Pontifícia Universidade Católica do Paraná

Pesquisa Árvore Multiway

TDE 2

Emily Pontes Fontana

Curitiba

2025

1. Estudo sobre árvores

Árvore B

A árvore B é uma estrutura de busca balanceada do tipo *multiway*, projetada especialmente para armazenamento em disco e indexação de grandes volumes de dados. Cada nó contém múltiplas chaves e pode ter vários filhos, garantindo que todas as folhas permaneçam no mesmo nível. As operações de busca, inserção e remoção possuem custo O(logn) em acessos ao disco, com balanceamento automático. Quando um nó fica cheio, ele é dividido (*split*) e a chave do meio é promovida ao nó pai. Essa estrutura é amplamente utilizada em bancos de dados e sistemas de arquivos, pois minimiza o número de acessos à memória secundária.

Árvore B+

A árvore B+ é uma variação da árvore B que separa a função de armazenamento da de indexação. Os nós internos contêm apenas chaves-guia que servem de índice, enquanto todas as chaves reais e dados são armazenados nas folhas. As folhas são interligadas por ponteiros, formando uma lista encadeada que permite varreduras sequenciais muito rápidas (tempo O(1) entre os blocos). Durante a inserção e remoção, os *splits* e *merges* ocorrem principalmente nas folhas, garantindo eficiência em consultas por intervalos. É o tipo mais usado em sistemas de banco de dados modernos por otimizar consultas sequenciais e por intervalo.

Contexto JAVA: Por ser voltada para o armazenamento externo, sua implementação em JAVA foca em otimizar a serialização e desserialização dos nós para classes que gerenciam o acesso a arquivos.

Árvore B*

A árvore B* é uma otimização da B+, voltada a melhorar a utilização do espaço e a reduzir divisões desnecessárias. Em vez de dividir imediatamente um nó cheio, ela tenta **redistribuir** as chaves entre nós irmãos com o auxílio do pai (técnica de **redistribuição local**), realizando a divisão apenas quando necessário. Isso mantém uma ocupação média dos nós mais alta (próxima de 66%), melhorando o desempenho em leituras e gravações. Sua estrutura mantém o balanceamento e a complexidade O(logn), mas com menor fragmentação e menos acessos ao disco. É utilizada em sistemas de arquivos e bancos de dados que exigem alta eficiência e menor desperdício de espaço.

Contexto JAVA: A lógica complexa de redistribuição exige uma gestão mais cuidadosa dos ponteiros e da lógica de persistência em disco, mas oferece um desempenho superior em cenários de alta taxa de inserção.

Árvore 2-3

A árvore 2-3 é uma forma simplificada de árvore B de ordem três, onde cada nó pode conter uma ou duas chaves e, consequentemente, dois ou três filhos. Todas as folhas estão no

mesmo nível, garantindo que a estrutura permaneça sempre perfeitamente balanceada. A inserção é feita descendo até uma folha e dividindo o nó quando necessário, promovendo a chave do meio. A remoção utiliza fusões ou redistribuições para preservar as propriedades da árvore. É uma estrutura clássica de estudo para entender balanceamento dinâmico em árvores de busca.

Contexto JAVA: É um modelo didático crucial para entender as árvores de balanceamento.

Árvore 2-3-4

A árvore 2-3-4 é uma extensão da árvore 2-3, permitindo que cada nó contenha de uma a três chaves e até quatro filhos. Essa flexibilidade reduz a altura da árvore e melhora o desempenho de inserções e buscas. Durante a inserção, nós cheios são divididos antes da descida para evitar a necessidade de retrocessos. A remoção segue a mesma lógica da árvore 2-3, usando fusões e redistribuições para manter o balanceamento.

Árvore Trie

A árvore Trie (ou *prefix tree*) é uma estrutura hierárquica voltada para o armazenamento e busca de strings ou sequências de caracteres. Cada aresta representa um caractere, e o caminho da raiz até um nó marcado como final de palavra forma uma palavra completa. A busca, inserção e remoção ocorrem em tempo proporcional ao comprimento da chave (O(m)), independente da quantidade total de palavras. É ideal para sistemas de autocompletar, correção ortográfica e compressão de prefixos. Embora ofereça excelente desempenho em consultas por prefixos, seu ponto fraco é o alto consumo de memória.

Árvore Patricia (variação da Trie)

A árvore Patricia (*Practical Algorithm To Retrieve Information Coded In Alphanumeric*) é uma versão compactada da Trie que elimina nós com apenas um filho. Caminhos lineares são fundidos em um único rótulo, reduzindo a profundidade e economizando espaço. Esse processo, chamado compressão de prefixos, melhora a eficiência de buscas sem alterar a lógica de navegação. É amplamente utilizada em sistemas de roteamento IP e em compressão de dicionários, pois combina alta velocidade de busca com baixo uso de memória. Sua operação é eficiente e prática para grandes volumes de dados.

Contexto JAVA: A implementação é mais complexa do que a Trie simples, exigindo a manipulação de bits e *substrings* para a compressão, mas é altamente valorizada em bibliotecas de redes.

Árvore R

A árvore R é uma estrutura especializada em indexação espacial e geográfica, voltada para objetos bidimensionais e tridimensionais. Cada nó armazena um conjunto de Retângulos Delimitadores Mínimos (*Minimum Bounding Rectangles – MBR*) que englobam os elementos ou subárvores abaixo dele. Buscas percorrem apenas os ramos cujos MBRs interceptam a região consultada, reduzindo acessos desnecessários. A inserção escolhe o nó que aumenta menos a área total do MBR, e divisões ocorrem conforme heurísticas de

sobreposição mínima. É amplamente usada em bancos de dados geográficos e sistemas de geoprocessamento.

Árvore R+

A árvore R+ é uma variação da R-Tree que elimina a sobreposição entre retângulos internos, permitindo que as regiões da árvore sejam mutuamente exclusivas. Para alcançar isso, objetos podem ser duplicados em nós diferentes, garantindo que cada consulta percorra apenas um caminho por objeto. Essa abordagem reduz falsos positivos e melhora o desempenho de consultas espaciais complexas. É especialmente eficiente em Sistemas de Informação Geográfica (SIG), modelagem 3D e indexação de imagens. O pequeno custo de duplicação é compensado por buscas mais rápidas e precisas.

Contexto JAVA: A lógica de duplicação exige que a classe objeto seja imutável ou que o processo de inserção gerencie cuidadosamente as cópias dos dados.

Árvore Quad (Quadtree)

A árvore Quad é usada para particionar recursivamente um espaço bidimensional em quatro quadrantes: nordeste, noroeste, sudeste e sudoeste. Cada nó representa uma região do espaço, e a subdivisão ocorre sempre que um quadrante ultrapassa um limite de capacidade ou contém um objeto. Essa estrutura é ideal para representar mapas, imagens ou áreas em simulações físicas. Variantes como *Point Quad* e *Region Quad* adaptam-se a diferentes tipos de dados espaciais. É amplamente aplicada em processamento de imagens, jogos eletrônicos, compressão espacial e detecção de colisões.

Contexto JAVA: A Quadtree é relativamente simples de implementar em JAVA, usando classes recursivas que definem as coordenadas de limite e um *array* ou lista para os quatro nós filhos. É uma escolha comum para estruturas de jogos 2D para otimizar a detecção de colisões.