UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”
Câmpus de Rio Claro

Relatório de estudo sobre grafos do tipo árvore

Grafos e Aplicações

Equipe:

André Luis Dias Nogueira
Felipe Melchior de Britto
Rafael Daiki Kaneko
Ryan Hideki Tadeo Guimarães
Vitor Marchini Rolisola

24/09/2024

Conteúdo

1 Resumo

Este trabalho tem como objetivo aplicar o algoritmo de Busca em Largura (BFS) em árvores binomiais geradas aleatoriamente, calculando a média das profundidades das ordenações geradas para todos os nós. O número de filhos por nó segue uma distribuição binomial com parâmetros n_{max} (número máximo de filhos) e p (probabilidade de ter filhos), sendo a geração de descendentes limitada pela profundidade d_{max}.

Realizamos experimentos variando os parâmetros n_{max}, p, e d_{max}, gerando ao menos 10 árvores aleatórias para cada combinação de valores. Os resultados obtidos foram apresentados em tabelas e gráficos, permitindo a análise das profundidades médias.

Um dos interesses do estudo era almejar a transição de fase esperada no modelo: para $n_{\text{max}} \cdot p < 1$, as árvores atingem no geral níveis sem descendentes enquanto para $n_{\text{max}} \cdot p > 1$, as árvores se expandem até o limite de profundidade d_{max}. Esse comportamente fornece uma visão mais detalhada sobre a estrutura de crescimento das árvores binomias conforme variamos seus parâmetros.

2 Introdução

Grafos são estruturas fundamentais em teoria dos grafos, utilizadas para modelar uma variedade de problemas em diferentes áreas, desde redes de computadores até genética.

Um grafo é uma estrutura matemática usada para modelar relações entre objetos de um conjunto. Ele é composto por dois conjuntos: um conjunto de vértices (ou nós) e um conjunto de arestas (ou arcos) que conectam esses vértices. Os vértices representam os objetos e as arestas representam as relações entre esses objetos. [1]

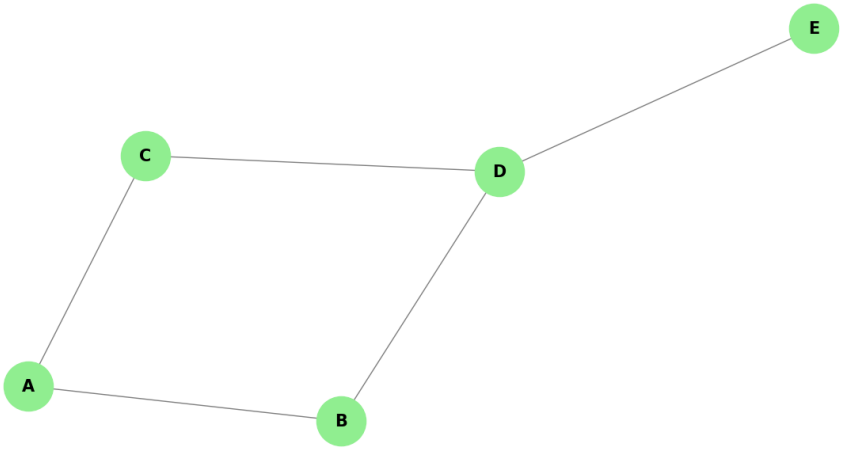

Figura 2.1: Exemplo de um grafo não orientado e sem peso

Por exemplo, na representação de uma rede social, os vértices podem representar pessoas e as arestas representam as conexões de amizade entre elas.

Um tipo especial de grafo, conhecido como árvore, apresenta propriedades únicas que tornam essa classe particularmente interessante para estudo.

Uma árvore é definida como um grafo não-orientado, conexo e acíclico, o que significa que não possui ciclos e, além disso, qualquer remoção de uma de suas arestas resulta em um grafo desconexo.[5] [2] Então suas características são:

- **Conectividade**: Para qualquer par de vértices u e v, existe exatamente um caminho que conecta u e v.
- **Aciclicidade**: O grafo não contém ciclos; ou seja; não é possivel iniciar em im vértice, seguir arestas e retornar ao mesmo vértice sem atravessar arestas repetidamente.
- **Número de arestas**: Se uma árvore possui n vértices, então ela possui exatamente $n - 1$ arestas.

Essas características permitem que árvores sejam a estrutura mínima necessária para

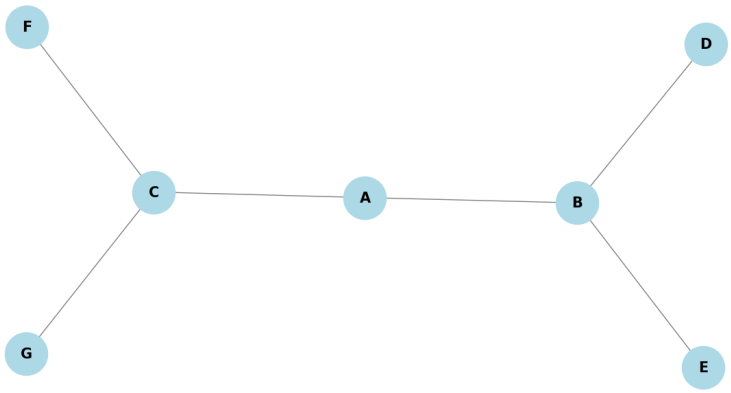

Figura 2.2: Exemplo de um grafo do tipo árvore

garantir a conectividade entre os vértices de um grafo com o menor número de arestas possíveis, um aspecto crucial para a otimização de recursos em diversos cenários práticos.

A análise de árvores em grafos tem implicações diretas em problemas de interligação, como o fornecimento de redes elétricas, onde o objetivo é minimizar o custo de conexão ao garantir que todas as unidades estejam conectadas. Além disso, árvores desempenham um papel importante na computação, particularmente em algoritmos de ordenação, como o Heapsort, e na modelagem de genealogias e redes hierárquicas.

Uma **árvore binomial** é uma estrutura de dados que representa uma coleção de árvores binomiais. A definição formal de uma árvore binomial é a seguinte[4] [3]:

Uma **árvore binomial** B_k é uma árvore que possui as seguintes propriedades:

- **Estrutura Recursiva**: Uma árvore binomial B_k é composta por 2^k nós e tem exatamente k árvores binomiais $B_{k-1}, B_{k-2}, \ldots, B_0$ como subárvores. A árvore B_k é obtida ao unir duas árvores B_{k-1}.
- **Propriedades dos Nós**:
  - O nó na raiz de B_k tem um grau de k (ou seja, ele possui k filhos).
  - A altura de B_k é k.
  - A árvore B_k possui 2^k folhas.
- **Organização dos Nós**: Os nós são organizados de tal forma que os valores dos nós na subárvore esquerda são menores ou iguais ao valor do nó pai, e os valores dos nós na subárvore direita são maiores.

As árvores binomiais são particularmente úteis em algoritmos de estrutura de dados, como em filas de prioridade.

O presente relatório tem o objetivo de explorar as propriedades matemáticas e aplicativas das árvores binomiais, abordando tanto sua definição formal quanto suas extensões, como arborescências e a aplicação em algoritmos de busca.

A introdução a essas ideias será contextualizada com base nas propriedades da conecti-

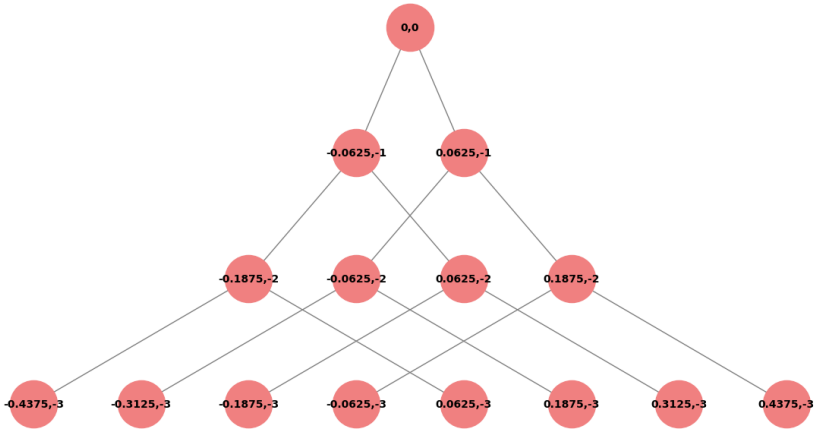

Figura 2.3: Grafo de uma árvore binomial

vidade e da aciclicidade, discutindo ainda como árvores binomiais podem ser vistas como estruturas mínimas e otimizadas para representação de relações complexas, ao mesmo tempo que mantêm a simplicidade computacional.

3 Implementação

Programa

A implementação do trabalho envolve a criação de um algoritmo que gera árvores binomiais com base nos parâmetros fornecidos: n_{max} (número máximo de filhos por nó), p (probabilidade de um nó gerar filhos) e d_{max} (profundidade máxima da árvore). Cada nó é representado por um conjunto de quatro posições em um vetor. A primeira posição do vetor, exclusivamente, armazena o número máximo de filhos do nó, enquanto as outras três posições armazenam o índice do pai e dos filhos, respectivamente, conforme o modelo que você está utilizando.

Para cada tripla de parâmetros, são geradas pelo menos 10 árvores aleatórias, utilizando a distribuição binomial para decidir quantos filhos cada nó terá. O algoritmo segue uma abordagem de Busca em Largura (BFS) para explorar os nós da árvore. A BFS percorre a árvore começando da raiz, explorando todos os nós em cada nível antes de passar para o próximo. Essa busca é gerenciada por uma fila que armazena os nós a serem explorados, garantindo que os nós sejam processados em ordem de nível.

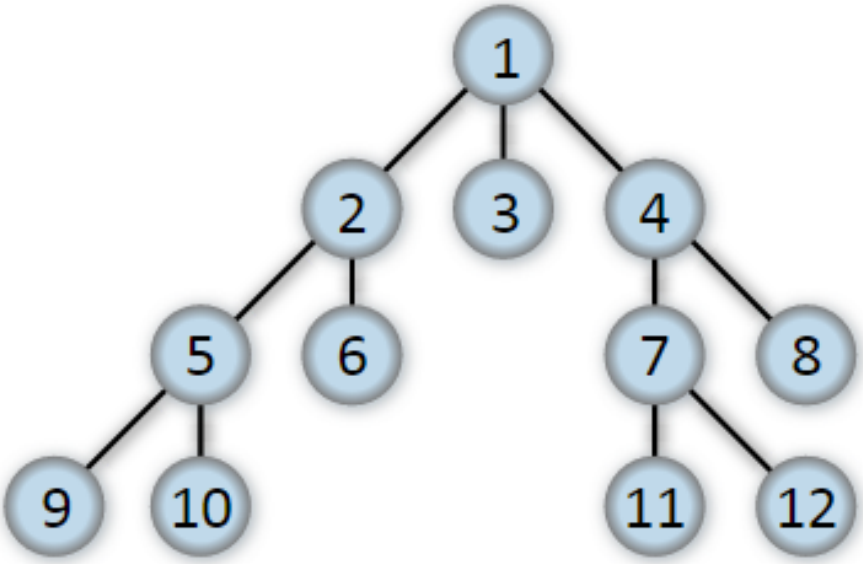

Figura 3.1: Exemplo da ordem de pesquisa do BFS

Durante a execução do BFS, são registrados a profundidade de cada nó e a profundidade total das árvores. Ao final de cada execução, é calculada a profundidade média da árvore, que será utilizada para gerar tabelas ou gráficos que mostram como essa profundidade varia de acordo com as diferentes combinações de n_{max}, p, e d_{max}.

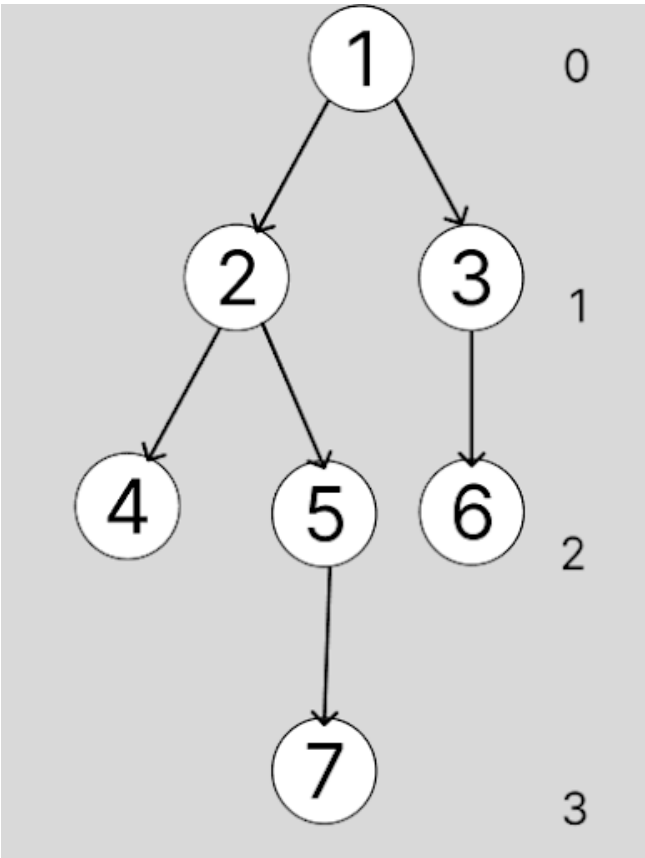

Figura 3.2: Exemplo do nível de profundidade do BFS

Além disso, o código precisa lidar com a transição de fase do modelo, que ocorre quando $n_{\text{max}} \cdot p < 1$, indicando que, em algum ponto, a geração de novos filhos para a árvore vai parar, resultando em níveis sem descendentes. Caso $n_{\text{max}} \cdot p > 1$, a geração de nós continuará até atingir o limite imposto por d_{max}.

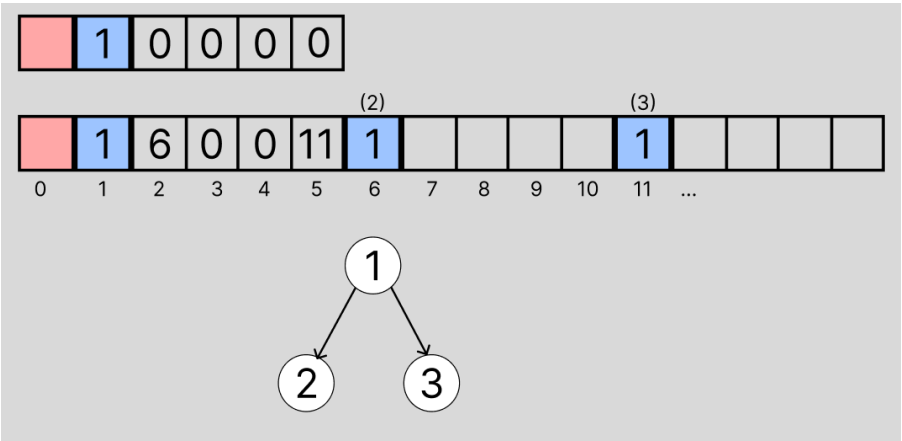

Figura 3.3: Estrutura da pilha do grafo

Portanto, a implementação é construída de maneira a não apenas gerar as árvores e calcular as profundidades, mas também simular a dinâmica de crescimento dessas árvores binomiais sob diferentes condições, explorando a variação desses parâmetros.

matemática

A árvore binomial é uma estrutura combinatória comumente usada em algoritmos e estruturas de dados, especialmente em contextos como heaps binomiais. Para compreender o uso de progressão geométrica e indução finita em árvores binomiais, é útil revisar o conceito básico dessas árvores e como essas duas ferramentas matemáticas são aplicadas nelas.

Progressão Geométrica

A progressão geométrica aparece na análise das árvores binomiais devido ao padrão de crescimento exponencial na quantidade de nós. Como mencionado, uma árvore binomial de ordem k possui 2^k nós. Ao observar diferentes ordens k, o número de nós em cada ordem segue uma progressão geométrica com razão 2:

- B_0 tem $1 = 2^0$ nó,
- B_1 tem $2 = 2^1$ nós,
- B_2 tem $4 = 2^2$ nós,

E assim por diante.

Se estivermos lidando com uma coleção de árvores binomiais (como em um heap binomial), o número total de nós segue a soma de uma progressão geométrica. Por exemplo, para uma coleção de árvores binomiais $B_0, B_1, \ldots, B_k$, o número total de nós seria:

$$S = 2^0 + 2^1 + 2^2 + \cdots + 2^k = 2^{k+1} - 1$$

Essa soma é a fórmula clássica da soma de uma progressão geométrica com razão 2.

Indução Finita

A indução finita (ou indução matemática) é uma técnica útil para provar propriedades sobre árvores binomiais, especialmente relacionadas ao número de nós, altura e estrutura recursiva. Ela é comumente usada para provar que uma árvore binomial de ordem k tem exatamente 2^k nós e altura k.

Exemplo de Prova por Indução:

Provar: Uma árvore binomial B_k tem 2^k nós.

1. Base da Indução: Para $k = 0$, B_0 é uma árvore com 1 nó, e $2^0 = 1$. Logo, a base está correta.

2. Hipótese de Indução: Suponha que para alguma ordem $k = n$, a árvore binomial B_n tem 2^n nós.

3. Passo de Indução: Mostre que a árvore binomial B_{n+1} tem 2^{n+1} nós. Pela definição de uma árvore binomial, B_{n+1} é composta por duas árvores B_n, conectadas por uma raiz. Cada B_n tem 2^n nós pela hipótese de indução. Assim, o número total de nós em B_{n+1} é:

$$2^n + 2^n = 2 \cdot 2^n = 2^{n+1}$$

Portanto, a propriedade é válida para $n + 1$, e a indução está concluída.

Então temos,

$$a_0 = 1$$

$$a_{n-1} = a_n \cdot (qtd.filho \cdot prob)^n$$

$$qtd_{filhos} = n_{max}$$

$$C = qtd_{filhos} \cdot prob$$

$$prob = p$$

$$profundidade = d_{max}$$

Usando Indução Finita:

$$a_n = C^{\frac{n(n+1)}{2}}$$

4 Resultados e discussão

Testes e resultados

Dentre os resultados, obtivemos diversos casos interessantes que nos ajudaram a compreender alguns aspectos da árvore binária implementada.

Dentre os testes, notamos a importância da variável P (probabilidade), tendo em vista a mudança que isso produz no resultado da média da profundidade

Outro grande aspecto, foi o tamanho da árvore, conforme aumentamos os parâmetros, ela cresce de forma muito rápida, de tal forma, que após certo ponto, não há como alocar memória suficiente.

Por fim, observamos que conforme o número de filhos aumenta, a árvore se torna mais esparsa.

5 Conclusão

Os resultados obtidos neste trabalho demonstram que, embora o problema das árvores binomiais apresente desafios consideráveis, fomos capazes de obter resultados positivos em situações que não extrapolavam os limites máximos dos parâmetros. Para combinações de n_{max}, p, e d_{max} que mantiveram o crescimento da árvore controlado, nosso algoritmo de Busca em Largura (BFS) foi bem-sucedido, fornecendo medições precisas das profundidades médias das árvores geradas.

No entanto, verificamos que, quando os parâmetros se aproximavam de seus valores máximos, o algoritmo encontrou dificuldades em lidar com a complexidade crescente, principalmente em cenários onde $n_{\text{max}} \cdot p > 1$, e a árvore se expandia rapidamente. Nessas situações, a BFS, que foi implementada para operar de maneira eficiente no caso médio, demonstrou limitações ao lidar com grandes quantidades de nós em profundidades elevadas, indicando a necessidade de otimizações para tratar casos extremos.

Apesar dessas dificuldades, nosso algoritmo se comportou de forma esperada em cenários moderados e foi capaz de capturar a dinâmica do modelo de árvores binomiais com precisão. As transições de fase observadas foram de acordo com as expectativas teóricas, e os resultados médios mostram correlação entre os parâmetros e a profundidade das árvores geradas. A implementação oferece uma base rica para futuros estudos e melhorias, especialmente no que tange a otimizações para lidar com grandes árvores e parâmetros no limite superior.

Bibliografia

[1] Emilio Bergamin Junior. *Aula 1 - Conceitos introdutórios de grafos*. 2024. URL: https://drive.google.com/file/d/1ZE6hkZ3LWcdHdRhw44ctXAguC0Iqbqzm/view.

[2] Emilio Bergamin Junior. *Aula 3 - Árvores*. 2024. URL: https://drive.google.com/file/d/1h2YFAkwwuRSOBUQLE6S-SH9_dJO_rktG/view.

[3] Daniel D. Sleator Robert E. Tarjan. ≪A data structure for dynamic trees≫. Em: *STOC '81: Proceedings of the thirteenth annual ACM symposium on Theory of computing* (1981). DOI: 10.1145/800076.

[4] Ronald L. Rivest e Clifford Stein Thomas H. Cormen Charles E. Leiserson. *Introduction to Algorithms*. First edition. MIT Press, 1990.

[5] Douglas B. West. *Introduction to Graph Theory*. Second edition. Pearson College Div, 2000.

www.de.ufpb.br/~tarciana/MPIE/Aula7.pdf
www.inf.ufsc.br/~andre.zibetti/probabilidade/aproximacao-binomial-poisson-pela-normal.html
www.scielo.br/j/pope/a/xyvK4fCGFR6g9s76W5FjJsB/#:~:text=Uma%20%C3%A1rvore%20binomial%20%C3%A9%20recombinante,por%20um%20movimento%20de%20subida

Fontes de imagens:
www.inf.ufsc.br/~andre.zibetti/probabilidade/aproximacao-binomial-poisson-pela-normal.html

Imagens da Introdução: Geradas com código no repositório:
https://github.com/FelipeMDB-UNESP/Grafos