Arquivos Sequenciais: Intercalação

Vanessa Braganholo

Baseado no Material de: Inhaúma Neves Ferraz (IC/UFF)

Cenário

- Diversos arquivos sequenciais ordenados
- Problema: gerar um único arquivo ordenado a partir dos vários arquivos de entrada

Cenário

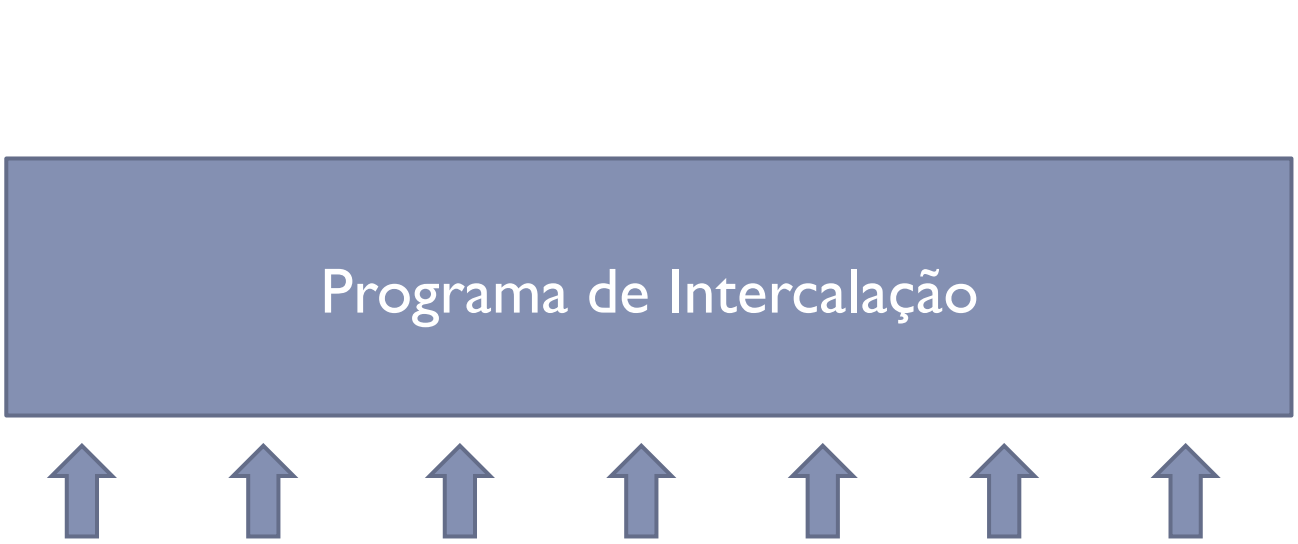
- Diversos arquivos sequenciais ordenados
- Problema: gerar um único arquivo ordenado a partir dos vários arquivos de entrada

- Como resolver este problema?
  - Discutam em grupo possíveis soluções para o problema

Algoritmo Básico

- De cada um dos arquivos a intercalar basta ter em memória um registro
- Considera-se cada arquivo como uma pilha
  - Topo da pilha: registro em memória
- Em cada iteração do algoritmo, o topo da pilha com menor chave é gravado no arquivo de saída e é substituído pelo seu sucessor
- Pilhas vazias têm topo igual a *high value*
- O algoritmo termina quando todos os topos da pilha tiverem *high value*

Esquema Básico de Intercalação



31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Arquivos por intercalar
(cada coluna representa um arquivo)

2
6
10
13
15
20
31
40
41
51
60
69
70
76

Livro pág. 55

Número de iterações

- A cada iteração, encontra-se a menor chave (O(n))
  - n é o número de arquivos a ordenar
- Número de iterações = número total de registros a serem ordenados

31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Mas…

- E se a quantidade de arquivos a intercalar for muito grande?
  - Encontrar o menor valor de chave pode ser uma tarefa custosa
  - Operação de busca da menor chave tem que ser repetida várias e várias vezes, até os arquivos terminarem

Otimização do Algoritmo

- Ávore Binária de Vencedores

Árvore binária de vencedores

- Nós folha representam as chaves que estão nos topos das pilhas dos arquivos a intercalar
- Cada nó interno representa o menor de seus dois filhos
- A raiz representa o menor nó da árvore

Árvore binária de vencedores

- Cada nó interno tem quatro componentes
  - Vencedor: valor da menor chave daquela sub-árvore
  - EndVencedor: ponteiro para o arquivo que tem aquela chave
  - Left: ponteiro para o filho da esquerda
  - Rigth: ponteiro para o filho da direita

Exemplo

- Arquivos a serem ordenados
  - Cada coluna abaixo representa um arquivo com suas respectivas chaves

31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

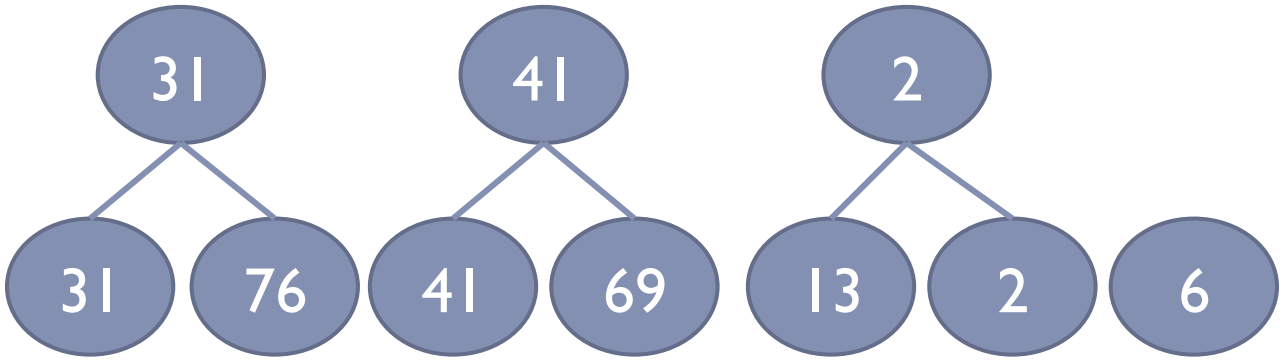
Árvore Binária de Vencedores

- Colocar em memória o primeiro registro de cada arquivo
  - Cada registro é um nó folha da árvore (aqui usamos apenas as chaves para simplificar)

31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Árvore Binária de Vencedores

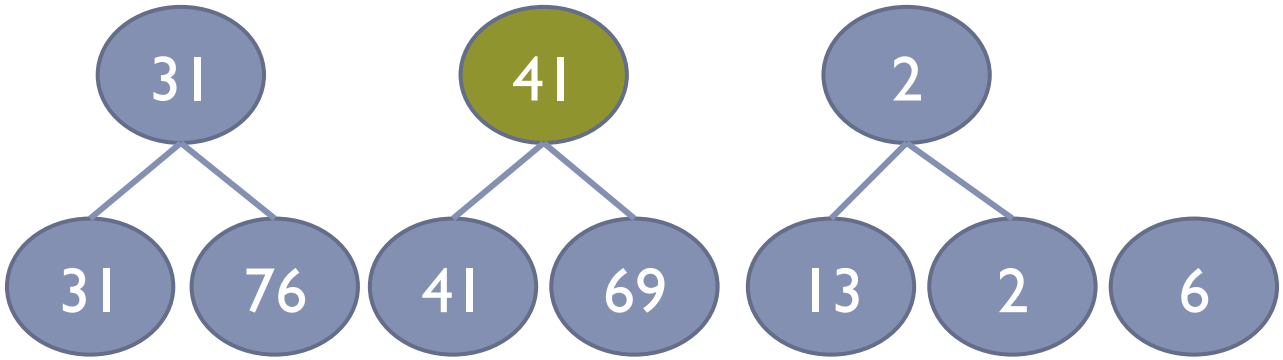
- Criar um nó raiz para cada 2 nós folha, com o menor dos dois valores



31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Árvore Binária de Vencedores

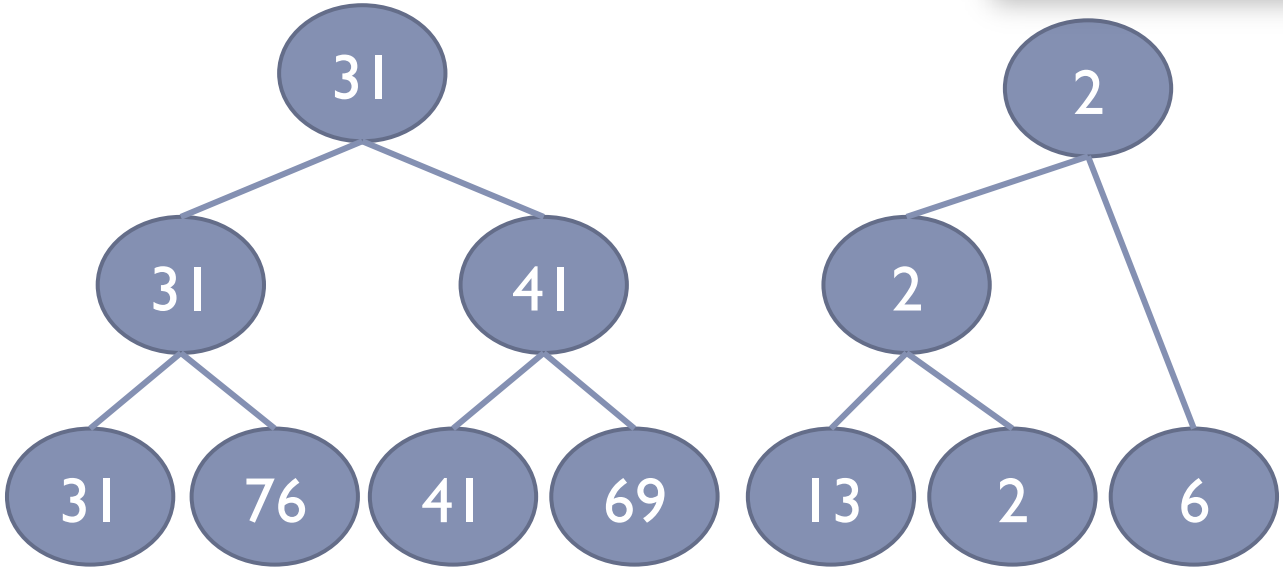
- Representação do nó interno 41
  - Vencedor: 41
  - EndVencedor: 3
  - Left: 41
  - Rigth: 69



31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Árvore Binária de Vencedores

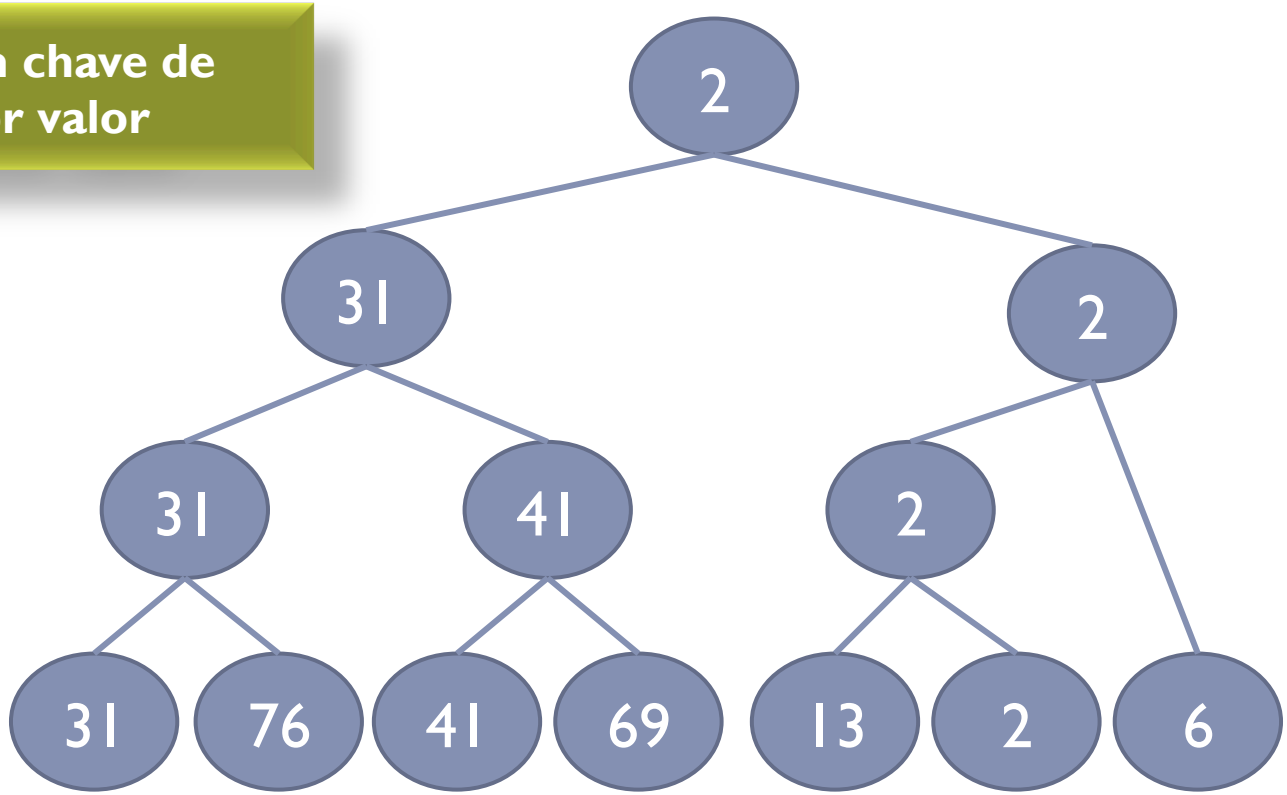
Atenção: valores de chave se repetem em vários níveis



31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Árvore Binária de Vencedores

Raiz tem chave de menor valor

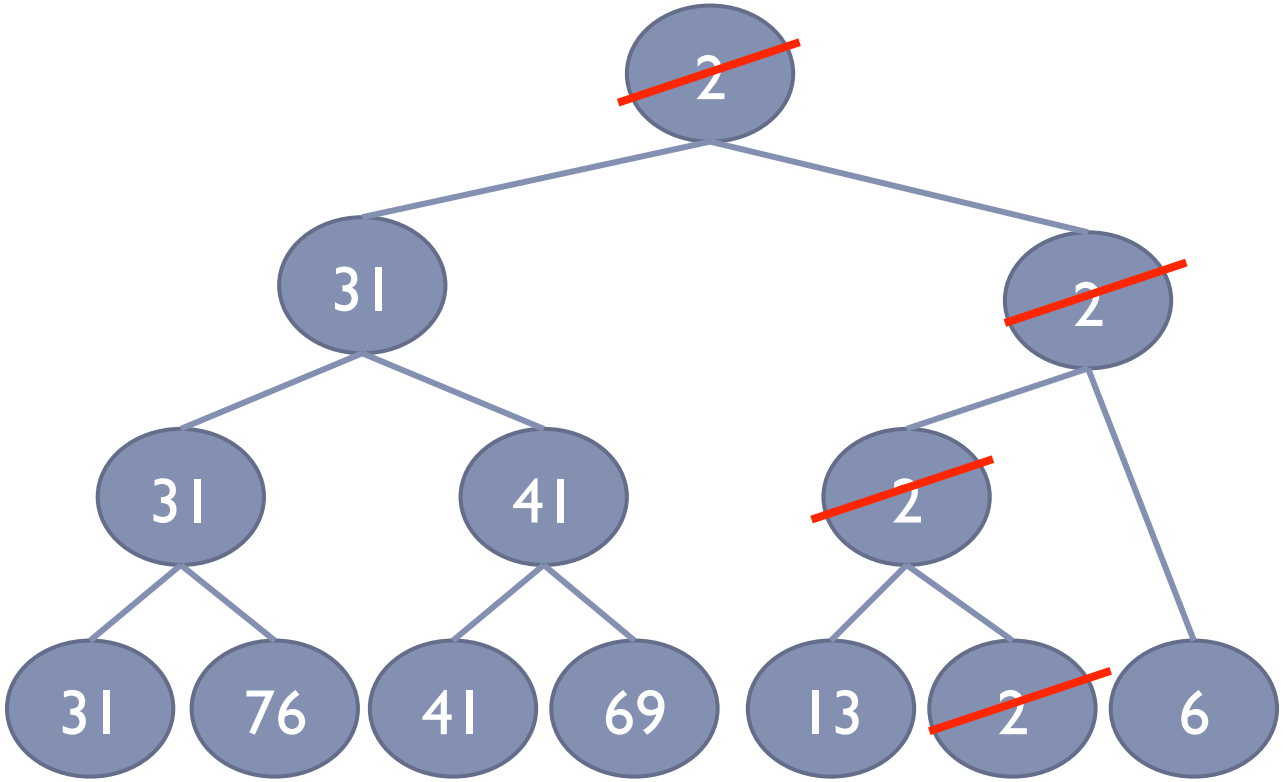


31	76	41	69	13	2	6
70				40	20	10
					51	15
						60

Uso da Árvore de Vencedores no algoritmo de Intercalação

- Chave da raíz é retirada e registro correspondente é inserido no arquivo

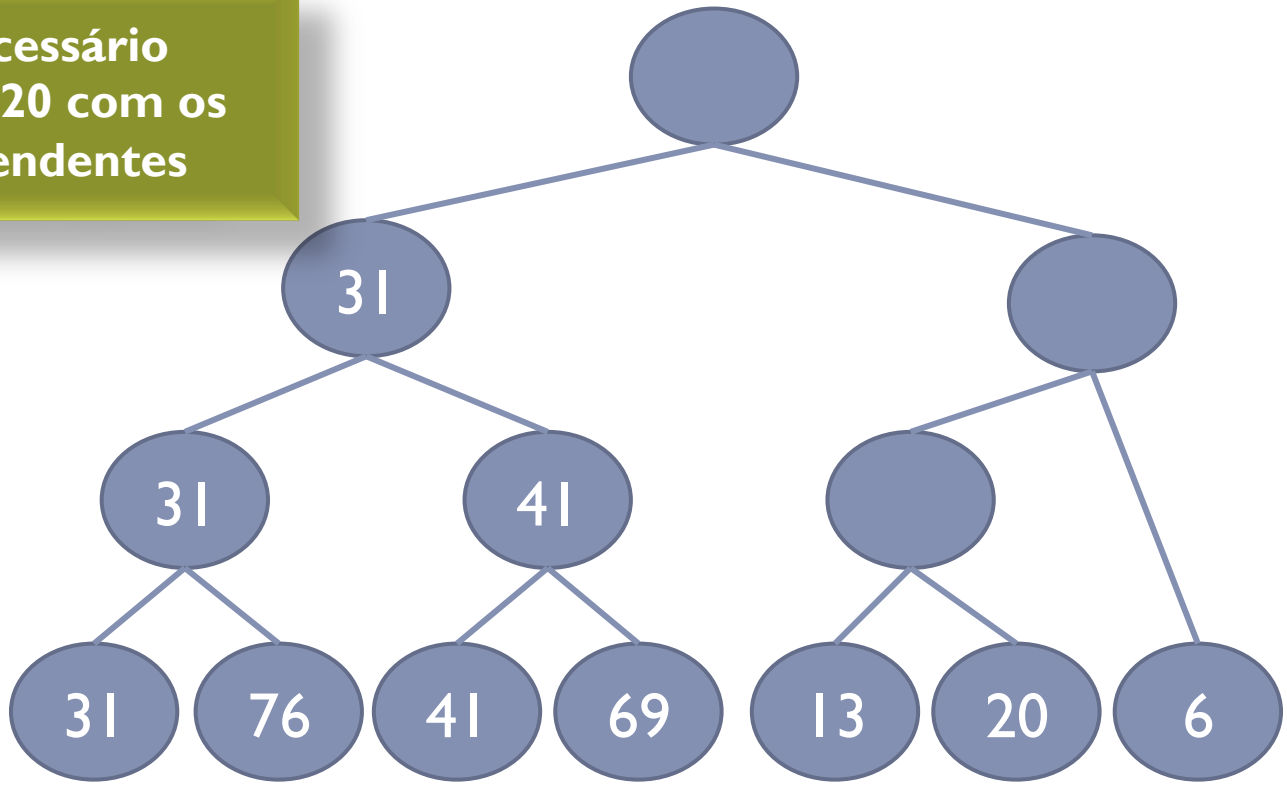
Árvore Binária de Vencedores



31	76	41	69	13	2	6
70	HV	HV	HV	40	20	10
HV				HV	51	15
					HV	60
						HV

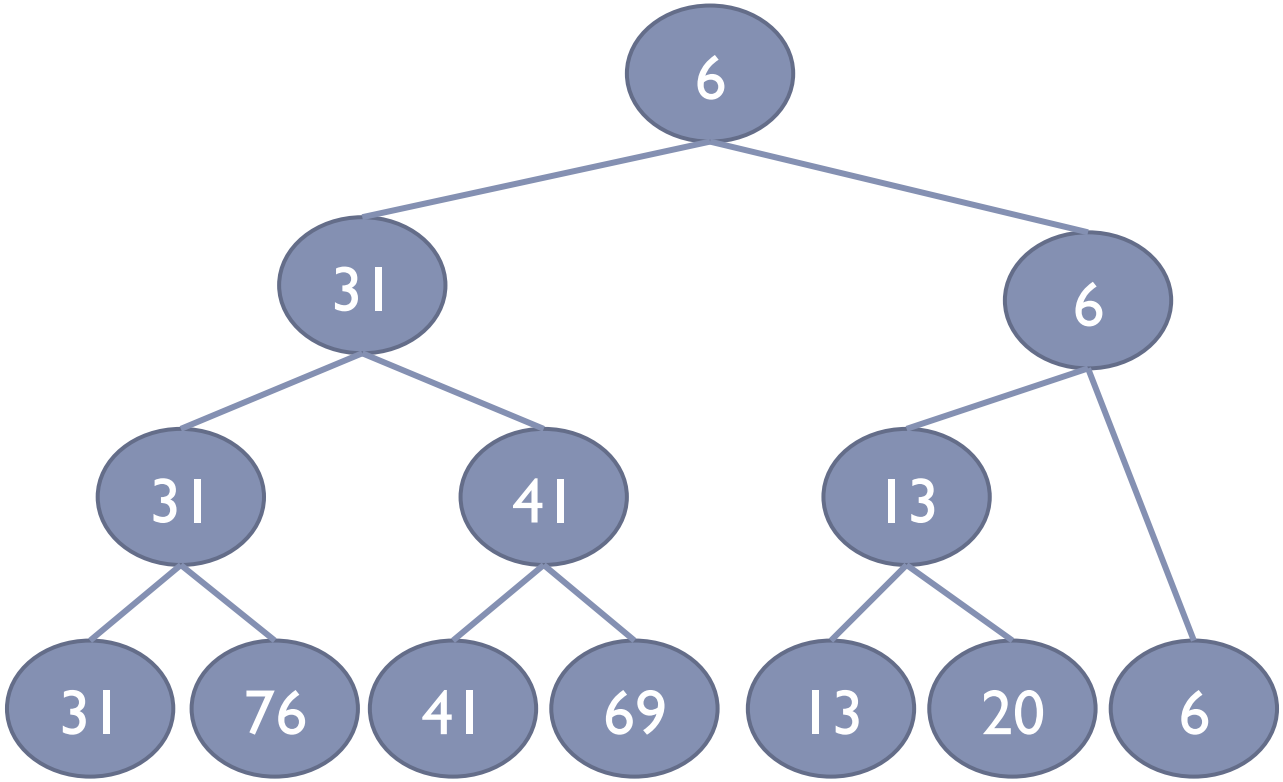
Árvore Binária de Vencedores

Só é necessário comparar 20 com os seus ascendentes



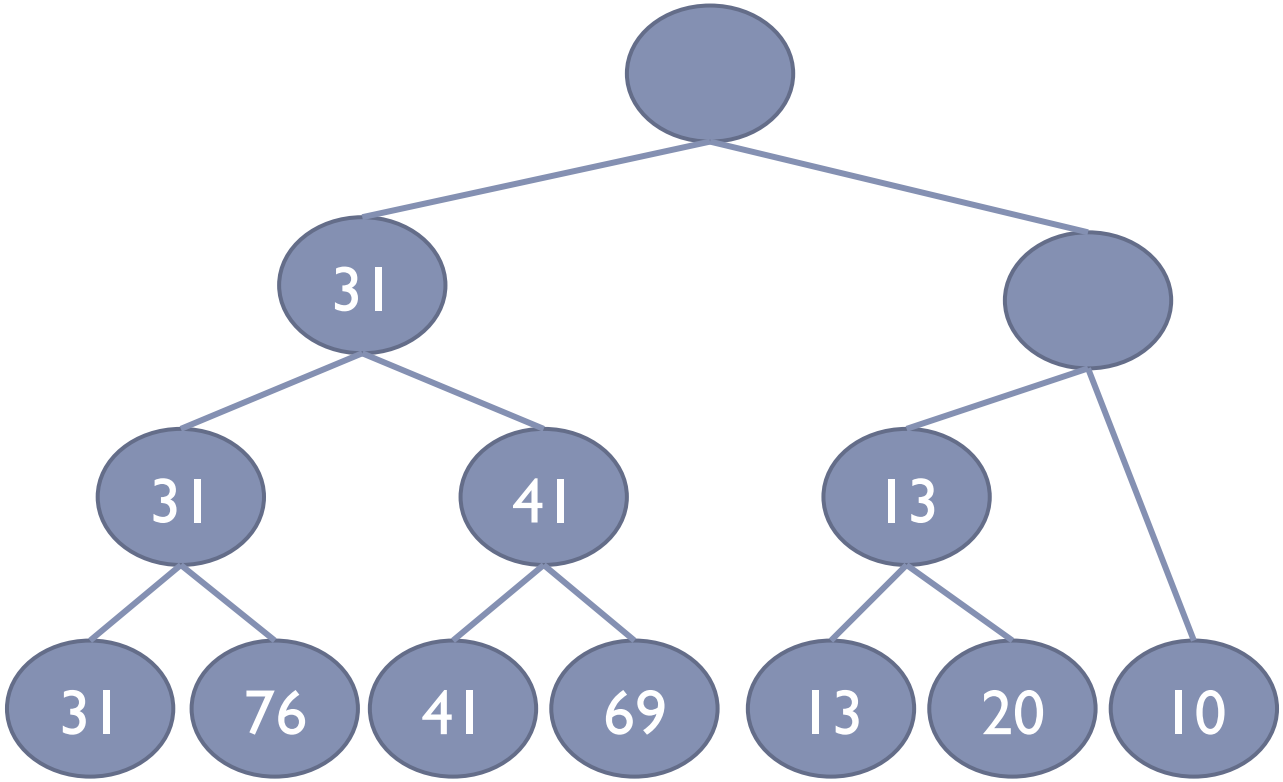
31	76	41	69	13	20	6
70	HV	HV	HV	40	51	10
HV				HV	HV	15
						60
						HV

Árvore Binária de Vencedores



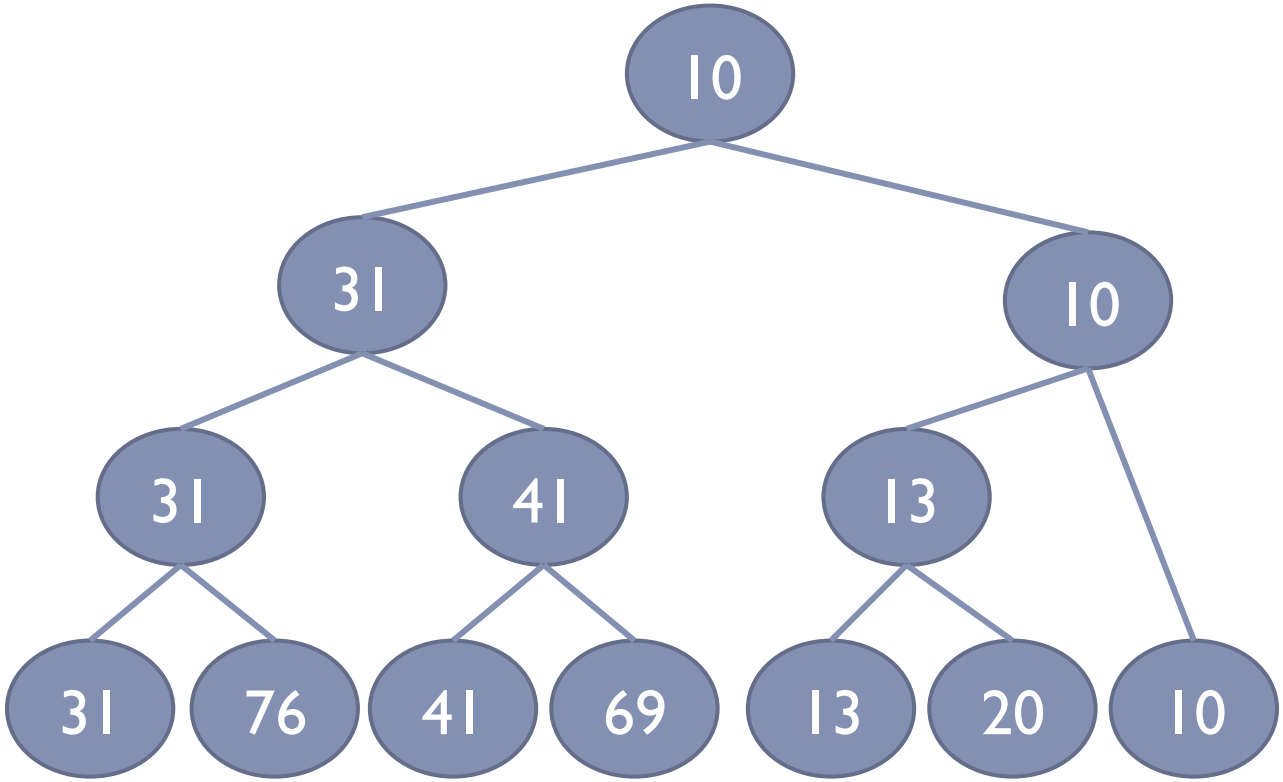
31	76	41	69	13	20	6
70	HV	HV	HV	40	51	10
HV				HV	HV	15
						60
						HV

Árvore Binária de Vencedores



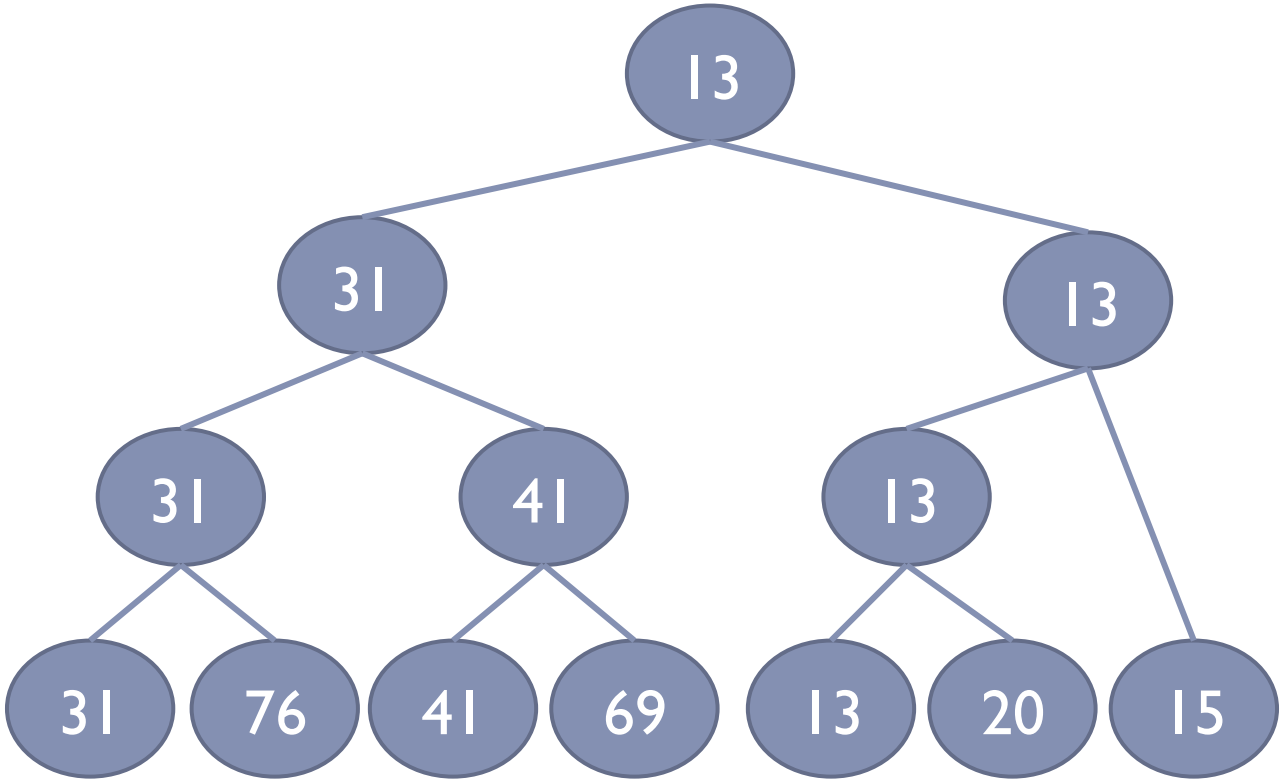
31	76	41	69	13	20	10
70	HV	HV	HV	40	51	15
HV				HV	HV	60
						HV

Árvore Binária de Vencedores



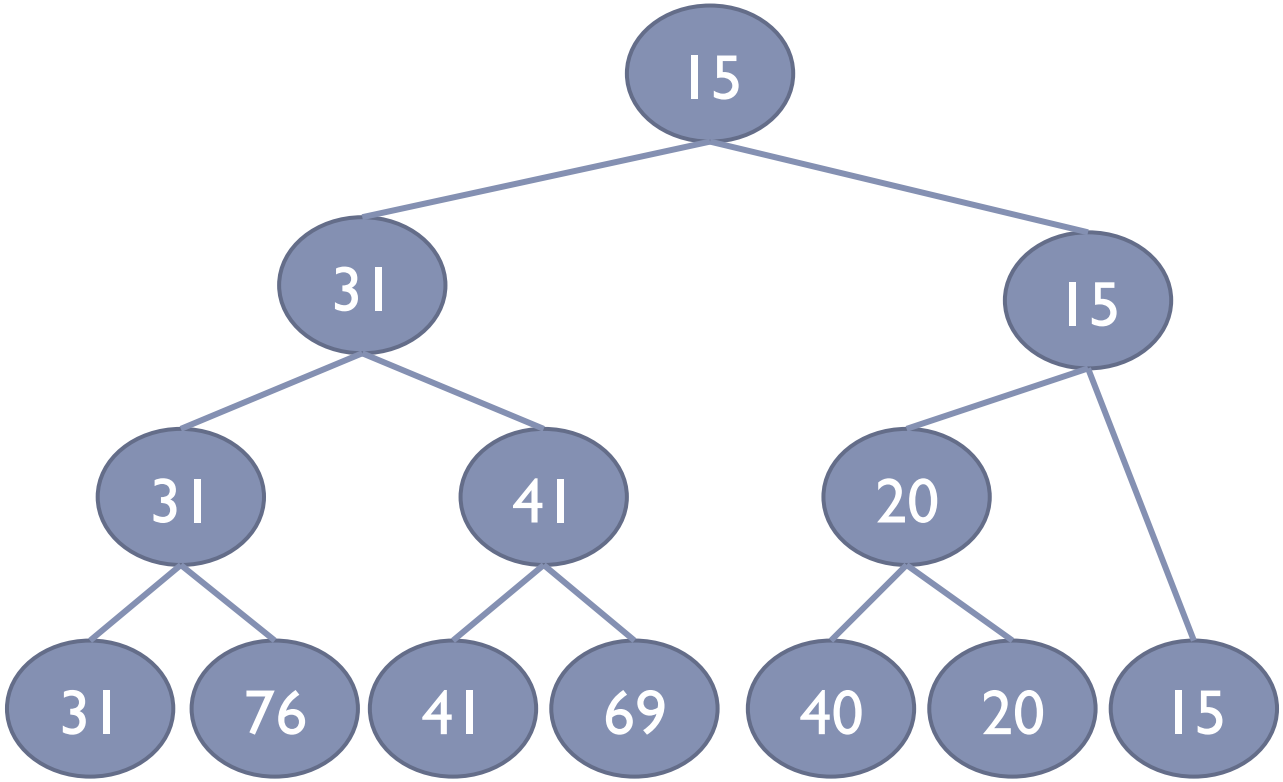
31	76	41	69	13	20	10
70	HV	HV	HV	40	51	15
HV				HV	HV	60
						HV

Árvore Binária de Vencedores



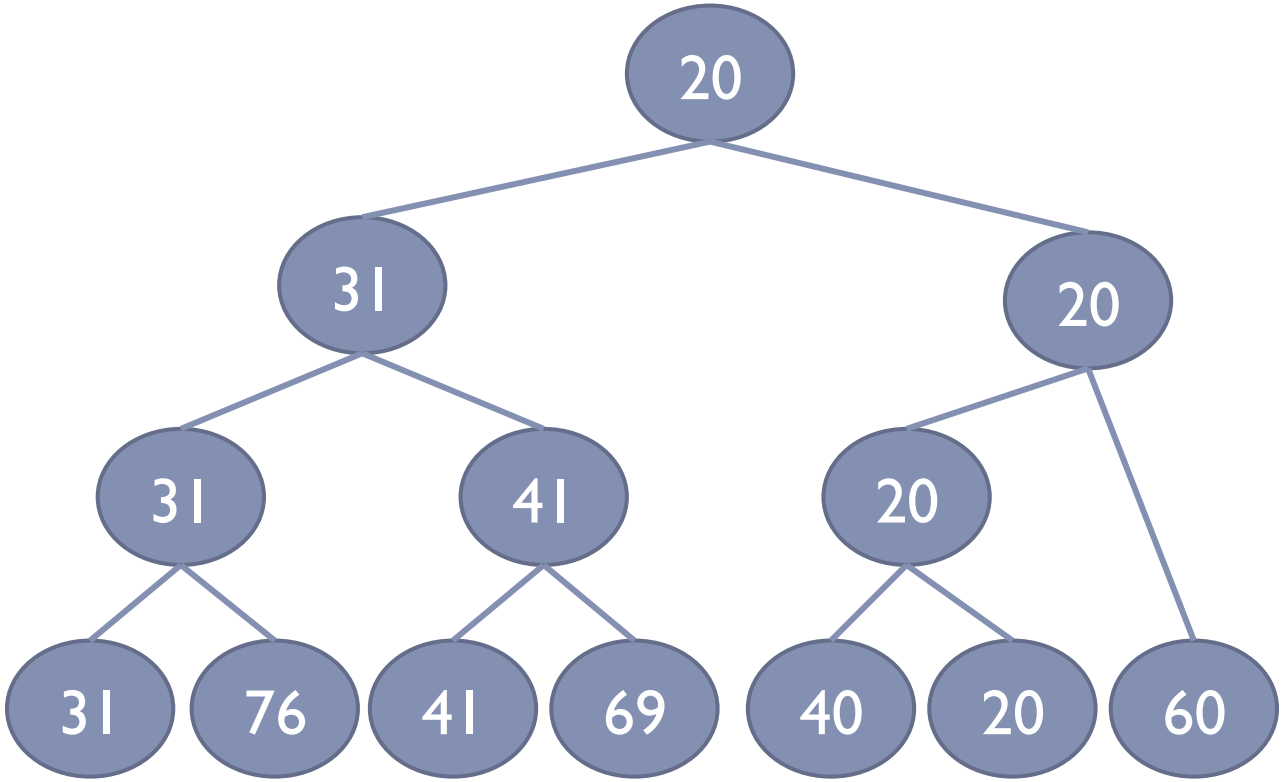
31	76	41	69	13	20	15
70	HV	HV	HV	40	51	60
HV				HV	HV	HV

Árvore Binária de Vencedores



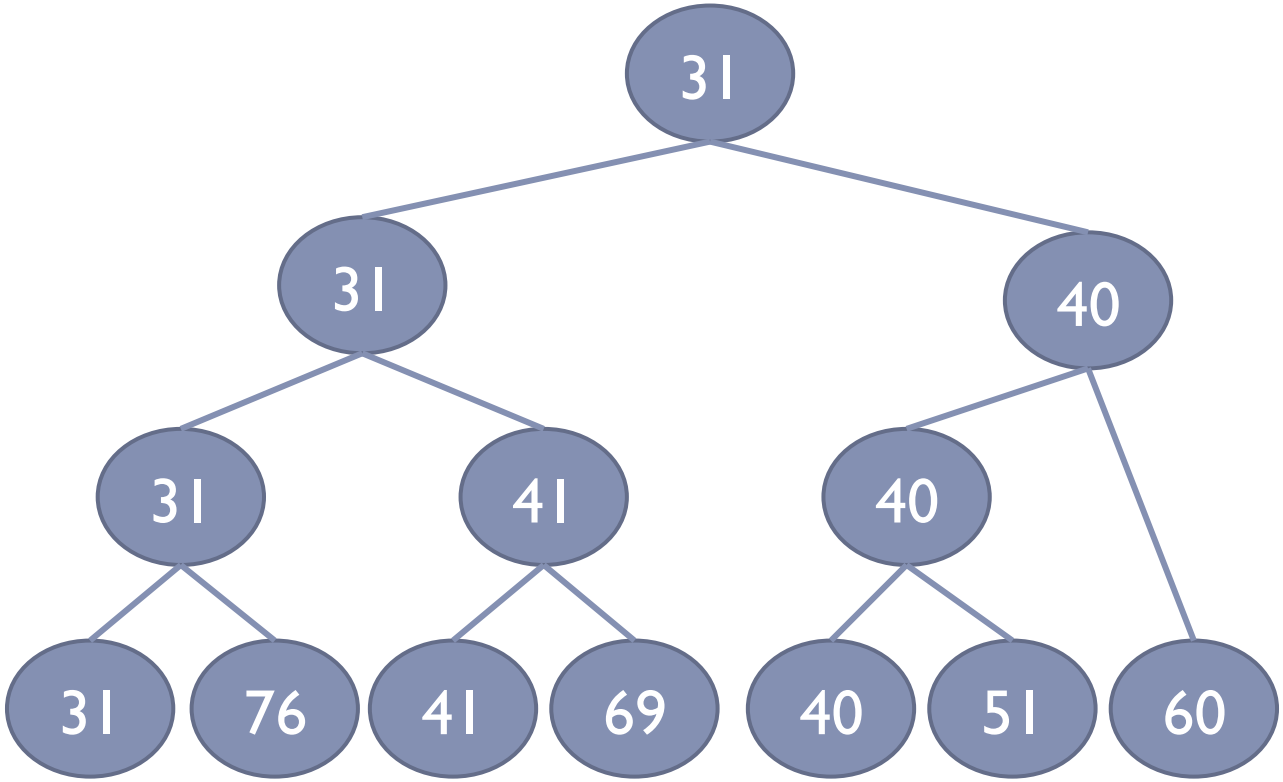
31	76	41	69	40	20	15
70	HV	HV	HV	HV	51	60
HV					HV	HV

Árvore Binária de Vencedores



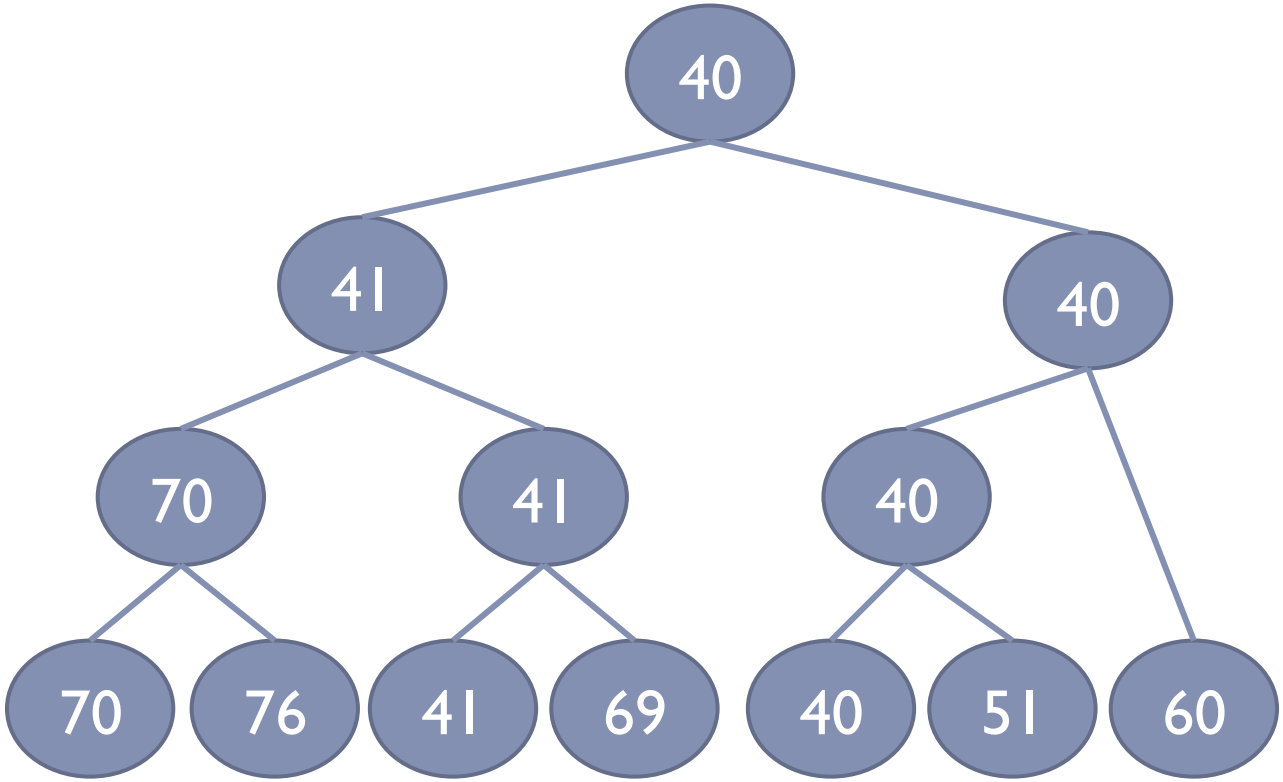
31	76	41	69	40	20	60
70	HV	HV	HV	HV	51	HV
HV					HV	

Árvore Binária de Vencedores



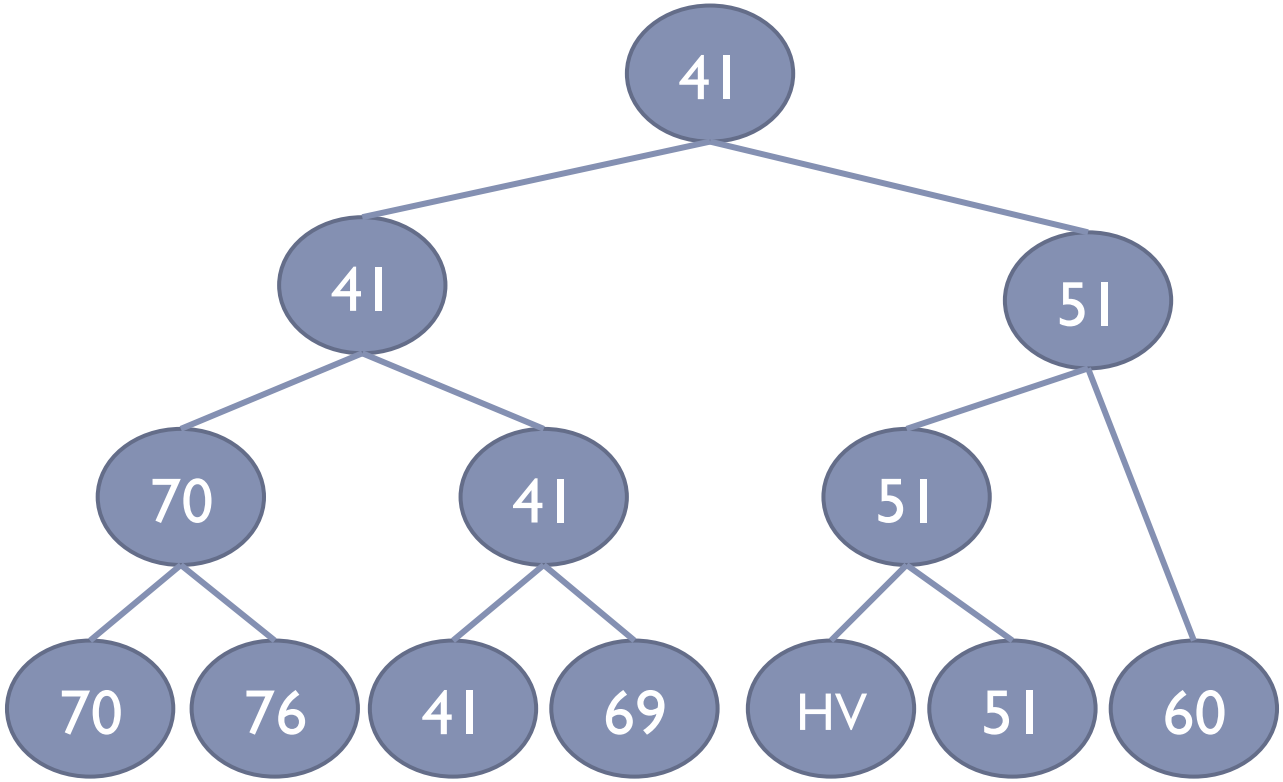
31	76	41	69	40	51	60
70	HV	HV	HV	HV	HV	HV
HV						

Árvore Binária de Vencedores



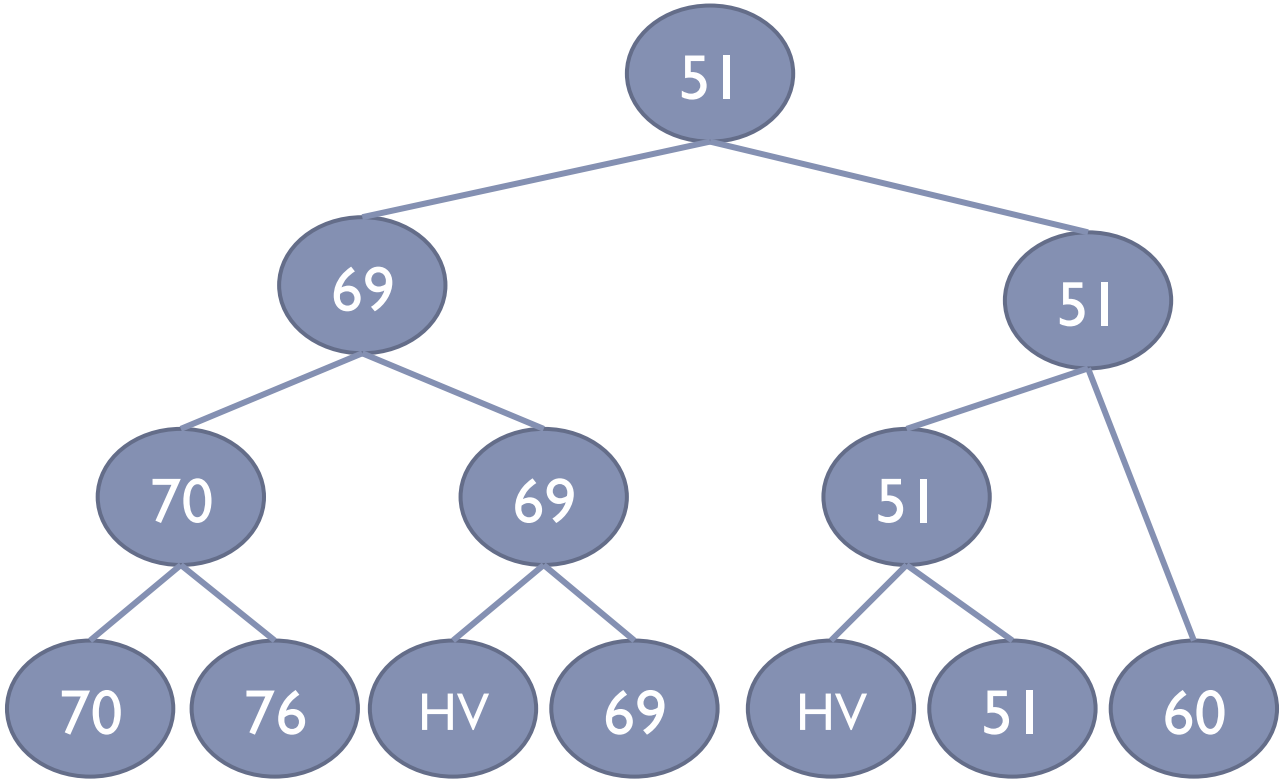
70	76	41	69	40	51	60
HV	HV	HV	HV	HV	HV	HV

Árvore Binária de Vencedores



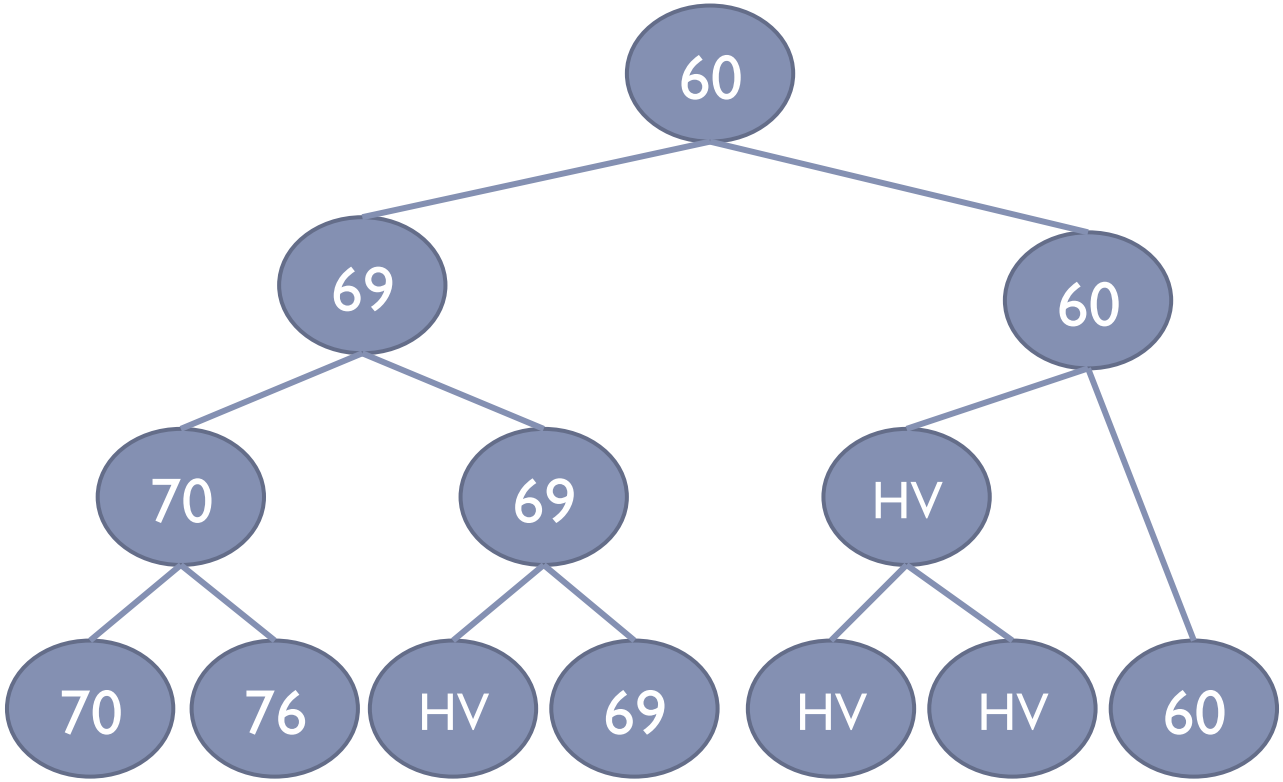
70	76	41	69	HV	51	60
HV	HV	HV	HV		HV	HV

Árvore Binária de Vencedores



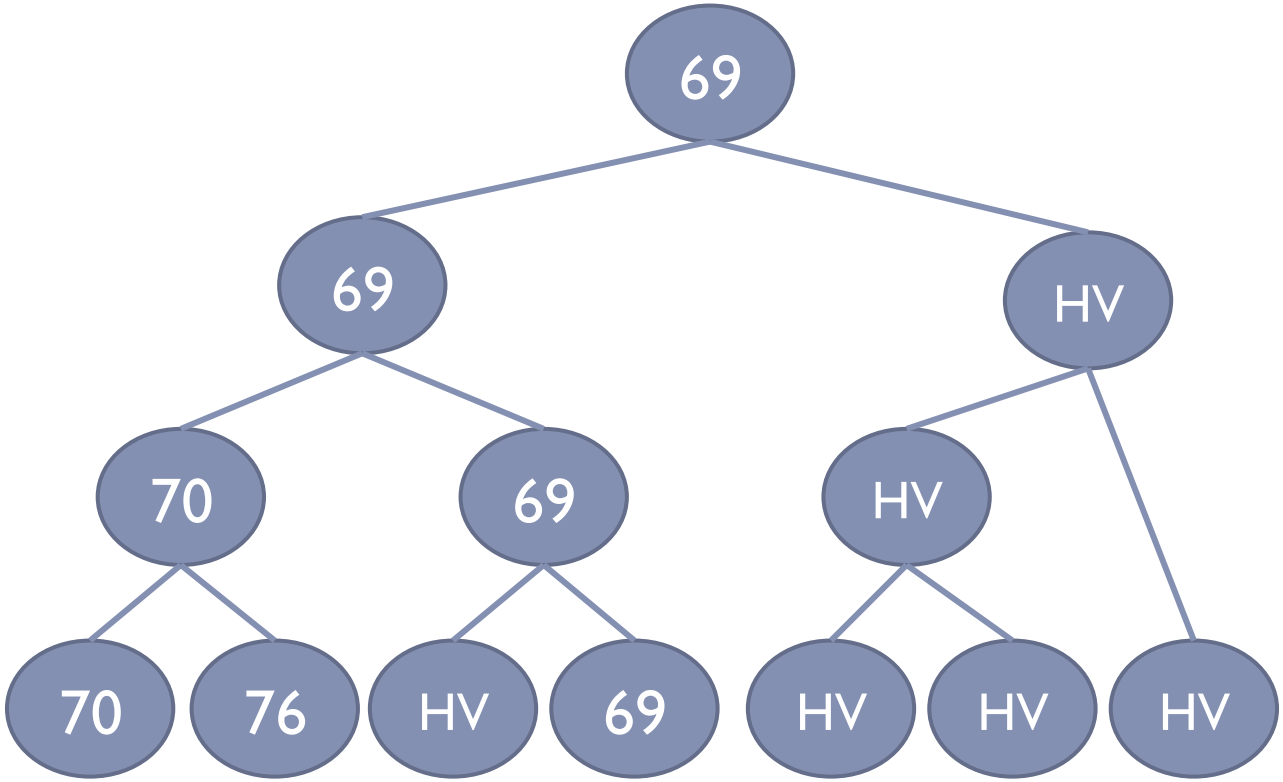
70	76	HV	69	HV	51	60
HV	HV		HV		HV	HV

Árvore Binária de Vencedores



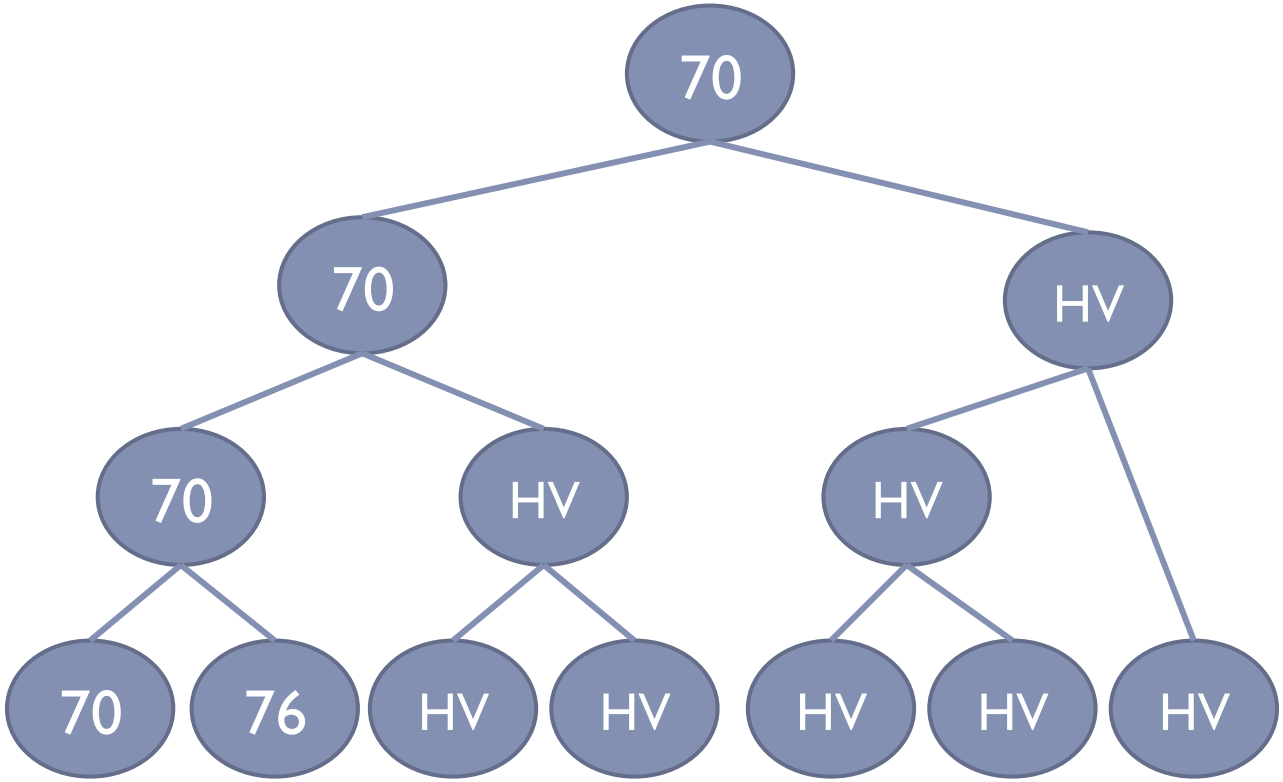
70	76	HV	69	HV	HV	60
HV	HV		HV			HV

Árvore Binária de Vencedores



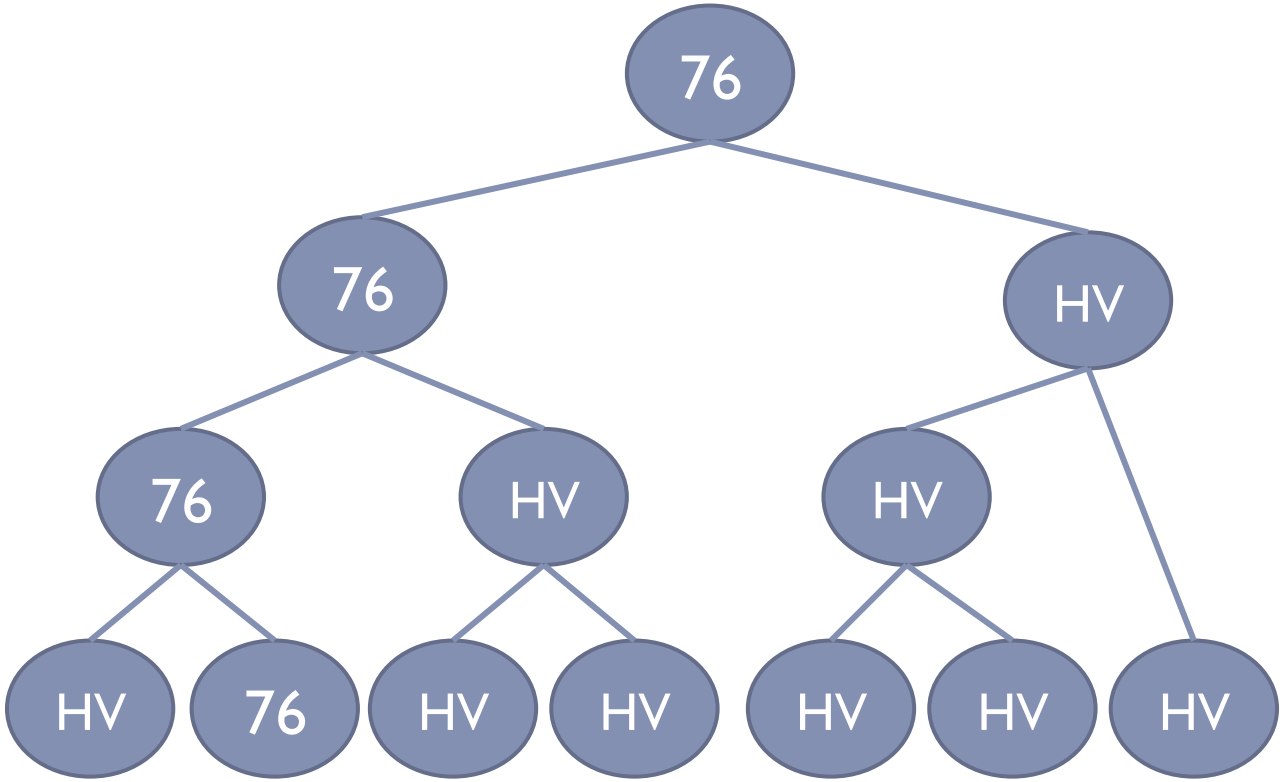
70	76	HV	69	HV	HV	HV
HV	HV		HV			

Árvore Binária de Vencedores



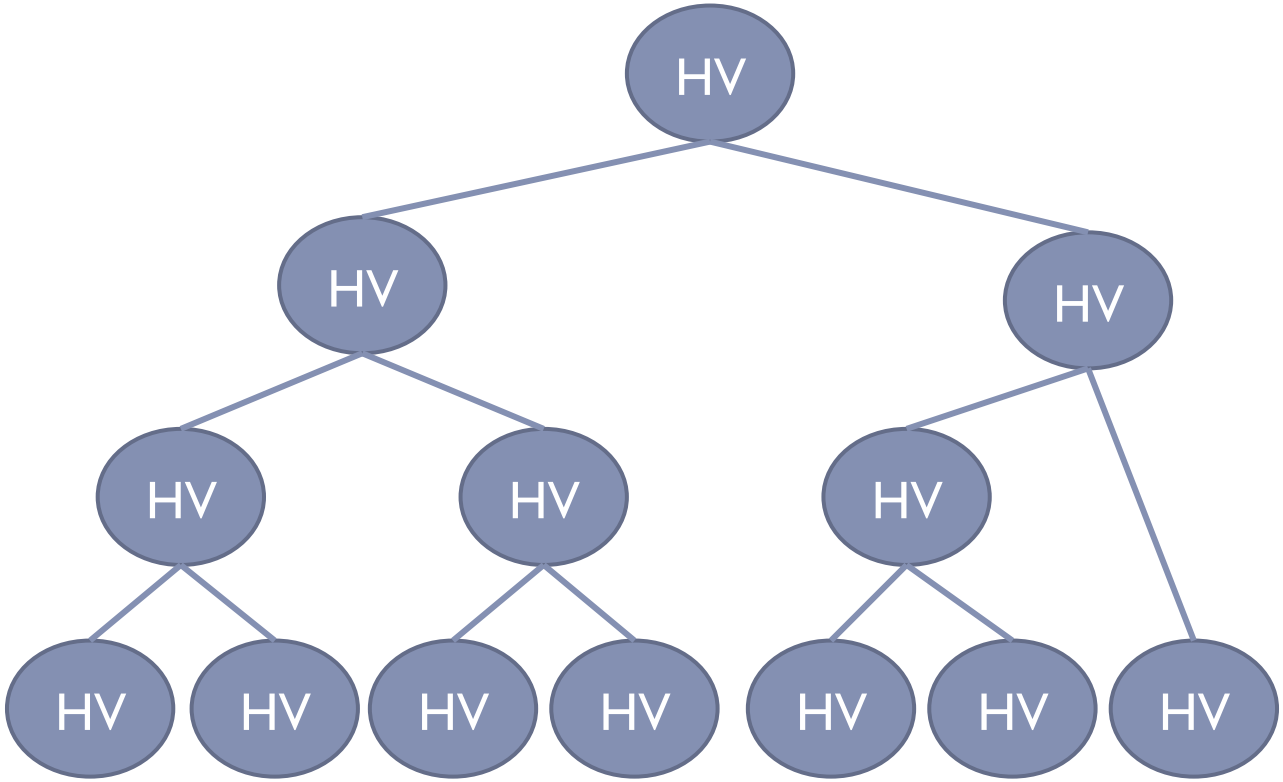
70	76	HV	HV	HV	HV	HV
HV	HV					

Árvore Binária de Vencedores



HV	76	HV	HV	HV	HV	HV
	HV					

Árvore Binária de Vencedores



HV	HV	HV	HV	HV	HV	HV

Discussão

- Montagem da árvore: **O(n)**
- A cada iteração, faz-se **log n** comparações (**n** é o número de arquivos a comparar)
- Número de iterações: número total de registros a serem ordenados

Exercício

- Montar a árvore de vencedores para a seguinte situação
- Simular a execução do algoritmo de intercalação

2	55	40	3	13	7	12	6	45	43	15
70			67	41	21	17		49	57	16
79			80			82				23
98										25

Exercício

- Implementar o algoritmo de intercalação utilizando árvore binária de vencedores
  - Entrada:
    - Arraylist de strings, com os nomes dos arquivos de entrada
    - Nome do arquivo de saída
  - Estrutura dos arquivos: Clientes (CodCliente, Nome, DataNascimento)