Explorando o uso de árvores B+ na Indexação de Dados por Similaridade

Jéssica N. B. de Farias[1], **Maria Camila N. Barioni**[1], **Humberto L. Razente**[1]

[1]Faculdade de Computação - Universidade Federal de Uberlândia (FACOM/UFU)

{jessica.farias,camila.barioni,humberto.razente}@ufu.br

Abstract. *This paper presents a new metric indexing method called GroupSim+ which allows narrowing the minimal cut-regions, when compared with related works. This characteristic of the method contributes to the optimization of similarity queries. The experiments performed with different datasets demonstrate the effectiveness of the approach adopted in the development of GroupSim+.*

Resumo. *Este artigo apresenta um novo método de indexação métrico denominado GroupSim+ que, quando comparado com trabalhos correlatos, permite um maior estreitamento das regiões mínimas de poda. Essa característica do método contribui para a otimização de consultas por similaridade. Os resultados dos experimentos realizados com diferentes conjuntos de dados reais demonstram a eficácia da abordagem adotada no desenvolvimento do GroupSim+.*

1. Introdução

Nas últimas décadas, o aumento do volume e da complexidade dos dados disponíveis para análise tem afetado o modo como os Sistemas de Gerenciamento de Bancos de Dados (SGBD) manipulam os dados. Esse fato gerou a necessidade do desenvolvimento de estruturas de indexação e operações de consulta próprias para atender essa mudança de paradigma. Dentro desse contexto, a otimização de consultas por similaridade tem sido uma questão de pesquisa importante na área de Bancos de Dados.

A otimização de consultas por similaridade, como as consultas por abrangência e aos k-vizinhos mais próximos, é normalmente realizada por meio da utilização de estruturas de indexação conhecidas como Métodos de Acesso Métricos (MAM). Nesses métodos, as comparações por similaridade requerem que os domínios de dados sejam representados em espaços métricos. Um espaço métrico é definido por um par $\langle \mathbb{S}, \delta() \rangle$, onde $\mathbb{S}$ é o domínio de dados e $\delta()$ é uma função de distância $\delta : \mathbb{S} \times \mathbb{S} \rightarrow R^+$ que obedece as propriedades de simetria, não-negatividade e desigualdade triangular. Essa última propriedade é especialmente importante para os MAM uma vez que permite podar regiões do espaço que certamente não contêm elementos que atendem as condições de uma consulta. Uma extensa revisão da área pode ser obtida em [Samet 2006].

Mapeamentos unidimensionais de dados têm sido empregados no desenvolvimento de MAM interessantes. A ideia básica desses métodos consiste em ordenar os elementos de dados considerando as distâncias em relação a um elemento de referência (pivô), como ilustrado na Figura 1. Isso permite a utilização de estruturas dinâmicas baseadas na relação de ordem total, como a árvore $B+$, para indexação dos dados. Exemplos de MAM baseados nessa estratégia são: *OmniB-Forest* [Traina-Jr et al. 2007], *iDistance* [Jagadish et al. 2005] e *GroupSim* [Razente et al. 2017]. A otimização das consultas por

similaridade de modo exato nesses métodos é baseada na propriedade de desigualdade triangular considerando cada pivô, o elemento de consulta e o raio de abrangência dado.

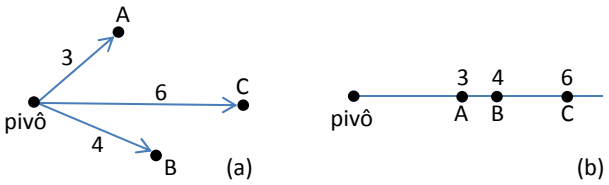

Figura 1. Pontos de um espaço bi-dimensional imersos em um mapeamento unidimensional. (a) Pontos 2D originais. (b) Mapeamento.

A principal característica do método *OmniB-Forest* é o emprego de uma árvore $B+$ para cada mapeamento unidimensional. O conjunto candidato de uma busca por abrangência nesse método é a intersecção dos anéis formados com base no elemento de consulta, o raio e o conjunto de pivôs. Já o método *iDistance* particiona o conjunto de dados considerando o conjunto de pivôs, armazenando-o em uma mesma árvore $B+$, por meio de uma constante para separar as partições. O método *GroupSim* emprega uma árvore $B+$ para indexação de um mapeamento unidimensional, entretanto, agrega as distâncias para os demais mapeamentos unidimensionais em cada entrada da estrutura, permitindo a computação de regiões mínimas de poda com um custo inferior aos outros métodos.

O trabalho apresentado nesse artigo visa contribuir para a otimização de consultas por similaridade por meio da exploração de uma nova abordagem baseada na utilização de árvores $B+$ na construção de índices dinâmicos. A abordagem adotada na proposta do novo MAM, denominado *GroupSim+*, explorou desvantagens identificadas nos MAM correlatos. Além disso, algoritmos eficientes de consulta por abrangência e aos k-vizinhos mais próximos também são propostos. Os experimentos iniciais ajudam a corroborar as hipóteses levantadas e a direcionar a realização de trabalhos futuros. O restante do artigo está organizado da seguinte forma. A Seção 2 descreve o problema e o novo MAM *GroupSim+*. A discussão a respeito dos resultados experimentais é apresentada na Seção 3. As conclusões e os trabalhos futuros são descritos na Seção 4.

2. O método GroupSim+

No método *OmniB-Forest* é necessário coletar os identificadores dos elementos até atingir o raio da consulta em todas as árvores $B+$ e então computar as intersecções entre essas regiões de abrangência enquanto no *iDistance* a poda é limitada apenas pelo anel do pivô de cada partição. Para tratar essas questões, o método *GroupSim* emprega uma árvore $B+$ para indexação das distâncias dos elementos em relação a um pivô e armazena um vetor de distâncias para os demais pivôs permitindo o descarte de identificadores pela região mínima de poda com custo menor. Entretanto, ao indexar os elementos de dados em relação a apenas um pivô, os anéis formados por um raio em uma consulta podem abranger um volume grande de identificadores. A hipótese para o desenvolvimento do método *GroupSim+* é que a combinação do particionamento do espaço com o armazenamento do vetor de distâncias para os demais pivôs resultará no estreitamento das regiões mínimas de poda em relação ao método *GroupSim*.

O *GroupSim+* é implementado por meio da extensão da árvore B+ de modo que os nós folha armazenem as seguintes entradas: $< chave, id, info[\,] >$, onde a ordenação

é dada pela $chave$, representada pela distância de um elemento ao pivô mais próximo, id é um identificador para o elemento e $info[\,]$ é um vetor de distâncias do elemento aos demais pivôs. Para a indexação de um conjunto dinâmico de elementos, seleciona-se inicialmente um conjunto estático de pivôs. A indexação de um elemento é feita da seguinte maneira. O elemento é inserido no final do arquivo de acesso aleatório (ORAF) e as distâncias para o conjunto de pivôs são calculadas. O particionamento dos dados na árvore $B+$ é dado pela adição de uma constante c multiplicada pelo número da partição à distância do elemento em relação ao pivô mais próximo, resultando na $chave$ de indexação. O id é a posição do elemento no ORAF e as distâncias para os demais pivôs são armazenadas em $info[\,]$. A constante c deve ser suficientemente grande para que não haja sobreposição entre os intervalos de diferentes partições.

Uma consulta por abrangência recebe como entrada um elemento de consulta q e um raio r e retorna uma lista de elementos que estão a até r de distância de q. No método *GroupSim+*, inicia-se procurando na árvore $B+$ pela chave mais próxima de $\delta(q, p_i) - r$ em cada partição i representada pelo pivô p_i. Todas as entradas com chaves no intervalo $\delta(q, p_i) - r$ a $\delta(q, p_i) + r$ devem ser verificadas (região em amarelo apresentada na Figura 2a), sendo que as distâncias armazenadas em $info[\,]$ permitem o descarte das entradas que certamente não fazem parte do conjunto resposta pela propriedade da desigualdade triangular (região em amarelo da Figura 2b), evitando assim a recuperação dos elementos do ORAF e a computação das distâncias em relação a q. As partições não cobertas pelo raio de consulta não precisam ser verificadas.

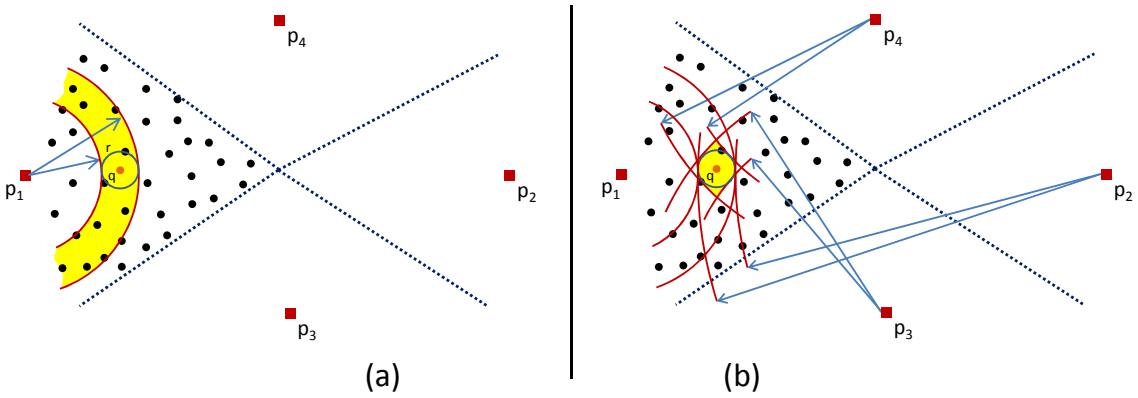

Figura 2. Região mínima de poda no *GroupSim+* para a partição representada pelo pivô p_1 com base no elemento de consulta q e raio r. (a) Anel restringe entradas da árvore $B+$ que precisam ser verificadas. (b) Distâncias armazenadas em $info[\,]$ permitem evitar a recuperação dos elementos do ORAF que não estiverem na região mínima de poda.

Uma consulta aos k-vizinhos mais próximos recebe como entrada um elemento de consulta q e um número k de elementos. Essa consulta retorna uma lista de até k elementos mais próximos, ordenados pela distância em relação a q. No método *GroupSim+*, inicia-se procurando na árvore $B+$ pela chave mais próxima de $\delta(q, p_i)$ na partição i representada pelo pivô p_i que contém o elemento de consulta q. A estratégia utilizada é a *start small and grow*, ou seja, uma constante Δ_r é utilizada como raio r inicial, verificam-se as entradas à esquerda e à direita até atingir o raio r, e caso decida-se continuar, adiciona-se Δ_r a r. Para as demais partições, se $\delta(q, p_i) - max_i \leq r$ na partição i representada pelo pivô p_i onde max_i é a maior distância entre os elementos da partição,

ou seja, se o raio da região de abrangência ultrapassa a partição que contém o elemento de consulta, devem ser verificadas as entradas em ordem decrecente a partir do limite max_i até $\delta(q, p_i) - r$. Para melhor desempenho, os nós folha da árvore $B+$ devem ser duplamente encadeados, para permitir percorrê-los tanto em ordem ascendente quanto descendente. Em todos os casos, as distâncias armazenadas em $info[\,]$ são utilizadas para verificar se cada elemento está dentro da região mínima de poda definida pela intersecção dos anéis obtidos a partir de cada pivô.

3. Experimentos

Os experimentos foram implementados[1] em C++ e executados em um computador equipado com CPU Intel Core i7-4770, 8 GB de memória principal e disco rígido de 1 TB. Os métodos foram avaliados em relação ao desempenho da realização de consultas por similaridade baseadas em raio e k-vizinhos mais próximos utilizando distância Euclidiana. Os conjuntos de pivôs foram selecionados pela heurística *MaxSum* [Socorro et al. 2011]. Os resultados para os seguintes conjuntos de dados são apresentados: Pendigits (10.992 elementos, 16 dimensões) e Covertype (581.012 elementos, 10 dimensões)[2]; Nasa (40.150 elementos, 20 dimensões) e Colors (112.682 elementos, 112 dimensões)[3].

O objetivo é avaliar o desempenho da nova estrutura em relação aos trabalhos correlatos considerando os seguintes critérios: o tempo, cálculos de distância e acessos ao disco. Para todos os métodos avaliados, a inclusão de um elemento envolve o cálculo das distâncias do elemento para cada elemento do conjunto de pivôs, a inclusão no final do arquivo ORAF, e a inclusão da entrada nas árvores $B+$. Assim, os métodos foram construídos com tempos de execução aproximadamente iguais. Foram selecionados aleatoriamente 100 elementos de cada conjunto de dados para realização das consultas.

Para os experimentos apresentados nas Figuras 3 e 4 os índices foram criados com a utilização de 6 pivôs. Na Figura 3 as consultas foram executadas com um raio variando de 1% a 10% da maior distância calculada entre todos os pares de elementos de cada conjunto (eixo das abcissas). Analisando esses gráficos é possível notar que considerando o número de acessos a disco, o método *iDistance* apresenta número de acessos inferior ao do *GroupSim+*. Entretanto, os métodos *OmniB-Forest*, *GroupSim* e *GroupSim+* apresentam comportamento similar, resultando em quantidades inferiores de cálculos de distância quando comparados com o *iDistance*. Isso ocorre devido ao fato que esses métodos utilizam todos os pivôs para definição da região mínima de poda enquanto o *iDistance* utiliza apenas um pivô em cada partição. Com relação ao tempo de execução das consultas, o *GroupSim+* apresentou redução de 41% para o raio $r = 1\%$ e 47% para o raio $r = 10\%$ em relação do *GroupSim* no conjunto Nasa.

Na Figura 4 as consultas foram executadas com $k = 10$ até 100 (eixo das abcissas). De maneira geral, o comportamento de todos os métodos na realização de consultas aos k-vizinhos mais próximos foi semelhante ao comportamento observado nas consultas por abrangência. Isso ocorre devido ao fato de que a otimização nesses algoritmos é realizada pela convergência da região de abrangência (definida por um raio) que contém k elementos.

[1] Biblioteca Arboretum: https://bitbucket.org/gbdi/arboretum

[2] UCI Machine Learning Repository: http://archive.ics.uci.edu/ml

[3] Metric Spaces Library: http://www.sisap.org/metricspaceslibrary.html

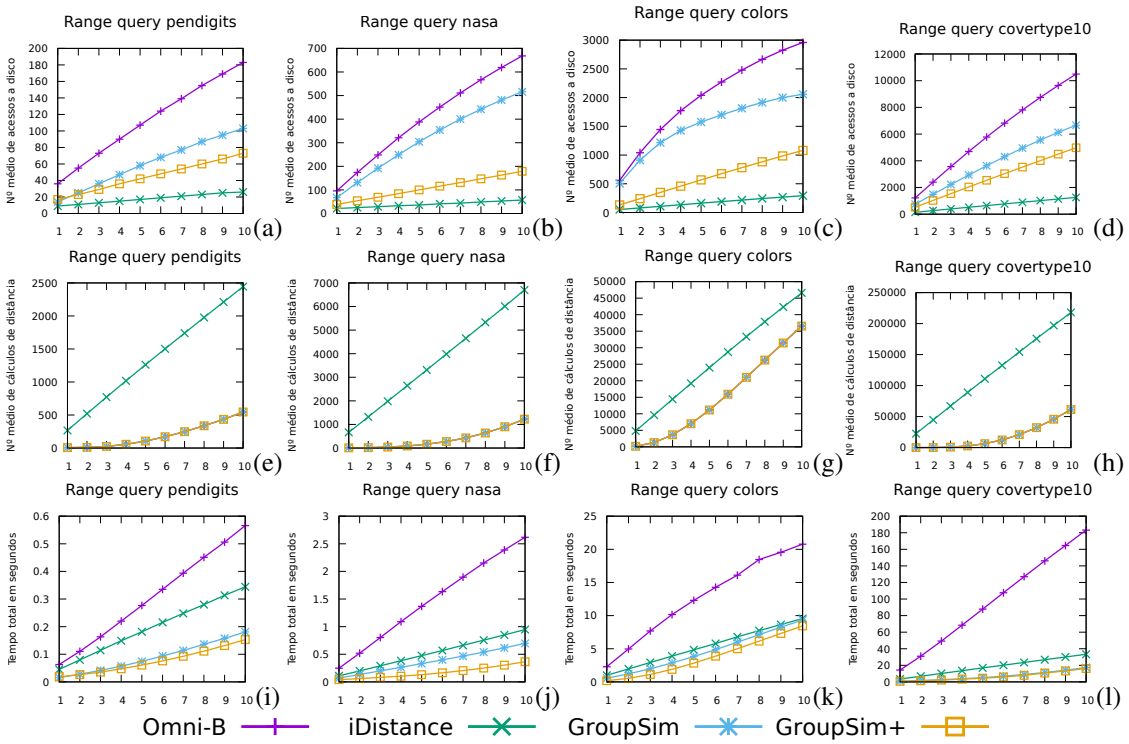

Figura 3. Consultas por abrangência. (a-d) acessos a disco. (e-h) cálculos de distância. (i-l) tempo total.

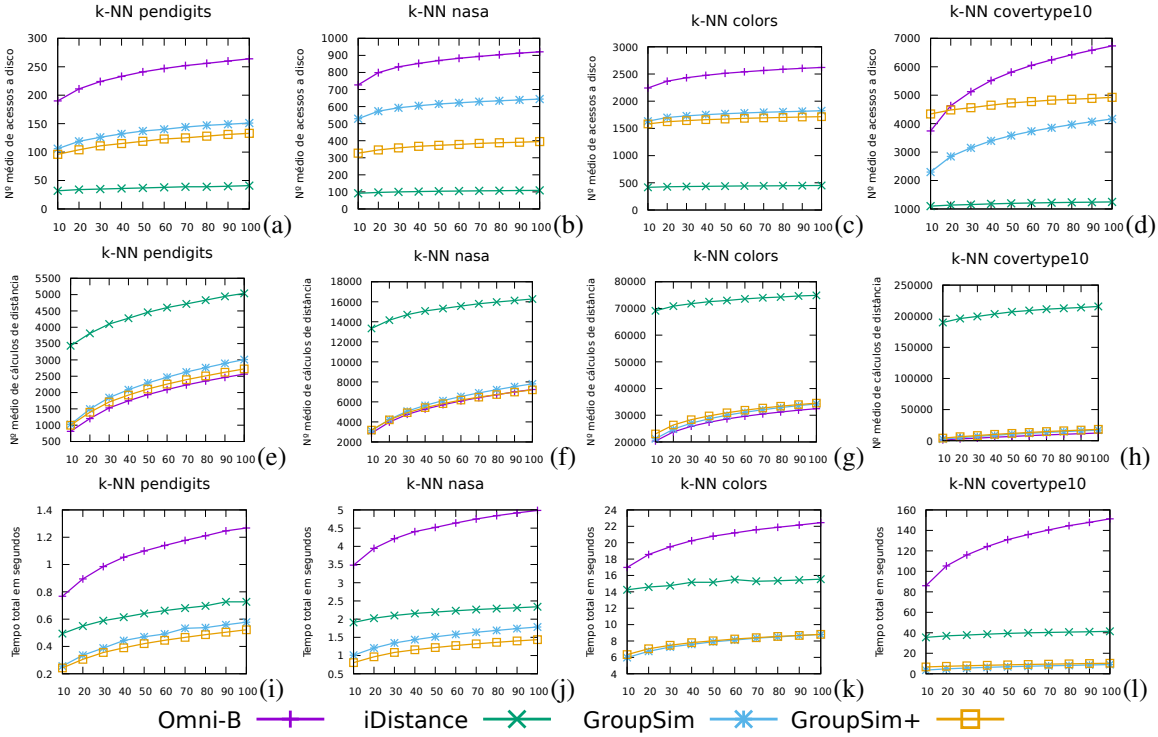

Figura 4. Consultas aos k-vizinhos mais próximos. (a-d) acessos a disco. (e-h) cálculos de distância. (i-l) tempo total.

A Figura 5 apresenta os resultados para consultas aos 10-vizinhos mais próximos executadas em índices construídos com 3 a 15 pivôs (eixo das abcissas). O aumento no número de pivôs impactou negativamente no tempo de execução do método *OmniB-Forest* para os conjuntos de dados testados, permitindo observar que a partir de 4 pivôs não houve diminuição nas regiões mínimas de poda e a inclusão de pivôs aumentou linearmente o tempo para computação da intersecção dos anéis. Os métodos *GroupSim* e *GroupSim+* apresentaram comportamento praticamente constante no tempo de execução para todos os conjuntos de pivôs testados. Apesar dos gráficos indicarem uma tendência de melhora no tempo de execução do *iDistance* com o aumento do número de pivôs, a inclusão de novos elementos em um cenário dinâmico pode reverter essa tendência. Pretende-se investigar essa hipótese em trabalho futuro.

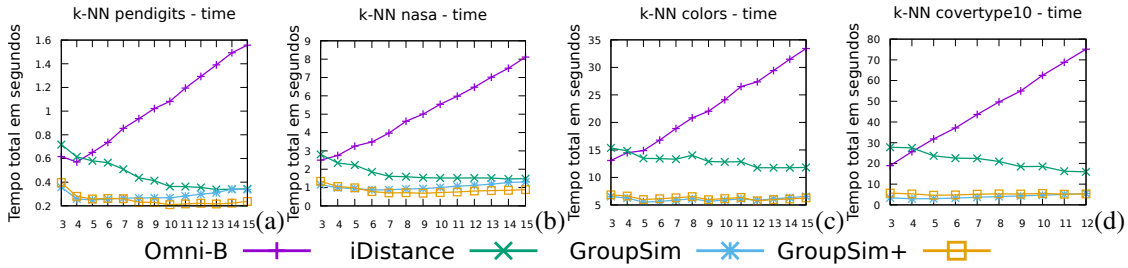

Figura 5. Consultas aos 10-vizinhos mais próximos executadas em índices construídos com 3 a 15 pivôs. (a-d) tempo total.

4. Conclusões

O *GroupSim+* considera as contribuições existentes no contexto de indexação de dados complexos utilizando mapeamentos unidimensionais e define um novo método de acesso métrico que visa aumentar o desempenho da recuperação por similaridade de elementos de dados por meio do estreitamento das regiões de busca e diminuição do número de cálculos de distância, a um mesmo custo de indexação. Como trabalho futuro, pretende-se explorar uma estratégia para redução dos acessos a disco.

Agradecimentos

Os autores agradecem ao CNPQ e a CAPES pelo apoio financeiro a este trabalho.

Referências

Jagadish, H., Ooi, B., Tan, K., Yu, C., and Zhang, R. (2005). iDistance: an adaptive b+tree based indexing method for nearest neighbor search. *ACM TODS*, 30(2):364–397.

Razente, H., Lima, R. B., and Barioni, M. C. (2017). Similarity search through one-dimensional embeddings. In *ACM SAC*, pages 874–879, Marrakech, Marrocos.

Samet, H. (2006). *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, San Francisco.

Socorro, R., Mico, L., and Oncina, J. (2011). A fast pivot-based indexing algorithm for metric spaces. *Pattern Recognition Letters*, 32(11):1511–1516.

Traina-Jr, C., Filho, R. F., Traina, A., Vieira, M. R., and Faloutsos, C. (2007). The omni-family of all-purpose access methods: A simple and effective way to make similarity search more efficient. *The VLDB Journal*, 16(4):483–505.