Linguagem C: Árvores AVL

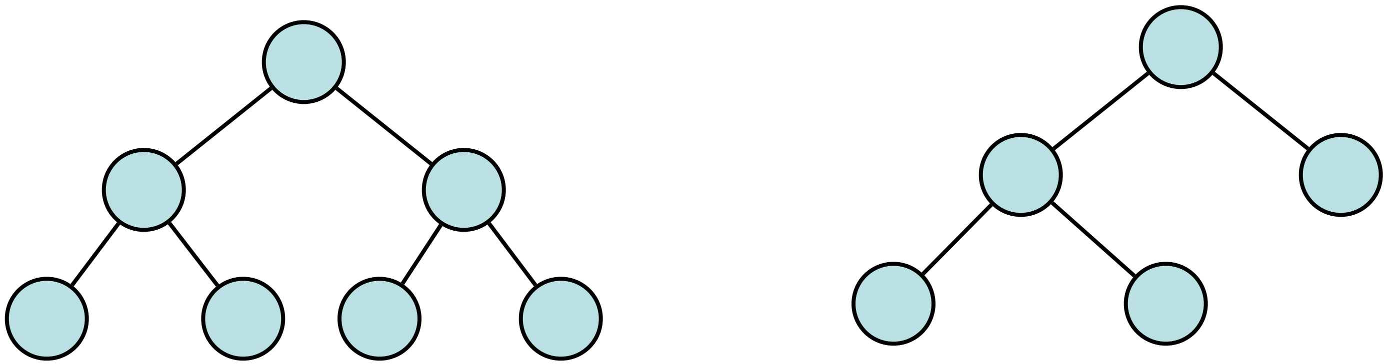
Luis Martí
Instituto de Computação
Universidade Federal Fluminense
lmarti@ic.uff.br - http://lmarti.com

Árvores Balanceadas

- As árvores binárias de pesquisa são, em alguns casos, pouco recomendáveis para as operações básicas (inserção, remoção e busca)
- Árvores binárias de pesquisa degeneradas tornam as operações básicas lentas O(n)

Árvores Balanceadas

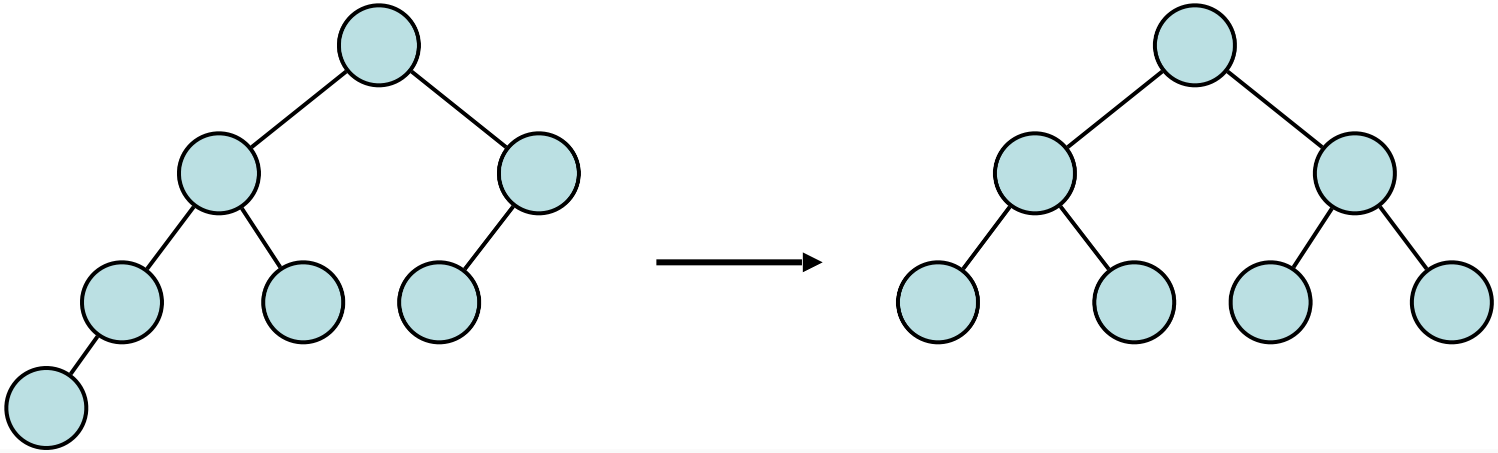
- Árvore binária completamente balanceada
  - Ocorre quando a árvore está cheia ou quase cheia com o nível n-1 completo



  - Uma árvore binária completa leva um tempo na ordem de O(log n) para operações de inserção, remoção e pesquisa. O que é, sem dúvida, muito bom

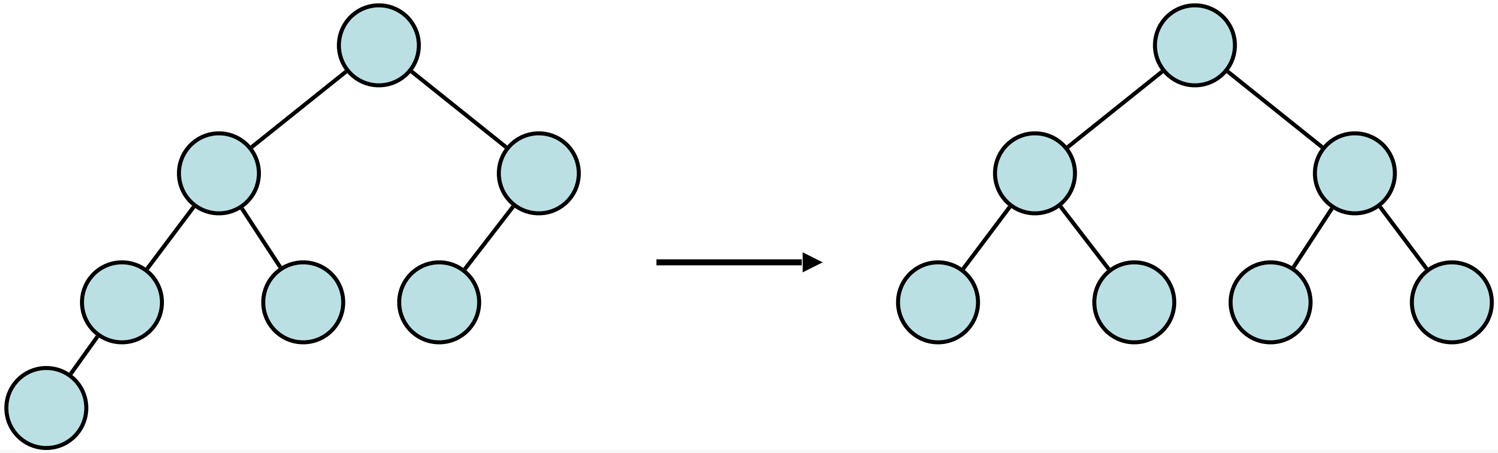
Árvores Balanceadas

- Árvore binária completamente balanceada
  - Após uma inserção ou remoção a árvore pode deixar de ser completa. A solução seria aplicar um algoritmo que tornasse a árvore novamente completa, porém o custo para realizar está operação seria de O(n)



Árvores Balanceadas

- Árvore binária completamente balanceada
  - Percebe-se que todos os nós tiveram sua posição na estrutura alterados
  - Na maioria dos casos, utiliza-se árvores quase balanceadas



Critérios para definir balanceamento

- Vários são os critérios (métodos) para definir balanceamento. Alguns são:
  - Restrições imposta na diferença das alturas das subárvores de cada nó. Ex. AVL
  - Redução do comprimento do caminho interno da árvore
  - Todos os nós folhas no mesmo nível

Árvores AVL

- Foram introduzidas por Adelson-Velskii e Landis em 1962
- São baseadas em árvore binárias de pesquisa
- A medida em que as operações de inserção e remoção são efetuadas a árvore é balanceada

Árvores AVL

- Definição:
  - Uma árvore binária T é dita AVL quando, para qualquer nó v de T, a diferença entre a altura das subárvores esquerda $h_e(v)$ e direita $h_d(v)$ é no máximo em módulo igual a 1.

OBS.: se uma árvore T é AVL, então todas as suas subárvores também são AVL

Árvores AVL

- Balanceamento de um nó
  - O fator de balanceamento:
    - É dado pela altura da subárvores da esquerda $h_e(v)$ menos a altura da subárvore da direita $h_d(v)$.

$$FB(v) = h_e(v) - h_d(v)$$

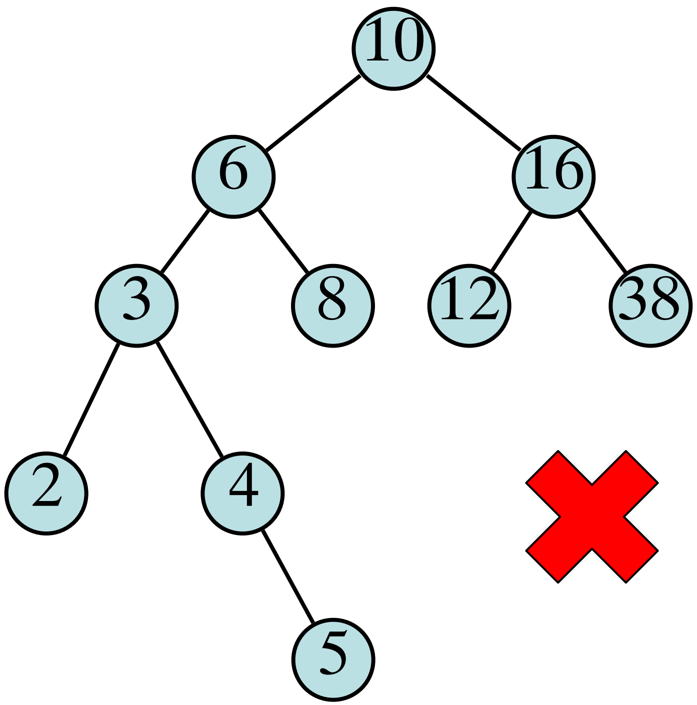
Árvores AVL

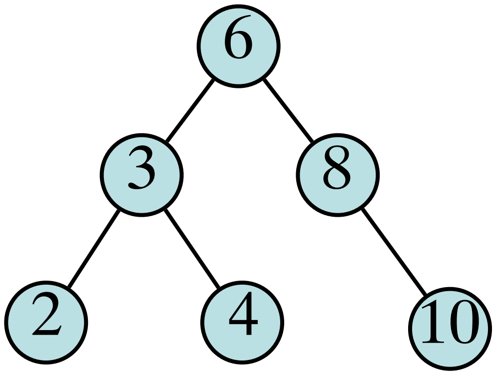
- Nós balanceados
  - São aqueles onde os valores de FB são -1, 0 ou 1

- FB(v):
  - +1: subárvore esquerda mais alta que a direita
  - 0: subárvore esquerda igual a direita
  - -1: subárvore direita mais alta do que a esquerda

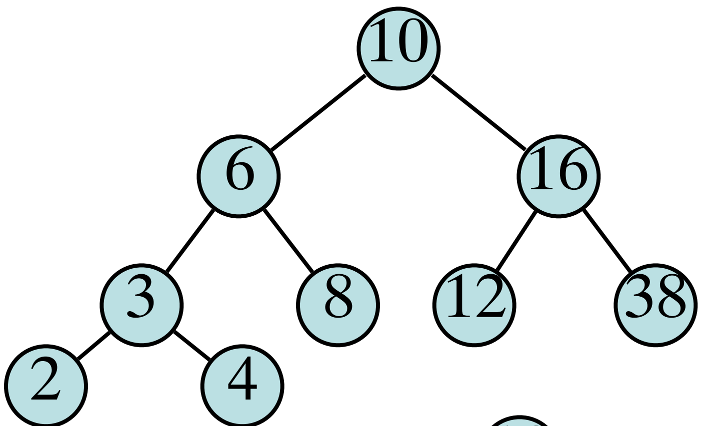
Árvores AVL

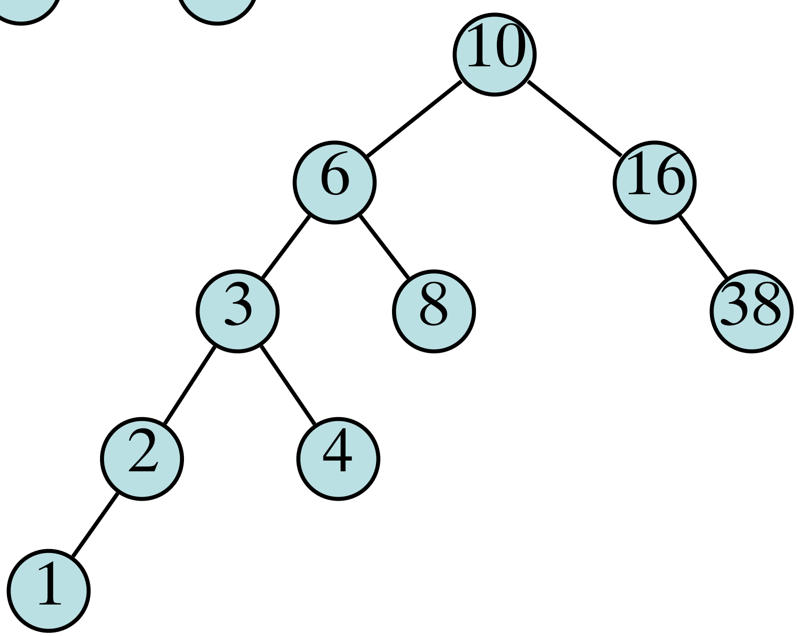
- Exemplos



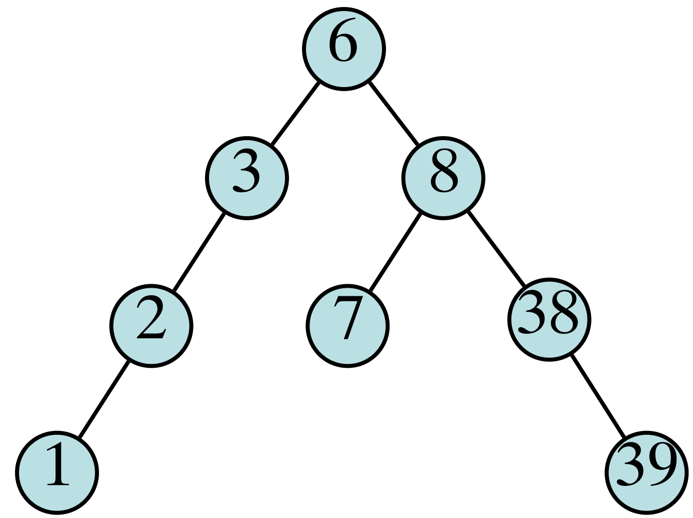


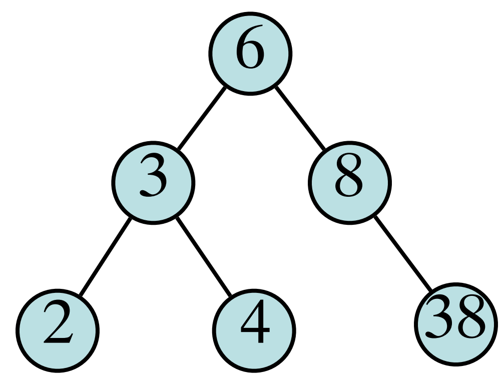
Árvores AVL



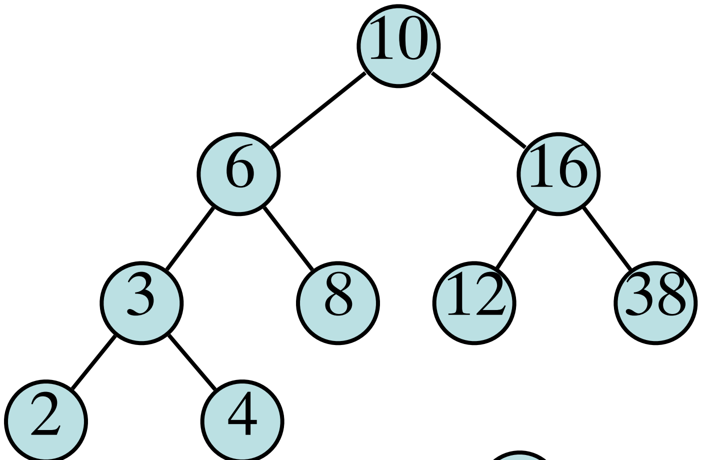


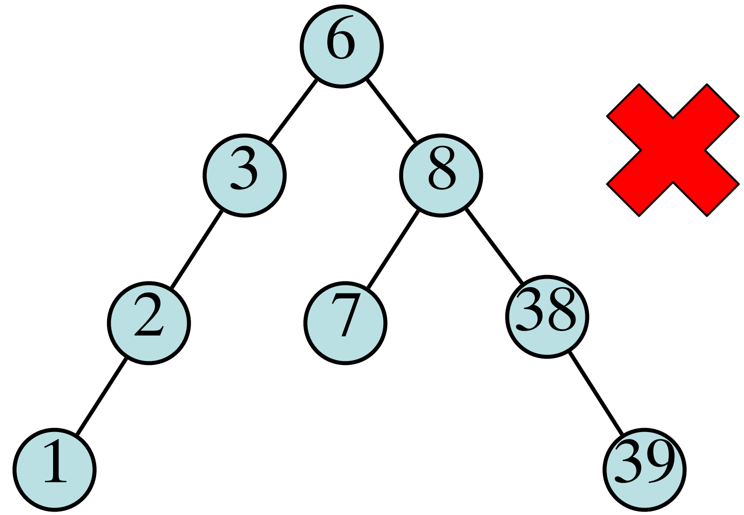
?

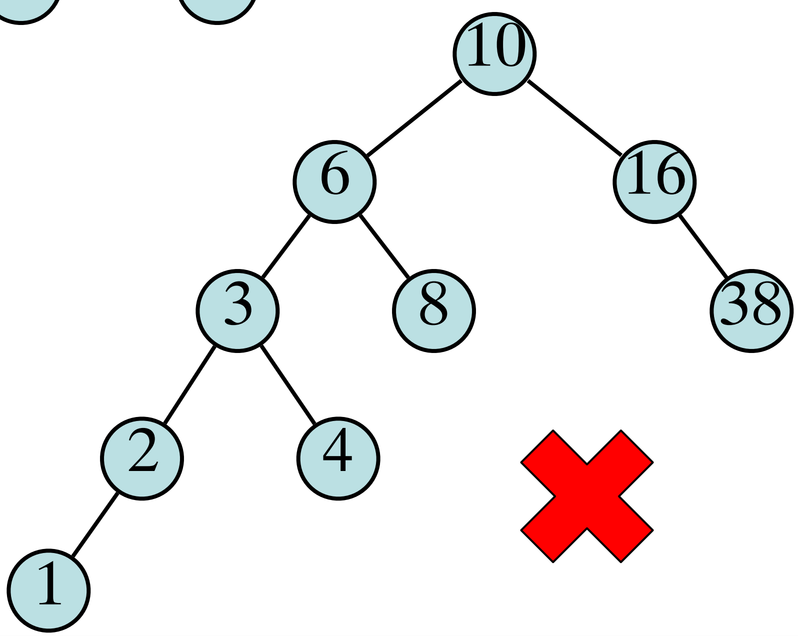


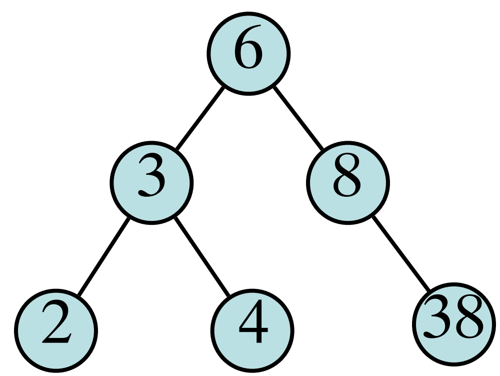


Árvores AVL



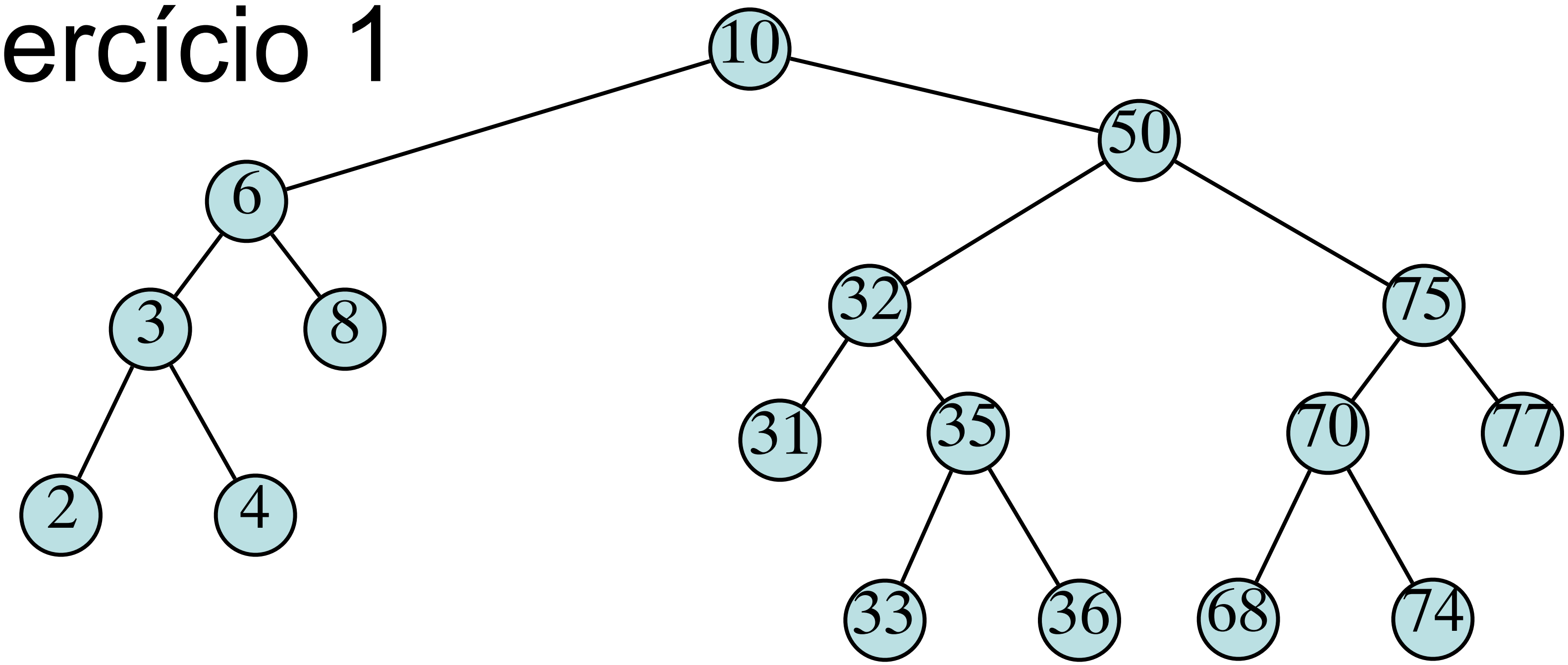






Árvores AVL

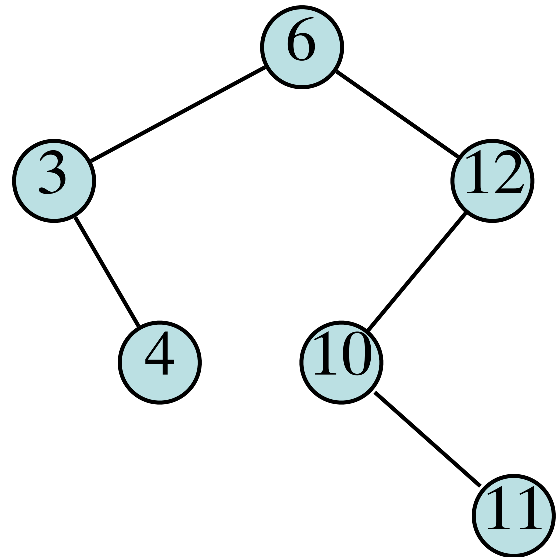
- Exercício 1



- Determinar o fator de balanceamento de cada nó
- Dizer se a árvore é AVL
- Verificar quais as possíveis posições para a inserção de elementos e em quais posições de inserção, a árvore se mantém ou se torna AVL

Árvores AVL

- Exercício 2



- Determinar o fator de balanceamento de cada nó
- Dizer se a árvore é AVL
- Verificar quais as possíveis posições para a inserção de elementos e em quais posições de inserção, a árvore se mantém ou se torna AVL

Árvore AVL de altura h

Qual o número mínimo de nós?

- Suponha altura(subesq(v)) = h-1
- Como queremos analizar o número mínimo de nós

altura(subd(v)) = h-2

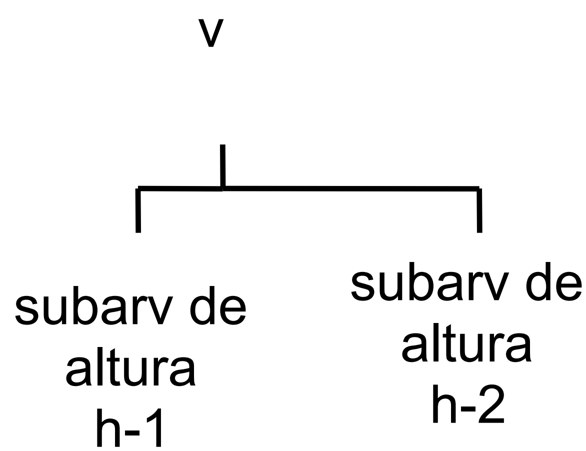
- árvore AVL construída recursivamente:

Seja T_h uma árvore AVL de altura h

h = 0 T_h tem 0 nós: $| T_h | = 0$

h = 1 $| T_h | = 1$

h = 2 $| T_h | = 1 + | T_{h-1} | + | T_{h-2} |$



Qual o número mínimo de nós?

h	$\lvert T_h \rvert$
0	0
1	1
2	2
3	4
4	7
5	12
6	20
7	33

Qual o número mínimo de nós?

- Por observação:
  - o h-ésimo elemento da sequência de Fibonacci é:

$$F_h = 0 \qquad \text{se } h = 0$$

$$F_h = 1 \qquad \text{se } h = 1$$

$$F_h = F_{h-1} + F_{h-2} \qquad \text{se } h > 1$$

análogo a $|T_h|$, mas diferindo de 1: $|T_h| = F_h + 1$

- pode-se provar que (exercício 5.3)

termo **< 1**
se **h>0**

$$F_h = 1/\sqrt{5}\ [(\ (1+\sqrt{5})^h/2) - ((1-\sqrt{5})^h/2\ )]$$

$$|T_h| = F_h + 1$$

Qual o número mínimo de nós?

$|T_h| > 1/\sqrt{5}\,[((1+\sqrt{5})^h/2) - ((1-\sqrt{5})^h/2)] - 1$

fazendo $a = ((1+\sqrt{5})/2)$ temos

$|T_h| > 1/\sqrt{5}\,a^h - 1$

$|T_h| + 1 > 1/\sqrt{5}\,a^h$

$\log_a (|T_h| + 1) > \log_a (a^h/\sqrt{5})$

$\log_a (|T_h| + 1) > h \log_a \sqrt{5}$ (mudando para base 2)

$\log_2 (|T_h| + 1)/ \log_2 a > h \sqrt{5}$

$\rightarrow h = O(\log n)$

Inserção AVL

- Procedimento:
  - Percorrer a árvore até o ponto de inserção (usando a operação de busca)
  - Inserir o novo elemento
  - Balancear a árvore (quando necessário fazer rotações)

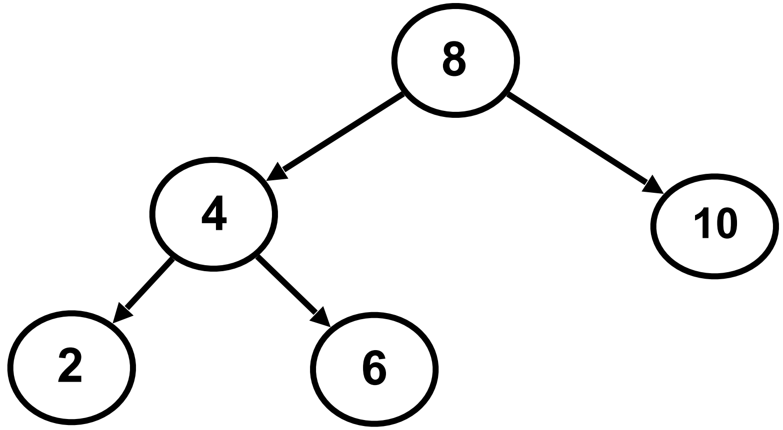
Inserção AVL

- Dada uma raiz *r* com sub-árvores *L* (*left*) e *R* (*right*), e supondo que a inserção deve ser feita na subárvore da **esquerda (L)**
- Existem 3 casos possíveis:
  1. Se hL = hR, então L e R ficam com alturas diferentes mas continuam balanceadas.
  2. Se hL < hR, então L e R ficam com alturas iguais e balanceamento é melhorado.
  3. Se hL > hR, então L fica ainda maior e balanceamento é violado ➔ necessita ser balanceada

Inserção AVL

- Da mesma forma, dada uma raiz *r* com sub-árvores *L* (*left*) e *R* (*right*) em que a inserção deve ser feita na subárvore da **direita (R)**
- Existem 3 casos possíveis:
  1. Se hL = hR, então L e R ficam com alturas diferentes mas continuam balanceadas.
  2. Se hL > hR, então L e R ficam com alturas iguais e balanceamento é melhorado.
  3. Se hL < hR, então R fica ainda maior e balanceamento é violado ➔ necessita ser balanceada

Exemplo



- Inserção de chaves 9 e 11 pode ser feita sem balanceamento (inclusive, melhorando o balanceamento da árvore)
- Inserção de chaves 1, 3, 5 e 7 exigem o re-balanceamento da árvore

Nós desbalanceados

- Nós desregulados ou desbalanceados
  - São aqueles onde os valores de FB são diferentes de -1, 0 ou 1

- FB(v):
  - >1: subárvore esquerda está desbalanceando o nó v
  - <-1: subárvore direita está desbalanceando o nó v

Rebalanceamento

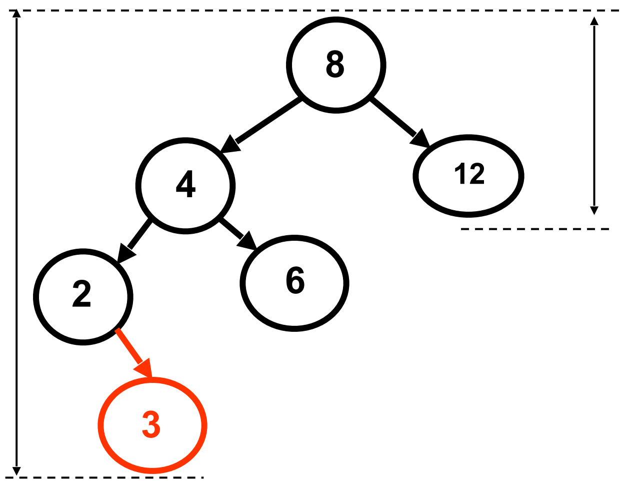
- Quando uma inserção ou remoção realizada em um nó altera o balanceamento da árvore, é necessário efetuar uma transformação na árvore, tal que:
  - O percurso em ordem fique inalterado em relação a árvore desbalanceada. Isto é, a árvore continua a ser uma árvore binária de pesquisa
  - A árvore transformada saiu de um estado de desbalanceamento para um estado de balanceamento

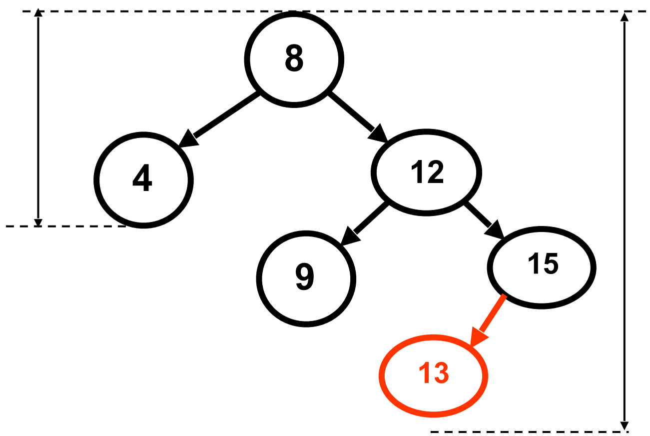
Rebalanceamento

- Os problemas de balanceamento das árvores AVL podem ser mapeados em dois casos:
  - Caso 1: o nó raiz de uma subárvore tem FB=2 (ou -2) e tem um filho com FB = 1 (ou -1), ou seja, com o mesmo sinal que o FB do nó pai.
  - Caso 2: o nó raiz de uma subárvore tem FB=2 (ou -2) e tem uma um filho com FB = -1 (ou 1), ou seja, com o sinal oposto ao FB do nó pai.

Inserção - Caso 1

- *Nó raiz da subárvore tem FB=2 (ou -2) e tem filho com FB=1 (ou -1) o qual tem o <u>mesmo</u> sinal que o FB do nó pai*



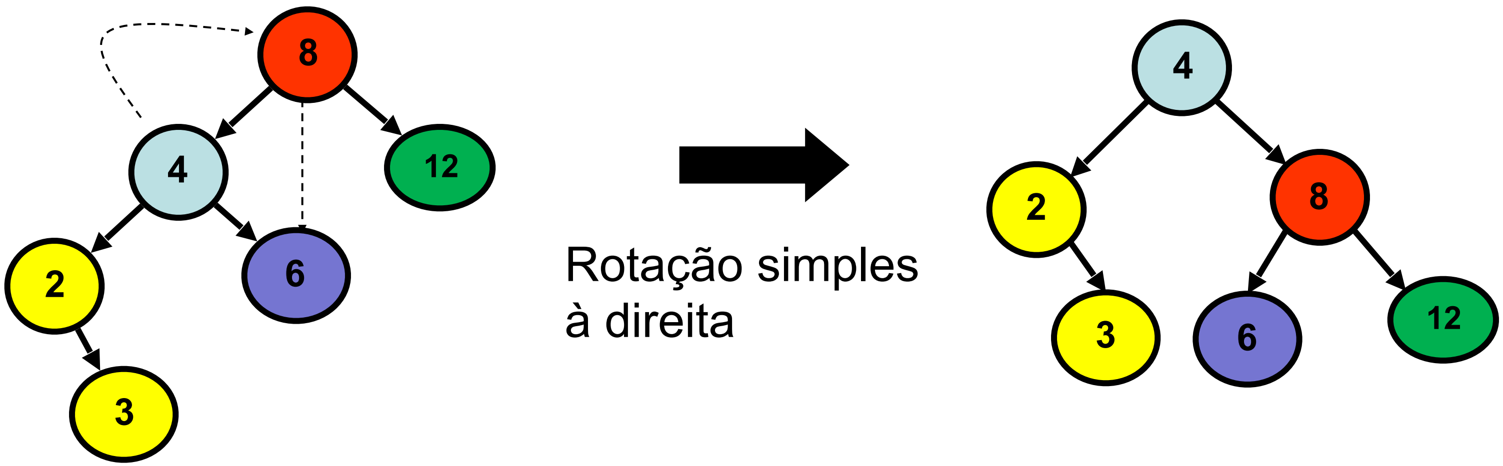


- <u>Solução</u>: rotação <u>simples</u> sobre o nó de FB=2 (-2).
  - Rotações são feitas à esquerda quando FB negativo, e à direita quando FB positivo.

Inserção - Caso 1a

FB(8) = 2

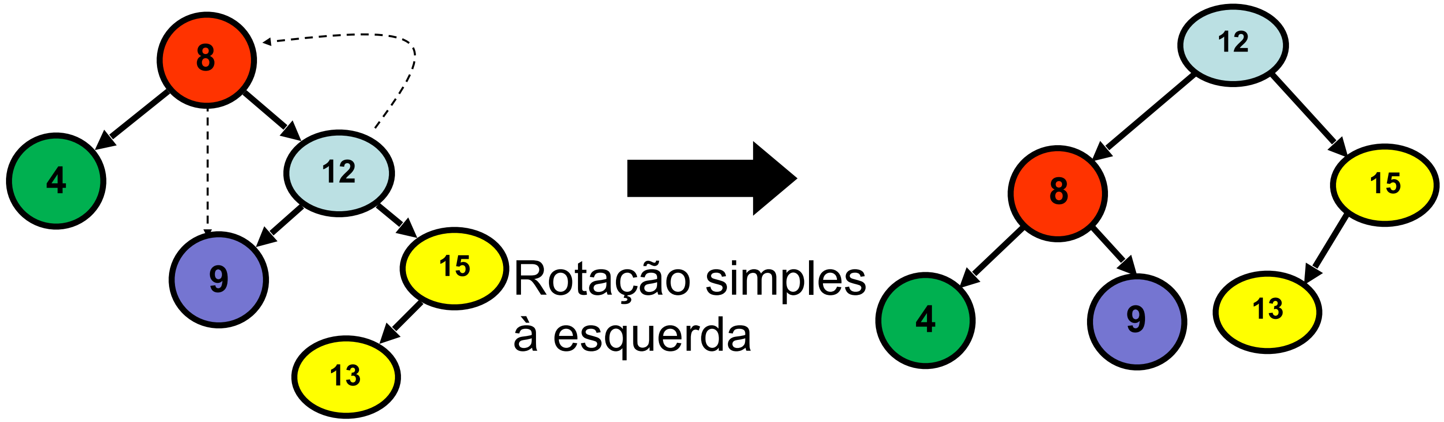
FB(4) = 1



Inserção - Caso 1b

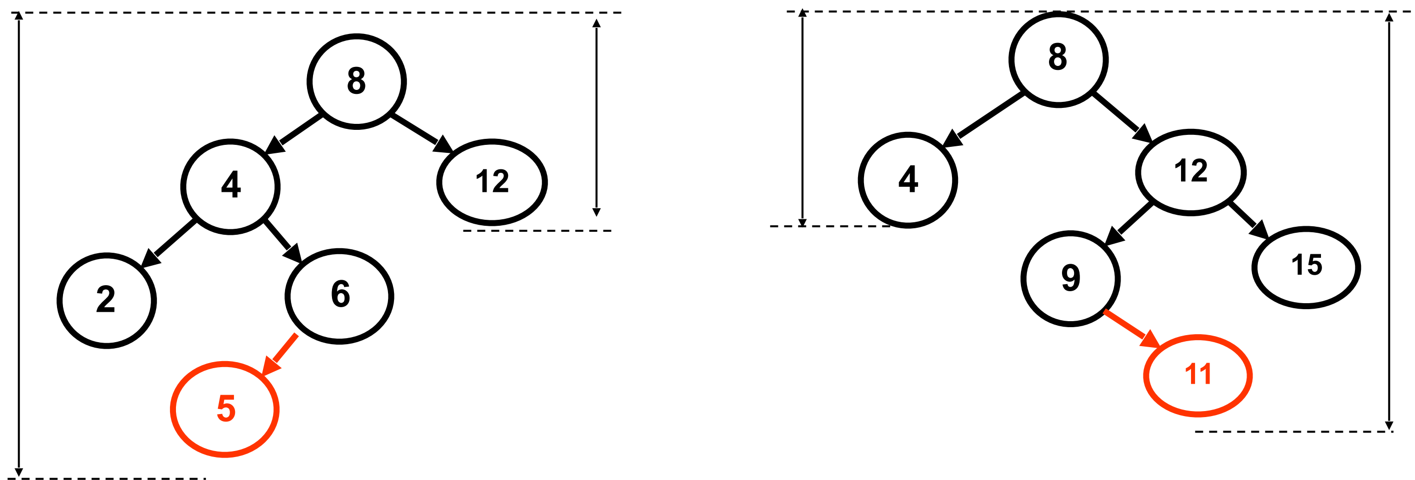
FB(8) = -2

FB(12) = -1



Inserção - Caso 2

- *Nó raiz da subárvore tem FB=2 (ou -2) e tem filho com FB=-1 (ou 1) o qual tem o sinal <u>oposto</u> ao do FB do nó pai*

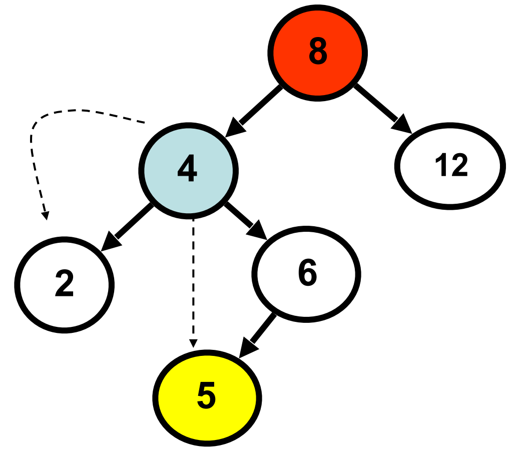


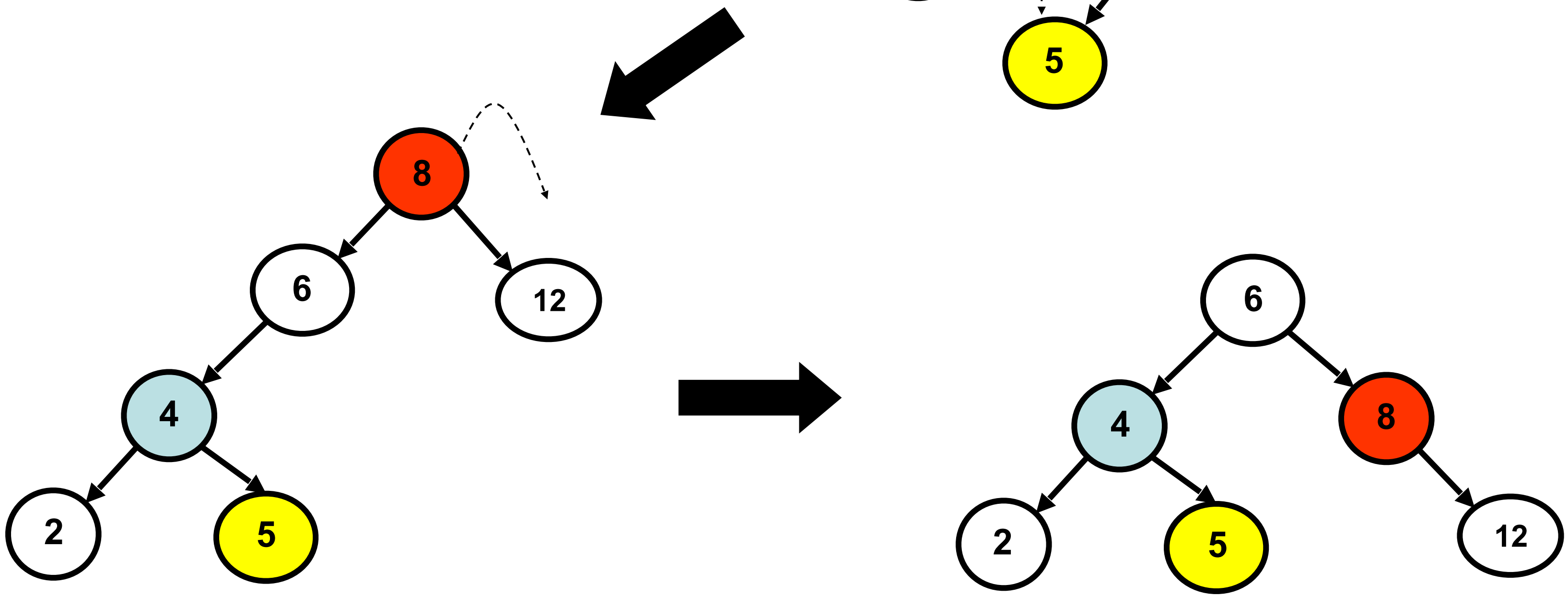
- <u>Solução</u>: rotação <u>dupla</u>
  - Primeiro a rotação sobre o nó com FB=1 (-1) na direção apropriada;
  - Em seguida, a rotação sobre o nó com FB=2 na direção oposta.

Inserção - Caso 2

FB(8) = 2

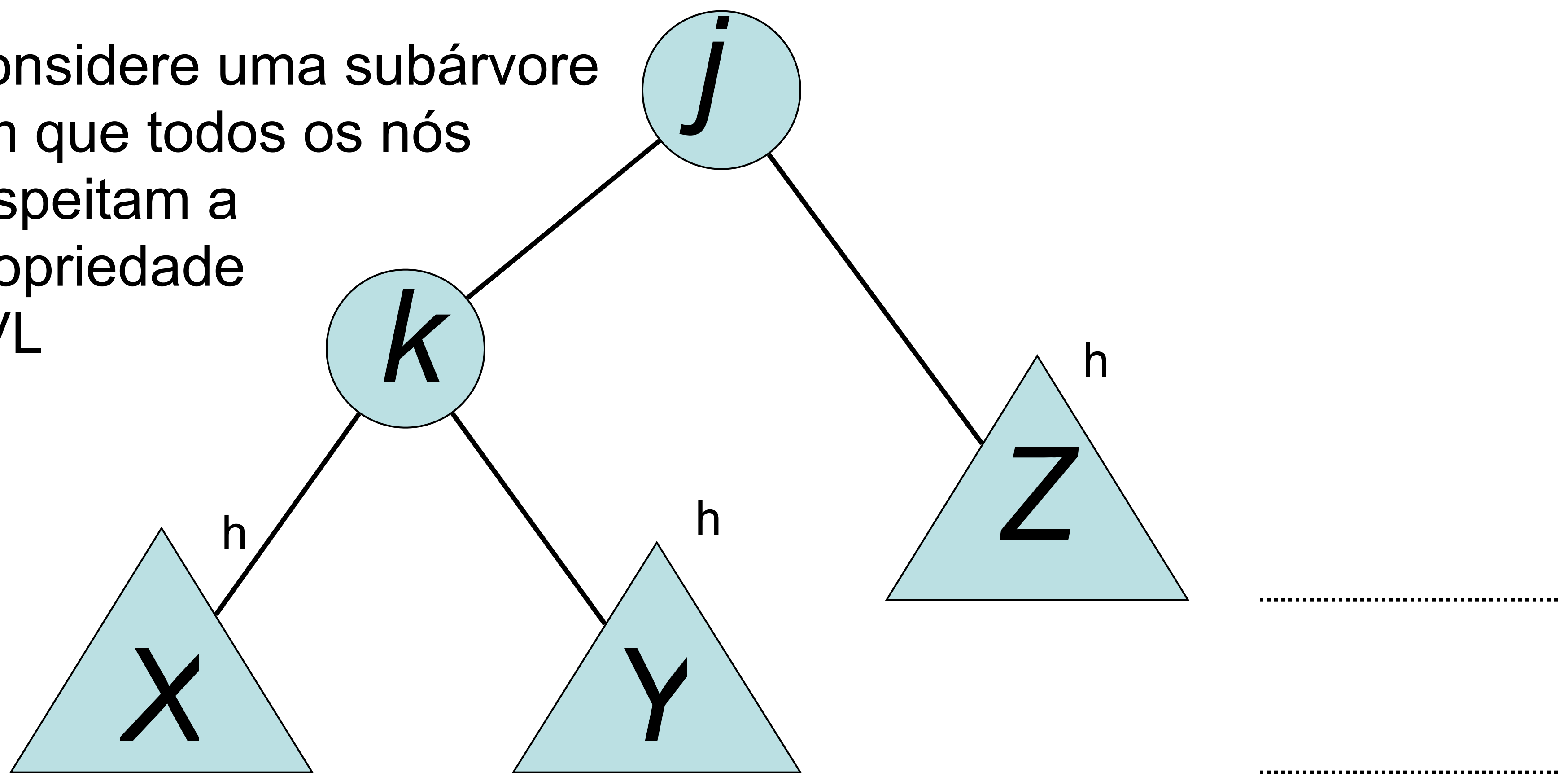
FB(4) = -1



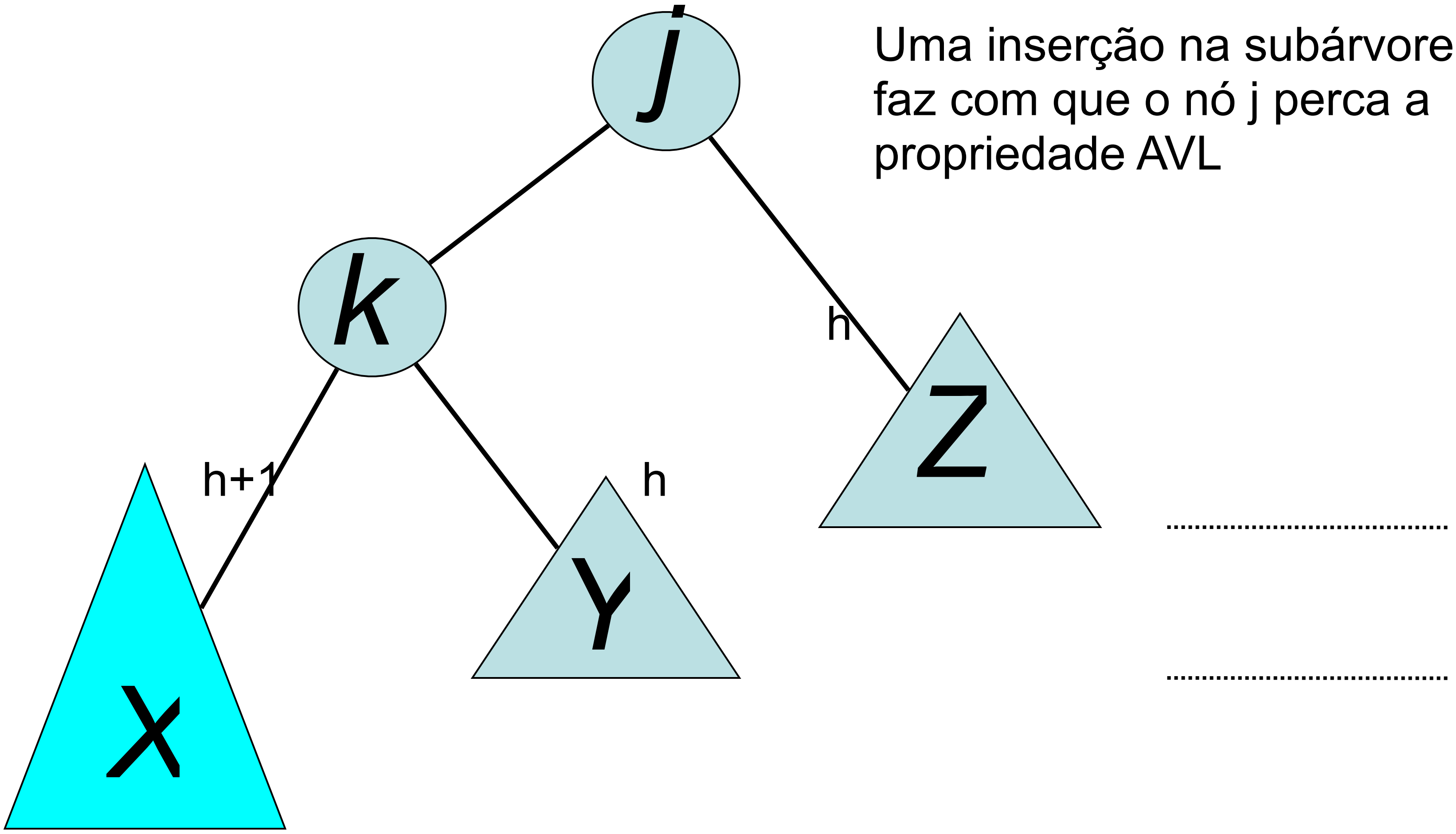


Generalização - Caso 1

Considere uma subárvore em que todos os nós respeitam a propriedade AVL

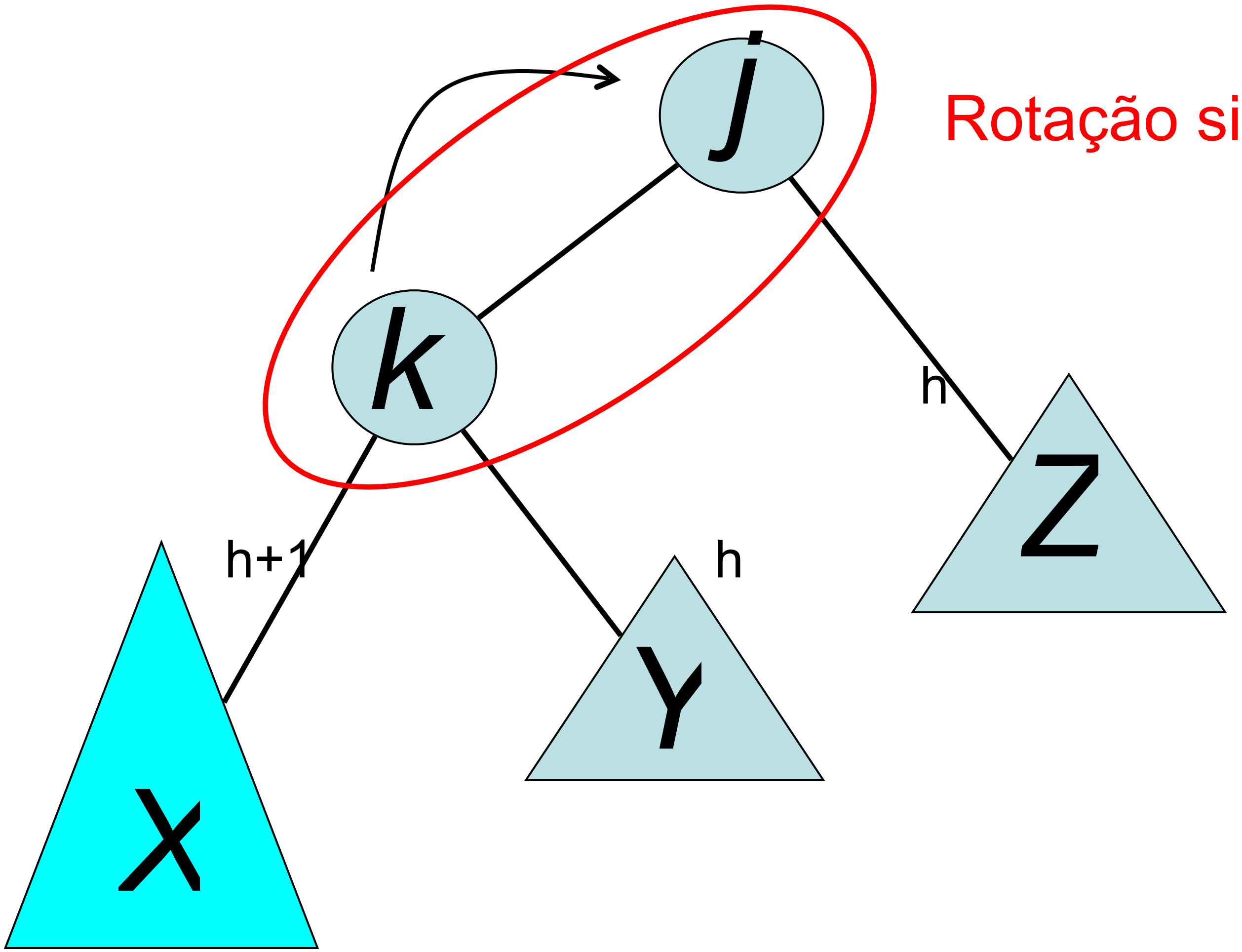


Generalização - Caso 1



Uma inserção na subárvore X faz com que o nó j perca a propriedade AVL

Generalização - Caso 1



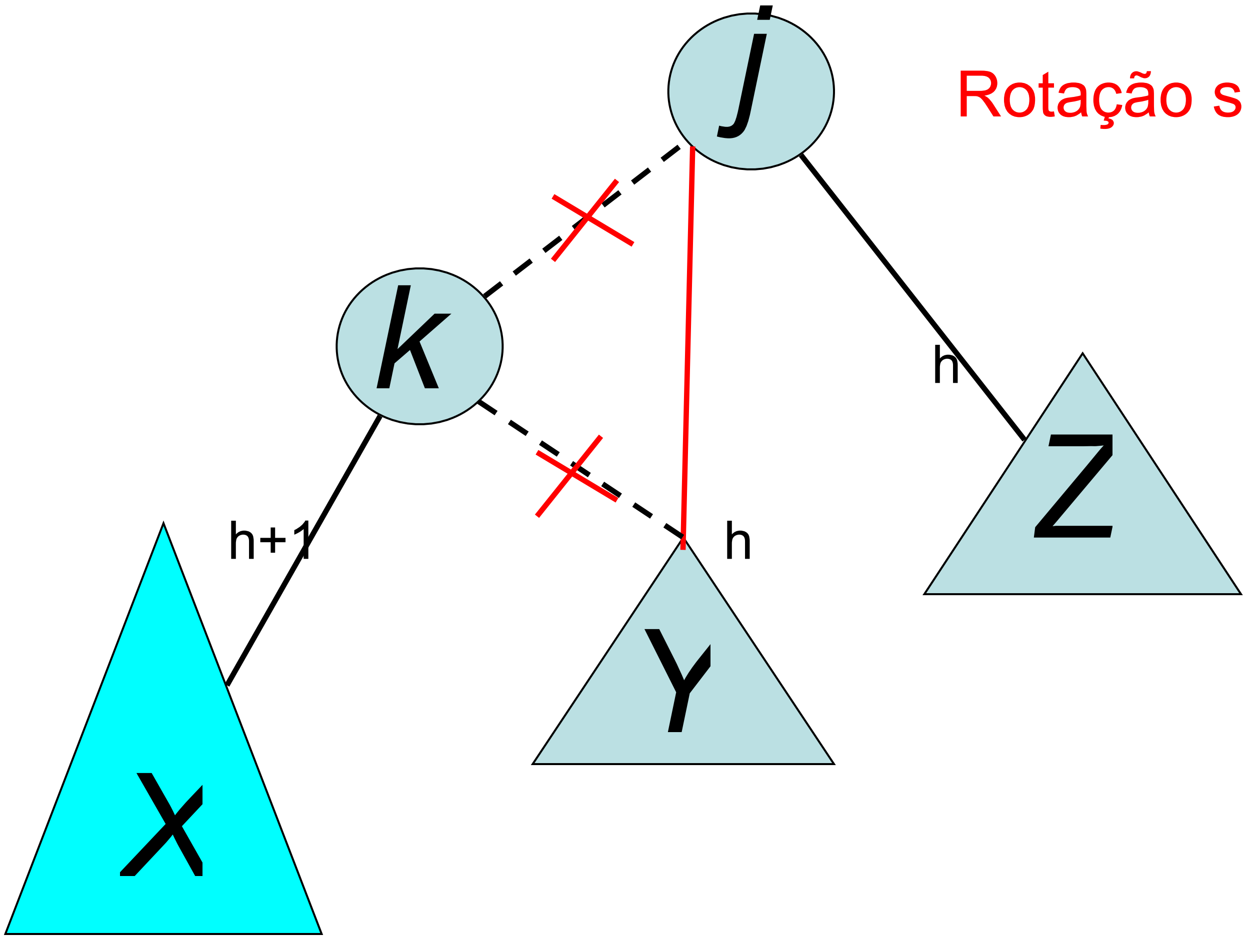
Rotação simples à direita

....

....

....

Generalização - Caso 1



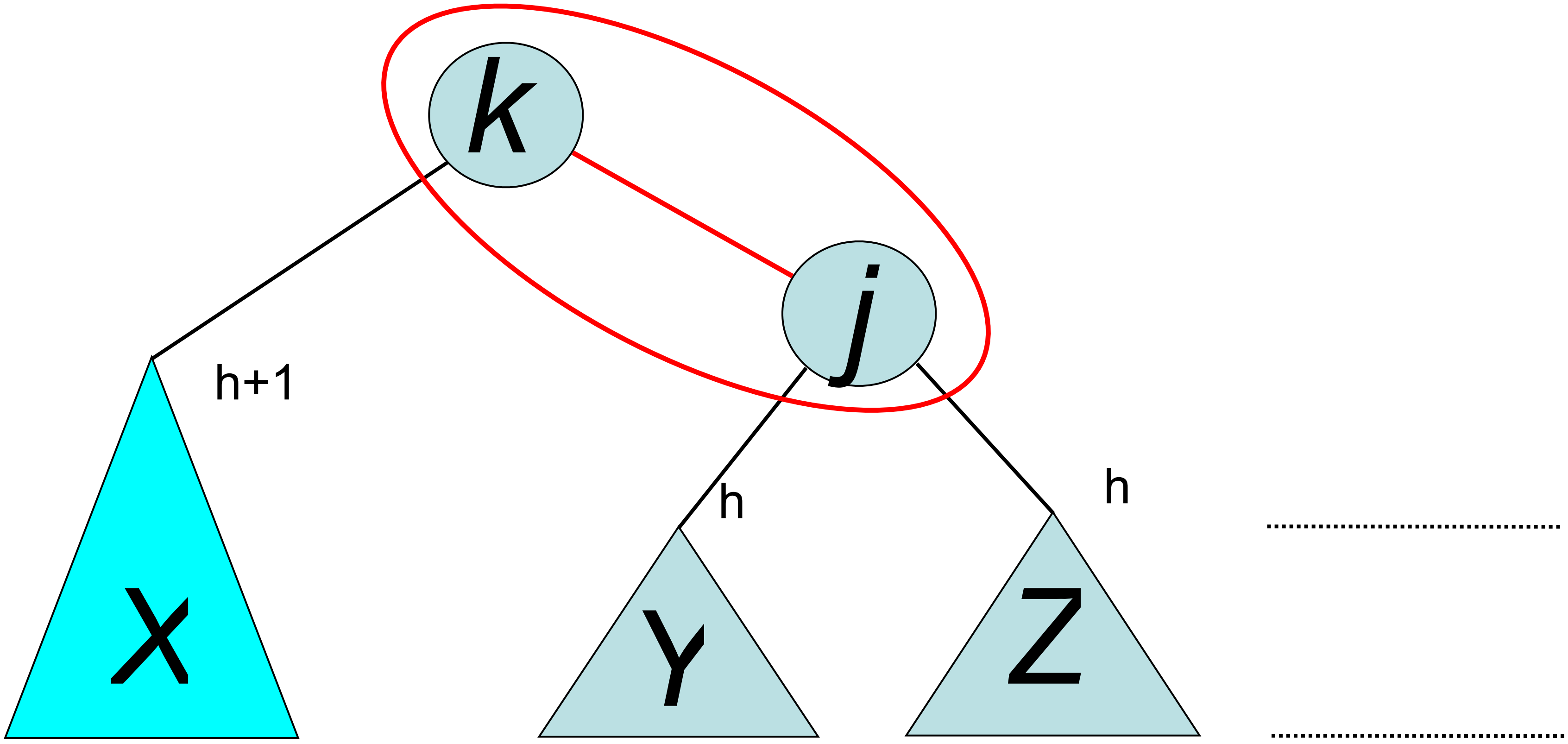
Rotação simples à direita

....

....

....

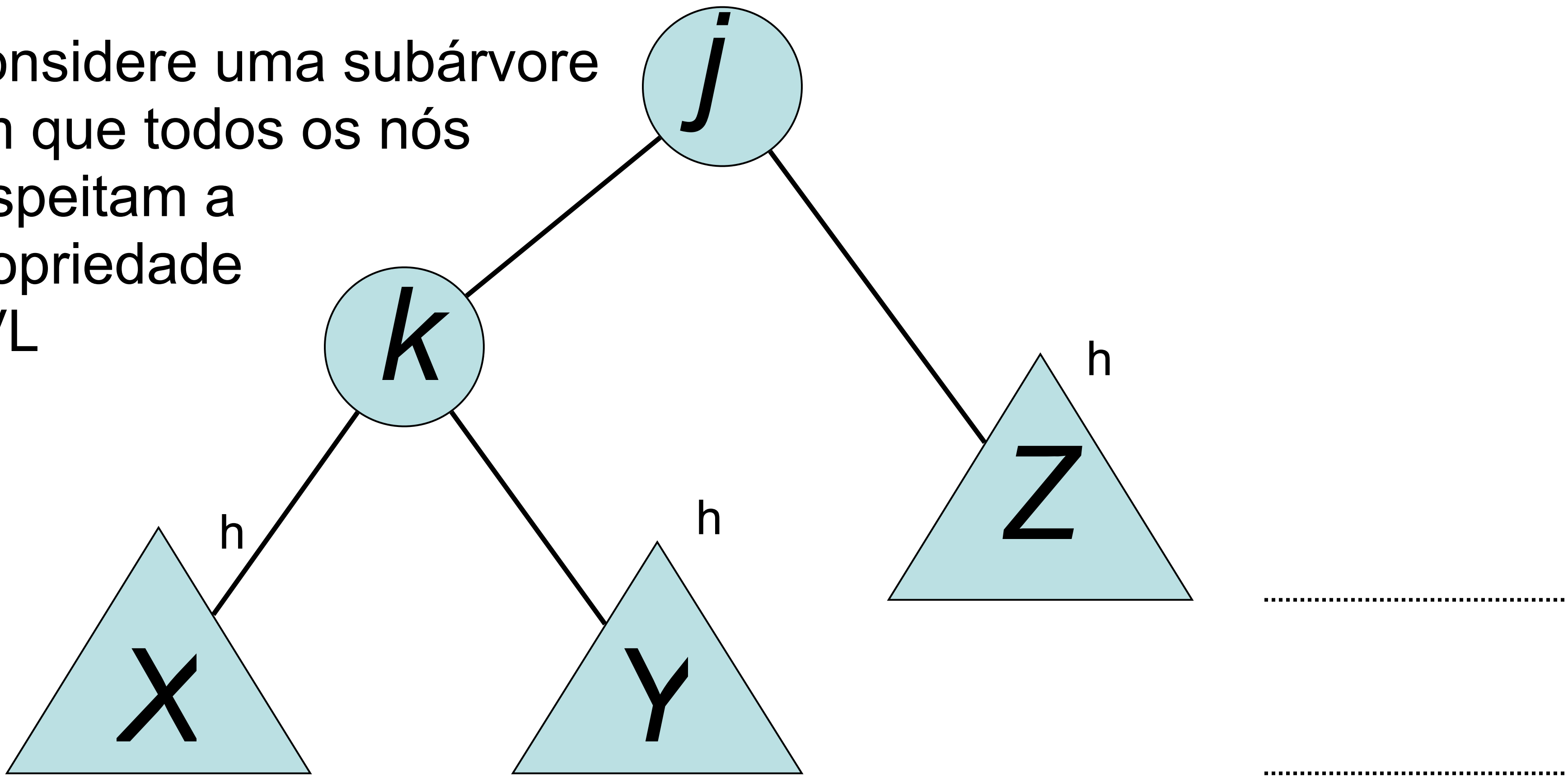
Generalização - Caso 1



Propriedade AVL foi restaurada!

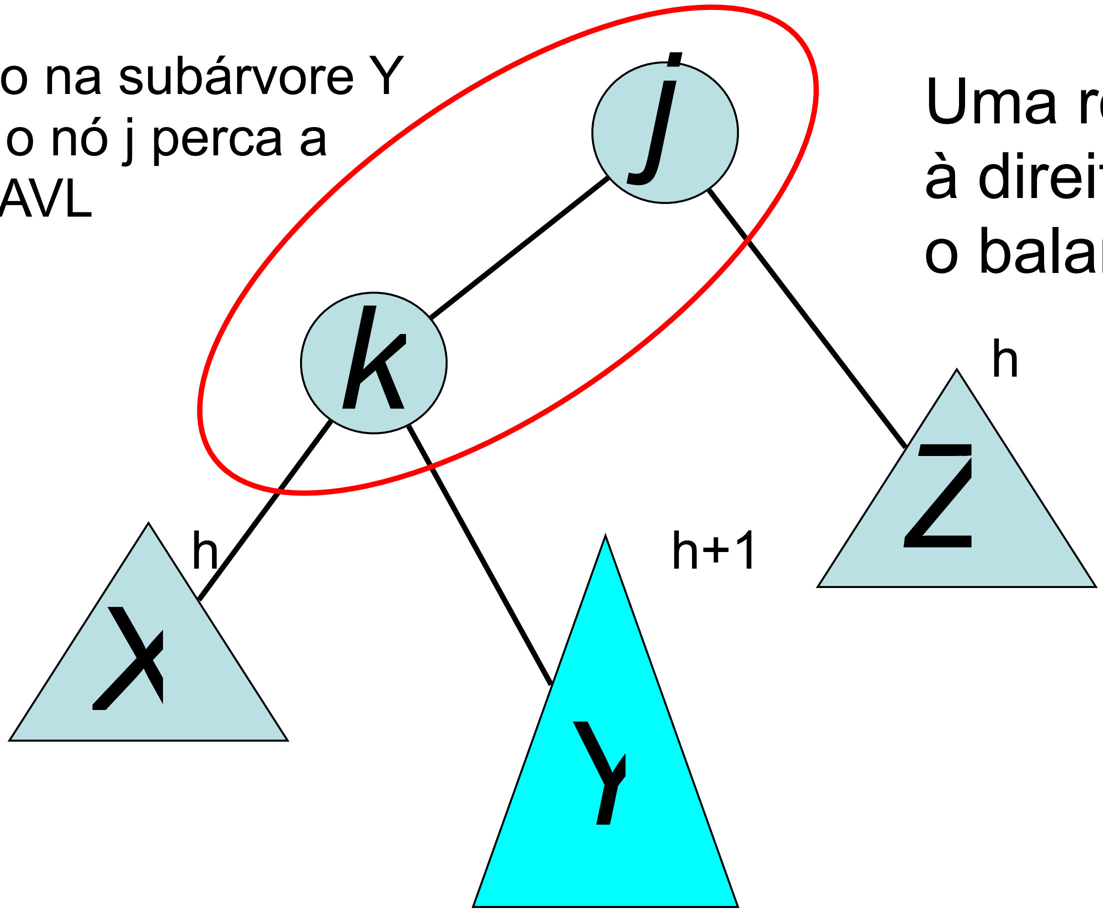
Generalização - Caso 2

Considere uma subárvore em que todos os nós respeitam a propriedade AVL



Inserção - Caso 2

Uma inserção na subárvore Y faz com que o nó j perca a propriedade AVL



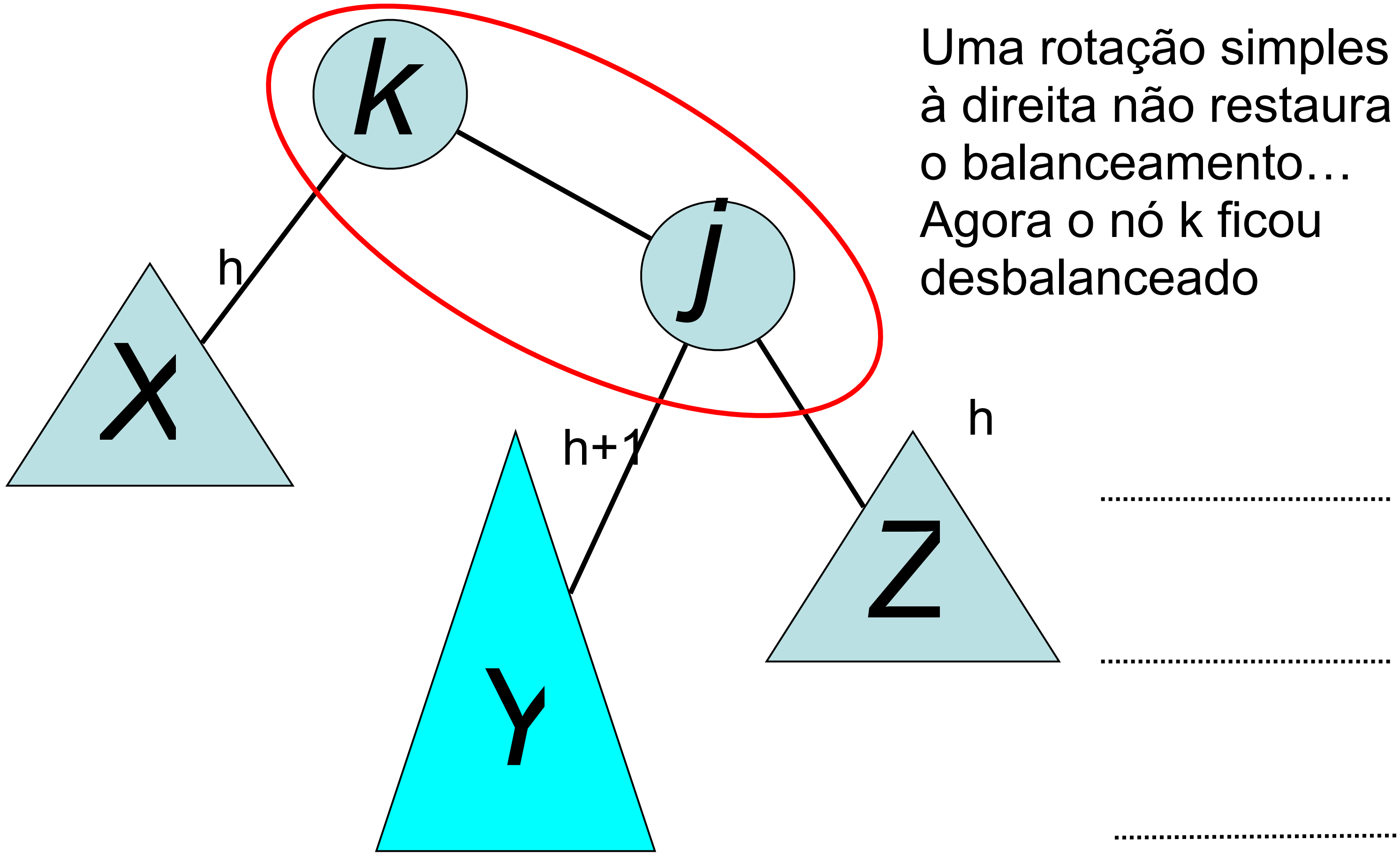
Uma rotação simples à direita restaura o balanceamento?

....

....

....

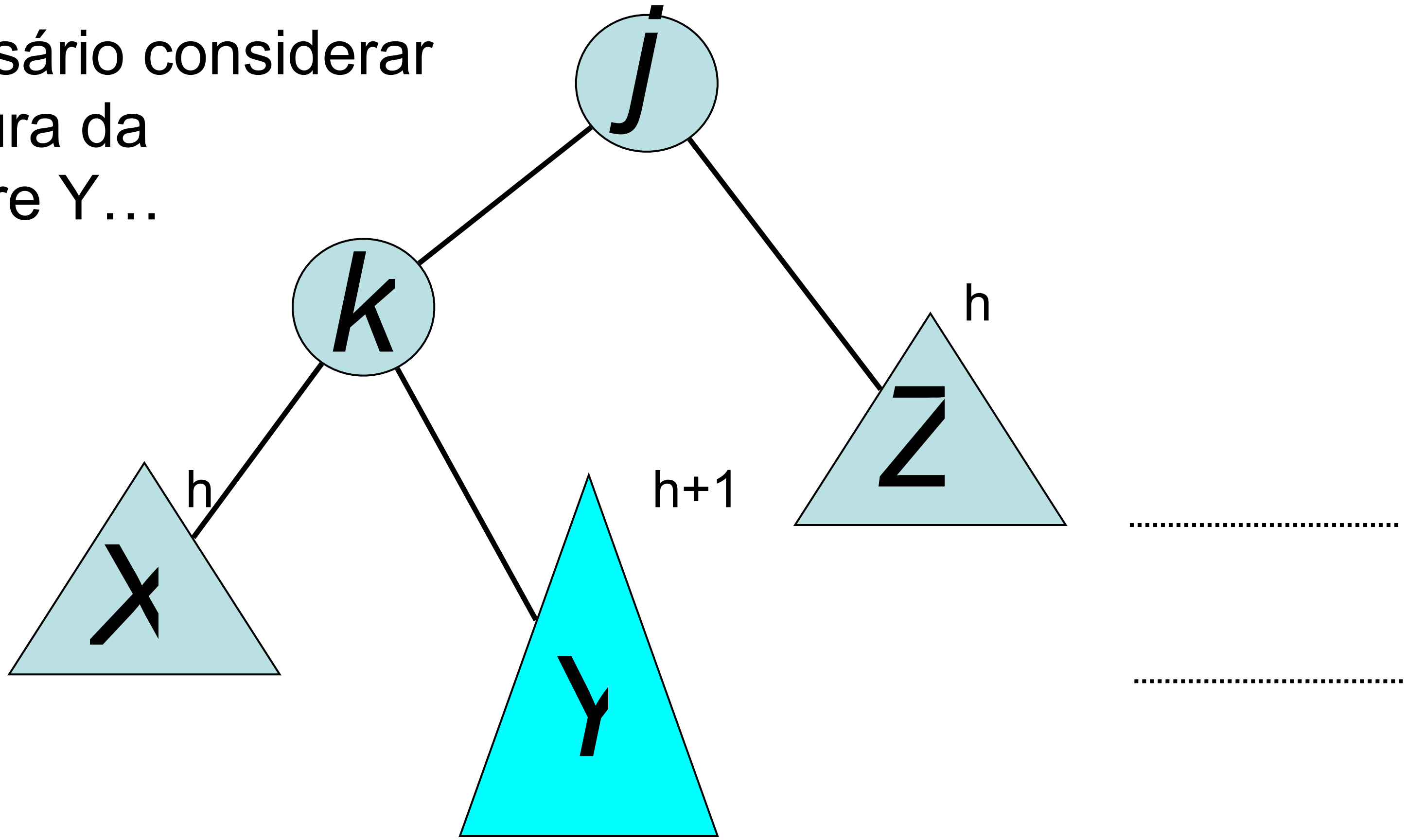
Inserção - Caso 2



Uma rotação simples à direita não restaura o balanceamento… Agora o nó k ficou desbalanceado

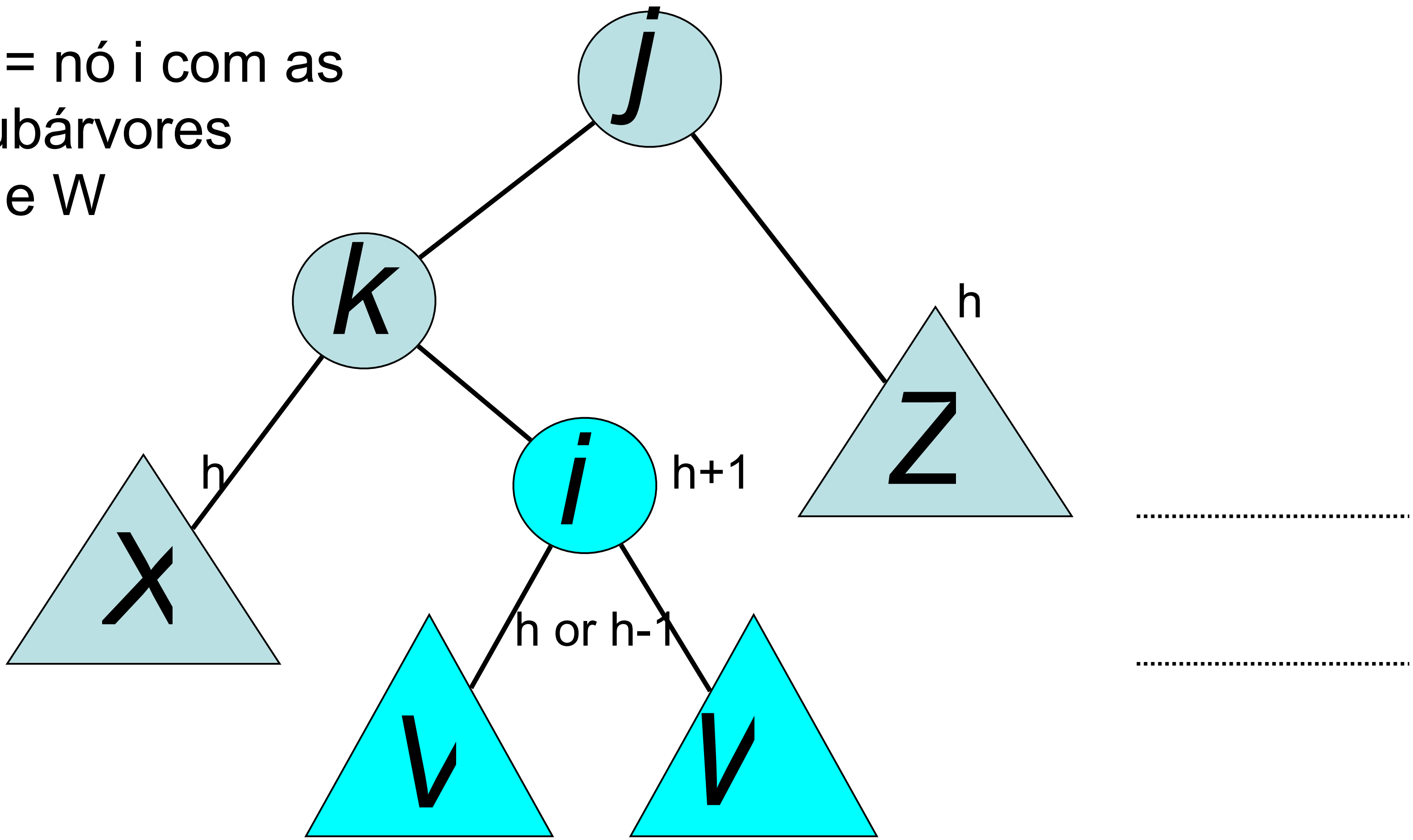
Inserção - Caso 2

É necessário considerar a estrutura da subárvore Y…

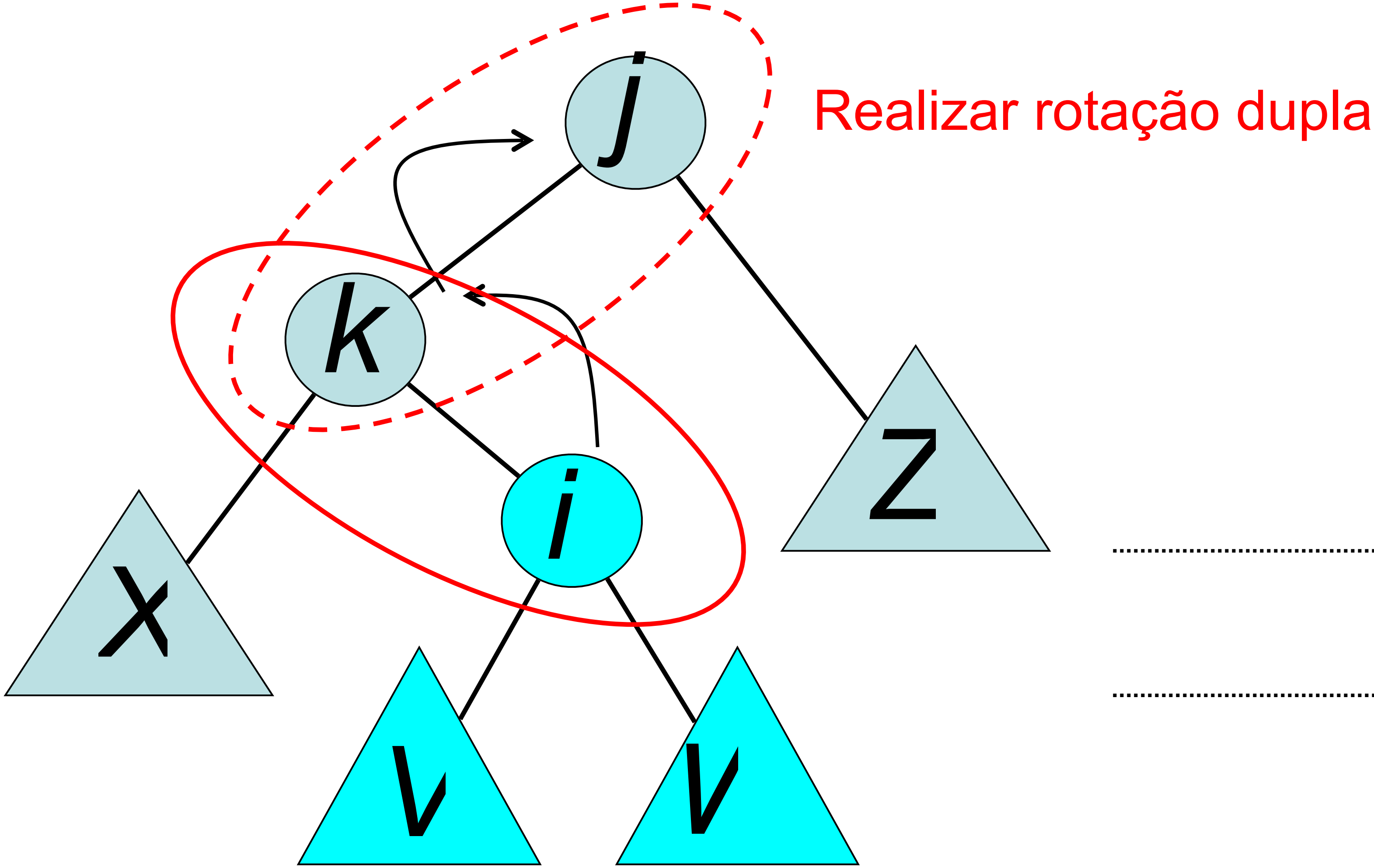


Inserção - Caso 2

Y = nó i com as subárvores V e W

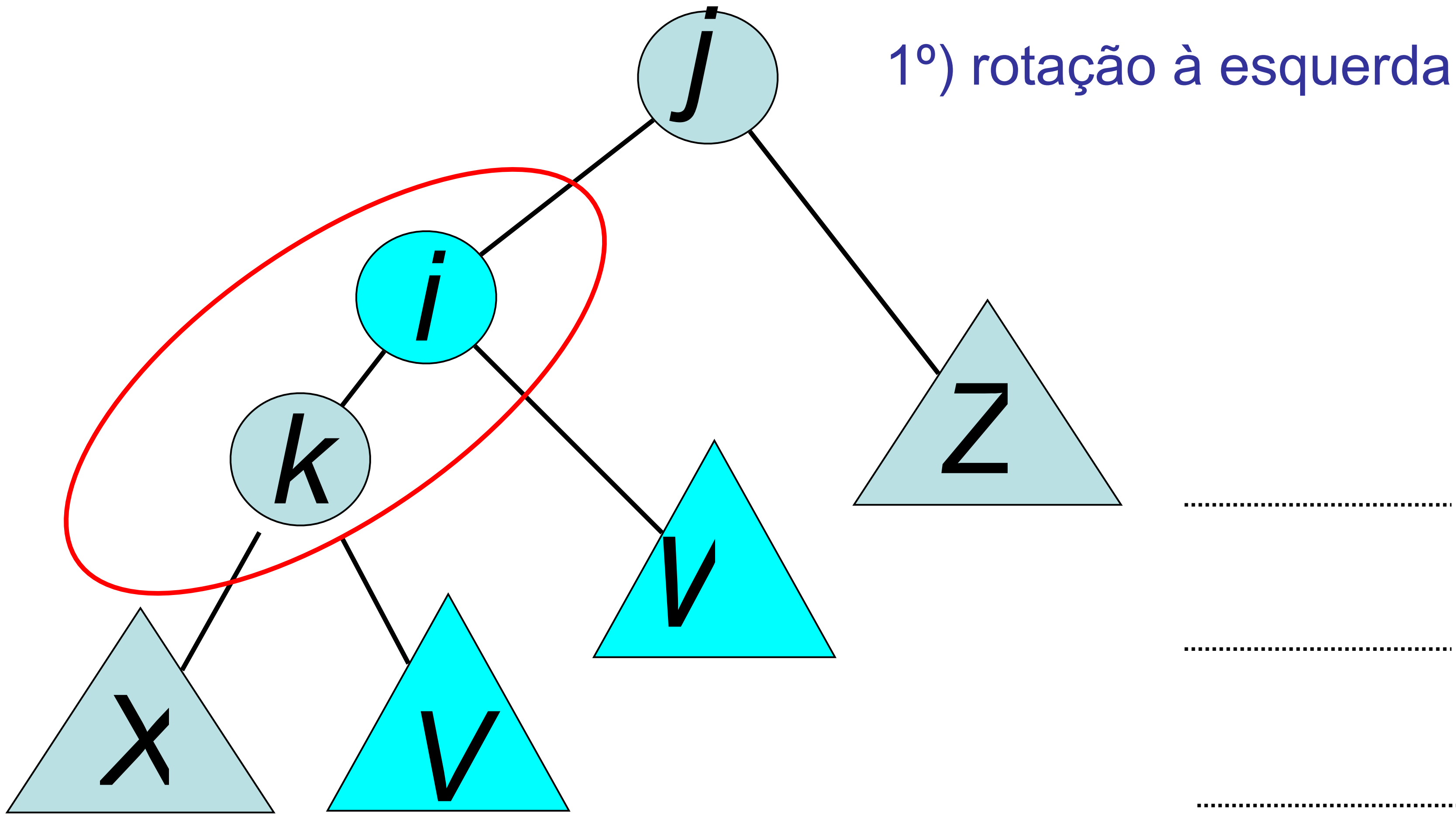


Inserção - Caso 2

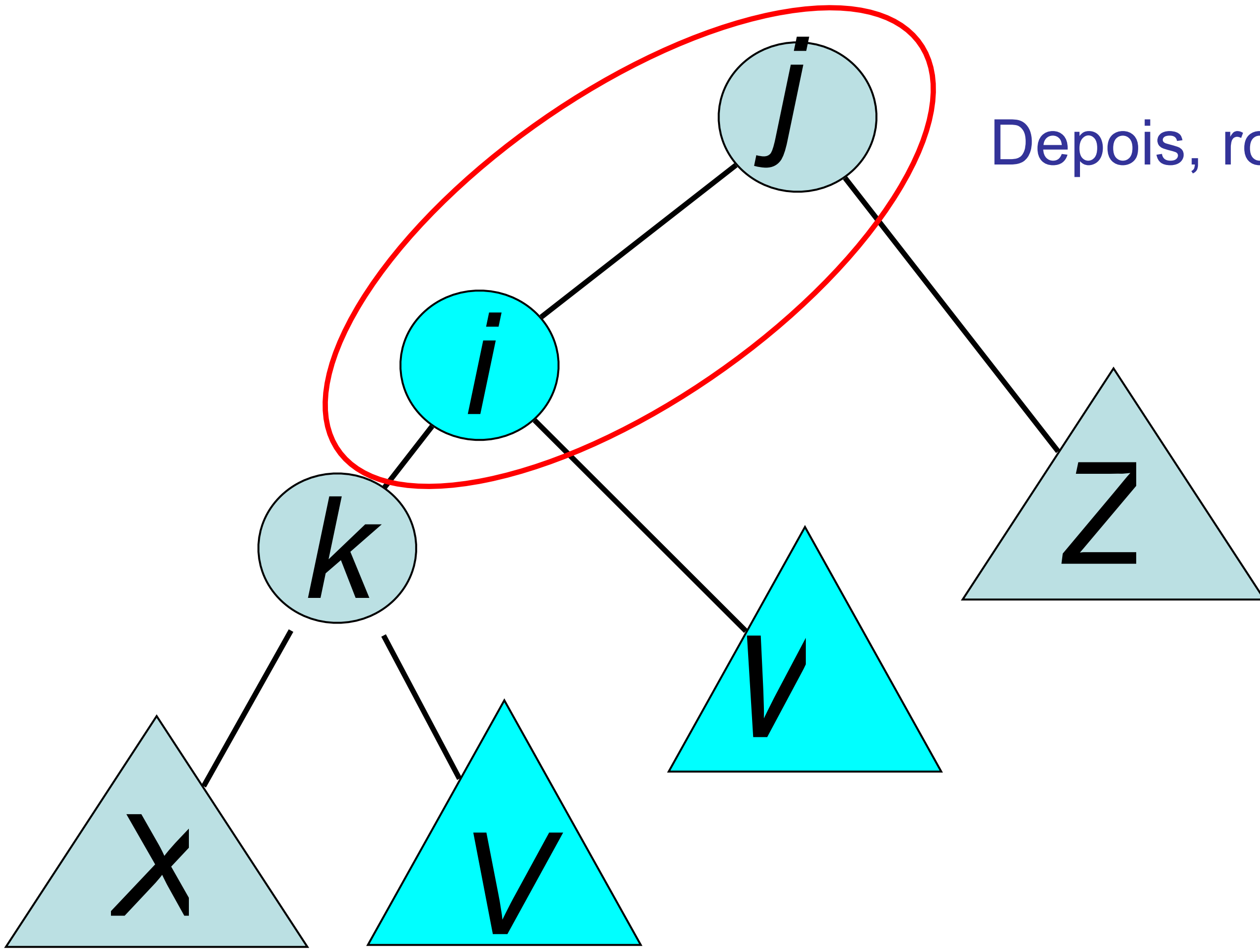


Inserção - Caso 2

1º) rotação à esquerda



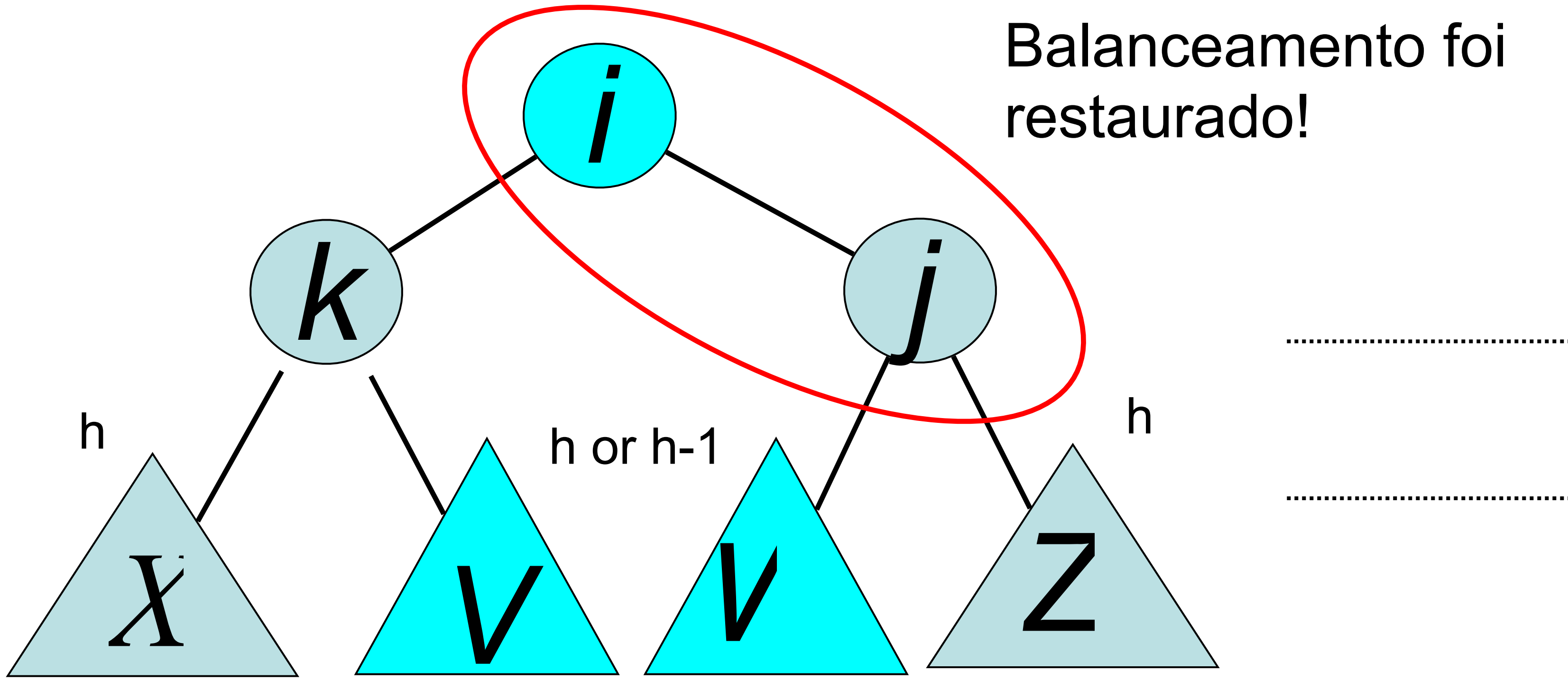
Inserção - Caso 2



Depois, rotação à direita

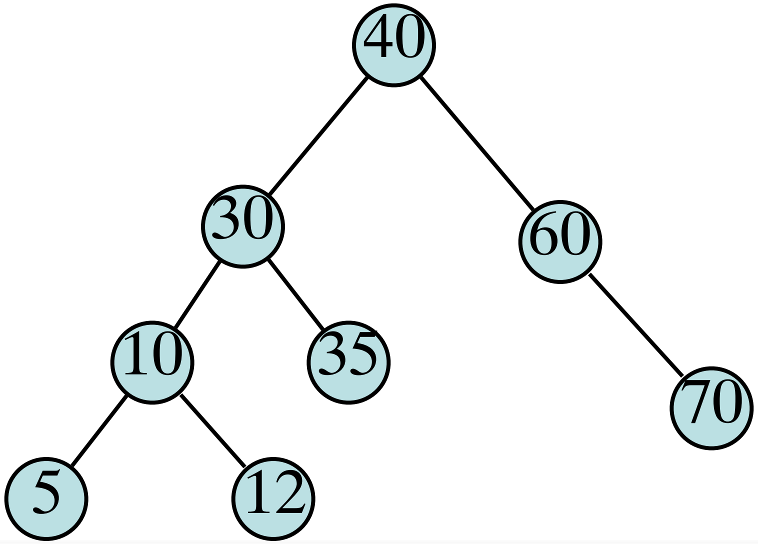
Inserção - Caso 2

Balanceamento foi restaurado!



Árvores AVL

- Exemplo
  - Inserir na árvore AVL abaixo os seguintes elementos: 3,33,11 e 9



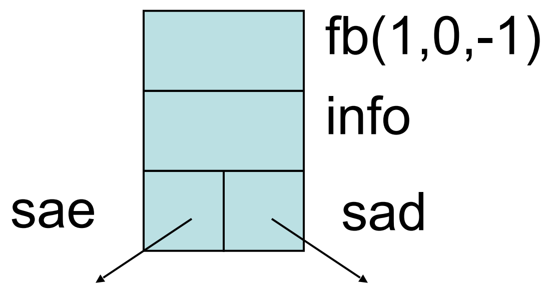
Árvores AVL

- Exemplo
  - Inserir na árvore AVL inicialmente vazia os seguintes elementos: 10,20,30,40,50,25,60,70,80 e 90

Árvores AVL

- Exemplo
  - Inserir na árvore AVL inicialmente vazia os seguintes elementos: 10,20,30,40,50,25,60,70,80 e 90

Implementação



Não há necessidade de armazenar a altura, apenas a diferença de altura, ou seja, o fator de balanceamento (fb), que tem que ser modificado no caminho de inserção, mesmo se não sejam realizadas rotações

Depois de ser realizada uma rotação (simples ou dupla), não é necessário voltar ao topo da árvore

Implementação da inserção

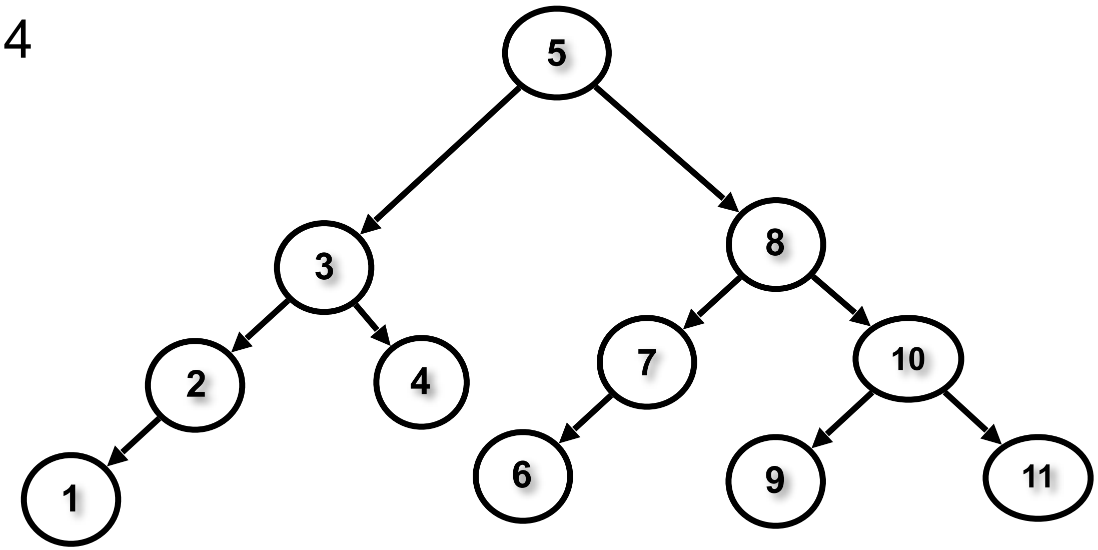
- Inserir como folha
  - apenas os nós no caminho da raiz até o ponto de inserção podem ter mudança na altura
  - Assim, após a inserção, retorna-se até a raiz, nó por nó, atualizando alturas
  - Se o fator de balanceamento atualizado de um nó é de 2 ou -2, ajustar árvore realizando a rotação necessária em torno do nó

Remoções

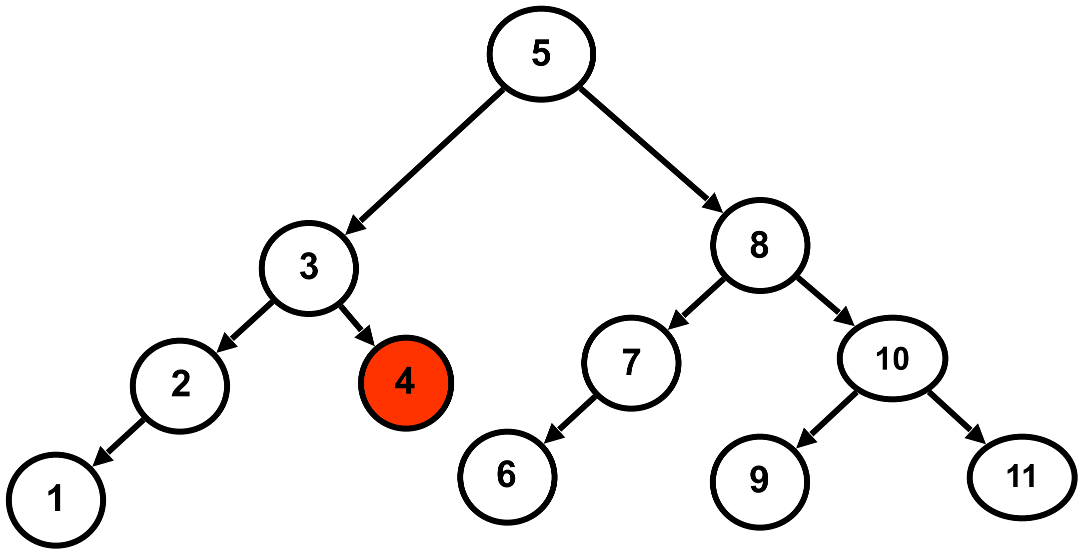
- Os problemas são semelhantes aos das inserções: podem ocorrer desbalanceamentos
- Existem duas situações possíveis:
  – Caso 1: simples, quando o nó removido é uma folha ou tem apenas 1 descendente.
  – Caso 2: o nó removido possui as duas subárvores.

Exemplo

Remover 4



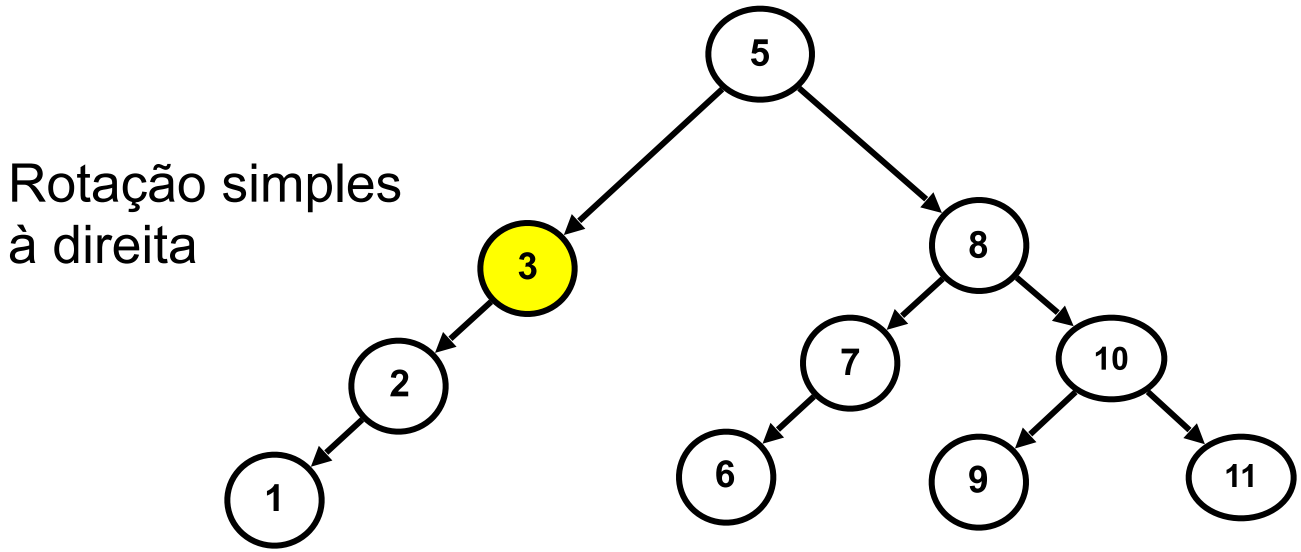
Exemplo



Caso 1

- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

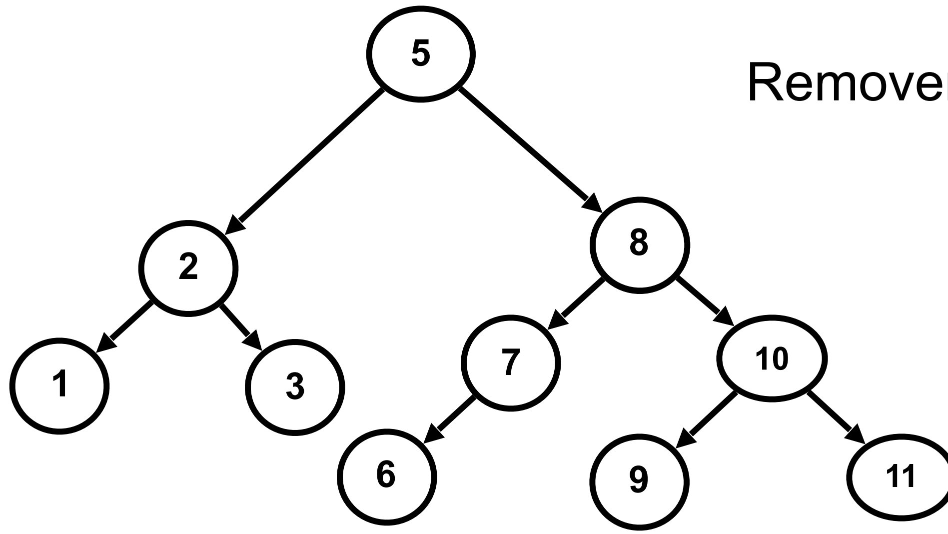
Exemplo



Caso 1

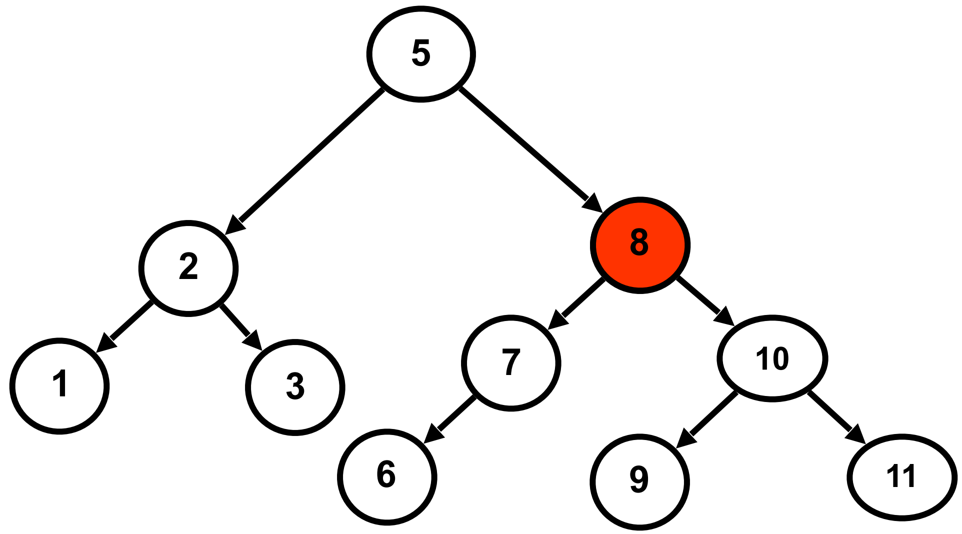
- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

Exemplo



Remover 8

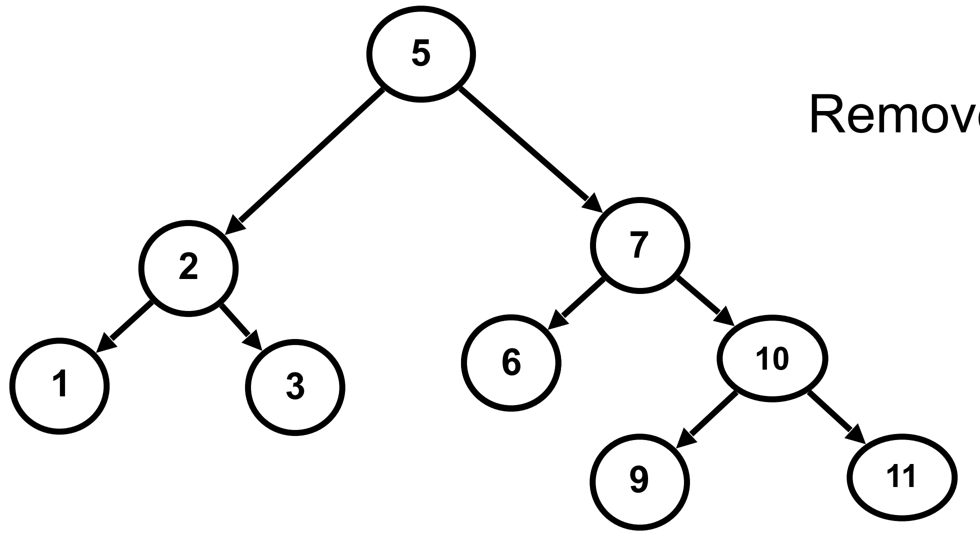
Exemplo



Caso 2

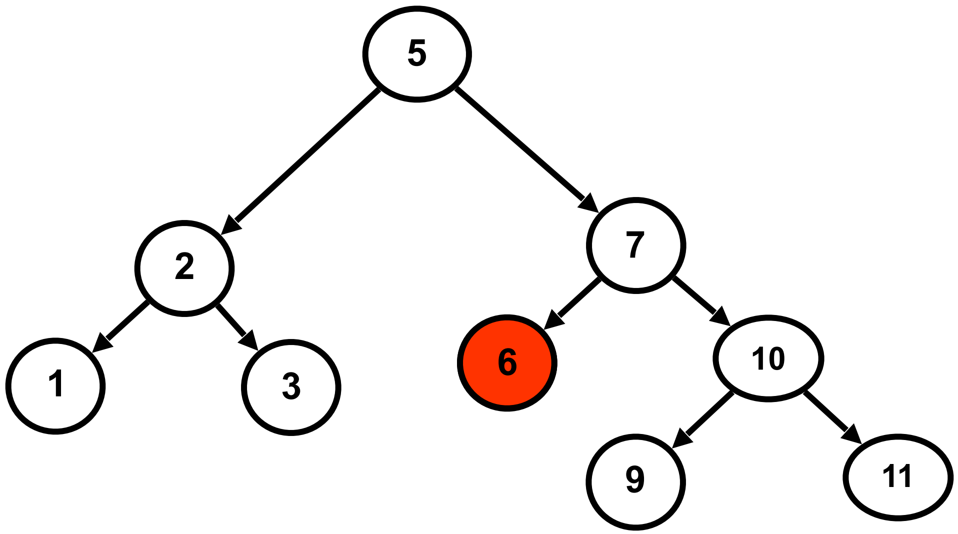
- Troca pelo maior valor a esquerda (ou menor à direita)
- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

Exemplo



Remover 6

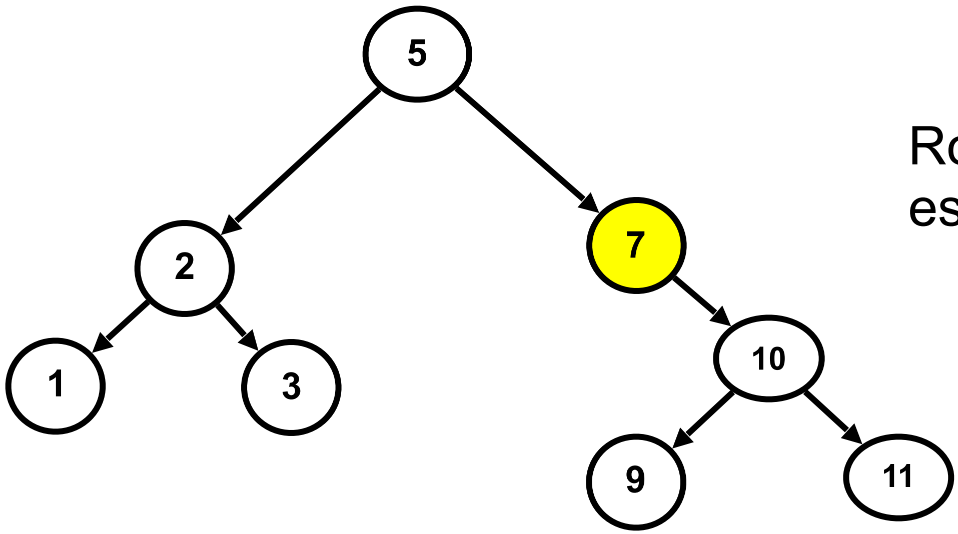
Exemplo



Caso 1

- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

Exemplo



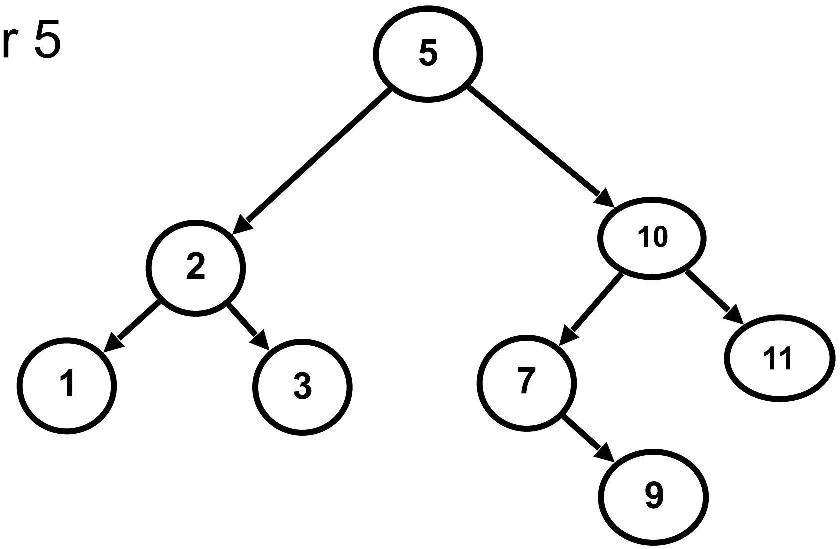
Rotação à
esquerda

Caso 1

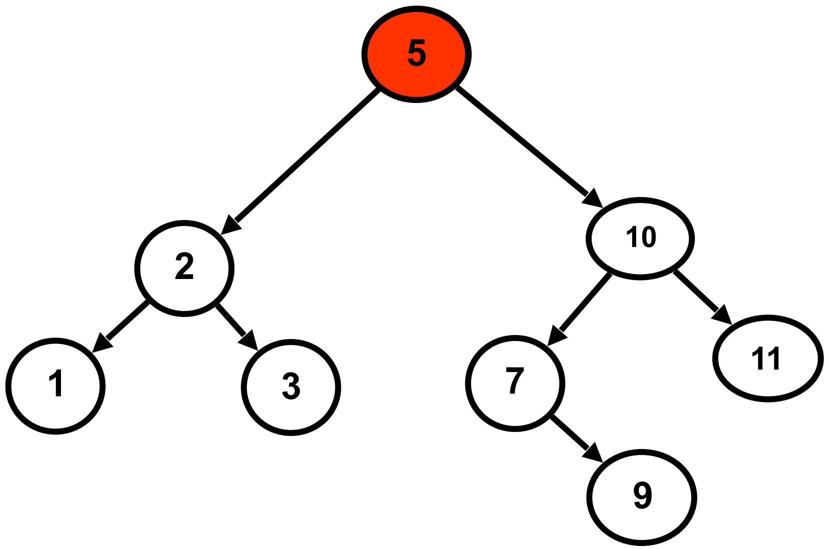
- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

Exemplo

Remover 5



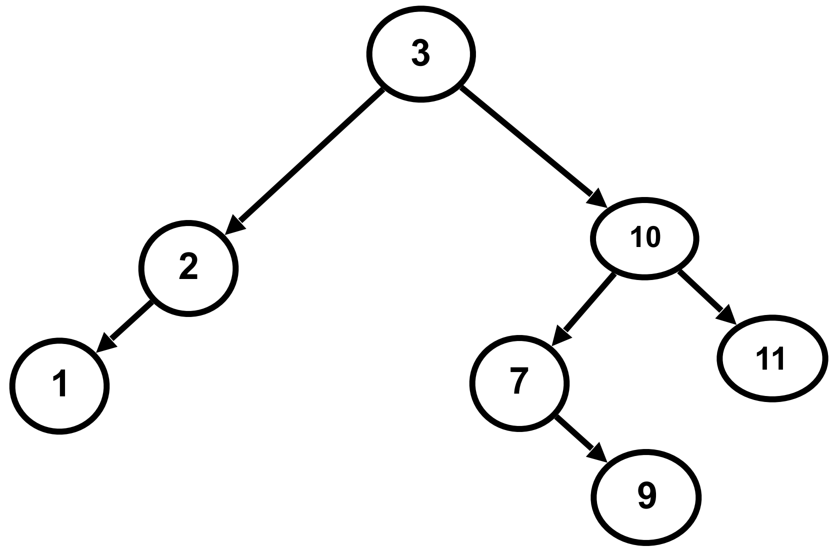
Exemplo



Caso 2

- Troca pelo maior valor a esquerda (ou menor à direita)
- Remove o nó
- Percorre a árvore, de baixo pra cima, balanceando a subárvore quando necessário

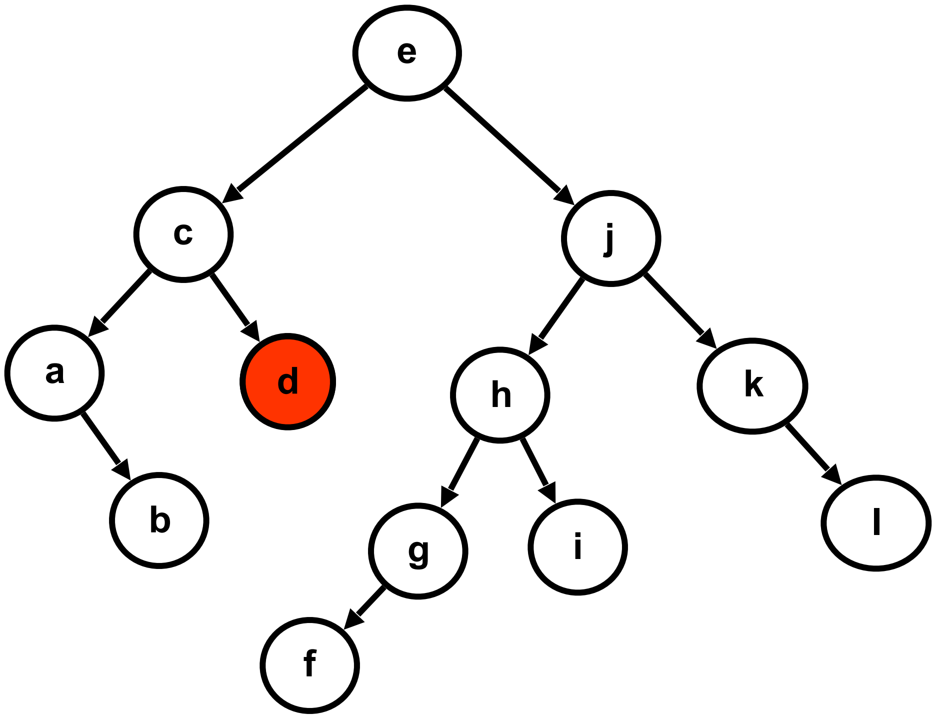
Exemplo



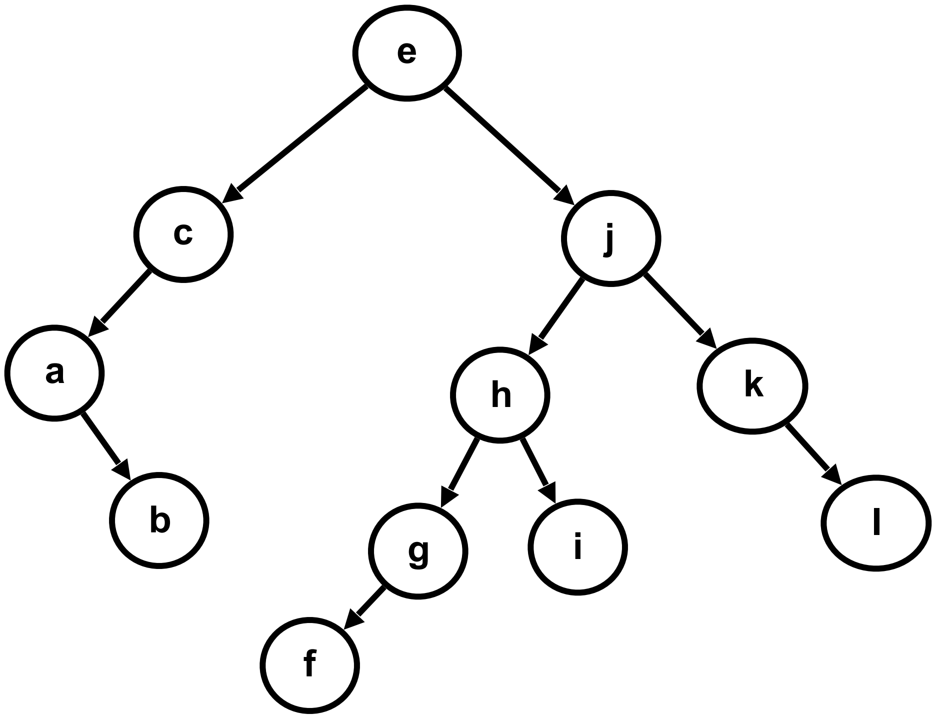
Problemas com as Árvores AVL

- Nem sempre uma rotação simples (ou uma dupla) resolve o problema de desbalanceamento dos nós
- Podem existir casos em que o número de rotações exigido seja $O(\log n)$ para tornar a árvore balanceada...
- Veja o exemplo a seguir, onde após a exclusão do nó *d*, diversas rotações são executadas na árvore...

Excesso de Rotações na Árvore AVL

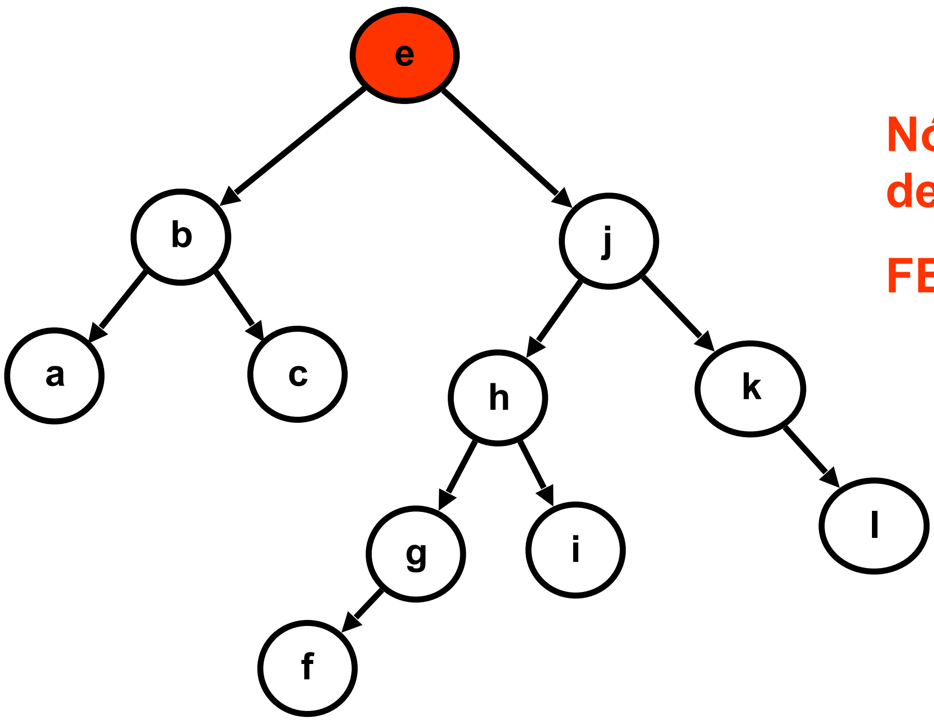


Excesso de Rotações na Árvore AVL



Rotação dupla esq(a)+dir(c)

Excesso de Rotações na Árvore AVL

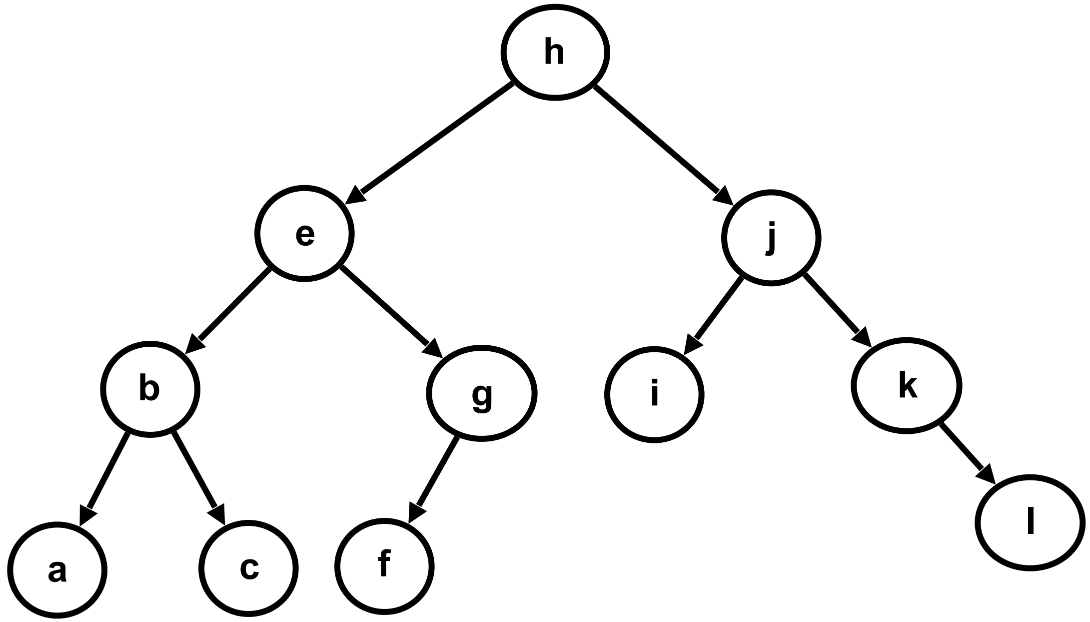


Nó e fica desbalanceado

FB(e) = -2

Excesso de Rotações na Árvore AVL

Rotação simples:
esq(e)



Prós e contras de Árvores AVL

Prós:

1. Busca é O(log n) pois AVLs são sempre balanceadas
2. Inserção e remoção também são O(logn)
3. A operação de balanceamento acrescenta apenas um fator constante na complexidade de inserção

Contras:

1. Dificuldade em programar & debugar; mais espaço para armazenar o fator de balanceamento
2. Assintóticamente mais rápido, mas rebalanceamento gasta tempo
3. Maioria das grandes buscas são realizadas em sistemas de bancos de dados em discos e usam outras estruturas (ex., árvores B)

Material adaptado por Luis Martí a partir dos slides de José Viterbo Filho que forem elaborados por Marco Antonio Casanova e Marcelo Gattas para o curso de Estrutura de Dados para Engenharia da PUC-Rio, com base no livro *Introdução a Estrutura de Dados*, de Waldemar Celes, Renato Cerqueira e José Lucas Rangel, Editora Campus (2004).