Estruturas de Dados Avançadas (INF1010)

Árvores AVL

Balanceamento de ABBs

À medida que são feitas inserções e remoções, uma ABB pode ficar **desbalanceada**

→ ineficiência para buscas

O balanceamento de ABBs tem como motivação realizar buscas em $\mathbf{O(\log_n)}$

A estratégia é diminuir a diferença de altura entre a sub-árvore à esquerda e a sub-árvore à direita de cada nó

Existem diversas propostas de estruturas que se mantém **balanceadas**

Árvore Binária de Busca Balanceada

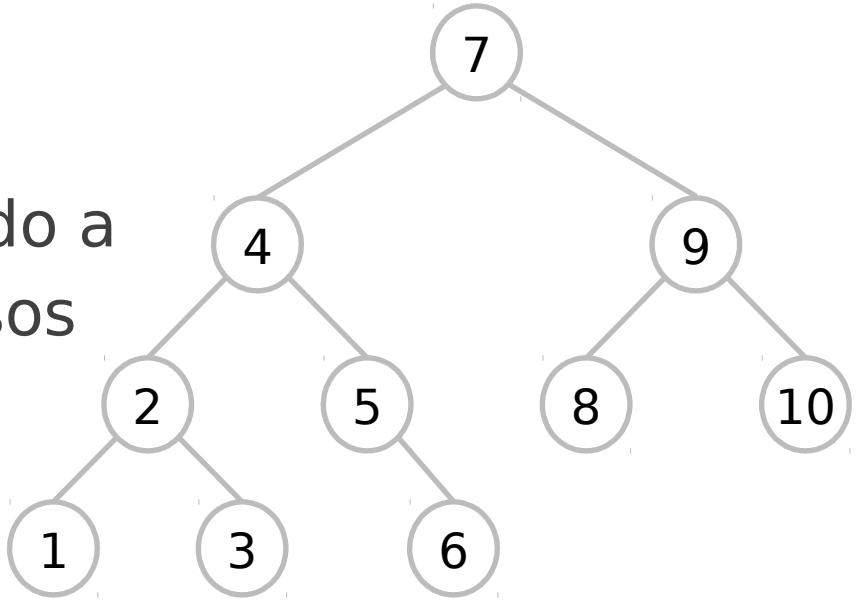
|he-hd| ≤ 1

- he = altura da sub-árvore esquerda
- hd = altura da sub-árvore direita

→ qualquer nó pode ser alcançado a partir da raiz em **O(log(n))** passos

→ (quase) todos os nós não terminais têm dois filhos

→ distribuição de nós próxima à de uma árvore cheia (altura mínima)

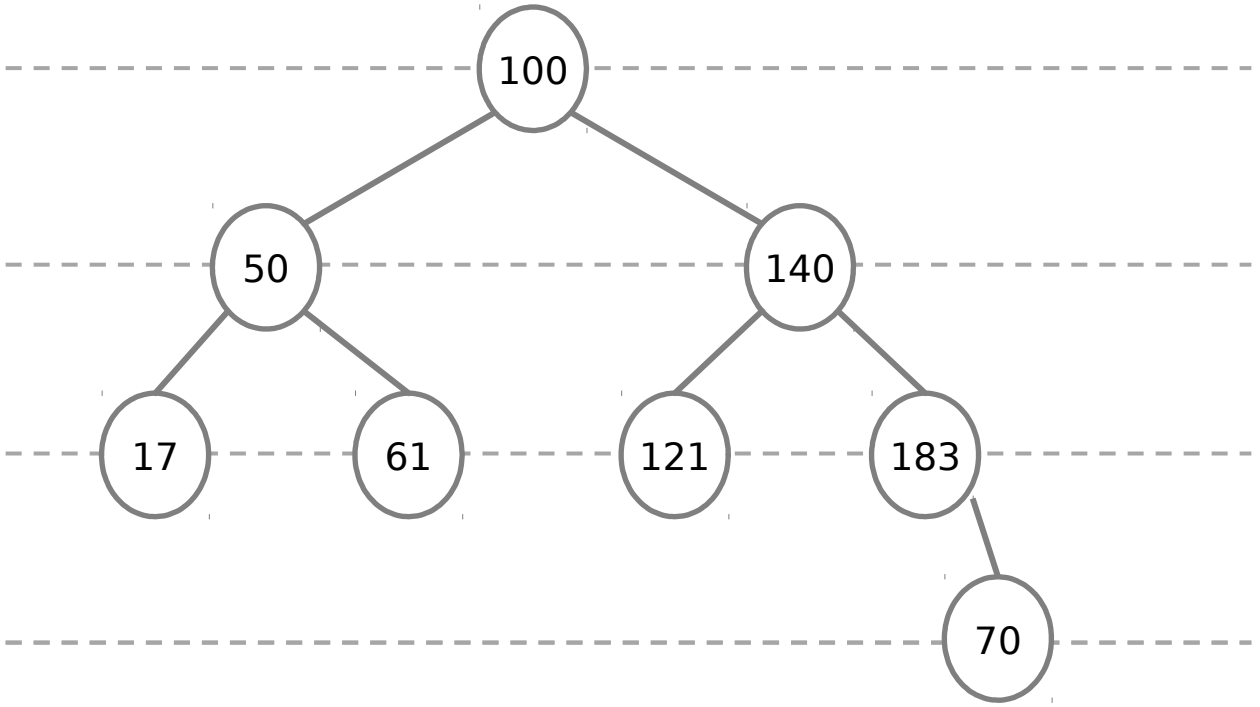


Árvores AVL

1962: AVLs
Adelson-Velsky e Landis

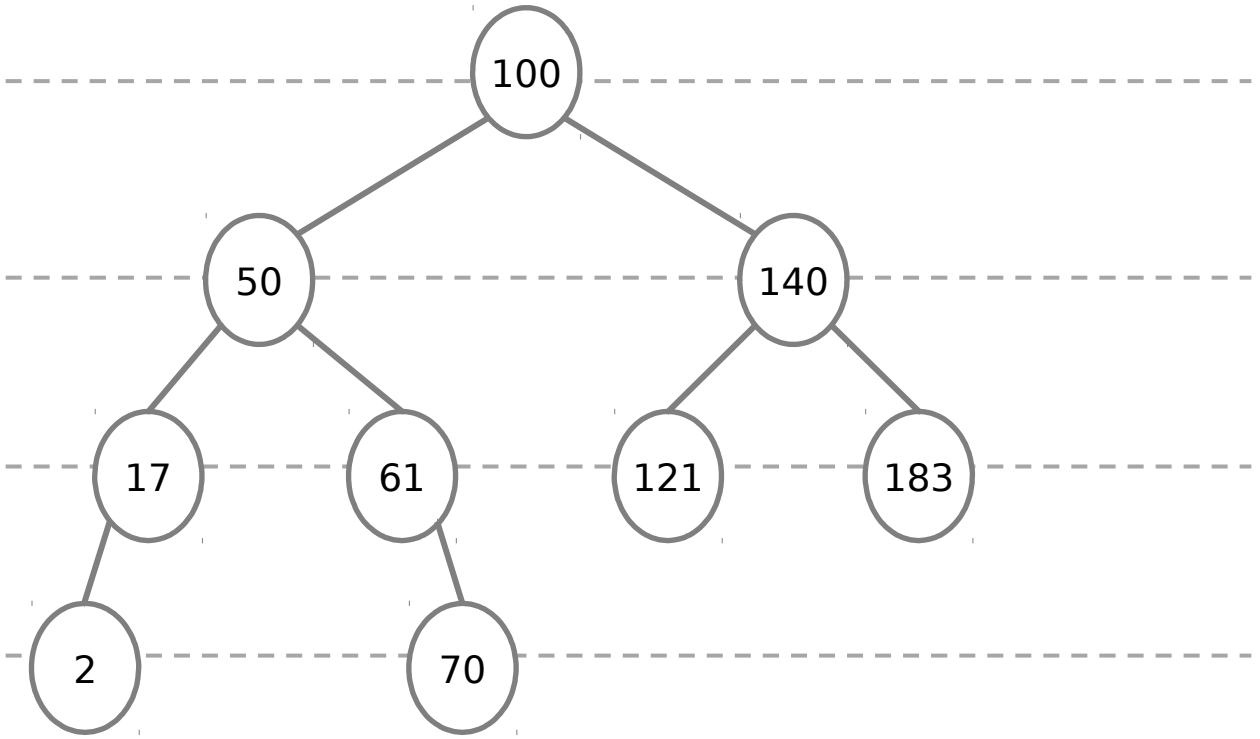
Árvore binária de busca construída de modo que, para todos os nós, a altura de sua sub-árvore à direita difere da altura de sua sub-árvore à esquerda de no máximo 1 unidade

Exemplos de Árvores AVL



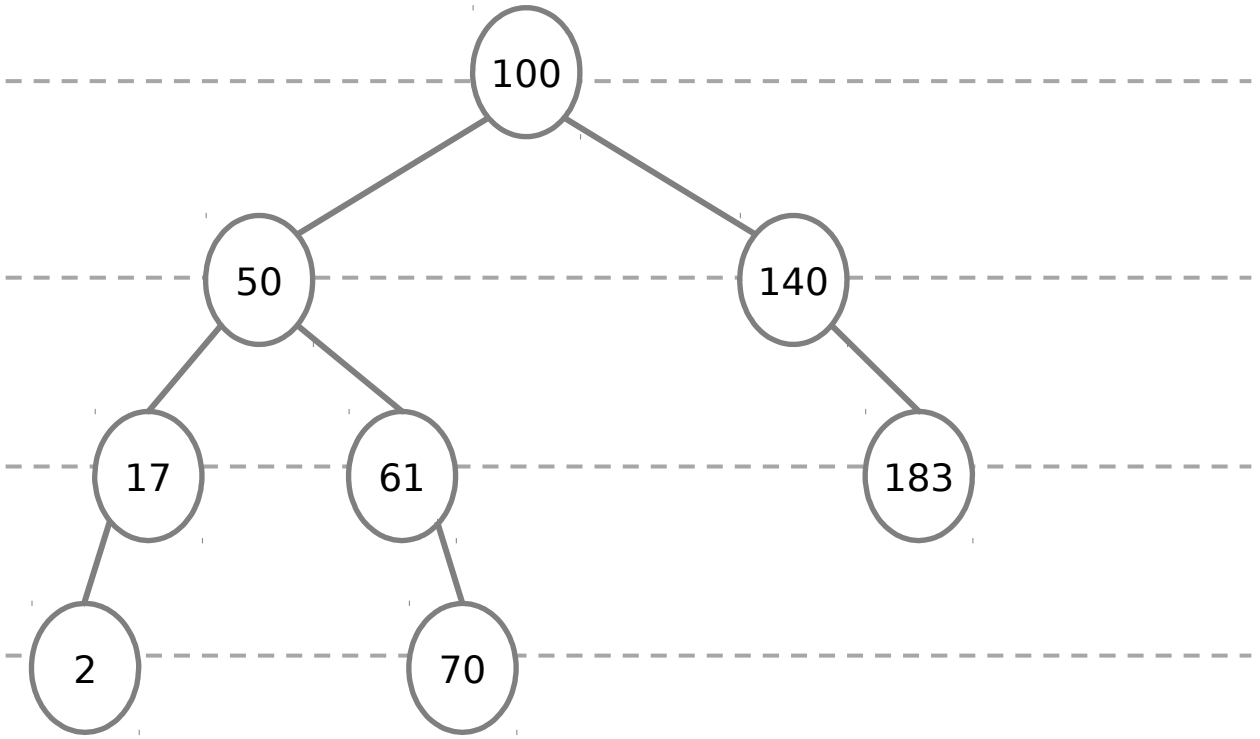
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Exemplos de Árvores AVL



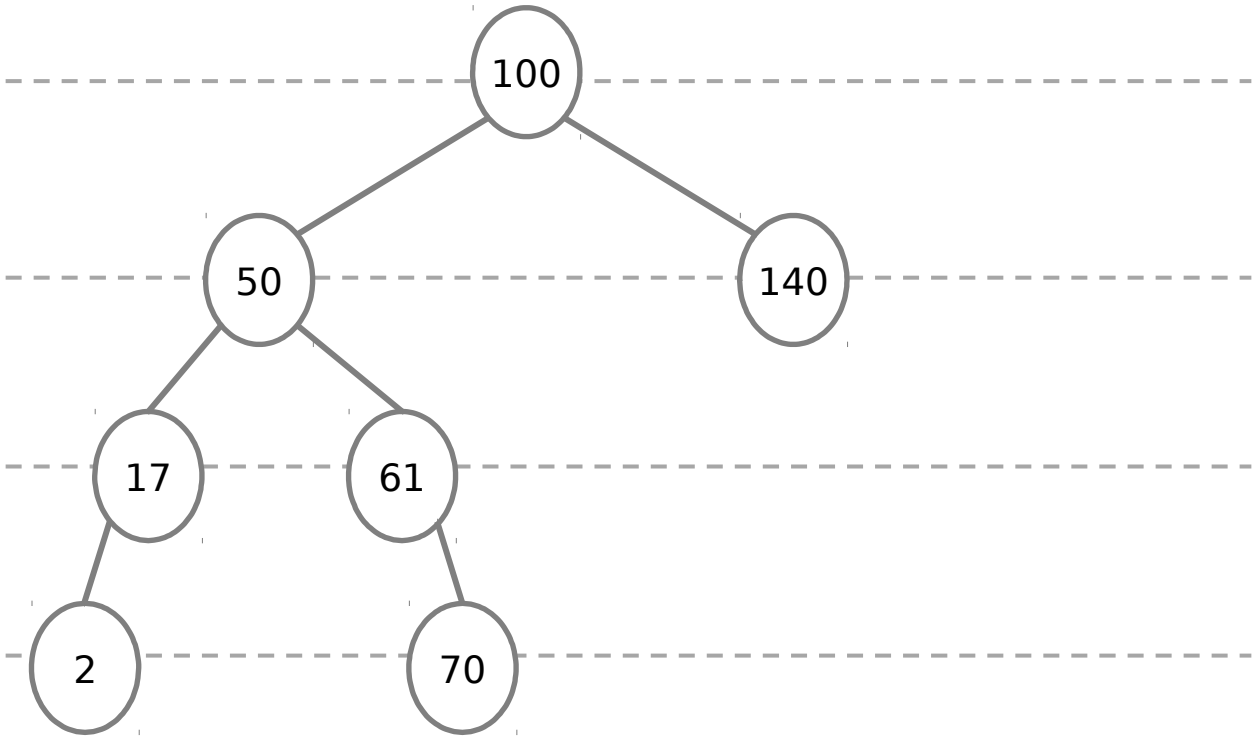
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Exemplos de Árvores AVL



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Exemplo de Árvore **não** AVL



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Fator de Balanceamento

fb = hd – he

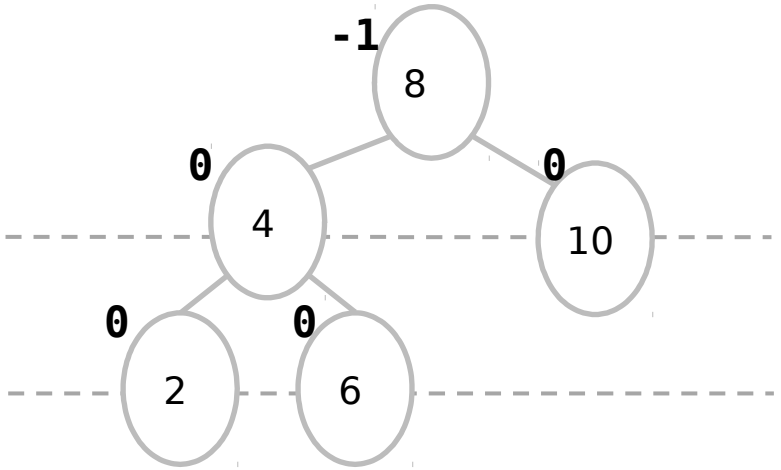
onde he = altura da sub-árvore à esquerda

hd = altura da sub-árvore à direita

```
struct _avl {
    int chave;
    int fb; /*fator de balanceamento*/
    struct _avl *esq;
    struct _avl *dir;
    tdados *dados;
};
```

Desequilíbrio – Caso 1a

árvore balanceada (equilibrada)



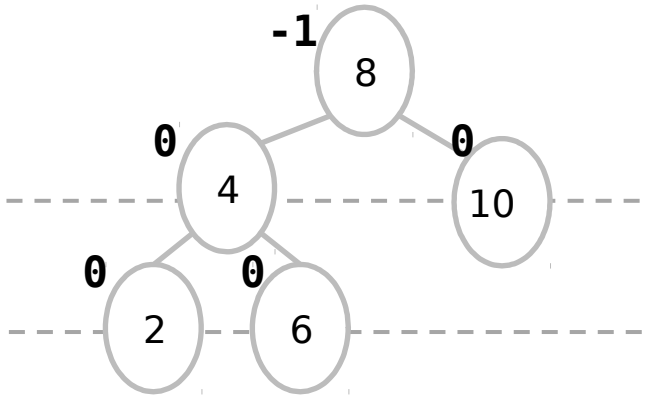
fb = hd – he

DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

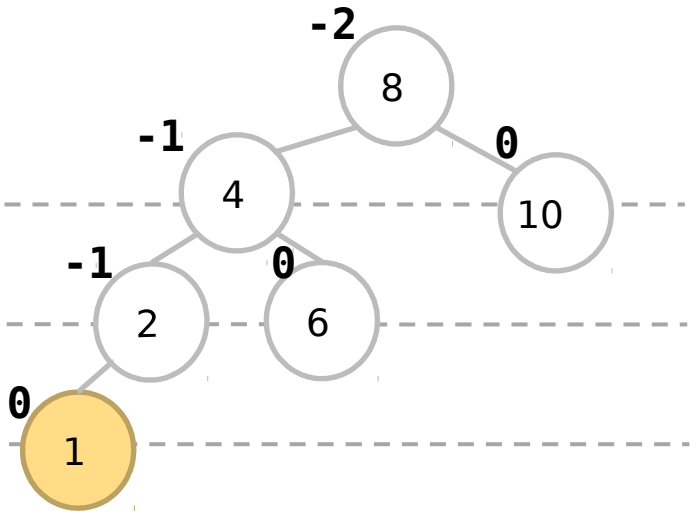
Desequilíbrio – Caso 1a

árvore balanceada (equilibrada)

fb = hd – he



desequilíbrio após inserir 1

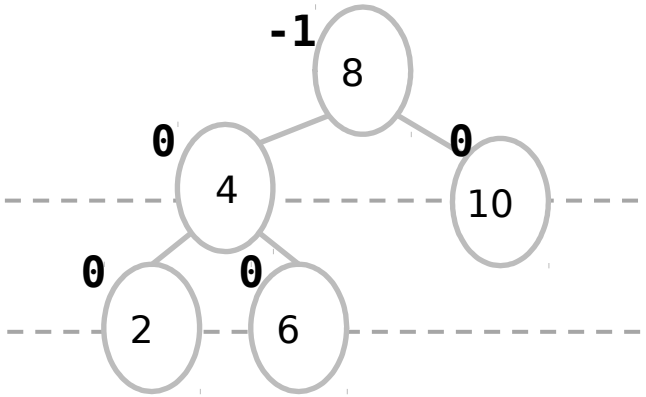


DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

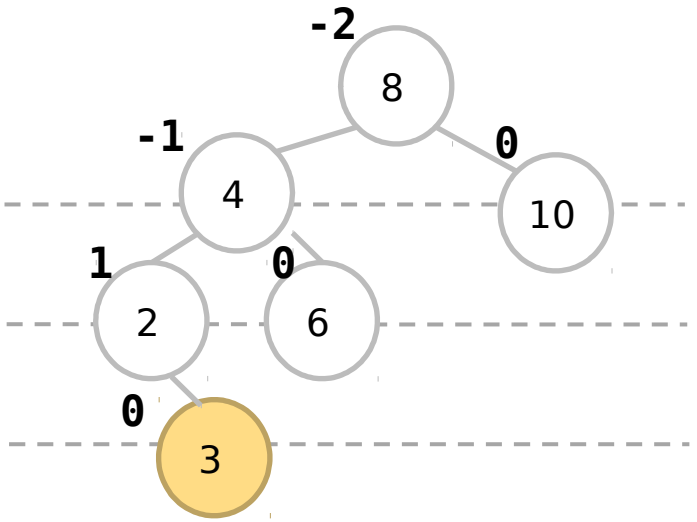
Desequilíbrio – Caso 1a

árvore balanceada (equilibrada)

fb = hd – he



desequilíbrio após inserir 3



DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

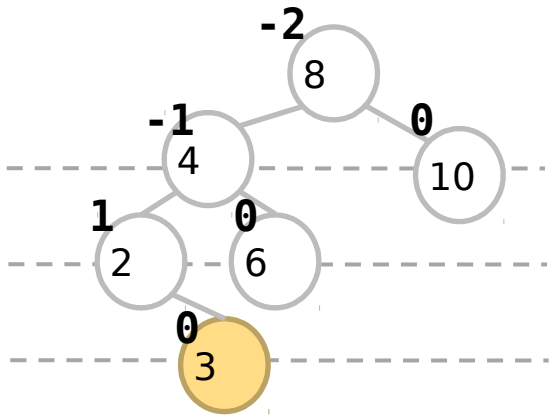
Esboço de Inserção em AVL

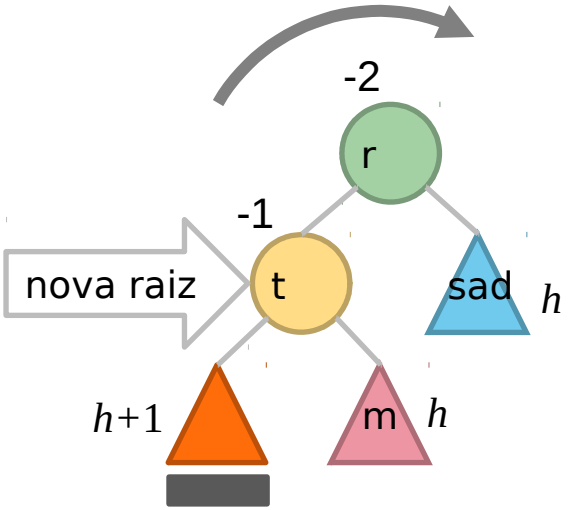
```
Avl* avl_insere (Avl* r, int chave, tdados *dados, int* cresceu) {
  if (r==NULL) {
     r = (Avl*) malloc(sizeof(Avl));
     r->esq = r->dir = NULL;
     r->chave = chave;
     r->dados = dados;
     r->fb = 0;
     *cresceu = 1;
  }
  else if (r->chave > chave) {
     r->esq = avl_insere (r->esq, chave, dados, cresceu);
     verifica se precisa consertar
   }
  else if (r->chave < chave) {
     r->dir = avl_insere (r->dir, chave, dados, cresceu);
     verifica se precisa consertar
  }
  return r;
}
```

DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

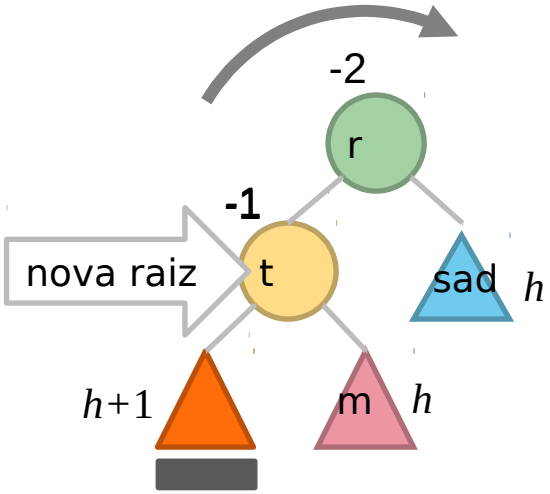
Caso 1a: Rotação à Direita

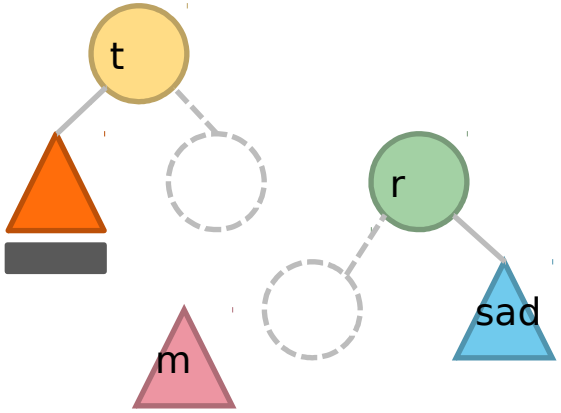
desequilíbrio após inserir 3

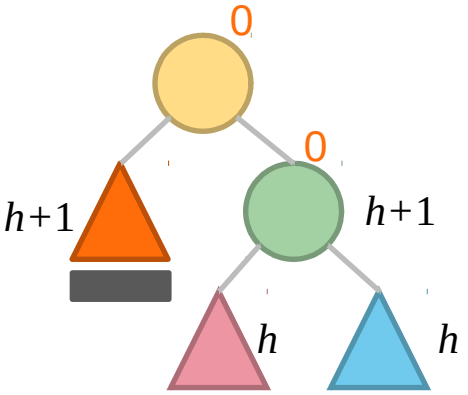




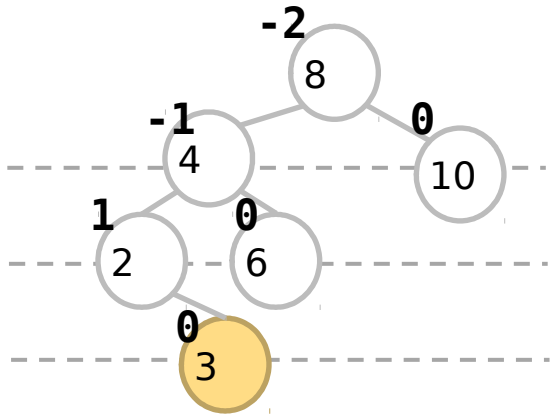
Caso 1a: Rotação à Direita





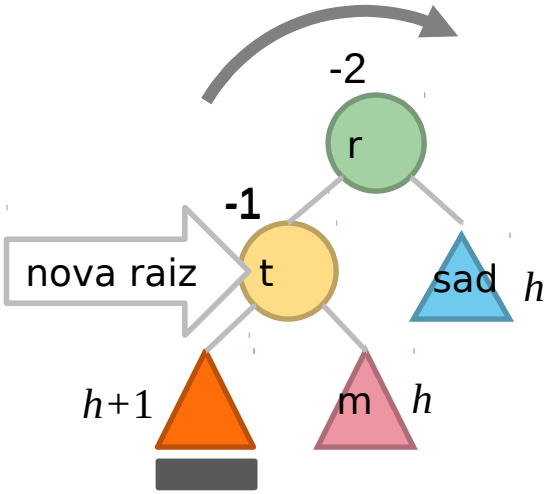


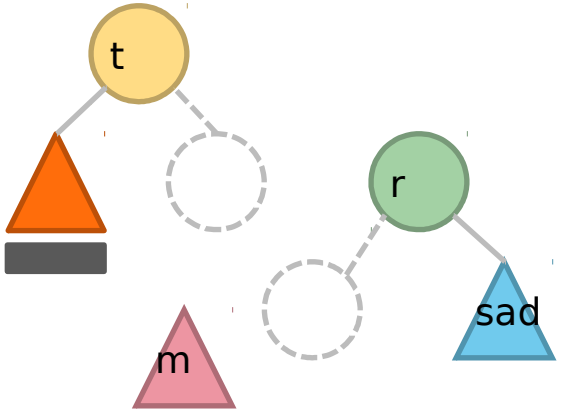
desequilíbrio após inserir 3

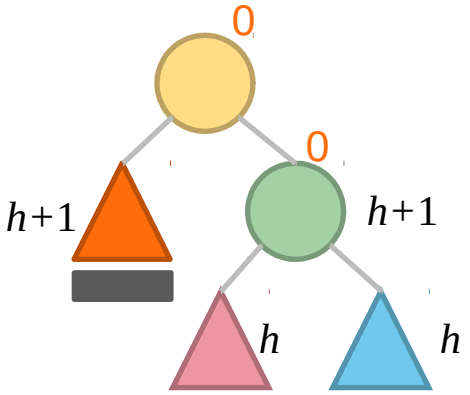


DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

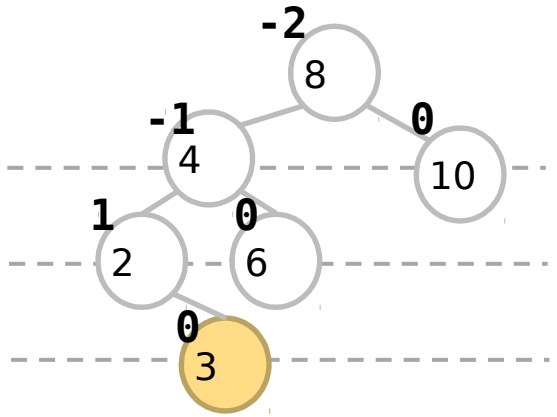
Caso 1a: Rotação à Direita



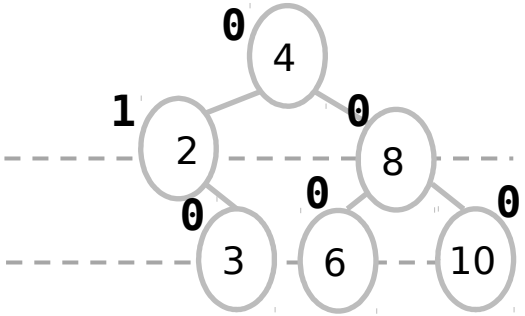




desequilíbrio após inserir 3



árvore balanceada (equilibrada)

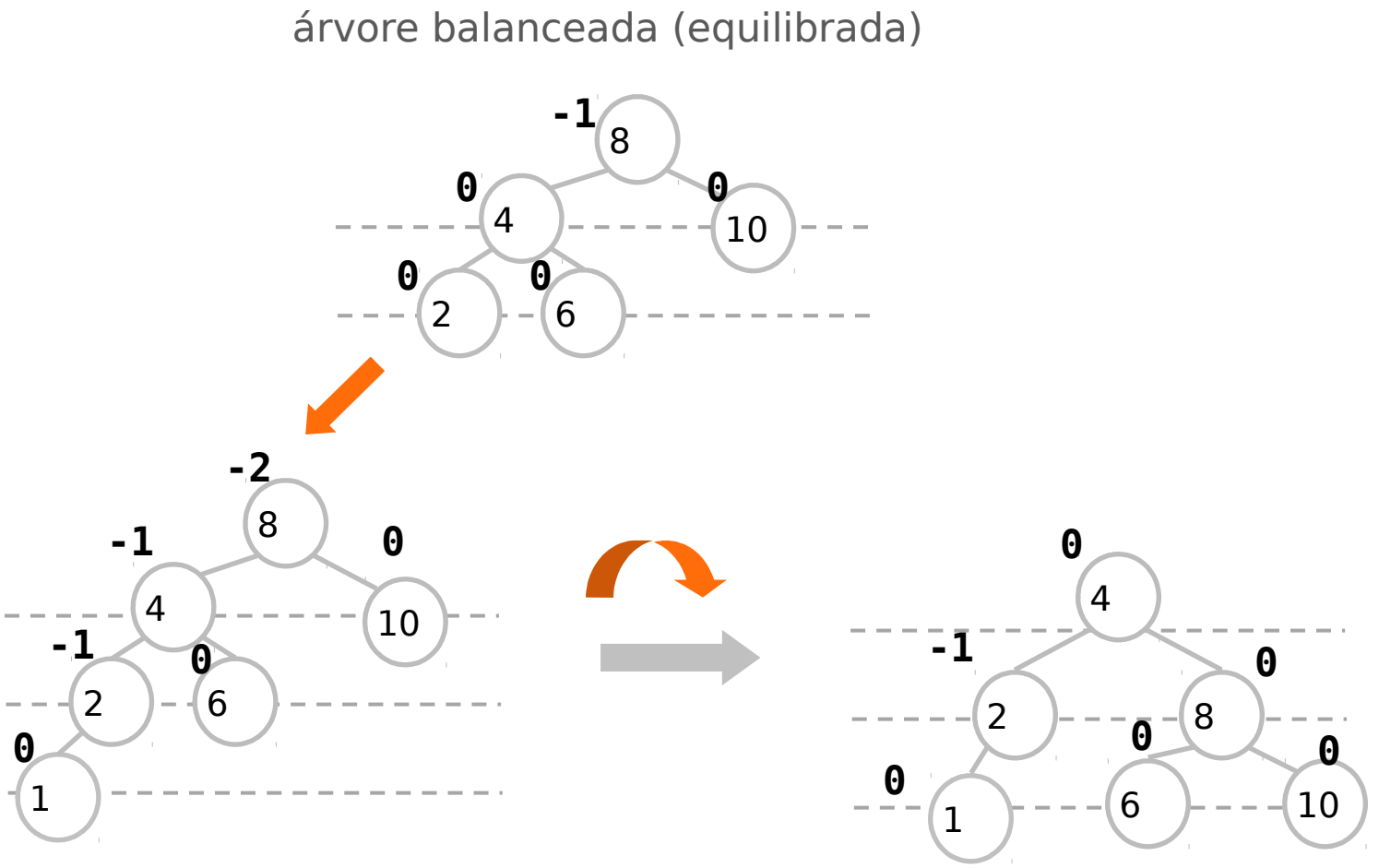


DEPARTAMENTO DE INFORMÁTICA PUC-RIO

Caso 1a: Exemplo de Rotação à Direita

uma rotação à direita

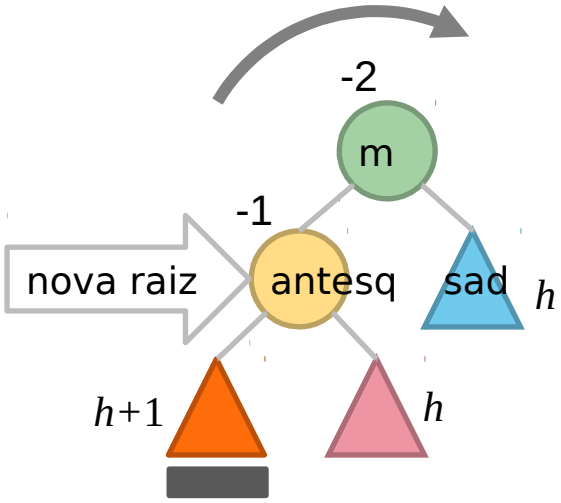
Caso 1a: Exemplo de Rotação à Direita

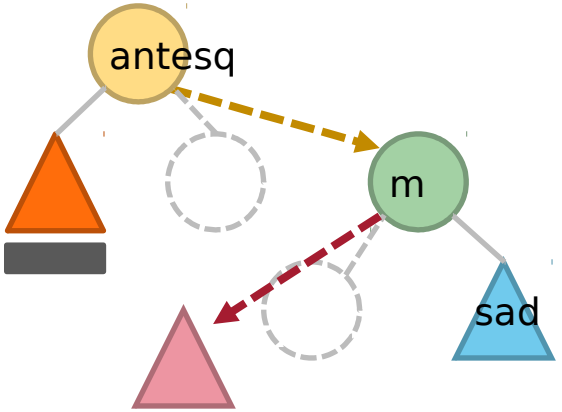


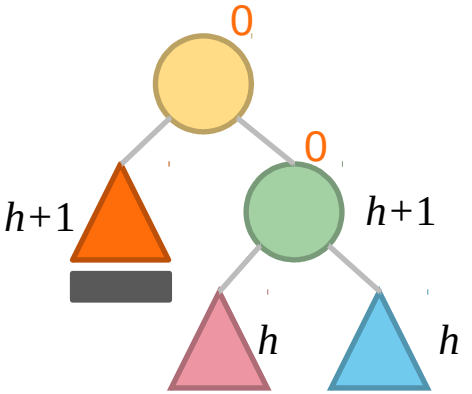
uma rotação à direita

DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 1a: Rotação à Direita

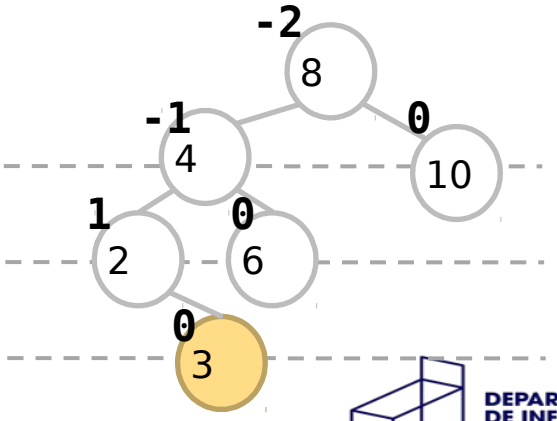







```
Avl* rotacao_direita(Avl *m) {
    Avl *antesq = m->esq;
    m->esq  = antesq->dir;
    antesq->dir  = m;
    return antesq; /* a raiz mudou */
}
```

desequilíbrio após inserir 3

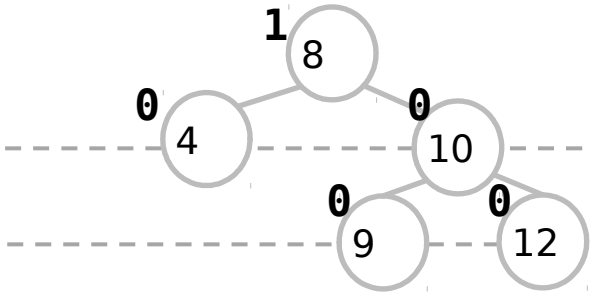


DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

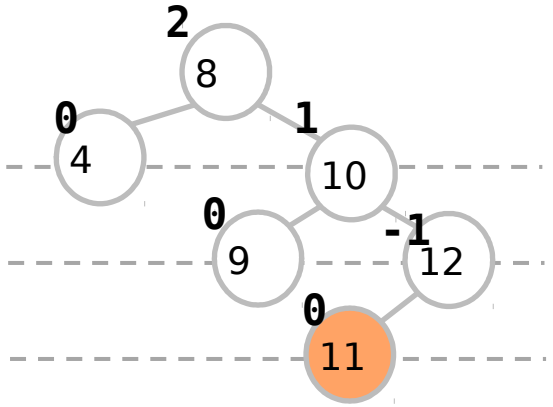
Desequilíbrio – Caso 1b

árvore balanceada (equilibrada)

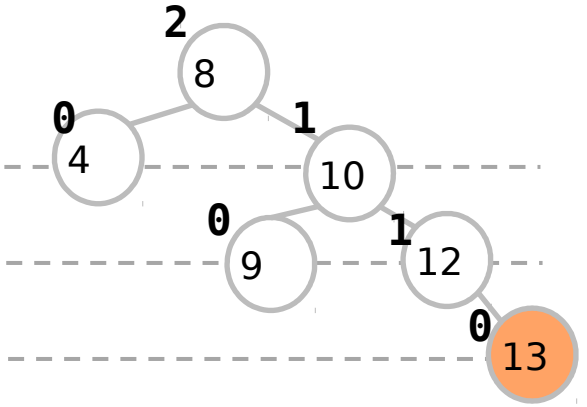
fb = hd – he



desequilíbrio após inserir 11



desequilíbrio após inserir 13



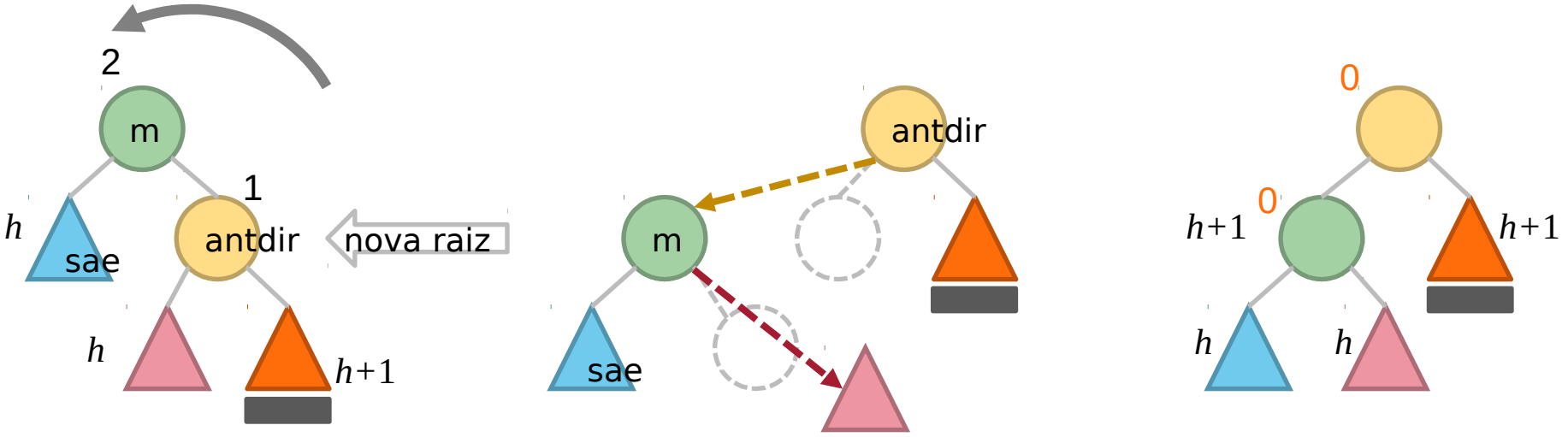
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 1b: Exemplo de Rotação à Esquerda

uma rotação à esquerda

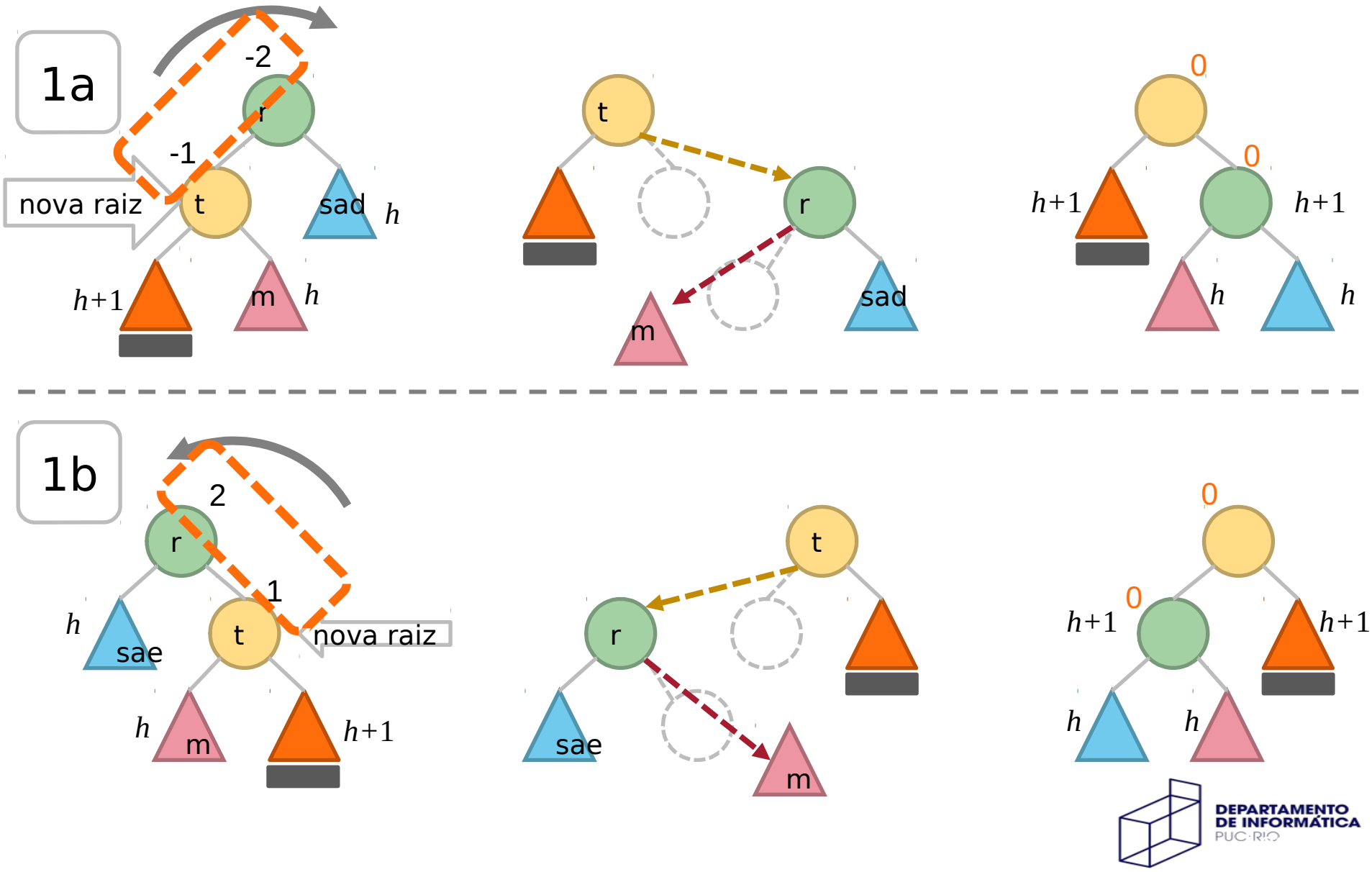
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Caso 1b: Rotação à Esquerda



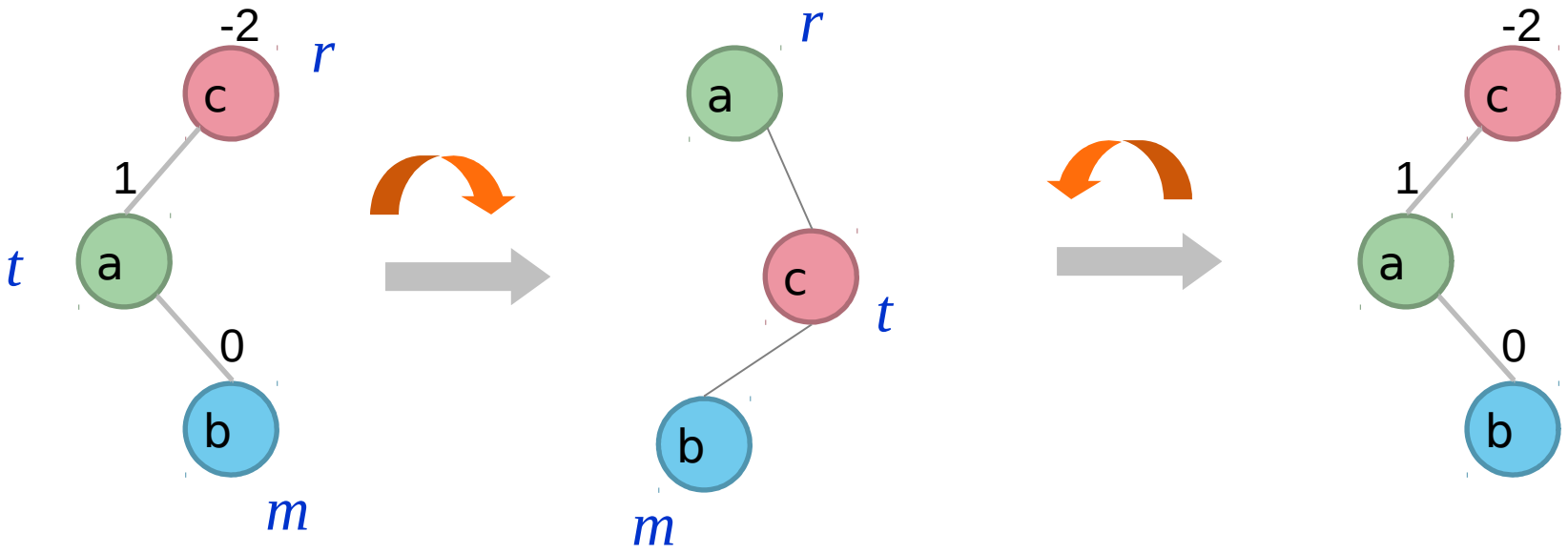

```
Avl* rotacao_esquerda(Avl *m) {
    Avl *antdir=m->dir;
    m->dir = antdir->esq;
    antdir->esq = m;
    return antdir;    /* a raiz mudou */
}
```

Casos Simples: fb **com mesmo sinal**



DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso onde rotação simples não resolve

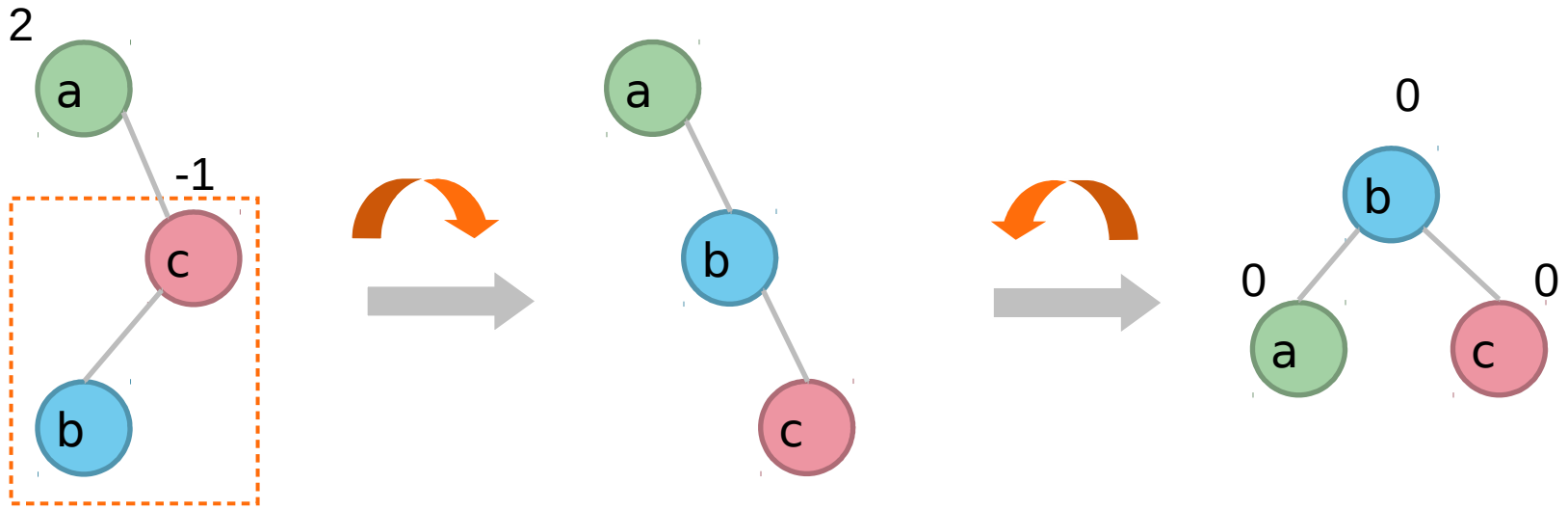


não resolve!

Caso 2a: Rotação Dupla

Fazemos uma rotação à direita na sub-árvore à direita...

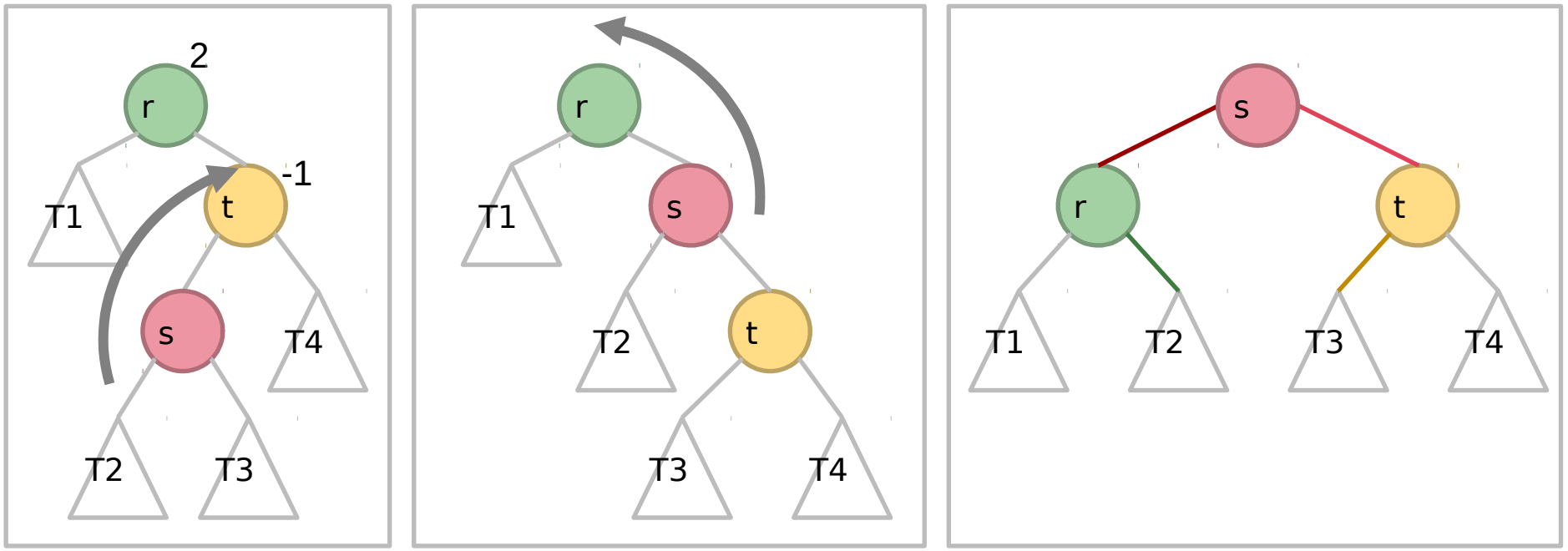
Caso 2a: Rotação Dupla



...seguida de uma rotação à esquerda

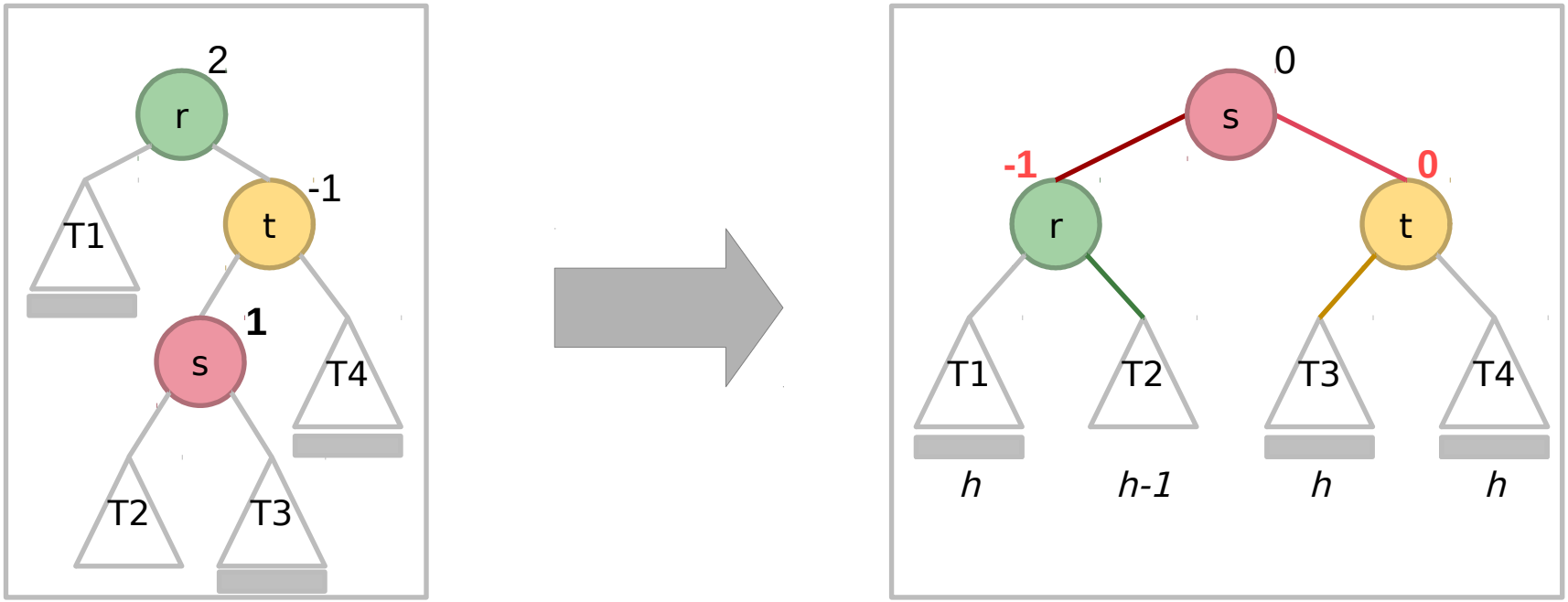
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 2a: Rotação Direita-Esquerda



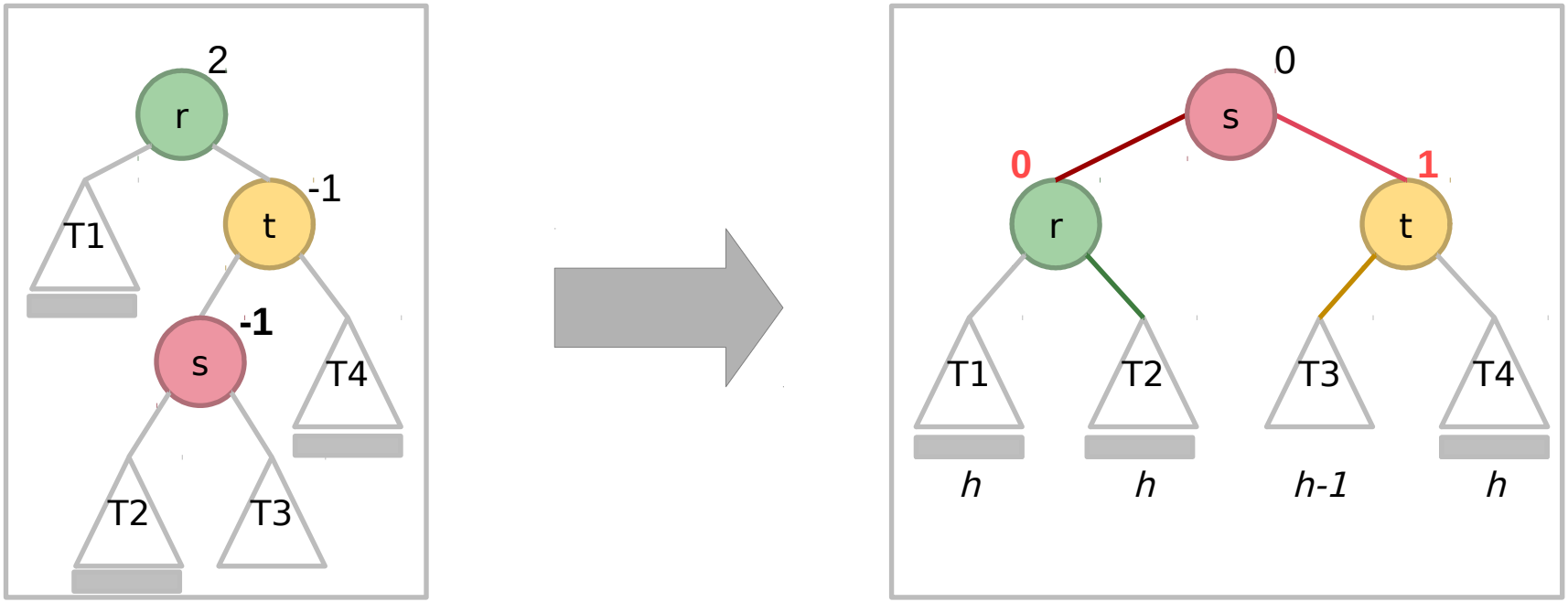
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Caso 2a.1: Rotação Direita-Esquerda



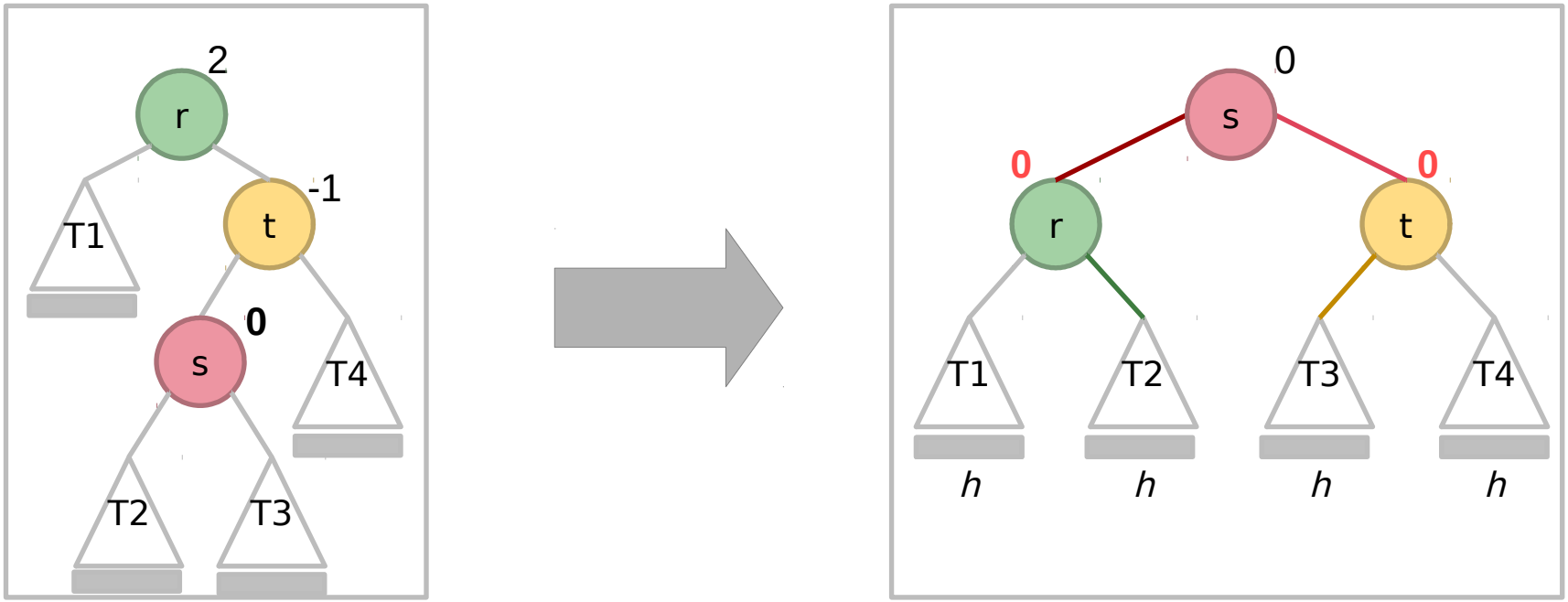
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 2a.2: Rotação Direita-Esquerda



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

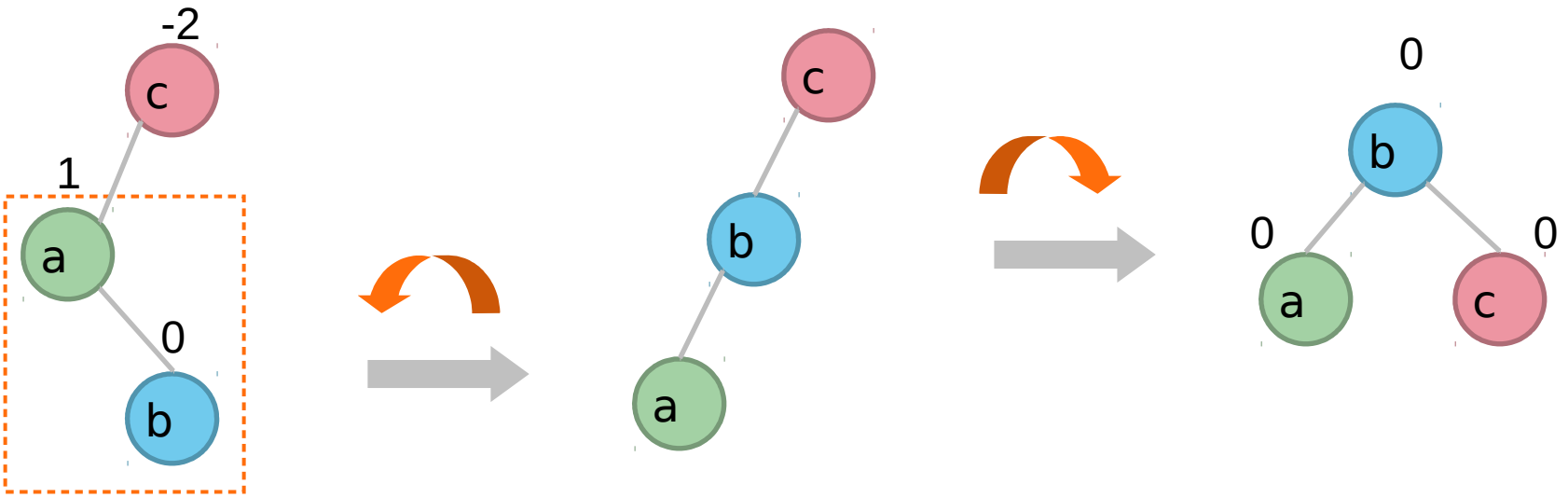
Caso 2a.3: Rotação Direita-Esquerda



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Caso 2b: Rotação Dupla

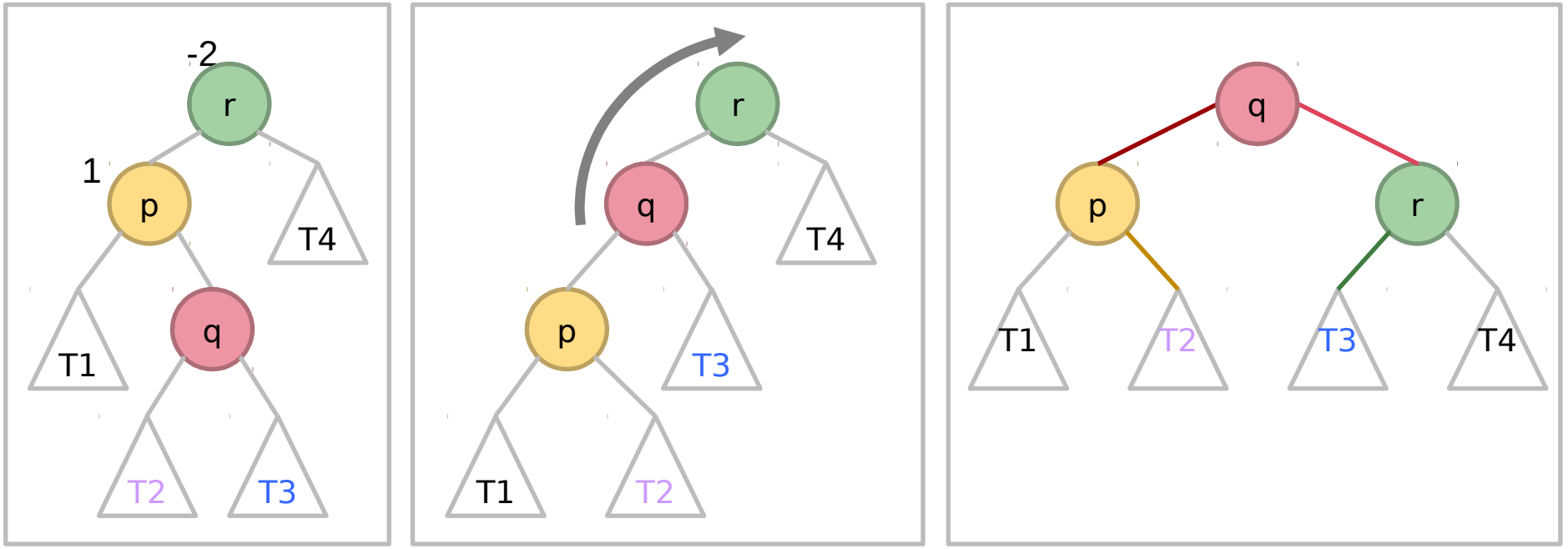
Uma rotação à esquerda na sub-árvore à esquerda...



...seguida de uma rotação à direita

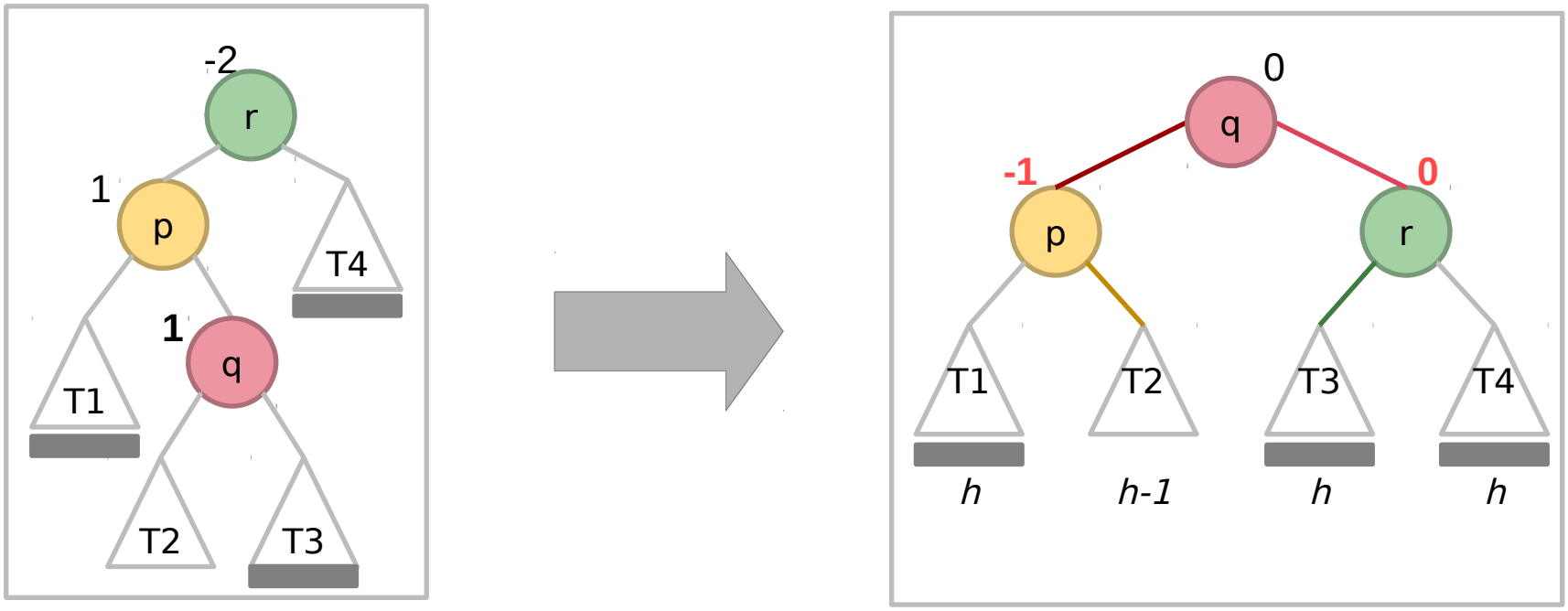
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 2b: Rotação Esquerda-Direita



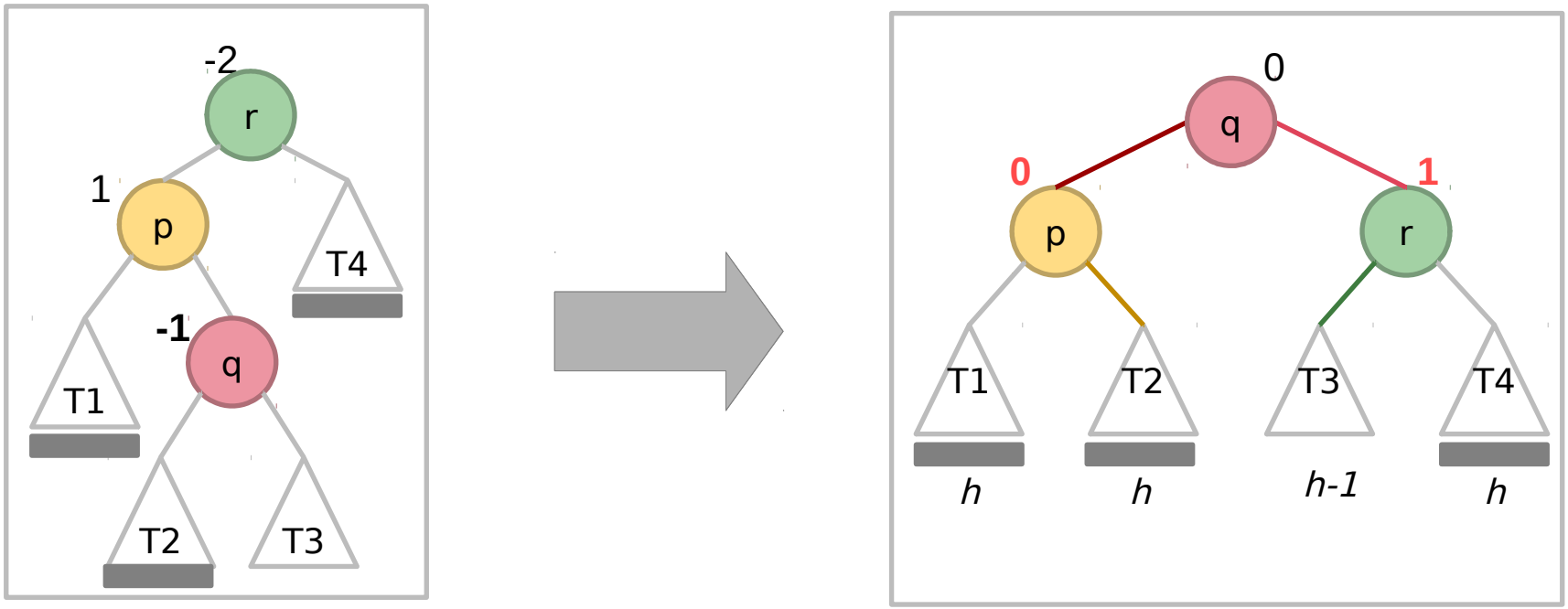
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Caso 2b.1: Rotação Esquerda-Direita



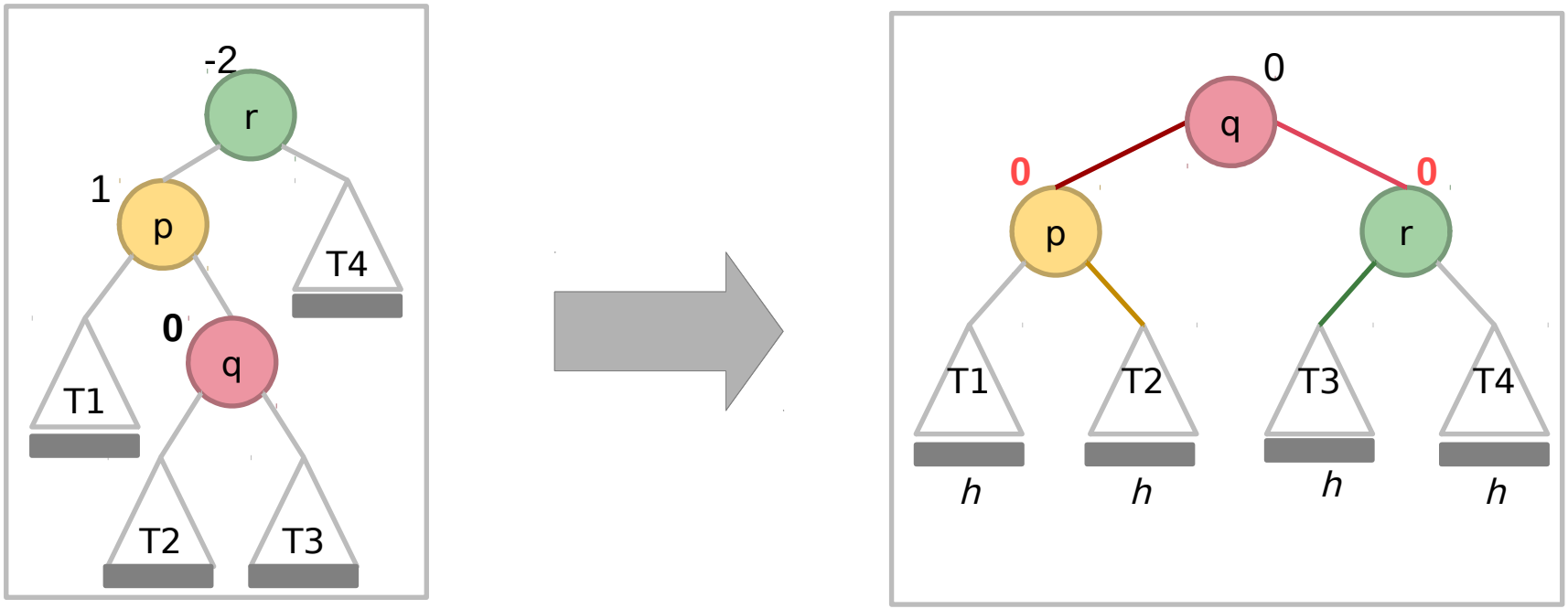
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 2b.2: Rotação Esquerda-Direita



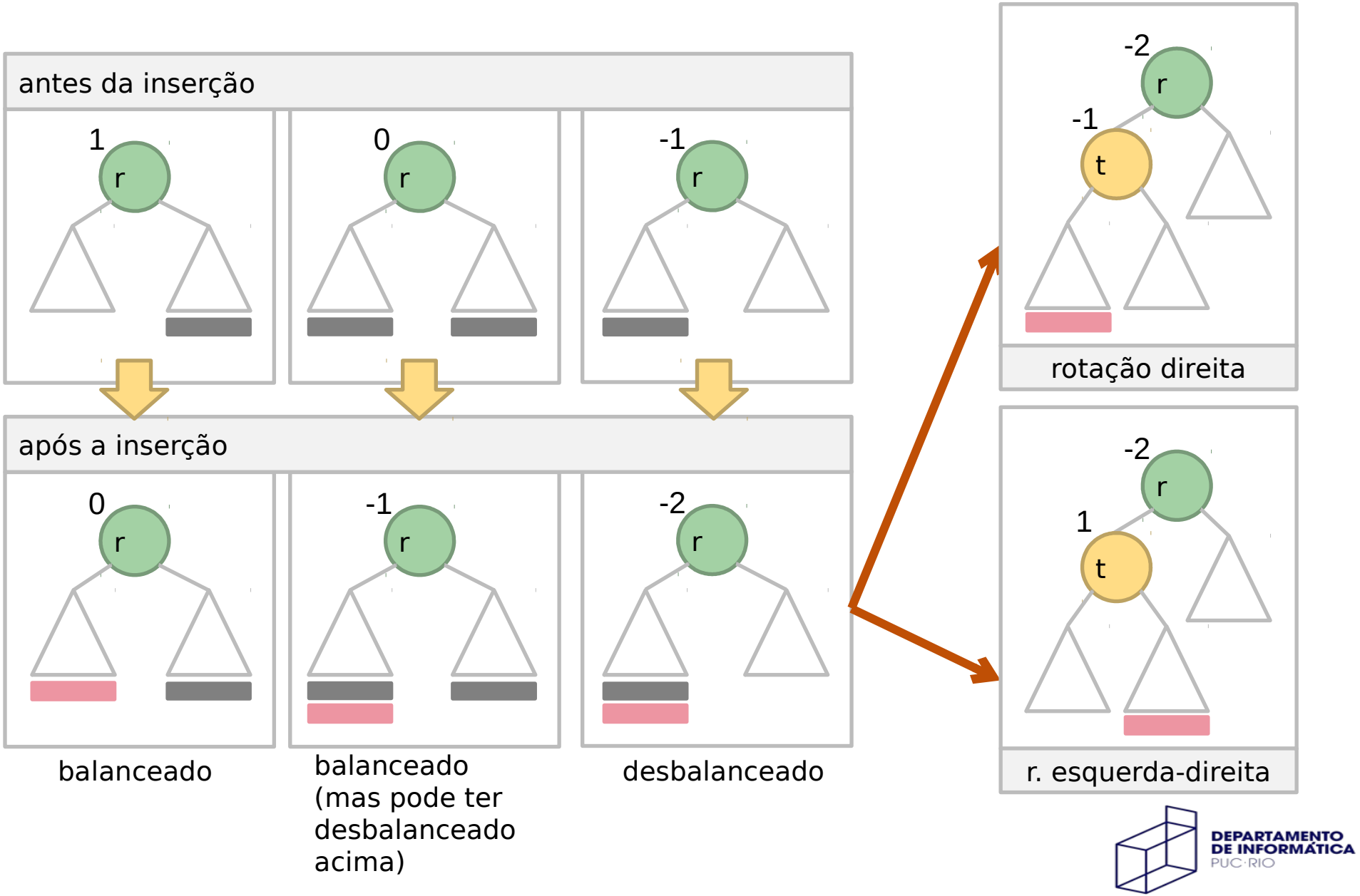
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Caso 2b.3: Rotação Esquerda-Direita



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Inserção à Esquerda



DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Inserção em AVL

```
Avl* avl_insere2(Avl* r, int chave, tdados *dados, int* cresceu) {
  if (r==NULL) {
     r = (Avl*) malloc(sizeof(Avl));
     r->esq = r->dir = NULL;
     r->chave = chave;
     r->dados = dados;
     r->fb = 0;
     *cresceu = 1;
  }
  else if (r->chave > chave) {
     r->esq = avl_insere2 (r->esq, chave, dados, cresceu);
     if (*cresceu) …
   }
  else if (r->chave < chave) {
     r->dir = avl_insere2 (r->dir, chave, cresceu);
     if (*cresceu) …
  }
  return r;
}
Avl* avl_insere(Avl* r, int chave, tdados *dados) {
    int cresceu;
    return avl_insere2 (r,chave,dados,&cresceu);
}
```

Inserção em AVL

```
Avl* avl_insere2(Avl* r, int chave, tdados *dados, int* cresceu) {
  if (r==NULL) {
     r = (Avl*) malloc(sizeof(Avl));
     r->esq = r->dir = NULL;
     r->chave = chave;
     r->dados = dados;
     r->fb = 0;
     *cresceu = 1;
  }
  else if (r->chave > chave) {
     r->esq = avl_insere2 (r->esq, chave, dados, cresceu);

     if (*cresceu) {      /* r cresceu à esquerda (ou seja, he aumentou)*/
       switch(r->fb) { /* análise do fator de balanceamento de r */

          case  1: /* antes: hd>he xxx depois: hd=he pois he aumentou */
          ...
          case  0: /* antes: hd=he xxx depois: hd<he pois he aumentou */
          ...
          case -1: /* antes: hd<he xxx depois: hd-he=-2 pois he aumentou */
          ...
       }
  } ....
```

Inserção em AVL

```
if (*cresceu) {      /* r cresceu à esquerda (ou seja, he aumentou)*/

  switch(r->fb) { /* análise do fator de balanceamento de r */

    case  1: /* antes: hd>he - depois: hd=he pois he aumentou */
      r->fb =  0; *cresceu = 0; break;

    case  0: /* antes: hd=he - depois: hd<he pois he aumentou */
      r->fb = -1; break;  /* avisa para cima que cresceu */

    case -1: /* antes: hd<he - depois: hd-he=-2 pois he aumentou */
      if (r->esq->fb == -1) r = avl_rotacao_direita(r);
            else r = avl_rotacao_esquerda_direita(r);
      *cresceu = 0; break;
  }
}
```

Remoções

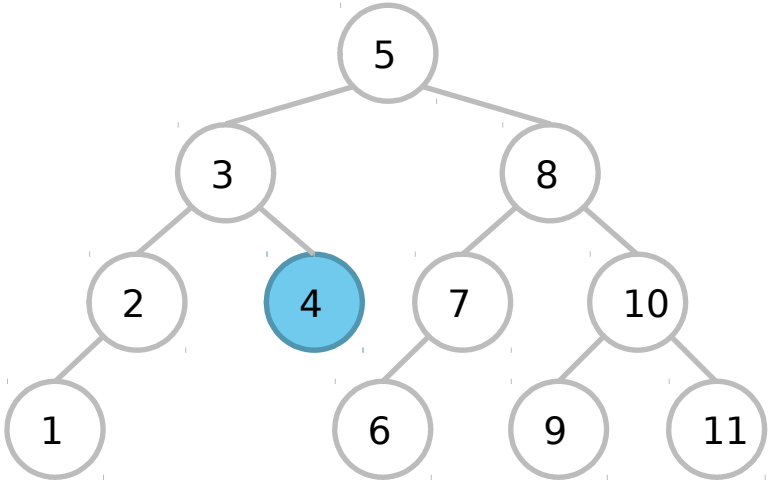
Problemas

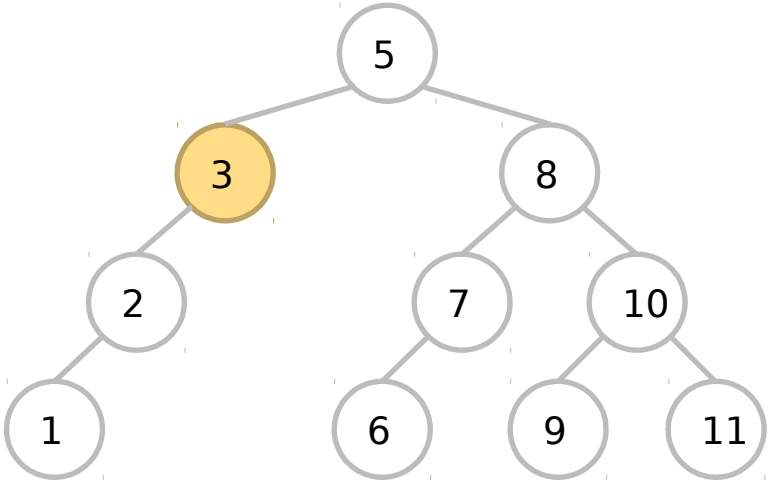
- semelhantes aos de inserções, ou seja, pode ocorrer desbalanceamento

Análise de Casos

- Caso 1: nó removido é uma folha ou possui apenas 1 filho
- Caso 2: nó removido tem as duas sub-árvores

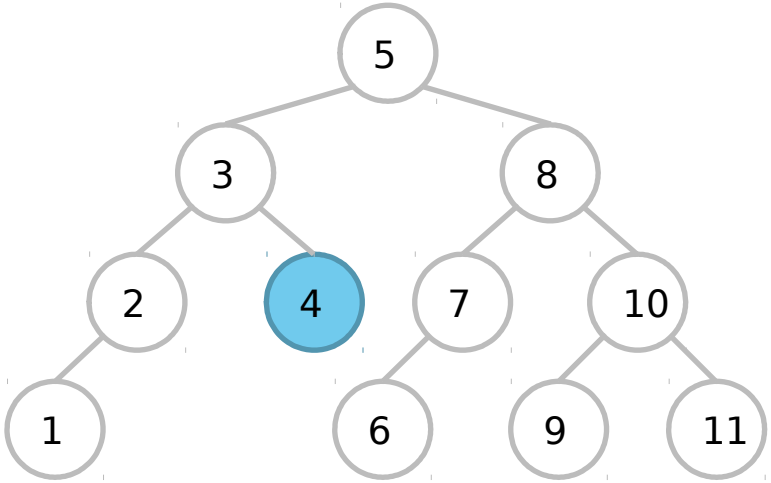
Exemplos de Remoção

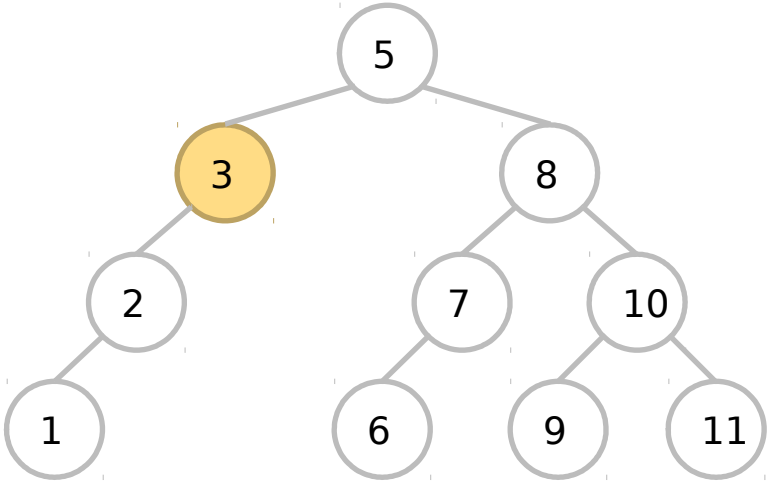


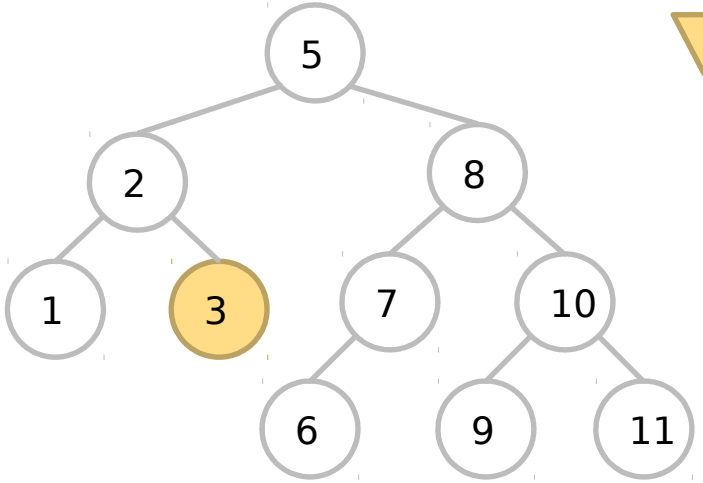


DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Exemplos de Remoção



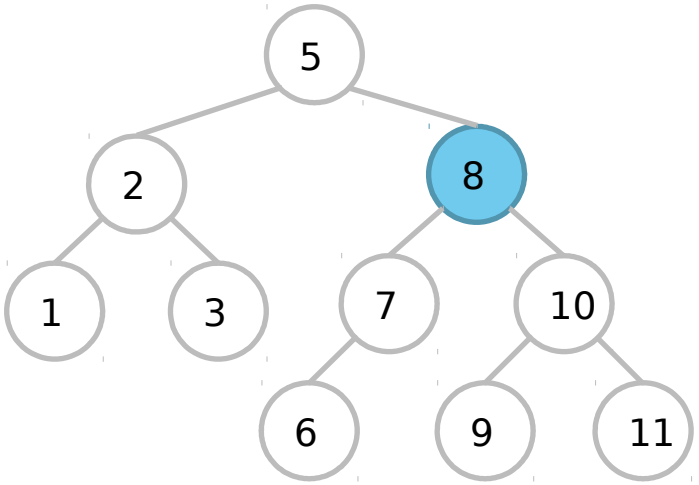


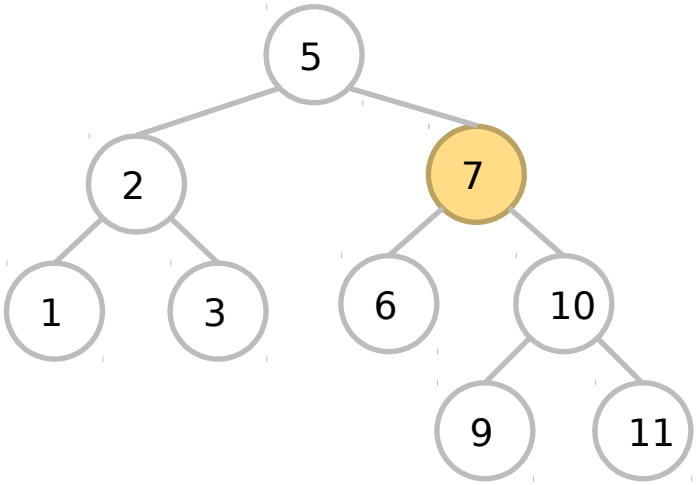


rotação à direita

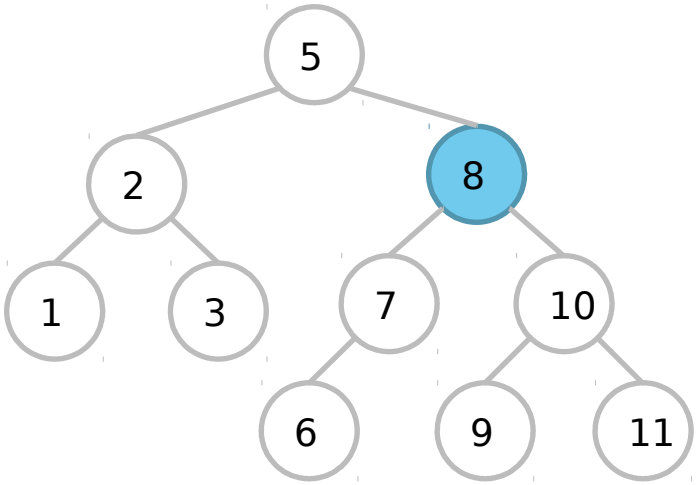
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

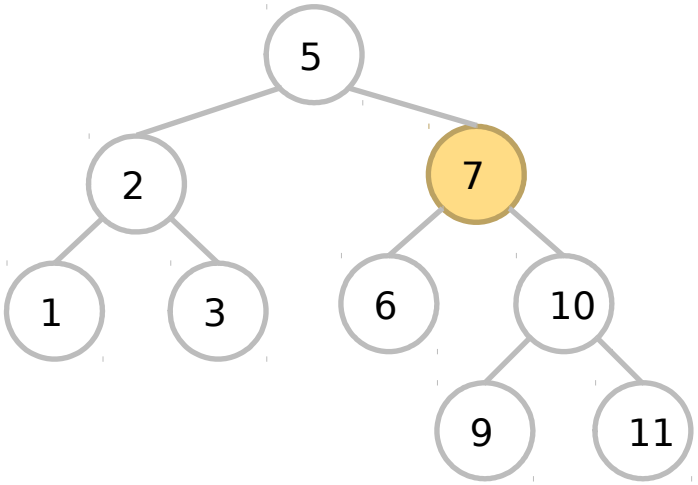
Exemplos de Remoção





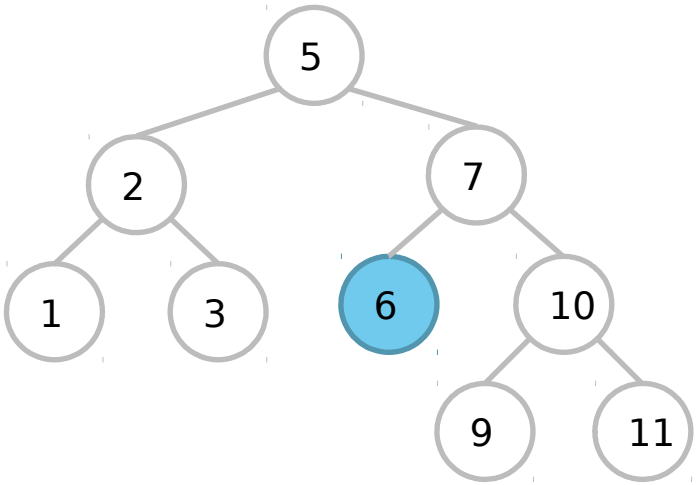
Exemplos de Remoção

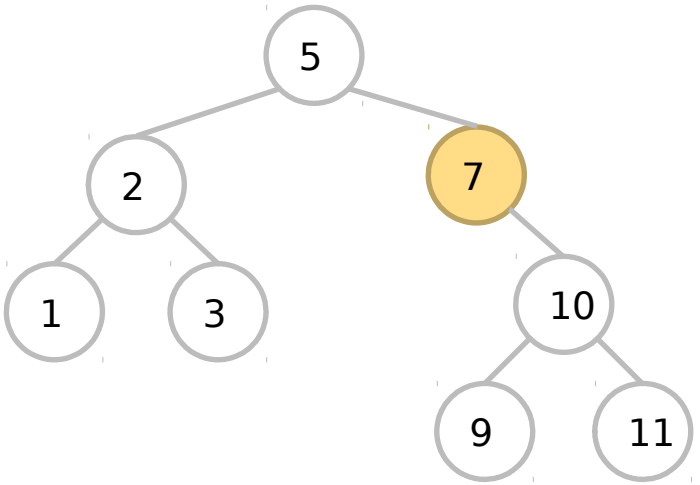




árvore balanceada (equilibrada)

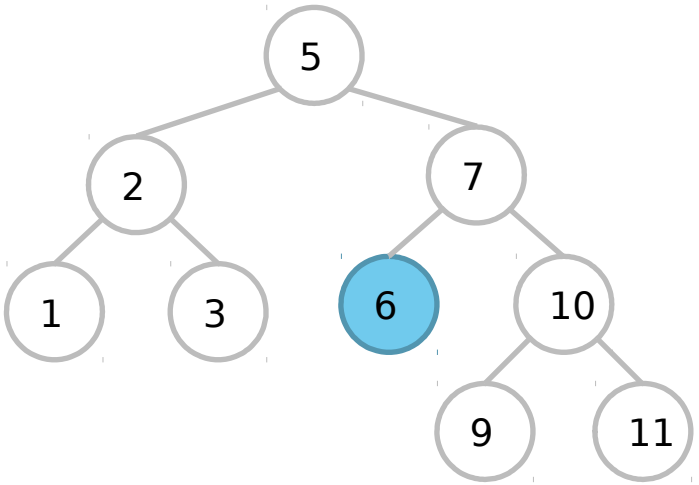
Exemplos de Remoção

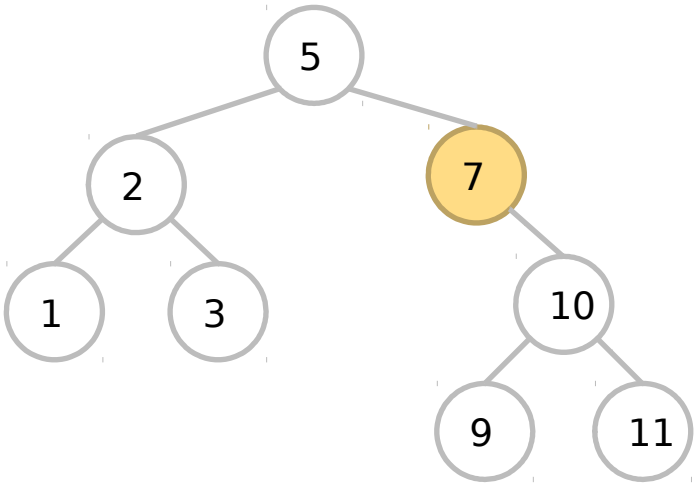


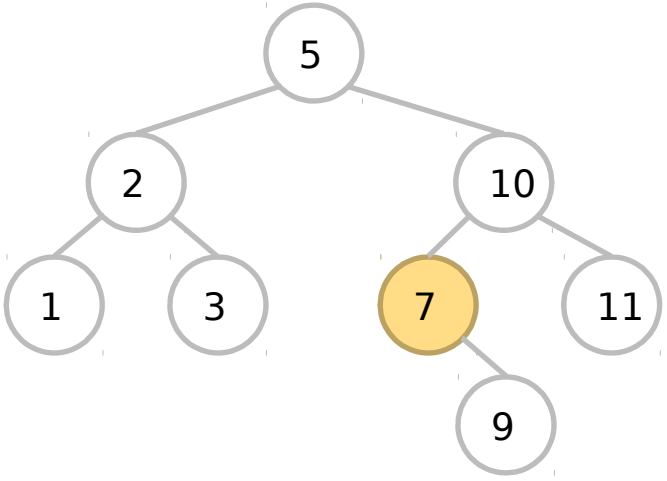


DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Exemplos de Remoção







rotação à esquerda

DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

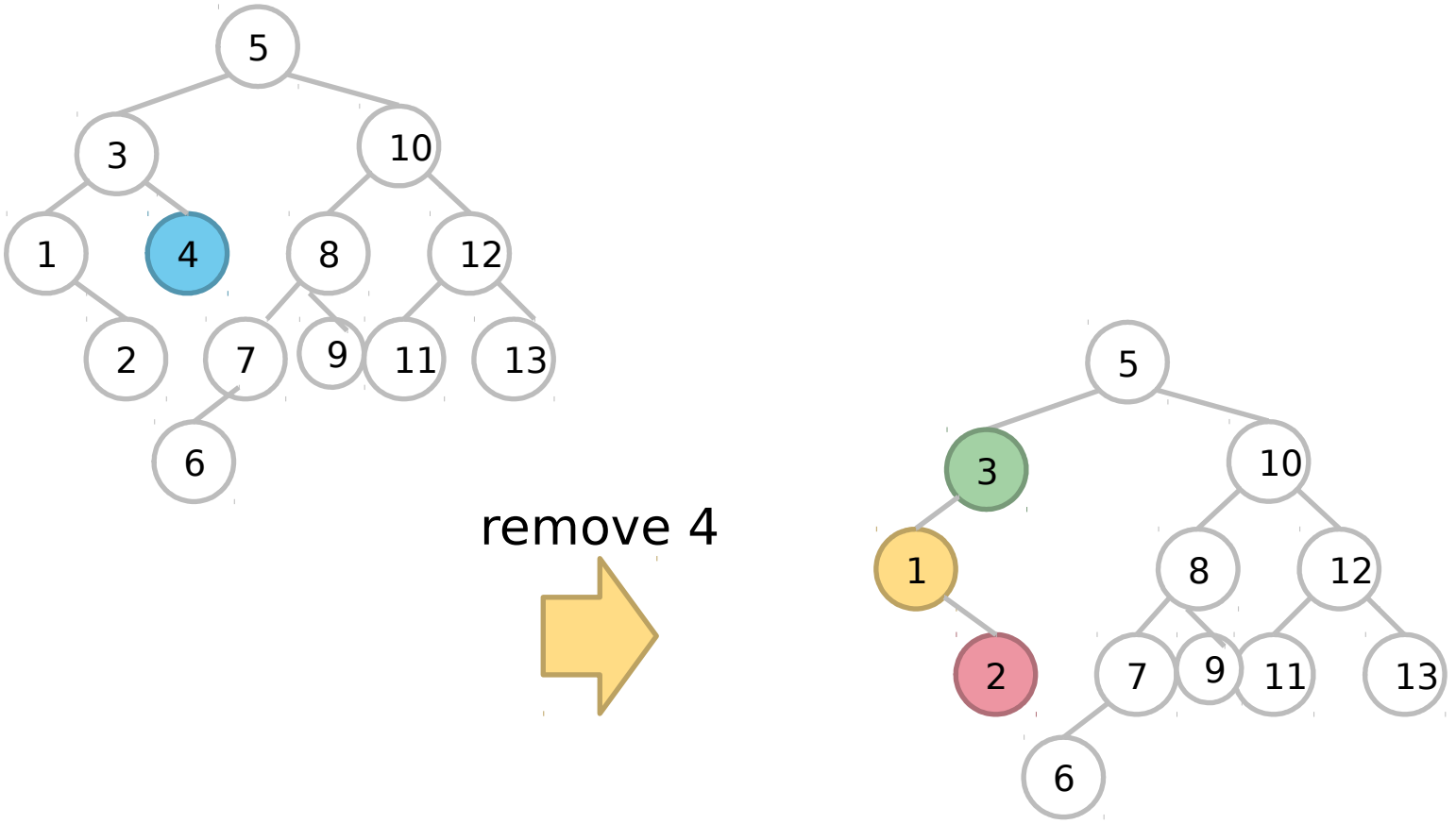
Problemas com Árvores AVL

Nem sempre uma ÚNICA rotação (simples ou dupla) resolve o problema de desbalanceamento dos nós

- há casos em que O(logn) rotações são necessárias

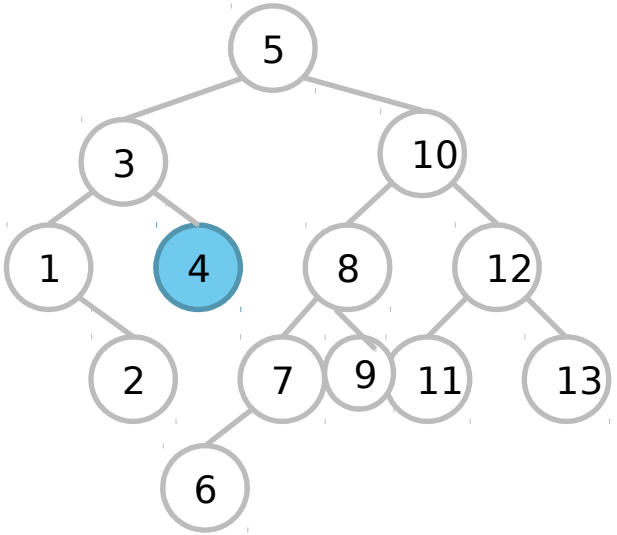
No exemplo a seguir a exclusão de um nó provoca diversas rotações

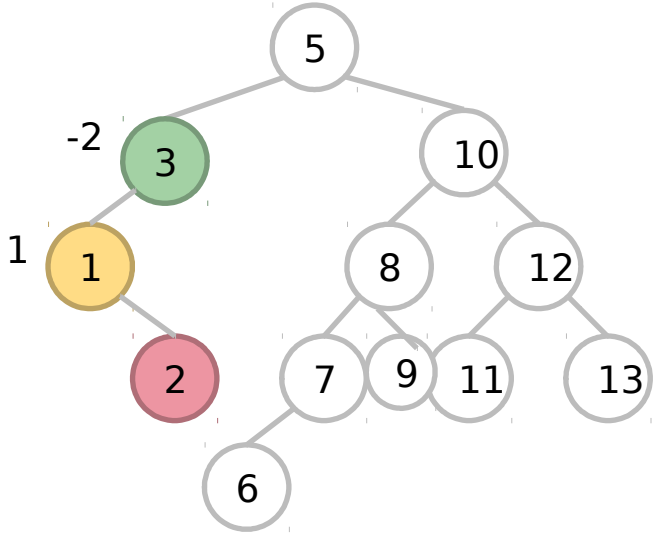
Múltiplas Rotações em Árvore AVL



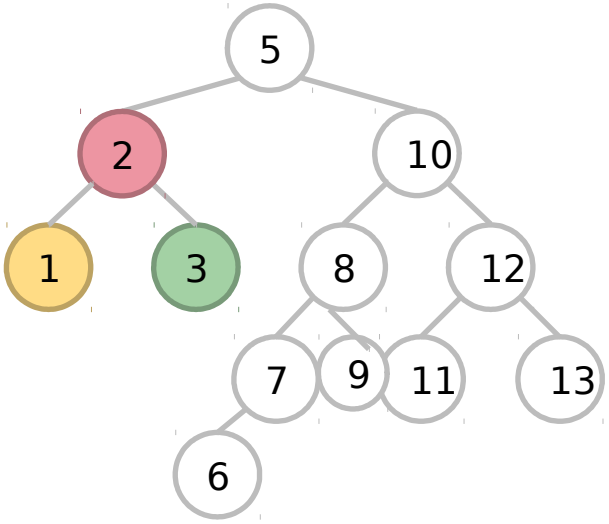
DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Múltiplas Rotações em Árvore AVL



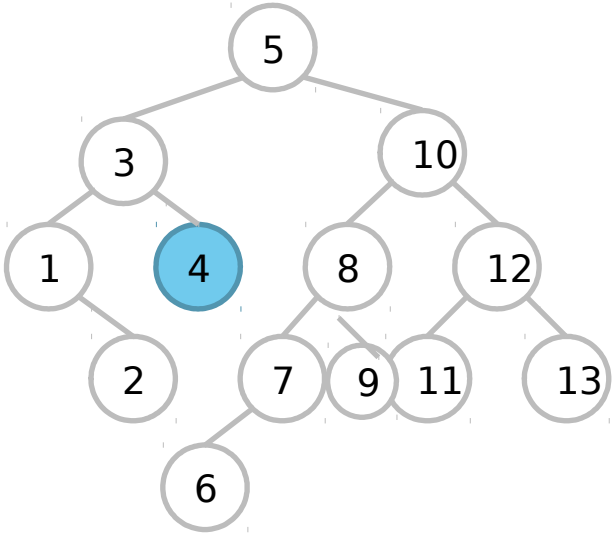


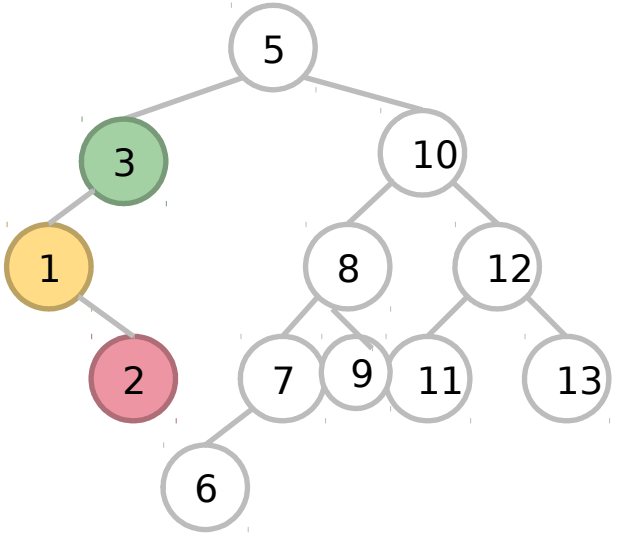
rotação dupla
esquerda-direita

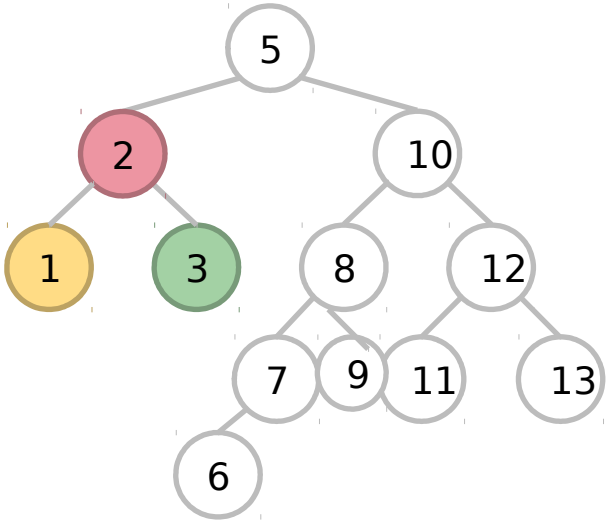


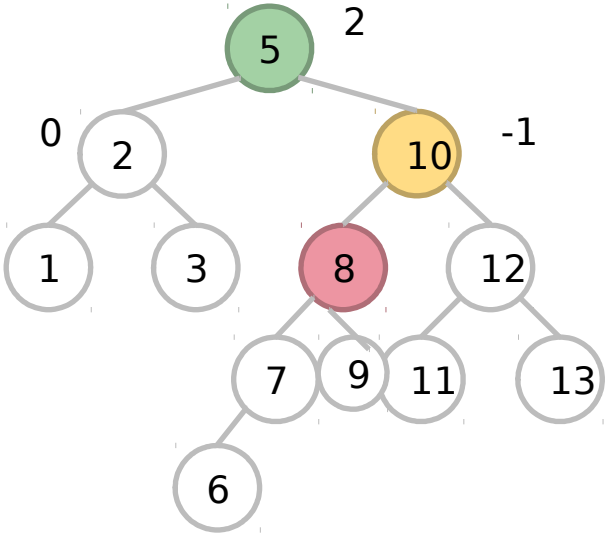
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Múltiplas Rotações em Árvore AVL

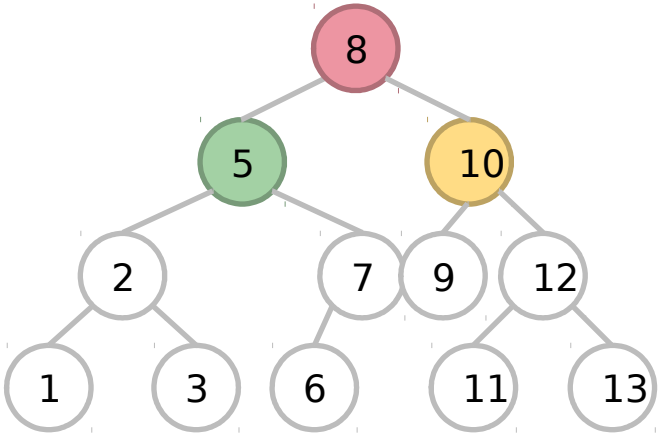








rotação dupla
direita-esquerda



DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

Esboço do Algoritmo de Remoção

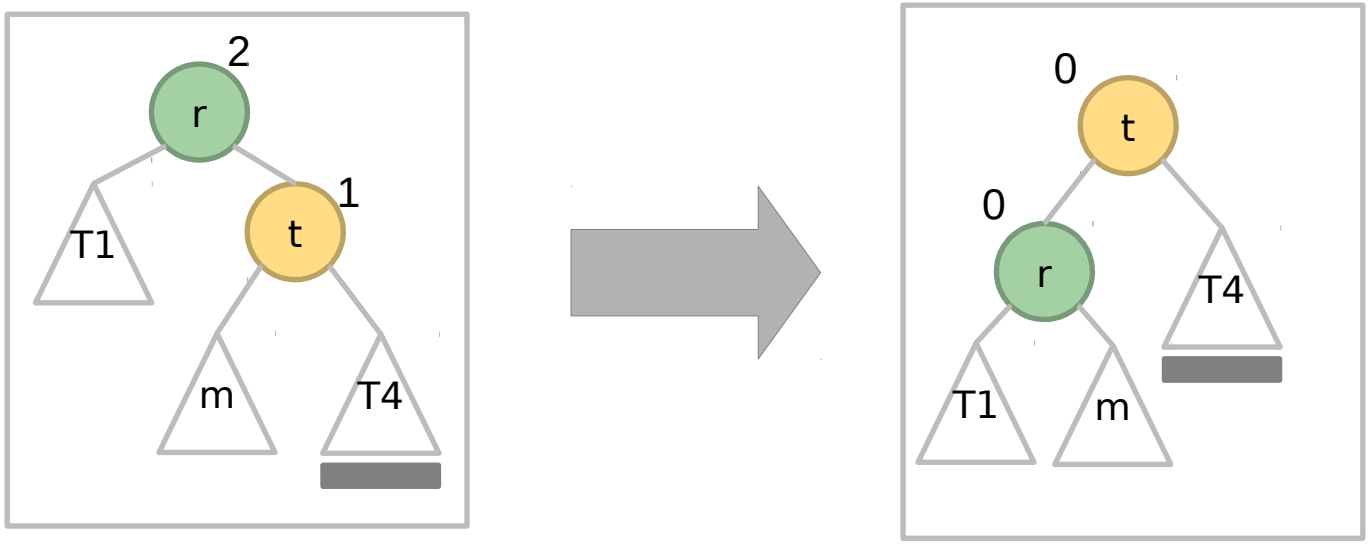
Pesquise o nó que contém a chave procurada, aplicando recursivamente o algoritmo de remoção

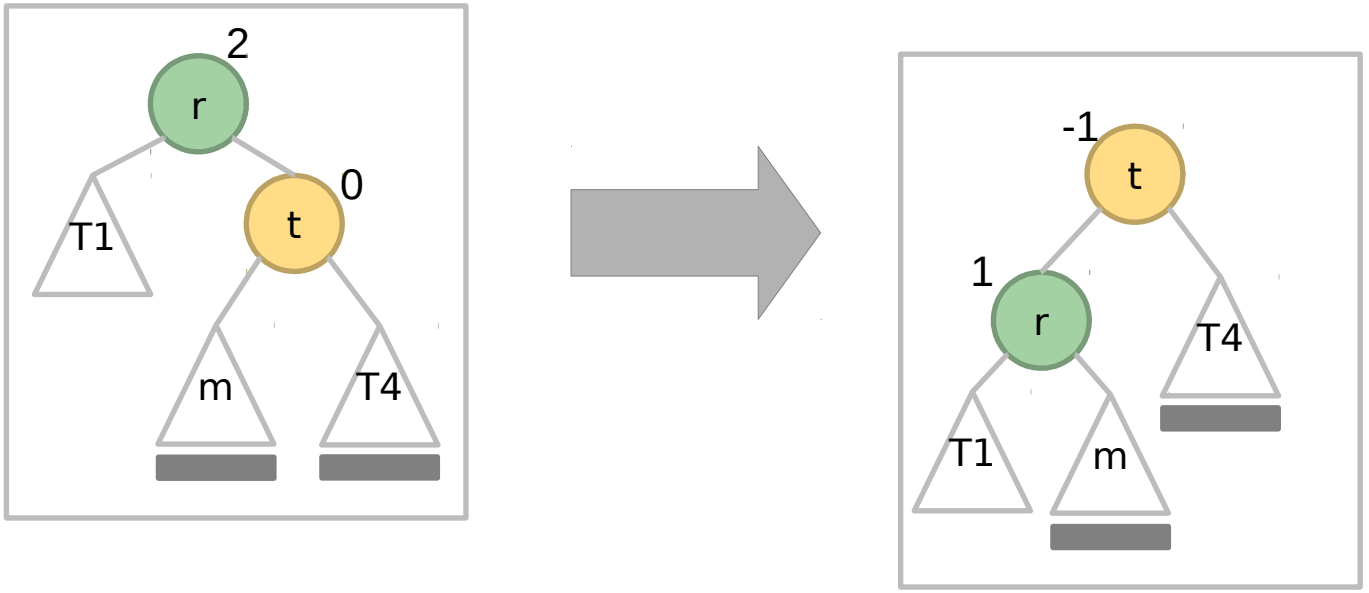
Quando achar o nó:

- faça a remoção como na ABB
- analise o balanceamento
- se o nó estiver desbalanceado (|fb| > 1) faça as rotações necessárias

Recursivamente, reporte uma mudança de altura ao pai para que ele faça as devidas correções

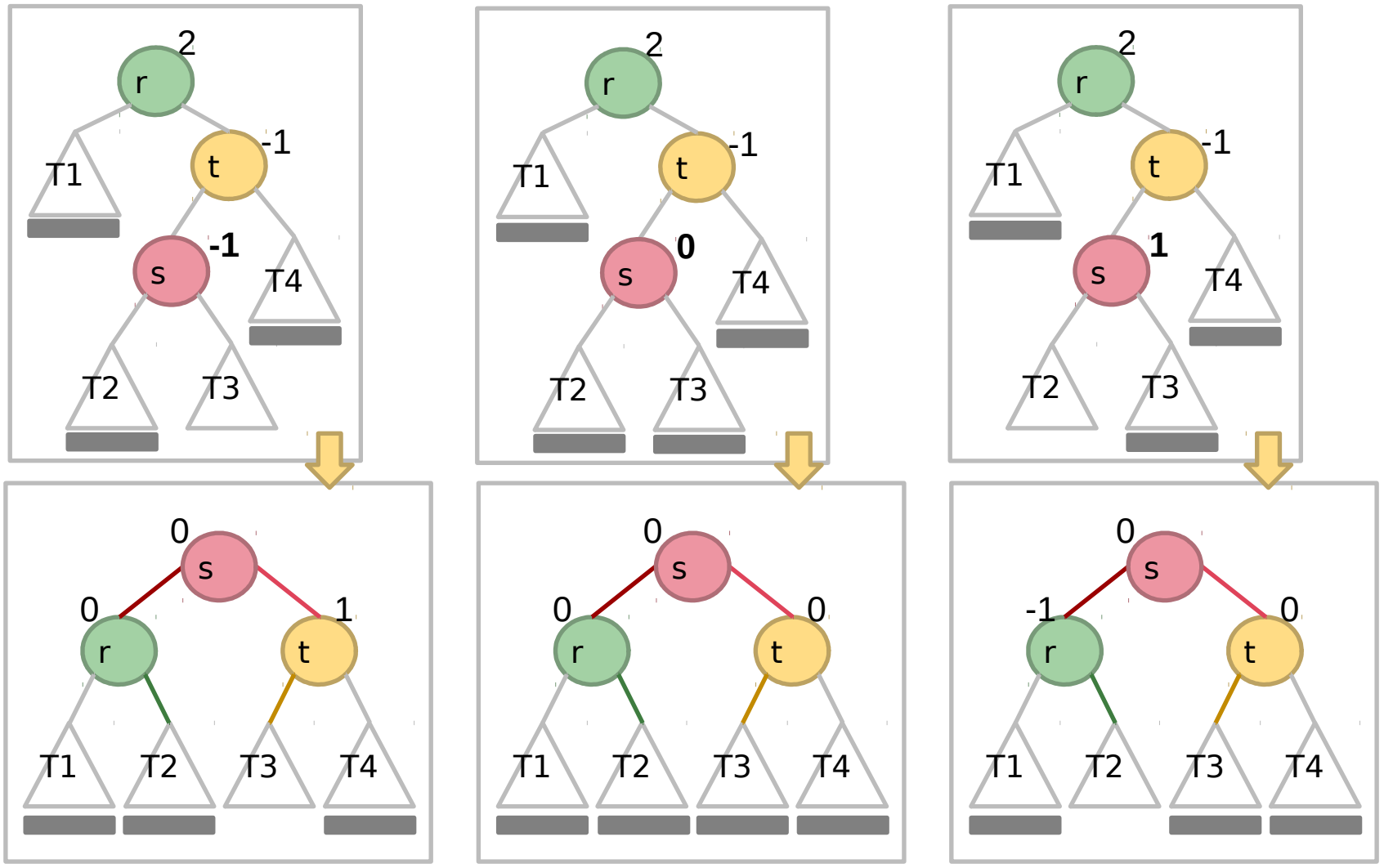
Rotação à esquerda (sae perdeu altura)





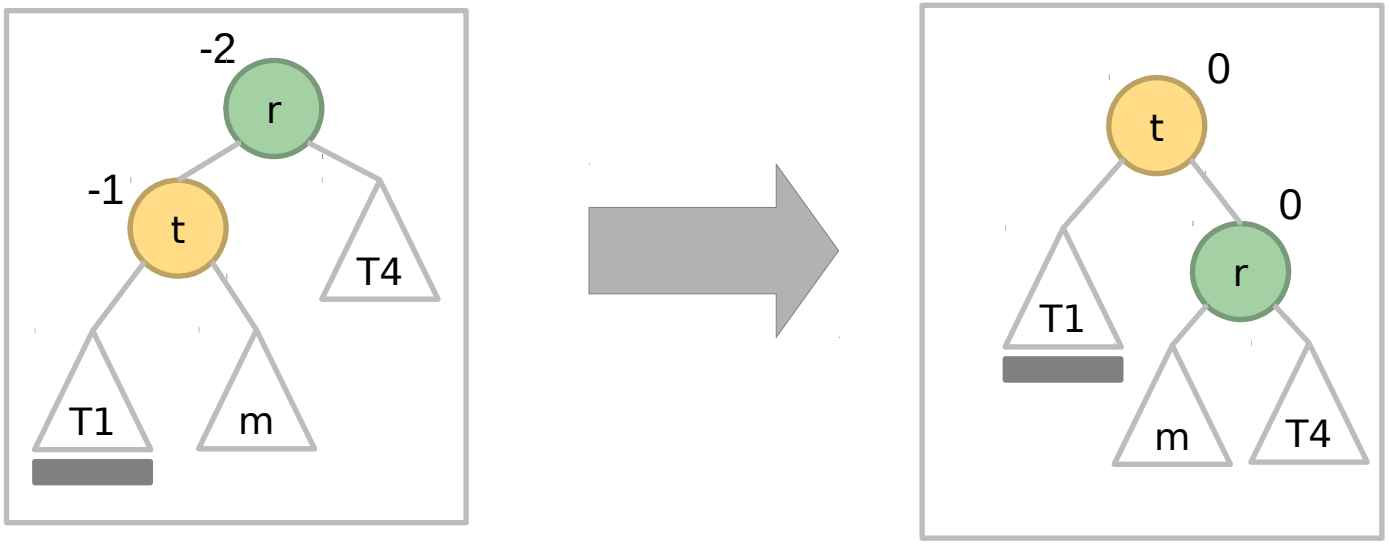
DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

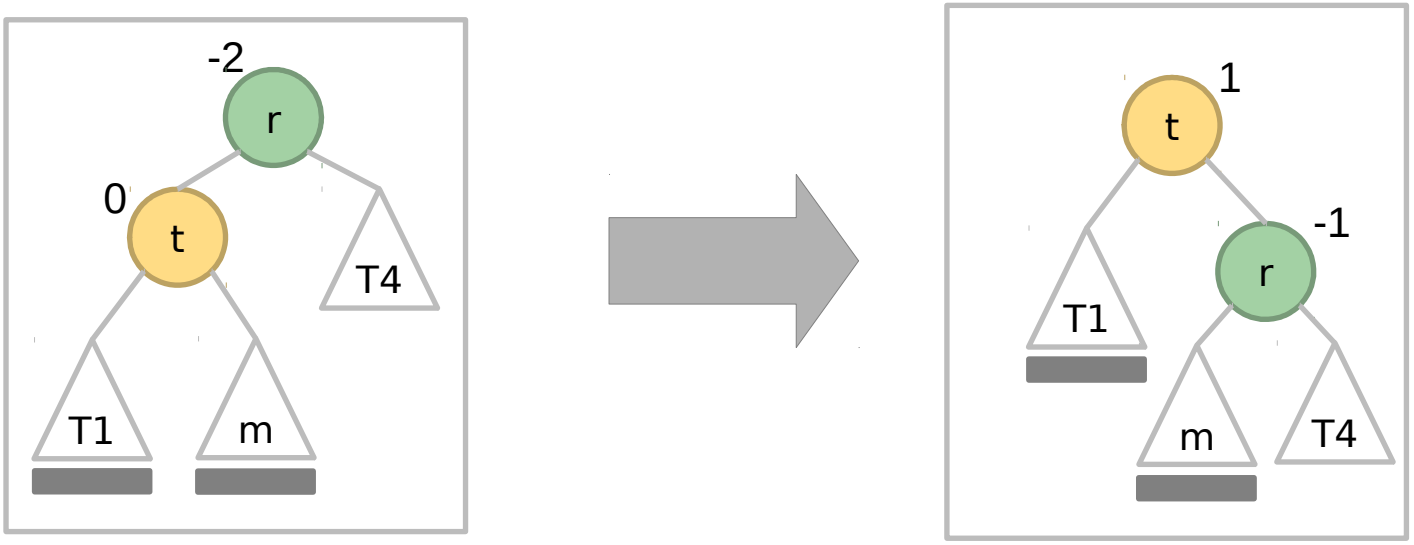
Rotação Dupla Direita-Esquerda



DEPARTAMENTO DE INFORMÁTICA
PUC·RIO

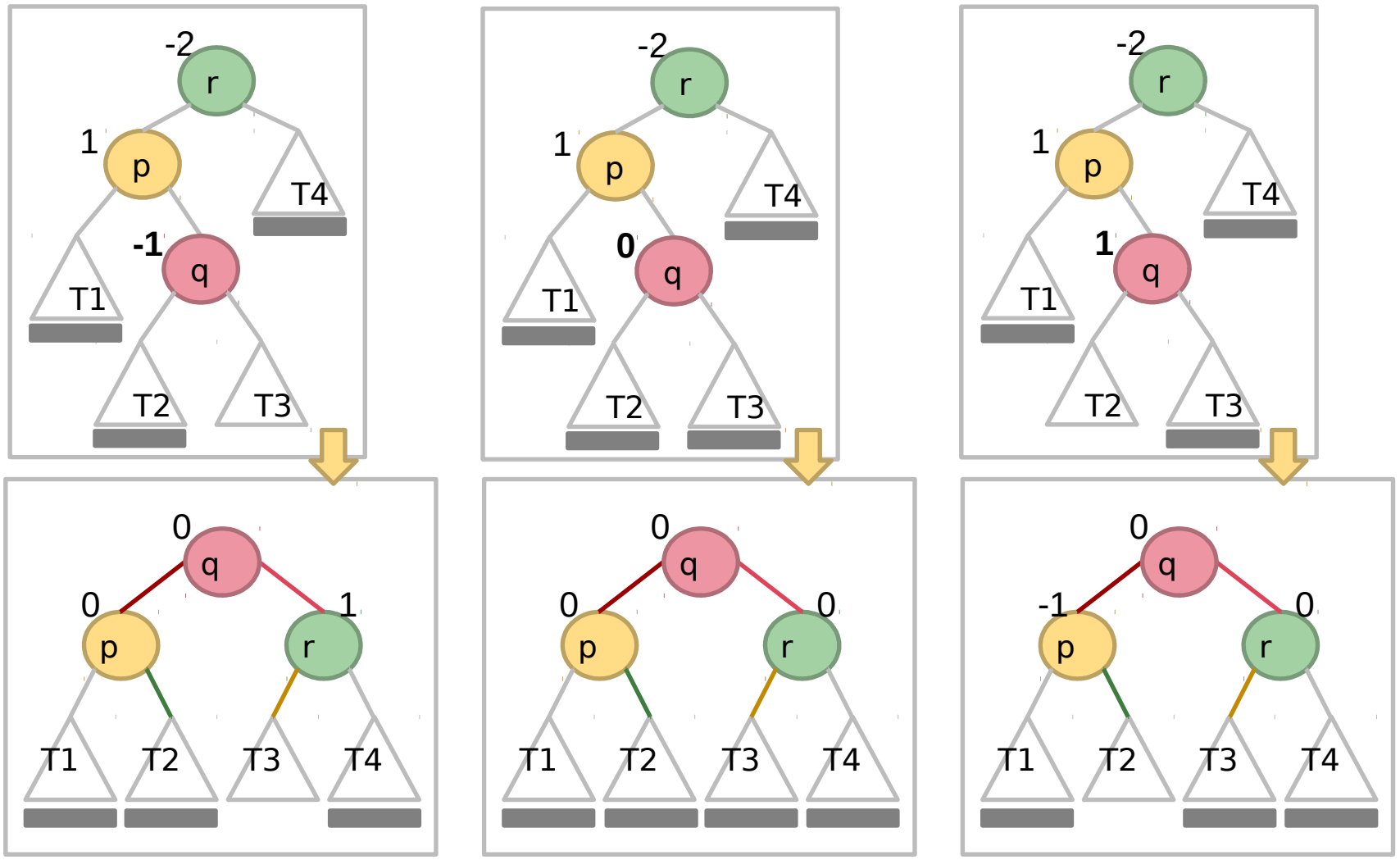
Rotação à direita (sad perdeu altura)





DEPARTAMENTO
DE INFORMÁTICA
PUC·RIO

Rotação Dupla Esquerda-Direita



DEPARTAMENTO DE INFORMÁTICA
PUC·RIO