. Revista Brasileira de Estatística, 2004.

BAILEY, T.C.; GATRELL, A.C. *Interactive spatial data analysis*. Harlow: Longman, 1995.

BARANAUSKAS, J.A. *Clustering*. Notas de aula - Departamento de Física e Matemática – DFM Universidade de São Paulo -USP [2003]. Disponível em <<http://www.fmrp.usp.br/augusto>> . Acesso em Janeiro de 2006.

BENETI,C.A.A.; ALVIM, E. L.; ANDRADE, S. M. G.; ASSUNÇÃO, L. A. R.; CAZETTA, A. F.; REIS, R. J., *RINDAT- Rede Integrada de Detecção de Descargas Atmosféricas no Brasil: situação atual, aplicações e perspectivas*. In: Congresso Brasileiro de Meteorologia, 2000, Rio de Janeiro. Anais Rio de Janeiro: SBMET, 2000. Seminário de Meteorologia por Radar e Física de nuvens. 1 CD-ROM

BENETI,C.A.A.; VASCONCELLOS, C.; CALVETTI, L.; MOREIRA, I, *Efeitos Geográficos e orográficos na incidência de descargas elétricas atmosféricas no Paraná – estudo preliminar*. In: Congresso Brasileiro de Meteorologia, SBMET,2002.

BRANDES,U.; GAERTLER,M.;WAGNER, D. *Lecture Notes in Computer Science: Experiments on Graph Clustering Algorithms*. Ed. Springer Berlin / Heidelberg. Volume 2832 , 2003 pp. 568 – 579.

BOGORNY,V. *Um estudo sobre o OpenGis:Um estudo sobre o OGC para Interoperabilidade e Distribuição de SIG*. Dissertação de Especialização PPGC UFRGS- Universidade Federal do Rio Grande do Sul. Porto Alegre, 1999.

BÜCHNER, R.L. *OBJETOS ESPACIAIS EM BANCO DE DADOS RELACIONAL: uma aplicação prática utilizando POSTGIS*. Monografia de Especialização. DEINFO - UEPG,2002

BURROUGH, P. *Principles of geographical information systems for land resources assessment*. Oxford, Clarendon Press, 1986.

CAMARA, G.; MONTEIRO, A. M. V; DRUCK, S.; CARVALHO, M.S. *Análise Espacial e Geoprocessamento*. In.: S.Druck, M.S. Carvalho, G.Câmara, A.M.V.

Monteiro (eds) *Análise Espacial de Dados Geográficos. EMBRABA/INPE, Rio de Janeiro, R.J, 2002. p 3-29.*

CAMARA, G.; MONTEIRO, A. M. V.; PAIVA, J. A. *et al. Towards a unified framework for spatial data models. J. Braz. Comp. Soc.*, 2000, vol.7, no.1, p.17-25.

CARLANTONIO, L.M.di *Novas Metodologias para Clusterização de dados*. Tese de Mestrado. Programa de Engenharia Civil. COPPE/UFRJ, Rio de Janeiro, 2001.

CARVALHO, M. S.. *Aplicação de métodos de análise espacial na caracterização de áreas de risco à saúde*. Tese de Doutorado - Universidade Federal do Rio de Janeiro, COPPE. Rio de Janeiro, 1997

CHU, J. *A Secure Socket Layer Proxy for a Kerberos Authenticated Intranet*. Computer Science 490a . <<http://zoo.cs.yale.edu/classes/cs490/98-99a/chu.john.avatar>> 1999. Acesso em Novembro, 2005.

CLUSTER ANALYSIS: *EM (Expectation Maximization) Clustering* [2004]. Disponível em <<http://www.statsoft.com/textbook/stcluan.html>>. Acesso em Abril 2006.

CORTEZ, S.S., *Tradução de Consultas Visuais para Cláusulas SQL*. Recife: UFPE, 2002.

COSTA, C.M. *A tecnologia de Bancos de Dados aplicada aos Sistemas de Informações Geográficas*. In: Anais do III Colóquio Brasileiro de Ciências Geodésicas, Curitiba, 2003.

COWEN D.J. *SIG versus CAD versus DBMS: what are the differences?*, In "Introductory readings in Geographic Information Systems". Londres: Taylor and Francis, 1988.

DATE, C. J. *Introdução a sistemas de bancos de dados*. Rio de Janeiro: Campus, 2000.

DIAS, C.R.; OCHI, L.S.”Desenvolvimento e Análise Experimental de Algoritmos Evolutivos para o Problema de Clusterização Automática em Grafos Orientados” In: VIII Simpósio Brasileiro de Redes Neurais - São Luis, MA,2004.

DIDAY, E. ;SIMON, J. C., Clustering analysis, *Communication and Cybernetics* 10, Springer-Verlag, Berlin, Heidelberg, Nova York, 1976

DOUGLAS, K.; DOUGLAS,S. *PostgreSQL: a comprehensive guide to building, programming, and administering PostgreSQL databases*. Indianapolis, Ind.: Ed. Sams. 2003

DPI-INPE. *Estimador de Núcleo*. Disponível em <<http://www.dpi.inpe.br/spring/usuario/kernel.htm>>. Acesso em de Junho de 2005.

DPI-INPE. *Características SIG* http://www.dpi.inpe.br/spring/usuario/geoproc.htm#carac_sig). Acesso em Janeiro 2006.

ECOCONSULT, 2005. *Sistemas de Licenciamento de Software Aberto: Análise e justificativa*. Disponível em: <<http://www.ecosconsult.com.br/license/analise.htm>>. Acesso em Maio 2005.

ESRI – *Environmental Systems Research Institute, Inc. Using ArcView GIS*. Redlands, 1996.

FREE BSD. *Free BSD Ports Collection* Disponível em <<http://www.FreeBSD.org/ports>>. Acesso em Fevereiro 2006

FERREIRA, K. ;QUEIROZ, G.; PAIVA, J.; CARTAXO, R.; CÂMARA,G. *Arquitetura de Software Para Construção de Bancos de Dados Geográficos Com SGBDObjeto-Relacionais* XVIII Simpósio Brasileiro de Bancos de Dados, Gramado, 2002.

Disponível em:<http://www.dpi.inpe.br/gilberto/papers/paper_sb2002.pdf>. Acesso em Janeiro 2005.

GAMA, J. *Métodos Descritivos: Análise de Agrupamentos*. Disponível em <www.liacc.up.pt/~jgama/Bdc/aglo.pdf>. Acesso em Fevereiro de 2006.

GARAFFA, I. M. *Análise da Adequação de uma hierarquia de Classes Básicas para Modelagem Conceitual de SIG através de um Estudo de Caso*. Porto Alegre: CPGCC da UFRGS, 1998. Dissertação de Mestrado. (anexos 1, 2, 3, 4)

GEOLIVRE. *GeoLivre LINUX 5.0: Notícias* –21 Janeiro 2006. Editoria: GNU/Linux. Disponível na internet em <http://www.geolivres.org.br/modules/news/>. Acesso em Janeiro 2006.

GEOS. Disponível em <<http://geos.refractive.net/>>. Acesso em Janeiro de 2006.

GIST. *The GiST Indexing Project*. Disponível em <<http://gist.cs.berkeley.edu/>>. Acesso em Janeiro 2006.

GNU .*O que é o Software Livre?* Free Software Foundation, Inc, Boston, , USA Disponível em: <<http://www.gnu.org/philosophy/free-sw.pt.html>>. Acesso em Janeiro de-2006.

GRASS. *Geographic Resource Analysis Support System*. Disponível em <www.grass.itc.it>. Acesso em Fevereiro-2006.

GUTTMAN,A. *R-trees: a dynamic index structure for spatial searching*. In ACM SIGMOD International Conference on Management of Data: Ed. B. Yomark (Boston: ACM Press), p. 47-54, 1984.

HAN, J.; KAMBER, M., *Cluster Analysis*. In: *Data Mining: Concepts and Techniques*, 1 ed., cap. 8: Morgan Kaufmann Publishers New York, EUA, 2001.

HELLERSTEIN, J. M.; NAUGHTON, J. F.; PFEFFER, A. *Generalized search trees for databases systems*. In: international Conference in VLDB, Zurich, Suíça. Proceeding: Morgan Kaufman, San Francisco, 1995, p. 562-573

HEYER, L. J.; KRUGLYAK, S.; YOOSEPH, S. *Exploring expression data: identification and analysis of coexpressed genes*. Genome Research, 1999.

HRUSCHKA, E. R.; EBECKEN, N. F. F. *A Genetic algorithm for cluster analysis*.: *IEEE Transactions on Evolutionary Computation*, 2001.

INPE, *Instituto Nacional de Pesquisas Espaciais*. Disponível em <www.inpe.br>. Acesso em Junho 2005.

ISO/IEC JTC 1/SC 32 *Data Management and Interchange SQL Multimedia and Application Packages*. Parte 3 No.0229, 1999.

JAIN, A. K.; MURITY, M. N.; FLYIN, P. J. *Data Clustering: A Review*. ACM Computing surveys. Vol.31, No 3, Setembro, 1999

JAIN A. K.; DUBES, R. C. *Algorithms for Clustering Data* Prentice-Hall advanced reference series: Prentice-Hall, Inc., Upper Saddle River. 1988.

KAESTNER, C. A. A. *Data Mining: Conceitos e Técnicas*. Disponível em <www.ppgia.pucpr.br/~kaestner/ls-dm/DM-MBA-SI-3.ppt>. Acesso em Janeiro 2006.

KASKI, S. *Computationally efficient approximation of a probabilistic model for document representation in the WEBSOM full-text analysis method*. Neural Processing Letters. (vol. 5, pp. 139-151).

KOHONEN, T. *The Self-Organizing Map*. Springer Series in Information Sciences. 3a Ed, 2001.

LASSEN, A. R. O. J.; OSTERBYE, K. *Object Relational Modeling, Centre for Object Technology (COT)*, 1998.

LIMA P.; CAMARA, G.; QUEIROZ G. *GeoBR: Intercâmbio Sintático e Semântico de Dados Espaciais*. IV Simpósio Brasileiro de Geoinformática, Caxambu, 2002.

LOZANO, F. *MySQL x PostgreSQL Quando Usar Cada Um*. III Encontro do Grupo de Usuários Linux de SC. Florianópolis, 2002.

MYSQL. Manual de Referência do MySQL. Disponível na internet em <<http://dev.mysql.com/doc/refman/5.0/pt/index.html>>. Acesso Janeiro 2006.

MARTINS, V. *Uma visão Geral sobre ODBC*. Revista Bate Byte. Edição 53, 1996. Disponível em <<http://www.pr.gov.br/batebyte/edicoes/1996/bb53/odbc.htm>>. Acesso em Dezembro 2004.

MAPSERVER. UMN Mapserver.<www.mapserver.gis.umn.edu>. Acesso em Fevereiro-2005.

MELO, C.H., GUERRA, M.A.M. *SGBD com Extensão Espacial e Sistemas de Geoinformação: Um Casamento Perfeito*. Revista Fonte. Vol.2 . Junho 2005.

MURRAY, C., 2003, *Oracle® Spatial User's Guide and Reference 10g*: Release 1 (10.1), Redwood City, Oracle Corporation, p. 602.

NASCIMENTO, E. *Previsão de Tempestades Severas utilizando-se parâmetros convectivos e Modelos de Mesoescala: Uma estratégia operacional adotável no Brasil*. Revista Brasileira de Meteorologia, Lemma –UFPR. 2005.

NETER, J.; WASSERMAN, W.; KUTNER, M.H. *Applied Linear Regression Models*. 2a.ed. Homewood: Irwin, 1989.

NEVES, M.C.; FREITAS, C.C; CÂMARA, G. *Mineração de Dados em Grandes Bancos de Dados Geográficos*. Relatório Técnico Programa de Ciência e Tecnologia para Gestão de Ecossistemas –INPE. Novembro, 2001.

NG, R. T.; HAN, J. "Efficient and Effective Clustering Methods for Spatial Data Mining, In: *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, Santiago, Chile, 1994. p.144-155.

NOSAKI, S.; PINTO Jr. O.; PINTO, I.R.C.A; FERRAZ, E.C. *Estudo dos Efeitos das Descargas Atmosféricas no Sistema de Distribuição da Elektro*. Resumo projeto de P&D. 2001.

OGIS- Open Geospatial Consortium, *OpenGIS Simple Features Specification for SQL*, Revision 1.1., 1999. Disponível em <<http://www.opengis.org/techno/specs/99-049.pdf>>. Acesso em Fevereiro 2005.

OPENSHAW, S.; WYMER, C. *Classifying and regionalizing census data*. In: OPENSHAW, S., Ed., *Census users' handbook*. Cambridge: GeoInformation International, 1995, p. 460.

ORACLE SPATIAL. Disponível em <http://www.oracle.com/technology/products/spatial/spatial_doc_index.html>. Acesso em Janeiro 2006.

ORACLE SPATIAL OPTION *Location-Based Services for Oracle9i* Disponível em <http://www.oracle.com/technology/products/spatial/htdocs/data_sheet_9i/9iR2_spatial_ds.html>. Acesso em Fevereiro, 2005.

QUEIROZ, G. R. *Algoritmos geométricos para bancos de dados geográficos: da teoria à prática no Terralib*. –São José dos Campos: INPE, 2003.

QUEIROZ, G. R.; FERREIRA, K. R. *Banco de Dados Geográficos*.-SGBD com extensões espaciais. Cap. VIII. Ed. MundoGeo. Curitiba, 2005.

QGIS. *Quantum GIS* – disponível em <<http://qgis.sourceforge.net/>>. Acesso em Fevereiro-2005.

PAGE, C., *Federating Datasets with Spatial Joins*. Astronomical Data Analysis Software and Systems. XIII ASP Conference Series, Vol. 314. 2004.

PINHEIRO, L.C; VAZ, M.S.M.G; MARTINHAGO, A.Z. *Proposta de uma Extensão do Padrão FGDC/CSGDM para Dados de Radar Meteorológico* Revista Publicatio UEPG - Ciências Exatas e da Terra, Ciências Agrárias e Engenharias, ano 13, número 3. Dezembro, 2005.

PINTO Jr, O.; PINTO, I.R.C.A., *Tempestades e relâmpagos no Brasil*, Ed. INPE, São José dos Campos, 2000, p.200.

POLITI, J.; Stephany, S.; DOMINGUES, M.O. ; MENDES JR., O. *Uma metodologia para representação Espaço-Temporal de ocorrências de descargas elétrica* Nuvem-solo. Programa de Pós Graduação em Computação aplicada CAP/NPE, 2004.

POSTGRESQL. *PostgreSQL: The world's most open source database*. Disponível em <<http://www.postgresql.org/>> Acesso em Fevereiro 2005.

PROJ4. Disponível em <<http://www.remotesensing.org/proj>> Acesso em Janeiro, 2006.

RAFAELI NETO, S.L. *Um Modelo Conceitual de Sistema de Apoio à Decisão Espacial para Gestão de Desastres por Inundações* Tese de Doutorado. Escola Politécnica da Universidade de São Paulo / Departamento de Engenharia. São Paulo. 2000.

RAMSEY, P. *PostGIS Manual*. em *PostGis Refrations*. Disponível em <<http://postgis.refrations.net/>>. Acesso em Março 2005.

RAMSEY, P. *PostGis Introduction and Evaluation*. Refrations Research. Disponível na Internet em <<http://dl.maptools.org/dl/omsug/osgis2004/postgis-report-2004-06.doc>>, 2004.

RAYMOND T. N.; HAN, J. *Efficient and Effective Clustering Methods for Spatial Data Mining* 20th International Conference on Very Large Data Bases, Santiago, Chile, 1994.

RODRIGUES, A.M. *Técnicas de Datamining Classificadas do Ponto de Vista do Usuário*. Tese de Doutorado Universidade Federal do Rio de Janeiro, COPPE.Rio de Janeiro, 2000.

SAMET, H. *Application of spatial data structures: computer graphics, image processing, and GIS*. MA: Addison-Wesley, 1990.

SILICON GENETICS. *Gene Spring – QT(Quality Threshold) Clustering*. Guia de Análises [2003]. Disponível na internet em <http://www.silicongenetics.com/Support/GeneSpring/GSnotes/analysis_guides/qt_clustering.pdf>. Acesso em Fevereiro, 2006.

SCHNEIDER, M. *Spatial data types for database systems*. Berlin Hidelberg:Springer-Verlag, 1997.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Sistemas de bancos de dados*. São Paulo:Makron Books, 1999.

SILVA, N.C. *Implementação de Redes "Parallel-Self Organizing Map" para Classificar Imagens de Radar*. Dissertação de Mestrado. Departamento de Ciência da Computação-Universidade De Brasília. Julho 1999.

SILVA, R. *Banco de Dados Geográficos: uma análise das arquiteturas DUAL(Spring) e Integrada (Oracle Spatial)*. Dissertação de Mestrado. Universidade de São Paulo – Área de Transportes. São Paulo, 2002.

SILVERMAN,B.W. *Density Estimation for Statistics and Data Analysis* (Monographs o Statitistic and Applied Probability 26):Chapman and Hall, Londres,1990.

SIMEPAR. *Simepar- Instituto Tecnológico do Paraná*. Disponível em <www.simepar.br> Acesso em Janeiro-2006.

SOUTO, M.C.P. de, *Uso de Técnicas Inteligentes na Análise de Dados Biológicos [2004]*. Disponível em

<www.dimap.ufrn.br/~marcilio/BIOINFORMATICA/BIO2004.1/BIO-aula-12-k-medias+SOM.ppt>. Acesso em Dezembro 2005.

STONEBRAKER, M.; ROWE, L. A.; HIROHAMA, M. *The implementation of Postgres*. IEEE Transactions on Knowledge and Data Engineering, v. 2, n. 1, p. 125-142, mar. 1990.

TERRA VIEW. *INPE- Terra View* Disponível na internet em <<http://www.dpi.inpe.br/terraview/index.php>>. Acesso em Janeiro 2005.

THUBAN. *Thuban an Interactive Geographic Data Viewer*. Disponível em <www.thuban.intevation.org>. Acesso em Fevereiro 2006.

UNB.”*Coordenadas UTM*” Disponível em <http://www.unb.br/ig/glossario/verbete/coordenadas_utm.htm> Acesso em Janeiro 2006.

VARELA,M.L.R. *Clustering de Documentos*. Relatório de Pesquisa. Universidade de Évora Departamento de Informática. Monte de Caparia, Portugal [2000] Disponível na internet em <http://www.di.uevora.pt/~pq/miaa/cluster_miaa.html>. Acesso em Fevereiro-2006

VAZ, M. S. M. G.; ZAMBIANCO JUNIOR, R.A.; ANTONELLI, R. A. METAMAPA – *Armazenamento e Indexação de Mapas Digitais via Metadados*. Revista Brasileira de Computação. Vol.2 No. 2, 2004.

VASCONCELLOS, C.A.; CUROTTO,C.L.; BENETI,C.A.A; SATO, F.; PINHEIRO, L.C. *Electrical thunderstorm nowcasting using lightning data mining*. 19th International Lightning Detection Conference (ILDC).Tucson, Arizona, EUA,2006.

WEKA. *Weka machine learning software*. University of Waikato Disponível em <<http://www.cs.waikato.ac.nz/~ml/>>.Acesso em Outubro 2005.

WIKIPEDIA 2005. *The free encyclopedia*. <<http://www.pt.wikipedia.org/wiki/>> Acesso em Maio 2005

WITTEN,I.H; FRANK,E. *Data Mining:practical machine learning tools and techniques with Java implementations*. Morgan Kauffmann Publishers. São Francisco,CA. 2000.

YAGER,R.; REESE G.; KING,T. *My SQL & mSQL*. Ed.: O'Reilly 1999

YATES,R.B; BERTHIER,A.R.N.; LONGMAN, W. *Modern Information Retrieval.*, Prentice-Hall.Maio 1999.

APÊNDICE 1

Experimento com 30 pontos -vizinhos

"2003-07-06 14:00:24 ",677244300,-55,-23.4344,-56.8424
"2003-07-06 14:00:28 ",881429800,-19,-24.9972,-54.3578
"2003-07-06 14:00:34 ",801693900,-22,-25.0663,-53.3884
"2003-07-06 14:00:38 ",766472100,-81,-25.0402,-53.2613
"2003-07-06 14:01:03 ",466717500,-30,-26.4175,-53.778
"2003-07-06 14:01:26 ",156116500,-30,-23.9135,-57
"2003-07-06 14:01:28 ",400846400,-26,-23.3995,-56.9172
"2003-07-06 14:01:29 ",958263700,-24,-25 9,-53.3312
"2003-07-06 14:01:38 ",201808935,-30,-23.2361,-56.8251
"2003-07-06 14:02:00 ",50218400,24,-25.4218,-53.3314
"2003-07-06 14:02:10 ",507497600,20,-24.9138,-538
"2003-07-06 14:02:23 ",328794900,-21,-26.3982,-53.7307
"2003-07-06 14:02:43 ",787866900,33,-23.9662,-56.1822
"2003-07-06 14:02:48 ",791638300,-42,-25.4131,-53.1797
"2003-07-06 14:02:49 ",197913200,-35,-25.4205,-53.2072
"2003-07-06 14:02:51 ",543745600,-22,-23.3649,-56.8169
"2003-07-06 14:02:58 ",845758800,-30,-23.9073,-57.0804
"2003-07-06 14:03:32 ",552393800,-28,-25.02,-53.3742
"2003-07-06 14:03:33 ",934648871,-29,-25.2518,-53.0212

APÊNDICE 2

Experimento 1: Dados de Entrada

----arquivo arff

@relation raio

@attribute raidatahora date "yyyy-MM-dd HH:mm:ss"

@attribute rainanosegundo real

@attribute raipicocorrente real

@attribute railatitude real

@attribute railongitude real

@data

"2006-01-01 00:21:16",477608000,-19,-25.4702,-51.7742

"2006-01-01 00:24:15",864950000,25,-25.0477,-48.8558

"2006-01-01 00:24:20",869615700,-30,-25.6903,-51.4354

"2006-01-01 06:04:24",954388800,77,-20.3626,-60.2984

"2006-01-01 09:52:18",60966800,-98,-31.5866,-47.1639

"2006-01-01 19:16:07",26797600,37,-22.6132,-54.689

"2006-01-01 21:51:52",792702500,43,-20.9283,-53.8211

"2006-01-01 22:09:03",132731300,134,-22.5853,-62.1743

"2006-01-02 00:20:22",58156100,-46,-25.4341,-54.2635

"2006-01-02 01:36:52",63524500,-80,-21.5085,-53.9542

"2006-01-02 08:38:41",73071300,22,-25.3014,-51.4485

"2006-01-02 15:50:26.13",379930000,-37,-26.534,-50.9012

"2006-01-02 16:43:11",20519250,38,-28.0145,-50.4688

"2006-01-02 18:00:51",65319949,-25,-24.9128,-48.0796

"2006-01-02 19:40:39",468282025,68,-20.2224,-59.371

"2006-01-02 21:56:39",695128700,-15,-24.6124,-48.9856

"2006-01-02 22:59:09",88426700,13,-25.8329,-48.6287

"2006-01-02 23:07:34",774710000,-20,-24.699,-49.9301

"2006-01-02 23:07:55",3094700,0,-25.6978,-48.5825

"2006-01-02 23:08:30",87228800,8,-25.7434,-48.5562

"2006-01-02 23:10:17",98581100,-37,-25.749,-48.5788
"2006-01-02 23:13:22",19838700,-9,-25.7544,-48.5576
"2006-01-02 23:22:22",825086900,19,-25.788,-48.5517
"2006-01-02 23:33:17",38205000,-18,-25.5455,-48.6431
"2006-01-02 23:53:54",55035900,-21,-24.435,-49.6404
"2006-01-02 23:53:54",80507300,-15,-24.436,-49.6532
"2006-01-02 23:56:00",96844900,-17,-24.5086,-50.366
"2006-01-03 00:34:09",12102700,-16,-25.0844,-48.298
"2006-01-03 02:16:33.16",81563946,40,-23.7113,-47.4614
"2006-01-03 11:35:55",564245450,-49,-27.0446,-43.6646
"2006-01-03 19:07:51",30209741,-9,-22.8048,-47.6155
"2006-01-03 20:21:14",2729400,-64,-21.9675,-48.4197
"2006-01-04 00:20:23",23146400,126,-20.0522,-58.0106
"2006-01-04 00:40:33",35475200,-150,-21.4869,-50.6656
"2006-01-04 01:02:25",581371000,82,-20.8268,-57.0363
"2006-01-04 01:23:22",816041600,131,-20.2923,-57.5349
"2006-01-04 16:55:44",36314800,20,-20.4478,-51.8341
"2006-01-04 17:26:40",34050250,-80,-28.9706,-37.9037
"2006-01-04 18:36:27",378870300,-29,-22.5228,-41.9428
"2006-01-04 20:01:41",86924000,11,-22.4669,-50.9256
"2006-01-04 20:29:34",53839200,-42,-16.5523,-49.0193
"2006-01-04 21:43:06",97358500,9,-21.9285,-51.6656
"2006-01-04 22:24:54",830592300,96,-21.3257,-58.8807
"2006-01-04 22:59:10",87663947,-14,-24.6045,-50.2053
"2006-01-05 00:22:47",58664800,-37,-22.499,-48.6408
"2006-01-05 00:36:41",48343000,-13,-24.0516,-50.2041
"2006-01-05 01:16:33",510410000,15,-24.9095,-50.9812
"2006-01-05 01:39:24",81543300,0,-24.1279,-50.5848
"2006-01-05 01:51:34",31180000,-15,-24.088,-50.5931
"2006-01-05 01:53:18",404649000,-21,-24.6356,-50.7242
"2006-01-05 01:57:09",11728369,-20,-24.6334,-50.7519
"2006-01-05 03:21:55",5000600,-69,-24.2763,-51.2725
"2006-01-05 03:25:10",5364429,-15,-24.4195,-51.8
"2006-01-05 03:49:09",45982700,13,-24.4568,-51.2878

"2006-01-05 04:09:48",71805400,-42,-24.3182,-51.4486
"2006-01-05 06:02:03",13767000,-21,-24.1595,-52.1589
"2006-01-05 06:48:10",62552600,-42,-24.1407,-52.5378
"2006-01-05 18:48:07",84558100,-33,-25.3155,-52.989
"2006-01-05 19:07:46",96484300,-30,-21.4716,-53.8731
"2006-01-05 19:34:47",28318900,-28,-24.5836,-50.7864
"2006-01-05 19:34:47",172130150,-22,-24.5939,-50.77
"2006-01-05 20:34:53",438900,-22,-24.5301,-50.9883
"2006-01-05 20:34:53",98438900,-21,-24.5304,-50.9889
"2006-01-05 20:53:41",215207600,23,-25.0321,-49.2703
"2006-01-05 21:00:11",294600,-22,-27.1328,-50.6817
"2006-01-05 21:06:16",65891300,-32,-21.1079,-56.576
"2006-01-05 21:29:47",82349900,-10,-25.9109,-49.369
"2006-01-05 23:55:13",798455400,23,-24.2106,-51.7313
"2006-01-06 22:58:33",8072400,-73,-27.5291,-49.601
"2006-01-07 08:15:17",990222500,34,-205,-51.6983
"2006-01-07 16:54:45",52275900,16,-22.5668,-50.886
"2006-01-07 17:27:14",417601497,-20,-19.5962,-47.9202
"2006-01-07 17:41:26",733489200,61,-29.3963,-51.7532
"2006-01-07 23:01:15",20560800,0,-19.843,-54.9261
"2006-01-08 01:59:25",16818400,-64,-17.4251,-54.9985
"2006-01-08 07:58:06",273001100,103,-34.6642,-60.0764
"2006-01-08 17:15:30",949637400,11,-21.5582,-52.1159
"2006-01-08 18:14:12",381452200,16,-21.9906,-50.5597
"2006-01-08 18:16:48",333219900,8,-22.5436,-514
"2006-01-08 18:30:07",72372100,12,-22.5603,-50.9397
"2006-01-08 18:45:01",985429800,14,-22.4188,-50.4834
"2006-01-08 19:01:17",166830000,22,-22.4113,-50.4693
"2006-01-08 19:05:09",41286800,10,-22.5853,-50.9857
"2006-01-08 19:06:24",921038800,15,-22.507,-50.6701
"2006-01-09 16:44:40",25699028,-24,-24.1974,-51.1271
"2006-01-09 17:14:28",214772767,147,-18.9381,-60.6893
"2006-01-09 17:37:01",82363830,-35,-24.367,-51.2862
"2006-01-09 18:10:11",971940800,0,-24.7662,-50.5052

"2006-01-09 18:16:01",39164285,-38,-24.739,-50.6741
"2006-01-09 18:21:31",36010800,61,-23.6705,-46.8972
"2006-01-09 18:27:34",113388800,19,-23.9501,-51.6751
"2006-01-19 01:43:39",218911400,78,-23.2278,-49.8798
"2006-03-04 22:38:37",178263002,-33,-19.8891,-41.6578
"2006-03-05 02:17:13",47013448,43,-18.4362,-54.1202
"2006-03-05 15:00:59",28162746,-17,-18.8763,-47.5764
"2006-03-05 16:33:29",767159900,-18,-19.3519,-45.386
"2006-03-05 23:04:48",229702788,-19,-18.3824,-46.9311
"2006-03-06 14:06:41",98451532,0,-22.2449,-46.0421
"2006-03-06 16:53:31",605363992,-63,-14.7552,-46.8234
"2006-03-07 16:55:39",768533500,-23,-18.9988,-49.2365

APÊNDICE 3

Experimento nº 1 pelo Método de Representação de Clustering: Dados de Entrada e Clustering associado

1. DATA: formato yyyy-mm-dd hh24:mi:ss
2. NANOSEGUNDO
3. PONTO: AsText() na coordenada geográfica
4. CLUSTER A QUE PERTENCE

1	2	3	4
2003-07-06 14:00:24	677244300	POINT(-56.8424 -23.4344)	2
2003-07-06 14:00:28	881429800	POINT(-54.3578 -24.9972)	
2003-07-06 14:00:34	686291200	POINT(-53.2624 -25.4211)	1
2003-07-06 14:00:34	801693900	POINT(-53.3884 -25.0663)	4
2003-07-06 14:00:38	766472100	POINT(-53.2613 -25.0402)	
2003-07-06 14:01:03	393227400	POINT(-53.7766 -26.4043)	3
2003-07-06 14:01:03	466717500	POINT(-53.778 -26.4175)	3
2003-07-06 14:01:26	156116500	POINT(-57 -23.9135)	5
2003-07-06 14:01:28	224791400	POINT(-56.9007 -23.3977)	2
2003-07-06 14:01:28	400846400	POINT(-56.9172 -23.3995)	2
2003-07-06 14:01:29	958263700	POINT(-53.3312 -25 9)	
2003-07-06 14:01:38	201808935	POINT(-56.8251 -23.2361)	
2003-07-06 14:02:00	50218400	POINT(-53.3314 -25.4218)	1
2003-07-06 14:02:10	507497600	POINT(-538 -24.9138)	
2003-07-06 14:02:23	74632600	POINT(-53.8606 -26.5946)	
2003-07-06 14:02:23	328794900	POINT(-53.7307 -26.3982)	3
2003-07-06 14:02:43	787866900	POINT(-56.1822 -23.9662)	
2003-07-06 14:02:48	637020500	POINT(-53.2008 -25.3935)	1
2003-07-06 14:02:48	701886213	POINT(-53.1826 -25.3955)	1
2003-07-06 14:02:48	791638300	POINT(-53.1797 -25.4131)	1
2003-07-06 14:02:49	81499100	POINT(-53.206 -25.4191)	1
2003-07-06 14:02:49	197913200	POINT(-53.2072 -25.4205)	1
2003-07-06 14:02:51	192550200	POINT(-56.8162 -23.3683)	2
2003-07-06 14:02:51	543745600	POINT(-56.8169 -23.3649)	2
2003-07-06 14:02:58	845758800	POINT(-57.0804 -23.9073)	5
2003-07-06 14:03:32	477928100	POINT(-57.2203 -23.4653)	
2003-07-06 14:03:32	552393800	POINT(-53.3742 -25.02)	4
2003-07-06 14:03:33	786226872	POINT(-53.0286 -25.2467)	6
2003-07-06 14:03:33	934648871	POINT(-53.0212 -25.2518)	6
2003-07-06 14:03:42	157879600	POINT(-53.2666 -25.2942)	

APÊNDICE 4

Experimento nº 1 pelo Método de Representação de Clustering: Passos realizados

```
---Cria tabela de vizinhos
create table tmpvizinhosa as select distinct(b.raidatahora||'
'||b.rainanosegundo) as datavizinho,a.raidatahora as dataponto
,a.rainanosegundo
as nanoponto,b.the_centrogeom
from tmptestea as a,
(select transform(buffer(transform(the_centrogeom,29192),8000),4618) as
buf,raidatahora,rainanosegundo,the_Centrogeom from tmptestea) as b
where buf && a.the_centrogeom
and contains(buf,a.the_centrogeom)
and (b.raidatahora||' '||b.rainanosegundo) <> (a.raidatahora||'
'||a.rainanosegundo);

---cria uma coluna para indicar o cluster no qual o ponto pertence
alter table tmpvizinhosa add column cluster integer;
update tmpvizinhosa set cluster=0;

repita a
passo 1: seta o que tem mais vizinhos como cluster n+1
update tmpvizinhosa set cluster=1 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

passo2: seta os vizinhos deste ponto como cluster n+1 também
update tmpvizinhosa set cluster=1 where dataponto||' '||nanoponto in
(select datavizinho from tmpvizinhosa where cluster=1) and cluster=0;

passo3: repita até return=0 seta cluster n+1 para os pontos que estão
sem cluster e aparecem como vizinho em algum ponto já marcado com este
cluster n+1
update tmpvizinhosa set cluster=1 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa
where cluster=1)) ;
fim
cluster:= cluster n+1;
fim;
--
update tmpvizinhosa set cluster=2 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

update tmpvizinhosa set cluster=2 where dataponto||' '||nanoponto in
```

```

(select datavizinho from tmpvizinhosa where cluster=2) and cluster=0;

update tmpvizinhosa set cluster=2 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa
where cluster=2)) ;

--
update tmpvizinhosa set cluster=3 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

update tmpvizinhosa set cluster=3 where dataponto||' '||nanoponto in
(select datavizinho from tmpvizinhosa where cluster=3) and cluster=0;

update tmpvizinhosa set cluster=3 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa
where cluster=3)) ;

--
update tmpvizinhosa set cluster=4 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

update tmpvizinhosa set cluster=4 where dataponto||' '||nanoponto in
(select datavizinho from tmpvizinhosa where cluster=4) and cluster=0;

update tmpvizinhosa set cluster=4 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa
where cluster=4)) ;

---
update tmpvizinhosa set cluster=5 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

update tmpvizinhosa set cluster=5 where dataponto||' '||nanoponto in
(select datavizinho from tmpvizinhosa where cluster=5) and cluster=0;

update tmpvizinhosa set cluster=5 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa
where cluster=5)) ;

---
update tmpvizinhosa set cluster=6 where dataponto||' '||nanoponto in
(select dataponto||' '||nanoponto from (select
count(datavizinho),dataponto,nanoponto from tmpvizinhosa where
cluster=0 group by dataponto,nanoponto order by 1 desc limit 1) as t);

update tmpvizinhosa set cluster=6 where dataponto||' '||nanoponto in
(select datavizinho from tmpvizinhosa where cluster=6) and cluster=0;

```

```
update tmpvizinhosa set cluster=6 where dataponto||' '||nanoponto in  
(select dataponto||' '||nanoponto from tmpvizinhosa where cluster=0  
and dataponto||' '||nanoponto in (select datavizinho from tmpvizinhosa  
where cluster=6)) ;
```

APÊNDICE 5

Resultado do Método de *Clustering* Espacial para 32 quilômetros

cluster	data hora	nanosegundo	ponto
2	06-07-2003 14:00:24	677244300	POINT(-56.8424 -23.4344)
	06-07-2003 14:00:28	881429800	POINT(-54.3578 -24.9972)
1	06-07-2003 14:00:34	686291200	POINT(-53.2624 -25.4211)
1	06-07-2003 14:00:34	801693900	POINT(-53.3884 -25.0663)
1	06-07-2003 14:00:38	766472100	POINT(-53.2613 -25.0402)
3	06-07-2003 14:01:03	393227400	POINT(-53.7766 -26.4043)
3	06-07-2003 14:01:03	466717500	POINT(-53.778 -26.4175)
4	06-07-2003 14:01:26	156116500	POINT(-57.098 -23.9135)
2	06-07-2003 14:01:28	224791400	POINT(-56.9007 -23.3977)
2	06-07-2003 14:01:28	400846400	POINT(-56.9172 -23.3995)
	06-07-2003 14:01:29	958263700	POINT(-53.3312 -25.9759)
2	06-07-2003 14:01:38	201808935	POINT(-56.8251 -23.2361)
1	06-07-2003 14:02:00	50218400	POINT(-53.3314 -25.4218)
1	06-07-2003 14:02:10	507497600	POINT(-53.2958 -24.9138)
3	06-07-2003 14:02:23	74632600	POINT(-53.8606 -26.5946)
3	06-07-2003 14:02:23	328794900	POINT(-53.7307 -26.3982)
	06-07-2003 14:02:43	787866900	POINT(-56.1822 -23.9662)
1	06-07-2003 14:02:48	637020500	POINT(-53.2008 -25.3935)
1	06-07-2003 14:02:48	701886213	POINT(-53.1826 -25.3955)
1	06-07-2003 14:02:48	791638300	POINT(-53.1797 -25.4131)
1	06-07-2003 14:02:49	81499100	POINT(-53.206 -25.4191)
1	06-07-2003 14:02:49	197913200	POINT(-53.2072 -25.4205)
2	06-07-2003 14:02:51	192550200	POINT(-56.8162 -23.3683)
2	06-07-2003 14:02:51	543745600	POINT(-56.8169 -23.3649)
4	06-07-2003 14:02:58	845758800	POINT(-57.0804 -23.9073)
	06-07-2003 14:03:32	477928100	POINT(-57.2203 -23.4653)

1	06-07-2003 14:03:32	552393800	POINT(-53.3742 -25.02)
1	06-07-2003 14:03:33	786226872	POINT(-53.0286 -25.2467)
1	06-07-2003 14:03:33	934648871	POINT(-53.0212 -25.2518)
1	06-07-2003 14:03:42	157879600	POINT(-53.2666 -25.2942)

APÊNDICE 6

Experimento 1: Resultado da utilização do Weka :

K-Means

cluster	data hora	nanosegundo	ponto
2	06-07-2003 14:00:24	677244300	POINT(-56.8424 -23.4344)
3	06-07-2003 14:00:28	881429800	POINT(-54.3578 -24.9972)
3	06-07-2003 14:00:34	686291200	POINT(-53.2624 -25.4211)
3	06-07-2003 14:00:34	801693900	POINT(-53.3884 -25.0663)
3	06-07-2003 14:00:38	766472100	POINT(-53.2613 -25.0402)
0	06-07-2003 14:01:03	393227400	POINT(-53.7766 -26.4043)
0	06-07-2003 14:01:03	466717500	POINT(-53.778 -26.4175)
1	06-07-2003 14:01:26	156116500	POINT(-57.098 -23.9135)
1	06-07-2003 14:01:28	224791400	POINT(-56.9007 -23.3977)
2	06-07-2003 14:01:28	400846400	POINT(-56.9172 -23.3995)
3	06-07-2003 14:01:29	958263700	POINT(-53.3312 -25.9759)
1	06-07-2003 14:01:38	201808935	POINT(-56.8251 -23.2361)
0	06-07-2003 14:02:00	50218400	POINT(-53.3314 -25.4218)
0	06-07-2003 14:02:10	507497600	POINT(-53.2958 -24.9138)
0	06-07-2003 14:02:23	74632600	POINT(-53.8606 -26.5946)
0	06-07-2003 14:02:23	328794900	POINT(-53.7307 -26.3982)
2	06-07-2003 14:02:43	787866900	POINT(-56.1822 -23.9662)
3	06-07-2003 14:02:48	637020500	POINT(-53.2008 -25.3935)
3	06-07-2003 14:02:48	701886213	POINT(-53.1826 -25.3955)
3	06-07-2003 14:02:48	791638300	POINT(-53.1797 -25.4131)
0	06-07-2003 14:02:49	81499100	POINT(-53.206 -25.4191)
0	06-07-2003 14:02:49	197913200	POINT(-53.2072 -25.4205)
1	06-07-2003 14:02:51	192550200	POINT(-56.8162 -23.3683)
2	06-07-2003 14:02:51	543745600	POINT(-56.8169 -23.3649)
2	06-07-2003 14:02:58	845758800	POINT(-57.0804 -23.9073)
2	06-07-2003 14:03:32	477928100	POINT(-57.2203 -23.4653)
3	06-07-2003 14:03:32	552393800	POINT(-53.3742 -25.02)
3	06-07-2003 14:03:33	786226872	POINT(-53.0286 -25.2467)
3	06-07-2003 14:03:33	934648871	POINT(-53.0212 -25.2518)
0	06-07-2003 14:03:42	157879600	POINT(-53.2666 -25.2942)

APÊNDICE 7

Experimento 1: Resultado da utilização do Weka : EM

cluster	data hora	nanosegundo	ponto
3	06-07-2003 14:00:24	677244300	POINT(-56.8424 -23.4344)
2	06-07-2003 14:00:28	881429800	POINT(-54.3578 -24.9972)
2	06-07-2003 14:00:34	686291200	POINT(-53.2624 -25.4211)
2	06-07-2003 14:00:34	801693900	POINT(-53.3884 -25.0663)
2	06-07-2003 14:00:38	766472100	POINT(-53.2613 -25.0402)
0	06-07-2003 14:01:03	393227400	POINT(-53.7766 -26.4043)
0	06-07-2003 14:01:03	466717500	POINT(-53.778 -26.4175)
3	06-07-2003 14:01:26	156116500	POINT(-57.098 -23.9135)
3	06-07-2003 14:01:28	224791400	POINT(-56.9007 -23.3977)
3	06-07-2003 14:01:28	400846400	POINT(-56.9172 -23.3995)
2	06-07-2003 14:01:29	958263700	POINT(-53.3312 -25.9759)
3	06-07-2003 14:01:38	201808935	POINT(-56.8251 -23.2361)
0	06-07-2003 14:02:00	50218400	POINT(-53.3314 -25.4218)
0	06-07-2003 14:02:10	507497600	POINT(-53.2958 -24.9138)
0	06-07-2003 14:02:23	74632600	POINT(-53.8606 -26.5946)
0	06-07-2003 14:02:23	328794900	POINT(-53.7307 -26.3982)
1	06-07-2003 14:02:43	787866900	POINT(-56.1822 -23.9662)
2	06-07-2003 14:02:48	637020500	POINT(-53.2008 -25.3935)
2	06-07-2003 14:02:48	701886213	POINT(-53.1826 -25.3955)
2	06-07-2003 14:02:48	791638300	POINT(-53.1797 -25.4131)
0	06-07-2003 14:02:49	81499100	POINT(-53.206 -25.4191)
0	06-07-2003 14:02:49	197913200	POINT(-53.2072 -25.4205)
3	06-07-2003 14:02:51	192550200	POINT(-56.8162 -23.3683)
3	06-07-2003 14:02:51	543745600	POINT(-56.8169 -23.3649)
1	06-07-2003 14:02:58	845758800	POINT(-57.0804 -23.9073)
3	06-07-2003 14:03:32	477928100	POINT(-57.2203 -23.4653)
2	06-07-2003 14:03:32	552393800	POINT(-53.3742 -25.02)
2	06-07-2003 14:03:33	786226872	POINT(-53.0286 -25.2467)
2	06-07-2003 14:03:33	934648871	POINT(-53.0212 -25.2518)
0	06-07-2003 14:03:42	157879600	POINT(-53.2666 -25.2942)

ANEXO 1

Código SRID: 4618 (GEO)

Auth_name: EPSG

Srtext:

GEOGCS["SAD69",DATUM["South_American_Datum_1969",SPHEROID["GRS 1967

(SAD69)",6378160,298.25,AUTHORITY["EPSG","7050"]],AUTHORITY["EPSG","6618"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4618"]]

Proj4:

+proj=longlat +ellps=aust_SA +no_defs

Código SRID: 29192 (UTM)

Auth_name: EPSG

Srtext:

PROJCS["SAD69 / UTM zone

22S",GEOGCS["SAD69",DATUM["South_American_Datum_1969",SPHEROID["GRS 1967

(SAD69)",6378160,298.25,AUTHORITY["EPSG","7050"]],AUTHORITY["EPSG","6618"]],PRIMEM["Greenwich",0,AUTHORITY["EPSG","8901"]],UNIT["degree",0.01745329251994328,AUTHORITY["EPSG","9122"]],AUTHORITY["EPSG","4618"]],PROJECTION["Transverse_Mercator"],PARAMETER["latitude_of_origin",0],PARAMETER["central_meridian",-

51],PARAMETER["scale_factor",0.9996],PARAMETER["false_easting",500000],PARAMETER["false_northing",1000000],UNIT["metre",1,AUTHORITY["EPSG","9001"]],AUTHORITY["EPSG","29192"]]

Proj4:

+proj=utm +zone=22 +south +ellps=aust_SA +units=m +no_defs