FILTROS RASTER PARA JUNÇÕES DE POLILINHAS

Leonardo Guerreiro Azevedo

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Jano Moreira de Souza, Ph.D.

Prof. Geraldo Zimbrão da Silva, D.Sc.

Profa. Claudia Bauzer Medeiros, Ph.D.

Prof. Claudio Esperança, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

AGOSTO DE 2001

AZEVEDO, LEONARDO GUERREIRO

Processamento Aproximado de Consultas em Bancos de Dados Espaciais Usando Assinaturas Raster [Rio de Janeiro] 2005.

XII, 108, p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia de Sistemas e Computação, 2005)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Banco de dados espaciais

2. Processamento aproximado de consultas

I. COPPE/UFRJ II. Título (série)

AGRADECIMENTOS

Aos professores Jano Moreira de Souza e Geraldo Zimbrão pela atenção, apoio, incentivo no desenvolvimento deste trabalho, e principalmente pela orientação que me proporcionaram desde o projeto final, ainda na graduação.

Ao Prof. Ralf H. Güting por ter me recebido em seu grupo de trabalho na Fernuniversität in Hagen (Alemanha), pelo carinho e pela orientação durante o meu doutorado sanduíche.

A minha amada esposa, Andréa, pela compreensão, por todo o carinho e apoio incondicional, desde o mestrado e durante todo o doutorado, sempre me dando força para continuar nesta empreitada.

Aos meus pais, Clelio e Jurêma, minha irmã, Alana, meus avós (Maurílio e Maria, e em memória Alfredo e Arminda) e amiga Edda que me estimularam a vir morar no Rio de Janeiro, com o intuito de prosseguir nos estudos, e agradeço também pela compreensão nas muitas vezes que fiquei ausente.

Aos grandes amigos Rodrigo Salvador, Patrícia Leal, Fernanda Baião, Prof. Blaschek, Prof.a Marta, Leonardo Aragão, meus sogros (Jorgina e Lauro), e muitos outros, cuja lista não caberia neste documento, que me apoiaram e incentivaram. Aos amigos da Alemanha, especialmente Frank e Romi Schreiber, Thomas e Claudia Steinhoff e Anne Jahn que deram atenção e carinho a mim e a minha esposa durante nossa estada na Alemanha.

A todos os funcionários e professores do Programa de Engenharia de Sistemas e Computação que colaboraram ao longo da evolução deste trabalho.

Ao CNPq pela ajuda financeira concedida.

Finalmente, a Deus que esteve sempre do meu lado.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

PROCESSAMENTO APROXIMADO DE CONSULTAS EM BANCOS DE DADOS ESPACIAIS USANDO ASSINATURAS RASTER

Leonardo Guerreiro Azevedo

Agosto/2005

Orientadores: Jano Moreira de Souza

Geraldo Zimbrão da Silva

Programa: Engenharia de Sistemas e Computação

O processamento tradicional de consultas tenta prover resultados exatos para as consultas maximizando o *throughput* e minimizando o tempo de resposta. Entretanto, em muitas aplicações, o tempo necessário para obter uma resposta exata é geralmente muito maior do que o desejado. Processamento aproximado de consultas vem ganhando evidência como uma abordagem alternativa para proporcionar ao usuário respostas para suas consultas em um tempo muito menor do que a abordagem tradicional, embora as respostas não sejam exatas. Existem várias técnicas para processamento aproximado de consultas em diversas áreas de pesquisa. Todavia, muitas delas são apenas apropriadas para processamento sobre dados tradicionais. Este trabalho propõe o uso de assinaturas *raster* para processamento aproximado de consultas em banco de dados espaciais. Nós enumeramos e propomos algoritmos para um conjunto de operações para processamento aproximado de consultas. Além disso, nós apresentamos resultados experimentais para algoritmos que nós propomos e implementamos. Os resultados demonstraram a eficiência de nossas propostas.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

APPROXIMATE QUERY PROCESSING IN SPATIAL DATABASES USING RASTER SIGNATURES

Leonardo Guerreiro Azevedo

August/2005

Advisors: Jano Moreira de Souza

Geraldo Zimbrão da Silva

Department: Systems and Computing Engineering

Traditional query processing provides exact answers to queries trying to maximize throughput while minimizing response time. However, in many applications, the response time of exact answers is often longer than what is acceptable. Approximate query processing has emerged as an alternative approach to give to the user an answer in a short time, although not an exact one. There is a large set of techniques for approximate query processing available in different research areas. However, most of them are only suitable for traditional data. This work proposes the use of raster signature for approximate query processing in spatial database. We enumerate and propose algorithms for a set of operations for approximate query processing. Besides, we present experimental results for the evaluation of the algorithms that we have proposed and implemented. We executed many experimental tests over real datasets, and the results demonstrate the effectiveness of our approach.

Índice

Índice de Figuras

Índice de Tabelas

1. Introdução

1.1. Motivação

O constante crescimento da capacidade de armazenamento de informações e a queda nos custos de *hardware* têm possibilitado que muitas aplicações lidem com grandes volumes de dados, envolvendo *Gigabytes*, *Terabytes* ou mesmo *Petabytes* de informações. Estes dados são geralmente armazenados em discos rígidos ou fitas. Como o tempo de acesso a disco é em torno de 10^4 e 10^7 vezes menor do que o tempo de acesso à memória principal, o simples acesso à informação pode demandar um grande período de tempo. Por outro lado, é importante que as consultas dos usuários sejam processadas rapidamente.

Um dos principais problemas na área de banco de dados é o processamento eficiente de consultas, ou seja, que usuários não tenham que esperar um tempo muito longo para receber respostas para suas requisições. Entretanto, existem muitas situações em que não é fácil atender a este requisito, por exemplo: o processamento de grandes volumes de dados requer um grande número de operações de entrada/saída que pode demandar um período longo de tempo (minutos ou mesmo horas de processamento); o acesso a dados remotos pode ser muito demorado devido a uma baixa velocidade de conexão ou a não disponibilidade temporária do dado. Além disso, em ambientes com critério de tempo de resposta estrito, um simples acesso a um determinado nível da hierarquia de armazenamento pode ser inaceitavelmente demorado. Por exemplo, para aplicações com tempo de resposta na ordem de milissegundos, um único acesso a disco pode ser considerado como tendo um tempo muito elevado (GIBBONS *et al*., 1997).

Somando-se a isso, receber uma resposta rápida pode ser mais importante para o usuário do que uma resposta acurada. Em outras palavras, a precisão da resposta da consulta poderia ser reduzida, e uma resposta aproximada retornada, desde que esta tenha tempo de processamento muito menor do que o necessário para obter a resposta exata, além de erro e intervalo de confiança aceitáveis.

Ambientes para os quais obter respostas exatas resulta em tempos indesejáveis de execução motivaram a pesquisa por técnicas para processamento aproximado de consultas. O objetivo é prover uma resposta estimada em um tempo muito menor do que

o necessário para computar uma resposta exata, evitando ou minimizando o número de acessos a disco para ler os dados reais (GIBBONS *et al*., 1997).

Existe um grande conjunto de técnicas para processamento aproximado de consultas disponível em diferentes áreas de pesquisa, tais como estimativa de seletividade e custo de consultas, mineração de dados, *data warehouse*, sistemas de suporte a decisão, OLAP (Processamento Analítico ou *Online Analytical Processing*), compactação de dados, recuperação da informação e visualização de dados. BARBARA *et al.* (1997) e HAN e KAMBER (2001) fazem um levantamento das técnicas existentes na literatura para processamento aproximado de consultas. Nestes trabalhos, muitas técnicas são descritas e avaliadas de acordo com os tipos de dados sendo reduzidos e as aplicações nas quais as técnicas podem ser empregadas. Todavia, a maior parte das técnicas é apropriada apenas para bancos de dados relacionais. Por outro lado, prover resposta para consulta em um tempo curto torna-se um desafio ainda maior na área de bancos de dados espaciais, onde os dados usualmente têm alta complexidade e aparecem em grandes volumes. Conseqüentemente, este é um importante tópico de pesquisa na área de banco de dados espaço-temporal, como apontado por RODDICK *et al*. (2004).

Além da complexidade e da grande quantidade de dados, RODDICK *et al*. (2004) enfatizam que muitas aplicações espaciais têm foco na recuperação de informações sumarizadas e aproximadas sobre objetos que satisfazem algum predicado espaço-temporal (por exemplo, “obter o número de carros que estarão no centro da cidade daqui a 10 minutos”), ao invés de estarem interessadas em informações exatas sobre qualificadores dos objetos (por exemplo, identificadores dos carros), as quais podem não estar disponíveis, ou mesmo serem irrelevantes. Por outro lado, técnicas para processamento exato de consultas sobre dados espaciais assumem que os atributos posicionais de objetos espaciais são conhecidos de forma precisa. Por muitos anos, a modelagem de dados espaciais tem implicitamente assumido que a extensão e, por conseguinte, os limites (ou bordas) de fenômenos espaciais são determinados de formas precisas e homogêneas, universalmente reconhecidas. A partir desta perspectiva, fenômenos espaciais são tipicamente representados por pontos, linhas e regiões (ou polígonos) descritos com precisão. Na prática, entretanto, fenômenos espaciais são conhecidos apenas de forma aproximada, ou seja, com erro dependendo da natureza da medida realizada ou da fonte de dados, como apontado por HEUVELINK (1998),

ZHANG e GOODCHILD (2002), e SCHNEIDER (1999). Assim, a resposta “exata”, na realidade, é uma aproximação, embora com valor muito próximo da resposta real.

Dados espaciais consistem de objetos espaciais representados por pontos, linhas, regiões (ou polígonos), janelas (ou retângulos), superfícies, e mesmo dados em dimensões maiores, que podem incluir também a dimensão tempo (SAMET, 1990). Exemplos de dados espaciais são: cidades, rios, rodovias, regiões administrativas, municípios, estados, áreas de utilização do solo, cadeias de montanhas etc. Atributos espaciais frequentemente aparecem juntos com atributos que representam informações não espaciais. Exemplos de dados não espaciais são: nomes de rodovias, endereços, números de telefones, nomes de cidades etc. Como dados espaciais e não espaciais estão intimamente relacionados, não é surpresa que muitas das questões que precisam ser estudadas são na realidade questões de bancos de dados.

Sistemas Gerenciadores de Bancos de Dados Espacias (SGBDE) provêem a tecnologia necessária para Sistemas de Informações Geográficas (SIG) e outras aplicações (GÜTING, 1994). Existe uma grande quantidade de aplicações na área de sistemas de bancos de dados espaciais, tais como: supervisão de tráfego, controle aéreo, previsão do tempo, planejamento urbano, otimização de rotas, cartografia, agricultura, administração de recursos naturais, monitoramento da costa, controle de fogo e epidemias (ARONOFF, 1989; TAO *et al*., 2003). Cada tipo de aplicação trata de diferentes características, escalas e propriedades espaço-temporais.

1.2. Cenários e Aplicações

Existem muitos cenários e aplicações nos quais uma resposta exata demorada pode ser substituída por uma resposta rápida e aproximada, desde que esta tenha a precisão desejada. Nesta seção, nós apresentamos alguns exemplos destes cenários e aplicações.

HELLERSTEIN *et al*. (1997) enfatizam que em Sistemas de Suporte a Decisão, o crescimento da competitividade dos negócios tem requerido uma indústria baseada na informação capaz de realizar maior uso dos dados acumulados. Dessa forma, técnicas para apresentação de dados úteis para tomadores de decisão em um tempo razoável tornam-se cruciais.

HELLERSTEIN *et al*. (1997) propõem também o uso de processamento aproximado de consultas durante a execução de seqüência de consultas *drill-down* em

mineração de dados *ad-hoc*. As consultas iniciais da seqüência são processadas somente com o intuito de determinar quais são as consultas mais relevantes.

PAPADIAS *et al*. (2001) propõem o uso de processamento aproximado de consultas em operações OLAP.

GIBBONS *et al*. (1997) enfatizam que respostas aproximadas podem prover informações sobre a aplicabilidade de uma consulta. Além disso, respostas aproximadas podem também ser usadas quando a consulta retorna respostas numéricas, e a precisão exata da resposta não é necessária, por exemplo, total, média, ou percentual para os quais apenas os primeiros dígitos de precisão são importantes (primeiros dígitos em um total de milhões, ou os primeiros percentis de um percentual).

FURTADO e COSTA (2002) e COSTA e FURTADO (2003) apresentam a necessidade de eficiência e escalabilidade quando grandes volumes de dados são acessados durante processos de análise em *data warehouse*. As aplicações executam consultas em grandes volumes de dados (*Gigabytes* ou *Terabytes* de informações) e seus usuários não estão dispostos em esperar minutos ou mesmo horas em qualquer um dos passos iterativos de análise e exploração de dados. Este problema tem motivado uma grande quantidade de pesquisas por estruturas de dados, estratégias e sistemas de processamento paralelo para prover significante aceleração e escalonamento nestas aplicações. Entretanto, em muitas organizações não é fácil justificar investimentos elevados para a aquisição de complexos sistemas de computação paralela ou distribuída que utilizam arquiteturas de *hardware* especiais. Por outro lado, o problema “tempo de resposta” pode ser resolvido em muitos casos retornando respostas aproximadas a partir de pequenos resumos dos dados reais.

IOANNIDIS e POOSALA (1995) e GIBBONS *et al*. (1997) propõem o uso de processamento aproximado de consultas para definir o plano de execução mais eficiente para uma consulta. DAS *et al*. (2004) indicam o seu uso para estimativa de seletividade a fim de retornar resultados aproximados com garantias de qualidade provadas estatisticamente.

Uma resposta aproximada também pode ser usada como uma resposta alternativa quando os dados não estão disponíveis em ambientes de *data warehouse* e em sistemas de armazenamento de dados distribuídos, como apontado por GIBBONS *et*

al. (1997), FALOUTSOS *et al*. (1997) e JAGADISH *et al*. (1995), ou em computação móvel, como enfatizado por MADRIA *et al*. (1998).

Em computação móvel, pode ser vantajoso relaxar critérios de completude ou precisão a fim de enviar para o usuário uma resposta aproximada em casos de baixa conexão de rede, não disponibilidade temporária dos dados, ou mesmo poucos recursos de conexão a internet ou de memória disponível para processar a consulta (MADRIA *et al.*, 1998).

DOBRA *et al*. (2002) sugerem o uso de processamento aproximado para tomada de decisões e para inferir *online* padrões de interesse, por exemplo, sobre *streams* contínuos de dados. Em ambientes de dados *stream*, usualmente estão disponíveis recursos limitados de memória para processamento das consultas em relação ao grande volume de dados. Portanto, é necessária a pesquisa por algoritmos para sumarização de dados *stream* em resumos concisos, com baixo custo de armazenamento, e que permitam o processamento aproximado de consultas com garantias de precisão nas respostas obtidas. Dessa forma, o processamento aproximado de consultas é apropriado à natureza exploratória da maioria das aplicações que realizam processamento sobre dados *stream*, por exemplo, aplicações de análise de tendência e detecção de fraudes/anomalias em dados de telecomunicações-redes, cujo objetivo é identificar padrões genéricos de interesse ou que estão “fora da ordem comum” ao invés de prover resultados exatos.

1.3. Definição do Problema

Consultas de seleção e junção são operações fundamentais em SGBDE. A seleção espacial recupera de um conjunto de dados os objetos que satisfazem algum predicado espacial em relação a um objeto *q* de referência. O tipo de seleção espacial mais comum é a consulta por janela, na qual o predicado é interseção e *q* define uma janela no espaço dos dados. Um exemplo de consulta seria “determinar todos os lagos que têm interseção com os limites de uma cidade”. O objeto referência que define a janela seria a cidade representada, por exemplo, como um retângulo. A operação junção espacial retorna, a partir de dois conjuntos de dados, os pares de objetos que satisfazem um predicado espacial, sendo que interseção é o predicado mais comum. Um exemplo de consulta seria “retornar todas as cidades que são atravessadas por rio(s)” (PAPADIAS *et al.*, 1999).

Junções espaciais foram amplamente estudadas na literatura, e existem várias abordagens para processamento de junções espaciais. ZHU *et al.* (2000) enfatizam que as abordagens tradicionais realizam a junção espacial em dois passos (ORENSTEIN, 1986; KOTHURI e RAVADA, 2001). Os autores propõem algoritmos eficientes para serem usados no segundo passo. Nesta abordagem de dois passos, apresentada na Figura 1, o primeiro passo emprega um Método de Acesso Espacial (MAE ou SAM – *Spatial Access Method*) a fim de reduzir o espaço de busca. O Retângulo Mínimo Envolvente (RME ou MBR – *Minimum Bounding Rectangle*) é o Método de Acesso Espacial comumente mais empregado. Este passo não tem como saída o resultado da operação de junção. Ao invés disso, ele retorna um conjunto de pares candidatos que correspondem a um super-conjunto da solução. O segundo passo, chamado de passo de refinamento, compara os pares resultantes do primeiro passo, lendo do disco e processando suas representações reais. Este é o passo mais custoso, e requer tempo de operações de Entrada/Saída para ler os objetos espaciais do disco, e tempo de UCP (Unidade Central de Processamento) para computar a interseção exata. Como resultado, pares candidatos provenientes do primeiro passo são descartados, sendo considerados como falsos candidatos, e outros são aceitos no conjunto resposta.

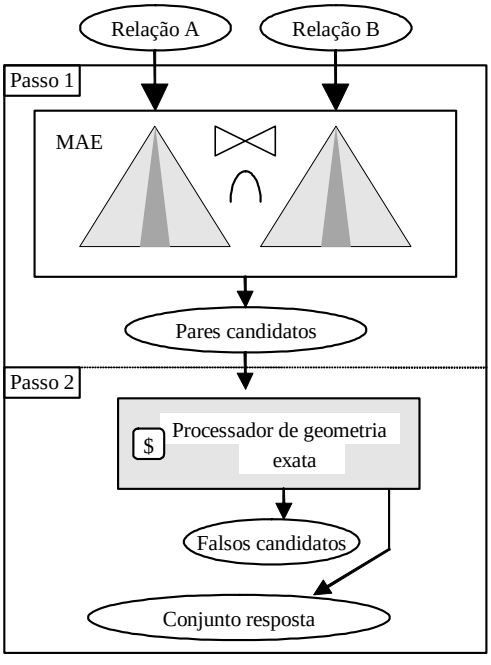


Figura 1. Arquitetura para processamento de junções espaciais em dois passos.

BRINKHOFF *et al.* (1994) propõem uma arquitetura de três passos para processamento de junção espacial chamada de Processador de Consultas em Múltiplos

Passos (PCMP ou MSQP – *Multi-Step Query Processor*), apresentado na Figura 2. Nesta arquitetura outro passo é adicionado entre o primeiro passo (MAE) e o segundo passo (processador de geometria exata). O novo passo proposto consiste em comparar os pares candidatos resultantes do primeiro passo através de um filtro geométrico. Um filtro geométrico usa uma representação compacta e aproximada do objeto tentando reter suas principais características, por exemplo, 4CRS (ZIMBRAO e SOUZA, 1998), *Convex Hull*, 5C, RMBR e outras encontradas em BRINKHOFF *et al.* (1993). Como resultado deste passo, nós temos três possibilidades: pares de objetos que fazem parte do conjunto solução (aceitos); pares que não fazem parte da solução (rejeitados); e, pares inconclusivos. Existem duas vantagens principais na inclusão deste passo. Primeiro, o tamanho da aproximação é apenas uma fração do tamanho do objeto espacial e, dessa forma, ela pode ser armazenada no índice, juntamente com o MBR dos objetos. Segundo, testar duas aproximações tem um custo de tempo de UCP muito menor do que testar os objetos reais.

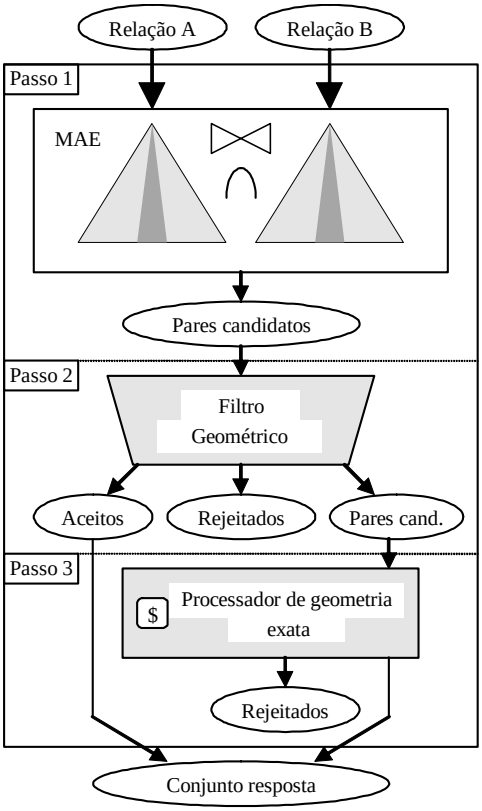


Figura 2. Arquitetura para processamento de consultas em múltiplos passos (MSQP) (BRINKHOFF *et al.*, 1994).

Em ambas as abordagens (processar a junção espacial em dois ou três passos) é necessário o processamento das geometrias exatas dos objetos, o qual consiste no passo mais custoso, consumindo elevados recursos de UCP e operações de Entrada/Saída. Até onde sabemos, não existe uma abordagem que proponha a não execução deste passo, retornando para o usuário uma resposta aproximada com um intervalo de confiança através do processamento da consulta sobre aproximações dos dados, não requerendo a leitura do disco e o processamento dos objetos reais. Uma arquitetura para esta nova abordagem é apresentada na Figura 3. Neste caso, o passo de Filtro Geométrico da arquitetura de dois passos apresentada na Figura 1 é substituído pelo "Processador aproximado de consultas", o qual processa as representações dos objetos reais a fim de retornar uma resposta aproximada para a consulta, juntamente com um intervalo de confiança.

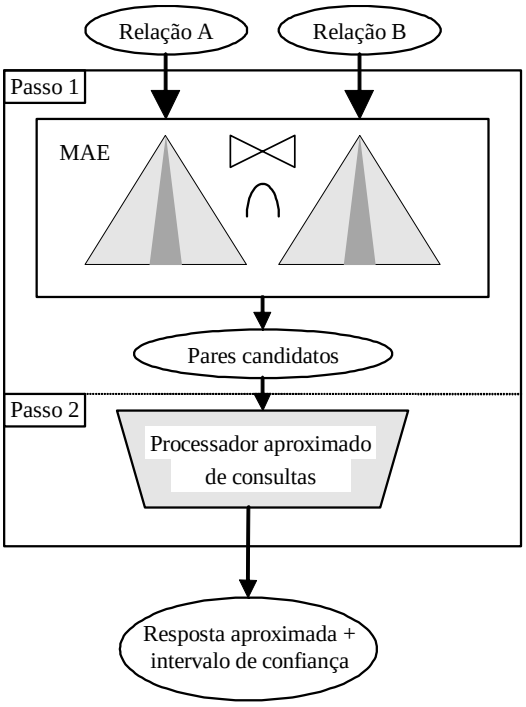


Figura 3. Arquitetura para processamento aproximado de junções espaciais.

1.4. Propostas deste Trabalho

Este trabalho está relacionado com o uso de aproximações *raster* para processamento aproximado de consultas. Nós propomos o uso da Assinatura *Raster* de Quatro Cores (4CRS ou *Four-Colors Raster Signature*), proposta por ZIMBRAO e SOUZA (1998), para processamento rápido e aproximado de consultas sobre conjuntos de dados formados por polígonos. Assinaturas 4CRS armazenam as principais

características dos dados em representações aproximadas e compactas que podem ser acessadas e processadas de forma mais rápida do que os dados reais. Assim, ao invés de acessar o dado real, nós propomos executar a consulta sobre as assinaturas 4CRS dos polígonos. Como resultado, uma resposta aproximada da consulta é retornada em um tempo muito menor do que o necessário para processar a resposta exata. A resposta é estimada e não é exata. Todavia, uma medida de precisão é retornada como um intervalo de confiança que permite que o usuário decida se a precisão da resposta é suficiente. Em geral, uma resposta aproximada é suficiente em várias aplicações.

Neste trabalho, além de apresentarmos operações espaciais que podem ser processadas de forma aproximada, nós propomos, implementamos, e avaliamos propostas de algoritmos para as operações apresentadas a seguir:

- Algoritmo para computar a área aproximada de polígonos (AZEVEDO *et al.*, 2004);
- Algoritmo para computar a área aproximada de polígonos dentro de janela (AZEVEDO *et al.*, 2004);
- Algoritmo para computar a área aproximada de interseção de polígonos (AZEVEDO *et al.*, 2005).

Nós também propomos novas fórmulas para computar o intervalo de confiança das respostas retornadas por estes algoritmos. Como outra contribuição deste trabalho, nós propomos um novo algoritmo eficiente para computar assinaturas 4CRS de polígonos.

Um exemplo de aplicação que pode se beneficiar da nossa proposta é a estimava da produção agrícola. De acordo com valores estimados de produção agrícola, muitas decisões devem ser tomadas, por exemplo, número e tamanho de depósitos para armazenar a colheita, meios de transportes que devem estar disponíveis, rodovias e ferrovias que devem ser (re)construídas etc. Muitas junções espaciais envolvendo sobreposição de planos temáticos tais como solo, áreas rurais, indicadores de pluviosidade, poluição, áreas que estão vulneráveis a ataques de pestes etc. devem ser avaliados para estimar a produção agrícola, o que pode demandar um grande período de tempo. Por outro lado, uma resposta rápida aproximada pode ser suficiente para estimar a produção agrícola.

Outro exemplo de consulta que poderia valer de um processamento aproximado para ser executada rapidamente é a consulta apresentada por VEENHOF *et al.* (1995): "recuperar todas as áreas rurais abaixo do nível do mar que possuam tipo de solo igual a areia e estejam distantes três milhas de lagos poluídos". Neste caso, para responder a esta consulta seria necessário executar junções espaciais envolvendo os seguintes planos temáticos:

- Uso da terra: considerando tipo de uso da terra igual à área rural;
- Solo: considerando tipo de solo igual à areia;
- Lagos: considerando os lagos poluídos;
- Altimétrico: correspondente à medição das elevações de pontos da superfície.

Para a consulta apresentada no exemplo, tanto o operador sobreposição como o operador distância seriam executados de forma aproximada a fim de retornar uma resposta rápida para o usuário.

1.5. Estrutura da Tese

Esta tese está dividida da seguinte forma:

- O Capítulo 1 é a presente introdução. Neste capítulo foi apresentada a motivação para este trabalho; foram relatados cenários e aplicações para processamento aproximado de consultas; o problema para o qual esta tese propõe soluções foi definido; e nossas propostas foram enumeradas.
- O Capítulo 2 faz um levantamento dos trabalhos relacionados encontrados na literatura.
- O Capítulo 3 apresenta detalhes da Assinatura *Raster* de Quatro Cores (4CRS), a qual foi escolhida como técnica para redução de dados para aproximar polígonos. Além disso, um novo algoritmo para geração de assinaturas 4CRS é proposto.
- O Capítulo 4 propõe o uso da Assinatura *Raster* de Quatro Cores para processamento aproximado de consultas. O objetivo é processar consultas sobre as assinaturas 4CRS de polígonos, retornando uma resposta aproximada para o usuário, bem como o intervalo de confiança da resposta

obtida. Dessa forma as representações reais dos objetos não são processadas, e o teste de geometria exata não é executado, o qual se constitui no passo mais custoso durante o processamento de consultas. No Capítulo 4, nós apresentamos direções para projeto, implementação e avaliação de novos algoritmos que estão de acordo com os requisitos dessa nova abordagem. Além disso, nós mostramos detalhes de três propostas de algoritmos para processamento aproximado de consultas, juntamente com propostas de fórmulas para calcular o intervalo de confiança das respostas obtidas.

- O Capítulo 5 é dedicado à apresentação de resultados de testes experimentais que demonstram a eficiência de nossa nova abordagem.
- Finalmente, no Capítulo 6, as conclusões e os trabalhos futuros são apresentados.

2. Trabalhos relacionados

Este capítulo apresenta uma revisão bibliográfica das principais técnicas da literatura para respostas aproximadas de consultas e redução de dados, e que são relevantes para o presente trabalho.

Existem muitas técnicas para processamento aproximado de consultas, todavia, muitas delas são apenas aplicáveis em bancos de dados tradicionais, não atendendo as peculiaridades dos bancos de dados espaciais. RODDICK *et al*. (2004) apontam como um tópico importante de pesquisa o desenvolvimento de novas técnicas para processamento aproximado de consultas que atendam aos requisitos de bancos de dados espaço-temporais.

BARBARA *et al*. (1997) e HAN e KAMBER (2001) apresentam revisões bibliográficas das principais técnicas existentes na literatura. Em ambos os trabalhos, são apresentadas taxonomias para a classificação das técnicas existentes, de acordo com os tipos de dados sendo reduzidos e as aplicações onde as técnicas podem ser empregadas. No entanto, a classificação proposta por HAN e KAMBER (2001), que WU *et al.* (2001) corroboram, é mais ampla e, portanto, será usada para apresentar alguns dos trabalhos existentes na literatura. A classificação proposta por HAN e KAMBER (2001) é apresentada a seguir, sendo detalhada nas próximas seções:

- Agregação de cubo de dados: corresponde à aplicação de operações de agregações durante a construção de cubo de dados (Seção 2.1);
- Redução de dimensões: dimensões ou atributos menos relevantes ou redundantes são detectados e removidos (Seção 2.2);
- Compactação de dados: mecanismos de codificação são aplicados para geração de dados reduzidos (ou compactados) sobre os quais as consultas são processadas (Seção 2.3);
- Redução de numerosidade: os dados reais são substituídos ou estimados por representações alternativas e menores. HAN e KAMBER (2001) subdividem esta classe de técnicas em técnicas parametrizadas e técnicas não-parametrizadas. Nas técnicas parametrizadas, um modelo é usado para estimar os dados, sendo necessário armazenar apenas os parâmetros para o

modelo, ao invés de armazenar os dados reais. Métodos não parametrizados armazenam representações reduzidas dos dados originais (Seção 2.4).

- Geração de hierarquias conceituais e discretização: valores dos atributos são substituídos por intervalos ou níveis conceituais mais amplos (Seção 2.5).

2.1. Agregação de Cubo de Dados

Agregação de cubo de dados refere-se à aplicação de operações de agregação durante a construção de cubos de dados. Por exemplo, considere um banco de dados que armazene as transações diárias das vendas realizadas por uma empresa e que um usuário tomador de decisão esteja interessado nas vendas anuais (total por ano), ao invés do detalhamento diário. Logo, os dados poderiam ser agregados a fim de que a informação resultante fosse um resumo do dado detalhado, representando os dados das vendas agrupados por ano. Sendo assim, o conjunto de dados resultante seria muito menor em volume, e sem perda de informação necessária para a tarefa de análise.

Cubos de dados provêem acesso rápido a dados pré-computados e sumarizados, dessa forma, eles devem ser usados sempre que possível para responder consultas sobre dados agregados, por exemplo, em consultas OLAP (*Online Analytical Processing*) ou operações de mineração de dados.

Dentre os trabalhos da literatura, destacamos aqueles apresentados por SARAWAGI e STONEBRAKER (1994), AGARWAL *et al.* (1996), e ROSS e SRIVASTAVA (1997) os quais apresentam algoritmos para consultas sobre cubos de dados e suas pré-computações sobre dados relacionais.

CANNATARO *et al.* (2002) propõem o uso de agregação de cubos de dados para sintetizar documentos XML. A semântica dos documentos XML é extraída e versões sintetizadas em representações multidimensionais são geradas. O documento XML é processado de forma que os elementos são considerados como tuplas de relações. Medidas e dimensões são escolhidas e uma representação multidimensional é construída, a qual é estruturada em um cubo de dados. Assim sendo, o documento é reorganizado de acordo com alguma função de agregação, resultando em uma versão resumida do dado original.

ZHOU *et al*. (1999) destacam que, nos processos envolvendo OLAP espacial e mineração sobre dados espaciais, o processamento dos dados espaciais ainda é um

gargalo, já que as novas aplicações analisam grandes quantidades de dados complexos empregando operações espaciais de alto custo computacional e de leitura/escrita no disco. Dessa forma, o desenvolvimento de algoritmos que atendam aos requisitos de OLAP espacial e de mineração de dados espaciais é um importante tópico de pesquisa de técnicas para processamento aproximado de consultas. ZHOU *et al*. (1999) propõem um algoritmo para a operação *polygon amalgamation* intensamente utilizada em OLAP espacial e mineração de dados espaciais. Dado um conjunto de polígonos de entrada, a operação *polygon amalgamation* produz como saída um novo polígono que corresponde à borda da união dos polígonos fontes. Neste trabalho, propomos um algoritmo para computar a união de polígonos de forma aproximada.

2.2. Redução de Dimensões

Durante tarefas de análise, os conjuntos de dados a serem processados podem conter uma grande quantidade de atributos dos quais muitos não são relevantes, ou são redundantes. Dessa forma, o conhecimento de um especialista no domínio da aplicação poderia ser utilizado para remoção destes atributos e, conseqüente, redução do volume de dados. Todavia, esta tarefa nem sempre é fácil de ser realizada e pode demandar muito tempo. Por outro lado, remover atributos relevantes ou manter atributos irrelevantes pode causar detrimento e confusão para algoritmos empregados, por exemplo, em processos de mineração. Isto pode produzir padrões descobertos de baixa qualidade. Além disso, manter atributos irrelevantes pode aumentar consideravelmente o tempo de processamento (HAN e KAMBER, 2001).

Através da técnica de redução de dimensões o tamanho do conjunto de dados é diminuído pela remoção de atributos (ou dimensões) não relevantes. Geralmente, métodos de seleção de subconjuntos de atributos são aplicados. Tais métodos tentam encontrar o subconjunto mínimo de atributos cujas distribuições de probabilidades das classes de dados resultantes são as mais próximas possíveis das distribuições originais (usando todos os atributos). Técnicas para seleção de atributos são apresentadas em vários livros textos citados em HAN e KAMBER (2001), além das propostas apresentadas por KOHAVI e JOHN (1997), DASH *et al.* (1997) e SIEDLECK e SKLANSKY (1988).

Propostas de técnicas para redução de dimensões não fazem parte do escopo deste trabalho, o qual tem objetivo de apresentar propostas de operações para

processamento aproximado sobre dados espaciais, não se preocupando com a relevância dos atributos dos objetos processados. Todavia, técnicas para redução de dimensões poderiam ser empregadas com o intuito de reduzir o volume de dados a ser processado pelas técnicas de processamento aproximado de consultas.

2.3. Compactação de Dados

Técnicas para compactação de dados executam transformações ou codificações dos dados gerando uma representação reduzida ou compacta dos dados originais. Existem técnicas sem perda de informação e técnicas com perda de informação. No primeiro caso, o dado original pode ser inteiramente reconstruído a partir do dado reduzido. No segundo caso, aproximações dos dados originais são obtidas a partir do dado compactado. Os dois métodos mais populares e efetivos para compactação de dados com perda são transformadas *wavelet* e Análise de Componentes Principais, os quais são apresentados a seguir (HAN e KAMBER, 2001).

O uso de transformadas *Wavelets* (DWT ou *Discrete Wavelet Transform*) para compactação de dados foi estudado inicialmente por MATIAS *et al.* (1998), os quais definem *Wavelet* como uma ferramenta matemática para decomposição hierárquica de funções. *Wavelets* representam uma função em uma forma grosseira do dado como um todo, juntamente com coeficientes de detalhes que influenciam a função em várias escalas (STOLLNITZ *et al.*, 1996). A transformada *Wavelet* transforma um vetor de dados em um vetor formado pelos coeficientes *Wavelet*. O vetor transformado pode ser truncado a fim de armazenar apenas os coeficientes mais significativos, obtendo assim a compactação desejada. Algoritmos que são eficientes perante dados esparsos podem ser executados diretamente sobre os coeficientes, sendo a computação extremamente rápida como, por exemplo, os algoritmos apresentados em CHAKRABARTI *et al.* (2001) e VITTER *et al.* (1998, 1999).

Outra técnica para compactação de dados é Análise de Componentes Principais (PCA ou *Principal Components Analysis*). PCA busca pelos c vetores ortogonais de k-dimensões ($k \leq c$) que melhor representem os dados. Os dados são então projetados nos vetores de k-dimensões, um espaço muito menor, permitindo a compactação dos dados. Diferentemente da técnica redução de dimensões, que promove a redução dos dados através da remoção de atributos irrelevantes, PCA combina a essência dos atributos criando um conjunto alternativo e menor de variáveis, sobre as quais os dados são

projetados. Técnicas PCA têm baixo custo computacional, podem ser aplicadas sobre atributos ordenados ou desordenados, além de tratarem dados esparsos e não regulares (HAN e KAMBER, 2001).

Outros exemplos de técnicas para compactação de dados são Quanticubes e FCompress apresentadas em FURTADO e MADEIRA (2000a, 2000b), respectivamente, que têm o objetivo de compactar tabelas de fatos e cubo de dados em *data warehouses*.

A técnica Quanticubes (FURTADO e MADEIRA, 2000a) inicialmente realiza uma análise dos dados determinando uma codificação de tamanho fixo ótima em relação a critérios de erro e espaço de armazenamento através de um procedimento de quantização. Em seguida, os valores dos cubos de dados são codificados, utilizando a codificação determinada anteriormente, e são armazenados para obter uma grande taxa de compactação. Por exemplo, uma célula de cubo de dados que ocupa quatro *bytes* é compactada em um código binário de tamanho determinado (por exemplo, nove bits). Finalmente, um processo de descompactação extremamente rápido "*on-the*-fly" dos valores individuais é realizado descompactando o dado e "desquantizando" os mesmos, através de operações de baixo nível.

Já a técnica FCompress tem como estratégia de compactação a substituição dos valores dos atributos das tuplas das tabelas de fato ou células do cubo por representações muito menores formadas por códigos binários. Assim como é realizado na técnica *Quanticubes*, os códigos binários são obtidos pela quantização (LLOYD, 1982; SAYOOD, 1996) dos valores dos atributos.

Este trabalho propõe o uso de assinaturas *raster* para processamento aproximado de consultas sobre polígonos. As assinaturas *raster* se constituem em uma forma de compactação através da representação dos objetos em um *bit-map* de tamanho reduzido construído sobre uma grade de células. A Assinatura *Raster* de Quatro Cores (4CRS ou *Four-Colors Raster Signature*), proposta por ZIMBRAO e SOUZA (1998) e detalhada na Seção 3.1, foi escolhida como forma de compactação dos dados devido aos resultados obtidos em relação as demais propostas da literatura, como demonstrado por ZIMBRAO e SOUZA (1998). Sendo assim, os algoritmos para processamento aproximado de consultas propostos neste trabalho processam assinaturas 4CRS.

2.4. Redução de Numerosidade

Segundo HAN e KAMBER (2001), as técnicas para redução de numerosidade podem ser divididas em técnicas parametrizadas e técnicas não parametrizadas. BARBARA *et al.* (1997) corroboram esta proposta de classificação. Nas técnicas parametrizadas, um modelo é utilizado para estimar os dados e apenas parâmetros são armazenados, ao invés de se armazenar os dados reais. Já técnicas não parametrizadas não assumem nenhum modelo para os dados, os quais são reduzidos através do armazenamento de representações menores.

2.4.1. Técnicas Parametrizadas

Modelos de regressão e modelos *log linear* (JOHNSON e WICHERN, 1992) são exemplos de técnicas parametrizadas. Regressão Linear (WONNACOTT e WONNACOTT, 1985) é a forma mais simples de regressão, na qual uma variável Y é modelada como uma função linear de outra variável X. Já Regressão Múltipla permite modelar uma variável Y como uma função linear de um vetor multidimensional (PRESS *et al.*, 1996). *Log linear* é uma metodologia para aproximação de distribuições de probabilidades multidimensionais discretas. Uma tabela multidimensional de probabilidades conjuntas é aproximada pelo produto de valores de tabelas de dimensões menores. Existem vários livros texto descrevendo modelos *log linear*, tais como AGRESTI (1990) e BISHOP *et al.* (1975). Algumas outras propostas da literatura são apresentadas em MALVESTUTO (1991) e PEARL (1988).

2.4.2. Técnicas Não Parametrizadas

Dentre as técnicas não parametrizadas destacamos histogramas, *clustering*, estruturas de índice e amostragem.

2.4.2.1. Histogramas

Histogramas aproximam um ou mais atributos dos dados de uma relação pelo agrupamento dos seus valores em subconjuntos. Os valores dos atributos e suas freqüências nos dados base são armazenados em cada subconjunto, de acordo com estatísticas de sumarização. POOSALA *et al.* (1996) propõem um *framework* para caracterização de histogramas, e apresentam resultados que permitem identificar a melhor estratégia a ser utilizada segundo critérios de tempo de construção e precisão das respostas obtidas. Histogramas têm as vantagens de serem simples, serem fáceis de se

construir e de manter, apresentarem baixo custo de execução, ocupar pequeno espaço para armazenamento e não requerem a representação de uma distribuição de probabilidade (FURTADO e MADEIRA, 1999).

2.4.2.2. *Clustering*

A técnica de *cluster* agrupa os dados (ou objetos) em subconjuntos de forma que objetos que participam de um mesmo grupo sejam muito semelhantes, enquanto que objetos de grupos diferentes sejam bastantes diferentes uns dos outros. Semelhança é geralmente definida em termos de "quão próximos" os objetos estão no espaço, baseado em uma função de distância. Na redução de dados, as representações dos *clusters* são usadas para substituir os dados reais durante as consultas e operações de análise. Algoritmos para *clustering* são apresentados por NG e HAN (1994), KAUFMAN e ROUSSEEUW (1990), ESTER *et al.* (1995), ZHANG *et al.* (1996) e WANG *et al.* (1997).

2.4.2.3. Estruturas de Índice

Estruturas de indexação multidimensional em árvore são principalmente utilizadas para prover mecanismos de acesso rápido aos dados. Por outro lado, tais estruturas também podem ser utilizadas para prover redução de dados de forma hierárquica através da determinação de agrupamentos (*clusters*) dos dados. Isto pode ser usado para prover respostas aproximadas para consultas. Uma estrutura de indexação divide recursivamente o espaço multidimensional para um dado conjunto de objetos. O nó raiz representa o espaço em sua totalidade. Dessa forma, informações podem ser armazenadas no índice em diferentes níveis de resolução ou abstração. Assim, pode ser possível responder consultas de forma aproximada, consultando apenas as informações armazenadas no índice, não necessitando acessar os dados reais. Se um filho de um nó for considerado como um subconjunto (*bucket*), então um índice em árvore pode ser utilizado como um histograma hierárquico. GAEDE e GÜNTHER (1998) apresentam uma revisão bibliográfica de mais de 25 estruturas para dados multidimensionais. Como exemplos se encontram as árvores B (BAYER e MCCREIHT, 1972), B+ (COMER, 1979), além dos índices multidimensionais, tais como R-Trees (GUTTMAN, 1984), Quad-Trees (FINKEL e BENTLEY, 1974) e suas variações (SAMET, 1990). Em AOKI (1998) é apresentado o uso de árvores de indexação para agregação de dados.

2.4.2.4. Amostragem

Amostragem pode ser utilizada para redução de dados através da representação dos dados reais sob a forma de conjuntos menores formados por amostras dos dados. BARBARA *et al.* (1997) ressaltam algumas aplicações que utilizam técnicas de amostragem como, por exemplo, otimização de consultas, processamento de consultas em paralelo, suporte para auditoria, mineração de dados e respostas aproximadas para consultas. Em BARBARA *et al.* (1997) também são apresentas várias técnicas existentes na literatura para amostragem de dados.

Uma das vantagens do uso de amostragem para redução de dados é que seu custo é proporcional ao número de amostras *n* em oposição ao tamanho da base *N*. O tamanho da amostra pode ser extremamente menor do que o tamanho da base.

2.5. Discretização e Geração de Hierarquias Conceituais

Técnicas de discretização são utilizadas para redução de valores de atributos contínuos dividindo a gama de valores em intervalos que são utilizados em substituição aos dados reais. Muitas das técnicas de discretização são recursivas e provêem um particionamento hierárquico dos valores dos atributos (HAN e KAMBER, 2001).

Hierarquias conceituais são utilizadas para redução de dados através da substituição de conceitos de níveis mais baixos por conceitos em níveis mais altos. Como exemplo, valores de um atributo idade poderiam ser substituídos por conceitos em níveis mais altos, tais como criança, jovem, adulto e idoso. Embora detalhes sejam perdidos durante o processo de generalização, o dado generalizado pode ter um maior significado sendo mais fácil interpretá-lo. Além disso, é necessário menos espaço para armazenamento (HAN e KAMBER, 2001).

HAN e KAMBER (2001) apresentam um conjunto de técnicas da literatura para geração automática de hierarquias conceituais a partir de atributos com valores discretos e categorias. Técnicas para discretização e geração de hierarquias conceituais também estão fora do escopo deste trabalho.

3. Novo Algoritmo para Computar Assinaturas *Raster* de Quatro Cores

Este capítulo apresenta em detalhes a Assinatura *Raster* de Quatro Cores (4CRS ou *Four-Color Raster Signature*), principal conceito ao qual este trabalho está relacionado; e propõe um novo algoritmo para geração das assinaturas. Este novo algoritmo foi utilizado para a geração das assinaturas 4CRS utilizadas nos nossos testes experimentais. Os resultados obtidos demonstram que as assinaturas podem ser geradas em um tempo muito curto, como será apresentado na Seção 5.3 (Tabela 8).

O capítulo está estruturado da seguinte maneira: na Seção 3.1 são apresentados detalhes da assinatura 4CRS; na Seção 3.2, conceitos preliminares relacionados à proposta do novo algoritmo para geração de assinaturas *raster* são apresentados; a Seção 3.3 propõe um novo algoritmo para geração de assinaturas 4CRS.

3.1. Assinatura Raster de Quatro Cores

A Assinatura *Raster* de Quatro Cores (4CRS ou *Four-Color Raster Signature*) foi proposta por ZIMBRAO e SOUZA (1998) com o objetivo de ser usada como estrutura para aproximação de polígonos no processamento de junção espacial. Como a sobreposição de objetos espaciais é de grande interesse para muitas aplicações, naquele trabalho a assinatura 4CRS foi empregada exclusivamente como filtro no teste de interseção de polígonos durante o processamento de junções espaciais. As características e vantagens da assinatura 4CRS sobre outros métodos motivaram seu uso na área de processamento aproximado de consultas. O objetivo dessa nova abordagem é reduzir o tempo necessário para processar a consulta, evitando acessar os dados reais, o que pode demandar um grande período de tempo, e processar a consulta de forma aproximada através da execução de um algoritmo eficiente sobre dados aproximados (assinaturas 4CRS dos objetos), muito menores do que os dados reais. Dessa forma, o tempo necessário para computar uma resposta aproximada é muito menor do que o tempo requerido para obter a resposta exata. Por outro lado, a resposta será estimada, e não exata. Logo, é também necessário que um intervalo de confiança seja retornado, a fim de prover uma medida de precisão da resposta obtida. Em geral, uma resposta aproximada é suficiente para o usuário poder tomar sua decisão desde que esta seja obtida em um tempo curto e tenha a precisão desejada.

3.1.1. Definição

A assinatura 4CRS (ZIMBRAO e SOUZA, 1998) de um polígono é uma aproximação *raster* representada por um pequeno mapa de *bits* de quatro cores em uma grade de células. A cada célula da grade é atribuída uma cor denotando o percentual de área do polígono dentro da célula, como apresentado na Tabela 1. Na

Figura 4, um exemplo de assinatura 4CRS é mostrado. A grade pode ter sua escala modificada a fim de obter uma representação mais precisa (maior resolução) ou mais compacta (menor resolução). Maiores detalhes sobre assinaturas 4CRS podem ser encontrados em ZIMBRAO e SOUZA (1998) e AZEVEDO *et al.* (2004).

Tabela 1. Tipos de células 4CRS

Bit value	**Cell type**	**Description**
00	Vazio	A célula não é interceptada pelo polígono
01	Pouco	A célula possui um percentual de 50% ou menos de interseção com o polígono
10	Muito	A célula possui um percentual de interseção com o polígono maior do que 50% e menor do que 100%.
11	Cheio	A célula é completamente ocupada pelo polígono.

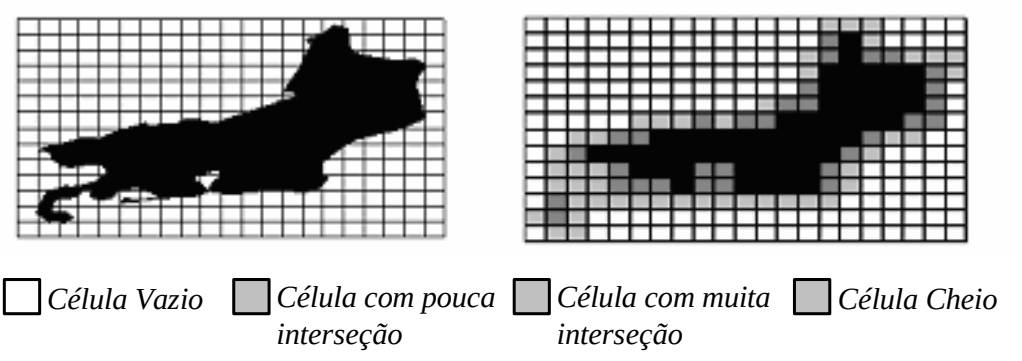


Figura 4. Exemplo de assinatura 4CRS.

3.1.2. Divisão do Espaço em Células

Assinaturas *raster* são construídas sobre grades de células. Ao testar as células de dois objetos, as células de suas assinaturas que se sobrepõem devem ser comparadas. Todavia, apenas podem ser comparadas células de mesmo tamanho e que se sobrepõem perfeitamente, ou seja, assinaturas que tenham a mesma resolução. Para estar de acordo com estes requisitos, a geração de grades deve seguir um padrão pré-definido. Se estes

requisitos não forem atendidos, se torna impossível comparar duas assinaturas, como apresentado na Figura 5.a. Portanto, o espaço deve ser dividido em células independente da posição do objeto. Em outras palavras, existe uma grade universal, e o sistema de coordenadas define a grade. Um algoritmo especificando um padrão para computar assinaturas *raster* é proposto em ZIMBRAO e SOUZA (1998), e é apresentado nesta subseção.

Os requisitos para construção de grades de células podem ser atendidos se nós restringirmos que o tamanho do lado de cada célula seja potência de dois (2^n), e que os vértices de cada célula sejam múltiplos da mesma potência de dois ($a \times 2^n$) no sistema de coordenadas. Dessa forma, nós garantimos que, se duas células de mesmo tamanho se sobrepõem, então elas estão perfeitamente sobrepostas uma na outra (Figura 5.b).

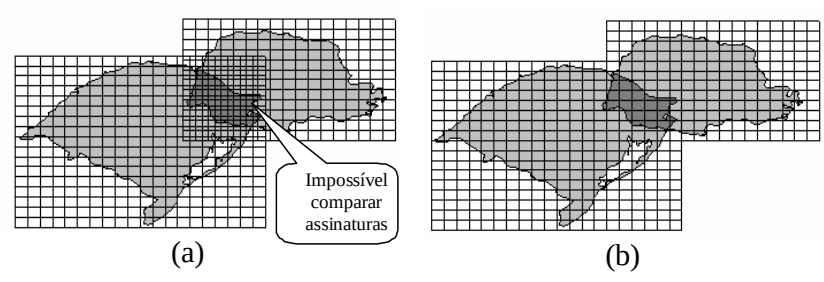


Figura 5. (a) Grades com células do mesmo tamanho, mas que não se sobrepõem perfeitamente, (b) sobreposição perfeita de células, permitindo a comparação das mesmas.

O espaço é dividido em $p \times q$ células com tamanho 2^n, o qual corresponde ao MBR-2^n (*Minimum Bounding Rectangle* ou Retângulo Mínimo Envolvente) do objeto. Em outras palavras, o MBR-2^n corresponde ao MBR cujas coordenadas são múltiplas de potência de dois e que define um espaço de $p \times q$ células com tamanho 2^n. Os vértices do MBR-2^n são ($2^n a_0$, $2^n b_0$) e ($2^n a_p$, $2^n b_q$), onde a_0, a_p, b_0, b_q e n são números inteiros. Além disso, n é escolhido tal que $(a_p - a_0)(b_q - b_0) \leq N$, onde $(a_p - a_0)$ é o número de células no eixo x, $(b_q - b_0)$ é o número de células no eixo y e N é o número máximo de células da grade. Como as assinaturas 4CRS são armazenadas nas folhas das árvores-R a serem utilizadas como índice no processamento de consultas espaciais, N é escolhido de forma que o tamanho médio das assinaturas resulte em uma árvore que produza bons resultados. Uma boa escolha é tentar manter o tamanho da assinatura perto de 3 ou 4 vezes do tamanho do MBR. Por exemplo, se cada entrada utiliza em média 80 *Bytes*,

uma página com 16 KB acomodará 100 ou 200 entradas e um grande conjunto de dados (1000 K objetos) leva a uma Árvore-R de apenas 3 ou 4 níveis.

O MBR-2^n é computado baseado no MBR do objeto, truncando suas coordenadas para potências de 2. A Figura 6 apresenta um algoritmo para computar MBR-2^n de um objeto.

```
algoritmo computarMBR2n(mbr, n, a0, ap, b0, bq )
   n := 0;  //2^0 é o tamanho mínimo de célula
   enquanto verdadeiro faça
     a0 := mbr.xmin / 2^n ;
     ap := mbr.xmax / 2^n ;

     b0 := mbr.ymin / 2^n ;
     bq := mbr.ymax / 2^n ;
     se (ap - a0)( bq - b0)≤ N
         retornar;
     senão
         n := n + 1;
```

Figura 6. Algoritmo para computar MBR-2^n.

A grade de células é representada pelos pontos $2^n a_0, 2^n a_1, \ldots, 2^n a_p$, que definem um conjunto de linhas paralelas ao eixo vertical, e os pontos $2^n b_0, 2^n b_1, \ldots, 2^n b_q$, que definem um conjunto de linhas paralelas ao eixo horizontal. Um exemplo de definição das coordenadas de células é apresentado na Figura 7.

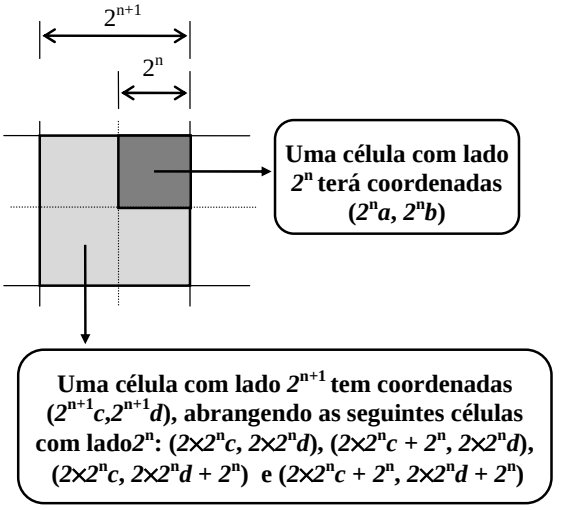


Figura 7. Exemplo de definição das coordenadas de células (ZIMBRAO e SOUZA, 1998).

Ao processar a consulta sobre duas assinaturas 4CRS, é essencial que ambas tenham a mesma resolução, ou seja, o mesmo tamanho de célula. Se isto não ocorrer, é necessário realizar uma mudança de escala, a qual é realizada agrupando-se células da assinatura com menor tamanho do lado de célula. O algoritmo para mudança de escala avalia a média da soma de valores numéricos atribuídos para cada tipo de célula, o qual representa o percentual de área do polígono dentro da célula. Para células *Vazio* e *Cheio* os valores numéricos são 0% e 100%, respectivamente, já que estes valores representam o percentual exato de área de interseção entre a célula e o polígono. Como neste trabalho estamos tratando de processamento aproximado, propomos usar como média percentual de área do polígono dentro de células *Pouco* e *Muito* os valores de 25% e 75%, respectivamente. Estes valores representam a área esperada do polígono dentro de células destes tipos. A Subseção 4.5.1 apresenta detalhes sobre o cálculo da área esperada do polígono dentro da célula.

3.2. Definições Preliminares

Esta seção apresenta alguns conceitos que são importantes para o entendimento da nossa proposta do novo algoritmo para geração de assinaturas 4CRS, o qual é apresentado na Seção 3.3.

Definição 1) Direção de ciclo

A definição de polígono usado neste trabalho segue a definição proposta por GÜTING e SCHNEIDER (1995) para região (aqui denominado polígono). Um polígono é capaz de representar um conjunto de áreas disjuntas (chamadas de faces) que podem conter buracos. O conjunto de pontos que definem uma face ou um buraco é chamado de ciclo. A direção do ciclo define onde se localiza a parte interna do polígono. Se nós seguirmos os segmentos do ciclo na direção dada pela sua lista de pontos, nós teremos que a parte interna do polígono, para cada segmento, se localiza sempre do mesmo lado (esquerda ou direita). Um ciclo no qual a parte interna do polígono está do lado esquerdo é chamado de anti-horário, enquanto que o ciclo cuja parte interna do polígono está do lado direito é chamado de horário. A Figura 8 apresenta exemplos de ciclos com direção horária e anti-horária.

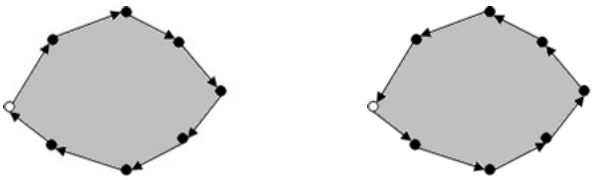


Figura 8. Exemplos de ciclos horário e anti-horário.

Definição 2) Atributo *DentroAcima* de um segmento

O atributo *DentroAcima* de um segmento tem valor *verdadeiro* quando a área interna do polígono se localiza acima do segmento ou, no caso do segmento ser uma linha vertical, o valor *verdadeiro* para este atributo indica que a área interna do polígono está à esquerda do segmento. A Figura 9 apresenta um exemplo de um polígono com os atributos *DentroAcima* de seus segmentos ajustados da seguinte forma:

- *v* (*verdadeiro*): a área interna do polígono localiza-se acima ou à esquerda do segmento;
- *f* (*falso*): a área interna do polígono localiza-se abaixo ou à direita do segmento.

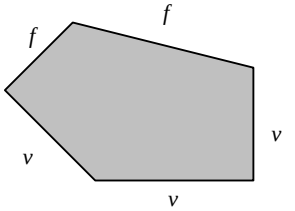


Figura 9. Exemplo de um polígono com os atributos *DentroAcima* de seus segmentos ajustados.

O algoritmo para ajustar o atributo *DentroAcima* de um segmento é apresentado na Figura 10. Na Figura 11, nós apresentamos um exemplo de execução do algoritmo para ajustar o atributo *DentroAcima* de um segmento formado pelos pontos p_1 e p_2 de um ciclo. Considere que estes pontos apresentam as seguintes características: p_1 vem antes de p_2 de acordo com a direção dada seguindo a lista de pontos do ciclo; p_1 é menor do que p_2, ou seja, $p_1.x_1 < p_2.x_2 \vee (p_1.x_1 = p_2.x_2 \wedge \ p_1.y_1 < p_2.y_2)$; e p_1 se localiza na espaço corresponde à região destacada na Figura 11.a. Dessa forma, nós podemos ajustar o valor do atributo *DentroAcima* do segmento de acordo com a direção do ciclo ao qual ele pertence. Se o ciclo é horário, então a parte interna do polígono está sempre localizada à direita do segmento, e como p_1 é menor do que p_2, a parte interna do polígono se localizará sempre abaixo ou à direita do segmento. Conseqüentemente o atributo *DentroAcima* receberá valor *falso* (Figura 11.b). Por outro lado, se o ciclo for anti-horário, a parte interna do polígono está sempre à esquerda do segmento, e o atributo *DentroAcima* será ajustado como *verdadeiro* (Figura 11.c).

```
boolean ajustarDentroAcima (segmento, direcaoDoCiclo)
pre-condição: segmento.p1 vem antes de segmento.p2 seguindo a direção
dada pela lista de pontos do ciclo.
  se ( ( ( segmento.p1 < segmento.p2 ) e
         ( direcaoDoCiclo é horária ) ) ou
       ( ( segmento.p1 > segmento.p2 ) e
         ( direcaoDoCiclo é anti-horária) ) )
     retornar falso;
  senão
     retornar verdadeiro;
```

Figura 10. Algoritmo para ajustar o atributo *DentroAcima* de um segmento.

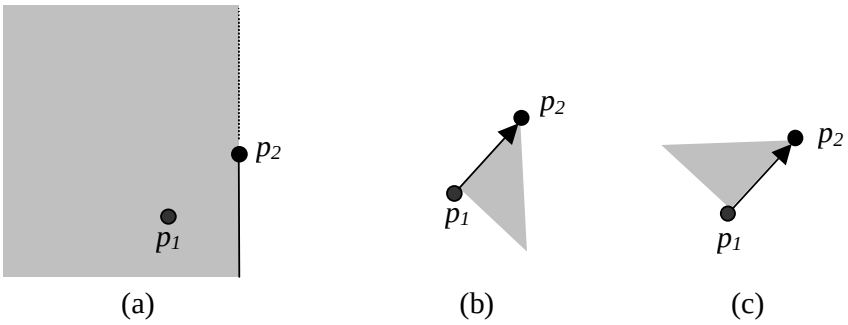


Figura 11. Exemplo de execução do algoritmo para ajustar atributo *DentroAcima* de um segmento.

Definição 3) *Turning Point*

O conceito de *Turning Point* foi proposto por LIANG e BARSKY (1983) e apontado por MAILLOT (1992). *Turning Point* é definido como o ponto de interseção de duas arestas de um polígono sendo recortado (durante uma operação de *clipping*) que

deve ser adicionado ao polígono resultante a fim de manter a conectividade do polígono original. A Figura 12.a apresenta um exemplo de polígono sendo recortado por uma janela retangular. Com o intuito de manter a conectividade do polígono original é necessário considerar as arestas correspondentes aos *turning points* destacados na Figura 12.b. Neste trabalho nós estendemos a definição de *turning point*, adicionando-lhe um atributo chamado *Direção* que especifica a direção para onde se encontra a área interna do polígono sobre a aresta onde o *turning point* se localiza (*Esquerda*, *Direita*, *Superior* ou *Inferior*), como apresentado na Figura 12.c. O polígono resultante do recorte é apresentado na Figura 12.d.

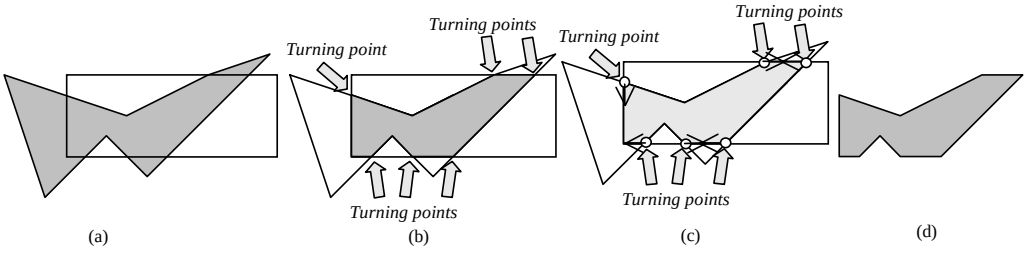


Figura 12. Exemplo de *turning points*.

3.3. Algoritmo para Geração de Assinatura 4CRS

Esta seção apresenta um algoritmo eficiente para geração de assinaturas 4CRS de polígonos. O algoritmo tem como entrada o conjunto de segmentos do polígono, e produz a assinatura 4CRS do polígono.

O algoritmo aqui proposto e apresentado na Figura 13 percorre os segmentos do polígono computando, para cada célula interceptada pelo segmento, a soma das áreas dos trapezóides formados pela parte do segmento dentro da célula e as arestas da célula. O valor do atributo *DentroAcima* do segmento é usado para determinar se a área dos trapezóides correspondentes ao segmento serão consideradas com valor positivo ou negativo. Após calcular a área do polígono dentro das células atravessadas por seus segmentos, procede-se com a execução de um algoritmo de *flood fill*, durante a qual as células não interceptadas pelos segmentos são classificadas em *Vazio* (células fora do polígono) e *Cheio* (células dentro do polígono), enquanto que as demais células, atravessadas por seus segmentos, são classificadas em *Pouco* ou *Muito*, de acordo com o percentual de área do polígono dentro da célula (Tabela 1).

```
Assinatura4CRS gerar4CRS(poligono)
  grade4CRS = obterMBR2n(poligono);
  para cada segmento s de poligono.segmentos faça
    para cada celula do grade4CRS interceptado por s faça
      se (SutherlandCohenLineClipping(s, segmentoRecortado,
          pontoDeIntersecao, ehPontoDeIntersecao, celula))
        se (nao ehPontoDeIntersecao)
          area = computarArea(trapezoide(segmentoRecortado,celula));
          se (segmento.DentroAcima)
            celula.area+=area;
          senão
            celula.area-=area;
  grade4CRS.floodFill();
```

Figura 13. Algoritmo para geração de assinatura 4CRS de polígono.

Neste trabalho estamos empregando o algoritmo proposto por Sutherland e Cohen para recorte de segmento, descrito por NEWMAN e SPROULL (1979). Este algoritmo foi escolhido por se provavelmente o método mais eficiente em casos de aceitação e rejeição triviais, os quais são os casos mais freqüentemente encontrados em recorte por janela. Este algoritmo pode ser implementado usando aritmética inteira ou de ponto flutuante; portanto, cobrindo um amplo conjunto de aplicações (MAILLOT, 1992).

O algoritmo proposto neste trabalho não assume uma orientação específica para o conjunto de segmentos do polígono, não requer execução de nenhuma operação de ordenação prévia dos segmentos, não está baseado na computação de números de paridade segundo um ponto de referência. Além disso, o algoritmo é capaz de tratar polígonos com múltiplas faces com buracos. O recorte de um segmento é completamente independente do recorte de outro segmento. Dessa forma, é possível empregar uma implementação paralela. A única pré-condição é que os segmentos tenham o atributo *DentroAcima* ajustado. Este atributo é utilizado para tratar *turning points* e para determinar se a área do trapezóide correspondente a um segmento deve ser considerada com valor positivo ou negativo.

3.3.1. Cálculo da Área do Polígono dentro da célula

O cálculo da área do polígono dentro de cada célula é realizado separadamente para cada segmento, de acordo com o trapezóide formado pela parte do segmento dentro da célula e as partes das arestas da célula que estão abaixo do segmento. O valor do atributo *DentroAcima* do segmento define se a área resultante será adicionada à área total do polígono dentro da célula ou será deduzida da mesma. Estamos assumindo, sem perda de generalidade, que o atributo *DentroAcima* com valor *falso* determina que a

área do trapezóide deve ser considerada com valor positivo, e valor *verdadeiro* do atributo *DentroAcima* define que a área deve ser considerada com valor negativo.

Considere o exemplo apresentado na Figura 14, na qual a célula *c* é atravessada pelo segmento *s* (Figura 14.a). A parte do segmento dentro da célula e as partes das arestas esquerda, direita e inferior da célula cobertas pelo segmento *s* definem o trapezóide cuja área será calculada e somada (atributo *DentroAcima* com valor *falso*) à área total do polígono dentro da célula.

(a) (b)

Figura 14. Exemplo de trapezóide correspondente a um segmento que atravessa a célula.

Considere outro exemplo, apresentado na Figura 15.a, no qual a célula *c* é atravessada por dois segmentos s_1 e s_2. Suponha que, durante a execução do algoritmo, o segmento s_1 é considerado primeiro do que o segmento s_2. Dessa forma, a área do trapezóide destacado na Figura 15.b é adicionada à área total do polígono dentro da célula. Em seguida, o trapezóide definido pelo segmento s_2 é gerado (Figura 15.c) e sua área é subtraída da área total do polígono dentro da célula. Logo, a área total resultante é igual a área do polígono apresentado na Figura 15.d. Finalmente, a célula *c* pode ser classificada de acordo com o percentual de área correspondente a fração da área calculada para o polígono dentro da célula divida pela área da célula. O mesmo resultado poderia ser alcançado se tivéssemos considerado inicialmente o segmento s_2. Primeiro nós computaríamos a área do trapezóide apresentado na Figura 15.c, que adicionaria área negativa a área total do polígono dentro da célula, e então calcularíamos a área do trapezóide mostrado na Figura 15.b, que seria somada a área total da célula. Como resultado a área obtida corresponderia novamente à área do trapezóide apresentado na Figura 15.d. Em outras palavras, não é necessária uma ordem específica para o tratamento dos segmentos do polígono, nem a execução de nenhuma operação de ordenação dos seus mesmos.

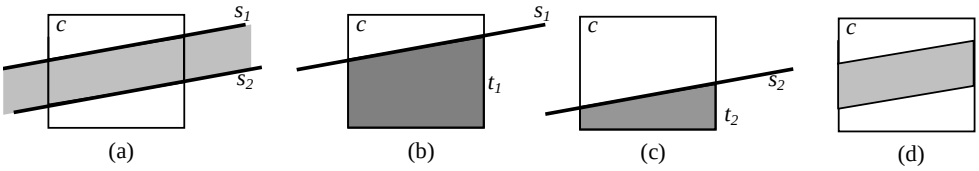


Figura 15. Exemplo de célula atravessada por dois segmentos.

Existem alguns casos especiais que devem ser considerados:

- Segmento completamente dentro da célula;
- Segmento parcialmente dentro da célula; e,
- O segmento mais acima tem atributo *DentroAcima* com valor igual a *verdadeiro*.

Quando o segmento está completamente dentro da célula como apresentado na Figura 16.a (segmento *s* dentro da célula *c*), o segmento não atravessa nem a aresta direita nem a aresta esquerda da célula, e a aresta inferior da célula não é completamente coberta pela área do polígono abaixo do segmento *s*. Dessa forma, o trapezóide cuja área será adicionada a área do polígono dentro da célula (apresentado na Figura 16.b) é formado pelos segmentos: segmento *s*; segmentos verticais definidos pelos pontos extremos de *s* e os pontos sobre a aresta inferior da célula (segmentos *l* e *r*); e, a parte da aresta inferior correspondente à projeção do segmento *s* sobre a mesma (segmento *b*).

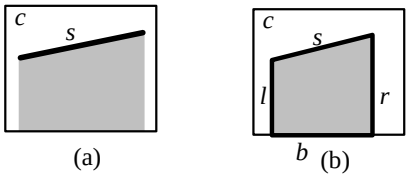


Figura 16. Exemplo de segmento completamente dentro da célula.

O caso do segmento estar parcialmente dentro da célula (Figura 17.a) é semelhante ao caso de segmento completamente dentro da célula. A única diferença é que, para o cálculo da área do polígono dentro da célula, a parte de uma das arestas verticais da célula será considerada como parte do trapezóide (Figura 17.b). Dessa forma, no caso do exemplo apresentado na Figura 17.b, o segmento correspondente ao ponto extremo direito de *s* e o ponto sobre a aresta inferior não é considerado para definição do trapezóide cuja área será calculada.

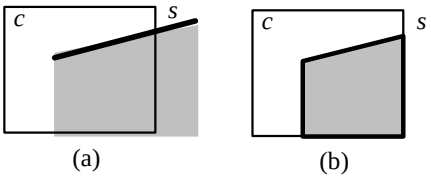


Figura 17. Exemplo de segmento parcialmente dentro da célula.

Um outro caso interessante ocorre quando o segmento mais superior dentro da célula tem atributo *DentroAcima* com valor igual a *verdadeiro*, como, por exemplo, apresentado na Figura 18.a. O trapezóide cuja área será subtraída da área do polígono dentro célula é apresentado na Figura 18.b, o qual define uma área negativa para a área do polígono dentro célula. Neste caso, para que a área da célula seja atribuída corretamente, temos que considerar também a área do retângulo (Figura 18.c) formado por: segmento correspondente à projeção do segmento *s* sobre a aresta superior da célula, denominado s_p; segmentos verticais definidos pelos pontos extremos de s_p e os pontos sobre a aresta inferior da célula (segmentos *l* e *r*); e a parte da aresta inferior que correspondente à projetção do segmento s_p sobre a mesma (segmento *b*). Desse modo, a área final corresponde ao trapezóide apresentado na Figura 18.d. Os retângulos construídos a partir da projeção de segmentos mais superiores sobre as arestas superiores das células são identificados pela análise dos *turning points* (Definição 3) das células, os quais são armazenados para cada aresta de célula, durante o processamento dos segmentos do polígono para cálculo da área do polígono dentro da célula.

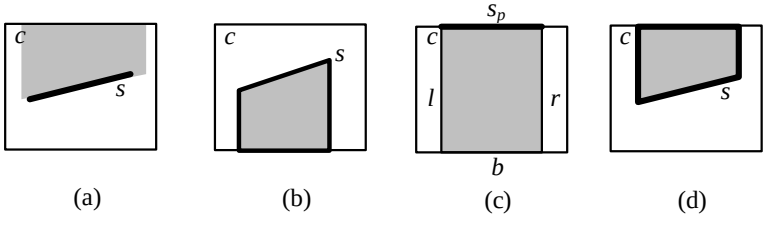


Figura 18. Exemplo de segmento superior com atributo *DentroAcima* igual a *verdadeiro.*

Dessa forma, o cálculo da área do polígono dentro da célula deve considerar tanto os trapezóides resultantes das partes dos segmentos do polígono dentro da célula, como também os retângulos correspondentes aos segmentos formados pelos pontos resultantes do recorte dos segmentos do polígono que se localizam sobre as arestas das células, chamados de *turning points* (Definição 3).

Considere o exemplo, apresentado na Figura 19, para classificação da célula *c* de acordo com a área do polígono *p* dentro da mesma. Durante o recorte dos segmentos do

polígono pelas arestas das células, armazenam-se os pontos que se localizam sobre a sua aresta superior (p_1 e p_2 - *turning points*) e para qual direção a área do polígono se encontra a partir de cada ponto. No caso de p_1, a área do polígono está para a direita, e, no caso de p_2, a área do polígono está para a esquerda (Figura 19.b). O segmento sobre a aresta superior da célula é obtido ligando-se os pontos consecutivos que têm direções opostas. Sendo assim, no exemplo apresentado na Figura 19, as partes dos segmentos do polígono contidos na célula determinarão trapezóides com área negativa (Figura 19.c) e o segmento sobre a aresta superior da célula determinará um retângulo com área positiva (Figura 19.d). Como resultado, a área final será igual à área correspondente ao polígono apresentado na Figura 19.e.

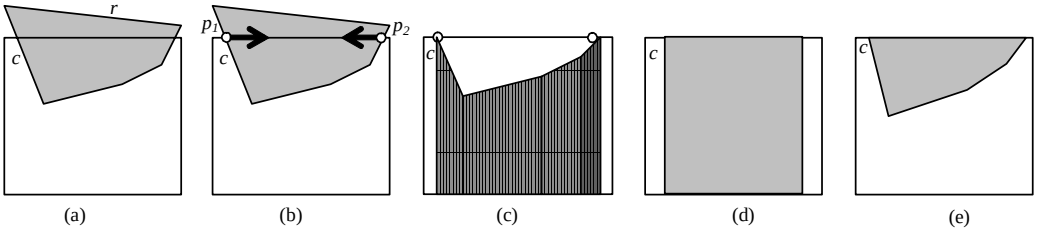


Figura 19. Exemplo de classificação de célula.

Após calcular a área do polígono dentro das células atravessadas por seus segmentos, procede-se com a execução de um algoritmo de *flood fill*, durante a qual as células não interceptadas pelos segmentos são classificadas em *Vazio* (células fora do polígono) e *Cheio* (células dentro do polígono), enquanto que as demais células, atravessadas por seus segmentos, são classificadas em *Pouco* ou *Muito*, de acordo com o percentual de área do polígono dentro da célula.

4. Processamento Aproximado de Consultas Usando Assinatura *Raster* de Quatro-Cores

A assinatura 4CRS (ZIMBRAO e SOUZA, 1998) foi inicialmente empregada como filtro no segundo passo do MSQP (*Multi-Step Query Processor*) (BRINKHOFF *et al*., 1994) no processamento de junções espaciais envolvendo conjuntos formados por polígonos. O objetivo foi reduzir o número de testes de geometria exata executados no último passo da arquitetura MSQP. Os bons resultados demonstraram a eficiência da assinatura 4CRS perante outras abordagens da literatura. No processamento da junção espacial, o número de respostas inconclusivas foi reduzido por um fator maior do que dois. Como resultado, a necessidade de ler do disco as representações dos polígonos e executar os testes de geometria exata foram reduzidos por um fator maior do que dois. A assinatura 4CRS também foi usada para testes de interseção entre conjuntos de polígonos e polilinhas (MONTEIRO *et al*., 2004) e os resultados obtidos também foram muito bons. Os experimentos realizados com dados reais demonstraram ganhos validando a eficiência da assinatura 4CRS. O número de testes de interseção exata foi reduzido em 59%. Em outras palavras, para 59% dos pares de objetos que têm interseção de MBR provenientes do primeiro passo, foi possível obter uma resposta processando-se apenas as assinaturas, sem executar o teste das geometrias exatas dos polígonos. Os totais de tempo de execução e número de acessos a disco foram ambos reduzidos na ordem de 48% em relação ao tempo de execução e número de acessos a disco necessários para processar a consulta sem utilizar as assinaturas durante o processamento.

As características da assinatura 4CRS e os bons resultados obtidos empregando-a como filtro geométrico no processamento de junção de polígonos motivaram o seu uso para processamento aproximado de consultas. Nesta nova abordagem, o objetivo é retornar para o usuário uma resposta aproximada que é obtida processando a consulta diretamente sobre as assinaturas 4CRS dos polígonos, sem acessar as representações reais dos objetos. Sendo assim, o passo de refinamento correspondente ao teste das geometrias exatas não é executado, o qual se constitui no passo mais custoso no processamento da consulta. Conseqüentemente, novos algoritmos devem ser projetados, implementados e avaliados para estarem de acordo com os requisitos dessa nova abordagem.

Este capítulo está dividido da seguinte forma. A Seção 4.1 apresenta a definição relacionada ao uso da Assinatura *Raster* de Quatro-Cores (4CRS) para processamento aproximado de consultas. A Seção 4.2 enumera e classifica um conjunto de operações que podem ser processadas de forma aproximada. As seções 4.3, 4.4 e 4.5 mostram detalhes de propostas de três algoritmos para processamento aproximado de consultas que foram implementados e avaliados neste trabalho. Fórmulas para calcular os intervalos de confiança das respostas retornadas por estes algoritmos são apresentadas na Seção 4.6. Finalmente, na Seção 4.7 são apontadas direções para projeto, implementação e avaliação dos algoritmos enumerados na Seção 4.2 para processamento aproximado de consultas.

4.1. Definição

O objetivo do processamento aproximado de consultas é prover respostas estimadas em um período de tempo muito menor do que o necessário para computar uma resposta exata, evitando ou minimizando o número de acessos a disco para ler os dados reais (GIBBONS *et al*., 1997). Outro importante aspecto relacionado ao processamento aproximado de consultas são técnicas para redução de dados, isto é, métodos que são usados para representar os dados de forma reduzida, armazenando as características mais importantes dos dados em uma representação concisa. GIBBONS *et al*. (1997) chamam esta representação dos dados como estruturas de sinopses dos dados. Como exemplo de sinopses, eles sugerem histogramas e amostras de grandes relações em ambientes de *data warehouse*.

Este trabalho diz respeito ao tratamento de grandes volumes de dados espaciais formados por polígonos. Nossa proposta é usar a assinatura 4CRS como estrutura de sinopse de polígonos, e processar a consulta sobre as assinaturas ao invés de acessar as representações reais dos objetos. Dessa forma, as geometrias exatas dos objetos não são processadas durante a execução da consulta, o qual é o passo mais custoso da junção espacial já que requer a busca e transferência de grandes objetos do disco para memória principal (BRINKHOFF *et al.*, 1994; LO e RAVISHANKAR, 1996). Além disso, o processamento exato emprega algoritmos complexos com elevado tempo de UCP para decidir se objetos atendem os qualificadores das consultas (BRINKHOFF *et al.*, 1993).

A Assinatura 4CRS pode ser usada para prover respostas rápidas e aproximadas para uma ampla classe de consultas. Nossa proposta está baseada na arquitetura

proposta por GIBBONS *et al*. (1997) para processamento aproximado de consultas em ambientes de *data warehouse*. Todavia, neste trabalho estamos focados no processamento aproximado de consultas envolvendo dados espaciais.

Em um ambiente para processamento de consultas de um SGBDE tradicional, consultas dos usuários são enviadas ao banco de dados que as processa e retorna uma resposta exata para o usuário. Na atualização do banco de dados, novos dados podem ser inseridos, ou dados existentes atualizados, ou mesmo excluídos da base de dados.

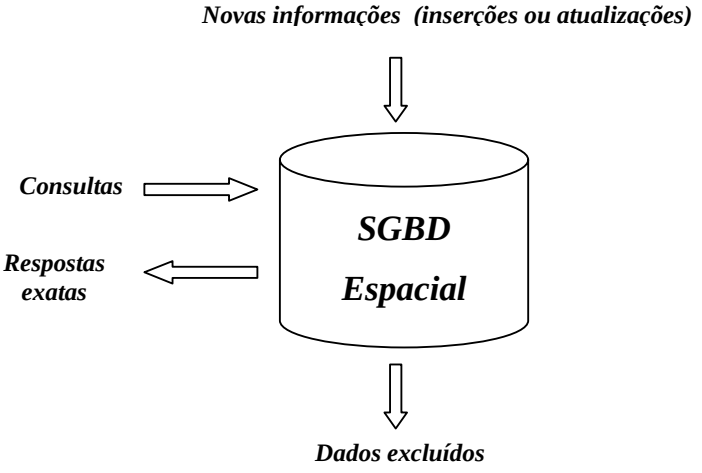


Figura 20. Ambiente para processamento de consultas em SGBDE tradicional.

Por outro lado, em uma arquitetura de SGBDE para prover respostas aproximadas para consultas, um novo componente é adicionado, o módulo para processamento aproximado (Figura 21). Nesta nova arquitetura, consultas são enviadas diretamente para o módulo de processamento aproximado, o qual é responsável por processar a consulta e retornar uma resposta aproximada para o usuário, juntamente com um intervalo de confiança que mostra a precisão da consulta. Se a precisão não é suficiente para o usuário tomar sua decisão, a consulta pode ser então processada pelo SGBDE, provendo uma resposta exata para o usuário.

O módulo de processamento aproximado armazena representações reduzidas dos dados reais para realizar o processamento aproximado. Portanto, é possível executar a consulta parcialmente sobre dados aproximados e parcialmente sobre dados reais como, por exemplo, quando não é garantido que as sinopses dos objetos sejam capazes de produzir a precisão desejada ou quando as representações reais dos mesmos são muito simples e têm baixo custo de processamento. Por exemplo, o cálculo da área exata de

um polígono com poucos vértices pode ser executado no mesmo tempo que o processamento aproximado sobre assinaturas 4CRS.

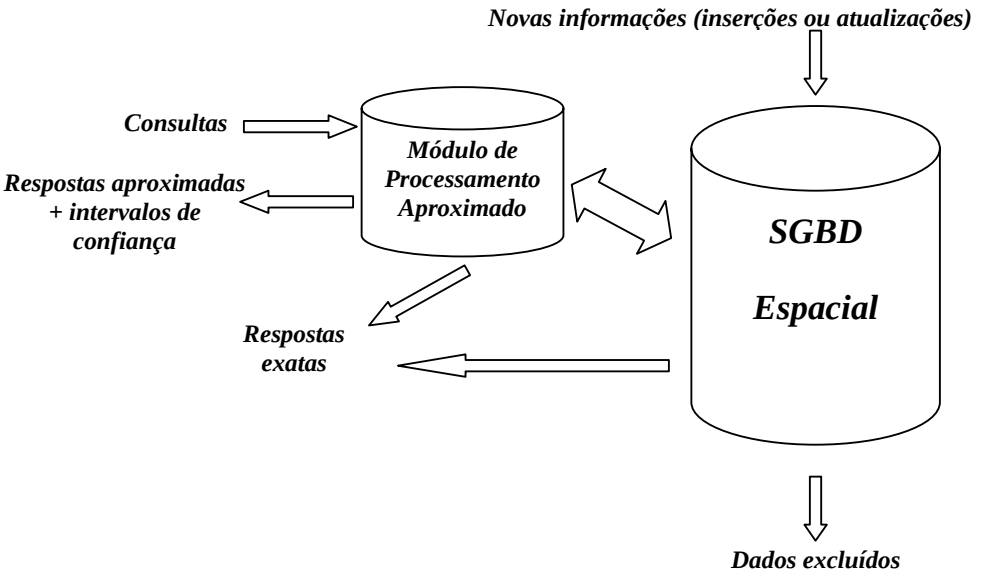


Figura 21. Arquitetura de SGBDE para prover respostas aproximadas para consultas.

Uma questão importante relacionado ao módulo de processamento aproximado é a manutenção dos dados. Quando novos dados chegam ou dados são atualizados, é também necessário armazenar as representações dos novos objetos no módulo ou atualizar as sinopses existentes. Dessa forma, é importante que as sinopses sejam computadas rapidamente. No caso de remoção de objetos, as sinopses devem ser removidas também da base de dados. Sendo assim, todas as informações que chegam ou saem do SGBDE têm que ser enviadas para o módulo de processamento aproximado a fim de mantê-lo atualizado.

4.2. Operações Aproximadas

Existem muitas operações que podem se beneficiar de um processamento rápido e aproximado a fim de que o usuário tenha respostas em um período curto de tempo, ao invés de ficar esperando muito tempo por uma resposta exata. Neste trabalho, apresentamos nossas propostas de algoritmos para processamento aproximado baseados na classificação de operações espaciais sugerida por GÜTING *et al.* (1995) e GÜTING e SCHNEIDER (1995) para a Álgebra *Rose*. Neste trabalho, apresentamos direções para pesquisa e implementação de algoritmos para processamento aproximado para as

operações que podem ser processadas de forma aproximada usando assinaturas 4CRS. Nesta seção nós enumeraremos estas operações.

4.2.1. Operações Espaciais que Retornam Números

- Área: operação para computar a área de polígonos que podem ser formados por múltiplas faces e conter buracos. Neste trabalho nós estendemos este conceito para incluir tanto área de polígono como também área de polígono dentro de janela e área de interseção de polígonos. Nós propomos algoritmos para processamento aproximado destas operações. Eles são descritos nas seções 4.3, 4.4 e 4.5, respectivamente. Na Seção 4.6, apresentamos propostas para calcular o intervalo de confiança das respostas retornadas na execução destes algoritmos. Além disso, nós implementamos e executamos testes experimentais dos mesmos, e os resultados são apresentados no Capítulo 5.

- Número de componentes: Este operador retorna o número de componentes de um valor espacial, por exemplo, número de vértices e número de faces de um polígono. Estes valores podem ser armazenados juntamente com o objeto (ou sua assinatura 4CRS). Além disso, como é uma tarefa simples obter estas informações e o custo de armazenamento é baixo, esta operação não necessita de processamento aproximado.

- Distância: retorna a distância mínima entre dois polígonos, uma proposta para processamento aproximado é apresentada na Subseção 4.7.1.

- Diâmetro: o diâmetro de um objeto espacial é definido como a maior distância entre qualquer um de seus componentes. A Subseção 4.7.2 propõe um algoritmo para esta operação.

- Comprimento: o operador comprimento calcula o comprimento de todos os segmentos de um objeto linhas (formado apenas por segmentos). Como este trabalho não trata de linhas, nós não propomos algoritmo para esta operação.

- Perímetro: calcula a soma dos comprimentos de todos os ciclos de uma região (ou polígono). Se estivermos interessados em computar apenas a soma dos comprimentos dos ciclos mais externos, não incluindo buracos, nós podemos usar a operação contorno para eliminar os buracos primeiro. A

Subseção 4.7.3 sugere um algoritmo para computar o perímetro aproximado de polígono.

4.2.2. Predicado Espacial

Estas operações comparam os valores espaciais de dois objetos em relação aos seus relacionamentos topológicos e retornam um valor booleano (verdadeiro ou falso). Devido à simplicidade de suas descrições nós apenas enumeramos as operações e apontamos para as subseções onde são apresentados os algoritmos para processamento aproximado.

- Igual (=) (Subseção 4.7.4)
- Diferente (≠) (Subseção 4.7.4)
- Disjunto (Subseção 4.7.5)
- Dentro (Subseção 4.7.6)
- Área disjunto (Subseção 4.7.5)
- Aresta disjunto (Subseção 4.7.5)
- Aresta dentro (Subseção 4.7.6)
- Vértice dentro (Subseção 4.7.6)
- Intercepta (Subseção 4.7.7)
- Toca: operação que retorna se dois segmentos têm exatamente uma extremidade em comum. Como neste trabalho estamos lidando com polígonos e não segmentos, não propomos um algoritmo para processamento aproximado para esta operação.
- Adjacente (Subseção 4.7.9)
- Existe borda em comum (Subseção 4.7.9).

4.2.3. Operações que Retornam Valores Espaciais

Este grupo de operações é formado pelos operadores que retornam valores espaciais únicos.

- Interseção (Subseção 4.7.7): retorna o polígono resultante da interseção de dois polígonos.

- Soma (ou união) (Subseção 4.7.10): retorna o polígono resultante da união de dois polígonos.
- Subtração (Subseção 4.7.11): retorna o polígono resultante da diferença de dois polígonos.
- Borda em comum (Subseção 4.7.9): retorna a borda em comum a dois polígonos.
- Vértices: retorna os vértices de um polígono. Esta operação pode ser executada apenas sobre os objetos reais, já que assinaturas 4CRS não armazenam informações sobre vértices.
- Contorno (Subseção 4.7.13): retorna o contorno (ou borda) de um polígono.
- Interior: O operador *interior* é aplicado a um objeto formado apenas por segmentos (objeto *linhas*) e produz um polígono formado por todas as faces que são circundadas pelos segmentos do objeto. Como este operador processa linhas e não polígonos, nós não o abordaremos neste trabalho.

4.2.4. Operadores Espaciais Aplicados sobre Conjunto de Objetos

Operadores desta classe têm como entrada conjunto de objetos; alguns têm como saída um outro conjunto de objetos.

- Soma (Subseção 4.7.10): O operador *soma* computa a união geométrica de todos os valores de um determinado atributo em um conjunto de objetos.
- Objeto mais próximo (Subseção 4.7.14): O operador *objeto mais próximo* retorna o objeto de um conjunto que é mais próximo de um valor de referência.
- Decomposição (Subseção 4.7.15): O operador *decomposição* tem como entrada um conjunto de objetos que possui um atributo espacial. Ele produz uma nova coleção de objetos da seguinte forma: para cada objeto do conjunto operando seu atributo é decomposto em componentes (um componente pode ser um ponto, segmento, face, etc.). Se um objeto possui *n* componentes, então *n* cópias do objeto original são produzidas cada qual contendo um componente do objeto original como valor do seu atributo espacial.

- Overlay (Subseção 4.7.8): O operador *overlay* permite que uma partição do plano seja superposta sobre outra, de forma que elas sejam combinadas em regiões áreas disjuntas.
- Fusão (Subseção 4.7.12): O operador *fusão* mescla os valores de um dado (conjunto de) atributo(s) espacial(is) com base na igualdade dos valores de outro (conjunto de) atributo(s) não espacial(is). Para cada grupo de valores não espaciais um (conjunto de) novo(s) valor(es) espacial(is) é criado como sendo a união geométrica do conjunto de valores espaciais do grupo.

4.3. Área Aproximada de Polígono

Esta seção apresenta detalhes do algoritmo para computar área aproximada de polígono a partir de sua assinatura 4CRS. Este algoritmo e seus resultados foram descritos por AZEVEDO *et al.* (2004).

O conceito de área esperada de célula, o qual é empregado por este algoritmo, é apresentado na Subseção 4.3.1, enquanto que a Subseção 4.3.2 é dedicada à apresentação do algoritmo propriamente dito.

4.3.1. Área Esperada de Célula

Esta subseção apresenta o cálculo de área esperada de célula, cujo conceito é empregado pelo algoritmo para computar área aproximada de polígono.

É fácil observar que a área esperada correspondente a uma célula do tipo *Vazio* é igual a 0% (zero por cento), já que não existe nenhuma área do polígono dentro da célula. Da mesma forma, células do tipo *Cheio* têm 100% de área esperada, pois a célula é completamente ocupada pelo polígono. Portanto, é apenas necessário determinar a área esperada correspondente aos tipos de células *Pouco* e *Muito*. Neste trabalho nós assumimos que células *Pouco* e *Muito* têm área esperada igual a 25% e 75%, respectivamente. Estes valores podem ser usados porque a grade da assinatura e o polígono são independentes um do outro, e é esperado que a distribuição percentual de área de polígono dentro da célula seja muito próxima da distribuição uniforme. De fato, nós computamos a distribuição percentual de área de polígono dentro da célula para o conjunto de dados correspondente aos municípios de Iowa (Estados Unidos da América) em intervalos de 1% e os resultados indicaram que a suposição de distribuição uniforme realmente se aplica. Além disso, como a medida usada para computar o intervalo de

confiança é a variância, nós comparamos a variância de área de polígono dentro da célula assumindo distribuição uniforme com a variância computada. Assumindo distribuição uniforme, célula *Pouco* possui variância igual a $(0,5-0)^2/12 = 1/48 = 0,020833$, pois este tipo de célula têm distribuição no intervalo (0; 0,50]. Célula *Muito* tem a mesma variância, já que tem distribuição no intervalo (0,50; 1,0). Sua variância é igual a $(1,0-0,5)^2/12 = 1/48 = 0,020833$. Em nossos testes executados sobre o conjunto de dados municípios de Iowa (EUA), as variâncias computadas foram 0,021978 para células pouco e 0,021952 para células *Muito*, cujos valores são muito próximos das variâncias assumindo distribuição uniforme. Sendo assim, a área esperada empregada pelo algoritmo que computa a área aproximada de polígono é apresentada na Tabela 2.

Tabela 2. Área esperada de acordo com tipos de célula.

Tipo de célula	Área esperada
Vazio	0%
Pouco	25%
Muito	75%
Cheio	100%

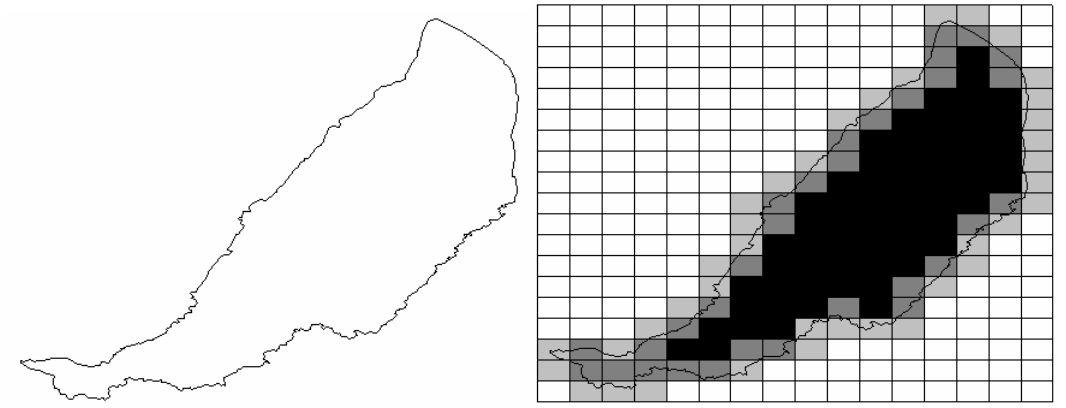
4.3.2. Algoritmo

O algoritmo para estimar a área de polígono avalia a soma da área esperada (Tabela 2) das células de sua assinatura 4CRS, e multiplica o valor resultante pela área da célula. O algoritmo é apresentado na Figura 22. O algoritmo emprega um vetor (areaEsperada) para armazenar os quatro valores de área esperada.

```
algoritmo areaAproximadaPoligono(assinatura4CRS)
  areaAproximada = 0;
  para cada celula c de assinatura4CRS.celulas faça
    areaAproximada += areaEsperada[c.tipo];
  areaCelula = assinatura4CRS.tamanhoDoLadoDaCelula   *
               assinatura4CRS.tamanhoDoLadoDaCelula;
  retornar areaAproximada * areaCelula;
```

Figura 22. Algoritmo para estimar área de polígono a partir de sua assinatura 4CRS.

Considere o exemplo apresentado na Figura 23, onde são mostrados um polígono, sua assinatura 4CRS e o cálculo aproximado de sua área. A área aproximada do polígono é igual 24.576.000, enquanto que a área exata é igual a 23.865.908, o que representa um erro de apenas 2,98% da área aproximada em relação à área exata. Na Seção 4.6.1, é apresentado uma proposta de cálculo para o intervalo de confiança para a resposta retornada por esta operação.



- Número de células *Vazio*: 210
- Número de células *Pouco*: 38
- Número de células *Muito*: 31
- Número de células *Cheio*: 61
- Área da célula: 262.144

Área aproximada = $(N_{Vazio} \times P_{Vazio} + N_{Pouco} \times P_{Pouco} + N_{Muito} \times P_{Muito} + N_{Cheio} \times P_{Cheio}) \times Área_{Célula}$

Área aproximada= (210 × 0 + 38 × 0,25 + 31 × 0,75 + 61 × 1) × 262.144 = 24.576.000

Área exata = 23.865.908

Erro no cálculo da área aproximada = 2,98%

Figura 23. Exemplo de cálculo de área aproximada de polígono.

4.4. Área Aproximada de Polígono Dentro de Janela

Esta seção é dedicada à apresentação do algoritmo para computar área aproximada de polígono dentro de janela a partir de sua assinatura 4CRS. Este algoritmo e seus resultados foram descritos por AZEVEDO *et al.* (2004).

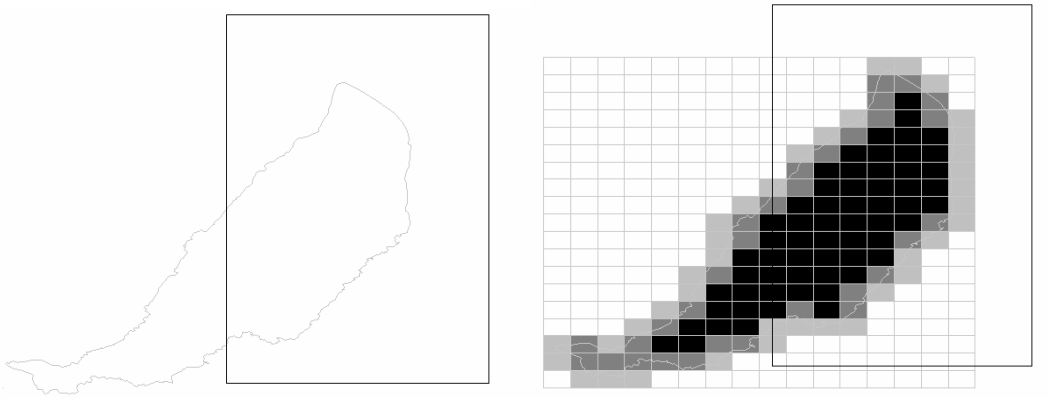
O algoritmo para estimar a área de polígono dentro de janela é muito semelhante ao algoritmo para estimar área de polígono, apresentado na Figura 22. Isto porque podemos considerar que a janela é uma grande célula *Cheio*, e que a interseção de uma célula *Cheio* com qualquer outro tipo de célula tem área esperada igual à área esperada do tipo de célula. A única diferença é que o polígono pode estar inteiramente contido na janela ou ser parcialmente sobreposto por ela. No primeiro caso, a área esperada do polígono dentro da janela é igual à área esperada do polígono. No segundo caso, temos que ter o cuidado de que algumas células são atravessadas pelas arestas da janela, e que apenas a parte da área da célula que está dentro da janela deve ser considerada. Neste caso, devemos multiplicar a área esperada da célula pelo fator correspondente a área da

célula dentro da janela dividida pela área da célula. O algoritmo é apresentado na Figura 24.

```
algoritmo areaAproximadaPoligonoJanela(assinatura4CRS,janela)
  areaAproximada = 0;
  areaCelula = assinatura4CRS.tamanhoDoLadoDaCelula   *
               assinatura4CRS.tamanhoDoLadoDaCelula;
  para cada celula c de assinatura4CRS.celulas dentro da janela faça
    areaAproximada += areaEsperada[c.tipo];
  para cada celula c de assinatura4CRS.celulas atravessadas pela
                                                        janela faça
    areaIntersecao = computarAreaIntersecao(c, janela);
    areaAproximada += areaEsperada[c.tipo] * areaIntersecao /
                                             areaCelula;
  retornar areaAproximada * areaCelula;
```

Figura 24. Algoritmo para computar a área aproximada de polígono dentro de janela a partir de sua assinatura 4CRS.

Considere o exemplo apresentado na Figura 25, onde são mostrados um polígono, sua assinatura 4CRS e o cálculo aproximado de sua área. A área aproximada do polígono é igual a 16.993.024, enquanto que a área exata é igual a 16.652.605, o que representa um erro de apenas 2,04% da área aproximada em relação à área exata. Na Seção 4.6.1, é apresentada uma proposta de cálculo para o intervalo de confiança para a resposta retornada por esta operação.



- Número de células *Vazio* dentro da janela: 64
- Número de células *Pouco* dentro da janela: 20
- Número de células *Muito* dentro da janela: 16
- Número de células *Cheio* dentro da janela: 44
- Número de células *Vazio* atravessadas pela janela: 8,31
- Número de células *Pouco* atravessadas pela janela: 1,53
- Número de células *Muito* atravessadas pela janela: 0,51
- Número de células *Cheio* atravessadas pela janela: 47,06
- Área da célula: 262.144

$$\text{Área aproximada} = (N_{Vazio} \times P_{Vazio} + N_{Pouco} \times P_{Pouco} + N_{Muito} \times P_{Muito} + N_{Cheio} \times P_{Cheio}) \times \text{Área}_{Célula}$$

$$\text{Área aproximada} = (72{,}31 \times 0 + 21{,}53 \times 0{,}25 + 16{,}51 \times 0{,}75 + 47{,}06 \times 1) \times 262.144 = 16.993.024$$

$$\text{Área exata} = 16.652.605$$

$$\text{Erro} = 2{,}04\%$$

Figura 25. Exemplo de cálculo de área aproximada de polígono dentro de janela.

4.5. Área Aproximada de Interseção de Polígono ×Polígono

Esta seção apresenta o algoritmo para computar a área aproximada de interseção de polígono × polígono a partir de suas assinaturas 4CRS. Este algoritmo e seus resultados foram descritos por AZEVEDO *et al.* (2005). A Subseção 4.5.1 apresenta o cálculo da área esperada de interseção de células, enquanto que a Subseção 4.5.2 mostra o algoritmo proposto.

4.5.1. Área Esperada de Interseção de Células

Nesta subseção, o cálculo de área esperada correspondente à sobreposição de tipos de células de mesmo tamanho é apresentado. A área esperada é empregada pelo algoritmo que computa a área aproximada de interseção de polígono × polígono, o qual é apresentado na Subseção 4.5.2.

É fácil observar que a área esperada correspondente à combinação de uma célula de tipo *Vazio* com qualquer outro tipo de célula resulta em área esperada igual a 0% (zero por cento). Analogamente, quando duas células do tipo *Cheio* se sobrepõem, a área esperada é de 100%. Dessa forma, nós temos que computar apenas as áreas esperadas para a interseção de outros tipos de células (*Pouco* × *Pouco*, *Pouco* × *Muito*, *Pouco* × *Cheio*, *Muito* × *Pouco*, *Muito* × *Muito*, *Muito* × *Cheio*), as quais estimamos pelo valor percentual médio das possíveis ocorrências de interseção entre duas células destes tipos.

Como os conjuntos de dados são razoavelmente independentes (por exemplo, não existe regra de que todos os limites de municípios devam ser definidos por cursos de rios), nós podemos assumir que as áreas esperadas (x_1 e x_2) correspondentes a duas células são também independentes. Assim, a área esperada pode ser computada como $x_1 \times x_2$. Como exemplo, considere que as áreas esperadas de polígono dentro de duas células *Pouco* são 10% e 15%, respectivamente. Dessa forma, a área esperada de

sobreposição entre estes tipos de células pode ser calculada como $0.01 \times 0.15 = 0.015$ (1.5%).

Entretanto, nós temos apenas informação a respeito do tipo de cada célula, o que significa que a área do polígono dentro da célula está em um determinado intervalo, como apresentado na Tabela 1. Dessa forma, a área correspondente à combinação (ou interseção) de células será estimada. Além disso, mesmo sabendo que a área é um valor contínuo, de modo a tornar mais simples a demonstração dos cálculos empregados, nós estamos assumindo que a área do polígono dentro da célula é computada como valores discretos, em passos de tamanho $1/n$ para n grande tendendo ao infinito ($n \rightarrow \infty$). Todos os valores são também tratados como percentuais.

Seja X uma variável randômica representando a área de interseção entre uma célula da grade e o polígono; $G(x_1, x_2)$ função que retorna a área de interseção entre dois tipos de células x_1 e x_2; e $p(x_1, x_2)$ a função de probabilidade conjunta das variáveis X_1 e X_2. A definição de média (ou valor esperado E) de duas variáveis X_1 e X_2 é apresentada na Equação 1 (LARSON, 1982).

$$\mathrm{E}[G(X_1, X_2)] = \sum_{x_1} \sum_{x_2} G(x_1, x_2) \times p(x_1, x_2) \ . \qquad (1)$$

Já que a área de interseção entre uma célula e o polígono é independente da área de interseção entre outra célula e o polígono, X_1 e X_2 são linearmente independentes e a função de probabilidade conjunta $p(x_1, x_2)$ pode ser expressa como $p(x_1, x_2) = p(x_1) \times p(x_2)$. Complementando, seja n o número dos possíveis valores observados para a área percentual do polígono dentro de célula. Então, as probabilidades $p(x_1)$ e $p(x_2)$ são iguais a $1/n$, já que cada valor de área de polígono dentro de célula tem mesma probabilidade de ocorrência. Além disso, a função $G(x_1, x_2)$ pode ser expressa pela multiplicação dos valores das áreas dos polígonos dentro das células. Portanto, para n interseções entre dois tipos de células, a área esperada $\mathrm{E}[G(x_1,x_2)]$ pode ser dada de forma aproximada pela Equação 2.

$$\mathrm{E}[G(X_1, X_2)] = \mu = \sum_{i=1}^{n} \sum_{j=1}^{n} \delta(x_1) \times \delta(x_2) \times p(x_1) \times p(x_2) \ , \qquad (2)$$

onde $\delta(x)$ é uma função que retorna o percentual de área do polígono dentro da célula. Esta função é expressa pelas equações 3 e 4, de acordo com o tipo de célula sendo avaliada.

$$\delta(k) = \frac{k}{2n}, \; 1 \le k \le n, \text{ para célula } Pouco. \tag{3}$$

$$\delta(k) = \left(\frac{k}{n} + \frac{1}{2}\right), \; 1 \le k \le n, \text{ para célula } Muito. \tag{4}$$

- **Área esperada de células *Pouco* × *Pouco***

$$\begin{aligned} &\mathrm{E}[G(Pouco_1, Pouco_2)] = \sum_{i=1}^{n}\sum_{j=1}^{n} \delta_i(pouco_1) \times \delta_j(pouco_2) \times p(pouco_1) \times p(pouco_2) = \\ &= \lim_{n\to\infty} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{i}{2n}\frac{j}{2n}\frac{1}{n}\frac{1}{n} = \lim_{n\to\infty} \frac{1}{4n^4}\sum_{i=1}^{n} i \sum_{j=1}^{n} j \; . \end{aligned} \tag{5}$$

Como a sequência aritmética $\sum_{k=1}^{n} k$ pode ser expressa como $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$, e usando a regra de L'Hôpital, a Equação 5 pode ser reescrita como apresentado na Equação 6.

$$\begin{aligned} &\mathrm{E}[G(Pouco_1, Pouco_2)] = \lim_{n\to\infty} \frac{1}{4n^4}\sum_{i=1}^{n} i \sum_{j=1}^{n} j = \lim_{n\to\infty} \frac{1}{4n^4}\sum_{i=1}^{n} i \frac{n(n+1)}{2} = \\ &= \lim_{n\to\infty} \frac{1}{4n^4}\frac{n(n+1)}{2}\frac{n(n+1)}{2} = \frac{1}{16}\lim_{n\to\infty} \frac{(n^2+n)(n^2+n)}{n^4} = \\ &= \frac{1}{16}\lim_{n\to\infty} \frac{n^4+2n^3+n^2}{n^4} = \frac{1}{16}\lim_{n\to\infty} \frac{4n^3+6n^2+2n}{4n^3} = \frac{1}{16}\lim_{n\to\infty} \frac{12n^2+12n+2}{12n^2} = \\ &= \frac{1}{16}\lim_{n\to\infty} \frac{24n+12}{24n} = \frac{1}{16}\lim_{n\to\infty} \frac{24}{24} = \frac{1}{16}. \end{aligned} \tag{6}$$

- **Área esperada de células *Pouco* × *Muito* e *Muito* × *Pouco***

$$\begin{aligned} &\mathrm{E}[G(Pouco, Muito)] = \sum_{i=1}^{n}\sum_{j=1}^{n} \delta_i(pouco).\delta_j(muito).p(pouco).p(muito) = \\ &= \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\left(\frac{j}{2n}+\frac{1}{2}\right)}{n^2} = \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{j}{2n} + \sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{1}{2}}{n^2} = \\ &= \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{j}{2n}}{n^2} + \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{1}{2}}{n^2}. \end{aligned} \tag{7}$$

Os dois limites apresentados na Equação 7 podem ser resolvidos pela soma das equações 8 e 9.

$$\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{j}{2n}}{n^2}=\lim_{n\to\infty}\frac{\frac{1}{4n^2}\sum_{i=1}^{n}i\sum_{j=1}^{n}j}{n^2}=\lim_{n\to\infty}\frac{1}{4n^4}\sum_{i=1}^{n}i\sum_{j=1}^{n}j=$$
$$=\lim_{n\to\infty}\frac{1}{4n^4}\frac{n(n+1)}{2}\frac{n(n+1)}{2}=\frac{1}{16}\lim_{n\to\infty}\frac{(n^2+n)(n^2+n)}{n^4}=\frac{1}{16}. \quad (8)$$

$$\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{1}{2}}{n^2}=\lim_{n\to\infty}\frac{\frac{1}{4n}\sum_{i=1}^{n}i\sum_{j=1}^{n}1}{n^2}=\lim_{n\to\infty}\frac{1}{4n^3}\sum_{i=1}^{n}i\sum_{j=1}^{n}1=$$
$$=\lim_{n\to\infty}\frac{1}{4n^3}\sum_{i=1}^{n}i\sum_{j=1}^{n}1=\lim_{n\to\infty}\frac{1}{4n^3}\sum_{i=1}^{n}i\times n=\lim_{n\to\infty}\frac{1}{4n^3}\frac{n(n+1)}{2}n= \quad (9)$$
$$=\frac{1}{8}\lim_{n\to\infty}\frac{n^3+n^2}{n^3}=\frac{1}{8}\lim_{n\to\infty}\frac{3n^2+2n}{3n^2}=\frac{1}{8}\lim_{n\to\infty}\frac{6n+2}{6n}=\frac{1}{8}\lim_{n\to\infty}\frac{6}{6}=\frac{1}{8}.$$

Aplicando as equações 8 e 9 na Equação 7, a área esperada de interseção entre células *Pouco* e *Muito* (ou entre células *Muito* e *Pouco*) é apresentada na Equação 10.

$$\mathrm{E}[G(Pouco,Muito)]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(pouco).\delta_j(muito).p(pouco).p(muito)=\frac{1}{16}+\frac{1}{8}=\frac{3}{16}. \quad (10)$$

- **Área esperada de células *Muito* ×*Muito***

$$\mathrm{E}[G(Muito_1,Muito_2)]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(muito_1).\delta_j(muito_2).p(muito_1).p(muito_2)=$$
$$=\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\left(\frac{i}{2n}+\frac{1}{2}\right)\sum_{j=1}^{n}\left(\frac{j}{2n}+\frac{1}{2}\right)}{n^2}=$$
$$=\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{j}{2n}+\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{1}{2}+\sum_{i=1}^{n}\frac{j}{2n}\sum_{j=1}^{n}\frac{1}{2}+\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}\frac{1}{2}}{n^2}= \quad (11)$$
$$=\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}\frac{j}{2n}}{n^2}+\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=}^{n}\frac{1}{2}}{n^2}+\lim_{n\to\infty}\frac{\sum_{j=1}^{n}\frac{1}{2}\sum_{j=1}^{n}\frac{j}{2n}}{n^2}+\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}\frac{1}{2}}{n^2}.$$

O primeiro limite apresentado na Equação 11 pode ser resolvido de acordo com a Equação 8, enquanto que os dois limites seguintes podem ser resolvidos de acordo com a Equação 9. O quarto limite pode ser resolvido segundo a Equação 12.

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}\frac{1}{2}}{n^2}=\underset{n\to\infty}{Lim}\frac{1}{4}\frac{\sum_{i=1}^{n}1\sum_{j=1}^{n}1}{n^2}=\frac{1}{4}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}1\sum_{j=1}^{n}1}{n^2}=\frac{1}{4}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}1\times n}{n^2}=$$
$$=\frac{1}{4}\underset{n\to\infty}{Lim}\frac{n\times n}{n^2}=\frac{1}{4}\underset{n\to\infty}{Lim}\frac{n^2}{n^2}=\frac{1}{4}. \quad (12)$$

Logo, aplicando as equações 8, 9 e 12 na Equação 11, a área esperada de interseção entre células *Muito* e *Muito* é apresentada na Equação 13.

$$\mathrm{E}\left[G(Muito_1,Muito_2)\right]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(muito_1).\delta_j(muito_2).p(muito_1).p(muito_2)=$$
$$=\frac{1}{16}+\frac{1}{8}+\frac{1}{8}+\frac{1}{4}=\frac{9}{16}. \quad (13)$$

- **Área esperada de células *Pouco* × *Cheio* e *Cheio* × *Pouco***

$$\mathrm{E}\left[G(Pouco,Cheio)\right]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(pouco).\delta_j(cheio).p(pouco).p(cheio)=$$
$$=\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}1}{n^2}=\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{i}{2n}\times n}{n^2}=\underset{n\to\infty}{Lim}\frac{1}{2}\frac{\sum_{i=1}^{n}i\times n}{n^3}=\underset{n\to\infty}{Lim}\frac{1}{2}\frac{\frac{n(n+1)}{2}\times n}{n^3}= \quad (14)$$
$$=\underset{n\to\infty}{Lim}\frac{1}{4}\frac{n^3+n^2}{n^3}=\frac{1}{4}.$$

- **Área esperada de células *Muito* × *Cheio* e *Cheio* × *Muito***

$$\mathrm{E}\left[G(Muito,Cheio)\right]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(muito).\delta_j(cheio).p(muito).p(cheio)=$$
$$=\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\left(\frac{i}{2n}+\frac{1}{2}\right)\sum_{j=1}^{n}1}{n^2}=\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}1+\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}1}{n^2}= \quad (15)$$
$$=\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{i}{2n}\sum_{j=1}^{n}1}{n^2}+\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}1}{n^2}.$$

O primeiro limite apresentado na Equação 15 pode ser resolvido segundo a Equação 14. Já o segundo limite da Equação 15 pode ser resolvido de acordo com a Equação 16.

$$\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{1}{2}\sum_{j=1}^{n}1}{n^2}=\lim_{n\to\infty}\frac{\sum_{i=1}^{n}\frac{1}{2}\times n}{n^2}=\lim_{n\to\infty}\frac{1}{2}\frac{n\times n}{n^2}=\frac{1}{2}\lim_{n\to\infty}\frac{n^2}{n^2}=\frac{1}{2}. \quad (16)$$

Aplicando as equações 14 e 16 na Equação 15, temos que a área esperada de interseção entre células *Muito* e *Cheio* é dada pela Equação 17.

$$\mathrm{E}[G(Muito,Cheio)]=\sum_{i=1}^{n}\sum_{j=1}^{n}\delta_i(muito).\delta_j(cheio).p(muito).p(cheio)=\frac{1}{4}+\frac{1}{2}=\frac{3}{4}. \quad (17)$$

A Tabela 3 apresenta a área esperada para interseção de tipos de células de acordo com os cálculos apresentados nesta subseção.

Tabela 3. Área esperada de sobreposição de tipos de células.

Tipos de células	**Vazio**	**Pouco**	**Muito**	**Cheio**
Vazio	0	0	0	0
Pouco	0	0,0625	0,1875	0,25
Muito	0	0,1875	0,5625	0,75
Cheio	0	0,25	0,75	1

4.5.2. Algoritmo

O algoritmo para computar a área de interseção entre polígonos avalia a soma da área esperada das células de suas assinaturas 4CRS que se interceptam, e multiplica o valor resultante pela área da célula. Como existem quatro tipos diferentes de células, existem dezesseis possíveis sobreposições de tipos de células e, consequentemente, dezesseis possibilidades de área esperada de sobreposição de células, como apresentado na Tabela 3. O algoritmo emprega uma matriz para armazenar as áreas esperadas. É importante observar que apenas são consideradas no cálculo as células que estão contidas no MBR de interseção de duas assinaturas 4CRS. O algoritmo é apresentado na Figura 26, e tanto células com mesmo tamanho de lado como também células com tamanho de lado diferentes são tratadas pelo algoritmo. É garantido que quando duas células se interceptam, então seus lados se sobrepõem perfeitamente, e quando os tamanhos de células são diferentes, sempre é possível garantir que a célula de menor

tamanho está contida completamente pela célula de maior tamanho, de acordo com a abordagem usada para computar a grade de células apresentada na Subseção 3.1.2.

```
algoritmo areaAproximadaDeIntersecao(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se (assinat4CRS1.tamanhoDoLadoDaCelula ==
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = assinat4CRS1;
    b4CRS = assinat4CRS2;
  senão
    s4CRS = menorTamanhoDoLadoDaCelula(assinat4CRS1, assinat4CRS2);
    b4CRS = maiorTamanhoDoLadoDaCelula(assinat4CRS1, assinat4CRS2);
  areaAproximada = 0;
  para cada celula b4CRS b contida em MBRintersec faça
     para cada celula s4CRS s que intercepta b faça
       areaAproximada += areaEsperada[s.tipo,b.tipo];
  areaCelula = s4CRS.tamanhoDoLadoDaCelula *
               s4CRS.tamanhoDoLadoDaCelula;
  retornar areaAproximada * areaCelula;
```

Figura 26. Algoritmo para computar a área aproximada de interseção de polígono × polígono.

4.6. Cálculo do Intervalo de Confiança

Ao executar uma consulta que retorna resultados aproximados, é importante mostrar para o usuário um intervalo de confiança para a resposta obtida a fim de que ele possa decidir se a precisão é suficiente. A medida de precisão usada neste trabalho está baseada no Teorema do Limite Central (STEEL e TORRIE, 1976), o qual se aplica quase sempre, independente da função de densidade. Segundo o Teorema do Limite Central, se uma população tem média μ e variância σ^2, então a distribuição da média das amostras derivadas desta distribuição tende a uma distribuição normal com média μ e variância σ^2/n quando o tamanho das amostras n aumenta. Dessa forma, em algum estágio, para tamanhos de amostras suficientemente grandes, independente se a variável é randômica ou discreta, a distribuição terá aproximadamente distribuição normal. Obviamente, a forma da função de densidade original terá algum efeito sobre o tamanho de amostras requerido, e uma distribuição assimétrica geralmente necessitará de um valor n maior do que uma distribuição simétrica. Entretanto, um tamanho de amostras igual a 30 é grande o suficiente para muitas distribuições. Portanto, para computar a área aproximada, se existem poucas células ou poucas combinações de células a serem processadas, a distribuição não tende a ser normal. Por outro lado, consultas envolvendo poucos polígonos podem ser executadas acessando os polígonos ao invés de processar as assinaturas 4CRS, o que, neste caso, implicará um pouco mais de gasto tempo de processamento.

O intervalo de confiança é computado como a soma dos intervalos de confiança para cada combinação de pares de tipos células. Consultando uma tabela de distribuição

normal, para um intervalo de confiança de 95%, a resposta varia nos limites ($\boldsymbol{\mu}\pm1.96\times(\boldsymbol{\sigma^2/n})^{1/2}$), e, para um intervalo de confiança de 99%, nos limites ($\boldsymbol{\mu}\pm2.576\times(\boldsymbol{\sigma^2/n})^{1/2}$). A Equação 18 foi utilizada para computar o intervalo de confiança das respostas obtidas nos experimentos executados.

$$Intervalo\ de\ Confiança = \sum_c n_c \times \left(\mu_c \pm p \times \sqrt{\frac{\sigma_c^{\ 2}}{n_c}} \right). \qquad (18)$$

- c: tipo de célula ou combinação de tipos de células dependendo do algoritmo sendo avaliado.
- μ_c: média (ou área esperada);
- σ_c^2: variância;
- p: intervalo de confiança escolhido, i.e., 1.96 para um intervalo de confiança de 95%;
- n_c: número de células.

Para obter o resultado em unidades de área é necessário multiplicar o resultado obtido pela área da célula.

Para cada algoritmo, μ_c, σ_c^2, e n_c assumem valores diferentes. Por causa da semelhança entre os algoritmos para computar área de polígono e para computar a área aproximada de polígono dentro de janela, estes parâmetros assumem os mesmos valores. Por outro lado, é necessário um cálculo diferente para computar estes parâmetros na avaliação do algoritmo que estima a área de interseção de polígono × polígono. Os cálculos são apresentados nas subseções 4.6.1 e 4.6.2.

4.6.1. Cálculo do intervalo de confiança para o algoritmo que calcula a área aproximada de polígono e para o algoritmo que computa a área aproximada de polígono dentro de janela

No caso do algoritmo que calcula a área aproximada de polígono e do algoritmo que calcula a área aproximada de polígono dentro de janela, μ_c corresponde à área esperada para cada tipo de célula (os valores são apresentados na Tabela 2), n_c é o número de células de tipo *c*, e σ_c^2 é a variância correspondente ao tipo de célula *c*. As variâncias para os tipos de células *Vazio* e *Cheio* são iguais a 0 (zero), já que estes tipos de células têm percentual exato de área do polígono dentro da célula, cujos valores são 0% e 100%, respectivamente. Por outro lado, é necessário estimar a variância para células dos tipos *Pouco* e *Muito*. Como estamos assumindo que a distribuição de área do

polígono dentro de célula é muito próxima da distribuição uniforme (como foi apresentado na Subseção 4.3.1), a variância para células *Pouco* é de $(0,5-0)^2/12 = 1/48 = 0.020833$, já que células pouco têm distribuição no intervalo (0; 0,50]. Células *Muito* têm distribuição com a mesma variância $(1-0,5)^2/12 = 1/48 = 0,020833$ no intervalo (0,5; 0,1).

Considere o seguinte exemplo. Uma consulta de janela produz 100 células *Pouco*, 120 células *Muito* e 400 células *Cheio*. Uma resposta com um intervalo de confiança de 95% pode ser calculada como é apresentado na Figura 27 (por simplicidade estamos assumindo que cada célula tem mesma área, com valor igual a 1).

- Células *Pouco*: $100\times(0,25 \pm 1,96\times(0,020833/100)^{1/2})=25\pm2,83$
- Células *Muito*: $120\times(0,75 \pm 1,96\times(0.020833/120)^{1/2})=90\pm3,10$
- Células *Cheio*: 400 (full cells have the exact area!)
- Total: $515\pm5,93$

Figura 27. Exemplo de cálculo com 95% de intervalo de confiança.

Dessa forma, o intervalo de confiança tem uma variância de ±5,93. Isto significa que, para um intervalo de confiança de 95%, a área aproximada envolvendo estes números de células tem um erro de no máximo ±1,15%, o que é um resultado suficiente para muitas aplicações. Da mesma forma, nós podemos calcular uma resposta com um intervalo de confiança de 99%, alterando o parâmetro 1,96 para 2,56 (Figura 28). O intervalo de confiança tem uma variância de ± 7,74, o que representa um erro máximo de ±1,50%.

- Células *Pouco*: $100\times(0,25 \pm 2,56 \times(0,020833/100)^{1/2})=25\pm3,70$
- Células *Muito*: $120\times(0,75 \pm 2,56\times(0,020833/120)^{1/2})=90\pm4,05$
- Células *Cheio*: 400 (full cells have the exact area!)
- Total: $515\pm 7,74$

Figura 28. Exemplo de cálculo com 99% de intervalo de confiança.

Reciprocamente, vamos analisar uma consulta envolvendo poucas células, por exemplo, 10 células *Pouco*, 12 células *Muito* e 10 células *Cheio*, como apresentado na Figura 29. Como afirmamos anteriormente, com poucas células, a distribuição não tende a ser Normal. Sendo assim, o erro da resposta tende a ser maior. Apesar disso, consultas envolvendo poucos polígonos (ou poucas células) podem ser executadas

acessando-se os polígonos, ao invés de processá-las de forma aproximada sobre as assinaturas 4CRS. O custo de tempo para execução da consulta exata, neste caso, não deve ser muito maior do que o custo para executar a consulta aproximada.

- Células *Pouco*: $10\times(0,25 \pm 1,96\times(10/48)^{1/2})=2,5\pm0,89$
- Células *Muito*: $12\times(0,75 \pm 1,96\times(12/48)^{1/2})=9\pm0,98$
- Células *Cheio*: 10 (full cells have the exact area!)
- Total: 21,5±1,87, o que significa um erro de no máximo ±8,9%.

Figura 29. Exemplo de calculo com 95% de intervalo de confiança envolvendo poucas células.

4.6.2. Cálculo do Intervalo de Confiança para o algoritmo que calcula a área aproximada de interseção de polígono × polígono

O cálculo do intervalo de confiança da resposta retornada pelo algoritmo que calcula a área aproximada de interseção de polígono × polígono depende que sejam calculados os valores de média e variância das áreas esperadas correspondentes à sobreposição de duas células de mesmo tamanho. Os valores de média são apresentados na Tabela 3 (Subseção 4.5.1), enquanto que o cálculo da variância para combinação de células é apresentado a seguir.

A área esperada correspondente à combinação de uma célula do tipo *Vazio* com qualquer outro tipo de célula é igual 0% (zero por cento), já que a interseção de células com estas características tem área igual a zero. Consequentemente, a variância da área esperada é igual a zero. Analogamente, quando duas células do tipo *Cheio* se sobrepõem, a área esperada é igual a 100%, e a variância também é igual a zero. Dessa forma, é necessário apenas computar as variâncias para as áreas esperadas de sobreposição dos outros tipos de células (*Pouco* × *Pouco*, *Pouco* × *Muito*, *Pouco* × *Cheio*, *Muito* × *Pouco*, *Muito* × *Muito*, *Muito* × *Cheio*). Nós assumimos as mesmas suposições que foram usadas para calcular a área esperada correspondente à sobreposição de tipos de células com o mesmo tamanho apresentadas na Subseção 4.5.1.

Os conjuntos de dados são razoavelmente independentes, e a área esperada correspondente à interseção de dois tipos de células com áreas iguais x_1 e x_2 pode ser calculada como $x_1 \times x_2$.

Mesmo sabendo que a área é um valor contínuo, com o intuito de tornar mais simples a demonstração dos cálculos, nós assumimos que a área da célula é computada como um valor discreto, em passos de tamanho 1/*n*, para *n* suficientemente grande ($n \rightarrow \infty$). Além disso, todos os valores são apresentados em percentuais.

Sejam *X* variável randômica representando a área de interseção entre uma célula da grade e o polígono; $G(x_1, x_2)$ função que retorne a área de interseção entre dois tipos de células x_1 e x_2; e $p(x_1, x_2)$ a função de probabilidade conjunta das variáveis X_1 e X_2. A variância da área de interseção de dois tipos de células pode ser expressa de acordo com a Equação 19.

$$\sigma^2 = \sum_{x_1} \sum_{x_2} \left(G(x_1, x_2) - \mu\right)^2 \times p(x_1, x_2) \ . \qquad (19)$$

Nós também assumimos que X_1 e X_2 são linearmente independentes e que a função de probabilidade conjunta $p(x_1, x_2)$ pode ser expressa como $p(x_1, x_2) = p(x_1) \times p(x_2)$; $p(x_1)$ e $p(x_2)$ podem ser expressos como $p(x_1) = p(x_2) = 1/n$; e, $G(x_1, x_2)$ corresponde à multiplicação da área de interseção do polígono dentro das células. Desse modo, a Equação 18 pode ser reescrita como a Equação 20.

$$\sigma^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\delta(x_1) \times \delta(x_2) - \mu\right)^2 \times p(x_1) \times p(x_2) \ , \qquad (20)$$

onde $\delta(x)$ é uma função que retorna o percentual de área de interseção do polígono dentro da célula. Esta função é expressa pelas equações 3 e 4 (Subseção 4.5.1).

- **Variância da área esperada de células *Pouco* × *Pouco***

A partir das equações 3, 4 e 20, a variância do percentual de área de interseção entre duas células *Pouco* pode ser calculada de acordo com a Equação 21.

$$\begin{aligned}
&\sigma^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\delta(pouco_1) \times \delta(pouco_2) - \mu_{pouco \times pouco}\right)^2 \times p(pouco_1) \times p(pouco_2) = \\
&= \lim_{n \to \infty} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{i}{2n} \times \frac{j}{2n} - \mu_{pouco \times pouco} \right)^2 \times \frac{1}{n} \times \frac{1}{n} \\
&= \lim_{n \to \infty} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{i^2 j^2}{16n^4} - \frac{ij\mu_{pouco \times pouco}}{2n^2} + \mu_{pouco \times pouco}^{\ 2} \right) \times \frac{1}{n^2} \\
&= \lim_{n \to \infty} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{i^2 j^2}{16n^6} - \lim_{n \to \infty} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{ij\mu_{pouco \times pouco}}{2n^4} + \lim_{n \to \infty} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{\mu_{pouco \times pouco}^{\ 2}}{n^2}
\end{aligned} \qquad (21)$$

$$= \underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{i^2 j^2}{16n^6} - \underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{ij\mu_{pouco\times pouco}}{2n^4} + \underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\mu_{pouco\times pouco}^{\ 2}}{n^2} \ .$$

Dado que a soma das sequências aritméticas $\sum_{k=1}^{n} k$ e $\sum_{k=1}^{n} k^2$ podem ser expressas como $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$ e $\sum_{k=1}^{n} k^2 = \frac{2n^3+3n^2+n}{6}$, e usando a regra de L'Hôpital, os três limites da Equação 21 podem ser resolvidos de acordo com as Equações 22, 23 e 24.

$$\underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{i^2 j^2}{16n^6} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} i^2 j^2}{n^6} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\frac{2n^3+3n^2+n}{6} \times \frac{2n^3+3n^2+n}{6}}{n^6} = \frac{1}{144} \ . \qquad \textbf{(22)}$$

$$\underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{ij\mu_{pouco\times pouco}}{2n^4} = \frac{\mu_{pouco\times pouco}}{2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} ij}{n^4} = \frac{\mu_{pouco\times pouco}}{2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} i \frac{n(n+1)}{2}}{n^4}$$
$$= \frac{\mu_{pouco\times pouco}}{2} \underset{n\to\infty}{Lim} \frac{\frac{n(n+1)}{2} \times \frac{n(n+1)}{2}}{n^4} = \frac{\mu_{pouco\times pouco}}{8} = \frac{1}{16} \times \frac{1}{8} = \frac{1}{128} \ . \qquad \textbf{(23)}$$

$$\underset{n\to\infty}{Lim} \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\mu_{pouco\times pouco}^{\ 2}}{n^2} = \mu_{pouco\times pouco}^{\ 2} \underset{n\to\infty}{Lim} \frac{n\times n}{n^2} = \mu_{pouco\times pouco}^{\ 2} = \left(\frac{1}{16}\right)^2 = \frac{1}{256} . \qquad \textbf{(24)}$$

Aplicando as equações 22, 23 e 24 na Equação 21, a variância do percentual de área de interseção entre duas células *Pouco* é igual ao valor apresentado na Equação 25.

$$\sigma^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(pouco_1)\times\delta(pouco_2) - \mu_{pouco\times pouco}\right)^2 \times p(pouco_1)\times p(pouco_2) =$$

$$= \frac{1}{144} - \frac{1}{128} + \frac{1}{256} = 0{,}003038194\ . \qquad (25)$$

- **Variância da área esperada de células *Pouco* × *Muito* e *Muito* × *Pouco***

$$\sigma^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(pouco_1)\times\delta(muito_2) - \mu_{pouco\times muito}\right)^2 \times p(pouco_1)\times p(muito_2) =$$

$$= \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\left[\frac{i}{2n}\left(\frac{j}{2n}+\frac{1}{2}\right)\right] - \mu_{pouco\times muito}\right)^2 \times\frac{1}{n}\times\frac{1}{n}$$

$$= \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{i^2 j^2}{16n^4} + \frac{i^2 j}{8n^3} + \frac{i^2}{16n^2} - \frac{ij\mu_{pouco\times muito}}{2n^2} - \frac{i\mu_{pouco\times muito}}{2n} + \mu_{pouco\times muito}{}^2\right)\times\frac{1}{n^2} \qquad (26)$$

$$= \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2 j^2}{16n^6} + \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2 j}{8n^5} + \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{16n^4} - \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{ij\mu_{pouco\times muito}}{2n^4}$$

$$- \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{pouco\times muito}}{2n^3} + \underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\mu_{pouco\times muito}{}^2}{n^2}\ .$$

Considere as equações apresentadas a seguir.

$$\underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2 j}{8n^5} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{1}{n^5}\sum_{i=1}^{n} i^2\,\frac{n(n+1)}{2} = \frac{1}{16}\underset{n\to\infty}{Lim}\frac{1}{n^5}\,\frac{2n^3+3n^2+n}{6}\left(n^2+n\right) =$$

$$= \frac{1}{96}\underset{n\to\infty}{Lim}\frac{2n^5+5n^4+4n^3+n^2}{n^5} = \frac{1}{48}\ . \qquad (27)$$

$$\underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{16n^4} = \frac{1}{16}\underset{n\to\infty}{Lim}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{n^4} = \frac{1}{16}\underset{n\to\infty}{Lim}\sum_{i=1}^{n}\frac{i^2}{n^4}n =$$

$$= \frac{1}{16}\underset{n\to\infty}{Lim}\frac{1}{n^4}\,\frac{2n^3+3n^2+n}{6}n = \frac{1}{96}\underset{n\to\infty}{Lim}\frac{2n^4+3n^3+n^2}{n^4} = \frac{1}{48}. \qquad (28)$$

$$\lim_{n\to\infty}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{ij\mu_{pouco\times muito}}{2n^4}=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{ij}{n^4}=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\frac{1}{n^4}\sum_{i=1}^{n}i\frac{n(n+1)}{2}=$$

$$=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\frac{1}{n^4}\sum_{i=1}^{n}i\frac{n(n+1)}{2}=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\frac{1}{n^4}\frac{n(n+1)}{2}\frac{n(n+1)}{2}= \quad (29)$$

$$=\frac{\mu_{pouco\times muito}}{8}\lim_{n\to\infty}\frac{n^4+2n^3+n^2}{n^4}=\frac{\mu_{pouco\times muito}}{8}=\frac{1}{8}\times\frac{3}{16}=\frac{3}{128}.$$

$$\lim_{n\to\infty}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{pouco\times muito}}{2n^3}=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i}{n^3}=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\sum_{i=1}^{n}\frac{i}{n^3}n=$$

$$=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\sum_{i=1}^{n}\frac{i}{n^3}n=\frac{\mu_{pouco\times muito}}{2}\lim_{n\to\infty}\frac{1}{n^3}\frac{n(n+1)}{2}n=\frac{\mu_{pouco\times muito}}{4}\lim_{n\to\infty}\frac{n^3+n^2}{n^3}= \quad (30)$$

$$=\frac{\mu_{pouco\times muito}}{4}=\frac{1}{4}\times\frac{3}{16}=\frac{3}{64}.$$

$$\lim_{n\to\infty}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\mu_{pouco\times muito}{}^2}{n^2}=\mu_{pouco\times muito}{}^2\lim_{n\to\infty}\frac{n\times n}{n^2}=\mu_{pouco\times muito}{}^2=\left(\frac{3}{16}\right)^2=\frac{9}{256}. \quad (31)$$

Aplicando as equações 22, 27, 28, 29, 30 e 31 na Equação 26, temos que a variância da área esperada de células *Pouco* × *Muito* ou *Muito* × *Pouco* pode ser calculada como apresentado na Equação 32.

$$\sigma^2=\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(pouco)\times\delta(muito)-\mu_{pouco\times muito}\right)^2\times p(pouco)\times p(muito)=$$

$$=\frac{1}{144}+\frac{1}{48}+\frac{1}{48}-\frac{3}{128}-\frac{3}{64}+\frac{9}{256}=0{,}013454861. \quad (32)$$

- **Variância da área esperada de células *Pouco* × *Cheio* e *Cheio* × *Pouco***

$$\sigma^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(pouco)\times\delta(cheio) - \mu_{pouco\times cheio}\right)^2 \times p(pouco)\times p(cheio) =$$

$$= \underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{i}{2n} - \mu_{pouco\times cheio}\right)^2}{n^2} = \underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{4n^2} - \frac{i\mu_{pouco\times cheio}}{n} + \mu_{pouco\times cheio}^{\ 2}}{n^2} = \quad (33)$$

$$= \underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{4n^2}}{n^2} - \underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{pouco\times cheio}}{n}}{n^2} + \underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_{pouco\times cheio}^{\ 2}}{n^2}.$$

Os três limites da Equação 33 podem ser resolvidos de acordo com as equações 34, 35 e 36.

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{4n^2}}{n^2} = \frac{1}{4}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}i^2}{n^4} = \frac{1}{4}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i^2\times n}{n^4} = \frac{1}{4}\underset{n\to\infty}{Lim}\frac{1}{n^4}\frac{2n^3+3n^2+n}{6}n = \quad (34)$$

$$= \frac{1}{24}\underset{n\to\infty}{Lim}\frac{2n^4+3n^3+n^2}{n^4} = \frac{1}{12}.$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{pouco\times cheio}}{n}}{n^2} = \mu_{pouco\times cheio}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i}{n}}{n^2} = \mu_{pouco\times cheio}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i\times n}{n^3} = \quad (35)$$

$$= \mu_{pouco\times cheio}\underset{n\to\infty}{Lim}\frac{1}{n^3}\frac{n(n+1)}{2}n = \frac{\mu_{pouco\times cheio}}{2}\underset{n\to\infty}{Lim}\frac{n^3+n^2}{n^3} = \frac{\mu_{pouco\times cheio}}{2} = \frac{1}{2}\times\frac{1}{4} = \frac{1}{8}.$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_{pouco\times cheio}^{\ 2}}{n^2} = \mu_{pouco\times cheio}^{\ 2}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}1}{n^2} = \mu_{pouco\times cheio}^{\ 2}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}n}{n^2} = \quad (36)$$

$$= \mu_{pouco\times cheio}^{\ 2}\underset{n\to\infty}{Lim}\frac{n\times n}{n^2} = \mu_{pouco\times cheio}^{\ 2} = \left(\frac{1}{4}\right)^2 = \frac{1}{16}.$$

Aplicando as equações 34, 35 e 36 na Equação 33, a variância da área esperada de células *Pouco* × *Cheio* e *Cheio* × *Pouco* pode ser calculada como apresentado na Equação 37.

$$\sigma^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(pouco)\times\delta(cheio) - \mu_{pouco\times cheio}\right)^2 \times p(pouco)\times p(cheio) = \tag{37}$$

$$= \frac{1}{12} - \frac{1}{8} + \frac{1}{16} = 0{,}02833333.$$

- **Variância da área esperada de células *Muito* × *Cheio* e *Cheio* × *Muito***

$$\sigma^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(muito)\times\delta(cheio) - \mu_{muito\times cheio}\right)^2 \times p(muito)\times p(cheio) =$$

$$= \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\left(\frac{i}{2n} + \frac{1}{2}\right) - \mu_{muito\times cheio}\right]^2}{n^2} =$$

$$= \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{4n^2} + \frac{i}{2n} + \frac{1}{4} - \frac{i\mu_{muito\times cheio}}{n} - \mu_{muito\times cheio} + \mu_{muito\times cheio}{}^2}{n^2} = \tag{38}$$

$$= \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{4n^2}}{n^2} + \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i}{2n}}{n^2} + \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{4}}{n^2} - \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{muito\times cheio}}{n}}{n^2} -$$

$$- \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_{muito\times cheio}}{n^2} + \lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_{muito\times cheio}{}^2}{n^2}.$$

Considere as equações a seguir.

$$\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i}{2n}}{n^2} = \frac{1}{2}\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} i}{n^3} = \frac{1}{2}\lim_{n\to\infty} \frac{\sum_{i=1}^{n} i\times n}{n^3} = \frac{1}{2}\lim_{n\to\infty} \frac{\frac{n(n+1)}{2}n}{n^3} = \tag{39}$$

$$= \frac{1}{4}\lim_{n\to\infty} \frac{n^3 + n^2}{n^3} = \frac{1}{4}.$$

$$\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{4}}{n^2} = \frac{1}{4}\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}1}{n^2} = \frac{1}{4}\lim_{n\to\infty} \frac{\sum_{i=1}^{n} n}{n^2} = \frac{1}{4}\lim_{n\to\infty} \frac{n^2}{n^2} = \frac{1}{4}. \tag{40}$$

$$\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i\mu_{muito\times cheio}}{n}}{n^2} = \mu_{muito\times cheio}\lim_{n\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} i}{n^3} = \mu_{muito\times cheio}\lim_{n\to\infty} \frac{\sum_{i=1}^{n} i\times n}{n^3} = \tag{41}$$

$$= \mu_{muito \times cheio} \underset{n\to\infty}{Lim} \frac{\frac{n(n+1)}{2} n}{n^3} = \frac{\mu_{muito \times cheio}}{2} \underset{n\to\infty}{Lim} \frac{n^3 + n^2}{n^3} = \frac{\mu_{muito \times cheio}}{2} = \frac{1}{2} \times \frac{3}{4} = \frac{3}{8}.$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \mu_{muito \times cheio}}{n^2} = \mu_{muito \times cheio} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} 1}{n^2} = \mu_{muito \times cheio} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} n}{n^2} =$$

$$= \mu_{muito \times cheio} \underset{n\to\infty}{Lim} \frac{n^2}{n^2} = \mu_{muito \times cheio} = \frac{3}{4}. \quad (42)$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \mu_{muito \times cheio}^{\ 2}}{n^2} = \mu_{muito \times cheio}^{\ 2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} 1}{n^2} = \mu_{muito \times cheio}^{\ 2} = \left(\frac{3}{4}\right)^2 = \frac{9}{16}. \quad (43)$$

Aplicando as equações 34, 39, 40, 41, 42 e 43 na Equação 38, a variância da área esperada de células *Muito* × *Cheio* e *Cheio* × *Muito* pode ser calculada como apresentado na Equação 44.

$$\sigma^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\delta(muito) \times \delta(cheio) - \mu_{muito \times cheio}\right)^2 \times p(muito) \times p(cheio) =$$

$$= \frac{1}{12} + \frac{1}{4} + \frac{1}{4} - \frac{3}{8} - \frac{3}{4} + \frac{9}{16} = 0{,}020833333. \quad (44)$$

- **Variância da área esperada de células *Muito* × *Muito***

$$\sigma^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\delta(muito_1) \times \delta(muito_2) - \mu_{muito \times muito}\right)^2 \times p(muito_1) \times p(muito_2) =$$

$$= \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \left[\left(\frac{i}{2n} + \frac{1}{2}\right) \times \left(\frac{j}{2n} + \frac{1}{2}\right) - \mu_{muito \times muito}\right]^2}{n^2} = \quad (45)$$

$$= \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{i^2 j^2}{16n^4} + \frac{i^2 j}{8n^3} + \frac{i^2}{16n^2} + \frac{ij^2}{8n^2} + \frac{ij}{4n^2} + \frac{i}{8n} + \frac{j^2}{16n^2} + \frac{j}{8n} + \frac{1}{16} - \frac{\mu ij}{2n} - \frac{\mu i}{2n} - \frac{\mu j}{2n} - \frac{\mu}{2} + \mu^2\right)}{n^2}$$

Considere as equações a seguir.

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \frac{i^2 j^2}{16n^4}}{n^2} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} i^2 j^2}{n^6} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} i^2 \frac{2n^3 + 3n^2 + n}{6}}{n^6} = \quad (46)$$

$$= \frac{1}{16}\underset{n\to\infty}{Lim}\frac{\frac{2n^3+3n^2+n}{6}\frac{2n^3+3n^2+n}{6}}{n^6} = \frac{1}{576}\underset{n\to\infty}{Lim}\frac{\left(2n^3+3n^2+n\right)^2}{n^6} = \frac{1}{144}.$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2 j}{8n^3}}{n^2} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}i^2 j}{n^5} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i^2\frac{n(n+1)}{2}}{n^5} =$$

$$= \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\frac{2n^3+3n^2+n}{6}\frac{n(n+1)}{2}}{n^5} = \frac{1}{96}\underset{n\to\infty}{Lim}\frac{\left(2n^3+3n^2+n\right)\times\left(n^2+n\right)}{n^5} = \frac{1}{48}. \quad (47)$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i^2}{16n^2}}{n^2} = \frac{1}{16}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}i^2}{n^4} = \frac{1}{16}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i^2\times n}{n^4} = \frac{1}{16}\underset{n\to\infty}{Lim}\frac{\frac{2n^3+3n^2+n}{6}n}{n^4} =$$

$$= \frac{1}{96}\underset{n\to\infty}{Lim}\frac{\left(2n^3+3n^2+n\right)\times n}{n^4} = \frac{1}{48}. \quad (48)$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{ij^2}{8n^3}}{n^2} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}ij^2}{n^5} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{j=1}^{n}j^2\frac{n(n+1)}{2}}{n^5} =$$

$$= \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\frac{2n^3+3n^2+n}{6}\frac{n(n+1)}{2}}{n^5} = \frac{1}{96}\underset{n\to\infty}{Lim}\frac{\left(2n^3+3n^2+n\right)\times\left(n^2+n\right)}{n^5} = \frac{1}{48}. \quad (49)$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{ij}{4n^2}}{n^2} = \frac{1}{4}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i\frac{n(n+1)}{2}}{n^4} = \frac{1}{4}\underset{n\to\infty}{Lim}\frac{\frac{n(n+1)}{2}\frac{n(n+1)}{2}}{n^4} =$$

$$= \frac{1}{16}\underset{n\to\infty}{Lim}\frac{\left(n^2+n\right)^2}{n^4} = \frac{1}{16}. \quad (50)$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{i}{8n}}{n^2} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}i}{n^3} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}i\times n}{n^3} = \frac{1}{8}\underset{n\to\infty}{Lim}\frac{\frac{n(n+1)}{2}n}{n^3} = \quad (51)$$

$$= \frac{1}{16} \underset{n\to\infty}{Lim} \frac{n^3 + n^2}{n^3} = \frac{1}{16}.$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{j^2}{16n^2}}{n^2} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} j^2}{n^4} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\sum_{j=1}^{n} j^2 \times n}{n^4} = \frac{1}{16} \underset{n\to\infty}{Lim} \frac{\frac{2n^3 + 3n^2 + n}{6} n}{n^4} =$$

$$= \frac{1}{96} \underset{n\to\infty}{Lim} \frac{\left(2n^3 + 3n^2 + n\right) \times n}{n^4} = \frac{1}{48}. \tag{52}$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{j}{8n}}{n^2} = \frac{1}{8} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} j}{n^3} = \frac{1}{8} \underset{n\to\infty}{Lim} \frac{\sum_{j=1}^{n} j \times n}{n^3} = \frac{1}{8} \underset{n\to\infty}{Lim} \frac{\frac{n(n+1)}{2} n}{n^3} =$$

$$= \frac{1}{16} \underset{n\to\infty}{Lim} \frac{n^3 + n^2}{n^3} = \frac{1}{16}. \tag{53}$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\mu_{muito\times muito} ij}{2n^2}}{n^2} = \frac{\mu_{muito\times muito}}{2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} ij}{n^4} =$$

$$= \frac{\mu_{muito\times muito}}{2} \underset{n\to\infty}{Lim} \frac{\frac{n(n+1)}{2} \frac{n(n+1)}{2}}{n^4} = \frac{\mu_{muito\times muito}}{8} \underset{n\to\infty}{Lim} \frac{\left(n^2 + n\right)^2}{n^4} = \tag{54}$$

$$= \frac{\mu_{muito\times muito}}{8} = \frac{1}{8} \times \frac{9}{16} = \frac{9}{128}.$$

$$\underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\mu_{muito\times muito} i}{2n}}{n^2} = \frac{\mu_{muito\times muito}}{2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} i}{n^3} = \frac{\mu_{muito\times muito}}{2} \underset{n\to\infty}{Lim} \frac{\sum_{i=1}^{n} i \times n}{n^3} =$$

$$= \frac{\mu_{muito\times muito}}{2} \underset{n\to\infty}{Lim} \frac{\frac{n(n+1)}{2} n}{n^3} = \frac{\mu_{muito\times muito}}{4} \underset{n\to\infty}{Lim} \frac{\left(n^2 + n\right) \times n}{n^3} = \tag{55}$$

$$= \frac{\mu_{muito\times muito}}{4} = \frac{1}{4} \times \frac{9}{16} = \frac{9}{64}.$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\mu_{muito\times muito}\,j}{2n}}{n^2}=\frac{\mu_{muito\times muito}}{2}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}j}{n^3}=\frac{\mu_{muito\times muito}}{2}\underset{n\to\infty}{Lim}\frac{\sum_{j=1}^{n}j\times n}{n^3}=$$
$$=\frac{\mu_{muito\times muito}}{2}\underset{n\to\infty}{Lim}\frac{\frac{n(n+1)}{2}n}{n^3}=\frac{\mu_{muito\times muito}}{4}\underset{n\to\infty}{Lim}\frac{\left(n^2+n\right)\times n}{n^3}= \quad (56)$$
$$=\frac{\mu_{muito\times muito}}{4}=\frac{1}{4}\times\frac{9}{16}=\frac{9}{64}.$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\mu_{muito\times muito}}{2}}{n^2}=\frac{\mu_{muito\times muito}}{2}\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}1}{n^2}=\frac{\mu_{muito\times muito}}{2}=\frac{1}{2}\times\frac{9}{16}=\frac{9}{32}. \quad (57)$$

$$\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\mu_{muito\times muito}{}^2}{n^2}=\mu_{muito\times muito}{}^2\underset{n\to\infty}{Lim}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}1}{n^2}=\mu_{muito\times muito}{}^2=\left(\frac{9}{16}\right)^2=\frac{81}{256}. \quad (58)$$

Aplicando-se as equações 46 a 58 na Equação 45, temos que a variância da área esperada de células *Muito* × *Muito* pode ser calculada como apresentado na Equação 59.

$$\sigma^2=\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\delta(muito_1)\times\delta(muito_2)-\mu_{muito\times muito}\right)^2\times p(muito_1)\times p(muito_2)=$$
$$=\frac{1}{144}+\frac{1}{48}+\frac{1}{48}+\frac{1}{48}+\frac{1}{16}+\frac{1}{16}+\frac{1}{48}+\frac{1}{16}+\frac{1}{16}-\frac{9}{128}-\frac{9}{64}-\frac{9}{64}-\frac{9}{32}+\frac{81}{256}= \quad (59)$$
$$=0{,}023871528.$$

A Tabela 4 apresenta os valores de variância para interseção de tipos de células.

Tabela 4. Variância da área esperada correspondente à sobreposição de tipos de células.

Tipos de células	**Vazio**	**Pouco**	**Muito**	**Cheio**
Vazio	0	0	0	0
Pouco	0	0,003038194	0,013454861	0,020833333
Muito	0	0,013454861	0,023871528	0,020833333
Cheio	0	0,020833333	0,020833333	0

Sendo assim, é possível retornar para o usuário um intervalo de confiança da resposta resultante do processamento aproximado da consulta. Considere como exemplo uma consulta que produza os seguintes pares de tipos de células 100 *Pouco* × *Pouco*, 40

Pouco × *Muito*, 70 *Pouco* × *Cheio*, 60 *Muito* × *Muito* e 200 *Cheio* × *Cheio*. Uma resposta com um intervalo de confiança de 95% é apresentada na Figura 30 (por simplicidade, estamos assumindo que todas as células têm a mesma área, igual a 1).

- *P*×*P*:100 × (0,0625 ± 1,96 × (0,0030382/100)$^{1/2}$) = 6,25 ± 1,0803
- *P*×*M*: 40 × (0,1875 ± 1,96 × (0,013454/40)$^{1/2}$) = 7,50 ± 1,4378
- *P*×*C*: 70 × (0,2500 ± 1,96 × (0,020833/70)$^{1/2}$) = 17,50 ± 2,3669
- *M*×*M*: 60 × (0,5625 ± 1,96 × (0,023872/60)$^{1/2}$)= 33,75 ± 2,3457
- *C*×*C*: 200 × 1 = 200 (Células *Cheio* têm area exata!)
- Total: 265 ± 7,2308.

Figura 30. Exemplo de cálculo de área de interseção de polígono × polígono para um intervalo de confiança de 95%.

Assim, a resposta tem uma variância de ±7,2308, o que significa que, para um intervalo de confiança de 95%, a resposta aproximada para uma consulta envolvendo números de células como apresentados no exemplo terá um erro máximo de ±2,7286%, um resultado aceitável para muitas aplicações. Para um intervalo de confiança de 99%, é necessário substituir nas fórmulas o valor 1,96 por 2,56. Neste caso, o valor calculado será de 265 ± 9,5034. A resposta terá uma variação de ±9,5034, o que significa um erro máximo de ±3,5862% em 99% dos casos.

4.7. Operações Aproximadas Usando Assinaturas 4CRS

Esta seção apresenta direções para pesquisa de algoritmos para processamento aproximado das operações apresentadas na Seção 4.2. Além disso, para algumas operações, além de descrições, apresentamos propostas de algoritmos. Com o intuito de facilitar a caracterização das propostas apresentadas, operações semelhantes são descritas na mesma subseção.

4.7.1. Distância

A distância entre dois polígonos pode ser calculada de forma aproximada a partir de suas assinaturas 4CRS computando-se a distância entre as células correspondentes às bordas dos polígonos, ou seja, células *Pouco* e *Muito*. O resultado estimado pode ser retornado como a média da distância mínima e máxima calculadas da seguinte forma. A distância mínima é obtida a partir do cálculo da distância entre as bordas mais externas das células (bordas adjacentes a células *Vazio*), enquanto que a distância máxima é calculada a partir das bordas mais internas destas células (i.e., bordas opostas às mais

externas). As distâncias mínimas e máximas podem ser utilizadas como forma de determinar um intervalo de confiança para a distância aproximada calculada. A Figura 31 apresenta um exemplo de determinação de distância entre dois polígonos empregando suas assinaturas 4CRS. Na Figura 31.a os dois polígonos são exibidos, enquanto que na Figura 31.b são mostradas suas assinaturas 4CRS. A Figura 31.c corresponde a um *zoom* do exemplo de combinações de células para cálculo da distância apresentado na Figura 31.b, evidenciando a determinação das distâncias mínima e máxima entre duas células.

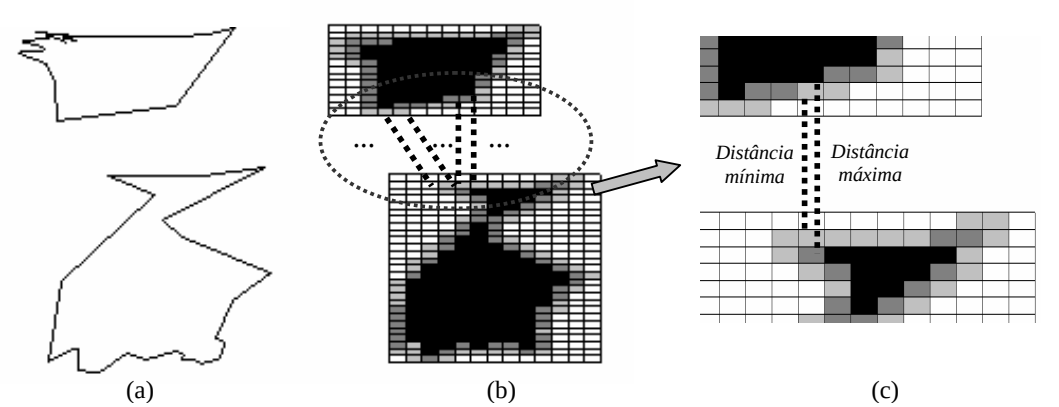


Figura 31. Exemplo de cálculo de distâncias mínima e máxima entre dois polígonos a partir de suas assinaturas 4CRS.

4.7.2. Diâmetro

O diâmetro de um objeto espacial é definido como a maior distância entre qualquer um de seus componentes. Dessa forma, no caso de polígonos, o diâmetro é a maior distância entre as faces que compõem o polígono. O cálculo do diâmetro pode ser realizado utilizando o mesmo algoritmo para cálculo de distância entre polígonos, desde que cada face tenha uma assinatura 4CRS diferente (Figura 32). Se todas as faces do polígono forem representadas em uma única assinatura, é importante que não se considere células da mesma face ao se calcular a distância entre as mesmas.

Figura 32. Exemplo de cálculo de diâmetro de um polígono com 3 faces a partir de suas assinaturas 4CRS.

4.7.3. Perímetro

A operação *perímetro* calcula a soma do tamanho de todos os ciclos de uma região (ou polígono). Se estivermos interessados em computar apenas a soma do tamanho dos ciclos mais externos, não incluindo buracos, nós podemos usar a operação contorno para eliminar os buracos. O cálculo do perímetro do polígono pode ser realizado de forma aproximada a partir de sua assinatura 4CRS. Uma maneira seria computar o perímetro como a média do perímetro externo e do perímetro interno da assinatura. A Figura 33 apresenta um exemplo de cálculo de perímetro de polígono usando assinatura 4CRS. A Figura 33.a exibe o polígono e na Figura 33.b é apresentada a sua assinatura 4CRS. A Figura 33.c apresenta o que seria considerado como perímetro externo da assinatura, enquanto que a Figura 33.d o perímetro interno. O perímetro externo poderia ser calculado como a soma do tamanho das arestas correspondentes a células de tipo diferente de *Vazio* que são adjacentes a células *Vazio* (como a célula c' apresentada na Figura 33.b, por exemplo, cujos lados *Esquerdo* e *Superior* seriam considerados no cálculo do perímetro) ou adjacentes às bordas do MBR da assinatura (como a célula c'' apresentada na Figura 33.b, cujos lados *Esquerdo* e *Superior* também seriam considerados para cálculo do perímetro). Por outro lado, o perímetro interno poderia ser calculado como a soma dos lados das arestas das células adjacentes a células que foram consideradas como parte do perímetro externo.

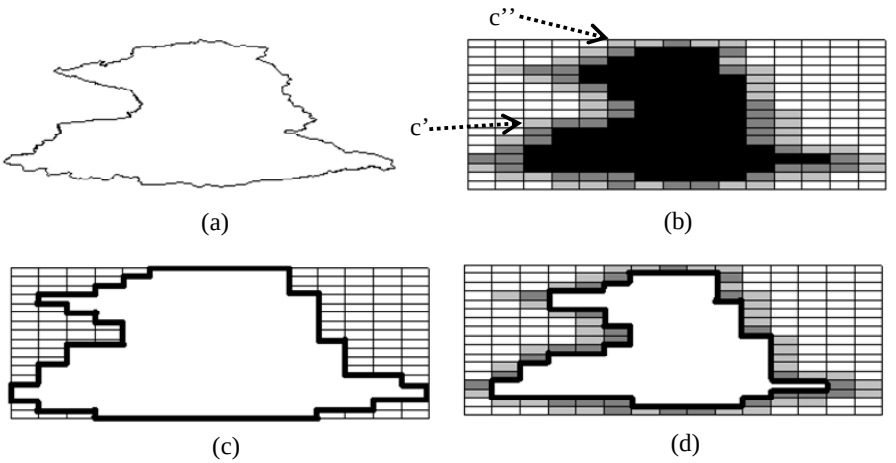


Figura 33. Exemplo de cálculo de perímetro de polígono usando assinatura 4CRS.

4.7.4. Igual e Diferente

No caso de processamento exato, as operações igual e diferente retornam exatamente se dois objetos são iguais ou não, respectivamente. Já em processamento aproximado usando assinaturas 4CRS, nem sempre é possível afirmar com exatidão que dois polígonos são idênticos. De outra forma, nós devemos definir uma função que retorne um valor *d* no intervalo [0,1] que indique o percentual de certeza de igualdade entre os objetos. Esse valor *d* é chamado de "grau de afinidade". Sendo assim, o teste de igualdade entre dois objetos seria realizado comparando-se as células de suas assinaturas 4CRS. Para cada comparação de pares de células um percentual de certeza é atribuído como resultado da comparação. O grau de afinidade final seria obtido pelo somatório desses percentuais dividido pelo total de comparações, se nenhum caso trivial for encontrado. Assim, por exemplo, ao comparar um par de células *Vazio* ou um par de células *Cheio*, podemos afirmar que os polígonos são 100% iguais nestas células, pois não existe área dos polígonos nas células (células *Vazio*), ou as células estão inteiramente contidas nos objetos (células *Cheio*). Por outro lado, no caso de comparações entre células *Pouco* e células *Muito* um raciocínio diferente deve ser empregado. Nossa proposta é usar as mesmas idéias empregadas para o cálculo aproximado de área de interseção de polígonos (Seção 4.5), calculando o valor que indica a probabilidade de dois objetos serem idênticos dentro de uma célula como sendo igual à área esperada correspondente à interseção de pares de células (Tabela 3). Assim, para casos exatos (comparações de células *Vazio* × *Vazio* e *Cheio* × *Cheio*) a igualdade (ou grau de afinidade) é 1, enquanto que para os outros casos o grau de afinidade é igual à área esperada. Por exemplo, dois objetos têm 6,25% de probabilidade de serem iguais quando são comparadas interseção de células *Pouco* × *Pouco*. Da mesma forma, quando compramos duas células *Muito* × *Muito*, os objetos têm 56,25% de chance de serem iguais.

Raciocínio semelhante pode ser utilizado para a operação diferente. Neste caso, se duas células de mesmo tipo forem comparadas, o percentual que indica o quão os objetos são diferentes é igual a 100% menos o percentual que indica o quão eles podem ser iguais, ou seja, o percentual que indica a diferença existente na sobreposição de células *Pouco* × *Pouco* é igual a "100% - 6,25%", enquanto que no caso de células *Muito* × *Muito* este percentual é representado por 100% - 56,25%.

O algoritmo que retorna se dois objetos são iguais é apresentado na Figura 34. O algoritmo retorna 0 se os polígonos não são iguais, caso contrário o algoritmo retorna o grau de afinidade correspondente à igualdade dos objetos. Analogamente, a Figura 35 apresenta uma proposta de algoritmo para retornar se dois objetos são diferentes. O algoritmo retorna 1 quando um caso trivial acontece, indicando que os objetos são "exatamente" diferentes. Caso contrário, é retornado o grau de afinidade correspondente.

```
real igual(assinat4CRS1, assinat4CRS2)
  se assinat4CRS1.tamnhoDoLadoDaCelula ≠
     assinat4CRS2.tamnhoDoLadoDaCelula
    retornar 0;
  se assinat4CRS1.nCelulas • assinat4CRS2.nCelulas
    retornar 0;
  se assinat4CRS1.mbr • assinat4CRS2.mbr
    retornar 0;
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula c1 de assinat4CRS1 faça
     para cada celula c2 de assinat4CRS2 que sobrepoe c1 faça
       se c1.tipo==c2.tipo
          se c1.tipo==VAZIO ou c1.tipo==CHEIO
            grauDeAfinidade += 1;
          senão se c1.tipo==POUCO
            grauDeAfinidade += 0.0625;
          senão
            grauDeAfinidade += 0.5625;
       senão
         retornar 0;
       nComparacoes++;
  retornar grauDeAfinidade / nComparacoes;
```

Figura 34. Algoritmo para estimar se dois polígonos são iguais.

```
real diferente(assinat4CRS1, assinat4CRS2)
  se assinat4CRS1.tamanhoDoLadoDaCelula •
     assinat4CRS2.tamanhoDoLadoDaCelula
    retornar 1;
  se assinat4CRS1.nCelulas • assinat4CRS2.nCelulas
    retornar 1;
  se assinat4CRS1.mbr • assinat4CRS2.mbr then
    retornar 1;
  afinityDegree = 0;
  nComparacoes = 0;
  para cada celula c1 de assinat4CRS1 faça
     para cada celula c2 de assinat4CRS2 que sobrepoe c1 faça
       se c1.tipo==c2.tipo
         se c1.type==VAZIO
            grauDeAfinidade += 1 - 0.0625;
         senão
            grauDeAfinidade += 1 - 0.5625;
       senão
         retornar 1;
       nComparacoes++;
  retornar grauDeAfinidade / nComparacoes;
```

Figura 35. Algoritmo para estimar se dois polígonos são diferentes.

Considere como exemplo uma consulta que produza 100 células *Vazio* × *Vazio*, 100 células *Pouco* × *Pouco*, 60 células *Muito* × *Muito* e 200 células *Cheio* × *Cheio*. Na

Figura 36 é apresentado um exemplo de determinação da igualdade entre dois objetos a partir de suas assinaturas 4CRS. No exemplo, pode-se inferir que os objetos são iguais com probabilidade igual a 74% (340 / 460 = 0,74). De outra forma, na Figura 37 é apresentado um exemplo de determinação de desigualdade entre polígonos. Neste caso, os polígonos têm 26% (120 / 460) de probabilidade de serem diferentes.

- *V*×*V*: 100
- *P*×*P*: 100 × 0,0625 = 6,25
- *M*×*M*: 60 × 0,5625 = 33,75
- C×C: 200 × 1 = 200
- Total: 340

Figura 36. Exemplo de determinação de igualdade entre polígonos a partir das células das assinaturas 4CRS.

- *V*×*V*: 100 × 0 = 0
- *P*×*P*: 100 × (1-0,0625) = 93,75
- *M*×*M*: 60 × (1-0,5625) = 26,25
- *C*×C: 200 × 0 = 0
- Total: 120

Figura 37. Exemplo de determinação de desigualdade entre polígonos a partir das células de suas assinaturas 4CRS.

4.7.5. Disjunção, Área disjunto, Aresta disjunto

Dois objetos são disjuntos se eles não têm nenhuma parte em comum. No caso de área disjuntos, os objetos não têm área em comum, podendo ter sobreposição de arestas. De outra forma, dois objetos são aresta disjuntos se eles não possuem sobreposição entre suas arestas.

A assinatura 4CRS pode ser utilizada para retornar se dois objetos são disjuntos, área disjuntos ou aresta disjuntos. Em alguns casos é possível obter uma resposta exata executando a consulta sobre as assinaturas. De outra forma, um resultado aproximado é retornado.

Ao comparar as assinaturas 4CRS de dois polígonos, se existir apenas sobreposições de células *Vazio* com outros tipos de células, pode-se afirmar com exatidão que os polígonos são disjuntos, área disjuntos e aresta disjuntos. Por outro lado, se existir pelo menos uma sobreposição de célula *Cheio* com célula *Pouco*, *Muito* ou *Cheio* conclui-se que os objetos não são disjuntos e nem área disjunto. Também é possível concluir com exatidão que dois polígonos são aresta disjuntos se não houver sobreposição entre células *Pouco* e *Muito*, ou seja, ou um polígono está completamente fora do outro, ou um está inteiramente contido no outro. Dessa forma, uma resposta aproximada é retornada apenas quando existem interseções entre células *Pouco* e *Muito*,

caso contrário é possível retornar uma resposta exata. Uma proposta para definir um valor aproximado para estes casos é usar a área esperada de sobreposição entre células, atribuindo-se pesos de 100% para os casos de sobreposição de células onde é possível se ter um resultado exato. Para os outros casos atribuem-se pesos iguais aos usados para cálculo de área aproximada de interseção de polígonos (Tabela 3), utilizando o complemento do valor da área esperada de interseção, já que estamos interessados em estimar a não interseção entre os objetos.

A Figura 38 apresenta o algoritmo para determinação se dois objetos são disjuntos. O mesmo algoritmo pode ser utilizado para determinar se dois objetos são área disjuntos. Na Figura 39, o algoritmo para retornar se dois objetos são aresta disjuntos é apresentado.

```
real disjunto(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se interMBR é NULO /*Os MBRs das assinaturas não se interceptam*/
    retornar 1;
  se (assinat4CRS1.tamanhoDoLadoDaCelula <
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = mudarEscala(assinat4CRS1,
                        assinat4CRS2.tamanhoDoLadoDaCelula);
    b4CRS = signat4CRS2;
  senão
    se (assinat4CRS1.tamanhoDoLadoDaCelula >
        assinat4CRS2.tamanhoDoLadoDaCelula)
      b4CRS = assinat4CRS1;
      s4CRS = mudarEscala(assinat4CRS2,
                          assinat4CRS1.tamanhoDoLadoDaCelula);
    senão
      s4CRS = assinat4CRS1;
      b4CRS = assinat4CRS2;
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula b4CRS b que contida em MBRintersec faça
     para cada celula s4CRS s que intercepta b faça
       se b.tipo == VAZIO ou s.tipo == VAZIO
          grauDeAfinidade += 1;
       senão
       se ( (b.tipo == POUCO) e
            (s.tipo == POUCO ou s.tipo == MUITO) ) ou
          ( (b.type == MUITO) e (s.tipo == POUCO))
          grauDeAfinidade += (1 - areaEsperada[s.tipo,b.tipo]);
       senão
          retornar 0;
       nComparacoes++;
   retornar grauDeAfinidade / nComparacoes;
```

Figura 38. Algoritmo para estimar se dois polígonos são disjuntos.

```
real arestaDisjunto(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se interMBR é NULO /*Os MBRs das assinaturas não se interceptam*/
    retornar 1;
  se (assinat4CRS1.tamanhoDoLadoDaCelula <
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = mudarEscala(assinat4CRS1,
                        assinat4CRS2.tamanhoDoLadoDaCelula);
    b4CRS = assinat4CRS2;
  senão
    se (assinat4CRS1.tamanhoDoLadoDaCelula >
        assinat4CRS2.tamanhoDoLadoDaCelula)
      b4CRS = assinat4CRS1;
      s4CRS = mudarEscala(assinat4CRS2,
                          assinat4CRS1.tamanhoDoLadoDaCelula);
    senão
      s4CRS = assinat4CRS1;
      b4CRS = assinat4CRS2;
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula b4CRS b contida em MBRintersec faça
     para cada celula s4CRS s que intercepta b faça
       se b.tipo == VAZIO ou s.tipo == VAZIO
          grauDeAfinidade += 1;
       senão
       se ( (b.tipo == POUCO ou b.tipo == MUITO) e
            (s. tipo == POUCO ou s.tipo == MUITO) )
          grauDeAfinidade += (1 - areaEsperada[s.tipo,b.tipo]);
       senão
          grauDeAfinidade += 1;
       nComparacoes++;
  retornar grauDeAfinidade / nComparacoes;
```

Figura 39. Algoritmo para estimar se dois polígonos são aresta disjuntos.

Considere uma consulta que retorne as seguintes sobreposições de pares de células: 200 células *Vazio* × *Vazio*; 100 células *Pouco* × *Pouco*, 40 células *Pouco* × *Muito*. A Figura 40 apresenta um exemplo de avaliação se dois polígonos são disjuntos. A probabilidade dos objetos serem área disjuntos é de 326,25 / 340 = 0,96 (ou 96% de probabilidade). O mesmo cálculo seria realizado para determinar se dois objetos são área disjuntos.

- *V*×V: 200 × 1 = 200
- *P*×P: 100 × (1 - 0,0625) = 93,75
- *P*×*M*: 40 × (1 - 0,1875) = 32,50
- Total: 326,25

Figura 40. Exemplo de avaliação se dois polígonos são disjuntos.

Considere a mesma consulta utilizada no exemplo apresentado na Figura 40, sendo que adicionando 100 sobreposições de células *Cheio* × *Cheio*, como apresentado na Figura 41. Neste caso, não é possível retornar que os dois objetos são aresta disjuntos por existir sobreposição de células *Cheio*. O peso atribuído para células *Cheio* × *Cheio* deve ser considerado no cálculo com peso igual a 1, já que se os polígonos tiverem interseção apenas em células cheias então os polígonos são aresta disjuntos. A

probabilidade dos polígonos serem aresta disjuntos é de aproximadamente 97% (426,25 / 440 = 0,968).

- *V*×*V*: 200 × 1 = 200
- *P*×*P*: 100 × (1 - 0,0625) = 93,75
- *P*×*M*: 40 × (1 - 0,1875) = 32,50
- *C*×*C*: 100 × 1 = 100

Figura 41. Exemplo de avaliação se dois polígonos são aresta disjuntos.

4.7.6. Dentro, Aresta Dentro, Vértice Dentro

A partir da assinatura 4CRS de dois polígonos, é possível afirmar com exatidão que um polígono P_1 está dentro de outro polígono P_2 se todas as células da assinatura 4CRS de P_1, diferentes de *Vazio*, são sobrepostas por células *Cheio* da assinatura 4CRS de P_2. De outra forma, se existir sobreposição de pelo menos uma célula não *Vazio* de P_1 com uma célula *Vazio* de P_2 então P_1 não está dentro de P_2. Já nos casos de ocorrerem sobreposições entre células *Pouco* e *Pouco* ou *Pouco* e *Muito* ou *Muito* e *Muito*, não é possível obter uma resposta exata. Assim, é necessário definir um valor aproximado para estes casos de sobreposições de células. Novamente, nossa proposta é utilizar a área esperada (Tabela 3) para retornar uma resposta aproximada para a consulta. Observe que o mesmo algoritmo pode ser utilizado para determinar se um polígono está dentro do outro ou se um polígono tem suas arestas dentro de outro polígono. Todavia, a determinação de que um polígono é vértice-dentro de outro polígono não pode ser definida de forma aproximada a partir de sua assinatura 4CRS, pois informações a respeito dos vértices do polígono não são armazenadas na assinatura. Sendo assim, é possível apenas retornar se um polígono não é vértice dentro de outro polígono quando não existe interseção entre eles. Todavia, se existir interseção, não é possível retornar uma resposta aproximada com um intervalo de confiança.

A Figura 42 apresenta um algoritmo para retornar se um polígono P_1 está dentro de outro polígono de acordo com suas assinaturas 4CRS. É importante notar que, de acordo com o algoritmo para geração de MBR-2^n, apresentado na Subseção 3.1.2 (Figura 6), se o tamanho de célula da assinatura 4CRS de P_1 for maior do que o tamanho de célula da assinatura 4CRS de P_2, então o polígono P_1 tem extensão maior do que o polígono P_2. Dessa forma, P_1 não está dentro de P_2.

```
real dentro(assinat4CRS1, assinat4CRS2)
  if (assinat4CRS1.tamanhoDoLadoDaCelula >
      assinat4CRS2.tamanhoDoLadoDaCelula)
    retornar 0;
  senão se (assinat4CRS1.tamanhoDoLadoDaCelula <
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = mudarEscala(assinat4CRS1,
                        assinat4CRS2.tamanhoDoLadoDaCelula);
    b4CRS = assinat4CRS2;
  senão
    s4CRS = assinat4CRS1;
    b4CRS = assinat4CRS2;
  MBRinter = MBRintersecao(s4CRS, b4CRS);
  se interMBR é NULO /*Não existe MBR de interseção*/
    retornar 1;
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula b4CRS b contida em interMBR faça
     para cada celula s4CRS s que intercepta b faça
       se b.tipo == VAZIO e s.tipo • VAZIO
          retornar 0;
       senão
       se (b.tipo == POUCO ou b.type == MUITO)
          se (s.tipo == POUCO ou s.type == MUITO)
            grauDeAfinidade += areaEsperada[s.tipo,b.tipo];
          senão se (s.tipo == VAZIO)
            grauDeAfinidade += 1;
          senão /*s.tipo == CHEIO*/
            retornar 0;
       senão /*b.tipo == CHEIO × qualquer s.tipo*/
          grauDeAfinidade += 1;
       nComparacoes++;
  retornar grauDeAfinidade / nComparacoes;
```

Figura 42. Algoritmo para avaliar se um polígono está dentro de outro polígono a partir de suas assinaturas 4CRS.

Considere uma consulta que retorne as seguintes sobreposições de pares de células: 200 células *Vazio* × *Vazio*; 100 células *Pouco* × *Pouco*, 40 células *Pouco* × *Muito*, 70 células *Pouco* × *Cheio*, 60 células *Muito* × *Muito* e células 200 *Cheio* × *Cheio*. A Figura 43 apresenta um exemplo de avaliação se o polígono está dentro do outro. A probabilidade do polígono estar dentro do outro é de 48% (321,25 / 670 = 0,48).

- *V*×*V*: 200 × 1 = 200
- *P*×*P*:100 × 0,0625 = 62,25
- *P*×*M*: 40 × 0,1875 = 7,5
- *P*×*C*: 70 × 0,2500 = 17,50
- *M*×*M*: 60 × 0,5625 = 33,75
- *C*×*C*: 200 × 1 = 200
- Total: 321,25.

Figura 43. Exemplo de avaliação se um polígono está dentro de outro polígono.

4.7.7. Intercepta e Interseção

A operação intercepta retorna se dois polígonos se interceptam, já a operação interseção retorna o polígono resultante da interseção.

O uso da assinatura 4CRS para retornar se dois polígonos se interceptam foi amplamente estudado em ZIMBRAO e SOUZA (1998). Todavia, naquele trabalho a assinatura 4CRS foi empregada como filtro responsável por reduzir o número de objetos que teriam suas representações geométricas testadas. Neste trabalho, propomos não acessar os objetos reais mesmo para as situações em que não é possível retornar uma resposta exata para a consulta. Neste caso, uma resposta aproximada é retornada e o grau de afinidade da resposta é calculado de acordo com a área esperada de interseção entre as células das assinaturas 4CRS (Tabela 3). A Figura 44 apresenta o algoritmo para avaliar de forma aproximada se dois polígonos se interceptam. Observe que em vários casos é possível retornar uma resposta exata.

Considere uma consulta que retorne as seguintes sobreposições de pares de células: 200 células *Vazio* × *Vazio*; 100 células *Pouco* × *Pouco*, 40 células *Pouco* × *Muito*. A Figura 44 apresenta um exemplo de teste de interseção de polígonos. A probabilidade de interseção é de 75% (279,75 / 370 = 0,75).

No caso do algoritmo para retornar o polígono resultante da interseção de dois polígonos a partir das assinaturas 4CRS dos mesmos pode-se adotar a seguinte abordagem: criar uma nova assinatura a partir das assinaturas 4CRS sendo testadas e gerar o polígono correspondente conectando-se os pontos médios das células da borda da nova assinatura (células *Pouco* e *Muito*). O tipo de cada célula da nova assinatura pode ser ajustado segundo os valores da Tabela 3, ou seja, se o resultado da interseção é um valor no intervalo (50%, 100%) então o tipo da célula da nova assinatura é *Muito*, se for um valor no intervalo (0%, 50%] o tipo da nova célula é *Pouco*. Por outro lado, 0% de interseção e 100% de interseção definem células *Vazio* e *Cheio*, respectivamente. A Figura 46 apresenta um algoritmo para computar a interseção entre dois polígonos a partir de suas assinaturas 4CRS. É importante ressaltar que, além do polígono que representa a interseção, também é retornado o grau de afinidade que mostra quão bem o polígono computado a partir das assinaturas 4CRS (polígono aproximado) representa o polígono que corresponderia à respostas exata.

```
real intercepta(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se (assinat4CRS1.tamanhoDoLadoDaCelula <
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = mudarEscala(assinat4CRS1,
                        assinat4CRS2.tamanhoDoLadoDaCelula);
    b4CRS = assinat4CRS2;
  senão
    se (assinat4CRS1.tamanhoDoLadoDaCelula >
        assinat4CRS2.tamanhoDoLadoDaCelula)
      b4CRS = assinat4CRS1;
      s4CRS = mudarEscala(assinat4CRS2,
                          assinat4CRS1.tamanhoDoLadoDaCelula);
    senão
      s4CRS = assinat4CRS1;
      b4CRS = assinat4CRS2;
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula b4CRS b e celula s4CRS s se interceptam em
                                                MBRintersec faça
       se (b.tipo == CHEIO) e (s.tipo • VAZIO)
         retornar 1;
       senão
         se (b.tipo == MUITO) e
            ( (s.tipo==MUITO) ou (s.tipo==CHEIO)
           retornar 1;
         senão
           se (b.tipo == POUCO) e (s.tipo==CHEIO)
             retornar 1;
           senão
             se (b.tipo == VAZIO) e (s.tipo ==VAZIO) entao
                grauDeAfinidade += 1;
             senão
               grauDeAfinidade += areaEsperada[s.tipo,b.tipo];
        nComparacoes++;
  retornar grauDeAfinidade / nComparacoes;
```

Figura 44. Algoritmo para avaliar se dois polígonos se interceptam.

- *V*×*V*: 200 × 1 = 200
- *P*×*P*: 100 × 0,0625 = 62,25
- *P*×*M*: 70 × 0,2500 = 17,50
- Total: 279,75.

Figura 45. Exemplo de teste de interseção de polígonos em que não foi possível retornar um valor exato testando-se as assinaturas 4CRS dos mesmos.

```
Poligono intersecao(assinat4CRS1, assinat4CRS2, grauDeAfinidade)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se (assinat4CRS1.tamanhoDoLadoDaCelula <
      assinat4CRS2.tamanhoDoLadoDaCelula)
    s4CRS = mudarEscala(assinat4CRS1,
                        assinat4CRS2.tamanhoDoLadoDaCelula);
    b4CRS = assinat4CRS2;
  senão
    se (assinat4CRS1.tamanhoDoLadoDaCelula >
        assinat4CRS2.tamanhoDoLadoDaCelula)
      b4CRS = assinat4CRS1;
      s4CRS = mudarEscala(assinat4CRS2,
                          assinat4CRS1.tamanhoDoLadoDaCelula);
    senão
      s4CRS = assinat4CRS1;
      b4CRS = assinat4CRS2;
  /*Criar assinatura 4CRS com apenas células VAZIO*/
  n4CRS = criarAssinatura(MBRintersec, VAZIO);
  grauDeAfinidade = 0;
  nComparacoes = 0;
  para cada celula b4CRS b e celula s4CRS s e celula n4CRS n
           que se interceptam em MBRintersec faça
       se (b.tipo == CHEIO)
         grauDeAfinidade = 1;
         n.tipo = s.tipo;
       se (s.tipo == CHEIO)
         grauDeAfinidade = 1;
         n.tipo = b.tipo;
       senão se (b.tipo == VAZIO) or (s.tipo == VAZIO)
         grauDeAfinidade = 1;
         n.type = VAZIO;
       senão se (b.tipo == MUITO) e (s.tipo ==MUITO)
         grauDeAfinidade = areaEsperada[s.tipo,b.tipo];
         n.type = MUITO;
       senão se /* ( (b.tipo == MUITO) e (s.tipo ==POUCO) ) ou */
                /* ( (s.tipo == MUITO) e (b.tipo ==POUCO) ) ou */
                /* ( (b.tipo == POUCO) e (s.tipo ==POUCO) ) */
         grauDeAfinidade = areaEsperada[s.tipo,b.tipo];
         n.type = POUCO;
       nComparacoes++;
  grauDeAfinidade = grauDeAfinidade / nComparacoes;
  retornar criarPoligono(n4CRS);
```

Figura 46. Algoritmo para computar a interseção entre dois polígonos a partir de suas assinaturas 4CRS.

4.7.8. Overlay

O operador *overlay* é definido por GÜTING e SCHNEIDER (1995) como uma operação que permite que uma partição do plano seja superposta sobre outra, de forma que elas sejam combinadas em regiões área disjuntas. Partições são dadas por conjuntos de objetos com um atributo do tipo polígono. O conjunto de objetos resultantes contém um objeto para cada novo polígono obtido da interseção de uma partição com um polígono da outra partição. Observe que os polígonos de uma partição não necessariamente precisam cobrir completamente o plano. Dessa forma, é possível que um polígono de uma partição não intercepte nenhum polígono da outra partição. Neste caso ele não será parte de nenhum novo objeto. Um exemplo da operação *overlay* é apresentado na Figura 47. O algoritmo para computar a interseção de dois polígonos a

partir de suas assinaturas 4CRS apresentado na Figura 46 pode ser empregado para computar o *overlay* de duas partições do plano.

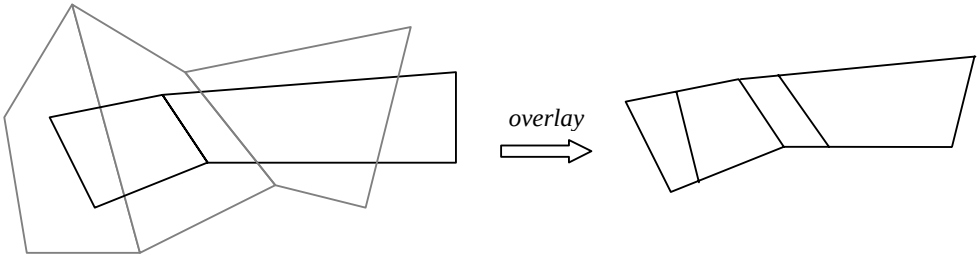


Figura 47. Exemplo de *overlay* de duas partições do plano.

4.7.9. Adjacente, Existe Borda em Comum, Borda em Comum

Dois polígonos são adjacentes se têm pelo menos uma parte de suas bordas em comum. Neste trabalho, estamos propondo usar o mesmo algoritmo empregado para retornar de forma aproximada se dois polígonos têm borda em comum para retornar também se dois polígonos são adjacentes.

Uma proposta de implementação de algoritmo para a operação para avaliar se dois polígonos possuem borda em comum seria empregar as medidas de área esperada (Tabela 3) utilizada para calcular a área aproximada de interseção de polígonos. A borda de um polígono é formada por segmentos, ou seja, a borda de um polígono não possui área. Além disso, a borda em comum a dois polígonos nas suas assinaturas 4CRS só pode ocorrer no caso de sobreposição de células *Pouco* ou *Muito*, que são as células que delimitam suas bordas. Sendo assim, é possível retornar uma resposta exata para o caso negativo em que não há sobreposição destes tipos de células. Por outro lado, em todos os casos positivos os valores retornados são aproximados devido às células envolvidas serem *Pouco* e *Muito*. A nossa proposta de algoritmo é retornar como probabilidade de existir borda em comum entre dois polígonos avaliando-se a soma dos pesos correspondentes às sobreposições de células *Pouco* × *Pouco*, *Pouco* × *Muito*, *Muito* × *Muito* dividido pelo total de sobreposições que envolvem células *Pouco* ou *Muito*. O algoritmo é apresentado na Figura 48.

```
real bordarEmComum(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se MBRintersec é NULO /*Não existe MBR de interseção*/
    retornar 1;
  se (assinat4CRS1.tamanhoDoLadoDaCelula ==
      assinat4CRS2.tamanhoDoLadoDaCelula) then
    s4CRS = assinat4CRS1;
    b4CRS = assinat4CRS2;
  else
    s4CRS = menorLadoDeCelula(assinat4CRS1, assinat4CRS2);
    b4CRS = maiorLadoDeCelula(assinat4CRS1, assinat4CRS2);
  grauDeAfinidade = 0;
  nSobreposicoes = 0;
  para cada celula b4CRS b contida em MBRintersec faça
     para cada celula s4CRS s que intercepta celula b faça
       se b.tipo == POUCO or b.tipo == MUITO
          nSobreposicoes += 1;
          se s.tipo == POUCO ou s.tipo == MUITO
            grauDeAfinidade += areaEsperada[s.tipo,b.tipo];
       senão
         se s.tipo == POUCO ou s.tipo == MUITO
            nSobreposicoes += 1;
            se b.tipo == POUCO ou b.tipo == MUITO
              grauDeAfinidade += areaEsperada [s.tipo,b.tipo];
  retornar grauDeAfinidade / nSobreposicoes;
```

Figura 48. Algoritmo para retornar se dois polígonos possuem borda em comum.

Uma proposta de algoritmo para retornar de forma aproximada a borda em comum de dois polígonos é a adaptação do algoritmo apresentado na Figura 48 de forma a gerar os segmentos correspondentes à borda em comum pela concatenação dos pontos médios das células *Pouco* e *Muito* adjacentes que têm sobreposição. A probabilidade de esta ser a borda em comum entre os polígonos pode ser também retornada, calculando-se da mesma forma que no algoritmo apresentado na Figura 48.

4.7.10. Soma

O operador soma computa a união de dois objetos (Operação que Retorna Valor Espacial – Subseção 4.2.3) ou de um conjunto de objetos (Operação Espacial Aplicada sobre Conjunto de Objetos – Subseção 4.2.4).

A união de dois polígonos pode ser computada de forma aproximada a partir das assinaturas 4CRS. Como resultado uma nova assinatura 4CRS é produzida representando a união das assinaturas que representam os polígonos.

Nesta subseção apresentamos uma proposta para computar a união de dois polígonos, a qual pode ser utilizada para retornar a união de vários polígonos computando uniões dois a dois.

Se dois polígonos não têm interseção de MBR, o resultado da união são os próprios polígonos, com suas faces representando faces do polígono união. Por outro lado, quando existe interseção entre os MBRs dos polígonos, é gerada uma nova

assinatura 4CRS, computada de acordo com o algoritmo apresentado na Figura 49. A partir da assinatura 4CRS (ou das assinaturas 4CRS geradas) computa-se o polígono que representa a união conectando-se os pontos médios das células *Pouco* e *Muito* da nova assinatura (função *computarPoligono*).

```
Poligono uniao(assinat4CRS1, assinat4CRS2)
  MBRintersec = MBRintersecao(assinat4CRS1, assinat4CRS2);
  se MBRintersec é NULO /*Não existe MBR de interseção*/
    poligono1 = computarPoligono(assinat4CRS1);
    poligono2 = computarPoligono(assinat4CRS2);
    poligono.adicionarFaces(poligono1);
    poligono.adicionarFaces(poligono2);
    retornar poligono;
  se (assinat4CRS1.tamanhoDoLadoDaCelula >
      assinat4CRS2.tamanhoDoLadoDaCelula)
    b4CRS = assinat4CRS1;
    s4CRS = mudarEscala(assinat4CRS2,
                        assinat4CRS1.tamanhoDoLadoDaCelula);
  senão
    se (assinat4CRS2.tamanhoDoLadoDaCelula >
        assinat4CRS1.tamanhoDoLadoDaCelula)
      b4CRS = assinat4CRS2;
      s4CRS = mudarEscala(assinat4CRS1,
                          assinat4CRS2.tamanhoDoLadoDaCelula);
  MBRuniao = computarMBRUniao(s4CRS.mbr,b4CRS.mbr)
  /*Criar assinatura 4CRS com apenas células VAZIO*/
  n4CRS = criarAssinatura(MBRuniao, b4CRS.tamanhoDoLadoDaCelula
                          VAZIO);
  para cada celula b4CRS b que intercepta celula n4CRS n faça
    n.tipo = b.tipo;
  para cada celula s4CRS s que intercepta celula n4CRS n faça
    se n.tipo == VAZIO ou s.tipo == CHEIO
      n.tipo = s.tipo;
    senão se n.tipo == POUCO and s.tipo == MUITO então
      n.tipo = s.tipo;
  retornar computarPoligono(n4CRS);
```

Figura 49. Algoritmo para computar a união de dois polígonos a partir de suas assinaturas 4CRS.

4.7.11. Subtração

A diferença de um polígono P_1 em relação a um polígono P_2 é formada pela parte de P_1 que não tem interseção com P_2. Sendo assim, uma proposta para calcular a diferença de dois polígonos (P_1 e P_2) a partir de suas assinaturas 4CRS é ajustar para *Vazio* as células da assinatura de P_1 que têm sobreposição com células da assinatura 4CRS de P_2 de tipo igual a *Muito* e *Cheio*. Para computar o polígono a partir da assinatura resultante deve-se considerar que células *Cheio* podem fazer parte da borda do novo polígono além de células *Pouco* e *Muito*. O algoritmo para computar a diferença entre dois polígonos a partir de suas assinaturas 4CRS é apresentado na Figura 50.

```
Poligono subtracao(assinat4CRS1, assinat4CRS2)
  interMBR = intersectionMBR(s4CRS, b4CRS);
  se interMBR is NULO /*Does not exist MBR intersection*/
    retornar computarPoligono(assinat4CRS1);
  se (assinat4CRS1.tamanhoDoLadoDaCelula >
      assinat4CRS2.tamanhoDoLadoDaCelula)
    b4CRS =  assinat4CRS1;
    s4CRS =  mudarEscala(assinat4CRS2,
                         assinat4CRS1.tamanhoDoLadoDaCelula);
  senao
  se (assinat4CRS2.tamanhoDoLadoDaCelula >
      assinat4CRS1.tamanhoDoLadoDaCelula)
     s4CRS = mudarEscala(assinat4CRS1,
                          assinat4CRS2.tamanhoDoLadoDaCelula);
     b4CRS = assinat4CRS2;
  senão
      b4CRS = assinat4CRS1;
      s4CRS = assinat4CRS2;
  para cada celula b de b4CRS b contida no MBRintersec faça
     para cada celula s4CRS2 s que intercepta celula b faça
       se s.tipo == MUITO ou s.tipo == CHEIO
         b.tipo = VAZIO;
  poligono = computarPoligono(s4CRS1,
                ConsiderarTambémCélulasCheiasParaComputarBorda);
  retornar poligono;
```

Figura 50. Algoritmo para computar a diferença entre dois polígonos a partir de suas assinaturas 4CRS.

4.7.12. Fusão

O operador *fusão* mescla os valores de um dado (conjunto de) atributo(s) espacial(is) com base na igualdade dos valores de outro (conjunto de) atributo(s) não espacial(is). Para cada grupo de valores não espaciais um (conjunto de) novo(s) valor(es) espacial(is) é criado como sendo a união geométrica do conjunto de valores espaciais do grupo. Na Figura 51, uma partição de municípios de acordo com o uso da terra é apresentada na parte esquerda da figura. Após a aplicação do operador *fusão*, Municípios vizinhos, com o mesmo uso da terra, são mesclados em uma única região (isto é, a borda em comum é apagada). O algoritmo apresentado na Figura 50 pode ser usado para computar a união.

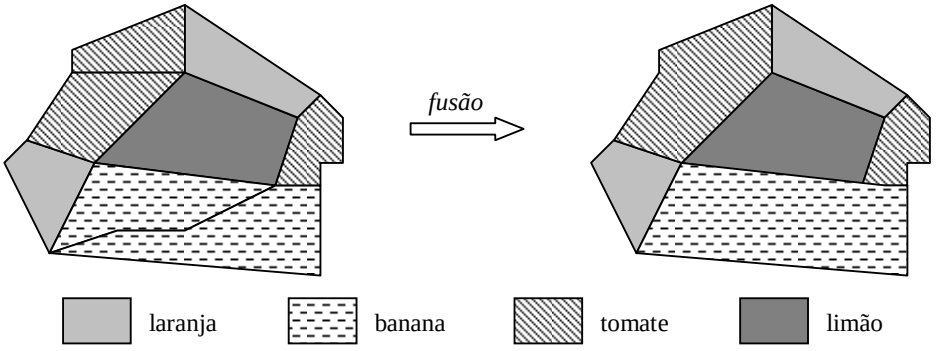


Figura 51. Fusão de municípios vizinhos de acordo com o uso da terra.

4.7.13. Contorno

O cálculo do contorno de um polígono pode ser realizado de forma aproximada a partir de sua assinatura 4CRS de forma semelhante ao cálculo de perímetro apresentado na Subseção 4.7.3. Nós podemos obter o contorno do polígono ligando-se os pontos médios das células *Pouco* e *Muito* da assinatura 4CRS do polígono, e podemos assumir como contorno máximo o polígono correspondente ao perímetro externo (Figura 33.c) e como contorno mínimo o polígono que corresponde ao perímetro interno (Figura 33.d).

4.7.14. Objeto mais próximo

O operador *objeto mais próximo* retorna o objeto (em um conjunto de objetos) que é mais próximo de um valor espacial de referência. No caso de processamento aproximado empregando assinaturas 4CRS, pode-se empregar a rotina que calcula a distância aproximada (Subseção 4.7.1) para retornar o objeto mais próximo. O percentual de certeza (ou grau de afinidade) da resposta poderia ser calculado como "1 – (distância máxima – distância mínima) / distância". Assim, por exemplo, se o objeto mais próximo está a distância 95 do valor de referência, e a distância mínima é 80 e a distância máxima 110, o grau de afinidade da resposta é 1 - (110-80)/95 = 1 – 30 / 95 = 0,68, ou seja, 68 % de grau de afinidade de que o objeto retornado é o objeto mais próximo do valor de referência.

4.7.15. Decomposição

O operador *decomposição* tem como entrada um conjunto de objetos que possuem um atributo espacial. Ele produz uma nova coleção de objetos da seguinte forma: para cada objeto do conjunto operando seu atributo é decomposto em

componentes (um componente pode ser um ponto, segmento, face, etc.). Se um objeto possui *n* componentes, então *n* cópias do objeto original são produzidas cada qual contendo um componente do objeto original como valor do seu atributo espacial.

Neste trabalho estamos tratando de polígonos representados por suas assinaturas 4CRS. As componentes dos polígonos são faces, e as componentes da assinatura de um polígono são assinaturas para cada face. Dessa forma, da aplicação do operador decomposição sobre a assinatura 4CRS de um polígono produz novos objetos que possuem um atributo cujo valor é a assinatura 4CRS de uma das faces do polígono original.

5. Testes Experimentais

Este capítulo é dedicado à apresentação dos resultados experimentais da avaliação do uso da assinatura 4CRS para processamento aproximado. Os seguintes algoritmos foram avaliados:

- Algoritmo para calcular a área aproximada de polígono (Seção 4.3);
- Algoritmo para calcular a área aproximada de polígono dentro de janela (Seção 4.4);
- Algoritmo para calcular a área aproximada de interseção de polígono × polígono (Seção 4.5).

Com o objetivo de avaliar a eficiência de nossas propostas nós comparamos o processamento aproximado contra o processamento exato. A avaliação dos algoritmos foi realizada segundo os critérios propostos por GIBBONS *et al.* (1997) para avaliar módulos para processamento aproximado de consultas. Os critérios propostos são:

- Cobertura: conjunto de consultas para as quais é possível prover respostas aproximadas;
- Tempo de resposta: tempo necessário para prover uma resposta aproximada para consulta;
- Precisão: precisão das respostas retornadas, e intervalos de confiança para a resposta obtida;
- Tempo de atualização: tempo necessário para manter os resumos dos dados (sinopses) atualizados.

O atendimento ao critério “cobertura” foi demonstrado no Capítulo 4, onde foram enumeradas várias operações para processamento aproximado (Seção 4.2), e foram apresentadas direções para pesquisa e propostas de algoritmos para processamento aproximado para estas operações usando assinaturas 4CRS (Seção 4.7). Além disso, nas seções 4.3, 4.4 e 4.5 foram detalhadas propostas para os três algoritmos cujos resultados experimentais serão apresentados neste capítulo. Na Seção 4.6 foram propostas fórmulas para cálculo do intervalo de confiança para as respostas retornadas por estes algoritmos.

Neste capítulo serão apresentados resultados experimentais relacionados aos testes dos três algoritmos enumerados anteriormente segundo os demais critérios de medida propostos por GIBBONS *et al*. (1997): espaço de armazenamento, tempo para atualização das assinaturas; tempo de resposta; e, precisão das respostas retornadas.

Na Seção 5.1, são apresentadas as características dos dados usados em nossos experimentos. A Seção 5.2 descreve o ambiente no qual os testes foram executados e as Árvores-R* empregadas. A Seção 5.3 mostra as características das assinaturas 4CRS, e, finalmente, na Seção 5.4, os resultados dos experimentos são relatados.

5.1. Conjuntos de Dados

Nos nossos experimentos foram usados dados reais de polígonos consistindo de limites de municípios, grupos de setores censitários, mapas geológicos e mapas hidrográficos do estado de Iowa (Estados Unidos da América), disponíveis no sítio "http://www.igsb.uiowa.edu/nrgis/gishome.htm", e dados de municípios do Brasil (IBGE, 1996). A fim de simular grandes volumes de dados, os dados de Iowa foram replicados seis vezes, seguindo a proposta de BRINKHOFF *et al.* (1994). Os polígonos originais foram deslocados aleatoriamente nas coordenadas *x* e *y*. No caso dos municípios do Brasil, foi realizada uma replicação (chamada de Municípios do Brasil') com o objetivo de executar o teste municípios do Brasil × municípios do Brasil' para avaliação do algoritmo que calcula a área aproximada de interseção de polígono × polígono. As características dos dados são apresentadas na Tabela 5.

Tabela 5. Conjuntos de dados testados

Conjuntos de dados		**Tamanho (KB)**	**# pol.**	**# segmentos**	**# segmentos médio**
Iowa	Setores censitários	38.824	17.844	1.764.588	98
	Topografia	61.748	20.070	3.780.552	188
	Mapas hidrográficos	6.904	2.544	475.434	186
	Municípios	25.288	12.216	1.059.438	86
	Mapas geológicos	21.856	9.984	640.428	64
Brasil	Municípios	9.840	4.645	399.002	85
	Municípios'	9.840	4.645	399.002	85
Média		24.900	10.278	1.216.921	113

*5.2. Ambiente de Teste e características das Árvores-R**

Testes foram executados em um computador Pentium IV 1.8 GHz com 512 MB de memória principal. Foi definido um tamanho de página igual 2.048 bytes para operações de Entrada/Saída.

Três tipos de testes forma executados, um para cada algoritmo.

- Teste do algoritmo que calcula a área aproximada de polígono;
- Teste do algoritmo que calcula a área aproximada de polígono dentro de janela;
- Teste do algoritmo que calcula a área aproximada de interseção de polígono $\times$ polígono.

No primeiro teste, a área aproximada e a área exata de todas as assinaturas 4CRS e dos polígonos foram calculadas e comparadas. O propósito deste teste foi apenas mostrar uma medida de qualidade das assinaturas, já que a área exata pode ser obtida de forma simples, armazenando-a em um atributo para cada polígono, e lendo a área deste atributo sempre que necessário, sem precisar processar o objeto ou a assinatura.

Para o segundo e o terceiro testes, a Árvore-R* (BECKMANN *et al*., 1990) foi escolhida como método de acesso espacial, cujo objetivo foi de reduzir o espaço de busca. Em outras palavras, a Árvore-R* foi usada para que fossem processados apenas os objetos que tivessem pelo menos interseção de MBR. Esta escolha deveu-se ao amplo uso da Árvore-R*, bem como aos ótimos resultados encontrados na literatura. Os métodos de acesso tradicionais empregam o MBR (*Minimum Bouding Rectangle* ou Retângulo Mínimo Envolvente). A execução dos métodos de acesso produz o que chamamos de conjunto de pares candidatos, já que este conjunto contém tanto os pares de polígonos que pertencem ao conjunto solução como também aqueles que têm apenas interseção de MBR. Da mesma forma como fora realizado nos trabalhos de BRINKHOFF *et al*. (1994) e ZIMBRAO e SOUZA (1998), para os nossos testes foram geradas Árvores-R* armazenando as assinaturas 4CRS como parte da chave dos polígonos, ou seja, as assinaturas 4CRS foram armazenadas nos nós folha do índice (Árvore-R*). Esta é uma abordagem razoável, pois as assinaturas são computadas apenas uma vez.

Dessa forma, no segundo teste, a consulta de janela foi primeiro executada contra o índice (Árvore-R*), e apenas os objetos que tinham interseção de MBR com a janela tiveram computadas as suas áreas aproximada e exata. Além disso, nós armazenamos um atributo área em cada nó folha da Árvore-R*. Assim, se o polígono estivesse completamente dentro da janela, sua área era obtida deste atributo, sem ser processada. Nós geramos 100 janelas aleatórias para testar o algoritmo que calcula a

área aproximada de polígono dentro de janela para os conjuntos de dados. As características das janelas foram omitidas devido à simplicidade de descrição.

O teste do algoritmo para cálculo da área aproximada de interseção de polígono x polígono pode ser descrito de acordo com as propostas de processamento de junção espacial apresentadas na Seção 1.3. No processamento aproximado, apenas os passos MAE e de filtro foram executados (Figura 3). Por outro lado, no processamento exato, nós executamos os passos MAE e de refinamento (Figura 1). Ou seja, depois de encontrar os objetos que têm interseção de MBR, as representações exatas dos objetos são lidas e processadas, e uma resposta exata é retornada. Tabela 6 apresenta as junções executadas para testar o algoritmo que computa a área aproximada de interseção de polígono x polígono.

Tabela 6. Junções executadas para testar o algoritmo que computa a área aproximada de interseção de polígono x polígono

Junções	**Conjunto de dados 1**	**Conjunto de dados 2**
Junção-1	Municípios do Brasil	Municípios do Brasil’
Junção-2	Municípios (Iowa)	Setores censitários (Iowa)
Junção-3	Municípios (Iowa)	Mapas geológicos (Iowa)
Junção-4	Municípios (Iowa)	Mapas hidrográficos (Iowa)
Junção-5	Setores censitários (Iowa)	Mapas hidrográficos s (Iowa)
Junção-6	Mapas hidrográficos (Iowa)	Mapas geológicos (Iowa)

O processamento aproximado de consultas foi realizado usando os algoritmos apresentados nas seções 4.3, 4.4 e 4.5, enquanto que o processamento exato foi realizado usando os algoritmos implementados na biblioteca GPC (*General Polygon Clipping*) disponível no sítio “http://www.cs.man.ac.uk/aig/staff/alan/ software/#gpc”.

Para realização dos testes nós geramos dois tipos de Árvores-R*: Árvores-R* armazenando assinaturas 4CRS nos nós folha a serem usadas no processamento aproximado; e, Árvores-R* sem armazenar as assinaturas 4CRS, usadas durante a execução da consulta exata. Dessa forma, os tamanhos das Árvores-R* sem armazenar as assinaturas são menores do que as Árvores-R* que as armazenam, consequentemente o número de acessos a disco no primeiro passo é menor quando as árvores sem armazenar as assinaturas são empregadas. Essa é uma abordagem justa, dado que não é necessário acessar as assinaturas 4CRS quando o processamento exato é executado. As características das Árvores-R* são apresentadas na Tabela 7. A coluna “Tipo” indica que as características apresentadas são de Árvores-R* armazenando assinaturas 4CRS ou sem armazená-las.

Tabela 7. Características das Árvores-R*

Conjunto de dados		Tipo	Tamanho (KB)	Tempo de geração (seg.)	Taxa de utilização dos nós folha (%)	Altura	# folhas
Iowa	Setores censitários	4CRS	2.124	19,04	69,98	3	1045
		-	1.160	17,93	69,81	3	570
	Mapas hidrográficos	4CRS	334	2,24	68,33	3	162
		-	162	2,14	75,35	2	79
	Municípios	4CRS	1.546	12,95	68,70	3	760
		-	800	11,97	69,50	3	392
	Mapas geológicos	4CRS	1.258	9,55	68,41	3	617
		-	644	9,32	70,46	3	316
Brasil	Municípios	4CRS	586	4,66	71,15	3	286
		-	284	4,07	75,05	3	138
	Municípios'	4CRS	582	4,92	71,63	3	284
		-	284	4,11	75,05	3	138
Média		4CRS	1.289	8,89	69,70	3	525
		-	663	8,26	72,54	3	272

5.3. Assinaturas 4CRS

Para gerar assinaturas 4CRS, é necessário escolher o número máximo de células (ZIMBRAO e SOUZA, 1998). Intuitivamente, quanto maior o número de células mais próxima do polígono original é a sua assinatura. Por outro lado, processar assinaturas 4CRS muito grandes pode levar a custos elevados de UCP e operações de Entrada/Saída. Com o objetivo de avaliar os efeitos de diferentes tamanhos de células, nós executamos testes experimentais do algoritmo que calcula a área aproximada de interseção de polígonos com números máximos de células iguais a 250, 500, 1000, 1500 e 2000. Este algoritmo foi escolhido por ser o de maiores custos tanto computacional como de operações de entrada e saída entre os três algoritmos testados.

O processamento aproximado foi avaliado contra o processamento exato. Os experimentos demonstraram que assinaturas com número máximo de células igual a 250 têm menor custo de armazenamento, mas a precisão não é a melhor. Por outro lado, as estimativas são melhores quando assinaturas com número máximo de células igual a 2000 são processadas; todavia, os custos de UCP e operações de Entrada/Saída são também maiores, devido ao tamanho das assinaturas. As figuras 52, 53, 54 e 55 resumem os resultados dos testes experimentais, mostrando:

- Requisitos de armazenamento: percentual resultante da fração do tamanho das assinaturas 4CRS dividido pelo tamanho dos conjuntos de dados (Figura 52);

- Erro das respostas aproximadas: percentual correspondente à diferença entre o valor aproximado e o valor exato dividido pelo valor exato (Figura 53);
- Tempo: percentual de tempo necessário para executar a consulta aproximada em relação ao tempo utilizado pelo processamento exato (Figura 54);
- Números de acessos a disco: percentual correspondente aos números de acessos a disco necessários para executar o processamento aproximado em relação ao processamento exato (Figura 55).

Na Seção 5.4, nós apresentamos em detalhes resultados quando foi escolhido 500 como número máximo de células, o qual produziu respostas aproximadas com média de erros e intervalos de confiança aceitáveis.

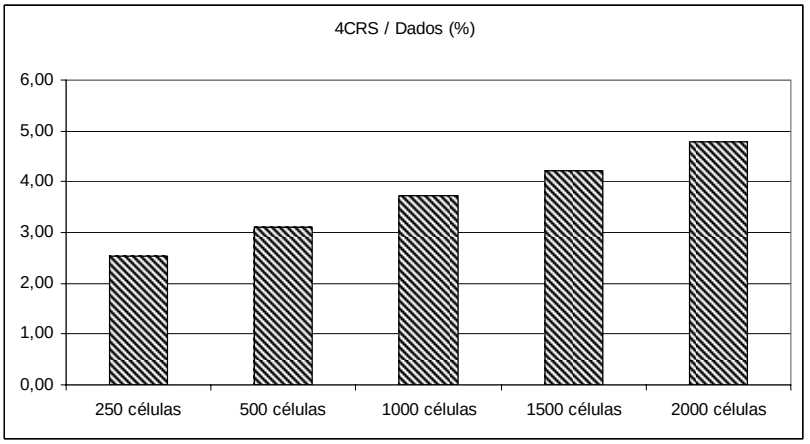


Figura 52. Requisitos de armazenamento para grades números máximos de células iguais a 250, 500, 1000, 1500 e 2000 para o algoritmo que computa a área aproximada de interseção de polígono × polígono.

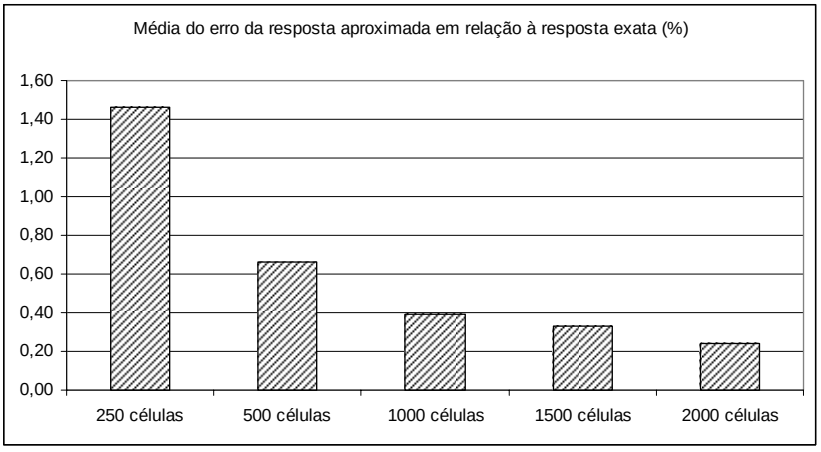


Figura 53. Requisitos de precisão para grades com números máximos de células iguais a 250, 500, 1000, 1500 e 2000 para o algoritmo que computa a área aproximada de interseção de polígono × polígono.

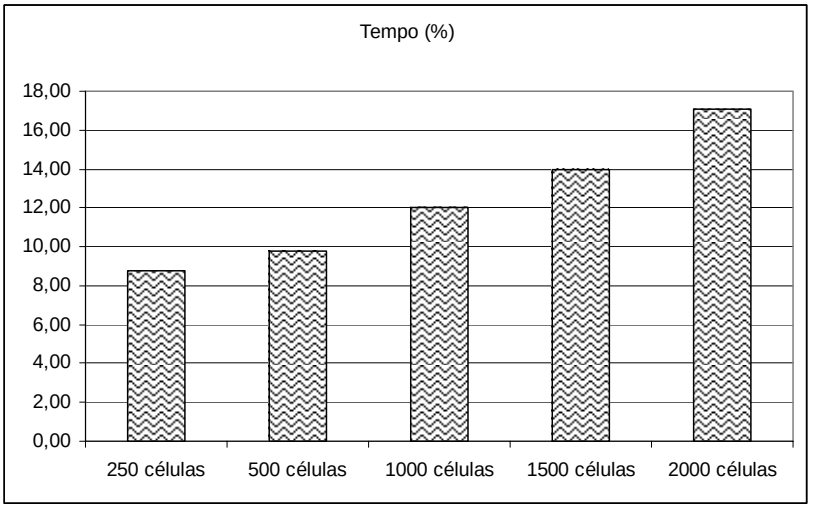


Figura 54. Requisitos de tempo para grades com números máximos de células iguais a 250, 500, 1000, 1500 e 2000 para o algoritmo que computa a área aproximada de interseção de polígono × polígono.

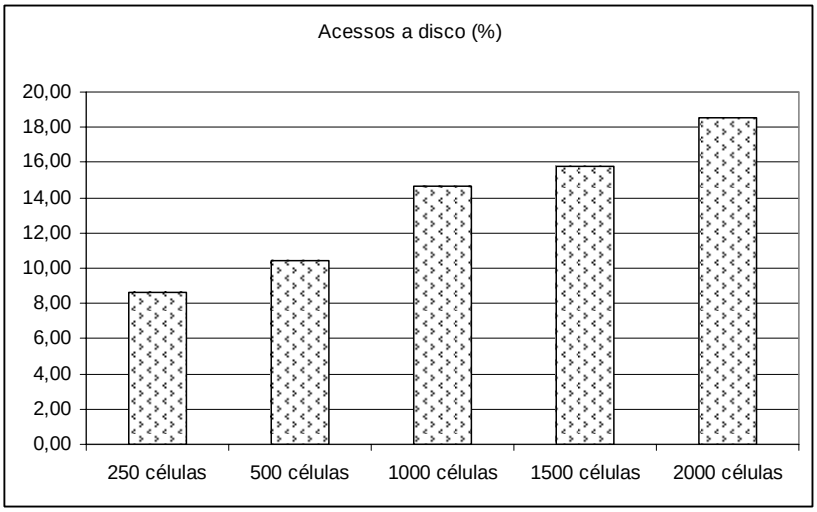


Figura 55. Requisitos de números de acesso a disco para grades com números máximos de células iguais a 250, 500, 1000, 1500 e 2000 para o algoritmo que computa a área aproximada de interseção de polígono × polígono.

A Tabela 8 apresenta as características das assinaturas 4CRS com número de máximo de células igual a 500. É importante notar que, neste caso, para armazenar assinaturas 4CRS dos dados reais utilizados nos nossos experimentos é necessário apenas em média 2,98% do espaço requerido para armazenar os dados reais. Ou seja, é necessário aproximadamente 30 vezes mais espaço para armazenar os dados reais do que para armazenar as assinaturas 4CRS. Além disso, o tempo de geração das assinaturas é muito pequeno. Para os dados reais utilizados nos nossos testes experimentais, a média de tempo de geração da assinatura foi de 15,42 segundos variando de 3,34 segundos para o menor arquivo de dados a 44,89 segundos para o maior arquivo de dados. Deve-se ressaltar que, para o tempo de geração das assinaturas, deve-se considerar não só o tamanho do arquivo, como também o número de segmentos dos objetos e o número máximo de células da assinatura.

Tabela 8. Características das assinaturas 4CRS com número máximo de células igual a 500

Conjuntos de dados		Dados (KB)	4CRS (KB)	4CRS / dados (%)	Tempo de geração (seg.)
Iowa	Setores censitários	38.824	1163	3,00	20,47
	Mapas hidrográficos	6.904	169	2,45	3,34
	Mapas topográficos	61.748	1455	2,36	44,89
	Municípios	25.288	838	3,31	15,5
	Mapas geológicos	21.856	676	3,09	12,09
Brasil	Municípios	9.840	329	3,34	5,84
	Municípios'	9.840	329	3,34	5,84
Média		24.900	708	2,98	15,42

5.4. Resultados Experimentais

Esta seção apresenta em detalhes os resultados experimentais obtidos com os testes dos algoritmos para processamento aproximado de consultas usando assinaturas 4CRS com um número máximo de células igual a 500. Nós avaliamos os seguintes critérios: precisão da resposta aproximada, incluindo intervalos de confiança; tempo de processamento; e número de acessos a disco. Os requisitos de espaço de armazenamento foram apresentados na Tabela 8 (Seção 5.3).

Os resultados obtidos demonstraram a eficiência no uso da assinatura 4CRS para processamento aproximado devido ao pequeno erro das respostas obtidas, ao baixo tempo de processamento da consulta e ao reduzido número de acessos a disco.

No caso do teste do algoritmo que computa a área aproximada de polígonos, o erro médio das respostas aproximadas é de 1,59%, enquanto que os intervalos de confiança de 95% e 99% têm erros médios iguais a 2,83% e 3,72% respectivamente (Tabela 9, coluna "Erro e intervalo de confiança"). Em outras palavras, as respostas aproximadas têm uma diferença em média de apenas 1,59% da resposta exata. Além disso, a fim de demonstrar a precisão da resposta retornada, um intervalo de confiança também é retornado para o usuário e, para este algoritmo, no caso de um intervalo de confiança de 95%, o erro é no máximo de 2,83%, enquanto que, para um intervalo de confiança de 99%, o erro é no máximo de 3,72%.

No caso do tempo de processamento, o algoritmo que computa a área aproximada de polígono é cerca de 2,6 vezes mais rápido do que o algoritmo que calcula o valor exato, pois necessita de 38,88% do tempo necessário para executar o processamento exato (Tabela 9 - coluna "Tempo de Processamento"). É importante ressaltar que o propósito deste teste foi de apenas mostrar uma medida de qualidade das

assinaturas, já que a área exata pode ser obtida de forma simples armazenando-a em um atributo para cada polígono, e lendo a área deste atributo sempre que necessário, sem precisar processar o objeto ou a assinatura.

Tabela 9. Resultados experimentais do teste do algoritmo que calcula a área aproximada de polígono.

	Erro e intervalo de confiança			**Tempo de processamento**		
Consultas	**Erro (%)**	**I. C. 95%**	**I.C. 99%**	**Proc. Aprox.**	**Proc. Exato**	**%**
Setores censitários	1,07	2,21	2,90	1,062	3,032	35,03
Mapas geológicos	1,53	3,18	4,17	0,609	1,078	56,49
Mapas hidrográficos	1,25	2,71	3,56	0,203	0,735	27,62
Municípios de Iowa	0,78	1,42	1,87	0,906	1,828	49,56
Mapas Topográficos	3,94	4,40	5,78	1,172	6,172	18,99
Municípios do Brasil	0,98	3,08	4,05	0,328	0,719	45,62
Média	1,59	2,83	3,72	0,71	2,26	38,88

No caso do algoritmo que computa a área aproximada de polígono dentro de janela, para cada conjunto de dados nós geramos 100 janelas aleatórias com tamanho do lado igual a 12,25% do tamanho do lado do MBR envolvendo o conjunto dos dados. O erro médio das respostas obtidas foi de 1,22%, enquanto que os intervalos de confiança de 95% e 99% tiveram erros médios de 1,69% e 2,22%, respectivamente (Tabela 10 – colunas “Erro e intervalo de confiança”). O processamento aproximado consumiu em média 6,54% do tempo necessário para computar a resposta exata, ou seja, o processamento aproximado foi cerca de 15 vezes mais rápido que o processamento exato (Tabela 10 – colunas “Tempo de processamento”).

O tempo total de execução pode não ser uma medida muito confiável de ganho de processamento, pois é totalmente dependente dos algoritmos empregados para realizar os cálculos. Além disso, o *cache* do Sistema Operacional pode influenciar no tempo de processamento. Dessa forma, também apresentamos o número total de acessos a disco utilizados, pois os objetos processados têm que ser ao menos lidos do disco. Na coluna “Número de acessos a disco” da Tabela 10, o número de acessos a disco realizados por ambos os processamentos são apresentados.

Tabela 10. Resultados experimentais do algoritmo que computa a área aproximada de polígono dentro de janela para janelas aleatórias com lado igual a 4% do lado do MBR dos dados

	Erro e intervalo de confiança			Tempo de processamento			Números de acessos a disco		
Consultas	**Erro (%)**	**I. C. 95%**	**I.C. 99%**	**Proc. Aprox.**	**Proc. Exato**	**%**	**Proc. Aprox.**	**Proc. Exato**	**%**
Setores censitários	0,46	0,57	0,74	2,42	27,83	8,70	379	300	126,33
Municípios de Iowa	0,29	0,28	0,36	1,61	21,13	7,62	286	246	116,26
Mapas Topográficos	1,60	3,13	4,11	2,41	115,30	2,09	405	731	55,40
Municípios do Brasil	2,53	2,78	3,66	1,20	15,53	7,75	147	157	93,63
Média	1,22	1,69	2,22	1,91	44,95	6,54	304	359	97,91

Para o processamento do algoritmo que calcula a área aproximada de polígono dentro de janela, nós armazenamos um atributo área em cada nó folha da Árvore-R*, representando a área do polígono. Assim, se o polígono estivesse completamente dentro da janela, sua área era obtida deste atributo, sem ser processada. A Tabela 11 apresenta algumas informações importantes relacionadas com este fato. Nesta tabela nós incluímos informações do número de objetos processados e número de objetos que não foram processados, pois estavam contidos na janela e a área foi lida do atributo. Sendo assim, em média apenas 7,71% dos objetos foram processados de forma aproximada. Isso explica o fato do processamento aproximado ter número médio de acessos a disco praticamente igual ao do processamento exato. Em alguns casos o número de acesso a disco do processamento aproximado foi maior do que o do processamento exato. Além disso, outro fato que contribuiu para o número de acessos a disco ser muito semelhante foi que as Árvores-R* utilizadas para executar o processamento aproximado têm taxas de utilização dos nós folha menores do que as taxas de utilização dos nós folhas das Árvores-R* utilizadas para processamento exato, pois as primeiras armazenam assinaturas 4CRS em seus nós folha, sendo necessário mais acessos a disco.

Tabela 11. Número de objetos processados pelo algoritmo que computa a área aproximada de polígono dentro de janela para janelas aleatórias com lado igual a 4% do lado do MBR dos dados

Consultas	**# objs.**	**# objs. dentro janela**	**# objs. proc.**	**%**
Setores censitários	2333	2196	137	5,87
Municípios de Iowa	1592	1455	137	8,61
Mapas Topográficos	2595	2426	169	6,51
Municípios do Brasil	873	787	86	9,85
Média	1848	1716	132,3	7,71

Outro teste experimental que realizamos foi avaliar os efeitos do aumento do tamanho da janela em relação ao MBR do conjunto dos dados (Tabela 12). Aumentando-se o tamanho do lado da janela o erro médio diminuiu (coluna “Erro

médio" - Tabela 12). O erro médio foi de 1,55% para janelas aleatórias com tamanho igual a 4% do tamanho do MBR dos dados, enquanto que para janelas com tamanho igual a 75%, o erro médio foi de 0,15%. Isso é explicado pelo fato de que, com o aumento do tamanho da janela, um maior número de objetos se localiza dentro da janela e as áreas são lidas do atributo área de cada polígono, cujos valores são exatos. Este fato é evidenciado na coluna "% de objetos processados", que mostra que o número de objetos realmente processados foi de 12,31%, para janelas com 4% de tamanho do MBR dos dados, e de apenas 2,40%, no caso de janelas de 75% de tamanho. Ou seja, no primeiro caso aproximadamente 87% dos objetos que tiveram interseção com a janela não foram processados, sendo o valor da área lido diretamente do atributo, enquanto que no segundo caso 97,60% dos objetos não foram processados.

Tabela 12. Resultados experimentais do algoritmo que computa a área aproximada de polígono dentro de janela considerando diferentes tamanhos de janelas geradas aleatoriamente

% tamanho do lado da janela	Erro médio	# médio de objetos por janela	# médio de objetos dentro janela	# médio de objetos processados	% de objetos processados
4%	1,55	788	691	97	12,31
12,25%	1,22	1848	1716	132	7,14
25%	0,53	3220	3062	159	4,92
50%	0,27	4901	4743	158	3,23
75%	0,15	5897	5755	142	2,40

Para avaliar o algoritmo que calcula a área de interseção aproximada de polígono × polígono, nós executamos as junções apresentadas na Tabela 6, calculando a área de interseção do conjunto de dados 1 e do conjunto de dados 2. Cada junção foi executada 20 vezes, e para cada execução nós geramos uma janela aleatória a fim de que fossem considerados apenas os pares de objetos que estivessem nesta janela. Para avaliar os efeitos do número de objetos retornados em cada junção, nós executamos dois testes diferentes. Em um dos testes as janelas aleatórias foram geradas com tamanho igual a 4% do lado do MBR envolvendo todo o espaço dos conjuntos de dados sendo testados, enquanto que no outro teste a janela foi gerada com tamanho de 12,25%. Os resultados são apresentados nas Tabela 13 e Tabela 14. Dado que os valores de ambas as tabelas são muito semelhantes, nós iremos analisar apenas os resultados correspondentes ao segundo teste (Tabela 14). A diferença mais relevante entre os testes está no fato de que, no segundo teste, cada janela é interceptada por mais objetos do que o número de interseções do primeiro teste. Como resultado, o número de células das

assinaturas 4CRS consideradas no cálculo do intervalo de confiança é maior, sendo este mais próximo da resposta aproximada.

Tabela 13. Resultados experimentais correspondentes a 20 execuções da área de interseção do conjunto de dados 1 × conjunto de dados 2 × janela aleatória com tamanho de 4% do lado do MBR envolvendo o espaço dos dados

	Erro e intervalo de confiança			Tempo de processamento			Número de acessos a disco			Média de objetos por janela
Junções	**Erro (%)**	**I. C. 95%**	**I.C. 99%**	**Proc. Aprox.**	**Proc. Exato**	**%**	**Proc. Aprox**	**Proc. Exato**	**%**	
Junção-1	0,78	2,97	3,91	6,28	73,03	8,60	2138	26826	7,97	1813
Junção-2	0,30	0,53	0,70	14,62	93,20	15,69	8691	50979	17,05	2590
Junção-3	0,83	1,39	1,82	12,48	109,34	11,41	5440	44289	12,28	2551
Junção-4	0,26	1,23	1,62	8,21	75,10	10,94	5747	27064	21,23	1591
Junção-5	0,44	1,29	1,70	17,38	110,85	15,67	4470	33862	13,20	1935
Junção-6	0,85	1,24	1,63	20,66	95,22	21,70	2736	22419	12,20	1267
Média	0,58	1,44	1,90	13,27	92,79	14,00	4870	34240	13,99	1958

Tabela 14. Resultados experimentais correspondentes a 20 execuções da área de interseção do conjunto de dados 1 × conjunto de dados 2 × janela aleatória com tamanho de 12.25% do lado do MBR envolvendo o espaço dos dados

	Erro e intervalo de confiança			Tempo de processamento			Número de acessos a disco			Média de objetos por janela
Junções	**Erro (%)**	**I.C. 95%**	**I.C. 99 %**	**Proc. Aprox.**	**Proc. Exato**	**%**	**Proc. Aprox.**	**Proc. Exato**	**%**	
Junção-1	1,05	2,48	3,26	18,00	272	6,62	6166	62353	9,89	4907
Junção-2	0,30	0,31	0,40	40,64	398	10,22	24512	133509	18,36	7394
Junção-3	0,79	0,82	1,08	33,68	392	8,60	15737	108508	14,50	6940
Junção-4	0,18	0,75	0,99	21,84	250	8,74	13378	65645	20,38	4305
Junção-5	0,45	0,74	0,97	46,14	390	11,84	13055	87123	14,98	5581
Junção-6	0,74	0,75	0,98	86,01	405	21,22	9504	59656	15,93	3886
Média	0,59	0,97	1,28	41,05	351	11,21	13725	86132	15,67	5502

Os resultados experimentais relacionados ao algoritmo que computa a área aproximada de polígono × polígono demonstraram a eficiência de nossa abordagem. O erro médio das respostas aproximadas foi de 0,59%, enquanto que os intervalos de confiança de 95% e 99% tiveram erros médios de 0,97% e 1,28%, respectivamente (Tabela 14 – colunas “Erro e intervalo de confiança”). O processamento aproximado é em média 9 vezes mais rápido do que o processamento exato, já que necessita de apenas 11% do tempo requerido para executar o processamento exato (Tabela 14 – colunas “Tempo de processamento”). Além disso, é necessário apenas 16% do número de acessos a disco do processamento exato para executar o processamento aproximado. Ou seja, o processamento exato requer em média 6 vezes mais acessos a disco do que o processamento aproximado.

6. Conclusões

Este trabalho propõe o uso de assinaturas *raster* para processamento aproximado de consultas em banco de dados espaciais. O objetivo é prover um resultado estimado muito próximo do valor exato, juntamente com um intervalo de confiança, em um tempo muito menor do que o necessário para computar a resposta exata. Isto pode ser obtido, por exemplo, evitando ou minimizando o número de acessos ao dado real. Nós propomos o processamento de consultas envolvendo polígonos usando suas assinaturas 4CRS, ou seja, processando representações compactas e aproximadas dos objetos, não acessando os dados reais. Dessa forma, as geometrias exatas dos objetos não são processadas durante o processamento da consulta, o qual é o passo mais custoso no processamento de consultas espaciais, já que requer a busca e transferência de grandes objetos do disco para a memória principal (BRINKHOFF *et al.*, 1994; LO e RAVISHANKAR, 1996). Além disso, os algoritmos utilizados para realizar o processamento exato utilizam complexas operações com elevado tempo de execução, para decidir se objetos atendem ao predicado da consulta (BRINKHOFF *et al.*, 1993). Existem muitos cenários e aplicações que podem se beneficiar de uma resposta rápida aproximada em detrimento a uma resposta exata e demorada, desde que a primeira tenha a precisão desejada, como apresentado na Seção 1.2.

Nós propomos usar assinaturas 4CRS para processamento aproximado de operações espaciais. Na Subseção 4.2 nós enumeramos estas operações, segundo a classificação proposta por GÜTING e SCHNEIDER (1995). Na Seção 4.7, nós apresentamos direções para pesquisa, implementação e avaliação destas operações, bem como, propostas de algoritmos para algumas das mesmas. Nas seções 4.3, 4.4 e 4.5, nós apresentamos em detalhes a implementação e avaliação das seguintes propostas de algoritmos para processamento aproximado:

- Algoritmo para computar área aproximada de polígono;
- Algoritmo para computar área aproximada de polígono dentro de janela;
- Algoritmo para computar área aproximada de interseção de polígono × polígono.

Na Seção 4.6, nós propomos fórmulas para calcular o intervalo de confiança das respostas retornadas por estes algoritmos. É importante também enfatizar que, na Seção

3.3, nós apresentamos uma nova proposta de algoritmo eficiente para geração de assinaturas 4CRS.

Nós avaliamos nossa abordagem comparando o processamento aproximado contra o processamento exato de acordo com os critérios propostos por GIBBONS *et al.* (1997) para avaliar módulos para processamento aproximado de consultas. Estes critérios são: cobertura; tempo de resposta; precisão; tempo de atualização; requisitos de armazenamento. Os resultados obtidos para os testes realizados foram excelentes, e demonstraram a eficiência do uso da assinatura 4CRS para processamento aproximado de consultas em banco de dados espaciais.

É possível prover respostas aproximadas para um amplo conjunto de consultas utilizando as assinaturas 4CRS, atendendo desta forma o critério de cobertura. Nas seções 4.2 e 4.7 nós enumeramos várias operações, além de apresentar direções para pesquisa e propostas de algoritmos para as mesmas, utilizando assinaturas 4CRS para processamento aproximado.

O tempo de processamento e o número de acessos a disco requeridos para executar o processamento aproximado são muito menores do que os necessários para executar o processamento exato (critério de tempo de resposta), além disso, as respostas aproximadas têm um erro muito pequeno (critério de precisão). Nós apresentamos em detalhes resultados experimentais para assinaturas de tamanhos pequenos. Na Seção 5.4, nós apresentamos a avaliação dos algoritmos propostos. Nós avaliamos o algoritmo que computa a área aproximada de polígono como uma medida de qualidade das assinaturas 4CRS, já que a área exata pode ser armazenada como um atributo do polígono, ou seja, seu valor pode ser lido diretamente deste atributo sem necessidade de executar qualquer processamento, seja este exato ou aproximado. O processamento aproximado foi 2,6 vezes mais rápido do que o processamento exato, e o erro da resposta aproximada foi de apenas 1,59%, enquanto que os intervalos de confiança de 95% e 99% tiveram em média erro máximo de 2,83% e 3,72%, respectivamente. No caso do algoritmo que computa a área aproximada de polígono dentro de janela, o erro foi em média de 1,22%, enquanto que intervalos de confiança de 95% e 99% tiveram erros médios de 1,69% e 2,22%, respectivamente. O processamento aproximado foi cerca de 15 vezes mais rápido do que o processamento exato. Entretanto, o número de acessos a disco foi praticamente o mesmo tanto para processamento aproximado como para processamento exato. Isto ocorreu porque nós criamos um atributo área para cada entrada de nó folha

da Árvore-R* para armazenar a área do polígono que a entrada aponta. Assim, se o polígono estivesse completamente dentro da janela, tanto o processamento exato como o aproximado não foram executados, e a área do polígono foi obtida diretamente a partir deste atributo. Dessa forma, o número de objetos processados foi pequeno. Além disso, outro fato que contribuiu para o número de acessos a disco ser muito semelhante foi que as Árvores-R* utilizadas para executar o processamento aproximado têm taxas de utilização dos nós folha menores do que as taxas de utilização dos nós folha das Árvores-R* utilizadas para processamento exato, pois as primeiras armazenam assinaturas 4CRS em seus nós folha, sendo necessário mais acessos a disco.

Os resultados mais importantes foram obtidos na avaliação do algoritmo que computa a área aproximada de interseção de polígono × polígono. Este é o algoritmo mais importante entre as três propostas apresentadas, já que ele requer o uso de operações que necessitam de elevado tempo de UCP e têm alto custo de operações de Entrada/Saída para procurar e ler os objetos do disco. As respostas aproximadas tiveram em média um erro de 0,6%, enquanto que os intervalos de confiança de 95% e 99% em média tiveram erros máximos de 0,97% e 1,28% respectivamente, o que é uma precisão suficiente para muitas aplicações. Além disso, o processamento aproximado foi de 5 a 15 vezes mais rápido do que o processamento exato em tempo de resposta, e de 5 a 10 vezes mais rápido em relação ao número de acessos a disco.

O critério de tempo de atualização é atendido pelo fato de que é possível gerar assinaturas 4CRS rapidamente. Utilizando o algoritmo por nós proposto (Seção 3.3), para os conjuntos de dados reais utilizados nos testes experimentais, foi necessário em média apenas 15,52 segundos para geração das assinaturas, ou seja, considerando a média do tamanho dos dados, são necessários apenas 15,52 segundos para gerar assinaturas a partir de um arquivo de 25 MB de dados, sendo processados mais do que 10.000 objetos. Obviamente, o tamanho do arquivo não é a única variável a influenciar no tempo de geração das assinaturas, outros fatores que devem ser considerados são os números de segmentos dos objetos e o número máximo de células da grade. Nos nossos testes utilizamos conjuntos de dados com números de segmentos médios variando de 64 a 188 segmentos e as assinaturas foram geradas considerando grades de no máximo 500 células.

Na Seção 5.3 (Tabela 8), é mostrado que a assinatura 4CRS atende aos requisitos de armazenamento, pois é necessário em média 30 vezes menos espaço de

armazenamento para armazenar as assinaturas 4CRS do que para armazenar os dados reais.

Como trabalhos futuros nós planejamos avaliar o processamento aproximado empregando assinaturas 4CRS contra outros algoritmos para computar a resposta exata. Pretendemos também implementar e avaliar o uso das assinaturas 4CRS para os outros algoritmos apresentados na Seção 4.7. Além disso, também temos interesse em pesquisar, implementar e avaliar algoritmos envolvendo outros tipos de dados, por exemplo, pontos e polilinhas, além de operações envolvendo diferentes tipos de objetos, tais como ponto e polígono, polilinha e polígono, ponto e polilinha. Em AZEVEDO *et al*. (2003) é proposta uma assinatura *raster* para polilinhas, chamada de Assinatura Raster Direcional de Cinco Cores (5CDRS ou *Five-Color Directional Raster Signature*). Naquele trabalho, a assinatura 5CRDS foi empregada como filtro geométrico no processamento da junção espacial em múltiplos passos envolvendo conjuntos de polilinhas. A 5CDRS obteve ótimos resultados perante outras propostas existentes na literatura. Nossa proposta é avaliar esta assinatura para processamento aproximado de consultas envolvendo polilinhas. Além disso, como a 5CDRS foi elaborada baseada na 4CRS, podemos utilizar ambas as assinaturas para processamento aproximado de consultas envolvendo polilinhas e polígonos. O processamento exato empregando a 5CDRS e a 4CRS como filtro geométrico no processamento da junção espacial foi avaliado por MONTEIRO *et al*. (2004) e os resultados obtidos também foram excelentes.

Nós também planejamos avaliar o uso de mais cores na geração das assinaturas 4CRS, por exemplo, oito cores. Nós acreditamos que é possível prover melhores precisões e intervalos de confiança mais próximos da resposta retornada, empregando mais cores na assinatura. Por outro lado, apesar de trazer o custo extra de mais um bit para armazenamento das cores, os requisitos de armazenamento podem ser mantidos aplicando-se métodos de compactação das assinaturas geradas. Outra alternativa é avaliar assinaturas com menos cores, por exemplo, apenas 3 cores (3CRS ou *Three-Color Raster Signature*), onde células *Pouco* e *Muito* são substituídas por uma célula chamada de inconclusiva. Esta assinatura pode ser computada de forma mais rápida do que assinaturas 4CRS, dado que não é necessário calcular a área do polígono dentro de cada célula, mas apenas saber que a célula é atravessada pelo polígono para classificá-las como inconclusiva. Consequentemente, nós podemos usar uma abordagem tardia, ou

seja, computar a assinatura apenas quando ela for necessária, ou gerar a assinatura "*on the fly*" para polígonos resultantes da execução de operações espaciais. Outra proposta de trabalho futuro é pesquisar algoritmos para determinar o número de células que leve a assinatura que melhor represente o polígono, baseado na complexidade do polígono (BRINKHOFF *et al.*, 1995). Uma abordagem simples seria computar assinaturas iniciando com número de células igual a um, e ir incrementando o número de células até que a proporção entre a área aproximada (células *Pouco* e *Muito*) e área exata atinjam um determinado limite.

7. Bibliografia

AGARWAL, S., R., DESHPANDE, P. M., GUPTA, A., *et al.*, 1996, "On the computation of multidimensional aggregates". In: *Proceedings of 22th International Conference on Very Large Data Bases*, pp. 506-521, Mumbai (Bombay), India, Sep.

AGRESTI, A, 1990, *Categorical Data Analysis*. 1 ed. New York, Wiley and Sons.

AOKI, P. M., 1998, "Generalizing "search" in generalized search trees". In: *Proceedings of the Fourteenth International Conference on Data Engineering*, pp. 380-389, Orlando, Florida, USA, Feb.

ARONOFF, S., 1989, *Geographic Information Systems*. 1 ed. Ottawa, Canada, WDL Publications.

AZEVEDO, L. G., MONTEIRO, R. S., ZIMBRAO, G., SOUZA, J. M., 2003, "Polyline Spatial Join Evaluation Using Raster Approximation". In: *GeoInformatica, Kluwer Academic Publishers* , vol. 7, n. 4, pp. 315-336.

AZEVEDO, L. G., MONTEIRO, R. S., ZIMBRAO, G., SOUZA, J. M., 2004, "Approximate Spatial Query Processing Using Raster Signatures". In: *Proceedings of VI Brazilian Symposium on GeoInformatics*, pp. 403-421, Campos do Jordao, Brazil, Nov.

AZEVEDO, L. G., ZIMBRAO, G., SOUZA, J. M., GÜTING, R. H., 2005, "Estimating the Overlapping Area of Polygon Join". In: *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, to be published, Angra dos Reis, Brazil, Aug.

BAYER, R., MCCREIHT, C., 1972, "Organization and maintenance of large ordered indexes". *Acta Informatica*, v. 1, n. 3, pp. 173-189.

BARBARA, D., DUMOUCHEL, W., C. FALOUTSOS, *et al*., 1997, "The New Jersey data reduction", *Bulletin of the Technical Committee on Data Engineering*, v. 20, n. 4 (Dec), pp. 3-45.

BECKMANN, N., KRIEGEL, H. P., SCHNEIDER, R., *et al*., 1990, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles". In:

Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, pp. 322-331, Atlantic City, NJ, USA, May.

BISHOP, Y., FIENBERG, S., HOLLAND, P., 1975, *Discrete Multivariate Analysis: Theory and Practice*. 1 ed. Cambridge, Massachusetts, USA, MIT Press.

BRINKHOFF, T., KRIEGEL, H. P., SCHNEIDER, R, 1993, "Comparison of Approximations of Complex Objects Used for Approximation-based Query Processing in Spatial Database Systems". In: *Proceedings of the Ninth International Conference on Data Engineering*, pp. 40-49, Vienna, Austria, Apr.

BRINKHOFF, T., KRIEGEL, H. P., SCHNEIDER, R., SEEGER, B., 1994, "Multi-step Processing of Spatial Joins", *ACM SIGMOD Record*, v. 23, n.2 (Jun), pp. 197-208.

CANNATARO, M., GUZZO, A., PUGLIESE, A., 2002, "Knowledge Management and XML: Derivation of Synthetic Views over Semistructured Data", *ACM SIGAPP Applied Computing Review*, v. 10, n. 1, pp. 33-36.

CHACKRABARTI, K., GAROFALAKIS, M., RASTOGI, R., *et al.*, 2001, "Approximate Query Processing Using Wavelets", *The International Journal on Very Large Data Bases*, v. 10, n. 2-3 (Sep.), pp. 199-223.

COMER, D., 1979, "The Ubiquitous B-tree". *Computing Surveys*, v. 11, n. 2 (Jun.), 122-137.

COSTA, J. P., FURTADO, P., 2003, "Time-Stratified Sampling for Approximate Answers to Aggregate Queries". In: *Proceedings of Eighth International Conference on Database Systems for Advanced Applications*, pp. 215-222, Kyoto, Japan, March.

DAS, A., GEHRKE, J., RIEDWALD, M., 2004, "Approximation Techniques for Spatial Data". In: *Proceedings of the 2004 ACM-SIGMOD International Conference on Management of Data*", pp. 695-706, Paris, France, Jun.

DASH, M., LIU, H., YAO, J., 1997, "Dimensionality reduction of unsupervised data", In: *Proceedings of the 9th International Conference on Tools with Artificial Intelligence*, pp. 532-539, Newport Beach, CA, USA, Nov.

DOBRA, A., GAROFALAKIS, M., GEHRKE, J. E., RASTOGI, R., 2002, "Processing complex aggregate queries over data streams". In: *Proceedings of the 2002*

ACM-SIGMOD International Conference on Management of Data, pp. 61-72, Madison, Wisconsin, USA, Jun.

ESTER, M., KRIEGEL, H. P., XU, X., (1995), "Knowledge Discovery in Large Spatial Databases: Focusing Techniques for Efficient Class Identification". In: *Proceedings of the 4th International Symposium on Advances in Spatial Databases*, pp. 67-82, Portland, Maine, USA, Aug.

FALOUTSOS, C., JAGADISH, H. V., SIDIROPOULOS, N. D., 1997, "Recovering information from summary data". In: *Proceedings of 23rd International Confeference on Very Large Data Bases*, pp. 36-45, Athens, Greece, Aug.

FINKEL, R. A., BENTLEY, J. L., 1974, "Quad-trees: A data structure for retrieval on composite keys", *ACTA Informatica*, v. 4, n. 1, pp. 1-9.

FURTADO, P., COSTA, J. P., 2002, "Time-Interval Sampling for Improved Estimations in Data Warehouses". In: *Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery*, pp. 327-338, Aix-en-Provence, France, Sep.

FURTADO, P., MADEIRA H., 1999, "Summary Grids: Building Accurate Multidimensional Histograms". In: *Proceedings of the Sixth International Conference on Database Systems for Advanced Applications*, pp. 187-194, Hsinchu, Taiwan, Apr.

FURTADO, P., MADEIRA, H., 2000a, "Data Cube Compression with Quanticubes". In: *Data Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery*, pp. 162-167, London, UK, Sep.

FURTADO, P., MADEIRA, H., 2000b, "FCompress: A New Technique for Queriable Compression of Fact and Datacubes". In: *Proceedings of 2000 International Database Engineering and Applications Symposium*, pp. 197-206, Yokohoma, Japan, Sep.

GAEDE, V., GÜNTHER, O., 1998, "Multidimensional Access Methods", *Computing Surveys*, v. 30, n. 2 (Jun), pp. 170-231.

GIBBONS, P. B., Matias, Y., Poosala, V., 1997, *Aqua project white paper*. Technical Report, Bell Laboratories, Murray Hill, New Jersey, USA.

GÜTING, R. H., 1994, "An Introduction to Spatial Database Systems", *The International Journal on Very Large Data Bases*, v. 3, n. 4 (Oct), pp. 357-399.

GÜTING, R.H., SCHNEIDER, M., 1995, "Realm-Based Spatial Data Types: The ROSE Algebra", *The International Journal on Very Large Data Bases*, v. 4, n. 2 (Apr), pp. 243 - 286.

GÜTING, R. H., DE RIDDER, T., SCHNEIDER, M., 1995, "Implementation of the ROSE Algebra: Efficient Algorithms for Realm-Based Spatial Data Types". In: *Proceedings of the 4th International. Symposium on Large Spatial Databases Systems*, pp. 216-239, Portland, USA, Aug.

GUTTMAN, A., 1984, "R-Tree: A dynamic index structure for spatial searching". In: *Proceeding of 1984 ACM-SIGMOD International Conference of Data Management*, pp. 47-57, Boston, MA, USA, Jun.

HAN, J., KAMBER, M., 2001, *Data Mining: concepts and techniques*. 1 ed., New York, Morgan Kaufmann publishers.

HELLERSTEIN, J. M., Haas, P. J., Wang, H. J., 1997, "Online aggregation". In: *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 171-182, Tucson, Arizona, USA, May.

HEUVELINK, G., 1998, *Error Propagation in Environmental Modeling with GIS*. 1 ed., London, UK, Taylor & Francis.

IBGE, 1996. *Malha Municipal Digital do Brasil - 1994*, Fundação Instituto Brasileiro de Geografia e Estatística, Rio de Janeiro, Brasil.

IOANNIDIS, Y. E., POOSALA, V. 1995, "Balancing histogram optimality and practicality for query result size estimation". In: *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, pp. 233-244, San Jose, California, USA, May.

JAGADISH, H. V., MUMICK, I. S., SILBERSCHATZ, A., 1995, "View maintenance issues in the chronicle data model". In: *Proceedings of the fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 113-124, San Jose, California, USA, May.

JOHNSON, G. H., WICHERN D. A., 1992, *Applied Multivariate Statistical Analysis*. 3 ed., Upper Saddle River, NJ, USA, Prentice Hall.

KAUFMAN, L., ROUSSEEUW, P. J., 1990, *Finding Groups in Data: an Introduction to Cluster Analysis.* 1 ed. New York, John Wiley & Sons.

KOHAVI, R., JOHN, G. H., 1997, "Wrappers for feature subset selection", *Artificial Intelligence*, v. 97, n. 1-2 (Dec), pp. 273-324.

KOTHURI, R. K., RAVADA, S., 2001, "Efficient Processing of Large Spatial Queries Using Interior Approximations". In: *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pp. 404-424, Redondo Beach, CA, USA, Jul.

LARSON, H. J., 1982, *Introduction to probability theory and statistical inference*. 3 ed. New York, John Wiley & Sons.

LIANG, Y., BARSKY, B., 1983, "An analysis and algorithm for polygon clipping", *Communications of the ACM*, v. 26, n. 11 (Nov), pp. 868-877.

LO, M. L., RAVISHANKAR, C. V., 1996, "Spatial Hash-Joins". In: *Proceedings of the 1996 ACM-SIGMOD International Conference on Management of Data*, pp. 247-258, Montreal, Quebec, Canada, Jun.

LLOYD, S. P., 1982, "Least squares quantization in PCM", *IEEE Transactions on Information Theory*, v. 28, n. 1 (Jan), pp. 129-136.

MADRIA, S. K., MOHANIA, M. K., RODDICK, J. F., 1998, "A Query Processing Model for Mobile Computing using Concept Hierarchies and Summary Databases". In: *The 5th International Conference of Foundations of Data Organization*, pp. 147-157, Kobe, Japan, Nov.

MAILLOT, P. G., 1992, "A new, fast method for 2D polygon clipping: analysis and software implementation". In: *ACM Transactions on Graphics*, v.11, n.3 (Jul), pp. 276-290.

MALVERSTUTO, F. M., 1991, "Approximating Discrete Probability Distributions with Decomposable Models", *IEEE Transactions on Systems, Man, Cybernetics*, v. 21, n. 5, pp. 1287-1294.

MATIAS, Y., VITTER, J. S., WANG, M., 1998, "Wavelet-based histograms for selectivity estimation", In: *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pp. 448 – 459, Seattle, Washington, USA, Jun.

MONTEIRO, R. S., AZEVEDO, L.G., ZIMBRAO, G., SOUZA, J. M., 2004, “Polygon and Polyline Join Using Raster Filters”. In: *Proceedings of the 9th International Conference on Database Systems for Advances Applications*, pp. 255-261, Jeju Island, Korea, Mar.

NEWMAN, W. M., SPROULL, R. F, 1979, *Principles of Interactive Computer Graphics*. 2 ed. New York, McGraw-Hill Book Company.

NG, R., HAN, J., 1994, “Efficient and Effective Clustering Method for Spatial Data Mining”. In: *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 144-155, Santiago de Chile, Chile, Sep.

ORENSTEIN, J. A, 1986, “Spatial query processing in an object-oriented database system”. In: *Proceedings of 1986 ACM SIGMOD International Conference on Management of Data*, pp. 326-336, Washington, D.C., USA, May.

PAPADIAS, D., KALNIS, P., ZHANG, J. *et al.* 2001, "Efficient OLAP Operations in Spatial Data Warehouses". In: *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pp. 443-459, Redondo Beach, CA, USA, Jul.

PAPADIAS, D., MAMOULIS, N., THEODORIDIS, Y., 1999, “Processing and optimization of multiway spatial joins using R-Trees”. In: *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 189-200, Philadelphia, Pennsylvania, USA, May-Jun.

PEARL, J., 1988, *Probabilistic Reasoning in Intelligent Systems*, 1 ed., San Francisco, CA, USA, Morgan Kauffman.

PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., FLANNERY, B. P. 1996, *Numerical Recipes in C, The Art of Scientific Computing*. 2 ed. Cambridge, MA, USA, Cambridge University Press.

POOSALA, V., IOANNIDIS, Y. E., HAAS, P. J., SHEKITA, E. J., 1996, “Improved Histograms for Selectivity Estimation of Range Predicates”, *ACM SIGMOD record*, v. 25, n. 2 (Jun), pp. 294-305.

RODDICK, J., EGENHOFER, M., HOEL, E., *et al.*, 2004. “Spatial, Temporal and Spatiotemporal Databases Hot Issues and Directions for PhD Research”, *SIGMOD Record*, v. 33, n. 2 (Jun), pp. 126-131.

ROSS, K., SRIVASTAVA, D., 1997, "Fast computation of sparse datacubes". In: *Proceedings of the 23rd International Conference on Very Large Data Bases*, 116-125, Athens, Greece, Aug.

SAMET, H., 1990, *The Design and Analysis of Spatial Data Structure*. 1 ed., Boston, Massachusetts, Addison-Wesley Publishing Company.

SARAWAGI, S., STONEBRAKER, M., 1994, "Efficient organization of large multidimensional arrays". In: *Proceedings of the Tenth International Conference on Data Engineering*, pp. 328-336, Houston, Texas, USA, Feb.

SAYOOD, K., 1996, *Introduction to data compression*. 2 ed., San Francisco, CA, USA, Morgan Kauffman.

SCHNEIDER, M., 1999, "Uncertainty management for spatial data in databases: Fuzzy spatial data types". In: *Proceedings of the 6th International Symposium on Advances in Spatial Databases*, pp. 330-351, Hong Kong, China, Jul.

SIEDLECK, W., SKLANSKY, J., 1988, "On automatic feature selection", *International Journal of Pattern Recognition and Artificial Intelligence*, v. 2, n.2, pp. 197-220.

STEEL, R. G. D., TORRIE, J. H., 1976, *Introduction to statistics*. 1 ed. New York, McGraw-Hill Book Company.

STOLLNITZ, E. J., DEROSE, T. D., SALESIN, D. H., 1996, *Wavelets for computer Graphics – Theory and Applications*. 1 ed. San Francisco, Morgan Kaufmann Publishers Inc.

TAO, Y., SUN, J., PAPADIAS, D., 2003, "Selectivity estimation for predictive spatio-temporal queries". In: *Proceedings of the 19th International Conference on Data Engineering*, pp. 417-428, Bangalore, India.

VEENHOF, H. M., APERS, P. M. G., HOUTSAMA, A. W., 1995, "Optimisation of Spatial Joins Using Filters". In: *Advances in Databases, 13th British National Conference on Databases*, pp. 136-154, Manchester, United Kingdom, July.

VITTER, J. S., WANG, M., 1999, "Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets". In: *Proceedings of 1999 ACM SIGMOD International Conference on Management of Data*, pp. 193-204, Philadelphia, PA, USA, May-Jun.

VITTER, J. S., WANG, M., IYES, B., 1998, “Data Cube Approximation and Histograms via Wavelets”. In: *Proceedings of the Seventh International Conference on Information and Knowledge Management*, pp. 96-104, Bethesda, Maryland, USA, Nov.

WANG, W., YANG, J., MUNTZ R., 1997, “STING: A Statistical Information Grid Approach to Spatial Data Mining”. In: Proc. *Proceedings of 23rd International Confeference on Very Large Data Bases*, pp. 186-195, Athens, Greece, Aug.

WU, Y., AGRAWAL, D., ABBADI, A.E., 2001, “Applying the Golden Rule of Sampling for Query Estimation”, ACM SIGMOD Record, v. 30, n.2 (Jun), pp. 449-460.

WONNACOTT, R. J., WONNACOTT, T. H., 1985, “Introductory Statistics”, 4 ed. New York, John Wiley & Sons.

ZHU, H., SU, J., IBARRA, O. H. 2000, ”Toward Spatial Joins for Polygons”. In: *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*, pp. 231-244, Berlin, Germany, Jul.

ZHANG, J., GOODCHILD, M., 2002, *Uncertainty in Geographical Information System*. 1 ed., Erewhon, NC, Taylor & Francis.

ZHANG, T., RAMAKRISHNAN R., LIVNY, M., 1996, “Birch: an Efficient Data Clustering Method for Very Large Databases”, *ACM SIGMOD Record*, v. 25, n. 2 (Jun), pp. 103-114.

ZHOU, X., TRUFFET, D. , HAN, J., 1999, “Efficient Polygon Amalgamation Methods for Spatial OLAP and Spatial Data Mining”. In: In: *Proceedings of the 6th International Symposium on Advances in Spatial and Temporal Databases*, pp. 167-187, Hong Kong, China, Jul.

ZIMBRAO, G., SOUZA, J. M., 1998, “A Raster Approximation for Processing of Spatial Joins”. In: *Proceedings of the 24rd International Conference on Very Large Data Bases*, pp. 558-569, New York City, New York, USA, Aug..