# Modelo de custo para consultas por similaridade em espaços métricos

***Gisele Busichia Baioco***

# Modelo de custo para consultas por similaridade em espaços métricos

***Gisele Busichia Baioco***

**Orientadora:** ***Profa. Dra. Agma Juci Machado Traina***
**Co-orientador:** ***Prof. Dr. Caetano Traina Junior***

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências de Computação e Matemática Computacional.

**USP – São Carlos**
**Dezembro de 2006**

*Aos meus pais.*

# Agradecimentos

Aos meus orientadores Profa. Dra. Agma Juci Machado Traina e Prof. Dr. Caetano Traina Junior, com quem compartilho os resultados deste trabalho, pela excelente orientação, incentivo e amizade.

A todos os integrantes do Grupo de Bases de Dados e Imagens do ICMC–USP que direta ou indiretamente contribuíram para esta realização.

Ao ICMC–USP pelo apoio institucional.

A todos da secretaria da pós-graduação do ICMC–USP pela atenção e competência.

A minha família pela compreensão e apoio nos momentos difíceis.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Resumo

Esta tese apresenta um modelo de custo para estimar o número de acessos a disco (custo de I/O) e o número de cálculos de distância (custo de CPU) para consultas por similaridade executadas sobre métodos de acesso métricos dinâmicos. O objetivo da criação do modelo é a otimização de consultas por similaridade em Sistemas de Gerenciamento de Bases de Dados relacionais e objeto-relacionais. Foram considerados dois tipos de consultas por similaridade: consulta por abrangência e consulta aos *k*-vizinhos mais próximos. Como base para a criação do modelo de custo foi utilizado o método de acesso métrico dinâmico *Slim-Tree*. O modelo estima a dimensão intrínseca do conjunto de dados pela sua dimensão de correlação fractal. A validação do modelo é confirmada por experimentos com conjuntos de dados sintéticos e reais, de variados tamanhos e dimensões, que mostram que as estimativas obtidas em geral estão dentro da faixa de variação medida em consultas reais.

# Abstract

This thesis presents a cost model to estimate the number of disk accesses (I/O costs) and the number of distance calculations (CPU costs) to process similarity queries over data indexed by dynamic metric access methods. The goal of the model is to optimize similarity queries on relational and object-relational Database Management Systems. Two types of similarity queries were taken into consideration: range queries and k-nearest neighbor queries. The dynamic metric access method Slim-Tree was used as the basis for the creation of the cost model. The model takes advantage of the intrinsic dimension of the data set, estimated by its correlation fractal dimension. Experiments were performed on real and synthetic data sets, with different sizes and dimensions, in order to validate the proposed model. They confirmed that the estimations are accurate, being always within the range achieved executing real queries.

# 1. INTRODUÇÃO

## 1.1. Contexto e motivação

Tradicionalmente, os Sistemas de Gerenciamento de Bases de Dados (SGBDs) foram desenvolvidos para cuidar de dados de tipos numéricos ou textuais curtos, sendo que os mais utilizados são aqueles construídos segundo o modelo relacional [Codd_1970]. Nesse modelo, todos os elementos de uma modelagem recaem sobre apenas dois construtores semânticos: atributos e relações. Dessa maneira, qualquer elemento do mundo real percebido pelo projetista (ou pela aplicação) como uma entidade ou objeto será necessariamente representado por uma relação, e suas propriedades serão representadas por atributos. O modelo relacional não provê mecanismos para associar dois ou mais valores de atributos, a não ser isolando-os em outra relação. Essa estrutura extremamente simples tem permitido ao modelo relacional obter o melhor desempenho dentre as várias alternativas existentes, em termos de velocidade de atualização e acesso aos dados. Porém, essa estrutura simples torna mais difícil o desenvolvimento de aplicativos que devem manipular tipos de dados estruturalmente complexos, ou seja, dados cuja estrutura é composta por outros atributos de tipos mais simples. Exemplos de dados com estrutura interna complexa são os dados multimídia (como imagens, áudio, texto e vídeo), dados multidimensionais, séries temporais, dados genéticos (cadeias de DNA) e impressões digitais.

Para aumentar a flexibilidade dos SGBDs apoiados no modelo relacional e, conseqüentemente, facilitar seu uso no desenvolvimento de aplicativos que tratam de objetos mais complexos, eles têm sido estendidos para incorporar recursos oriundos do desenvolvimento dos modelos orientados a objetos [Bertino_1994], que comprometam o mínimo possível o seu desempenho. A partir desse enfoque surgiram os SGBDs denominados objeto-relacionais [Cattell_1994]. Esses SGBDs têm sido alvo dos maiores investimentos em

desenvolvimento por parte das empresas de software fornecedoras de SGBDs relacionais, sendo que todas as grandes empresas (Oracle, IBM, Sybase, Microsoft, entre outras) têm versões objeto-relacionais de seus principais produtos [Oracle Corporation_2005] [IBM Corporation_2006].

Diante desse cenário, a linguagem de acesso padrão a SGBDs relacionais, a SQL (*Structured Query Language*), teve sua terceira versão, a chamada ANSI/ISO SQL:1999, ou ORDBMS (*Object-Relational Database Management Systems*) SQL [Eisenberg_1999], desenvolvida principalmente para suportar extensões objeto-relacionais [O'Neil_2001]. A linguagem SQL: 1999 padronizou duas construções especificamente voltadas para a extensão "objeto" dos SGBDs: suporte a métodos definidos pelo usuário – UDF (*User Defined Functions*), em linguagens de programação (como C++ e Java); e tipos de dados definidos pelo usuário – UDT (*User Defined Types*), com a possibilidade de uso de coleções de dados (como listas e arranjos). Após a SQL:1999, foi publicada a mais recente versão da linguagem SQL, a SQL:2003 [Eisenberg_2004], que revisou a versão anterior mantendo o suporte a extensões objeto-relacionais.

UDFs permitem a incorporação de código escrito pelo usuário, idealmente otimizado, para auxiliar os processos de armazenamento e recuperação de informação em bases de dados. UDTs permitem a representação de objetos como propriedades de outros objetos armazenados em uma tupla. Dessa maneira, é possível definir um objeto tão complexo quanto se queira, como por exemplo, imagens e áudio, como uma das propriedades de uma relação. Um exemplo pode ser o registro de informações sobre pessoas em uma secretaria de segurança pública, para o qual se cria uma tabela com atributos textuais e numéricos descritivos, dois atributos para foto frontal e de perfil, e um arranjo de dez impressões digitais.

Todos os dados do registro de pessoas devem poder ser consultados/recuperados. Isto é, os dados complexos (fotos e impressões digitais) também precisam ser consultados/recuperados da mesma maneira que os dados simples. Uma primeira abordagem para a recuperação de dados complexos baseia-se na utilização de textos descritivos sobre o conteúdo desses dados. Essa abordagem, usualmente denominada abordagem **semântica**, é interessante quando é possível descrever toda a semântica dos dados de maneira textual, em particular toda a semântica que poderá ser necessária para responder consultas. Entretanto, nem sempre é

possível descrever todos os detalhes, por exemplo de uma imagem, que possam ser necessários em consultas futuras. Desse modo, está crescendo o uso da chamada **abordagem sintática,** a qual se baseia na extração de características de baixo nível do dado complexo, e que podem ser obtidas automaticamente. A abordagem sintática é base dos sistemas de recuperação de imagens por conteúdo (do inglês *Content-Based Image Retrieval* – CBIR) [Lew_2006] que permitem recuperar imagens utilizando-se características delas extraídas automaticamente. Já a abordagem semântica é também aplicada a bases de dados médicas que utilizam sistemas de PACS (*Picture Archiving and Comunication Systems*) [Cao_2000] [Müller_2004], que armazenam imagens de exames de pacientes juntamente com os respectivos laudos, e suportam a recuperação dos mesmos por consultas textuais aos laudos [Adelhard_1999]. Nessa aplicação, o médico radiologista em geral procura descrever detalhes que são importantes para o laudo em questão, não se preocupando com outras características que não interessam ao atual quadro clínico do paciente. O ideal é poder obter os benefícios das duas abordagens de modo integrado.

Uma área de pesquisa muito intensa atualmente é o desenvolvimento de maneiras de recuperar dados complexos por seu conteúdo, ou seja, utilizando a abordagem sintática. Como a comparação de dados complexos é muito custosa do ponto de vista computacional, a técnica fundamental adotada é a **extração de características** [Smeulders_2000] [Müller_2004], as quais são armazenadas juntamente com os dados. As características extraídas são indexadas, e o processo de recuperação inicialmente as utiliza para filtrar os dados complexos, de maneira que poucas comparações são efetuadas diretamente nos dados. Cabe ao usuário escolher os dados de seu interesse entre o resultado do processo de filtragem, que lhe é apresentado diretamente.

O processo de comparação de dados complexos é usualmente computacionalmente caro, pois envolve a execução de algoritmos e métodos que quantificam a similaridade entre eles. Constata-se, então, a necessidade do desenvolvimento de técnicas para otimizar consultas a dados complexos em bases de dados objeto-relacionais, utilizando as características extraídas e indexadas dos mesmos. Com esse objetivo, pretende-se adotar o enfoque básico do modelo relacional, de que cada tipo de dado define um domínio, de onde atributos de uma relação têm seus valores amostrados, agora estendendo esse conceito para tratar domínios complexos, tais como imagens, áudio, vídeo ou estruturas genéticas. Assim, da mesma maneira que um SGBD

puramente relacional não indexa todos os números de uma base de dados, ou mesmo de uma relação em uma mesma estrutura, mas cria estruturas de indexação separadas para cada atributo, também os dados complexos terão uma estrutura de indexação separada para cada atributo complexo, mesmo que mais de um atributo amostre seus dados em um mesmo domínio. Por exemplo, fotos de perfil e fotos frontais são indexadas em estruturas independentes. Além disso, dado que várias características independentes (como por exemplo, histograma de cor e histograma de textura) podem ser extraídas e indexadas de cada atributo de tipo complexo, a escolha de qual, ou quais, estruturas de indexação devem ser utilizadas para responder uma consulta é uma decisão a ser tomada pelo processo otimizador da consulta, com base em estimativas de seletividade e modelos de custo criados para cada índice.

Desse modo, a motivação para o presente trabalho é a necessidade de métodos de estimativa de seletividade em características extraídas de dados complexos, de indexação dessas características, e de modelos de custo de acesso para as estruturas de índice associadas aos atributos com dados complexos. Esses métodos de estimativa de seletividade e modelos de custos tanto poderão incluídos diretamente nos SGBDs relacionais quanto ser tratados como UDFs a serem incorporadas aos SGBDs objeto-relacionais como funções de apoio ao otimizador de consultas. Da mesma maneira, as estruturas de indexação, incluindo as coleções de características extraídas, poderão ser tratadas como UDTs em SGBDs objeto-relacionais ou ser incluídas como estruturas adicionais em SGBDs relacionais para tratamento dos objetos de tipos complexos.

## 1.2. Objetivos do trabalho

Os SGBDs relacionais e objeto-relacionais usualmente seguem a arquitetura cliente-servidor, onde aplicações cliente solicitam operações de armazenagem e recuperação de dados para um ou mais servidores de dados. Os servidores recebem as solicitações dos clientes por meio de comandos na linguagem SQL, analisam tais comandos, e criam um plano de execução para atender à solicitação. A execução de uma consulta pode ser bastante demorada. Assim, diversas alternativas são pré-avaliadas para a criação de um plano de execução, escolhendo-se uma que otimize a execução. Para isso, existe um módulo do servidor, denominado otimizador de consultas, que avalia diversos fatores que podem afetar o desempenho do

processo de execução de uma consulta, tais como a utilização de estruturas de indexação, a seqüência das operações, quais operadores utilizar (dado que propriedades algébricas permitem expressar a mesma consulta de várias maneiras, usando diferentes operadores) e a melhor configuração da memória disponível para *cache* das relações em memória [O'Neil_2001] [Elmasri_2003].

Este trabalho utilizou como base de desenvolvimento a estrutura de indexação para espaços métricos desenvolvida pelo Grupo de Bases de Dados e Imagens - GBdI - do ICMC, a *Slim-Tree* [Traina Jr._2000b] [Traina Jr._2002a]. A *Slim-Tree* foi empregada para criar um modelo de custo para consultas por similaridade em espaços métricos, que poderá ser utilizado para o desenvolvimento de modelos para outras estruturas dinâmicas de indexação em espaços métricos. Embora alguns trabalhos iniciais tenham sido efetuados na direção de estimativas de seletividade [Belussi_1995] [Traina Jr._2000a] e modelos de custo [Ciaccia_1998] [Böhm_2000] em espaços métricos, a criação de um modelo de custos e previsão de seletividade completo, dentro do paradigma relacional, era uma tarefa nunca empreendida e que foi realizada por este trabalho.

## 1.3. Principais contribuições

Esta tese apresenta, como principal contribuição, o desenvolvimento de um modelo de custo para consultas por similaridade a dados complexos, com enfoque em dados do tipo imagem, representados por conjuntos de vetores de características e indexados usando um método de acesso métrico dinâmico.

O modelo de custo proposto estima o número de acessos a disco e o número de cálculos de distância para os dois tipos principais de consultas por similaridade: consulta por abrangência (*range query*) e consulta aos *k*-vizinhos mais próximos (*k-nearest neighbor query*).

Foram desenvolvidos dois conjuntos de equações para a estimativa de custos. O primeiro se baseia em parâmetros globais do conjunto de dados, o que proporciona uma estimativa de custo inicial de maneira rápida, baseada em parâmetros que representam o conjunto de maneira global. Entretanto, essas estimativas iniciais não conseguem muitas vezes identificar variações locais que ocorrem devido à distribuição regional dos dados. Desse modo, o

segundo conjunto de equações trabalha esse aspecto e aprimora as estimativas considerando custos reais de consultas previamente executadas no conjunto de dados. Essa estimativa local considera pesos de custos estimados e reais previamente medidos como função de distância da consulta corrente e de consultas anteriormente executadas e armazenadas. Esse procedimento demanda armazenar poucas consultas, minimizando o custo de encontrar uma consulta previamente armazenada que se qualifique para o processo de aprimoramento.

Finalmente, a eficácia do modelo de custo proposto é confirmada por experimentos com conjuntos de dados sintéticos e reais, de variados tamanhos e dimensões, que mostram que as estimativas obtidas em geral estão dentro da faixa de variação medida em consultas reais.

## 1.4. Organização do trabalho

No capítulo 2 são apresentados os principais conceitos envolvidos em consultas por conteúdo em bases de dados complexos: consultas por similaridade e estruturas de indexação.

No capítulo 3 são levantados trabalhos existentes na literatura sobre estimativa de seletividade para consultas por similaridade e modelos de custo para métodos de acesso a dados complexos.

No capítulo 4, são apresentadas a descrição e a delimitação do problema tratado por esta tese, descrevendo a hipótese para sua solução e caracterizando-o de acordo com o contexto da revisão da literatura realizada nos capítulos 2 e 3.

O capítulo 5 apresenta o modelo de custo para consultas por similaridade em espaços métricos proposto.

O capítulo 6 apresenta os resultados de experimentos efetuados para comprovar a eficácia do modelo de custo proposto.

Finalmente, o capítulo 7 apresenta as conclusões finais e propostas para futuras pesquisas.

# 2. CONSULTAS POR CONTEÚDO EM BASES DE DADOS COMPLEXOS

## 2.1. Introdução

Dados multimídia como imagens, áudio, texto e vídeo, são tratados neste trabalho como tipos de dados complexos. O termo "Tipos de Dados Complexos" refere-se a dados cuja estrutura interna é composta por vários atributos mais simples (mesmo que essa estrutura não seja reconhecida pelo SGBD) e, em geral, representam conjuntos volumosos de informação, requerendo grandes quantidades de *bytes* de memória para armazenamento. Além de armazenados, dados complexos precisam ser consultados por seu conteúdo. Consultas por conteúdo a esses dados devem ser realizadas utilizando critérios de similaridade, sendo denominadas **consultas por similaridade**. Os principais conceitos em torno de consultas por similaridade, incluindo: espaço métrico, funções de distância métricas e tipos mais usuais de consulta por similaridade são abordados na Seção 2.2.

Técnicas de indexação para agilizar a consulta em conjuntos volumosos de dados têm sido estudadas desde os tempos em que os SGBDs relacionais tratavam apenas de dados convencionais. Resultados recentes têm mostrado que as consultas por similaridade também têm seu desempenho melhorado a partir do uso de **estruturas de indexação para dados complexos**. Desse modo, a Seção 2.3 trata de métodos de acesso existentes para dados complexos, enfocando as estruturas de indexação para espaços métricos, incluindo uma visão geral do método de acesso métrico dinâmico *Slim-Tree*, o qual é utilizado nos demais capítulos desta tese como base para o modelo de custos proposto.

## 2.2. Consultas por similaridade

A primeira maneira de recuperar (consultar) dados complexos que foi desenvolvida, denominada abordagem **semântica**, utilizava textos descritivos do conteúdo dos dados, sendo a consulta realizada diretamente nos textos [Adelhard_1999]. Existem vários problemas inerentes a essa abordagem, tais como o fato de o texto descritivo original não permitir pesquisas imprevistas em aplicações subseqüentes e a falta de uniformidade das descrições textuais dos dados complexos, já que o mesmo dado analisado por pessoas diferentes poderá receber textos descritivos distintos. Além disso, a necessidade de intervenção humana inviabiliza o acesso a componentes dos dados (por exemplo, partes de uma imagem), os quais são gerados em grandes volumes, requerendo um processo de geração das descrições mais automático. Atualmente, busca-se realizar a consulta em dados complexos diretamente pelo seu conteúdo, utilizando um processo automático e que aproveita as características inerentes ao próprio dado. Essa maneira de recuperação de dados complexos é denominada abordagem **sintática**.

O processo que utiliza a abordagem sintática conhecido por **recuperação baseada em conteúdo** (*content-based retrieval*) [Lew_2006] utiliza uma função (algoritmo) que processa a estrutura interna dos dados complexos extraindo outros dados que podem ser comparados no lugar dos objetos complexos, para aproximar em algum grau uma estimativa de similaridade entre os dados complexos. Ou seja, utiliza-se um algoritmo de processamento do dado complexo visando obter informação que capture a **essência** do dado complexo segundo algum um aspecto específico [Traina_2004]. Essa essência do dado é usualmente denominada por **característica** (*feature*) do mesmo, o processamento do dado é denominado de **extração da característica**, e o algoritmo utilizado é denominado um **extrator de características** [Smeulders_2000] [Müller_2004]. Um extrator em geral recupera diversos valores numéricos ou textuais, que descrevem o dado complexo segundo o aspecto tratado e, portanto, diz-se que o dado é descrito por um **vetor de características** (*feature vector*). Em geral, procura-se extrair as mesmas características que o especialista no domínio de dados utiliza no processo de análise dos mesmos. Por exemplo, no caso de imagens as características mais utilizadas são distribuições de cores, forma e textura; para áudio extraem-se entre outros dados a freqüência e a altura do comprimento de onda.

Uma vez extraídas as características dos dados complexos armazenados em uma base de dados, estas são utilizadas nas operações de comparação efetuadas para recuperar os dados complexos que respondem às consultas efetuadas. Como a comparação envolve vetores de características, o processo tende a ser bem mais sofisticado do que o utilizado para a recuperação de tipos de dados convencionais, como dados numéricos (números inteiros, números reais, data, hora, etc) e textuais curtos (códigos de identificação, siglas, etc.), onde se busca a coincidência entre os valores dos dados por critérios que envolvem: **igualdade**, em que o interesse é por valores exatamente coincidentes; e **ordem**, em que o interesse é por valores maiores ou menores que um valor fornecido.

É importante ressaltar que critérios de comparação baseados em igualdade e ordem não são adequados para a comparação de dados complexos. Ou seja, não há benefício em realizar consultas como, por exemplo: obtenha as imagens de pacientes com tumor no cérebro cuja tomografia seja igual à do paciente em estudo. Dificilmente (na prática nunca) as tomografias de dois tumores serão exatamente iguais, mesmo que os tumores tenham a mesma classificação e sejam até do mesmo paciente. O critério mais adequado para casos assim é o de **similaridade** [Aslandogan_1999] [Gao_2005], no sentido de avaliar o significado do conteúdo dos dados complexos. A consulta anterior faria mais sentido se definida como: obtenha os pacientes com tumor no cérebro cuja tomografia seja **similar** à do paciente em estudo. O grau de similaridade e como ela será medida são parâmetros que precisam ser definidos para que se possa efetuar consultas desse tipo.

Não existe uma formulação geral para a avaliação da similaridade entre dados complexos, pois essa avaliação depende das necessidades da aplicação e é, portanto, altamente dependente do domínio em que está sendo utilizada. Entretanto, qualquer modo de avaliação de similaridade toma dois dados complexos como parâmetros de entrada e retorna uma medida que pode ser quantificada como um valor real positivo, que corresponde ao grau de similaridade entre os mesmos [Böhm_2001]. Quando o dado complexo tem suas características essenciais extraídas e representadas por um vetor de características, o processo de avaliação de similaridade deve tratar o par de vetores que representam o par de objetos complexos que devem ser comparados.

A avaliação da similaridade é usualmente feita utilizando **funções de distância**. Tais funções podem ser definidas matematicamente, e em geral são realizadas como algoritmos computacionais que recebem dois dados complexos de um mesmo domínio e retornam a "distância", ou grau de dissimilaridade, entre os mesmos. Idealmente a função de distância deve ser definida de maneira a ser coerente com a noção de semelhança percebida pelo ser humano, ou seja, deve retornar valores relativamente pequenos para dados parecidos (próximos entre si) e relativamente grandes para dados bem diferentes (distantes um do outro). Como a distância é mensurada sobre as características extraídas dos dados complexos, elas capturam a informação segundo um critério específico, portanto é importante ressaltar que uma operação de comparação por similaridade considera somente o critério avaliado.

A avaliação da similaridade é usualmente tratada como um processo separado dos demais processos envolvidos na armazenagem e busca por conteúdo de dados complexos, tais como representação, especificação e avaliação de consultas, indexação e recuperação. Essa separação é importante, pois o processo de cálculo de similaridade é totalmente dependente do domínio da aplicação, enquanto para os demais processos essa dependência é de grau menor.

Consultas que usam o grau de dissimilaridade entre dados complexos para obter a resposta são denominadas **consultas por similaridade** e envolvem: uma função de distância; um objeto de busca, também considerado como o centro da consulta, que é o dado a partir do qual se deseja encontrar os mais semelhantes; e um conjunto de parâmetros que depende do tipo de consulta por similaridade a ser realizado. Os tipos mais comuns de consulta por similaridade são **consulta por abrangência** e **consulta aos *k*-vizinhos mais próximos**, os quais são abordados na Seção 2.2.2.

Como o processo de extração de características tende a ser muito caro do ponto de vista computacional, os vetores de características são armazenados na base de dados juntamente com os dados complexos, a partir dos quais os dados passam a ser comparados e indexados (Seção 2.3). Assim, em uma consulta por similaridade, o processo de recuperação inicialmente utiliza as características extraídas dos dados (já indexadas e armazenadas na base de dados) como filtros de informação, de maneira que poucas comparações são efetuadas diretamente nos dados complexos, o que em geral é feito pelo usuário, que escolhe os dados

de seu interesse a partir do resultado do processo de filtragem. Vale ressaltar que mais de um vetor de características pode ser utilizado, de modo a melhorar a filtragem.

Consultas por similaridade são efetuadas em domínios de dados complexos, representados por dois modelos principais como apresentado em [Gaede_1998] [Chávez_2001] [Samet_2006]. Para isso, as definições a seguir são necessárias.

> **Definição 2.1 – Modelo de Espaço Vetorial:** no modelo de espaço vetorial os dados complexos são descritos por vetores de características, tratados como coordenadas de pontos no espaço *e*-dimensional, onde *e* corresponde à quantidade de elementos (atributos) que compõem o vetor de características. Nesse modelo, a abordagem mais comum é que a similaridade (dissimilaridade) seja avaliada por uma das funções de distância de Minkowski (Seção 2.2.1);

> **Definição 2.2 – Modelo de Espaço Métrico:** para alguns domínios, a extração de vetores de características, com a mesma dimensão para todos os objetos pode ser uma tarefa muito complicada, ou até inviável, como no caso em que os tipos ou número de características variam para cada dado complexo, ou seja, não há dimensão definida. Nesse caso, define-se o modelo de espaço métrico, onde a similaridade entre os objetos é avaliada a partir da definição de uma função de distância métrica.

Considerando os dois modelos definidos anteriormente, consultas por similaridade são apoiadas por estruturas de dados para espaços métricos, que englobam tanto dados vetoriais com dimensão finita (modelo de espaço vetorial) quanto dados adimensionais (modelo de espaço métrico). O conceito de espaços métricos é abordado na Seção 2.2.1.

### 2.2.1. Espaços métricos

De acordo com [Chávez_2001] [Samet_2006], um **espaço métrico** é um par $\mathbb{M} = (\mathbb{S}, d)$ onde $\mathbb{S}$ é um domínio ou universo de objetos válidos e $d(\ )$ é uma função de distância métrica (ou simplesmente, métrica). O subconjunto finito $S \subseteq \mathbb{S}$, de cardinalidade (número de elementos) $|S|$, representa o conjunto de objetos onde as consultas serão efetuadas, ou seja, os objetos complexos armazenados na base de dados.

A métrica $d(\ )$ definida por $\mathbb{S} \times \mathbb{S} \rightarrow \mathbb{R}^+$ corresponde à medida de distância (dissimilaridade) entre dois objetos, e quanto menor o valor dessa distância, mais próximos ou semelhantes eles serão. Uma métrica deve satisfazer às seguintes propriedades:

1. **Simetria:** $\forall\, x, y \in \mathbb{S}, d(x, y) = d(y, x)$;

2. **Não-negatividade:** $\forall\, x, y \in \mathbb{S}, x \neq y, d(x, y) > 0$ e $d(x, x) = 0$;

3. **Desigualdade triangular:** $\forall\, x, y, z \in \mathbb{S}, d(x, y) \leq d(x, z) + d(z, y)$.

Um caso particular de espaço métrico é o chamado **espaço vetorial com dimensão finita**, ou simplesmente **espaço vetorial**, onde os $e$ elementos que compõem o vetor de características são representados por $e$ coordenadas de valores reais, $(x_1, ..., x_e)$. Nesse caso, as métricas mais comuns são as da família $L_p$ (ou Minkowski), definidas por:

$$L_p((x_1,...,x_e),(y_1,...,y_e)) = \left( \sum_{i=1}^{e} |x_i - y_i|^p \right)^{1/p}$$

A Figura 1 ilustra o conjunto de pontos que estão à mesma distância $r$ a partir de um objeto $s_0 \in \mathbb{S}$, para diferentes funções de distância da família $L_p$. Na figura, a métrica $L_1$, também conhecida como **Distância de Bloco** ou ***Manhattan***, corresponde ao somatório do módulo das diferenças entre as coordenadas. Nesse caso, o conjunto de pontos no plano à mesma distância $r$ da origem forma um losango. A métrica $L_2$ corresponde à função usual para distância entre vetores, conhecida como **Distância Euclidiana**. O conjunto de pontos no plano que estão à mesma distância $r$ considerando a métrica $L_2$ para o ponto de referência forma uma circunferência. A métrica $L_\infty$, conhecida como ***Infinity***, é obtida ao se calcular o limite de $L_p$ quando $p$ tende ao infinito. O conjunto de pontos no plano que estão à mesma distância $r$, considerando a métrica $L_\infty$, do objeto de referência, forma um quadrado.

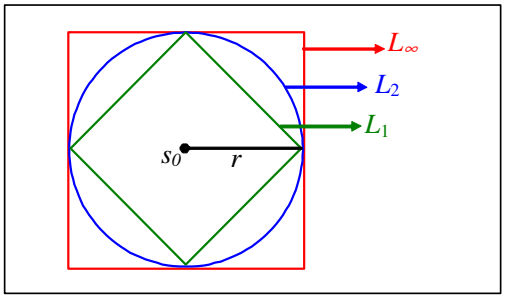
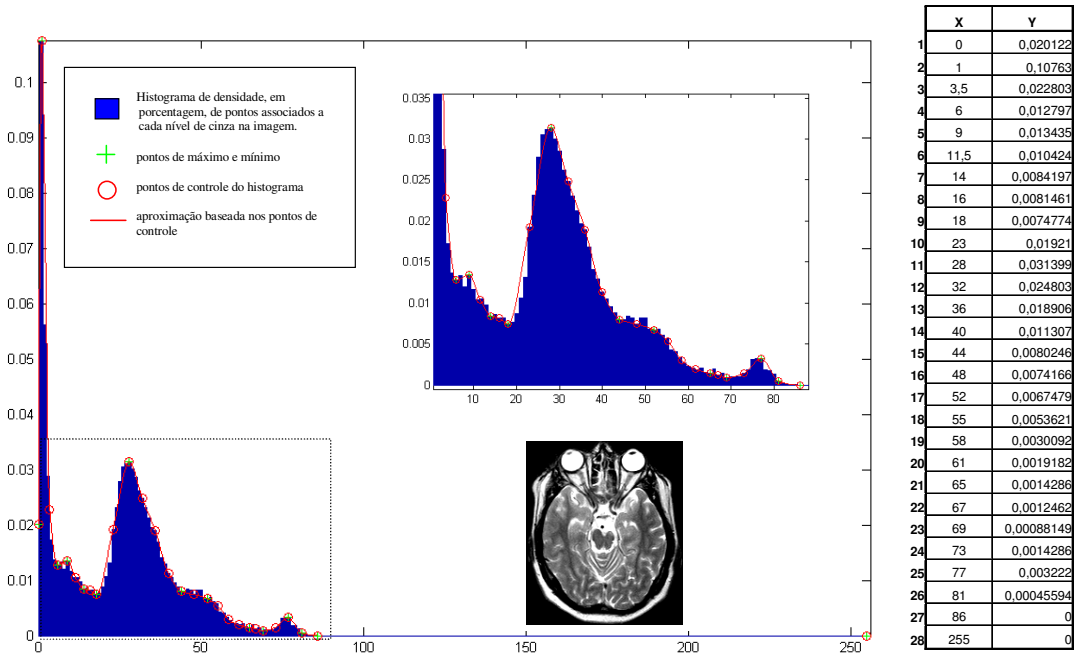



**Figura 1:** Representação dos pontos no plano situados à distância r a partir de um objeto $s_0$, considerando diferentes funções de distância métricas da família $L_p$

A família $L_p$ é bem vasta, e outras variantes podem ser obtidas a partir de sua definição, como por exemplo, o uso de pesos diferentes para cada coordenada. Os três exemplos citados anteriormente foram escolhidos por serem muito utilizados em consultas por similaridade.

Como já mencionado, em alguns domínios de dados complexos não é possível extrair o mesmo número de características de todos os objetos (gerando vetores de características adimensionais), tornando-se necessário definir uma função de distância métrica. Por exemplo, palavras de uma língua podem ser comparadas com a função de distância métrica $L_{Edit}$: considerando duas cadeias de caracteres $x$ e $y$, a distância $L_{Edit}(x, y)$ retorna a quantidade mínima de caracteres que precisam ser substituídos, removidos ou inseridos em $x$ para que se torne igual a $y$. Por exemplo, $L_{edit}$(‘gato’, ‘rato’) = 1 (uma substituição) e $L_{edit}$(‘gato’, ‘gaita’) = 2 (uma substituição e uma remoção).

Outro exemplo de domínio de dados complexos adimensionais são os histogramas métricos [Bueno_2002] [Traina_2002] [Traina_2003] extraídos de imagens. Um histograma métrico é composto por um número variável de *buckets*. Um *bucket* é equivalente ao *bin* do histograma normalizado. No entanto, enquanto o número de *bins* de um histograma depende apenas da resolução de luminosidade da imagem e, portanto, é fixo para uma coleção de imagens obtidas com equipamentos de mesmo tipo, os *buckets* não precisam ser regularmente espaçados e, portanto, seu número é variável. Cada *bucket* corresponde a um segmento de reta obtido pela aproximação linear por partes do histograma original da imagem. Como cada histograma original será aproximado por um conjunto diferente de segmentos de reta, não há um número fixo de *buckets* nos histogramas métricos. A Figura 2 apresenta o histograma original de uma imagem com os pontos de controle que definem o histograma métrico da mesma imagem.




| | X | Y |
|---|---|---|
| 1 | 0 | 0,020122 |
| 2 | 1 | 0,10763 |
| 3 | 3,5 | 0,022803 |
| 4 | 6 | 0,012797 |
| 5 | 9 | 0,013435 |
| 6 | 11,5 | 0,010424 |
| 7 | 14 | 0,0084197 |
| 8 | 16 | 0,0081461 |
| 9 | 18 | 0,0074774 |
| 10 | 23 | 0,01921 |
| 11 | 28 | 0,031399 |
| 12 | 32 | 0,024803 |
| 13 | 36 | 0,018906 |
| 14 | 40 | 0,011307 |
| 15 | 44 | 0,0080246 |
| 16 | 48 | 0,0074166 |
| 17 | 52 | 0,0067479 |
| 18 | 55 | 0,0053621 |
| 19 | 58 | 0,0030092 |
| 20 | 61 | 0,0019182 |
| 21 | 65 | 0,0014286 |
| 22 | 67 | 0,0012462 |
| 23 | 69 | 0,00088149 |
| 24 | 73 | 0,0014286 |
| 25 | 77 | 0,003222 |
| 26 | 81 | 0,00045594 |
| 27 | 86 | 0 |
| 28 | 255 | 0 |

**Figura 2:** Histograma de uma imagem, com os pontos de controle que definem seu histograma métrico. Extraída de [Bueno_2002].

O número de *buckets* em um histograma métrico depende do erro de aceitação no processo de aproximação da curva linear por partes sobre o histograma original da imagem. Cada *bucket k* corresponde a um par $<b_k, h_k>$, onde $b_k$ é o índice do *bin* mais à direita do histograma original representado no *bucket k*, e $h_k$ é o valor normalizado do *bin* mais à direita representado no *bucket k*.

Para fazer o cálculo da distância entre histogramas métricos foi desenvolvido em [Bueno_2002] um novo algoritmo baseado no cálculo da diferença entre histogramas, considerando que cada um deles ocupa uma área caracterizada pela distribuição de *pixels* e que a diferença entre estas áreas indica quão dissimilares são os histogramas. Utilizando essa concepção pode-se concluir que, quando dois histogramas métricos similares são comparados, a diferença entre suas áreas de distribuição é pequena. Formalmente, a função de distância métrica, denominada *DM*, calcula a distância entre dois histogramas métricos, dada pela área não sobreposta entre as duas curvas que representam os histogramas métricos, isto é, dados dois histogramas métricos de duas imagens *A* e *B*, $M_H(A)$ e $M_H(B)$, a distância entre elas é dada por:

$$DM\left(M_H(A), M_H(B)\right) = \int_{x=0}^{passos} \left| M_H\left(A_{<bx,hx>}\right) - M_H\left(B_{<bx,hx}\right) \right| dx$$

A Figura 3 ilustra um exemplo de como calcular a distância entre dois histogramas métricos usando a métrica *DM*( ). Na Figura 3(a) os dois histogramas são sobrepostos, e são mostrados os pontos de intersecção e aqueles que limitam os *buckets*. Nas Figuras 3(b) até 3(d) é mostrado como tais pontos são utilizados para calcular a área dentro de cada região, de acordo com o algoritmo que calcula *DM*( ).

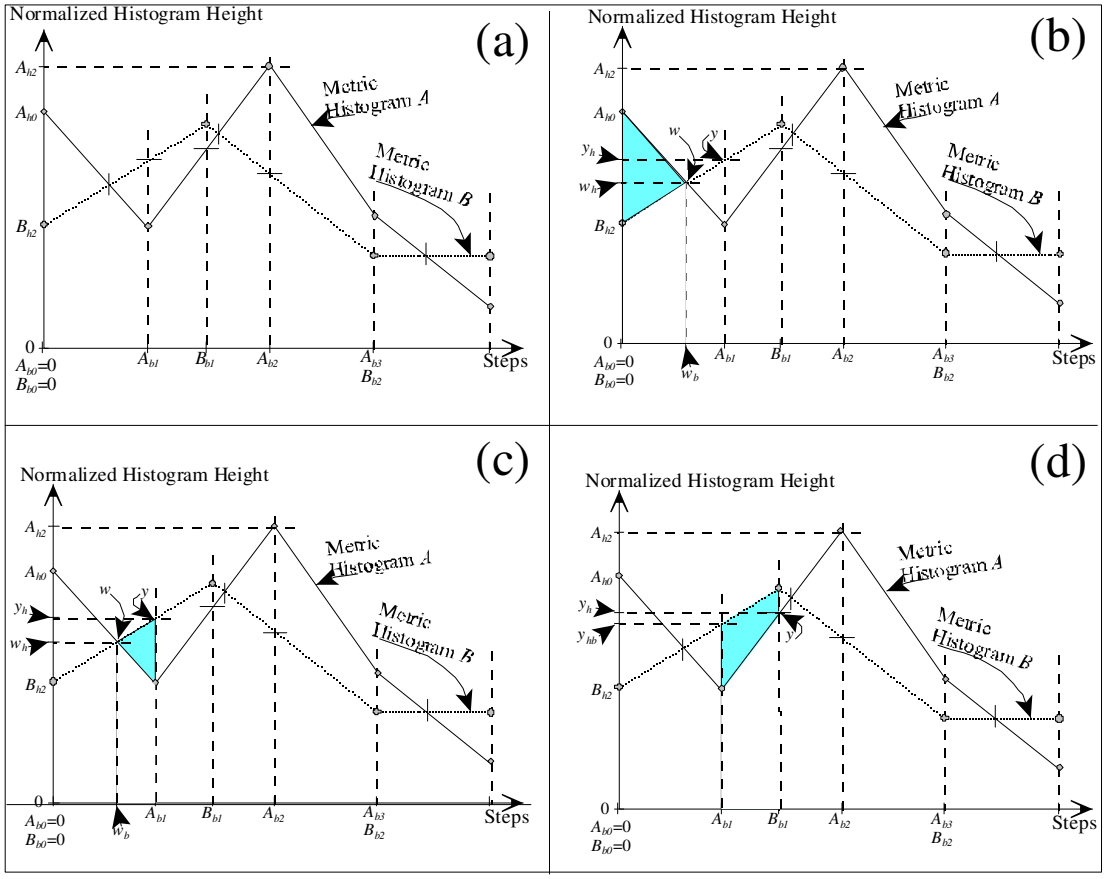



**Figura 3:** Distância entre dois histogramas métricos calculando a área entre eles usando a métrica *DM*( ). (a) Dois histogramas métricos A e B, e os pontos usados para especificar os passos do algoritmo que calcula *DM*( ); (b) Primeiro passo do algoritmo que calcula *DM*( ), exemplificando quando os dois $M_H$ se intesectam; (c) Segundo passo do algoritmo que calcula *DM*( ); (d) Terceiro passo do algoritmo que calcula *DM*( ). Extraída de [Bueno_2002].

### 2.2.2. Tipos de consultas por similaridade

Considerando o espaço métrico $\mathbb{M} = (\mathbb{S}, d)$ e $S \subseteq \mathbb{S}$, os dois tipos fundamentais de consultas por similaridade mais comuns são definidos como [Chávez_2001] [Samet_2006]:

> **Definição 2.3 – Consulta por Abrangência** (*Range Query – RQ*): uma consulta por abrangência recupera todos os objetos que diferem no máximo até dado grau $r$ de um objeto central de busca, ou seja, a consulta $RQ(s_q, r_q)$ visa recuperar objetos situados a uma distância máxima $r_q$ (raio de busca) do objeto central de busca $s_q$, onde $s_q \in \mathbb{S}$ (Figura 4(a)). Formalmente, pretende-se encontrar o subconjunto resposta $R \subseteq S$ que atenda a $R = \{x \in S \mid d(s_q, x) \leq r_q\}$.

Um exemplo de *RQ* é: "Encontre as estrelas que estão a, no máximo, 10 anos-luz de distância do Sol", ou seja, *RQ*('Sol', 10), onde $\mathbb{S}$ é o conjunto dos astros, o subconjunto $S \subseteq \mathbb{S}$ é um banco de dados contendo os astros conhecidos, $d(\,) \equiv L_2$ e a dimensão é 3.

> **Definição 2.4 – Consulta aos *k*-Vizinhos mais Próximos** (*k-Nearest Neighbor Query – KNNQ*): uma consulta aos $k$-vizinhos mais próximos recupera os $k$ objetos mais semelhantes a um objeto de busca, ou seja, a consulta $KNNQ(s_q, k)$ visa a recuperar os $k$ objetos mais próximos do objeto central de busca $s_q$, onde $s_q \in \mathbb{S}$ (Figura 4(b)). Formalmente, pretende-se encontrar o subconjunto resposta $R \subseteq S$ que atenda a $R = \{x \in S \mid |R| = k \text{ e } \forall x \in R, \forall y \in (S - R), d(s_q, x) \leq d(s_q, y)\}$. Em caso de empate, comum onde $d(\,)$ retorna valores discretos, a resposta pode conter mais do que apenas $k$ elementos.

Um exemplo de *KNNQ* é: "Encontre as 5 estrelas mais próximas do Sol", ou seja, *KNNQ*('Sol', 5), onde S é o conjunto dos astros, o subconjunto $S \subseteq \mathbb{S}$ é um banco de dados contendo os astros conhecidos, $d(\,) \equiv L_2$ e a dimensão é 3.

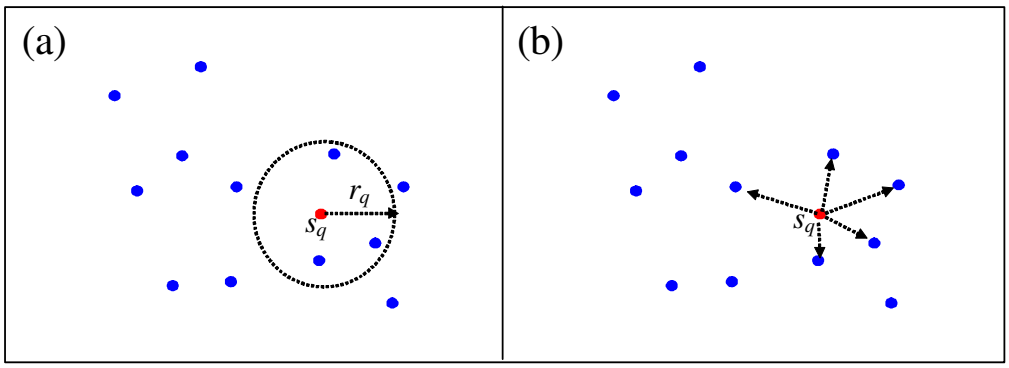



**Figura 4:** Exemplos esquemáticos dos tipos de consultas por similaridade: (a) Consulta por abrangência; (b) Consulta aos 5-vizinhos mais próximos.

Os dois tipos de consulta por similaridade apresentados podem ser facilmente executados a partir da inspeção seqüencial de todos os objetos de um conjunto fornecido. Ou seja, considerando o conjunto de objetos $S$, o objeto de busca $s_q$ e o raio de busca $r_q$, para se responder à consulta $RQ(s_q, r_q)$ cada objeto $s_i$ de $S$ é comparado com o objeto de busca e, se $d(s_q, s_i) \leq r_q$, então $s_i$ é inserido no conjunto resposta $RQ$.

Para a consulta $KNNQ(s_q, k)$, é comum usar como estrutura auxiliar uma lista de prioridade $L_k$, que organize os candidatos a vizinho mais próximo, à medida que são encontrados. A lista de candidatos $L_k$ permanece ordenada pelo valor da distância entre cada candidato e o objeto de busca. O processo pode ser visto como uma variação daquele descrito para $RQ(s_q, r_q)$, com a diferença de que o raio de busca $r_q$ é atualizado à medida que um novo objeto é inserido em $L_k$, passando a valer a distância do último candidato, ou seja, do vizinho mais distante até o momento. Começando com um raio de busca infinito, cada vez que um candidato é inserido na lista, o raio de busca diminui, reduzindo a chance dos próximos objetos comparados serem qualificados como candidatos.

Apesar da simplicidade dos processos descritos, se a cardinalidade do conjunto $S$ for muito alta ou se a função de distância métrica utilizada envolver algoritmos muito demorados, o tempo total para a obtenção da resposta pode se tornar inaceitável, sendo necessário o uso de alguma técnica de indexação para agilizar tais processos. A Seção 2.3 apresenta algumas técnicas de indexação para dados complexos.

## 2.3. Estruturas de indexação para dados complexos

O desempenho das consultas por similaridade em ambientes altamente dinâmicos, isto é, ambientes com muitas operações de inserção e remoção de dados, é afetado por dois fatores principais: quantidade de acessos a disco e quantidade de comparações entre objetos efetuadas por cálculos de distância. O disco é acessado para a obtenção dos dados armazenados, pois, em geral, a quantidade e o tamanho dos objetos são tão grandes que é inviável armazená-los em memória principal. O tempo de comparação depende da complexidade algorítmica da função de distância usada para avaliar o grau de dissimilaridade entre os objetos. Quanto maior o número de comparações e/ou quanto mais complexa a função de distância, pior será o desempenho das consultas [Hjaltason_2003].

Consultas por similaridade podem ter seu desempenho melhorado a partir do uso de estruturas de indexação que sejam capazes de gerenciar eficientemente o armazenamento e a recuperação em memória secundária (disco). Para tanto, diferentes técnicas de indexação, também chamadas de Métodos de Acesso (MA), têm sido propostas [Gaede_1998] [Böhm_2001] [Traina Jr._2000b] [Santos Filho_2001] [Traina Jr._2002b] [Hjaltason_2003] [Vieira_2004].

De acordo com [Korn_2001], a estrutura de indexação dos MAs se assemelha com a de uma técnica de indexação muito utilizada para dados convencionais, a $B^+$-*Tree* [Comer_1979]: os objetos são armazenados em nós folhas visando agrupar aqueles com alto grau de semelhança entre si. Cada objeto é armazenado em exatamente um nó. Os nós folhas são organizados hierarquicamente por meio de nós internos, que também procuram agrupar as folhas e as subárvores de modo a manter juntos os objetos mais semelhantes. Cada entrada de um nó interno "aponta" para exatamente uma subárvore ou uma folha. Habitualmente, a estrutura das entradas nas folhas é diferente da dos nós internos, sendo que as entradas de todos os nós internos apresentam a mesma estrutura. A estrutura das entradas é específica para cada método. Como em qualquer estrutura hierárquica, todas as operações de manipulação da árvore (inserção, remoção e consultas) são iniciadas pelo nó raiz, o qual armazena os endereços das demais subárvores. Em geral, é interessante que essas estruturas sejam balanceadas pela altura, ou seja, todas as folhas se encontram no mesmo nível da árvore.

De acordo com os modelos definidos na Seção 2.2, os MAs podem ser divididos em duas classes:

- **Métodos de Acesso Espaciais** (MAEs), ou Métodos de Acesso a Dados Espaciais: são voltados para o modelo de espaço vetorial, onde os objetos são representados por vetores em um espaço *e*-dimensional. Exemplos de MAEs dinâmicos são: a *R-Tree* [Guttman_1984] e suas variantes, $R^*$*-Tree* [Beckmann_1990] e $R^+$*-Tree* [Sellis_1987], a *k-d-B-Tree* [Robinson_1981], a *TV-Tree* [Lin_1994] e a *SR-Tree* [Katayama_1997];

- **Métodos de Acesso Métricos** (MAMs), ou Métodos de Acesso a Dados Métricos: são voltados para o modelo de espaço métrico, onde apenas a distância entre os objetos é levada em consideração. Exemplos de MAMs dinâmicos são: a *M-Tree* [Ciaccia_1997], a *Slim-Tree* [Traina Jr._2000b] [Traina Jr._2002a], métodos da família *OMNI* [Santos Filho_2003] [Traina Jr._2005], a *DF-Tree* [Traina Jr._2002b] e a *DBM-Tree* [Vieira_2004] .

Os MAMs surgiram como uma alternativa aos MAEs, pois os superam ao processarem de modo eficiente consultas por similaridade tanto com tipos de dados vetoriais, quanto com tipos de dados adimensionais (não-vetoriais). Porém, enquanto existem na literatura várias propostas de MAEs com capacidade de gerenciar armazenamento em memória secundária, tendo os primeiros surgido por volta da década de 1980 [Gaede_1998], apenas em 1997 foi propsto o primeiro MAM realmente dinâmico e com suporte a disco, a *M-Tree* [Ciaccia_1997], seguida pela *Slim-Tree* [Traina Jr._2000b] em 2000.

É importante ressaltar que todos os MAs são capazes de executar as mesmas operações, tais como a inserção individual de objetos, bem como as mesmas consultas básicas por similaridade. A diferença entre eles está no desempenho que cada um apresenta em cada consulta. Assim, embora as estruturas de indexação baseadas em árvore apresentem desempenho muito bom para consultas com alta seletividade, elas tendem a degradar quando uma consulta retorna mais do que (tipicamente) 10% dos objetos indexados (este resultado faz parte de conhecimento já bastante difundido e aceito pela comunidade de bases de dados [DeWitt_1991]). Logo, quando isso acontece, é preferível utilizar a busca seqüencial, mesmo

que exista um índice criado para o atributo de busca. Esse é um exemplo simples de um tipo de escolha que um processo de otimização de consultas deve fazer.

A seção seguinte apresenta uma visão geral do MAM dinâmico *Slim-Tree*, o qual será usado como base para a criação do modelo de custo proposto por este trabalho.

## 2.4. O MAM *Slim-Tree*

A *Slim-Tree* [Traina Jr._2000b] [Traina Jr._2002a] é um MAM dinâmico, consistindo em uma árvore balanceada que cresce *bottom-up*, ou seja, das folhas para a raiz. Como em outras árvores métricas (por exemplo, a *M-Tree*), os objetos que compõem o conjunto de dados são agrupados em páginas de disco de tamanho fixo, onde cada página corresponde a um nó da árvore.

A idéia geral de todo MAM consiste em selecionar um ou mais objetos (representantes) do conjunto de objetos e organizar os demais a partir deles. A *Slim-Tree* armazena todos os objetos nas folhas, organizando-os hierarquicamente na árvore. Essa hierarquia é construída a partir da seleção de objetos, denominados representantes, que definem centros de regiões no espaço de dados. Cada região possui um raio de cobertura, e apenas os objetos que forem cobertos pelo raio de cobertura de uma determinada região podem ser armazenados nesse nó. As entradas em um nó folha (*LeafNode*) são formadas pelos dados que compõem o objeto indexado, por seu código de identificação e pelo valor da distância entre ele e seu representante. Assim, a estrutura dos nós folhas que armazenam todos os objetos é:

$$LeafNode \text{ [vetor de } <Oid_i , d(s_i , rep(s_i)), s_i>]$$

onde, $Oid_i$ é o identificador do objeto $s_i$ e $d(s_i, rep(s_i))$ é a distância entre o objeto $s_i$ e o representante deste nó folha $rep(s_i)$.

As entradas de um nó interno, denominado nó índice (*IndexNode*), são compostas pelos dados de uma subárvore, ou seja, o objeto representante, o raio e o ponteiro para a subárvore; e, se a entrada não estiver na raiz, pela distância entre esse objeto e o seu representante armazenado no nó pai. A estrutura dos nós índices é a seguinte:

*IndexNode* [vetor de $<s_i, r_i, d(s_i, rep(s_i)), ptr(Ts_i), Nentries(ptr(Ts_i))>$]

onde, $s_i$ armazena o objeto que é o representante da subárvore apontada por $ptr(Ts_i)$, e $r_i$ é o raio de cobertura da região. A distância entre $s_i$ e o centro deste nó $rep(s_i)$ é armazenada em $d(s_i, rep(s_i))$. O ponteiro $ptr(Ts_i)$ indica o nó raiz da subárvore cuja raiz é $s_i$. O número de entradas presentes nos nós apontados por $ptr(Ts_i)$ é armazenado em $Nentries(ptr(Ts_i))$.

A Figura 5 apresenta uma visão geral da organização de 19 objetos, rotulados de A até S, armazenados em um *Slim-Tree* de 3 níveis, onde a raiz encontra-se no nível zero e os objetos no nível das folhas (nível 2), com nós com capacidade máxima 3.

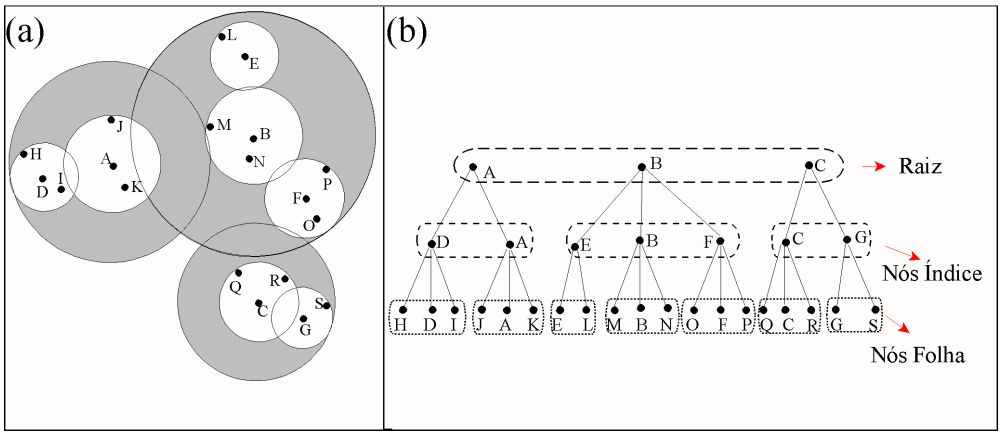



**Figura 5:** Exemplo de Slim-Tree: (a) representação estrutural; (b) representação hierárquica com os representantes e seus raios.

Assim como outras estruturas *bottom-up* (por exemplo, a *B-Tree*), o mecanismo de inserção de objetos na *Slim-Tree* é o seguinte: para cada novo objeto a ser inserido, o algoritmo de inserção percorre a árvore a partir da raiz para encontrar um nó folha cujo raio de cobertura possa abranger o novo objeto; se nenhum nó se qualifica, seleciona-se o nó cujo centro está mais perto do novo objeto; caso mais de um nó se qualifique, o algoritmo *ChooseSubtree*() é executado para selecionar o nó onde será inserido o novo objeto. Esse processo é aplicado recursivamente para todos os níveis da árvore. A *Slim-Tree* possui três opções para o algoritmo *ChooseSubtree*():

- *Random* (Aleatório) - seleciona aleatoriamente, dentre os nós que se qualificam, um nó para inserir o novo objeto;

- *MinDist* (Distância Mínima) – dentre os que se qualificam, seleciona o nó cuja distância de seu representante para o novo objeto seja a menor;

- *MinOccup* (Ocupação Mínima) - seleciona o nó que esteja com o menor número de objetos armazenados, dentre os que se qualificam. Essa é a opção padrão.

É interessante notar que utilizando a opção *MinOccup* do algoritmo *ChooseSubtree*() obtém-se árvores mais compactas (com maior taxa de ocupação dos nós), o que resulta em um número menor de acessos a disco para responder consultas por similaridade. Entretanto, a taxa de sobreposição entre os nós aumenta. Já a opção *MinDist* tende a gerar árvores mais altas e com menor taxa de ocupação e sobreposição de nós.

Durante o processo de inserção de objetos pode acontecer do nó escolhido já ter atingido a sua taxa de ocupação máxima. Nesse caso deve-se alocar um novo nó no mesmo nível do anterior, e os objetos que estavam nesse nó, mais o novo objeto a ser inserido devem ser então redistribuídos entre os dois nós. A *Slim-Tree* possui as seguintes opções para efetuar a quebra de nós (*splitting*):

- *Random* (Aleatório) - seleciona aleatoriamente os dois objetos representantes para os novos nós, e os demais objetos são distribuídos entre eles pela menor distância entre o objeto e o representante. Deve-se respeitar a taxa de ocupação mínima dos nós;

- *MinMax* (Mínimo dos Maiores Raios) - consideram-se como candidatos a representantes todos os possíveis pares de objetos. Associa-se, tentativamente, a cada objeto do par de representantes os demais objetos. Serão escolhidos como representantes o par de objetos que minimizar o raio de cobertura da subárvore resultante;

- MST (*Minimal Spanning Tree*) - constrói-se a árvore de caminho mínimo, MST [Kruskal_1956], e a aresta mais longa da MST é removida. Dessa maneira obtém-se dois agrupamentos, e o objeto mais central de cada um dos dois agrupamentos resultantes é selecionado como representante do nó. Essa opção produz *Slim-Trees* tão boas quanto as criadas utilizando a opção *MinMax*, em uma fração do tempo. Assim, essa é a opção padrão de quebra de nós.

Note-se que a *Slim-Tree* cresce um nível quando a raiz da árvore está completa e um novo elemento deve ser inserido nela. Nesse caso a raiz divide-se e uma nova raiz deve ser criada com dois representantes, e dessa maneira a árvore cresce um nível.

Uma medida importante a ser obtida a partir de uma *Slim-Tree* é o *fat-factor*, que permite determinar quanto uma árvore métrica está próxima de ser ótima, isto é, sem sobreposição de nós. Assim, dado que $T$ é uma árvore métrica de altura $H$ e com $N$ nós, $N \geq 1$, e que $|S|$ é o total de objetos de $S$, o *fat-factor* da árvore métrica $T$ é [Traina Jr._2000b]:

$$fat(T) = \frac{I_c - H \cdot |S|}{|S|} \cdot \frac{1}{(N - H)}$$

onde $I_c$ é o número total de nós acessados para responder uma consulta pontual (*point query*) para cada objeto na árvore e $H \cdot |S| \leq I_c \leq N \cdot |S|$, ou seja, $H \cdot |S|$ ocorre para uma árvore ótima e, neste caso, $fat(T)=0$; $N \cdot |S|$ ocorre no pior caso de sobreposição e, então, $fat(T)=1$. Assim, *fat(T)* retorna valores no intervalo [0,1].

A Figura 6 ilustra quatro casos de sobreposição de nós e seus respectivos *fat-factors*. Nessa figura, o representante de um nó, que está no centro do mesmo, está sendo indicado conectado ao elemento mais distante dele no nó, o que também delineia o raio desse nó. Considerando uma árvore métrica $T$ de dois níveis, ou seja, a raiz e o nível mostrado na figura, tem-se $H=2$, $|S|=13$ e $N=3$. Para o primeiro e o segundo caso, $I_c=26$ e, portanto, $fat(T)=0$; no terceiro caso $I_c=28$, resultando em $fat(T)=2/13=0.15$; finalmente, para o quarto caso $I_c=39$ e, então, $fat(T)=1$.

É importante ressaltar que, usando o *fat-factor* e mais um mecanismo para reorganizar a árvore (*Slimdown*), ambos disponíveis na implementação padrão da *Slim-Tree*[1], é possível reduzir a sobreposição de nós da árvore.

[1] Presente na plataforma *Arboretum*, encontrada em http://gbdi.icmc.usp.br/downloads.php

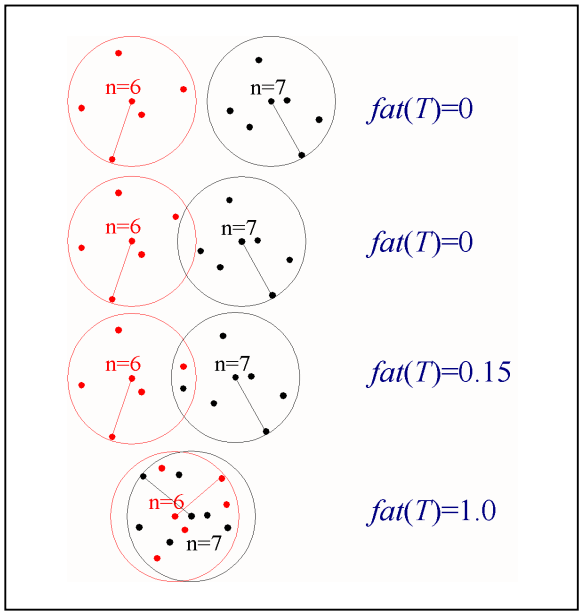



**Figura 6:** Exemplo de sobreposição entre dois nós de uma árvore métrica *T*, ilustrando o melhor caso com *fat(T)*=0, o pior caso com *fat(T)*=1.0 e um caso intermediário *fat(T)*=0.15.

Seguindo a proposta de [Korn_2001], neste trabalho será utilizada a *Slim-Tree*, a qual segue a abordagem de uma técnica de indexação muito utilizada para dados convencionais, a $B^+$-*Tree*, e também por ser a única que permite quantificar a sobreposição entre nós, que é o principal problema de MAMs. À medida que a sobreposição de nós aumenta, a eficiência das estruturas de indexação diminui, uma vez que todos os nós cobertos por uma região de consulta têm que ser processados durante as operações de busca.

## 2.5. Considerações finais

Consultas por conteúdo em dados multimídia utilizam o critério de similaridade (semelhança), sendo assim denominadas consultas por similaridade. Em ambientes dinâmicos (ambientes com inserção e remoção de dados, após a criação da estrutura de dados), o desempenho de consultas por similaridade pode ser comprometido, sendo necessário o uso de técnicas de indexação para dados complexos para solucionar esse problema. É importante ressaltar que o custo computacional para efetuar consultas por similaridade tem ordem de grandeza maior do que para processar consultas tradicionais. Desse modo, a possibilidade de poder estimar o número de operações necessárias para processar consultas por similaridade propicia o conhecimento de um parâmetro importante para o otimizador de consultas de um SGBD.

Em relação à *Slim-Tree*, de acordo com os conceitos apresentados neste capítulo pode-se concluir que é uma estrutura de indexação que permite realizar consultas por similaridade de maneira eficiente, minimizando tanto o número de cálculos de distância quanto o de acessos a disco. Outro aspecto importante é que a *Slim-Tree* foi desenvolvida com o objetivo de minimizar a sobreposição de nós, provendo mecanismos para mensurar o grau de sobreposição entre eles, bem como reorganizar os dados na árvore de modo a diminuir tal sobreposição.

# 3. OTIMIZAÇÃO DE CONSULTAS POR SIMILARIDADE

## 3.1. Introdução

No momento da solicitação de uma consulta, os SGBDs criam um roteiro de execução da consulta, pré-avaliando diversas alternativas, visando otimizar sua execução. Para isso, existe um módulo nos SGBDs apoiados no modelo relacional, denominado **otimizador de consultas**, que avalia diversos fatores que podem afetar o desempenho do processo de execução de uma consulta incluindo, entre outras: utilização de estruturas de indexação, seqüência das operações, quais operadores utilizar (propriedades algébricas permitem expressar a mesma consulta de várias maneiras) [Traina Jr._2006] e configuração da memória disponível para *cache* das relações [O'Neil_2001] [Elmasri_2003].

Para alcançar esse objetivo, o otimizador de consultas realiza operações de **estimativa de seletividade** e **previsão de custo** de acesso aos dados. Funções de estimativa de seletividade e modelos de custo para consultas em dados convencionais são amplamente utilizadas pelos SGBDs relacionais atuais. Em relação à otimização de consultas por conteúdo em dados complexos, alguns trabalhos iniciais têm sido efetuados na direção de estimativas de seletividade e modelos de custo em espaços métricos, os quais são abordados neste capítulo.

A Figura 7 ilustra os passos típicos para a execução de uma consulta em um SGBD, destacando o módulo otimizador de consultas, que é o alvo deste trabalho. Inicialmente a consulta passa pela **análise léxica**, que identifica os elementos léxicos da linguagem existentes no texto da consulta, seguida da **análise sintática**, que analisa a consulta para determinar se ela está formulada de acordo com as regras sintáticas da linguagem de consulta, e seguida de uma **validação** que verifica se todos os atributos e relacionamentos são válidos de acordo com a semântica do banco de dados a ser consultado. Essa fase inicial gera uma representação interna da consulta a ser utilizada pelo **otimizador de consultas** que tem a

função de produzir um plano de execução eficiente para a consulta, o qual será utilizado pelo **gerador de código** para gerar o código que irá executar aquele plano. Finalmente, o **processador em tempo de execução** executa o código da consulta, a fim de obter o resultado aguardado.

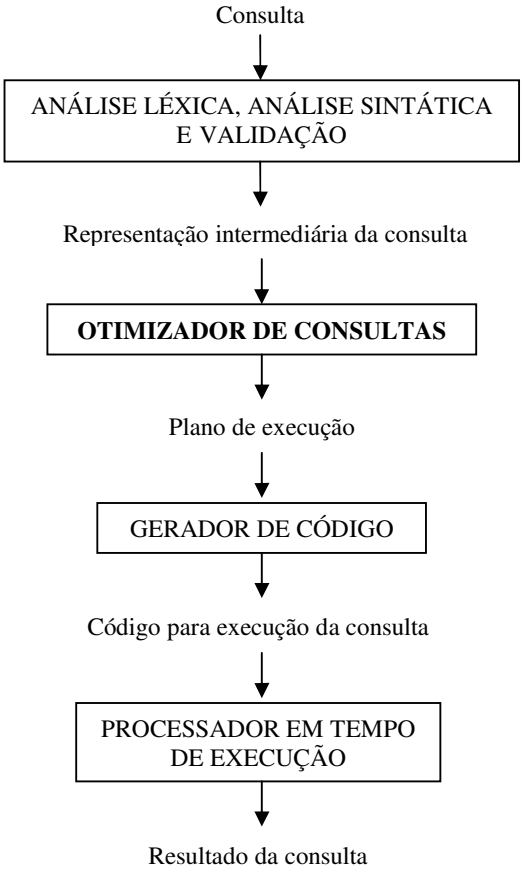



**Figura 7:** Passos para o processamento, otimização e execução de uma consulta por um SGBD.

A estimativa de seletividade (Seção 3.2) é o primeiro passo para se desenvolver equações de estimativa do custo computacional de uma consulta (Seção 3.3). O objetivo é tentar saber, de antemão, qual será o custo da consulta, visando a alterar o plano de execução e melhorar o desempenho final. Ou seja, se existirem opções diferentes para executar uma consulta, as estimativas de seletividade e de custo podem indicar qual opção é mais adequada para recuperar os dados, com o menor custo computacional possível.

## 3.2. Estimativa de seletividade para consultas por similaridade

Estimar a seletividade de consultas significa estimar a proporção de objetos que farão parte do conjunto resposta da mesma, em relação ao total de objetos armazenados. Assim, dado um conjunto de objetos *S* e uma consulta por abrangência (*Range Query – RQ*, abordada na Seção 2.2.2) definida por $RQ(s_q, r_q)$, onde $s_q$ e $r_q$ são, respectivamente, o objeto e o raio de busca, estimar a seletividade de $RQ(s_q, r_q)$ aplicada em *S* significa estimar a quantidade de objetos de *S* que estão na região de busca definida por $RQ(s_q, r_q)$.

A partir da estimativa de seletividade é possível prever o custo computacional de uma consulta, o que inclui o número de acessos a disco, a quantidade de memória e o tempo total necessários para realizar a consulta. Supondo a existência de diferentes métodos de acesso, pode-se decidir qual deles deve ser usado para otimizar o plano de execução visando a redução do custo da consulta. Há casos em que uma busca seqüencial simples pode ser menos onerosa do que o uso de uma estrutura de indexação, por exemplo, quando o raio de busca de uma consulta por abrangência é relativamente grande, em comparação com o diâmetro do conjunto de dados.

De acordo com [Belussi_1995] [Böhm_2000] [Gunopulos_2005], o principal fator que influencia a seletividade de consultas por similaridade é o conjunto de objetos onde as consultas serão efetuadas, mais especificamente a quantidade de objetos do conjunto, a distribuição dos objetos no espaço (métrico ou vetorial) e as dimensões do espaço. A Figura 8 ilustra alguns fatores que influenciam a seletividade de consultas por similaridade, considerando consultas por abrangência: as Figuras 8(a) e 8(b) mostram diferentes distribuições dos objetos no espaço; a Figura 8(a) mostra que o tamanho do raio em relação ao diâmetro do conjunto de dados também influencia a seletividade, mesmo com os objetos uniformemente distribuídos; quando a distribuição dos objetos não é uniforme, a posição do objeto central de busca em relação aos demais objetos do conjunto influi na seletividade, ou seja, como pode ser observado na Figura 8(b), os objetos $s_1$ e $s_3$ estão em posições diferenciadas, porém $r_1 = r_3$, mas com seletividades diferentes. Assim, o problema que surge é como modelar a distribuição dos objetos.

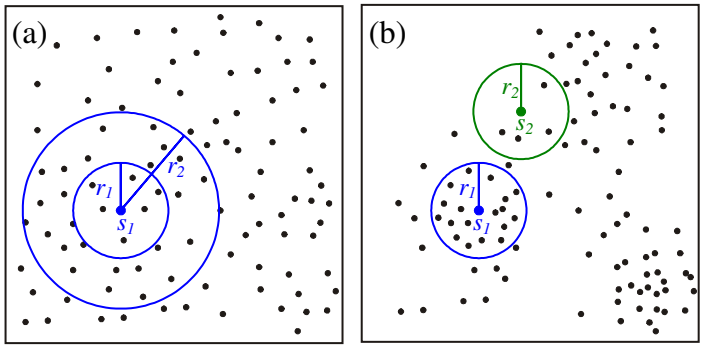



**Figura 8:** Consultas por abrangência e suas respectivas seletividades: (a) dados uniformemente distribuídos e consultas $RQ_1(s_1, r_1)$ e $RQ_2(s_1, r_2)$ com mesmo centro e $r_1 < r_2$; (b) dados agrupados e as consultas $RQ_1(s_1, r_1)$ e $RQ_2(s_2, r_2)$ com centros diferentes e $r_1 = r_2$.

Considerando o modelo de espaço vetorial, estudos iniciais sobre estimativa de seletividade pressupõem que os objetos estão uniformemente distribuídos no espaço. Nesse caso, a estimativa de seletividade é obtida considerando a dimensão em que os dados estão imersos no espaço, denominada **dimensão de imersão** (*embedded dimension*). Em [Faloutsos_1994], Faloutsos e Kamel questionam a suposição da uniformidade da distribuição dos objetos no espaço, argumentando que as coordenadas (dimensões) dos vetores em conjuntos de dados reais tendem a estar correlacionadas, levando a uma distribuição não uniforme. Segundo [Böhm_2000] [Samet_2006], a correlação de dimensões significa que os objetos estão distribuídos em uma dimensão mais baixa do espaço. Desse modo, a dimensão a ser considerada para a estimativa de seletividade pode ser mais baixa que a dimensão de imersão, denominada **dimensão intrínseca**.

A **dimensão de imersão** de um conjunto de dados em um espaço vetorial é dada pelo número total de coordenadas que definem o espaço. A Figura 9 apresenta conjuntos de dados pontuais alinhados *M, N* e *P*, considerados imersos em espaços de uma, duas e três dimensões, respectivamente. Porém, usar três dimensões para caracterizar o conjunto *P* demanda a utilização de um espaço de memória muito maior do que o realmente ocupado pelo conjunto.

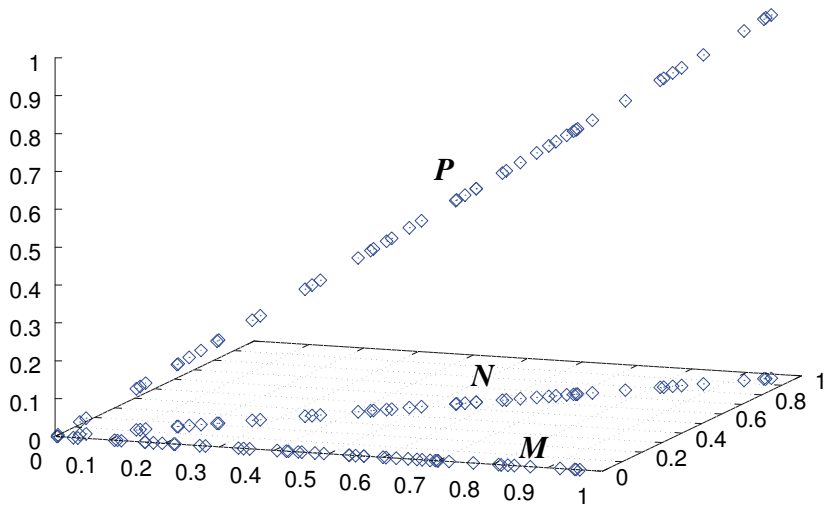



**Figura 9:** Conjuntos de dados pontuais M, N e P, distribuídos ao longo de uma linha e imersos em uma (M), duas (N) e três (P) dimensões. Extraída de [Santos Filho_2003].

A **dimensão intrínseca** (*intrinsic dimension*) de um conjunto de dados corresponde ao número mínimo de variáveis livres necessárias para representar os dados. Assim, um conjunto definido em um espaço *e*-dimensional (dimensão de imersão) possui dimensão intrínseca *D* ($D \leq e$), se os dados podem ser totalmente imersos em um subespaço *D*-dimensional. Seguindo essa propriedade, verifica-se que a dimensão intrínseca dos conjuntos *M*, *N* e *P* na Figura 9 é sempre um.

Segundo [Korn_2001], a **maldição da dimensionalidade** (*dimensionality curse*) é um problema que ocorre na indexação de dados em altas dimensões, no qual a eficiência degrada exponencialmente em função do aumento da dimensionalidade. Tanto nos MAEs quanto nos MAMs, indexar vetores em alta dimensão provoca problemas como o aumento da sobreposição de nós e do custo de processamento decorrente da comparação entre os objetos (cálculos de distância), resultando na necessidade de métodos para redução da dimensão dos conjuntos de dados [Aggarwal_2004] [Ye_2005] [Samet_2006]. Isso faz com que, em altas dimensões, a busca seqüencial possa se tornar mais eficiente do que com o uso de uma estrutura de indexação. Em [Weber_1998] [Beyer_1999] observa-se que, para dados uniformemente distribuídos, acima de 15 dimensões nenhum MAE seria mais eficiente que a busca seqüencial. Assim, considerando a dimensão de imersão do conjunto de dados pode-se chegar a uma estimativa de seletividade pessimista, que pode não ser real caso a dimensão intrínseca do conjunto seja menor que a de imersão.

A distinção entre dimensão de imersão e dimensão intrínseca tem sido muito utilizada no contexto de dados espaciais para avaliar quanto a distribuição de um conjunto diverge da distribuição uniforme [Korn_2001]. A importância da dimensão intrínseca é ainda maior no contexto dos espaços métricos, onde não se aplica a definição de dimensão de imersão.

Em [Faloutsos_1994], Faloutsos e Kamel propõem o uso da teoria de fractais para estimar a dimensão intrínseca de dados espaciais e mensurar o quanto a distribuição do conjunto diverge da distribuição uniforme. Com base nos resultados apresentados em [Faloutsos_1994], Belussi e Faloutsos apresentam fórmulas de estimativa de seletividade para consultas em dados espaciais [Belussi_1995] [Belussi_1998]. Um trabalho equivalente voltado para dados métricos pode ser encontrado em [Traina Jr._2000a].

Os trabalhos [Faloutsos_1994] [Belussi_1998] [Traina Jr._2000a] [Korn_2001] ressaltam que conjuntos de dados reais tendem a ter dimensão intrínseca razoavelmente baixa, sendo que, de todos os conjuntos testados, nenhum apresentou valor superior a 10 (a dimensão de imersão variou entre 2 e 16). Outra observação é que conjuntos uniformemente distribuídos apresentam dimensão intrínseca igual à sua dimensão de imersão, o que torna a razão entre esses valores uma maneira simplificada de avaliar quanto a distribuição do conjunto diverge da distribuição uniforme.

Uma maneira de se obter a dimensão intrínseca de um conjunto de dados, a qual será usada por este trabalho, é usar sua dimensão fractal. Desse modo, as sessões seguintes apresentam respectivamente, o conceito de dimensão de correlação fractal e as fórmulas para estimativa de seletividade em consultas espaciais, de acordo com [Belussi_1995].

### 3.2.1. Dimensão de correlação fractal

Um conjunto de pontos é classificado como um fractal se o mesmo apresentar a propriedade de auto-similaridade (exata ou estatística) em uma ampla faixa da escala de visualização. Como exemplo pode-se citar o triângulo de Sierpinsky, cuja construção teórica é descrita pelo processo ilustrado na Figura 10 e descrita a seguir: dado um triângulo eqüilátero ABC, retira-se o triângulo central A'B'C'. Dos três triângulos eqüiláteros remanescentes, cujos lados têm

comprimento igual à metade do lado do triângulo original, retira-se novamente o triângulo central, e assim sucessivamente.

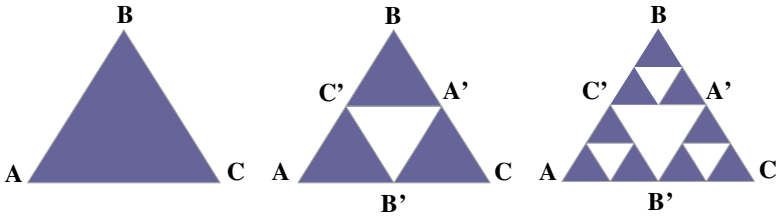



**Figura 10:** Três primeiras iterações da construção do triângulo de Sierpinski.

Pode-se observar na Figura 11 que, após várias iterações, o triângulo resultante possui "buracos" em qualquer escala, sendo que cada triângulo interior é uma miniatura do todo. O que caracteriza um fractal é justamente essa propriedade de auto-similaridade, ou seja, as partes do fractal são similares (exatamente ou estatisticamente) ao fractal como um todo.

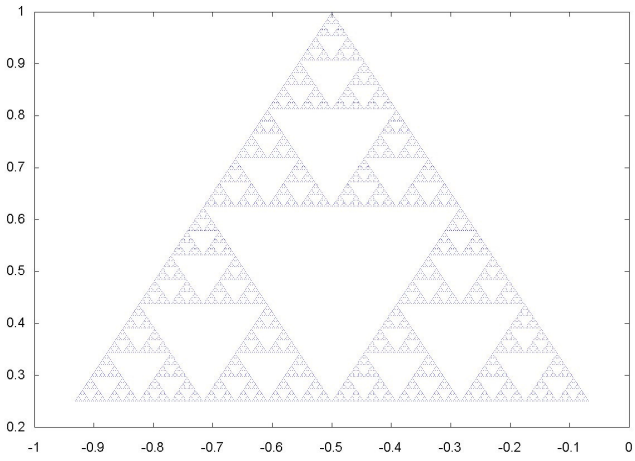



**Figura 11:** Triângulo de Sierpinski após várias iterações

Os fractais possuem características não muito convencionais. Por exemplo, o triângulo de Sierpinsky possui as seguintes propriedades:

- **área nula:** supondo a área do primeiro triângulo da Figura 11 igual a uma unidade, após o primeiro passo a área passará a ser 3/4. Após $i$ iterações, a área passará a valer $(3/4)^i$ e, dessa forma, após infinitas iterações, a área será proporcional a $\lim_{i\to\infty}(3/4)^i$;

- **perímetro infinito:** supondo o perímetro do primeiro triângulo da Figura 11 igual a uma unidade, após o primeiro passo o perímetro passará a ser (1 + (1/2)). Após *i* iterações, o perímetro passará a valer $(1 + (1/2))^i$ e, dessa forma, após infinitas iterações, o perímetro será proporcional a $\lim_{i\to\infty}(1 + (1/2))^i$.

Desse modo, o triângulo de Sierpinsky não corresponde a um objeto Euclidiano de dimensão unitária, caso contrário teria perímetro finito, e nem a um objeto Euclidiano de dimensão 2, visto que possui área nula. Esse problema é resolvido ao se considerar dimensões fracionárias, também chamadas de **dimensões fractais**.

Existem vários métodos para se calcular dimensões fractais [Schroeder_1991]. De acordo com [Belussi_1995], a **Dimensão de Correlação Fractal** é muito útil para o estudo de estimativas de seletividade, sendo um caso especial da **Dimensão Fractal Generalizada.**

A **Dimensão Fractal Generalizada** representa uma família de dimensões fractais voltada para conjuntos finitos estatisticamente auto-similares. Considerando um conjunto de pontos *P* imerso em um espaço *e*-dimensional (dimensão de imersão *e*), o qual é dividido por um (hiper-) quadriculado com células de lado *r*; a porcentagem $p_i$ de pontos que incidem na *i*-ésima célula; e a potência de peso *q* associada a todas as células, então a família de dimensões fractais generalizadas $D_q$ é definida da seguinte forma:

> **Definição 3.1 – Dimensão Fractal Generalizada $D_q$:** para um conjunto de pontos *P* com a propriedade de auto-similaridade no intervalo de escalas ($r_1$, $r_2$), a dimensão fractal generalizada $D_q$ é obtida por:

$$D_q \equiv \frac{1}{q-1}\frac{\partial \log \sum_i p_i^q}{\partial \log(r)} = \text{constante} \qquad \text{onde: } q \neq 1,\ r \in (r_1, r_2) \qquad (3.1)$$

Como dito anteriormente, um caso especial é o da **Dimensão de Correlação Fractal**, $D_2$, onde $q = 2$. Considerando $q = 2$, a Equação 3.1 fica:

$$D_2 \equiv \frac{\partial \log \sum_i p_i^2}{\partial \log(r)} = \text{constante} \qquad (3.2)$$

Como a dimensão de correlação fractal caracteriza o grau de dependência entre as coordenadas do espaço onde o fractal está definido (ou seja, os atributos que o definem) [Schroeder_1991], ela pode ser usada também para prever o grau de coesão entre os elementos do fractal. A seção seguinte ilustra uma aplicação da teoria dos fractais e da dimensão de correlação fractal para estimar a quantidade de objetos que são recuperados em consultas espaciais sobre objetos que apresentam auto-similaridade.

### 3.2.2. Estimativa de seletividade em consultas espaciais

Esta seção apresenta as equações para estimativa de seletividade de consultas espaciais de acordo com [Belussi_1995], as quais utilizam o conceito de dimensão de correlação fractal abordado na seção anterior. Todo o processo de dedução das fórmulas apresentadas pode ser encontrado também em [Belussi_1995].

Os tipos de consultas espaciais considerados em [Belussi_1995] são as **consultas espaciais por abrangência** (*range query*) e as **junções espaciais**. Uma consulta espacial por abrangência em um conjunto de pontos *P* visa obter os pontos que fazem interseção com a região de busca definida no espaço. Uma consulta de junção espacial em um conjunto de pontos *P* visa obter todos os pares únicos de pontos distintos, cuja distância entre eles seja menor ou igual a um raio $\varepsilon$.

A seletividade para consultas espaciais por abrangência e junções espaciais pode ser expressa, respectivamente, pelas seguintes equações:

$$Sel_{range}(\varepsilon) = \frac{\overline{nb}(\varepsilon) + 1}{N} \tag{3.3}$$

e

$$Sel_{join}(\varepsilon) = \frac{\overline{nb}(\varepsilon)}{(N-1)} \tag{3.4}$$

Onde a função $\overline{nb}(\varepsilon)$ retorna o número médio de vizinhos à distância máxima $\varepsilon$ de um ponto qualquer e *N* é o número de pontos contidos no conjunto de pontos *P*. Para estimar $\overline{nb}(\varepsilon)$,

considerando-se o formato da região de busca (quadrado, círculo, diamante entre outros, que são especificados pela função de distância empregada na consulta por similaridade), denominado *shape*, utiliza-se o seguinte teorema:

> **Teorema 3.1 –** Dados: um conjunto de pontos $P$; o número de pontos $N$ em $P$; a dimensão de correlação fractal $D_2$ do conjunto $P$; a dimensão de imersão $E$ do conjunto $P$; o volume relativo $Vol(\varepsilon, shape)$ de uma consulta de formato *shape* de raio $\varepsilon$; e o volume relativo $Vol(\varepsilon,\square)$ de uma consulta de formato quadrado (ou seja, um hiper-cubo) de raio $\varepsilon$, o número médio de vizinhos de $P$ é dado por:

$$\overline{nb}(\varepsilon, shape) = \left( \frac{Vol(\varepsilon, shape)}{Vol(\varepsilon,\square)} \right)^{D_2/E} \times (N-1) \times 2^{D_2} \times \varepsilon^{D_2} \qquad (3.5)$$

A partir do Teorema 3.1 e das Equações 3.3 e 3.4, é possível estimar a seletividade para consultas por abrangência ($Sel_{range}(\varepsilon)$) e para consultas de junção espacial ($Sel_{join}(\varepsilon)$). Note-se que $Vol(\varepsilon,\square)$ corresponde ao volume de uma região do espaço definidos pela métrica $L_\infty$ de raio $\varepsilon$, ou seja, um hiper-cubo de lado $\varepsilon$. Como $L_\infty$ é a métrica mais abrangente, qualquer outra métrica definirá um volume mais restritivo, tornando a razão expressa na Equação 3.5 menor do que 1.

## 3.3. Modelos de custo para métodos de acesso a dados complexos

Como já foi dito anteriormente, a previsão do custo computacional de uma consulta é muito importante para otimizar a execução de consultas pelos SGBDs. Essa previsão de custo é realizada por meio de equações que permitem revisar o plano de execução de consultas, com o intuito de melhorar seu desempenho [Belussi_1998]. É importante lembrar que existem casos onde a busca seqüencial pode ser menos onerosa que o uso de um método de indexação.

O primeiro modelo de custo para consultas em dados complexos foi proposto em [Friedman_1977]. Esse modelo estima o número de acesso a nós folhas em *k-d-Trees* durante consultas de 1-vizinho mais próximo. Esse método foi estendido para estimar a quantidade de acessos a disco (nós) com a *R-Tree* em [Faloutsos_1987]. Trabalhos relacionados foram apresentados, tais como [Weber_1998] [Böhm_2001] [Aggarwal_2004], procurando incluir

consultas por abrangência e aos *k*-vizinhos mais próximos, diferentes funções de distância da família de Minkowsky e diferentes tamanhos e formatos (definidos pela função de distância) de nós. Todos esses trabalhos são voltados para dados espaciais que supõem a distribuição uniforme dos dados no espaço.

Ainda considerando dados espaciais, alguns trabalhos passaram a analisar conjuntos de dados reais com distribuições não uniformes, usando a teoria dos fractais. O primeiro foi apresentado em [Faloutsos_1994] e outros exemplos são: [Belussi_1998] e [Böhm_2000]. Esses trabalhos apresentam equações para estimativa de acessos a disco considerando, principalmente, a *R-Tree*. O trabalho apresentado em [Tao_2004] faz a estimativa de acessos a disco para consulta aos *k*-vizinhos mais próximos, considerando distribuições uniformes e não uniformes dos conjuntos de dados, e não utiliza a teoria de fractais. Esse trabalho usa a *R*-Tree*.

Definir equações de custo para métodos de acesso espaciais não é uma tarefa simples. Além do problema da modelagem da distribuição dos dados para permitir a estimativa de seletividade, conforme discutido na Seção 3.2, existe o problema da sobreposição dos nós, que aflige a grande maioria dos métodos de acesso dinâmicos. Uma prática para facilitar a análise, muito comum nos trabalhos de modelagem de custo de consultas por similaridade, consiste em pressupor a distribuição uniforme dos dados, o que muitas vezes não é correto e leva a erros significativos nas estimativas [Korn_2001].

Considerando modelos voltados para o espaço métrico e MAMs dinâmicos, os principais trabalhos são [Ciaccia_1998] [Traina Jr._1999] e [Traina Jr._2000a], todos voltados para a *M-Tree*. O problema aqui está, mais uma vez, com o modelo de distribuição dos dados. Por se tratar um modelo para dados métricos, não é possível assumir a distribuição uniforme, pois o espaço métrico não provê estruturas para tal suposição (não é possível definir volume, por exemplo). No entanto, novamente por ser mais simples, o trabalho apresentado em [Ciaccia_1998] pressupõe que a distribuição do valor das distâncias entre os objetos é uniforme e define as equações de custo. Já o trabalho de Traina et al. [Traina Jr._2000a] segue a linha de [Belussi_1998] e modela as equações de estimativa de custo considerando a distribuição fractal dos conjuntos de dados, refletindo melhor o comportamento dos conjuntos de dados reais.

Um ponto interessante nos trabalhos voltados para os métodos de acesso métricos é que os modelos consideram não só o número de acessos a disco, mas também o número de comparações de objetos (o número de cálculos da função de distância). Isso ocorre porque, em geral, esses métodos indexam a distância entre os objetos e procuram minimizar o número de comparações durante as consultas. Outra consideração freqüente é que a função de distância tende a ser razoavelmente complexa e que o custo computacional para executá-las não é muito diferente do custo de acessar o disco. Essas observações se tornam ainda mais evidentes ao se estudar os trabalhos da área, pois na maioria deles as principais medidas comparadas na seção de resultados são: o número de cálculos de distância, o número de acessos a disco e o tempo total de consultas.

## 3.4. Considerações finais

Como pode ser observado nos levantamentos de trabalhos sobre estimativa de seletividade para consultas por similaridade e previsão de custo de métodos de acesso para dados complexos, não existe consenso entre os pesquisadores sobre como caracterizar a distribuição de conjuntos de dados complexos. Em geral, os métodos são introduzidos e só em trabalhos complementares são apresentadas as equações de estimativa de custo. Trabalhos mais recentes indicam uma tendência de se utilizar os conceitos sobre dimensão intrínseca e dimensão fractal para a modelagem das equações, pois são mais adequados para caracterizar os conjuntos de dados reais e são compatíveis com conjuntos de dados métricos em geral.

Diante do que foi abordado ao longo deste capítulo, o modelo de custo proposto neste trabalho foi desenvolvido pressupondo que os conjuntos de dados são indexados por um MAM dinâmico, que os dados não estão uniformemente distribuídos e que os mesmos são dados complexos em geral, podendo ser vetoriais ou puramente métricos, desde que haja uma métrica bem definida para comparar os objetos. Para as estimativas de custo, este trabalho considera não só o número de acessos a disco, mas também o número de cálculos da função de distância métrica.

# 4. DESCRIÇÃO DO PROBLEMA

## 4.1. Introdução

Neste capítulo será descrito e analisado o problema que esta tese procura resolver, tal como colocado a seguir.

**Problema: Criação de um modelo de custo eficaz e eficiente para previsão de seletividade de consultas por similaridade em espaços métricos.**

Como visto no Capítulo 2, os critérios de comparação, baseados em igualdade ou em relações de ordem não são úteis em consultas por conteúdo a dados de domínios complexos, tais como imagens, vídeos, dados espaciais, séries temporais, seqüências de dados genéticos, entre outros. Como também já discutido no Capítulo 2, o critério mais adequado para consultas a dados complexos é o de similaridade. A avaliação da similaridade é feita utilizando funções de distância, que comparam dois dados complexos de um mesmo domínio e retornam um valor numérico que é menor quanto mais os dados são similares. Desse modo, consultas por similaridade envolvem: um conjunto de objetos e uma função de distância usada para medir a "distância", ou grau de dissimilaridade, entre os mesmos, sendo que as funções de distância mais úteis são as que apresentam as propriedades de uma métrica.

O desempenho de consultas por similaridade é afetado por dois fatores principais: o número de acessos a disco, para obtenção/atualização dos dados armazenados; e o número de comparações entre objetos efetuadas por cálculos de distância [Chávez_2001]. Devido ao alto custo computacional para recuperar e efetuar comparações entre dados complexos, consultas por similaridade são usualmente auxiliadas por estruturas de dados para espaços métricos. Usando a propriedade de desigualdade triangular do espaço métrico é possível descartar objetos, minimizando comparações e acessos a disco (ver Seção 2.2.1).

Como abordado na Seção 2.3, vários Métodos de Acesso Métricos (MAMs) têm sido propostos na literatura para melhorar o desempenho de consultas por similaridade em espaços métricos. Como MAMs distintos podem apresentar desempenhos diferentes frente a uma mesma consulta, o módulo otimizador de consultas dos SGBDs realiza estimativas de seletividade e de custo de consultas visando escolher o MAM mais apropriado.

Constata-se então a necessidade de um modelo de custo eficiente para que o módulo otimizador de consultas de um SGBD possa escolher a melhor opção de processamento de consultas por similaridade.

## 4.2. Delimitação do problema e hipótese para solução

A hipótese deste trabalho é a de que um modelo de custo efetivo para consultas por similaridade em espaços métricos deve considerar três fatores: o **tipo de consulta por similaridade**, a **dimensionalidade do conjunto de dados** e as **características do MAM**.

Nesta tese são considerados os dois tipos básicos de consultas por similaridade (Seção 2.2.2): consulta por abrangência (*Range Query – RQ*) e a consulta aos *k*-vizinhos mais próximos (*k-Nearest Neighbor Query – KNNQ*). Consultas aos *k*-vizinhos mais próximos podem ser consideradas como um caso especial de consultas por abrangência [Berchtold_1997], ou seja, uma consulta aos *k*-vizinhos mais próximos equivale a uma consulta por abrangência com raio de cobertura considerado inicialmente infinito.

Como abordado na Seção 3.2, a distinção entre a dimensão de imersão e a dimensão intrínseca tem sido muito utilizada no contexto de dados espaciais para avaliar quanto a distribuição de um conjunto diverge da distribuição uniforme [Korn_2001]. A importância da dimensão intrínseca é ainda maior no contexto de espaços métricos, onde usualmente não se pode aplicar a definição da dimensão de imersão, pois mesmo dados adimensionais são permitidos.

De acordo com [Schroeder_1991] e [Traina Jr._1999], a maioria dos conjuntos de dados reais não segue distribuições estatísticas tradicionais (*Gaussian* ou *Poisson*), comportando-se freqüentemente como fractais. A dimensão intrínseca do conjunto de dados pode então ser

obtida usando a dimensão de correlação fractal [Schroeder_1991] [Traina Jr._2000c]. Assim, consegue-se obter um valor de dimensão intrínseca para um conjunto de dados mesmo em um espaço adimensional.

Para um conjunto de objetos $S$ com dimensão intrínseca $D$ correspondendo à sua dimensão de correlação fractal, o número médio de distâncias menores que um raio de cobertura (raio de busca) $r_q$ segue uma lei de potências, ou seja, o número médio de vizinhos $nb(r_q)$ dentro de uma dada distância $r_q$ é proporcional a $r_q$ elevado a $D$ [Belussi_1995] [Traina Jr._1999]:

$$nb(r_q) \propto r_q^{\,D} \qquad (4.1)$$

Deve-se observar que a dimensão de correlação fractal $D$ não varia em relação ao tamanho do conjunto de dados, ou seja, $D$ tem o mesmo valor mesmo após inserções e remoções de dados do conjunto, o que é importante para um cálculo rápido de estimativa de seletividade.

De acordo com [Traina Jr._1999], dado um conjunto de objetos $S$ com dimensão intrínseca $D$ indexado usando uma árvore métrica com $N$ nós e com maior raio de cobertura $r$, e uma consulta por abrangência $RQ(s_q, r_q)$, onde $s_q$ é o objeto central de busca e $r_q$ é o raio de cobertura, o número estimado de acessos a disco $DA$ é dado por:

$$DA(r_q, r, N, D) \propto \frac{1}{r^D} \sum_{i=1}^{N} (r_i + r_q)^D \qquad (4.2)$$

Como pode ser notado, para o cálculo de $DA$ é necessário conhecer o raio de cobertura $r_i$ de cada nó da árvore métrica, o que demanda percorrer toda a árvore para estimar o número de acessos a disco para responder a consultas por abrangência. Então, faz-se necessário um método mais eficiente para estimar o custo de acessos a disco, o qual deve levar em consideração características do MAM a ser utilizado.

O modelo de custo proposto neste trabalho estima o número de acessos a disco e o número de cálculos de distância para consultas por abrangência e para consultas aos $k$-vizinhos mais próximos em espaços métricos. O modelo considera a consulta aos $k$-vizinhos mais próximos como um tipo especial de consulta por abrangência. A dimensão intrínseca do conjunto de dados é obtida usando a dimensão de correlação fractal, e a estimativa de seletividade considera que o número médio de vizinhos dentro de uma dada distância é obtido pela

Equação 4.1. A fim de se obter um método mais eficiente para estimar o custo de acessos a disco do que o apresentado pela Equação 4.2, o modelo de custo proposto leva em consideração características de métodos de acesso métricos dinâmicos baseados em árvore. Como plataforma de desenvolvimento e experimentos foi considerado o MAM dinâmico *Slim-Tree*, brevemente detalhado na Seção 2.4.

## 4.3. Caracterizando o problema

Em relação à otimização de consultas por conteúdo a dados complexos, embora alguns trabalhos iniciais tenham sido efetuados na direção de se obter estimativas de seletividade e modelos de custo para consultas espaciais [Belussi_1995] e em espaços métricos [Ciaccia_1998] [Traina Jr._1999] [Böhm_2000], não existe consenso entre os pesquisadores sobre como caracterizar a distribuição dos conjuntos de dados em tais espaços.

Como abordado anteriormente na Seção 3.3, os primeiros modelos de custo para consultas a dados complexos são voltados para dados espaciais, supondo a distribuição uniforme dos dados no espaço, o que mais tarde foi rejeitado [Christodoulakis_1984]. Ainda considerando dados espaciais, trabalhos posteriores passaram a analisar conjuntos de dados reais com distribuições não uniformes, usando a teoria dos fractais. Esses trabalhos apresentam equações para estimativa de acessos a disco para a *R-tree*.

Considerando espaços métricos, pouco tem sido feito com relação aos métodos de acesso métricos dinâmicos. O problema está, mais uma vez, no modelo de distribuição dos dados. O trabalho apresentado em [Ciaccia_1998] pressupõe que a distribuição do valor das distâncias entre os objetos é uniforme e define as equações de custo baseado nessa pressuposição, o que não é realista e superestima a dimensão real dos dados [Christodoulakis_1984]. Esta tese segue a linha de [Traina Jr._1999] e [Belussi_1998] e considera a distribuição fractal dos conjuntos de dados para as estimativas de seletividade, de modo a obter um modelo de custo mais eficiente e que reflita melhor o comportamento de conjuntos de dados reais.

Em espaços métricos, deve-se considerar também que um modelo de custo efetivo deve estimar não só o número de acessos a disco, mas também o número de comparações de objetos, ou seja, o número de execuções da função de distância métrica. Isso ocorre porque os

MAMs indexam a distância entre os objetos e procuram minimizar o número de comparações durante as consultas. Outra consideração freqüente é que a função de distância tende a ser razoavelmente complexa e que seu custo computacional não é muito diferente do custo de acessar o disco. Entretanto, trabalhos anteriores que consideram a distribuição não uniforme do valor das distâncias entre os objetos, estimam apenas o custo de acessos a disco. Esses trabalhos também consideram apenas consultas por abrangência. O modelo de custo proposto por este trabalho considera ambos, o número de acessos a disco e o número de cálculos de distância, tanto para consultas por abrangência quanto para consultas aos *k*-vizinhos mais próximos.

# 5. O MODELO DE CUSTO PROPOSTO

## 5.1. Introdução

A estimativa do custo computacional de uma consulta é muito importante para possibilitar a otimização da execução de consultas pelos SGBDs. Ela é realizada por meio de equações que permitem revisar o plano de execução de consultas, com o intuito de melhorar seu desempenho. Há casos em que uma busca seqüencial simples pode ser menos onerosa que o uso de um MAM, como por exemplo, quando o raio de cobertura de uma consulta por abrangência é relativamente grande, em relação ao diâmetro do conjunto de dados, isto é, mais que 10% da base de dados se qualifica para responder à consulta.

Este capítulo apresenta inicialmente um método para a estimativa de seletividade, tanto para consultas por abrangência quanto consultas aos *k*-vizinhos mais próximos. Considerando que para espaços métricos um modelo de custo deve estimar o número de acessos a disco (custo de I/O) e o número de cálculos da função de distância (custo de CPU), são apresentadas equações para a estimativa de custo de acessos a disco e do custo de cálculos de distância para consultas por abrangência (Seção 5.3) e para consultas aos *k*-vizinhos mais próximos (Seção 5.4). Para isso considera-se que os dados estão indexados por métodos de acesso métricos dinâmicos tradicionais em que todos os objetos estão nas folhas da árvore, como a *M-tree* e a *Slim-Tree*. Além disso, o método proposto utiliza também o conceito de *fat-factor* para as árvores métricas. Uma informação importante é a dimensão intrínseca do conjunto de dados, que é calculada usando a dimensão de correlação fractal.

Mesmo considerando a dimensão intrínseca do conjunto de dados e o *fat-factor* da árvore métrica, o uso apenas de informações globais sobre o conjunto de dados e o MAM não caracteriza adequadamente o conjunto de dados de modo a obter estimativas de custo precisas. Desse modo, a Seção 5.5 apresenta uma estratégia que emprega a distribuição dos dados no

local do centro da consulta para melhorar as estimativas realizadas. Isso é feito utilizando informações de consultas previamente executadas, ajustando-se o processo de estimativa para regiões distintas do conjunto de dados. A estimativa fica, então, sensível também à distribuição local dos dados.

No intuito de facilitar o acompanhamento do desenvolvimento das equações, a Tabela 1 sumariza a definição dos símbolos utilizados.

**Tabela 1:** Definição de símbolos.

| **Símbolos** | **Definições** |
|---|---|
| M | Espaço métrico. |
| S | Domínio ou universo de objetos válidos. |
| $S$ | Conjunto de objetos onde as consultas serão efetuadas, $S \subseteq$ S. |
| $\|S\|$ | Cardinalidade (número de elementos) de $S$. |
| $\|S_h\|$ | Número de objetos de $S$ armazenados em cada nível $h$ de uma árvore métrica. |
| $d(x,y)$ | Função de distância métrica entre os objetos $x,y \in S$. |
| $RQ(s_q, r_q)$ | Consulta por abrangência (*RQ – Range Query*). |
| $KNNQ(s_q, k)$ | Consulta aos $k$-vizinhos mais próximos (*KNNQ – k-Nearest Neighbor Query*) |
| $s_q \in$ S | Objeto central de uma consulta. |
| $s_s \in$ S | Objeto central da consulta previamente armazenada. |
| $r_q$ | Raio de cobertura de uma consulta. |
| $r_s$ | Raio de cobertura de uma consulta previamente armazenada. |
| $r$ | Maior raio de cobertura de uma árvore métrica. |
| $r_h$ | Raio de cobertura médio de um nó em um nível $h$ de uma árvore métrica. |
| $r_{levelh}$ | Raio de cobertura médio de um nível $h$ de uma árvore métrica. |
| $r_{leaf}$ | Raio de cobertura médio de um nó folha de uma árvore métrica. |
| $N$ | Número total de nós de uma árvore métrica. |
| $N_h$ | Número estimado de nós em cada nível $h$ de uma árvore métrica. |
| $N_{leaves}$ | Número estimado de nós folhas de uma árvore métrica. |
| $H$ | Número total de níveis de uma árvore métrica. |
| $D$ | Dimensão intrínseca, correspondendo à dimensão de correlação fractal. |
| $nb(r_q)$ | Número médio de vizinhos dentro de uma dada distância $r_q$. |
| $fat(T)$ | Fator de sobreposição – *fat-factor* – de uma árvore métrica $T$. |
| $C_{eff}$ | Capacidade efetiva de um nó de uma árvore métrica. |
| $C$ | Capacidade máxima de um nó de uma árvore métrica. |
| $u$ | Utilização média de um nó de uma árvore métrica. |
| $k$ | Número de objetos recuperados por uma consulta. |
| $k_s$ | Número de objetos recuperados por uma consulta previamente armazenada. |
| $Sel_{RQ}(r_q, r, D)$ | Estimativa de seletividade de uma consulta por abrangência. |
| $Sel_{KNNQ}(k)$ | Estimativa de seletividade de uma consulta aos $k$-vizinhos mais próximos. |
| $DA_{optimal}(r_q, r, H, D)$ | Estimativa do número de acessos a disco para uma árvore métrica ótima, considerando consultas por abrangência com raio de cobertura $r_q$. |
| $DA_{optimal}(k, r, H, D)$ | Estimativa do número de acessos a disco para uma árvore métrica ótima, considerando consultas aos $k$-vizinhos mais próximos que recupera $k$ objetos. |
| $DA_g(r_q, r, H, D)$ | Estimativa do número de acessos a disco para consultas por abrangência com raio de cobertura $r_q$, considerando a sobreposição de nós da árvore métrica. |
| $DA_g(k, r, H, D)$ | Estimativa do número de acessos a disco para consultas aos $k$-vizinhos mais próximos que recupera $k$ objetos, considerando a sobreposição de nós da árvore métrica. |
| $DA_q$ | Número de acessos a disco de uma consulta $q$. |
| $DA_s$ | Número de acessos a disco de uma consulta previamente armazenada. |

| | |
|---|---|
| $DA_{RQ}$ | Estimativa final do número de acessos a disco para uma consulta por abrangência considerando $DA_g(r_q, r, H, D)$ e $DA_s$. |
| $DA_{KNNQ}$ | Estimativa final do número de acessos a disco para uma consulta aos $k$-vizinhos mais próximos considerando $DA_g(k, r, H, D)$ e $DA_s$. |
| $DC_{optinal}(r_q, r, H, D)$ | Estimativa do número de cálculos de distância para uma árvore métrica ótima, considerando consultas por abrangência com raio de cobertura $r_q$. |
| $DC_{optinal}(k, r, H, D)$ | Estimativa do número de cálculos de distância para uma árvore métrica ótima, considerando consultas aos $k$-vizinhos mais próximos que recupera $k$ objetos. |
| $DC_g(r_q, r, H, D)$ | Estimativa do número de cálculos de distância para consultas por abrangência com raio de cobertura $r_q$, considerando a sobreposição de nós da árvore métrica. |
| $DC_g(k, r, H, D)$ | Estimativa do número de cálculos de distância para consultas aos $k$-vizinhos mais próximos que recupera $k$ objetos, considerando a sobreposição de nós da árvore métrica. |
| $DC_q$ | Número de cálculos de distância de uma consulta $q$. |
| $DC_s$ | Número de cálculos de distância de uma consulta previamente armazenada. |
| $DC_{RQ}$ | Estimativa final do número de cálculos de distância para uma consulta por abrangência considerando $DC_g(r_q, r, H, D)$ e $DC_s$. |
| $DC_{KNNQ}$ | Estimativa final do número de cálculos de distância para uma consulta aos $k$-vizinhos mais próximos considerando $DC_g(k, r, H, D)$ e $DC_s$. |
| $K_p$ | Constante de proporcionalidade. |
| $K_{DA}$ | Constante de proporcionalidade para acessos a disco. |
| $K_{DC}$ | Constante de proporcionalidade para cálculos de distância. |
| $p_e$ | Porcentagem da estimativa de acessos a disco e de cálculos de distância. |
| $p_s$ | Porcentagem de acessos a disco e de cálculos de distância de uma consulta previamente armazenada. |

## 5.2. Estimativa de seletividade

Estimar a seletividade de consultas significa estimar a proporção de objetos que farão parte do conjunto-resposta da mesma, em relação ao total de objetos armazenados.

Considerando o espaço métrico $\mathbb{M} = (\mathbb{S}, d)$, dado um conjunto de objetos $S \subseteq \mathbb{S}$ e uma **consulta por abrangência** definida por $RQ(s_q, r_q)$, onde $s_q \in S$ e $r_q$ são, respectivamente, o objeto central de busca e o raio de cobertura, estimar a seletividade de $RQ(s_q, r_q)$ aplicada em $S$ significa estimar a proporção de objetos de $S$ que fazem interseção com a região de busca definida por $RQ(s_q, r_q)$ em relação a $|S|$ (total de objetos de $S$), ou seja, a estimativa de seletividade $Sel_{RQ}(r_q)$ é dada por:

$$Sel_{RQ}(r_q) = \frac{número\ médio\ de\ vizinhos\ dentro\ da\ distância\ r_q}{|S|} \qquad (5.1)$$

A Equação 4.1 determina que o número médio de vizinhos $nb(r_q)$ dentro de uma dada distância $r_q$ é proporcional a $r_q$ elevado a $D$. Considerando uma constante de

proporcionalidade $K_p$, tem-se que o número médio de vizinhos $nb(r_q)$ dentro de uma dada distância $r_q$ é dado por:

$$nb(r_q) = K_p \cdot r_q^{\,D} \tag{5.2}$$

Considerando que $|S| = nb(r) = K_p \cdot r^D$, onde $r$ é o maior raio de cobertura possível. Então, a Equação 5.1 pode ser reescrita da seguinte maneira:

$$Sel_{RQ}(r_q, r, D) = \left(\frac{r_q}{r}\right)^D \tag{5.3}$$

Ainda considerando o espaço métrico $\mathbb{M} = (\mathbb{S}, d)$, dado um conjunto de objetos $S \subseteq \mathbb{S}$ e uma **consulta aos *k*-vizinhos mais próximos** definida por $KNNQ(s_q, k)$, onde $s_q \in S$ é o objeto central de busca e $k$ é o número de objetos mais semelhantes a $s_q$ a serem recuperados, estimar a seletividade de $KNNQ(s_q, k)$ aplicada a $S$ significa estimar a proporção dos $k$ objetos de $S$ a serem recuperados em relação a $|S|$, ou seja, a estimativa de seletividade $Sel_{KNNQ}(k)$ é dada por:

$$Sel_{KNNQ}(k) = \frac{k}{|S|} \tag{5.4}$$

## 5.3. Modelo de custo para consultas por abrangência

Como abordado na Seção 4.2, a Equação 4.2 estima o número de acessos a disco de consultas por abrangência baseada no raio de cobertura $r_i$ de cada nó da árvore métrica. Porém, esse método não é eficiente, pois demanda percorrer toda a árvore.

Assim, esta seção propõe desenvolver um método para estimar o número de acessos a disco provável de ser necessário para responder a uma consulta sem a necessidade de percorrer toda a árvore métrica. O modelo de custo proposto considera também o número de cálculos de distância de uma consulta por abrangência. Com esse objetivo, considera-se o uso de um MAM dinâmico, com as mesmas características da *Slim-Tree* (Seção 2.4) como plataforma de desenvolvimento.

### 5.3.1. Custo de acessos a disco

Para a construção do modelo de custo para acessos a disco, é importante colocar a seguinte definição.

> **Definição 5.1:** A capacidade efetiva $C_{eff}$ dos nós de uma árvore métrica equivale ao número médio de objetos armazenados em um nó não-raiz da árvore. Então, considerando que cada nó da árvore tem capacidade máxima para armazenar $C$ objetos e tem uma média de utilização de $u\%$, a capacidade efetiva $C_{eff}$ é:

$$C_{eff} = \frac{C \cdot u}{100} \tag{5.5}$$

Considerando-se que uma árvore de altura $H$ tem $H$ níveis, onde a raiz corresponde ao nível $h=0$ e as folhas ao nível $h = H$-1, chega-se ao seguinte lema.

> **Lema 5.1:** O número de objetos de $S$ que podem ser armazenados em cada nível da árvore $|S_h|$ é dado por:

$$|S_h| = C_{eff}{}^{h+1},\ h = 0,1,...,H-1 \tag{5.6}$$

**Prova:** Assumindo uma árvore com $C_{eff}$ máxima em todos os nós, tem-se a Equação 5.6. □

Nas árvores métricas tradicionais, todos os $|S|$ objetos são armazenados nas folhas. Assim, considerando que a altura $h = H - 1$ para as folhas, $|S_{H-1}| = |S_{folhas}| = |S|$. Logo, usando o Lema 5.1, tem-se que:

$$|S_{H-1}| = C_{eff}{}^{(H-1)+1} \Rightarrow |S| = C_{eff}{}^{H} \Rightarrow \sqrt[H]{|S|} = C_{eff} \Rightarrow C_{eff} = |S|^{\frac{1}{H}} \tag{5.7}$$

Considerando-se uma árvore métrica ótima, isto é, uma árvore com características ideais (o número de objetos em cada nó é aproximadamente o mesmo, cada nó cobre os objetos mais próximos e não existe sobreposição entre nós), têm-se os dois lemas seguintes.

**Lema 5.2:** Considerando que a dimensão intrínseca do conjunto de objetos *S* é obtida pela dimensão de correlação fractal *D*, o raio de cobertura médio $r_h$ de um nó em um nível *h* é dado por:

$$r_h = \sqrt[D]{|S|^{\frac{-h}{H}}} \tag{5.8}$$

**Prova:** Inicialmente calcula-se o raio de cobertura das folhas da árvore métrica. Dado que os objetos de *S* estão armazenados nas folhas da árvore, o número estimado de nós folhas pode ser expresso, em função da capacidade efetiva $C_{eff}$ dos nós, como:

$$N_{leaves} = \frac{|S|}{C_{eff}} \tag{5.9}$$

Os objetos de *S* são agrupados nas folhas da árvore de acordo com um raio de cobertura médio $r_{leaf}$, onde cada grupo de objetos consiste em um nó folha. O número estimado de grupos de objetos (nós folhas) $N_{leaves}$ com raio de cobertura médio $r_{leaf}$, necessários para cobrir |S| objetos de um conjunto de objetos *S* com dimensão intrínseca igual à dimensão de correlação fractal *D*, é dado por [Schroeder_1991]:

$$N_{leaves} = \frac{1}{r_{leaf}^{\ D}} \tag{5.10}$$

Combinando-se as Equações 5.7, 5.9 e 5.10, tem-se que o raio de cobertura médio de um nó folha é dado por:

$$\frac{|S|}{C_{eff}} = \frac{1}{r_{leaf}^{\ D}} \Rightarrow r_{leaf}^{\ D} = \frac{C_{eff}}{|S|} \overset{(5.7)}{\Rightarrow} r_{leaf}^{\ D} = \frac{|S|^{\frac{1}{H}}}{|S|} \Rightarrow r_{leaf} = \sqrt[D]{|S|^{\frac{-(H-1)}{H}}} \tag{5.11}$$

Assumindo uma árvore com $C_{eff}$ máxima em todos os níveis, tem-se que o número estimado de nós em cada nível *h* é dado por:

$$N_h = \frac{|S_h|}{C_{eff}} \tag{5.12}$$

O mesmo processo realizado para obter a Equação 5.11 pode ser usado para estimar o raio de cobertura médio $r_h$ dos nós de cada nível $h$ da árvore. Assim, considerando que as folhas estão no nível $h = H-1$, basta substituir $H-1$ por $h$ na Equação 5.11 para obter a Equação 5.8.

□

**Lema 5.3:** Para uma árvore métrica ótima com o maior raio de cobertura igual a $r$ e $H$ níveis, armazenando um conjunto de objetos $S$ com dimensão intrínseca $D$, o número estimado de acessos a disco necessário para responder a consultas por abrangência com raio de cobertura $r_q$ é dado por:

$$DA_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \qquad (5.13)$$

**Prova:** A Equação 4.2 possibilita realizar a estimativa de custo de acessos a disco para uma consulta por abrangência em uma árvore métrica, desde que se conheça o raio de cobertura $r_i$ de cada nó $i$ da árvore, visando obter a sumarização dos raios de cobertura $r_i$ dos $N$ nós. Entretanto, conhecer os raios de cobertura $r_i$ de todos os $N$ nós de uma árvore métrica demanda percorrer a árvore toda, o que não é eficiente. Utilizando as Equações 5.8 e 5.12 é possível obter o valor do raio de cobertura médio dos nós de cada nível de uma árvore métrica. Ou seja, a Equação 5.8 determina o raio de cobertura médio $r_h$ de um nó em um nível $h$ da árvore métrica e a Equação 5.12 permite estimar o número de nós $N_h$ em cada nível da árvore. Então, o raio de cobertura médio para cada nível $h$, $r_{levelh}$, de uma árvore métrica é obtido por:

$$r_{levelh} = N_h \cdot r_h \qquad (5.14)$$

A sumarização dos raios de cobertura $r_i$ dos $N$ nós de uma árvore métrica pode, então, ser estimada a partir do raio de cobertura médio de cada nível $h$ da árvore da seguinte maneira:

$$\sum_{i=1}^{N} r_i = \sum_{h=0}^{H-1} N_h \cdot r_h \qquad (5.15)$$

Usando o resultado obtido em 5.15, a Equação 4.2 pode ser reescrita como:

$$DA(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} N_h (r_h + r_q)^D \qquad (5.16)$$

Combinando-se a Equação 5.16 com os resultados obtidos em 5.6, 5.7, 5.8 e 5.12 tem-se a Equação 5.13, da seguinte maneira:

$$DA(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} N_h (r_h + r_q)^D \overset{(5.8)}{\Rightarrow} DA_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} N_h \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \overset{(5.12)}{\Rightarrow}$$

$$DA_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} \frac{|S_h|}{C_{eff}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \overset{(5.6)}{\Rightarrow} DA_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} C_{eff}{}^h \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \overset{(5.7)}{\Rightarrow}$$

$$DA_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D$$

□

A Equação 5.13 considera uma árvore métrica ótima. Entretanto, para uma árvore que não tem características ótimas, o número estimado de acessos a disco será maior, isto é:

$$DA(r_q,\ r,\ H,\ D) > DA_{optimal}(r_q,\ r,\ H,\ D)$$

Um aspecto importante a ser considerado para uma estimativa de custo de acessos a disco mais precisa é o problema da sobreposição dos nós dos MAMs. Uma medida importante obtida de uma árvore métrica, como a *Slim-Tree*, é o *fat-factor* (Seção 2.4), o qual quantifica a sobreposição de nós de uma árvore métrica. A sobreposição entre nós da árvore é o que faz com que mais subárvores tenham que ser percorridas durante o processo de consulta aos dados. Assim, uma árvore com fator de sobreposição (*fat-factor*) alto demandará processar um número correspondentemente alto das suas subárvores. O lema a seguir considera esse fato.

> **Lema 5.4:** O número de acessos a disco para uma consulta por abrangência em uma árvore métrica, considerando parâmetros globais do conjunto de dados é obtido por:
>
> $$DA_g(r_q, r, H, D) = DA_{optimal}(r_q, r, H, D)(1 + fat(T)) + K_{DA} \qquad (5.17)$$

**Prova:** Usando o *fat-factor* (calculado da maneira mostrada na Seção 2.4) tem-se que para uma árvore métrica *T* com maior raio de cobertura *r* e *H* níveis, armazenando um conjunto de

objetos $S$ com dimensão intrínseca $D$, o número estimado de acessos a disco necessário para responder a consultas por abrangência com raio de cobertura $r_q$ é dado por:

$$DA_g(r_q, r, H, D) \propto DA_{optimal}(r_q, r, H, D)(1 + fat(T)) \qquad (5.18)$$

A Equação 5.18 estima o número de acessos a disco, exceto pela constante de proporcionalidade $K_{DA}$, a ser calculada a partir do número de acessos a disco de uma consulta previamente executada $DA_q$. Considerando que esse número pode ser calculado a partir de seu custo estimado somado à constante de proporcionalidade $K_{DA}$, tem-se que $K_{DA}$ é dada por:

$$DA_q = DA_g(r_q, r, H, D) + K_{DA} \Rightarrow K_{DA} = DA_q - DA_g(r_q, r, H, D) \qquad (5.19)$$

Assim, somando a constante de proporcionalidade $K_{DA}$ à Equação 5.18, obtém-se a Equação 5.17.

□

### 5.3.2. Custo de cálculos de distância

Esta seção mostra como estimar o custo com os cálculos de distância em uma consulta por abrangência, considerando um MAM dinâmico com características semelhantes a da *Slim-Tree*.

> **Lema 5.5:** O número estimado de cálculos de distância necessários para responder uma consulta por abrangência com raio de cobertura $r_q$, utilizando uma árvore métrica ótima com maior raio de cobertura $r$ e $H$ níveis, armazenando um conjunto de objetos $S$ com dimensão intrínseca $D$, é dado por:

$$DC_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h+1}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \qquad (5.20)$$

**Prova:** Parte-se do princípio que o custo estimado para os cálculos de distância pode ser obtido a partir do custo estimado de acessos a disco. O custo de acessos a disco resulta no número de nós acessados para responder a uma consulta, cujo resultado é obtido pela Equação 5.13. Cada nó armazena um número de objetos, e para todos eles será necessário realizar um cálculo de distância. A capacidade efetiva de cada nó é dada por $C_{eff}$, obtida pela Equação 5.7

em termos do total de objetos do conjunto *S* e do total de níveis *H* da árvore. Desse modo, usando a Equação 5.13 para realizar a estimativa de nós acessados em cada nível *h* da árvore, pode-se multiplicar o número de nós acessados em cada nível *h* por $C_{eff}$ para estimar o número de objetos acessados em cada nível *h* da árvore. Formalmente tem-se que:

$$DC_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h}{H}} \cdot C_{eff} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D \overset{(5.7)}{\Rightarrow} DC_{optimal}(r_q, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h+1}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + r_q \right)^D$$

□

Da mesma maneira como foi feito para a estimativa do número de acessos a disco, um aspecto importante a ser considerado para uma estimativa de número de cálculos de distância mais precisa é a sobreposição de nós dos MAMs, calculada usando o *fat-factor*.

**Lema 5.6:** A estimativa global do número de cálculos de distância para uma consulta por abrangência é dada por:

$$DC_g(r_q, r, H, D) = DC_{optimal}(r_q, r, H, D)(1 + fat(T)) + K_{DC} \qquad (5.21)$$

**Prova:** Usando o *fat-factor* tem-se que, para uma árvore métrica *T* com o maior raio de cobertura *r* e *H* níveis, armazenando um conjunto de objetos *S* com dimensão intrínseca *D*, o número estimado de cálculos de distância necessário para responder a consultas por abrangência com raio de cobertura $r_q$ é dado por:

$$DC_g(r_q, r, H, D) \propto DC_{optimal}( (r_q, r, H, D)(1 + fat(T)) \qquad (5.22)$$

A Equação 5.22 estima o número de cálculos de distância, exceto pela constante de proporcionalidade $K_{DC}$. Considerando que o número de cálculos de distância de uma consulta previamente executada $DC_q$ pode ser obtido a partir de seu custo estimado somado à constante de proporcionalidade $K_{DC}$, tem-se que $K_{DC}$ é dada por:

$$DC_q = DC_g(r_q, r, H, D) + K_{DC} \Rightarrow K_{DC} = DC_q - DC_g(r_q, r, H, D) \qquad (5.23)$$

Assim, somando a constante de proporcionalidade $K_{DC}$ à Equação 5.22, obtém-se a Equação 5.21.

□

## 5.4. Modelo de custo para consultas aos *k*-vizinhos mais próximos

Como abordado na Seção 4.2, consultas aos *k*-vizinhos mais próximos podem ser consideradas como um caso especial de consultas por abrangência, ou seja, uma consulta aos *k*-vizinhos mais próximos equivale a uma consulta por abrangência com raio de cobertura $r_q$ a ser determinado. O problema está em conseguir estimar $r_q$, pois os algoritmos que implementam consultas aos *k*-vizinhos mais próximos baseiam-se em iterações que vão ajustando $r_q$ a medida em que os *k* objetos são recuperados. Esse tipo de ajuste não é eficaz para uma estimativa de custo.

Assim, para estimar o número de acessos a disco e de cálculos de distância de uma consulta aos *k*-vizinhos mais próximos, este trabalho propõe um método para estimar o raio de cobertura $r_q$, a ser utilizado nas equações de custo propostas na seção 5.3 para consultas por abrangência. Para tanto, tem-se o seguinte lema:

> **Lema 5.7:** Para uma consulta aos *k*-vizinhos mais próximos, o raio de cobertura estimado $r_q$ é dado por:

$$r_q = \sqrt[D]{\frac{k \cdot r^D}{|S|}} \tag{5.24}$$

**Prova:** Considerando-se que para uma consulta aos *k*-vizinhos mais próximos o número de vizinhos dentro de uma dada distância $r_q$ é dado por *k*, a Equação 5.2 pode ser reescrita como:

$$nb(r_q) = K_p \cdot {r_q}^D \Rightarrow k = K_p \cdot {r_q}^D \Rightarrow {r_q}^D = \frac{k}{K_p} \Rightarrow r_q = \sqrt[D]{\frac{k}{K_p}} \tag{5.25}$$

A constante de proporcionalidade $K_p$ pode ser obtida pela Equação 5.2, considerando o maior raio de cobertura *r*, sendo que o número de vizinhos dentro de *r* é dado por |*S*|:

$$nb(r) = K_p \cdot r^D \Rightarrow |S| = K_p \cdot r^D \Rightarrow K_p = \frac{|S|}{r^D} \tag{5.26}$$

Combinando-se a Equação 5.25 com o resultado obtido pela Equação 5.26 tem-se a Equação 5.24, da seguinte maneira:

$$r_q = \sqrt[D]{\frac{k}{K_p}} \overset{(5.26)}{\Rightarrow} r_q = \sqrt[D]{\frac{k}{\frac{|S|}{r^D}}} \Rightarrow r_q = \sqrt[D]{\frac{k \cdot r^D}{|S|}}$$

□

### 5.4.1. Custo de acessos a disco

Considerando a estimativa do número de acessos a disco para consultas aos *k*-vizinhos mais próximos, tem-se o seguinte lema:

> **Lema 5.8:** Para uma árvore métrica ótima com maior raio de cobertura *r* e *H* níveis, armazenando um conjunto de objetos *S* com dimensão intrínseca *D*, o número estimado de acessos a disco necessário para responder a consultas aos *k*-vizinhos mais próximos é dado por:

$$DA_{optimal}(k, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + \sqrt[D]{\frac{k \cdot r^D}{|S|}} \right)^D \qquad (5.27)$$

**Prova:** A Equação 5.13 possibilita realizar a estimativa do custo de acessos a disco para consultas por abrangência com raio $r_q$. Em uma consulta aos *k*-vizinhos mais próximos, $r_q$ é obtido pela Equação 5.24. Assim, substituindo-se $r_q$ da Equação 5.13 pelo resultado obtido na Equação 5.24, tem-se a Equação 5.27.

□

A Equação 5.27 considera uma árvore métrica ótima. Como abordado anteriormente, para árvores que não tem características ótimas, o número estimado de acessos a disco será maior. Então, considerando novamente o problema da sobreposição dos nós dos MAMs e usando o *fat-factor* tem-se o seguinte lema:

**Lema 5.9:** O número de acessos a disco para uma consulta aos *k*-vizinhos mais próximos em uma árvore métrica, considerando parâmetros globais do conjunto de dados é dado por:

$$DA_g(k, r, H, D) = DA_{optimal}(k, r, H, D)(1 + fat(T)) + K_{DA} \quad (5.28)$$

**Prova:** Usando o *fat-factor* tem-se que para uma árvore métrica $T$ com maior raio de cobertura $r$ e $H$ níveis, armazenando um conjunto de objetos $S$ com dimensão intrínseca $D$, o número estimado de acessos a disco necessário para responder a consultas aos $k$-vizinhos mais próximos que recupera $k$ objetos é dado por:

$$DA_g(k, r, H, D) \propto DA_{optimal}(k, r, H, D)(1 + fat(T)) \quad (5.29)$$

A Equação 5.29 estima o número de acessos a disco, exceto pela constante de proporcionalidade $K_{DA}$, a ser calculada a partir do número de acessos a disco de uma consulta previamente executada $DA_q$. Considerando que esse número pode ser calculado a partir de seu custo estimado somado à constante de proporcionalidade $K_{DA}$, tem-se que $K_{DA}$ é dada por:

$$DA_q = DA_g(r_q, r, H, D) + K_{DA} \Rightarrow K_{DA} = DA_q - DA_g(r_q, r, H, D) \quad (5.30)$$

Assim, somando a constante de proporcionalidade $K_{DA}$ à Equação 5.29, obtém-se a Equação 5.28.

□

### 5.4.2. Custo de cálculos de distância

Considerando-se a estimativa do número de cálculos de distância para consultas aos *k*-vizinhos mais próximos, tem-se o seguinte lema:

**Lema 5.10:** O número estimado de cálculos de distância requeridos em uma consulta aos *k*-vizinhos mais próximos que recupera *k* objetos, utilizando uma árvore métrica ótima com maior raio de cobertura *r* e *H* níveis, armazenando um conjunto de objetos *S* com dimensão intrínseca *D*, é dado por:

$$DC_{optimal}(k, r, H, D) \propto \frac{1}{r^D} \sum_{h=0}^{H-1} |S|^{\frac{h+1}{H}} \left( \sqrt[D]{|S|^{\frac{-h}{H}}} + \sqrt[D]{\frac{k \cdot r^D}{|S|}} \right)^D \quad (5.31)$$

**Prova:** A Equação 5.20 possibilita realizar a estimativa de custo de cálculos de distância para consultas por abrangência com raio $r_q$. Em uma consulta aos *k*-vizinhos mais próximos, $r_q$ é, então, obtido pela Equação 5.24. Assim, substituindo-se $r_q$ da Equação 5.20 pelo resultado obtido em 5.24, tem-se a Equação 5.31.

□

Assim como foi feito para a estimativa do número de acessos a disco, para uma estimativa de número de cálculos de distância mais precisa considera-se a sobreposição de nós dos MAMs, calculada usando o *fat-factor*.

**Lema 5.11:** A estimativa global do número de cálculos de distância para uma consulta aos *k*-vizinhos mais próximos é dada por:

$$DC_g(k, r, H, D) = DC_{optimal}(k, r, H, D)(1 + fat(T)) + K_{DC} \quad (5.32)$$

**Prova:** Usando o *fat-factor*, tem-se que, para uma árvore métrica *T* com o maior raio de cobertura *r* e *H* níveis, armazenando um conjunto de objetos *S* com dimensão intrínseca *D*, o número estimado de cálculos de distância necessário para responder a consultas aos *k*-vizinhos mais próximos que recupera *k* objetos $r_q$ é dado por:

$$DC_g(r_q, r, H, D) \propto DC_{optimal}(r_q, r, H, D)(1 + fat(T)) \quad (5.33)$$

A Equação 5.33 estima o número de cálculos de distância, exceto por uma constante de proporcionalidade $K_{DC}$. Considerando que o número de cálculos de distância efetuados por uma consulta previamente executada $DC_q$ pode ser obtido a partir de seu custo estimado somado à constante de proporcionalidade $K_{DC}$, tem-se que $K_{DC}$ é dada por:

$$DC_q = DC_g(r_q, r, H, D) + K_{DC} \Rightarrow K_{DC} = DC_q - DC_g(r_q, r, H, D) \quad (5.34)$$

Assim, somando a constante de proporcionalidade $K_{DC}$ à Equação 5.33, obtém-se a Equação 5.32.

□

## 5.5. Aprimoramento das estimativas de custo com dados locais

As equações de custo propostas até aqui baseiam-se apenas em parâmetros globais do conjunto de dados, o que proporciona a obtenção de uma estimativa de custo inicial de maneira rápida. Entretanto, essas estimativas iniciais muitas vezes não conseguem identificar variações locais da distribuição dos dados. Desse modo, nesta seção é proposto um outro método para estimativas de custos, estas considerando esse aspecto, o que aprimora as estimativas mesmo em conjuntos de dados que apresentam significativas variações locais na distribuição dos dados.

Módulos otimizadores de consultas podem armazenar informações sobre consultas previamente executadas para auxiliar as estimativas de custo de novas consultas. Partindo desse princípio, é possível melhorar as estimativas de custo usando, além das equações para estimativas de custo propostas anteriormente, os custos reais de algumas consultas previamente executadas.

Tanto para consultas por abrangência como para consultas aos *k*-vizinhos mais próximos, o valor final das estimativas é calculado usando: uma porcentagem $p_e$ do valor dos custos estimados de acessos a disco $DA_g$, obtido pela Equação 5.17 para consultas por abrangência e pela Equação 5.28 para consultas aos *k*-vizinhos mais próximos, e do valor de cálculos de distância $DC_g$, dado pela Equação 5.21 para consultas por abrangência e pela Equação 5.32 para consultas aos *k*-vizinhos mais próximos; e uma porcentagem $p_s$ do valor real do número de acessos a disco $DA_s$ e cálculos de distância $DC_s$ de uma consulta previamente armazenada, tanto por abrangência quanto aos *k*-vizinhos mais próximos. Assim, o valor final das estimativas de acesso a disco *DA* e cálculos de distância *DC* é obtido pelas equações a seguir.

Para consultas por abrangência:

$$DA_{RQ} = DA_g\left(r_q, r, H, D\right)\cdot p_e + DA_s \cdot p_s \tag{5.35}$$

$$DC_{RQ} = DC_g\left(r_q, r, H, D\right)\cdot p_e + DC_s \cdot p_s \tag{5.36}$$

Para consultas aos *k*-vizinhos mais próximos:

$$DA_{KNNQ} = DA_g(k, r, H, D) \cdot p_e + DA_s \cdot p_s \quad (5.37)$$

$$DC_{KNNQ} = DC_g(k, r, H, D) \cdot p_e + DC_s \cdot p_s \quad (5.38)$$

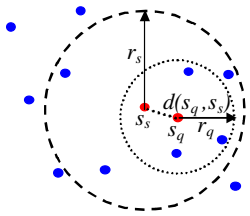
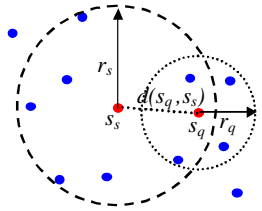
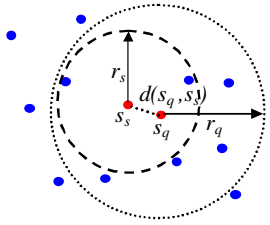
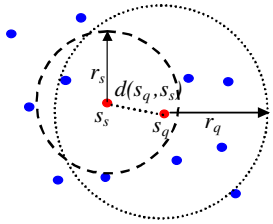
Usando essa abordagem, faz-se necessário armazenar os valores reais de custo de execução de consultas. Tanto para consultas por abrangência quanto para consultas aos *k*-vizinhos mais próximos, as informações necessárias a serem armazenadas para as consultas previamente executadas são: o centro da consulta armazenada $s_s$; o raio de cobertura da consulta armazenada $r_s$; o número de objetos recuperados pela consulta armazenada $k_s$; o número de acessos a disco e cálculos de distância da consulta armazenada $DA_s$ e $DC_s$, respectivamente. Com essas informações, os dados de uma mesma consulta armazenada podem ser utilizados para melhorar tanto consultas por abrangência quanto aos *k*-vizinhos mais próximos.

Quando uma nova consulta a ser executada ocorre perto de uma consulta previamente armazenada, então os valores reais de custo armazenados são considerados usando-se as Equações 5.35 e 5.36 para consultas por abrangência, ou as equações 5.37 e 5.38 para consultas aos *k*-vizinhos mais próximos. Porém, quando essa proximidade não acontece, apenas as estimativas globais de custo são consideradas usando-se as Equações 5.17 e 5.21 ou as Equações 5.28 e 5.32, dependendo do tipo da consulta.

A consulta previamente armazenada a ser considerada nos cálculos deve ter o centro próximo ao da nova consulta. Neste trabalho considera-se que "próximo" é quando a distância entre o centro da nova consulta $s_q \in S$ e o centro da consulta armazenada $s_s \in S$, $d(s_q, s_s)$ é menor ou igual ao raio de cobertura da consulta armazenada $r_s$.

Uma vez encontrada uma consulta por abrangência previamente armazenada com centro próximo, o cálculo de $p_e$ e $p_s$ depende de dois fatores: da proporção entre o número médio de vizinhos dentro de uma dada distância $r_s$ e o número médio de vizinhos dentro de uma dada distância $r_q$, dados pela Equação 5.2, e da proporção de $d(s_q, s_s)$ em relação à $r_s$, de acordo com algumas situações mostradas na Tabela 2.

**Tabela 2:** Cálculo das porcentagens de custos estimados $p_e$ e de custos armazenados $p_s$ para consultas por abrangência.

<table>
<tr><td colspan="2">1. O raio da consulta corrente é menor ou igual ao raio da consulta armazenada: $r_q \leq r_s$</td></tr>
<tr><td rowspan="2"> 
</td><td>1.1. A consulta corrente está totalmente contida na consulta armazenada: $d(s_q, s_s) + r_q \leq r_s$</td></tr>
<tr><td>$p_s = \frac{{r_q}^D}{{r_s}^D}$<br>$p_e = 1 - p_s$</td></tr>
<tr><td rowspan="2"> 
</td><td>1.2. A consulta corrente não está totalmente contida na consulta armazenada: $d(s_q, s_s) + r_q > r_s$</td></tr>
<tr><td>$p_s = \frac{{r_q}^D}{{r_s}^D} \cdot \frac{d(s_q, s_s)}{r_s}$<br>$p_e = 1 - p_s$</td></tr>
<tr><td colspan="2">2. O raio da consulta corrente é maior que o raio da consulta armazenada: $r_q > r_s$</td></tr>
<tr><td rowspan="2"> 
</td><td>2.1. A consulta armazenada está totalmente contida na consulta corrente: $d(s_q, s_s) + r_s \leq r_q$</td></tr>
<tr><td>$p_s = \frac{{r_s}^D}{{r_q}^D}$<br>$p_e = 1 - p_s$</td></tr>
<tr><td rowspan="2"> 
</td><td>2.2. A consulta armazenada não está totalmente contida na consulta corrente: $d(s_q, s_s) + r_s > r_q$</td></tr>
<tr><td>$p_s = \frac{{r_s}^D}{{r_q}^D} \cdot \frac{d(s_q, s_s)}{r_s}$<br>$p_e = 1 - p_s$</td></tr>
</table>

Uma vez encontrada uma consulta aos $k$-vizinhos mais próximos previamente armazenada com centro próximo, o cálculo de $p_e$ e $p_s$ depende da proporção entre o número de objetos recuperados pela consulta previamente armazenada $k_s$ e o número $k$ de objetos recuperados pela consulta corrente, de acordo com as condições apresentadas na Tabela 3.

**Tabela 3:** Cálculo das porcentagens de custos estimados $p_e$ e de custos armazenados $p_s$ para consultas aos k-vizinhos mais próximos.

| |
|---|
| 1. O número de objetos recuperados pela consulta armazenada é maior que o número de objetos da consulta corrente: $k_s > k$ |
| $$p_s = \frac{k}{k_s}$$ $$p_e = 1 - p_s$$ |
| 2. O número de objetos recuperados pela consulta armazenada é menor ou igual ao número de objetos da consulta corrente: $k_s \leq k$ |
| $$p_s = \frac{k_s}{k}$$ $$p_e = 1 - p_s$$ |

Deve-se notar que uma consulta armazenada é considerada próxima a uma consulta corrente se a região da consulta armazenada cobre o centro da consulta corrente. A busca por consultas armazenadas que cobrem o centro da consulta corrente pode se tornar custosa com o aumento do número de consultas armazenadas. Entretanto, armazenam-se apenas as consultas cuja estimativa teve um erro maior que 10%. Esse valor foi obtido empiricamente.

Os passos a serem seguidos para a execução de uma nova consulta por abrangência estão detalhados no Algoritmo 1.

---

**Algoritmo 1** – Executa e armazena informações de uma consulta por abrangência.

---

**Entrada:** a consulta por abrangência $RQ(s_q, r_q)$, com centro $s_q$ e raio $r_q$.

**Saída:** a informação armazenada para processamentos futuros.

1. Calcula o número estimado de acessos a disco e de cálculos de distância, $DA_g(r_q, r, H, D)$ e $DC_g(r_q, r, H, D)$, usando as Equações 5.17 e 5.21;
2. Busca por uma consulta por abrangência previamente armazenada que tenha centro próximo ao da consulta corrente pelo critério: $d(s_q, s_s) \leq r_s$;
3. Se não encontrou consulta com centro próximo, então $p_e = 1$ e $p_s = 0$, ou seja, $DA_{RQ} = DA_g(r_q, r, H, D)$ e $DC_{RQ} = DC_g(r_q, r, H, D)$. Caso contrário, calcula $p_e$ e $p_s$ de acordo com a Tabela 2;
4. Calcula $DA_{RQ}$ e $DC_{RQ}$ usando as Equações 5.35 e 5.36;
5. Executa a consulta por abrangência $RQ(s_q, r_q)$;
6. Se o erro entre o número estimado de acessos a disco $DA_{RQ}$ e de cálculos de distância $DC_{RQ}$, e o número real de acessos a disco e de cálculos de distância for maior que 10%, então armazenam-se os dados da consulta corrente para serem usados em consultas futuras.

---

Para a execução de uma nova consulta aos *k*-vizinhos mais próximos deve-se seguir os passos detalhados no Algoritmo 2.

---

**Algoritmo 2** – Executa e armazena informações de uma consulta aos *k*-vizinhos mais próximos.

**Entrada:** a consulta aos *k*-vizinhos mais próximos $KNNQ(s_q, k)$, com centro $s_q$ e que recupera $k$ objetos.

**Saída:** a informação armazenada para processamentos futuros.

1. Calcula o número estimado de acessos a disco e de cálculos de distância, $DA_g(k, r, H, D)$ e $DC_g(k, r, H, D)$, usando as Equações 5.28 e 5.32;
2. Busca por uma consulta aos *k*-vizinhos mais próximos previamente armazenada que tenha centro próximo ao da consulta corrente pelo critério: $d(s_q, s_s) \leq r_s$;
3. Se não encontrou consulta com centro próximo, então $p_e = 1$ e $p_s = 0$, ou seja, $DA_{KNNQ} = DA_g(k, r, H, D)$ e $DC_{KNNQ} = DC_g(k, r, H, D)$. Caso contrário, calcula $p_e$ e $p_s$ de acordo com a Tabela 3;
4. Calcula $DA_{KNNQ}$ e $DC_{KNNQ}$ usando as Equações 5.37 e 5.38;
5. Executa a consulta aos *k*-vizinhos mais próximos $KNNQ(s_q, k)$;
6. Se o erro entre o número estimado de acessos a disco $DA_{KNNQ}$ e de cálculos de distância $DC_{KNNQ}$, e o número real de acessos a disco e de cálculos de distância for maior que 10%, então armazenam-se os dados da consulta corrente para serem usados em consultas futuras.

---

## 5.6. Considerações finais

O modelo de custo apresentado neste capítulo estima o número de acessos a disco e o número de cálculos de distância para consultas por abrangência e aos *k*-vizinhos mais próximos em espaços métricos, considerando o uso de um MAM dinâmico.

Inicialmente foram apresentadas equações de estimativa de custos que utilizam apenas parâmetros globais do conjunto de dados: o maior raio de cobertura e o número total de níveis da árvore métrica; e a dimensão intrínseca do conjunto de dados, correspondendo à dimensão de correlação fractal. Para minimizar os cálculos do modelo de custos, a equação que estima o número de cálculos de distância é obtida a partir da equação de estimativa de custo de acessos a disco, reutilizando a maior parte dos cálculos.

Entretanto, estimativas baseadas apenas em parâmetros globais do conjunto de dados não conseguem identificar variações locais dependentes da distribuição dos dados. Então, foram apresentadas equações que melhoram as estimativas considerando, além das estimativas

globais, informações de consultas previamente executadas e que estejam próximas da consulta corrente.

# 6. RESULTADOS EXPERIMENTAIS

## 6.1. Introdução

Para avaliar a eficiência e a eficácia do modelo de custo proposto para estimar o número de acessos a disco e o número de cálculos de distância no processamento de consultas por similaridade, foram utilizados conjuntos de dados reais e sintéticos, os quais serão detalhados neste capítulo.

A Seção 6.2 descreve os conjuntos de dados utilizados. Na Seção 6.3 encontram-se os gráficos que comparam as estimativas com os valores reais de custos para consultas por abrangência, juntamente com análise dos resultados e na Seção 6.4 encontram-se os resultados para consultas aos *k*-vizinhos mais próximos.

## 6.2. Descrição dos conjuntos de dados

Para avaliar o modelo de custo proposto foram utilizados vários conjuntos de dados reais e sintéticos, com tamanhos e dimensões variadas, sendo que foram selecionados para serem apresentados alguns conjuntos de dados significativos, e que retratam e exemplificam bem os resultados obtidos. Esses conjuntos de dados são descritos a seguir:

- **MGCounty:** conjunto de coordenadas geográficas de 27.282 intersecções de vias, ruas e rodovias de *Montgomery County*, *Maryland, EUA*, com dimensão de correlação fractal *D* igual a 1,81. A Figura 12 ilustra a distribuição dos dados desse conjunto, que como pode ser observado, não é uniforme;

- **Cidades:** conjunto referente a 5.507 cidades do Brasil, com três atributos contendo o nome, a latitude e a longitude das cidades, apresentando $D$ igual a 1,81. A distribuição não-uniforme dos dados desse conjunto é mostrada na Figura 13;

- **Currency**: contém 2.311 taxas de câmbio das moedas de seis países, normalizadas utilizando o dólar canadense como referência. As taxas foram obtidas diariamente por um período de 10 anos, com $D$ igual a 2,6;

- **CorelHisto:** conjunto de histogramas de cores extraídos de 68.040 imagens diversas, com 32 atributos e $D$ igual a 3,6;

- **Palavras:** conjunto de 24.893 palavras em inglês, com $D$ igual a 5,7. É um conjunto de dados adimensional;

- **MetricHisto:** conjunto de histogramas métricos extraídos de 4.497 imagens, com $D$ igual a 2,23. O histograma métrico, abordado na Seção 2.2.1, possui número de elementos variável, dependendo somente da imagem em análise e não de todo o conjunto de imagens. Trata-se, portanto, de um conjunto de dados adimensional;

- **Sintético6D:** conjunto de 20.000 dados sintéticos gerados aleatoriamente, sem atributos correlacionados, ou seja, $D$ é igual à dimensão de imersão.

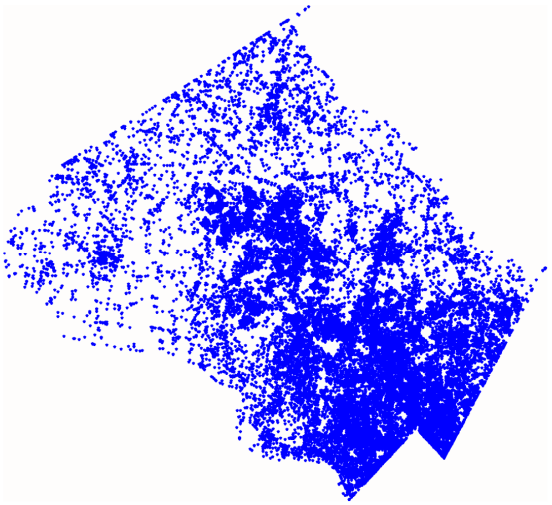


**Figura 12:** Distribuição dos dados do conjunto **MGCounty**.

**Figura 13:** Distribuição dos dados do conjunto **Cidades**.

A Tabela 4 sumariza as principais informações dos conjuntos de dados, incluindo o *fat-factor* e a métrica utilizada.

**Tabela 4:** Informações sobre os conjuntos de dados usados nos experimentos.

| Conjunto de Dados | Número de Objetos | Dimensão | | fat-factor | Métrica |
|---|---|---|---|---|---|
| | | Imersão | Intrínseca *D* | | |
| MGCounty | 27.282 | 2 | 1,81 | 0,01 | Euclidiana – $L_2$ |
| Cidades | 5.507 | 3 | 1,81 | 0,03 | Euclidiana – $L_2$ |
| Currency | 2.311 | 6 | 2,6 | 0,02 | Euclidiana – $L_2$ |
| CorelHisto | 68.040 | 32 | 3,6 | 0,12 | Euclidiana – $L_2$ |
| Palavras | 24.893 | - | 5,7 | 0,59 | $L_{Edit}$ |
| MetricHisto | 4.497 | - | 2,23 | 0,15 | *DM* |
| Sintético6D | 20.000 | 6 | 6 | 0,07 | Euclidiana – $L_2$ |

Na maioria dos conjuntos de dados foram usadas funções de distância da família $L_p$, pois esses conjuntos são vetoriais (caso particular de espaço métrico – Seção 2.2.1). Entretanto, os conjuntos de dados **Palavras** e **MetricHisto** são puramente métricos, ou seja, contém dados complexos não vetoriais. Portanto, esses conjuntos de dados não têm dimensão de imersão. Como abordado na Seção 2.2.1, para o conjunto de dados **Palavras** utilizou-se a função de distância métrica $L_{Edit}$ que compara duas palavras contando o número mínimo necessário de inserções, remoções e substituições de letras para transformar uma palavra na outra. Para o conjunto **MetricHisto** a comparação entre objetos é feita por uma função de distância métrica denominada *DM*.

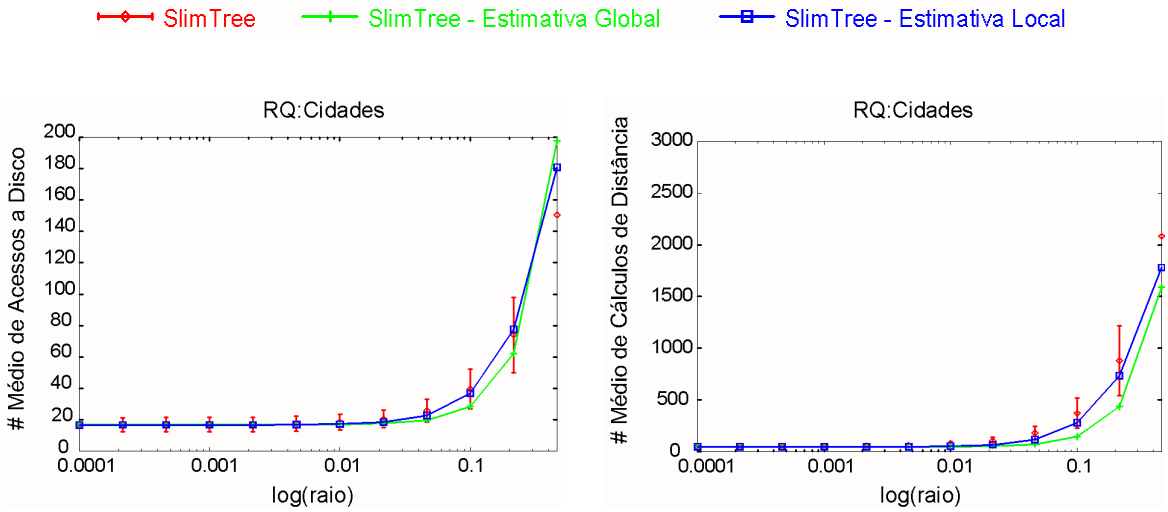
## 6.3. Resultados para consultas por abrangência

Os resultados dos testes são apresentados em gráficos que comparam o número real de acessos a disco e de cálculos de distância quando consultas por abrangência são processadas usando a *Slim-Tree* (curva SlimTree), a estimativa global de acessos a disco e cálculos de distância calculada usando as equações de custo propostas na Seção 5.3 (curva SlimTree – Estimativa Global) e a estimativa de custos local como abordada na Seção 5.5 (curva SlimTree – Estimativa Local), para os conjuntos de dados descritos na seção anterior.

No caso das curvas reais e com estimativa local, cada ponto no gráfico corresponde à média de 500 consultas com o mesmo raio, com diferentes objetos centrais de busca. Os 500 objetos centrais de busca são amostras extraídas dos próprios conjuntos de dados, de maneira que são amostras com grande probabilidade de serem usadas em consultas reais. No caso das curvas reais, para cada ponto no gráfico também é calculado o desvio padrão, ou seja, os valores mínimo e máximo do número de acessos a disco e cálculos de distância para a média das 500 consultas com o mesmo raio. No caso da estimativa global, o raio é o único parâmetro da consulta considerado, ou seja, o resultado para o número de acesso a disco e cálculos de distância será o mesmo para qualquer uma das 500 consultas.

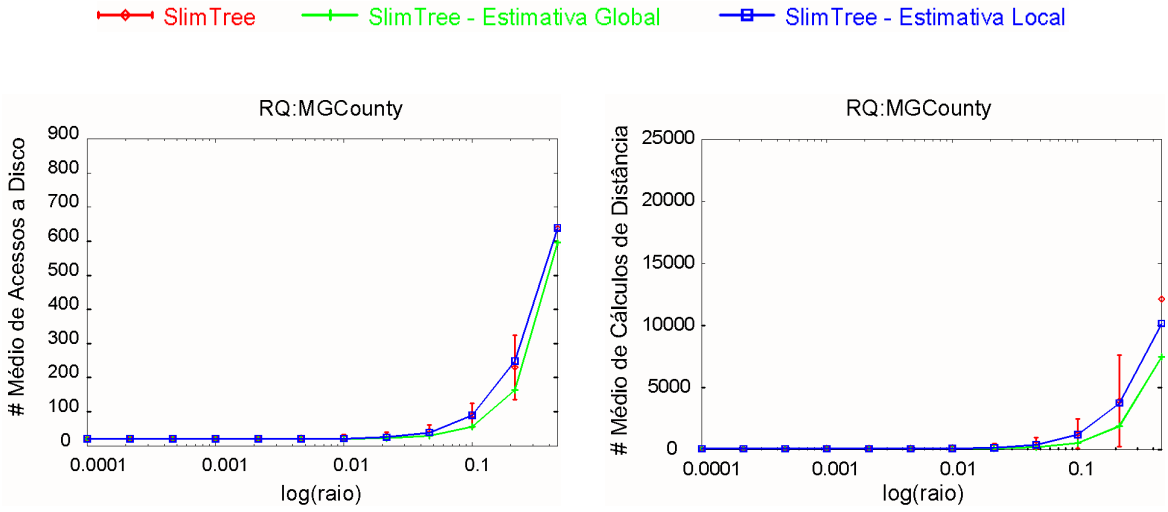
Os gráficos apresentam os valores dos raios normalizados em relação ao maior raio de cobertura da árvore métrica iniciando em 0.0001 até um valor próximo a 1. Entretanto, nesse intervalo o valor máximo de interesse é 0.1 que tipicamente representa 10% dos objetos indexados.

As Figuras 14 e 15 apresentam os conjuntos de dados de baixa dimensão de imersão e com dimensão intrínseca com valor bem próximo à dimensão de imersão. Para ambos os conjuntos de dados o modelo de custo mostrou-se bastante eficaz. Considerando raio $\leq 0.1$, tanto as estimativas globais como as locais ficam muito próximas das medidas reais; para o raio $> 0.1$, as estimativas locais ainda continuam eficazes.




**(a)** **(b)**

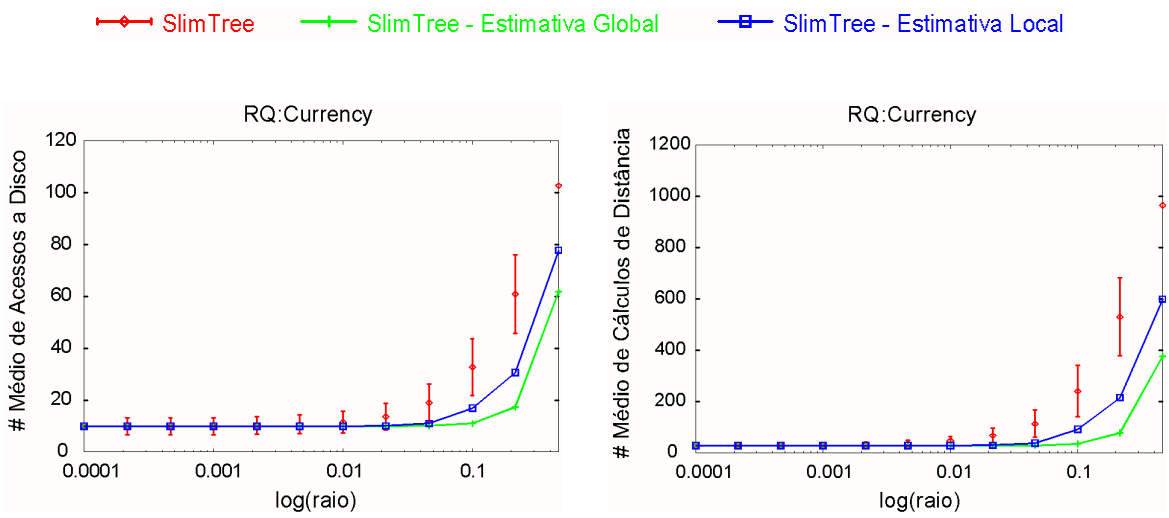
**Figura 14:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **Cidades**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

**Figura 15:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **MGCounty**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

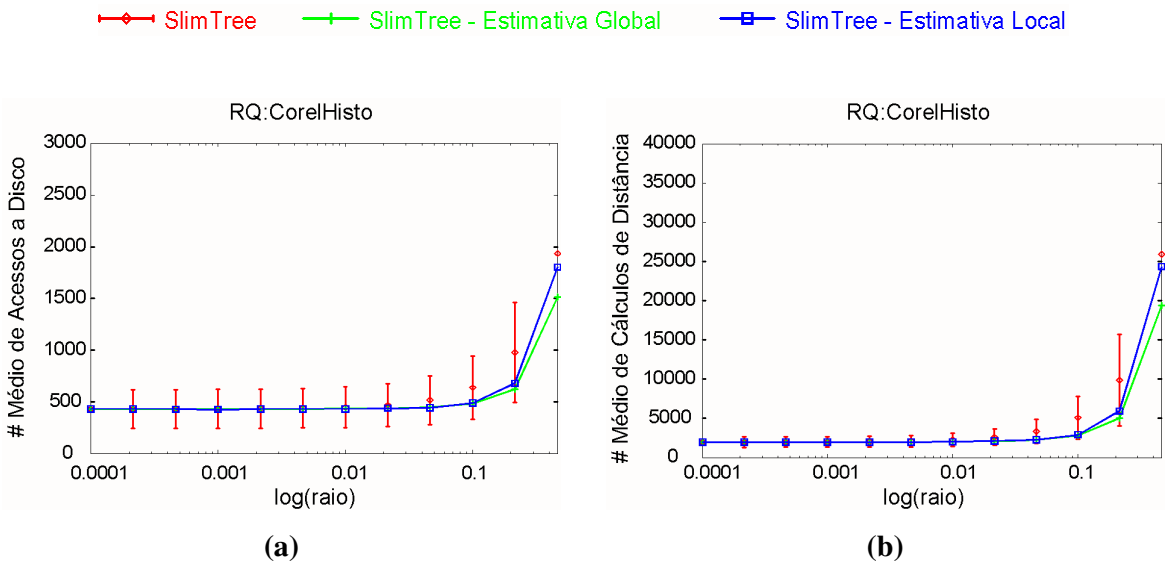
Considerando um valor de dimensão de imersão médio, e dimensão intrínseca baixa, a Figura 16 mostra os resultados das estimativas para o conjunto de dados **Currency**. Como pode ser observado, o modelo de custo se mostra eficaz até raio = 0.01, tanto para estimativas globais quanto para estimativas locais. Até raio = 0.1 as estimativas locais ainda ficam dentro do desvio padrão das medidas reais. Nesse conjunto, as estimativas locais não conseguiram melhorar muito os resultados, pois este é um conjunto pequeno, contendo poucos dados se comparado a outros conjuntos de dados testados. Esse fato afeta o mecanismo de tratamento local para construção do modelo de custo, uma vez que as estimativas locais dependem de resultados de consultas previamente executadas.




**(a)** **(b)**

**Figura 16:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **Currency**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

Como exemplo de um conjunto de dados de alta dimensão de imersão, tem-se o conjunto de dados **CorelHisto**, cujo resultado de testes com as estimativas de custo é mostrado na Figura 17. Como pode ser observado, novamente o modelo de custo mostrou-se eficaz com relação às estimativas globais e locais, ficando sempre dentro do desvio padrão das medidas reais e, para o raio até 0.1, as medidas estimadas ficam bem próximas das medidas reais.

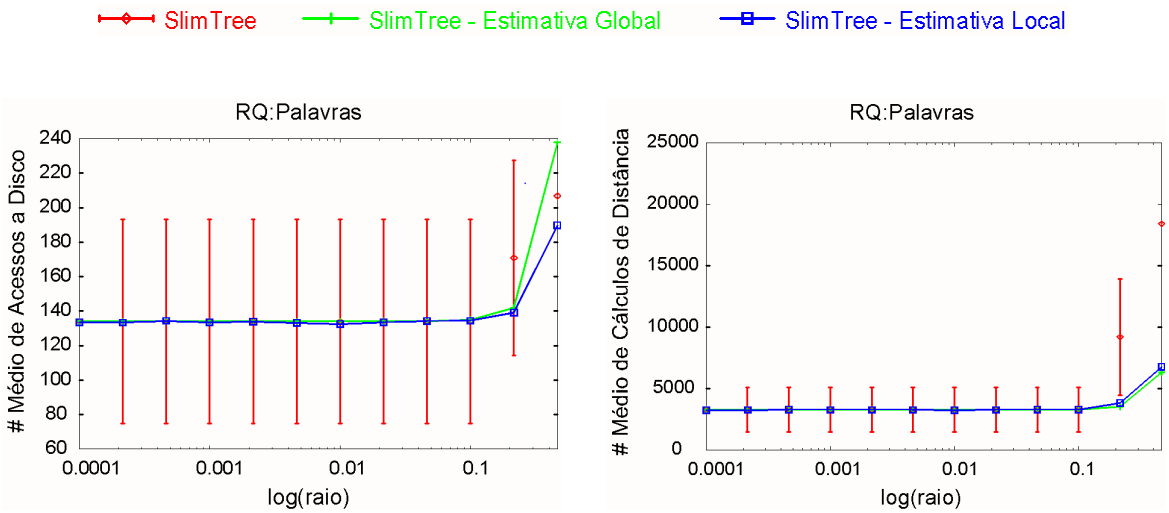



**(a)** **(b)**

**Figura 17:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **CorelHisto**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

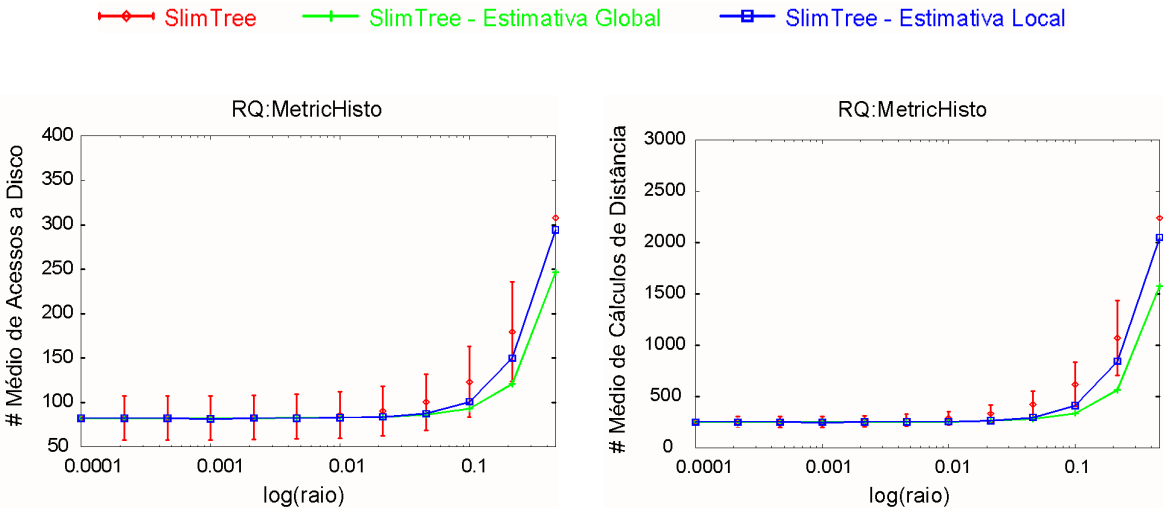
Para os conjuntos de dados adimensionais (puramente métricos), os resultados mostrados nas Figuras 18 e 19 também mostram que o modelo de custo é bastante eficaz. No caso do conjunto de dados **Palavras** (Figura 18), pode-se considerar que o *fat-factor* influenciou muito nos resultados positivos do modelo, pois esse conjunto gera uma árvore com muita sobreposição, *fat-factor* = 0,59, que como pode ser visto na Tabela 4, é um valor bem mais alto que o dos outros conjuntos de dados.

A Figura 20 ilustra os resultados para dados sintéticos com dimensão de imersão e dimensão intrínseca igual a 6. Como pode ser observado, tanto as estimativas globais quanto as estimativas locais ficaram muito próximas das medidas reais.



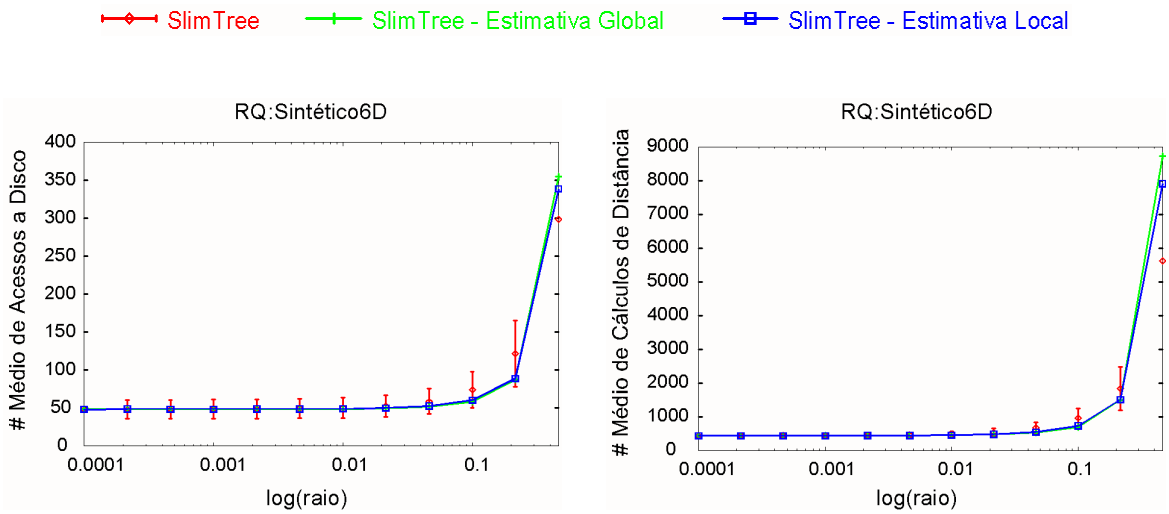

**(a)** **(b)**

**Figura 18:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **Palavras**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

**Figura 19:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **MetricHisto**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

**Figura 20:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas por abrangência**, para o conjunto de dados **Sintético6D**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

Analisando os resultados experimentais obtidos, pode-se concluir que o modelo de custo proposto é eficaz. Considerando consultas por abrangência que retornam até 10% dos objetos indexados (raio $\leq 0.1$ nos gráficos), as estimativas ficam majoritariamente dentro do desvio padrão das medidas reais e, como pode ser observado nos gráficos, os valores estimados ficam muito próximos das medidas reais. Acima de 10% de seletividade as estimativas ainda ficam dentro do desvio padrão das medidas reais na maioria dos conjuntos de dados testados. Os piores casos encontrados foram os conjuntos **Currency** e **Palavras** para estimativa de cálculos de distância, mas abaixo de 10% de seletividade o modelo de custo ainda mostrou-se eficaz mesmo nesses casos.

Em relação à eficiência pode-se considerar que, para consultas por abrangência, o modelo é eficiente uma vez que utiliza parâmetros globais do conjunto de dados: o raio de cobertura da consulta a ser executada $r_q$; o maior raio de cobertura $r$ e a altura $H$ da árvore métrica, valores que podem ser obtidos/atualizados durante o processo de indexação e armazenados para serem utilizados pelo modelo de custo; e a dimensão de correlação fractal $D$ que não varia em

relação ao tamanho do conjunto de dados, ou seja, *D* tem o mesmo valor mesmo após inserções e remoções de dados do conjunto. Deve-se considerar também que a equação que estima o número de cálculos de distância é obtida a partir da equação de estimativa de custo de acessos a disco, reutilizando a maior parte dos cálculos. No caso das estimativas locais, a busca por consultas armazenadas pode se tornar custosa com o aumento do número de consultas armazenadas. Entretanto, além de serem armazenadas apenas as consultas cuja estimativa teve um erro grande, pode-se utilizar outros recursos para melhorar o desempenho, como por exemplo, remover as consultas previamente armazenadas totalmente cobertas pelo raio de cobertura da consulta corrente e usar uma estrutura de indexação para esses dados.

## 6.4. Resultados para consultas aos *k*-vizinhos mais próximos

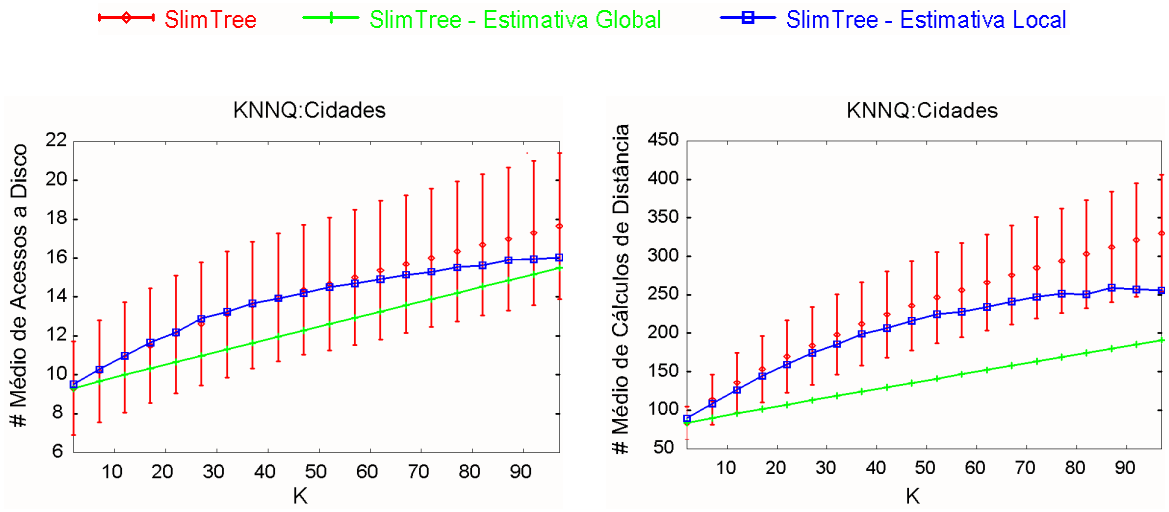
Assim como para consultas por abrangência, os resultados dos testes para consultas aos *k*-vizinhos mais próximos são apresentados em gráficos que comparam o número real de acessos a disco e de cálculos de distância usando a *Slim-Tree* (curva SlimTree), a estimativa global de acessos a disco e cálculos de distância calculada usando as equações de custo propostas na Seção 5.4 (curva SlimTree – Estimativa Global) e a estimativa de custos local como abordada na Seção 5.5 (curva SlimTree – Estimativa Local), para os conjuntos de dados descritos na Seção 6.2.

No caso das curvas reais e com a estimativa final, cada ponto no gráfico corresponde à média de 500 consultas com o mesmo valor de *k*, com diferentes objetos centrais de busca. Os 500 objetos centrais de busca são amostras extraídas dos respectivos conjuntos de dados. No caso das curvas reais, para cada ponto no gráfico também é calculado o desvio padrão, ou seja, os valores mínimo e máximo do número de acessos a disco e cálculos de distância para a média das 500 consultas com o mesmo valor de *k*. No caso da curva estimada, o *k* é o único parâmetro da consulta considerado, ou seja, o resultado para o número de acesso a disco e cálculos de distância será o mesmo para qualquer uma das 500 consultas.

Para os testes foram utilizados valores de *k* variando de 0 até um valor próximo a 100. Entretanto, deve-se observar que para a maioria das consultas aos *k*-vizinhos mais próximos os valores de interesse de *k* são pequenos, aproximadamente até 10 ou 20 no máximo. Por isso, os valores mais importantes são os obtidos para *k* pequenos.

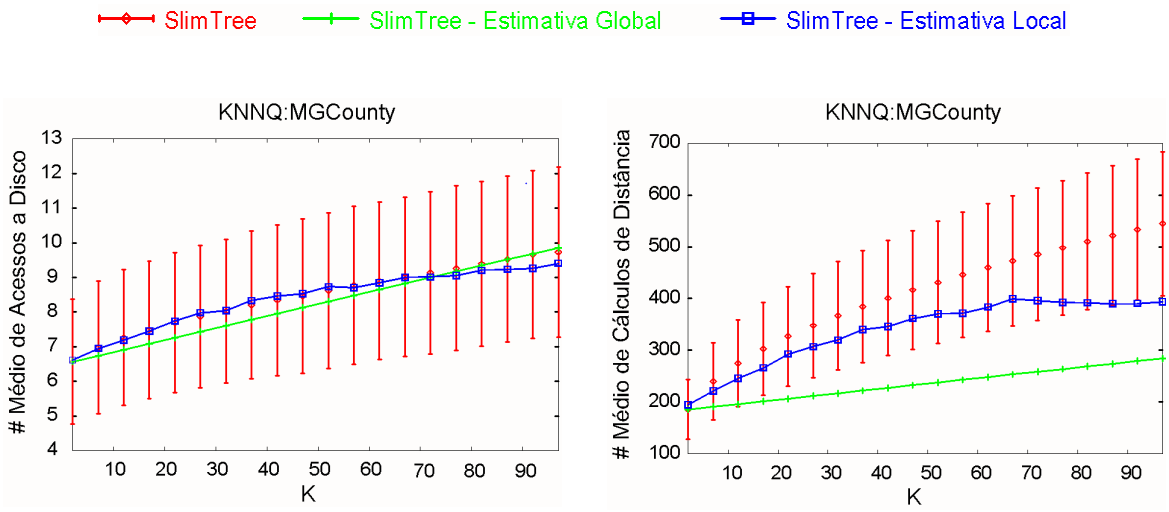
Para conjuntos com valores de dimensão de imersão pequenos (ver Figuras 21 e 22), o modelo de custo se mostrou eficaz principalmente no caso das estimativas locais, com as medidas bem próximas das medidas reais principalmente para estimativa de acessos a disco. Para o número de cálculos de distância, as estimativas globais ficam fora do desvio padrão. Entretanto, considerando que a maioria das consultas aos *k*-vizinhos mais próximos os valores de interesse de *k* são pequenos, as estimativas ainda ficam dentro do desvio padrão das medidas reais.

A Figura 23 mostra os resultados das estimativas de custo para o conjunto de dados **Currency**, com dimensão de imersão média. Os resultados são similares aos dos conjuntos de dados com dimensão de imersão baixa, com estimativas locais melhores, próximas das medidas reais tanto para o número de acessos a disco quanto para cálculos de distância.



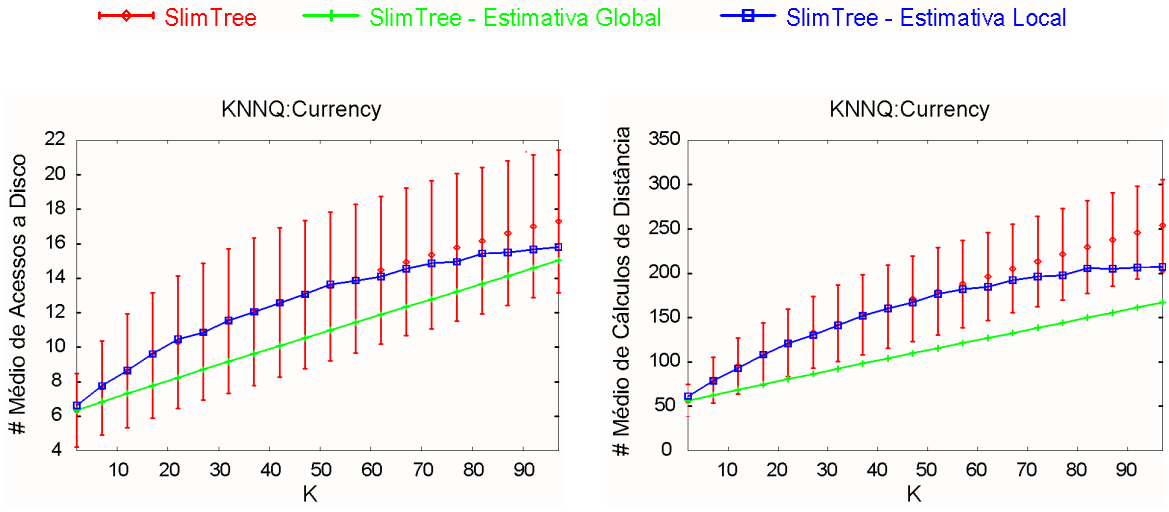

**(a)** **(b)**

**Figura 21:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas aos *k*-vizinhos mais próximos**, para o conjunto de dados **Cidades**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

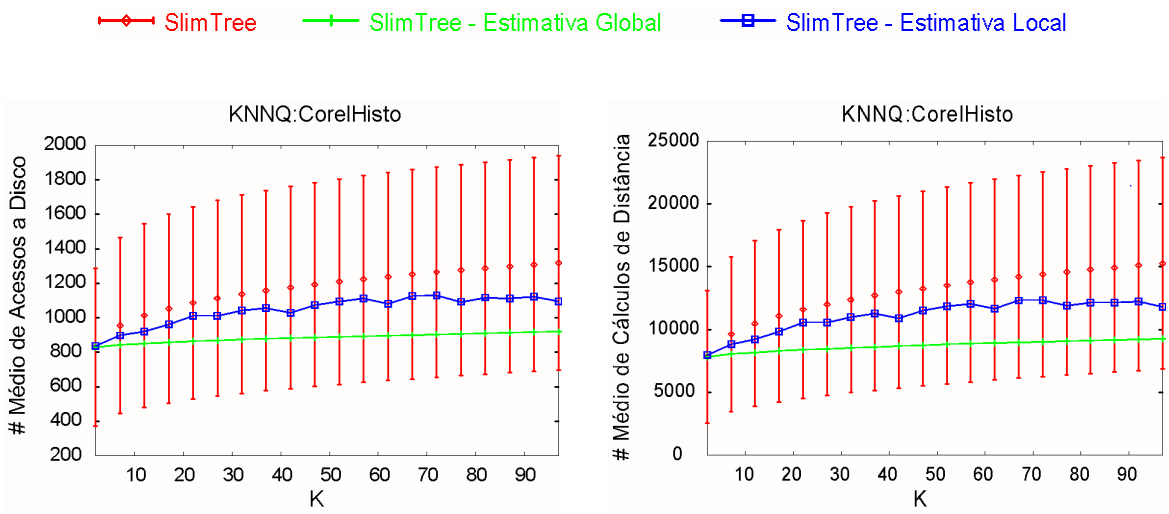
**Figura 22:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas aos *k*-vizinhos mais próximos**, para o conjunto de dados **MGCounty**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

**Figura 23:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas aos *k*-vizinhos mais próximos**, para o conjunto de dados **Currency**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

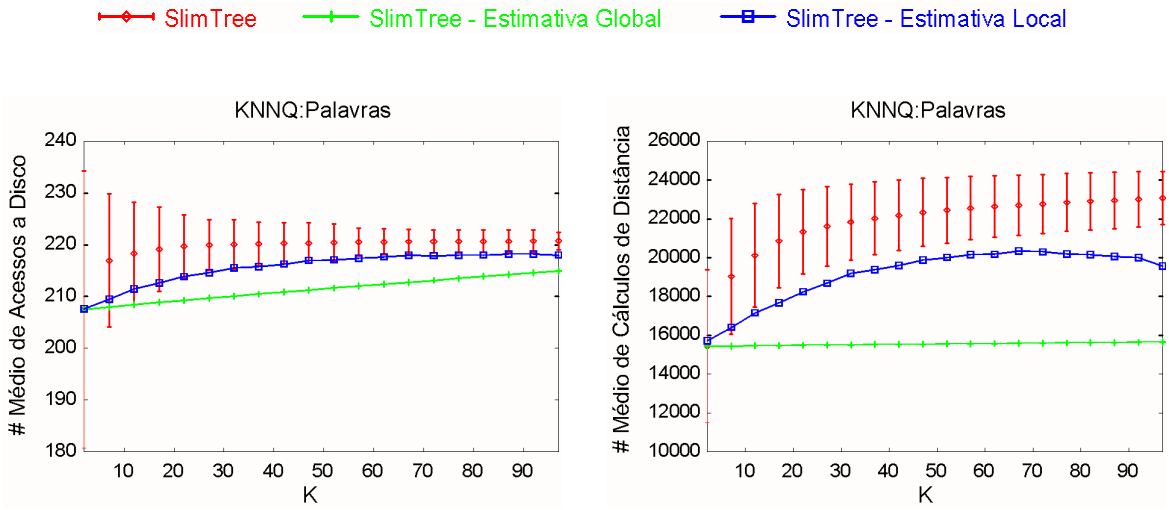
Para o conjunto de dados **CorelHisto** (ver Figura 24), com alta dimensão de imersão, as curvas estimadas ficam sempre dentro do desvio padrão das medidas reais.




**(a)** **(b)**

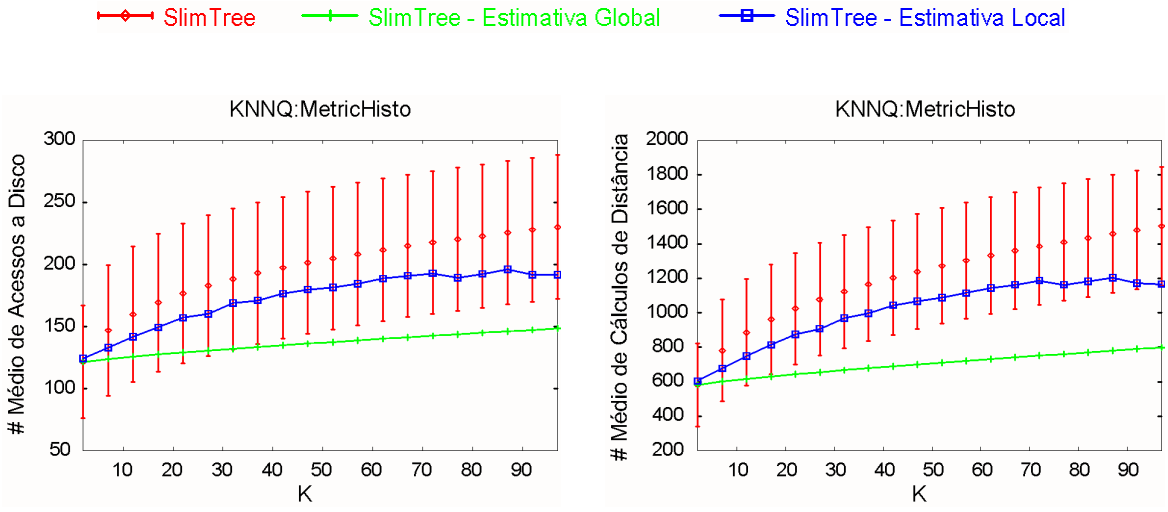
**Figura 24:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de consultas aos *k*-vizinhos mais próximos, para o conjunto de dados **CorelHisto**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

Para os conjuntos de dados adimensionais (puramente métricos) os resultados dos testes são mostrados nas Figuras 25 e 26. As estimativas globais não seguiram muito perto os valores reais, principalmente para o número de cálculos de distância. Entretanto, para o conjunto de dados **MetricHisto** (Figura 26) ainda é possível obter uma estimativa global razoável para valores pequenos de *k*. Note-se no entanto que a variação real das consultas é também muito grande, como pode ser visto pelo desvio padrão mostrado. As estimativas locais ficam dentro do desvio padrão das medidas estimadas, sendo que os piores casos ocorrem para o conjunto de dados **Palavras**. Mas, novamente para valores pequenos de *k* ainda é possível obter uma estimativa local útil.




**(a)** **(b)**

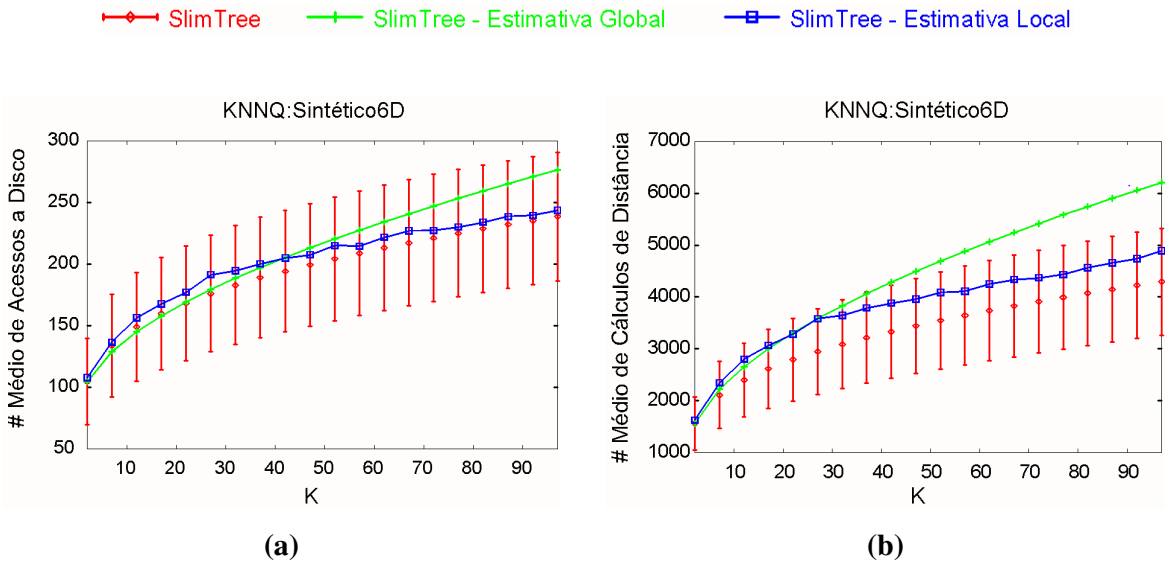
**Figura 25:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas aos *k*-vizinhos mais próximos**, para o conjunto de dados **Palavras**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.




**(a)** **(b)**

**Figura 26:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de **consultas aos *k*-vizinhos mais próximos**, para o conjunto de dados **MetricHisto**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

Considerando dados sintéticos, com dimensão de imersão média e dimensão intrínseca igual a dimensão de imersão, tem-se que as estimativas locais e globais estão dentro do desvio padrão das medidas reais, como pode ser observado na Figura 27. O pior caso ocorre para estimativas globais do número de cálculos de distância, mas as medidas estimadas ainda ficam dentro do desvio padrão das medidas reais para valores pequenos/médios de *k*.




**(a)** **(b)**

**Figura 27:** Comparação entre o número de acessos a disco e o número de cálculos de distância reais (SlimTree), estimados com parâmetros **globais** (SlimTree – Estimativa Global) e estimados utilizando informações **locais** sobre o conjunto de dados (SlimTree – Estimativa Local) de consultas aos *k*-vizinhos mais próximos, para o conjunto de dados **Sintético6D**: (a) Custo de Acessos a Disco; (b) Custo de Cálculos de Distância.

Analisando os resultados experimentais nota-se que as estimativas de custo globais ficam fora do desvio padrão em alguns conjuntos de dados. Isso deve-se ao erro na estimativa do raio da consulta a partir de *k*. Entretanto, considerando as estimativas locais que usam dados de consultas previamente executadas, pode-se concluir que o modelo de custo proposto é bastante eficaz. As estimativas locais ficam sempre dentro do desvio padrão das medidas reais e, em alguns casos, os valores estimados ficam muito próximos das medidas reais. Ainda, considerando que, para a maioria das consultas aos *k*-vizinhos mais próximos, os valores de interesse de *k* são pequenos, as estimativas ainda ficam dentro do desvio padrão das medidas reais na maioria dos conjuntos de dados testados. O pior caso encontrado foi o conjunto de dados **Palavras**, mas para valores pequenos de *k* ainda é possível obter alguma estimativa relevante.

Assim como para consultas por abrangência, em relação à eficiência pode-se considerar que, para consultas aos *k*-vizinhos mais próximos, o modelo é eficiente uma vez que utiliza parâmetros globais do conjunto de dados: o valor de *k* da consulta a ser executada; o maior raio de cobertura *r* e a altura *H* da árvore métrica, a serem obtidos/atualizados durante o processo de indexação; e a dimensão de correlação fractal *D* que não varia em relação ao tamanho do conjunto de dados. As consultas aos *k*-vizinhos mais próximos foram consideradas por este trabalho como consultas por abrangência com a proposta de uma equação para determinar o raio de cobertura. Assim, a eficiência do modelo de custo proposto pode ser considerada a mesma para ambos os tipos de consultas.

## 6.5. Considerações finais

Os experimentos realizados consideraram conjuntos de dados reais e sintéticos, alguns vetoriais e outros puramente métricos. Os resultados dos experimentos foram mostrados por meio de gráficos que comparam o número real de acessos a disco e de cálculos de distância quando consultas por abrangência e aos *k*-vizinhos mais próximos são processadas usando a *Slim-Tree*, a estimativa global de acessos a disco e cálculos de distância e a estimativa local de custos usando dados de consultas previamente executadas.

De maneira geral, os resultados dos experimentos mostraram que o modelo de custo proposto neste trabalho é eficaz, com as curvas estimadas dentro do desvio padrão das medidas reais na maioria dos casos e, em alguns casos os valores estimados ficam muito próximos das medidas reais.

# 7. CONCLUSÕES

## 7.1. Considerações gerais

Consultas por similaridade em espaços métricos, considerando ambientes dinâmicos, podem ter seu desempenho bastante melhorado por meio do uso de métodos de acesso métricos. Entretanto, há casos em que uma simples busca seqüencial pode ser menos onerosa do que o uso de um método de acesso, como por exemplo, quando o raio de busca de uma consulta por abrangência é relativamente grande, em relação ao diâmetro do conjunto de dados. É importante ressaltar também que o custo computacional para efetuar consultas por similaridade tem ordem de grandeza bem maior do que para processar consultas tradicionais. Desse modo, a possibilidade de estimar o custo computacional para processar consultas por similaridade propicia dispor de um parâmetro importante para o otimizador de consultas de um Sistema de Gerenciamento de Bases de Dados.

Em uma consulta por similaridade em espaços métricos envolvendo dados complexos, as comparações são realizadas usando uma função de distância atuando como uma métrica. Essão função é muito custosa, na grande maioria das vezes. Por isso, é importante que um modelo de custo possa estimar o número de cálculos da função de distância métrica, além do número de acessos a disco.

Um modelo de custo para consultas por similaridade deve considerar a dimensionalidade do conjunto de dados e a distribuição dos objetos no espaço. A alta dimensionalidade provoca problemas nas estruturas de indexação, como o aumento da sobreposição de nós dos métodos de acesso e do custo de processamento decorrente da comparação entre os objetos (cálculos de distância). Considerar que os dados estão uniformemente distribuídos no espaço implica em considerar a dimensão em que os dados estão imersos no espaço – dimensão de imersão. Entretanto, na maioria dos conjuntos de dados reais, os atributos tendem a estar

correlacionados, levando a uma distribuição não uniforme, mais compacta e, conseqüentemente, os dados estão distribuídos em uma dimensão intrínseca menor. Assim, considerando a dimensão de imersão do conjunto de dados pode-se chegar a estimativas de custo pessimistas, que podem não ser reais caso a dimensão intrínseca do conjunto seja menor que a de imersão. No caso de dados métricos, que são adimensionais, a utilização do conceito de dimensão intrínseca para estimativas de custo é ainda mais relevante. Além disso, a distribuição não é uniforme, fazendo com que consultas com os mesmos parâmetros mas feitas em regiões distintas do espaço apresentem custos bastante diferentes.

Diante desse cenário, este trabalho desenvolveu um modelo de custo para consultas por similaridade em espaços métricos pressupondo o seguinte:

- os conjuntos de dados são indexados por um método de acesso métrico dinâmico;
- há uma métrica bem definida para a comparação dos objetos dos conjuntos de dados complexos em geral (vetoriais ou adimensionais);
- os dados do conjunto não estão uniformemente distribuídos, sendo que a dimensão intrínseca do conjunto é calculada usando a dimensão de correlação fractal.

O modelo proposto considera para as estimativas de custo não só o número de acessos a disco, mas também o número de cálculos da métrica. Outro aspecto que um modelo de custo para consultas por similaridade deve considerar é o tipo da consulta. Os tipos mais usuais e demandados na literatura, e desse modo considerados pelo modelo de custo proposto por este trabalho, são as consultas por abrangência e as consultas aos *k*-vizinhos mais próximos.

Para o desenvolvimento de um modelo de custo eficiente e eficaz também é necessário considerar as características do método de acesso a ser utilizado para a indexação dos dados complexos. Este trabalho escolheu como plataforma de desenvolvimento a *Slim-Tree*, que é um método de acesso métrico dinâmico, baseado em árvore, e cuja estrutura segue a abordagem da técnica de indexação $B^+$*-Tree*, muito utilizada para dados convencionais. Essa escolha foi feita com o objetivo de que o modelo de custo proposto possa ser utilizado como base para o desenvolvimento de modelos para outras estruturas dinâmicas de indexação em espaços métricos, uma vez que, como a *Slim-Tree*, a maioria dos métodos de acesso também

segue a abordagem da $B^+$-*Tree*. Uma característica importante da *Slim-Tree* que também influenciou em sua escolha, é a possibilidade de quantificação da sobreposição de nós por meio do *fat-factor,* uma vez que a sobreposição entre nós é o principal problema dos métodos de acesso métricos.

Os experimentos realizados para validar o modelo de custo proposto mostraram que as medidas estimadas ficam muito próximas das medidas reais para boa parte dos conjuntos de dados, principalmente para as consultas por abrangência. Considerando as condições limites para que a utilização de estruturas de indexação contribua para a eficiência de consultas por similaridade (raio da consulta por abrangência menor que 10% do diâmetro do conjunto de dados e valores pequenos de $k$ para consultas aos $k$-vizinhos mais próximos), as medidas estimadas sempre ficam dentro do desvio padrão das medidas reais. Assim, os experimentos confirmaram que o modelo de custo proposto por este trabalho é eficaz e pode ser aplicado a vários tipos de conjuntos de dados, mesmo quando a árvore métrica apresenta muita sobreposição de nós. Em relação à eficiência pode-se considerar que o modelo é eficiente uma vez que utiliza parâmetros globais do conjunto de dados, que podem ser obtidos/atualizados durante o processo de indexação e armazenados para serem utilizados no momento do cálculo das estimativas de custo. Deve-se considerar também que a equação que estima o número de cálculos de distância é obtida a partir da equação de estimativa de custo de acessos a disco, reutilizando a maior parte dos cálculos.

## 7.2. Principais contribuições

A principal contribuição desta tese é o desenvolvimento de um modelo de custo para consultas por similaridade a dados complexos, com enfoque em dados do tipo imagem. Os dados complexos devem ser representados por conjuntos de vetores de características e indexados usando um método de acesso métrico dinâmico.

Foram desenvolvidos dois conjuntos de equações, ambos considerando a estimativa do número de acessos a disco e do número de cálculo de distância necessários para o processamento de consultas por abrangência e aos $k$-vizinhos mais próximos:

- **Estimativa de custo global:** o primeiro conjunto de equações se baseia apenas em parâmetros globais do conjunto de dados: o raio de cobertura da consulta a ser executada; o maior raio de cobertura e o número total de níveis (altura) da árvore métrica, valores obtidos/atualizados durante o processo de indexação; e a dimensão de correlação fractal *D* que não varia em relação ao tamanho do conjunto de dados. Deve-se considerar também que a equação que estima o número de cálculos de distância é obtida a partir da equação de estimativa de custo de acessos a disco, reutilizando a maior parte dos cálculos. Essas características proporcionam uma estimativa de custo inicial de maneira rápida. Entretanto, essas estimativas iniciais não conseguem muitas vezes identificar variações locais que ocorrem devido à distribuição regional dos dados.

- **Estimativa de custo local:** o segundo conjunto de equações considera variações locais do conjunto de dados e aprimora as estimativas considerando custos reais de consultas previamente executadas. Essa estimativa local considera pesos de custos estimados e reais previamente medidos como função de distância da consulta corrente e de consultas anteriormente executadas e armazenadas. Esse procedimento demanda armazenar apenas poucas consultas, minimizando o custo de encontrar uma consulta previamente armazenada que se qualifique para o processo de aprimoramento.

Uma outra contribuição do trabalho que vale ressaltar é que o modelo de custo leva em consideração o principal problema que afeta os métodos de acesso métricos, a sobreposição de nós. Caso esse aspecto não fosse considerado, o modelo seria eficaz apenas para conjuntos de dados cuja estrutura de indexação ficasse o mais perto possível de ser ótima, isto é, sem sobreposição de nós.

Pode ainda ser considerada como contribuição decorrente deste trabalho a estimativa do raio inicial de uma consulta aos *k*-vizinhos mais próximos, usando apenas uma equação. Esse aspecto pode ser utilizado também para a otimização de algoritmos que implementam consultas aos *k*-vizinhos mais próximos, que utilizam como raio inicial o maior raio de cobertura da árvore métrica a ser ajustado na medida em que os *k* objetos são recuperados. O raio inicial pode, então, ser estimado pela equação proposta neste trabalho, reduzindo o número de ajustes.

## 7.3. Propostas para trabalhos futuros

Como continuidade imediata deste trabalho propõe-se incorporar o modelo de custo proposto aos Sistemas de Gerenciamento de Bases de Dados objeto-relacionais atuais que contemplem a arquitetura de *extensible indexing and optimization frameworks* [Stonebraker_1986]. Basicamente as equações de custo podem ser tratadas como funções definidas pelo usuário (UDF – *User Defined Functions*) para apoiar o otimizador de consultas. Da mesma maneira, a estrutura de indexação, neste caso a *Slim-Tree*, incluindo os conjuntos de características extraídas dos dados complexos, podem ser tratadas como tipos definidos pelo usuário (UDT – *User Defined Types*).

Outras pesquisas que podem ser realizadas a partir deste trabalho recaem diretamente sobre o modelo de custo proposto, tais como:

- Aprimoramento do modelo para consultas aos *k*-vizinhos mais próximos, mais especificamente melhorando a estimativa inicial do raio das consultas;

- No caso das estimativas locais, pode-se definir uma estrutura de armazenamento e indexação para otimizar a recuperação de consultas previamente armazenadas;

- Aplicar o modelo para outros métodos de acesso métricos visando verificar a sua facilidade de adaptação. Provavelmente, para estruturas de indexação com características similares às da *Slim-Tree* (por exemplo, a *M-Tree*), a sua aplicação será direta;

- Estender o modelo para suportar outras consultas por similaridade, como por exemplo, operações envolvendo junção.

Acredita-se que essas pesquisas futuras sejam determinantes para a consolidação do modelo de custo proposto por este trabalho e, principalmente, para o uso do modelo em sistemas comerciais de bancos de dados que manipulam dados complexos.

# REFERÊNCIAS BIBLIOGRÁFICAS


[Adelhard_1999] K. Adelhard, S. Nissen-Meyer, C. Pistitsch, U. Fink, M. Reiser, "*Functional Requirements for a HIS-RIS-PACS Interface Design Including Integration of "Old" Modalities*," *Methods of Information in Medicine*, vol. 38,1999, pp. 1-8.

[Aggarwal_2004] C. C. Aggarwal, " *An Efficient Subspace Sampling Framework for High-Dimensional Data Reduction, Selectivity Estimation, and Nearest-Neighbor Search*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 16,October 2004, pp. 1247-1262.

[Aslandogan_1999] Y. A. Aslandogan and C. T. Yu, "*Techniques and Systems for Image and Video Retrieval*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 11,Jan/Feb 1999, pp. 56-63.

[Beckmann_1990] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, "*The R*-tree: An Efficient and Robust Access Method for Points and Rectangles*," presented at ACM SIGMOD International Conference on Management of Data, 1990, pp. 322-331.

[Belussi_1995] A. Belussi and C. Faloutsos, "*Estimating the Selectivity of Spatial Queries Using the Correlation Fractal Dimension*," presented at International Conference on Very Large Databases (VLDB), Zurich, Switzerland, September 11-15, 1995, pp. 299-310.

[Belussi_1998] A. Belussi and C. Faloutsos, "*Self-Spacial Join Selectivity Estimation Using Fractal Concepts*," *ACM Transactions on Information Systems*, vol. 16,April 1998, pp. 161-201.

[Berchtold_1997] S. Berchtold, C. Böhm, D. A. Keim, H.-P. Kriegel, "*A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space*," presented at ACM Symposium on Principles of Database Systems (PODS), Tucson, AZ, May 12-14, 1997, pp. 78-86.

[Bertino_1994] E. Bertino, *Object Oriented Database Systems*: Addison-Wesley, 1994.

[Beyer_1999] K. Beyer, J. Godstein, R. Ramakrishnan, U. Shaft, "*When is "Nearest Neighbor" Meaningful?*," presented at International Conference on Database Theory (ICDT), Jerusalem, Israel, January 10-12, 1999, pp. 217-235.

[Böhm_2000] C. Böhm, "*A Cost Model for Query Processing in High Dimensional Data Spaces*," *ACM Transactions on Database Systems (TODS)*, vol. 25,June 2000, pp. 129 - 178.

[Böhm_2001] C. Böhm, S. Berchtold, D. A. Keim, "*Searching in High-Dimensional Spaces - Index Structures for Improving the Performance of Multimedia Databases*," *ACM Computing Surveys*, vol. 33,September 2001, pp. 322 - 373.

[Bueno_2002] J. M. Bueno, "*Suporte à Recuperação de Imagens Médicas baseada em Conteúdo através de Histogramas Métricos*," in *Departamento de Ciências de Computação*. São Carlos, SP: Universidade de São Paulo, pp. 146.

[Cao_2000] X. Cao and H. K. Huang, "*Current Status and Future Advances of Digital Radiography and PACS*," *IEEE Engineering in Medicine and Biology Magazine*, vol. 9,Sept-Oct, pp. 80-88.

[Cattell_1994] R. Cattell, *Object data management: Object-Oriented and Extended Relational*: Addison-Wesley, 1994.

[Chávez_2001] E. Chávez, G. Navarro, R. A. Baeza-Yates, J. L. Marroquín, "*Searching in Metric Spaces*," *ACM Computing Surveys*, vol. 33,September 2001, pp. 273-321.

[Christodoulakis_1984] S. Christodoulakis, " *Implications of Certain Assumptions in Database Performance Evaluation*," *ACM Transactions on Database Systems (TODS)*, vol. 9, pp. 163-186.

[Ciaccia_1997] P. Ciaccia, M. Patella, P. Zezula, "*M-tree: An efficient access method for similarity search in metric spaces*," presented at International Conference on Very Large Databases (VLDB), Athens, Greece, September 1997, pp. 426-435.

[Ciaccia_1998] P. Ciaccia, M. Patella, P. Zezula, "*A Cost Model for Similarity Queries in Metric Spaces*," presented at ACM Symposium on Principles of Database Systems (PODS), Seattle, Washington, 1998, pp. 59-68.

[Codd_1970] E. F. Codd, "*A Relational Model of Data for Large Shared Data Banks.*," *Communications of the ACM (CACM)*, vol. 13,June, 1970, pp. 377-387.

[Comer_1979] D. Comer, "*The Ubiquitous B-Tree*," *ACM Computing Surveys*, vol. 11,June 1979, pp. 121-137.

[DeWitt_1991] D. DeWitt, "The Wisconsin Benchmark: Past, Present, and Future," in *The Benchmark Handbook for Database and Transaction Processing Systems*, vol. 1, J. Gray, Ed.: Morgan Kaufmann, 1991.

[Eisenberg_1999] A. Eisenberg and J. Melton, "*SQL-1999, Formerly known as SQL3*," *ACM SIGMOD Records*, vol. 28,March 1999, pp. 131-138.

[Eisenberg_2004] A. Eisenberg, J. Melton, K. Kulkarni, J.-E. Michels, F. Zemke, "*SQL:2003 Has Been Published*," *ACM SIGMOD Records*, vol. 33,March 2004, pp. 119-126.

[Elmasri_2003] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, Fourth ed: Addison-Wesley, 2003.

[Faloutsos_1987] C. Faloutsos, T. Sellis, N. Roussopoulos, "*Analisys of Object Oriented Spatial Acess Methods*," presented at ACM International Conference on Management of Data (SIGMOD), San Francisco, CA, 1987, pp. 426-439.

[Faloutsos_1994] C. Faloutsos and I. Kamel, "*Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension*," presented at ACM Symposium on Principles of Database Systems (PODS), Minneapolis, MN, 1994, pp. 4-13.

[Friedman_1977] J. H. Friedman, J. H. Bentley, R. A. Finkel, "*An Algorithm for Finding Best Matches in Logarithmic Expected Time*," *ACM Transactions on Mathematical Software*, vol. 3,1977, pp. 209-226.

[Gaede_1998] V. Gaede and O. Günther, "*Multidimensional Access Methods*," *ACM Computing Surveys*, vol. 30,June 1998, pp. 170-231.

[Gao_2005] L. Gao and X. S. Wang, "*Continuous Similarity-Based Queries on Streaming Time Series*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 17,October 2005, pp. 1320-1332.

[Gunopulos_2005] D. Gunopulos, G. Kollios, V. J. Tsotras, C. Domeniconi, "*Selectivity estimators for multidimensional range queries over real attributes*," *The International Journal on Very Large Databases*, vol. 14,April 2005, pp. 137 - 154.

[Guttman_1984] A. Guttman, "*R-Tree : A dynamic Index Structure for Spatial Searching*," presented at ACM SIGMOD International Conference on Management of Data, Boston, MA, 1984, pp. 47-57.

[Hjaltason_2003] G. R. Hjaltason and H. Samet, "*Index-driven similarity search in metric spaces*," *ACM Transactions on Database Systems (TODS)*, vol. 21,December 2003, pp. 517 - 580.

[IBM Corporation_2006] IBM Corporation, *IBM DB2 9 Manuals - Developing SQL and External Routines*. Armonk, NY: IBM, 2006.

[Katayama_1997] N. Katayama and S. Satoh, "*The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries*," presented at ACM SIGMOD International Conference on Management of Data, Tucson, Arizona, USA, May 13-15, 1997, pp. 369-380.

[Korn_2001] F. Korn, B.-U. Pagel, C. Faloutsos, "*On the 'Dimensionality Curse' and the 'Self-Similarity Blessing'*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 13,January/February 2001, pp. 96-111.

[Kruskal_1956] J. B. Kruskal, "*On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem*," *Proceedings of the American Mathematical Society*, vol. 7,1956, pp. 48-50.

[Lew_2006] M. S. Lew, N. Sebe, C. Djeraba, R. Jain, "*Content-Based Multimedia Information Retrieval: State of the Art and Challenges*," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 2,February 2006, pp. 1-19.

[Lin_1994] K.-I. D. Lin, H. V. Jagadish, C. Faloutsos, "*The TV-Tree: An Index Structure for High-Dimensional Data*," *The International Journal on Very Large Databases*, vol. 3,October 1994, pp. 517-542.

[Müller_2004] H. Müller, N. Michoux, D. Bandon, A. Geissbuhler, "*A Review of Content-based Image Retrieval Systems in Medical Applications-Clinical Benefits and Future Directions*," *International Journal of Medical Informatics (IJMI)*, vol. 73,February 2004, pp. 1-23.

[O'Neil_2001] P. O'Neil and E. O'Neil, *Database - Principles, Programming and Performance*, 2nd ed. San Francisco, CA: Morgan Kaufmann Publishers, 2001.

[Oracle Corporation_2005] Oracle Corporation, *Oracle Database Application Developer's Guide - Object-Relational Features 10g Release 2 (10.2)*. Redwood Shores, CA: Oracle, 2005.

[Robinson_1981] J. T. Robinson, "*The K-D-B-Tree: A Search Structure For Large Multidimensional Dynamic Indexes*," presented at ACM SIGMOD International Conference on Management of Data, pp. 10-18.

[Samet_2006] H. Samet, *Foundations of Multidimensional and Metric Data Structures*. San Francisco, CA: Morgan Kaufmann, 2006.

[Santos Filho_2001] R. F. Santos Filho, A. J. M. Traina, C. Traina Jr., C. Faloutsos, "*Similarity Search without Tears: The OMNI Family of All-purpose Access Methods*," presented at IEEE International Conference on Data Engineering (ICDE), Heidelberg, Germany, April 2-6, 2001, pp. 623-630.

[Santos Filho_2003] R. F. Santos Filho, "*Métodos de Acesso Métricos para suporte a consultas por similaridade: Apresentação da Técnica Omni*," in *Departamento de Ciências de Computação*. São Carlos, SP: Universidade de São Paulo, pp. 141.

[Schroeder_1991] M. Schroeder, *Fractals, Chaos, Power Laws*, 6 ed. New York: W. H. Freeman, 1991.

[Sellis_1987] T. K. Sellis, N. Roussopoulos, C. Faloutsos, "*The R+-tree: A Dynamic Index for Multi-dimensional Objects*," presented at International Conference on Very Large Databases (VLDB), Brighton, England, September 1-4, 1987, pp. 507-518.

[Smeulders_2000] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, "*Content-Based Image Retrieval at the End of the Early Years*," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22,December 2000, pp. 1349-1380.

[Stonebraker_1986] M. Stonebraker, "*Inclusion of New Types in Relational Data Base Systems*," presented at IEEE International Conference on Data Engineering (ICDE), Los Angeles, CA, February 5-7, 1986, pp. 262-269.

[Tao_2004] Y. Tao, J. Zhang, D. Papadias, N. Mamoulis, "*An Efficient Cost Model for Optimization of Nearest Neighbor Search in Low and Medium Dimensional Spaces*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 16,October 2004, pp. 1169-1184.

[Traina_2002] A. J. M. Traina, C. Traina Jr., J. M. Bueno, P. M. d. A. Marques, "*The Metric Histogram: A New and Efficient Approach for Content-based Image Retrieval*," presented at Sixth IFIP Working Conference on Visual Database Systems, Brisbane, Australia, May 29-31, 2002, pp. 297-311.

[Traina_2003] A. J. M. Traina, C. Traina Jr., J. M. Bueno, F. J. T. Chino, P. M. d. A. Marques, "*Efficient Content-based Image Retrieval through Metric Histograms*," *World Wide Web Journal (WWWJ)*, vol. 6,June 2003, pp. 157-185.

[Traina_2004] A. J. M. Traina, A. G. R. Balan, L. M. Bortolotti, C. Traina Jr., "*Content-based Image Retrieval Using Approximate Shape of Objects*," presented at 17th IEEE Symposium on Computer-Based Medical Systems (CBMS´04), Bethesda, Maryland, June 24-25, 2004, pp. 91-96.

[Traina Jr._1999] C. Traina Jr., A. J. M. Traina, C. Faloutsos, "*Distance exponent : a new concept for selectivity estimation in metric trees*," Carnegie Mellon University - School of Computer Science, Pittsburgh-PA USA, Research Paper CMU-CS-99-110, March 1999, pp. 15.

[Traina Jr._2000a] C. Traina Jr., A. J. M. Traina, C. Faloutsos, "*Distance Exponent: a New Concept for Selectivity Estimation in Metric Trees*," presented at IEEE International Conference on Data Engineering (ICDE), San Diego - CA, February 28 - March 3, 2000, pp. 195.

[Traina Jr._2000b] C. Traina Jr., A. J. M. Traina, B. Seeger, C. Faloutsos, "*Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes*," presented at International

Conference on Extending Database Technology (EDBT), Konstanz, Germany, March 27-31, 2000, pp. 51-65.

[Traina Jr._2000c] C. Traina Jr., A. J. M. Traina, L. Wu, C. Faloutsos, "*Fast feature selection using fractal dimension*," presented at Brazilian Symposium on Databases (SBBD), João Pessoa, PB, october 2-4, 2000, pp. 158-171.

[Traina Jr._2002a] C. Traina Jr., A. J. M. Traina, C. Faloutsos, B. Seeger, "*Fast Indexing and Visualization of Metric Datasets Using Slim-trees*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 14,March/April 2002, pp. 244-260.

[Traina Jr._2002b] C. Traina Jr., A. J. M. Traina, R. F. Santos Filho, C. Faloutsos, "*How to Improve the Pruning Ability of Dynamic Metric Access Methods*," presented at International Conference on Information and Knowledge Management (CIKM), McLean, VA, USA, November 4-9, 2002, pp. 219-226.

[Traina Jr._2005] C. Traina Jr., R. F. Santos Filho, A. J. M. Traina, M. R. Vieira, C. Faloutsos, "*The OMNI-Family of All-Purpose Access Methods: A Simple and Effective Way to Make Similarity Search More Efficient*," *The International Journal on Very Large Databases*, vol. no prelo,2006.

[Traina Jr._2006] C. Traina Jr., A. J. M. Traina, M. R. Vieira, A. S. Arantes, C. Faloutsos, "*Efficient Processing of Complex Similarity Queries in RDBMS through Query Rewriting*," presented at ACM 15th International Conference on Information and Knowledge Management (CIKM'06), Arlington - VA, 6-11 de novembro, 2006, pp. 4-13.

[Vieira_2004] M. R. Vieira, C. Traina Jr., A. J. M. Traina, F. J. T. Chino, "*DBM-Tree: A Dynamic Metric Access Method Sensitive to Local Density Data*," presented at Brazilian Symposium on Databases (SBBD), Brasília, DF, October18-21, 2004, pp. 33-47.

[Weber_1998] R. Weber, H.-J. Schek, S. Blott, "*A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces*," presented at International Conference on Very Large Databases (VLDB), New York City, NY, August 24-27, 1998, pp. 194-205.

[Ye_2005] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, V. Kumar, "*IDR/QR: An Incremental Dimension Reduction Algorithm via QR Decomposition*," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 17,September 2005, pp. 1208-1222.