Estrutura de Dados - 2º. período de 2017

Segunda Avaliação a Distância

1. (2,0) Sejam T uma árvore binária e $f(v)$ o número de nós pertencentes à subárvore de T cuja raiz é v. Descreva um algoritmo que imprima os nós de T em ordem não crescente de seus valores $f(v)$ (o nó raiz será o primeiro a ser impresso).

   Resposta: Seja V um vetor de tamanho n, tal que n é o número de nós de T. Cada $V[i]$ possui um ponteiro PT_i para uma lista encadeada L_i. Inicialmente $PT_i = \lambda$ para todo i. Cada elemento de L_i contém um ponteiro para o próximo elemento de L_i e um campo que guarda o identificador de um nó da árvore. Efetuamos um percurso na árvore (em pré-ordem, por exemplo) de modo que, ao visitarmos o nó v da árvore, inserimos um novo elemento na lista $L_{f(v)}$ que guarda o indicador do nó v. Ao final do percurso teremos cada nó de T representado em alguma lista encadeada de V. Finalmente percorremos todas as listas de V de L_n a L_1 imprimindo todos os elementos de L_i antes de qualquer elemento de L_j, para $j < i$.

   Como o percurso da árvore pode ser feito em tempo $\Theta(n)$, assim como o preenchimento das listas encadeadas e posteriormente os seus percursos, temos uma complexidade total igual a $\Theta(n)$.

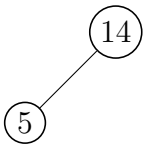
2. (1,0) Prove ou dê contra-exemplo: Dado n um inteiro positivo qualquer, sempre existe uma árvore AVL com exatamente n nós.

   Resposta: Para mostrar que a afirmação é verdadeira, basta mostrarmos uma árvore AVL T para cada valor de n. Tal árvore pode ser obtida tomando T como uma *árvore binária completa*, ou seja, aquela em que todo nó com alguma subárvore vazia pertence ou ao penúltimo ou ao último nível. Dessa forma podemos tomar T como uma árvore binária completa em que os nós são preenchidos da esquerda para a direita e que cada elemento de T é tal que todos os elementos da subárvore esquerda não são maiores que o elemento de v e que todos os elementos da subárvore direita não são menores que o elemento de v. Dessa forma T é uma árvore binária de busca em que a subárvore esquerda de cada elemento possui altura no máximo uma unidade maior que sua subárvore direita. Portanto T é uma árvore AVL.

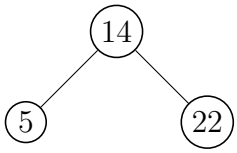
3. (1,5) A partir de uma árvore inicialmente vazia, desenhe a árvore AVL resultante da inserção dos nós com chaves $14, 5, 22, 3, 8, 7$ (nesta ordem).

   Resposta: A Figura 1 representa a sequência de inclusões e rotações necessárias para que a árvore obtida a cada etapa continue balanceada. O símbolo ($*$) é usado sobre um nó para indicar que o mesmo tornou-se desbalanceado.

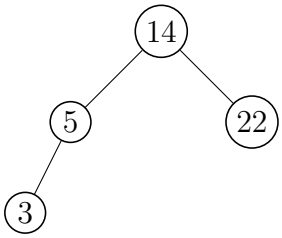
(a) Inclusão do elemento 14.



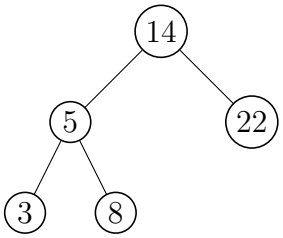
(b) Inclusão do elemento 16.



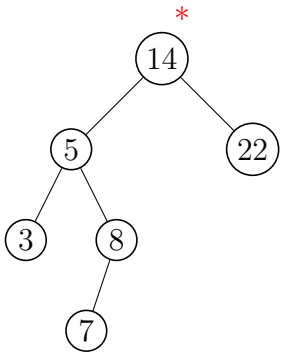
(c) Inclusão do elemento 22.



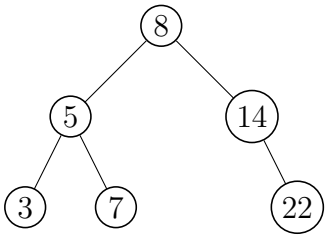
(d) Inclusão do elemento 3.



(e) Inclusão do elemento 8.



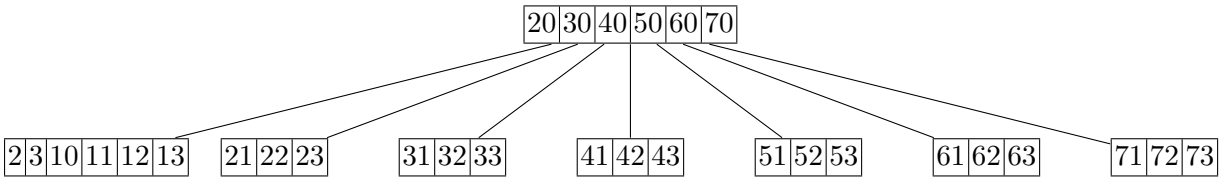
(f) Inclusão do elemento 7.



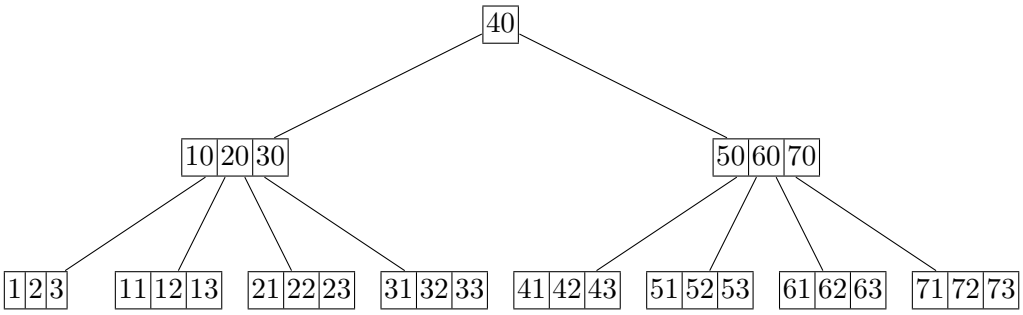
(g) Rotação dupla direita.

Figura 1: Sequência de inclusões para construção de uma árvore AVL.

4. (2,0) Desenhe uma árvore B de ordem $d = 3$ com dois níveis. (Os valores nos nós ficam à sua escolha.) A seguir, escolha uma nova chave de forma que a sua *inserção* exija uma cisão propagável. Desenhe a árvore B resultante após a inserção.



Após a inclusão do elemento 1:

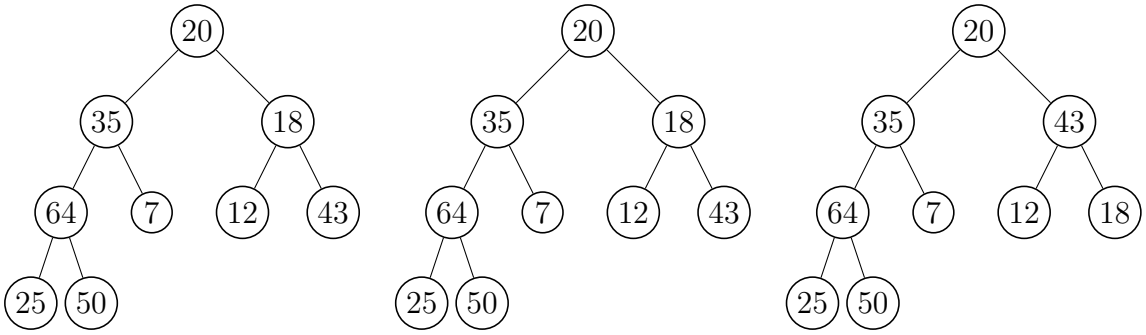


5. (2,0) Demonstre o passo a passo da aplicação do algoritmo de construção de heaps às seguintes prioridades: $20, 35, 18, 64, 07, 12, 43, 25, 50$.

   Resposta: A Figura 2 representa a sequência de passos na construção do heap.

6. (1,5) Desenhe o passo a passo do algoritmo de construção de uma árvore de Huffman para o seguinte conjunto de frequências: $f_1 = 1$, $f_2 = 1$, $f_3 = 2$, $f_4 = 4$, $f_5 = 8$, $f_6 = 16$. A árvore que você desenhou é a única possível?

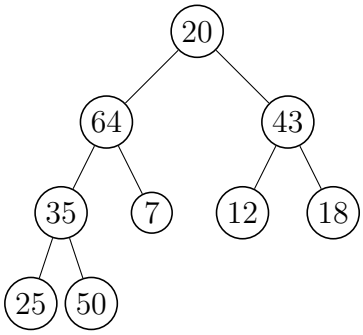
   Resposta: A Figura 3 representa a sequência de passos na construção da árvore de Huffman. Pelo algoritmo, deve-se escolher sempre os dois menores valores de raiz de modo a uni-las formando uma nova árvore a cada passo, cuja raiz possui valor igual a soma das raízes de suas duas subárvores que possuem os menores valores em suas raízes. Podemos ver que a cada passo há apenas uma escolha a ser feita dos dois menores valores de raízes, sendo os dois primeiros iguais a 1 para f_1 e f_2. A árvore gerada pela união dos mesmos possui raiz com valor 2 e, como apenas f_3 possui valor 2, apenas uma escolha de árvore pode ser feita neste caso. Em todos os outros a análise é análoga.



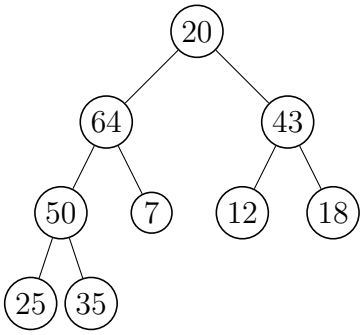
(a) Vetor inicial.

(b) Descer 64.

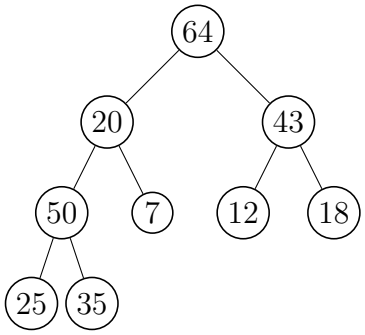
(c) Descer 18.



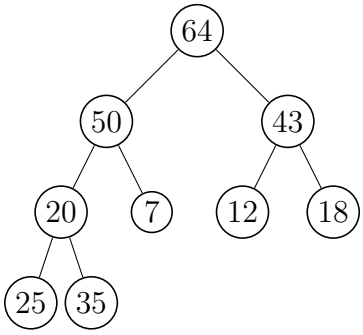
(d) Descer 35.



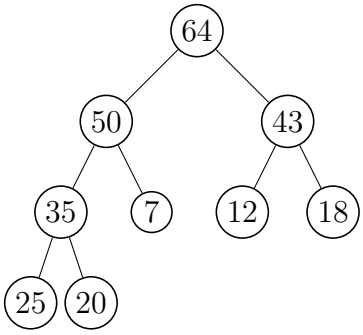
(e) Descer 35.



(f) Descer 20.



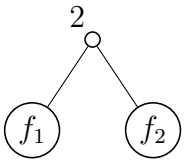
(g) Descer 20.



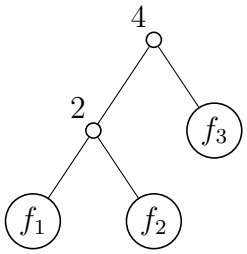
(h) Descer 20.

Figura 2: Sequência de operações para construção de um heap.

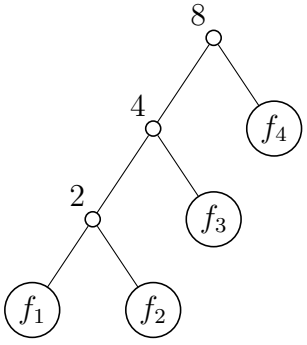
(a) Árvore com os primeiros 2 elementos.



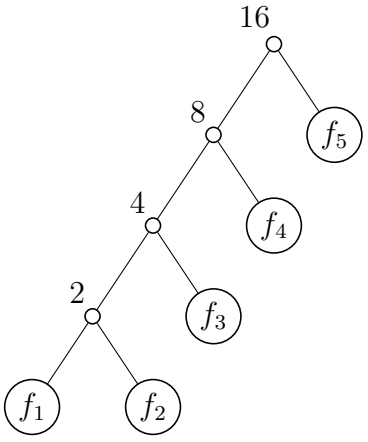
(b) Árvore com os primeiros 3 elementos.



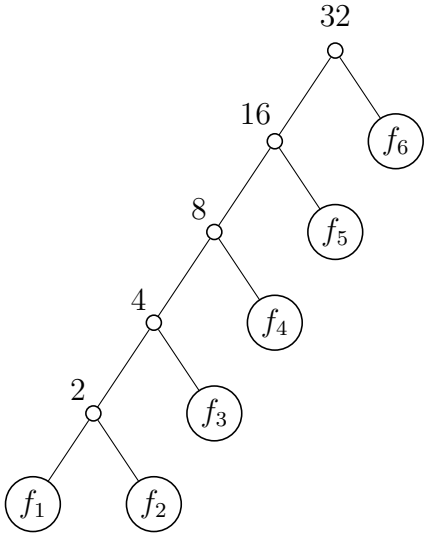
(c) Árvore com os primeiros 3 elementos.



(d) Árvore com os primeiros 4 elementos.



(e) Árvore com os primeiros 5 elementos.



(f) Árvore final.

Figura 3: Sequência de construção da árvore de Huffman.