



UFPI - CCN - DC

Estruturas de Dados

Árvores 2-3 de Busca

Prof. Raimundo Moura

rsm@ufpi.edu.br

Árvore 2-3 de Busca

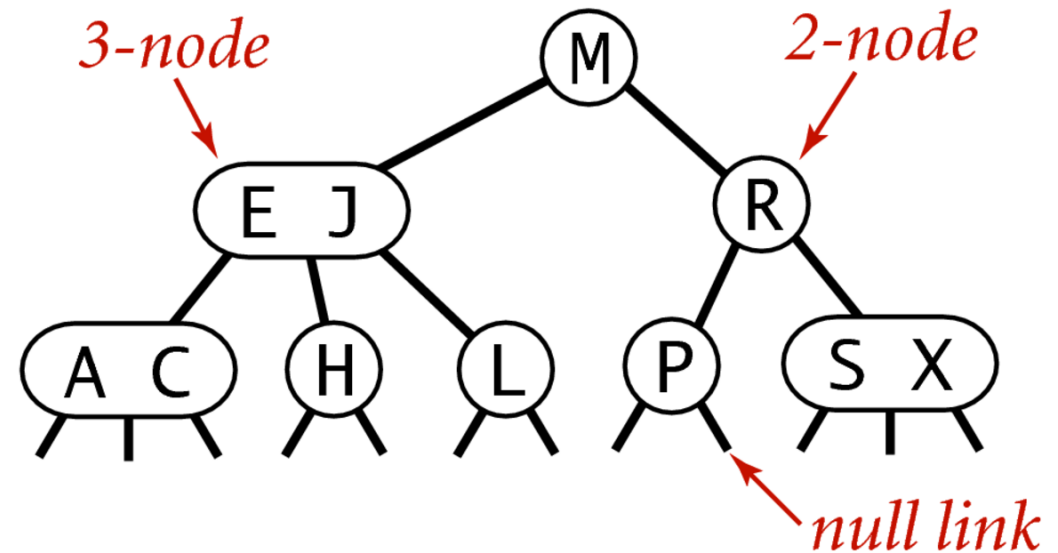
- Como implementar uma tabela de símbolos em uma **BST** de modo que a árvore permaneça aproximadamente **balanceada** (ou seja, tenha altura próxima de $\lg N$, sendo N o número de nós) qualquer que seja a sequência de buscas e inserções aplicada à árvore?

Árvore 2-3 de Busca

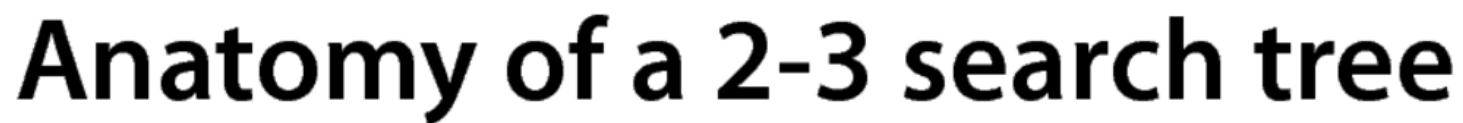
- ▶ Por que a altura de uma BST cresce?
 - ▶ Cada inserção (*put*) pode criar um novo nó e assim pode aumentar a altura da árvore.
- ▶ **Solução:** colocar mais de uma chave em cada nó!
- ▶ **Difícil:** fazer isso de modo que nenhum nó tenha mais que 2 chaves.

Árvore 2-3 de Busca

- A árvore, que era binária, torna-se ternária. Na verdade, uma mistura de binária e ternária.



Anatomy of a 2-3 search tree



Árvore 2-3 de Busca (2-3 Search Tree)

- ▶ uma árvore vazia;
- ▶ ou um *nó simples*, que contém uma chave e dois links: um link esquerdo para uma árvore 2-3 que tem chaves menores que a chave do nó e um link direito para uma árvore 2-3 que tem chaves maiores;
- ▶ ou um *nó duplo*, que contém duas chaves e três links: um link esquerdo para uma árvore 2-3 que tem chaves menores; um link do meio para uma árvore 2-3 que tem chaves entre as duas chaves do nó; e um link direito para uma árvore 2-3 que tem chaves maiores.

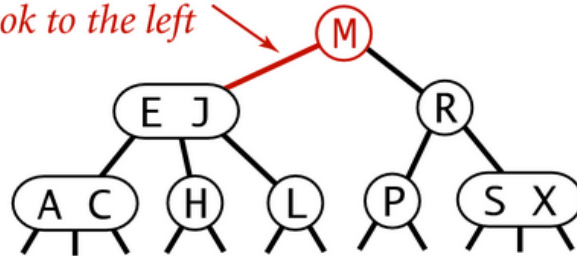
Árvore 2-3 de Busca (2-3 Search Tree)

- ▶ Árvores 2-3 têm esse nome porque cada nó tem 2 ou 3 links
- ▶ Toda árvore 2-3 é *perfeitamente balanceada*: todos os links **null** estão no mesmo nível.

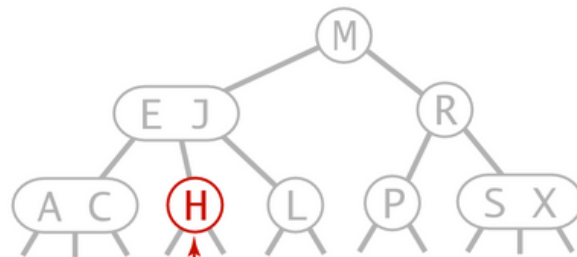
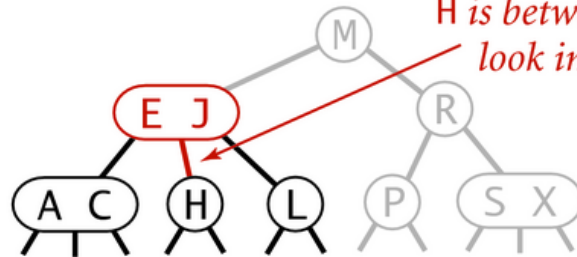
Buscas em Árvore 2-3

successful search for H

*H is less than M so
look to the left*



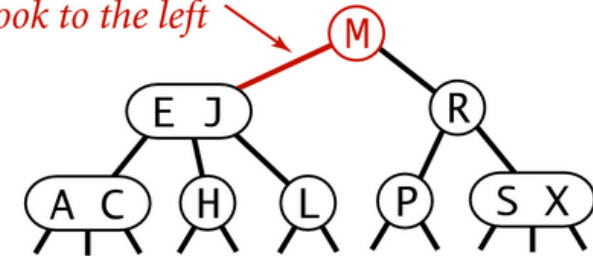
*H is between E and J so
look in the middle*



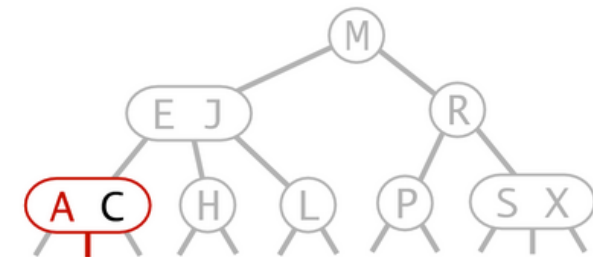
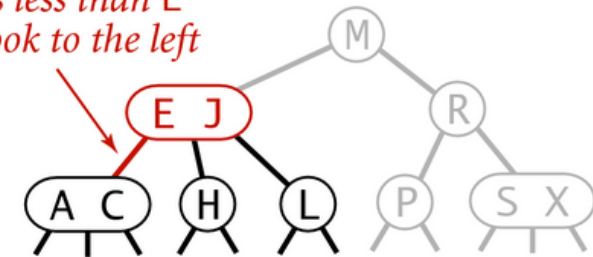
found H so return value (search hit)

unsuccessful search for B

*B is less than M so
look to the left*



*B is less than E
so look to the left*



*B is between A and C so look in the middle
link is null so B is not in the tree (search miss)*

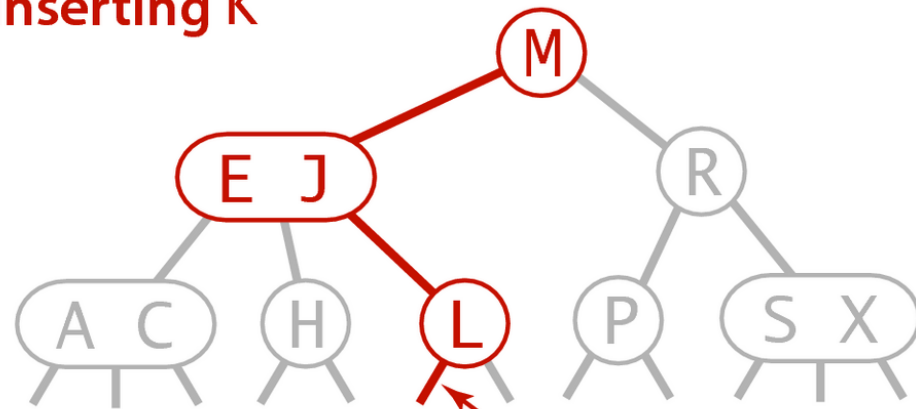
Search hit (left) and search miss (right) in a 2-3 tree

Árvore 2-3: Inserção

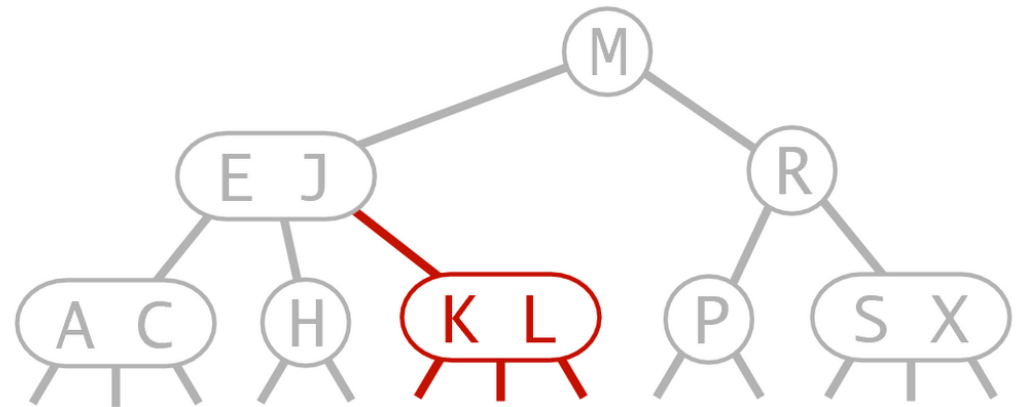
➤ Inserção em um nó simples:

- A operação não estraga o balanceamento

inserting K



search for K ends here



*replace 2-node with
new 3-node containing K*

Insert into a 2-node

Árvore 2-3: Inserção

- Inserção em um nó duplo isolado

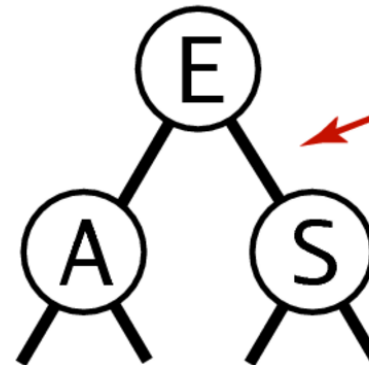
inserting S



← no room for S



← make a 4-node



split 4-node into this 2-3 tree

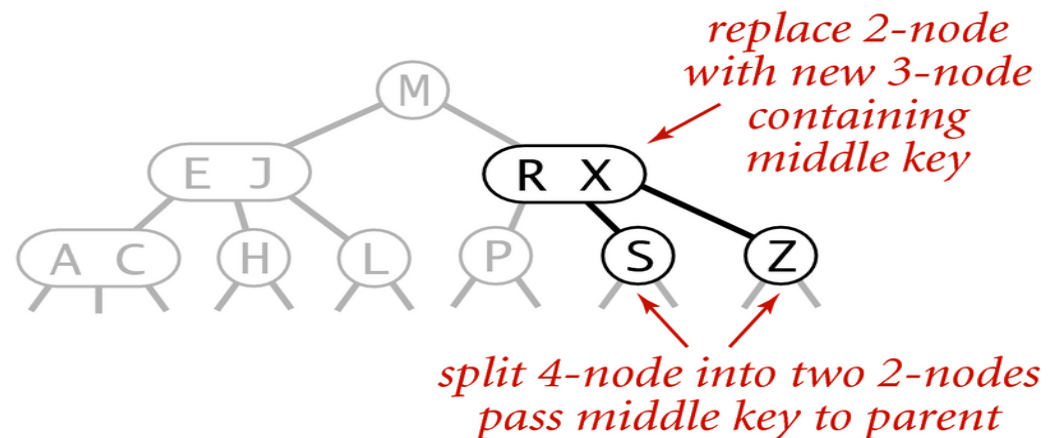
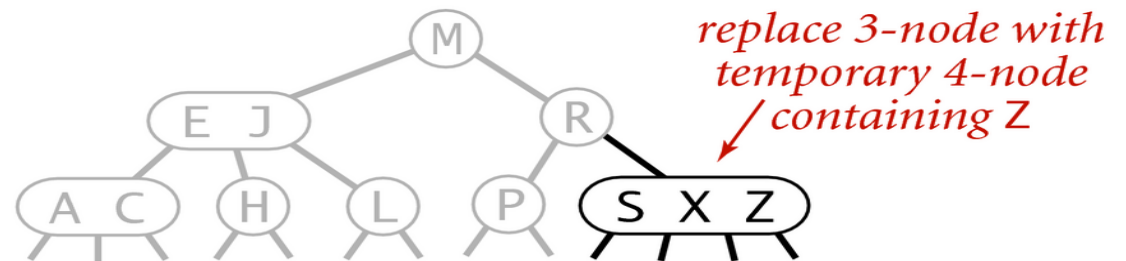
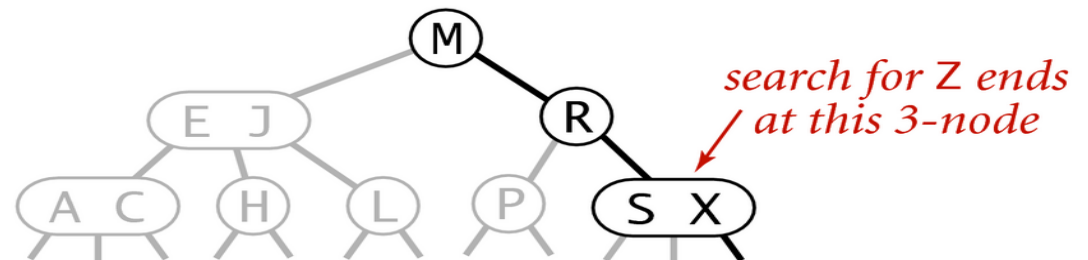
Insert into a single 3-node

Árvore 2-3: Inserção

➤ Inserção em um nó duplo cujo pai é um nó simples:

- A operação não estraga o balanceamento

inserting Z



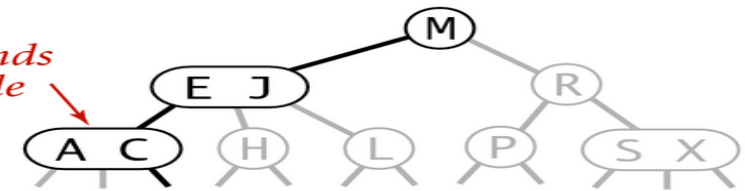
Insert into a 3-node whose parent is a 2-node

Árvore 2-3: Inserção

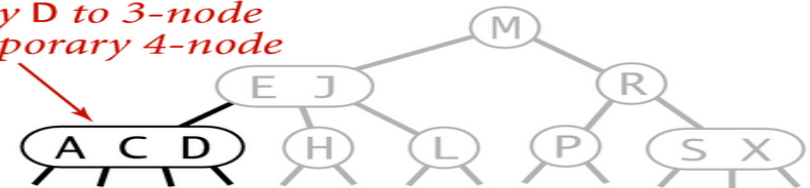
➤ Inserção em um nó duplo cujo pai é um nó duplo

inserting D

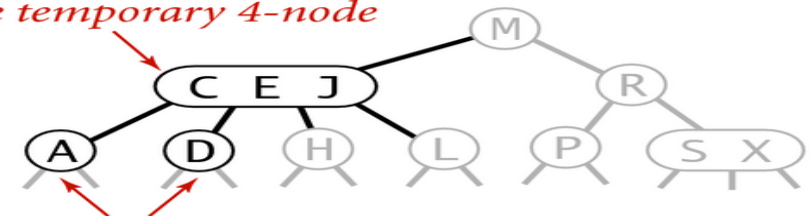
search for D ends at this 3-node



add new key D to 3-node to make temporary 4-node

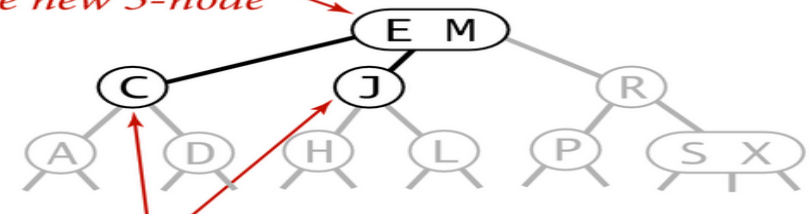


add middle key C to 3-node to make temporary 4-node



split 4-node into two 2-nodes pass middle key to parent

add middle key E to 2-node to make new 3-node



split 4-node into two 2-nodes pass middle key to parent

Insert into a 3-node whose parent is a 3-node

Árvore 2-3: Inserção

- ▶ Inserção em um nó duplo cujo pai é um nó duplo:
 - Repetir a operação subindo em direção à raiz até encontrar um nó simples (nesse caso a altura não aumenta) ou até encontrar a raiz (nesse caso a altura aumenta). A operação não estraga o balanceamento.

Árvore 2-3: Inserção

- Divisão da raiz:
 - Não estraga o balanceamento, mas aumenta a altura

inserting D

search for D ends at this 3-node



add new key D to 3-node to make temporary 4-node

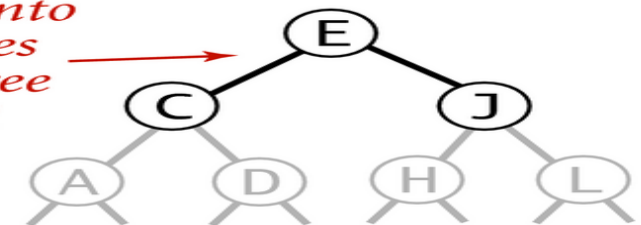


add middle key C to 3-node to make temporary 4-node



split 4-node into two 2-nodes pass middle key to parent

split 4-node into three 2-nodes increasing tree height by 1



Splitting the root

SEARCHX MPL (Parte 1)

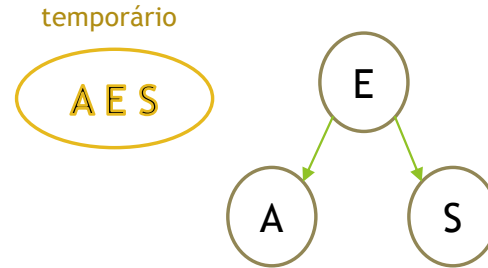
1° (S):



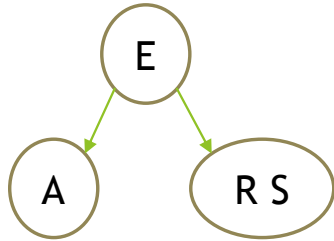
2° (E):



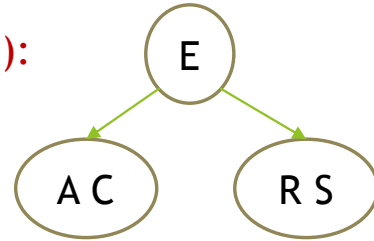
3° (A):



4° (R):



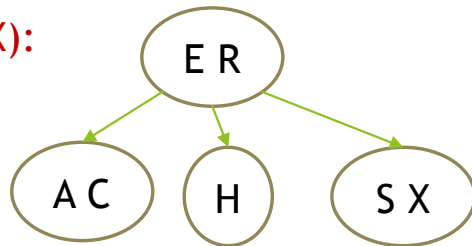
5° (C):



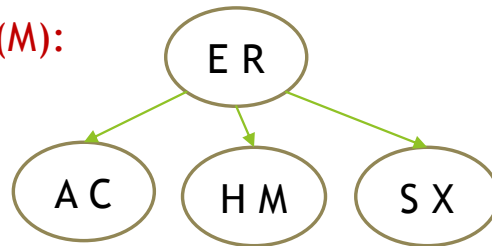
6° (H):



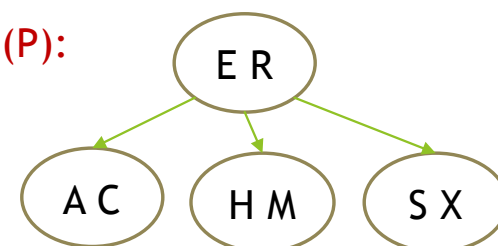
7° (X):



8° (M):

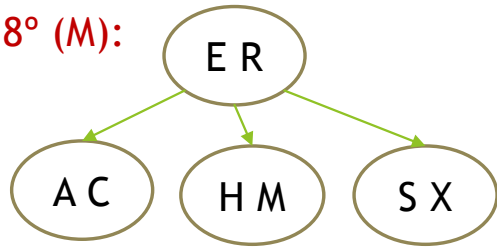


9° (P):

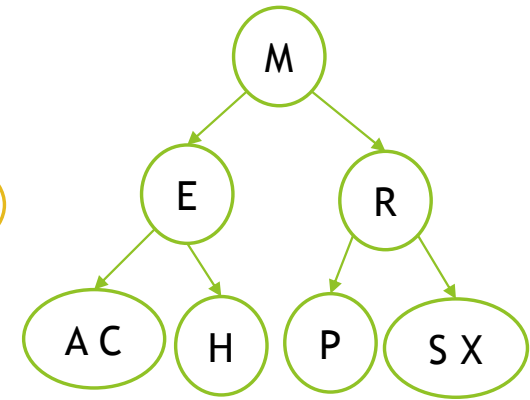
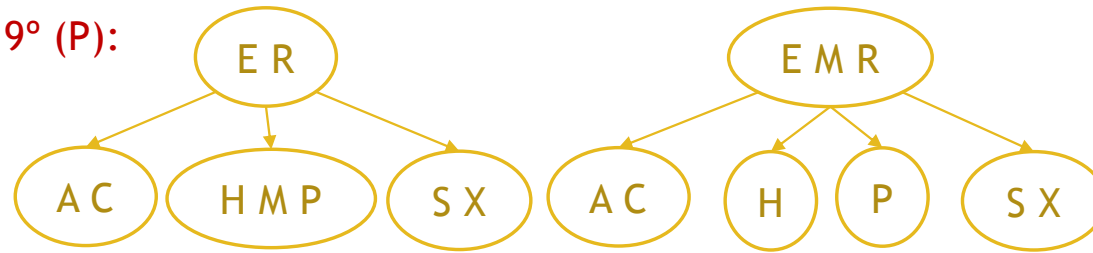


SEARCHX MPL (Parte 2)

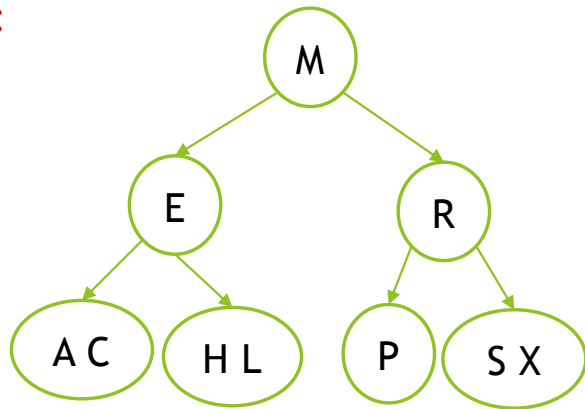
8° (M):



9° (P):



10° (L):



Árvore 2-3

➤ Construção da árvore: **passo a passo**
(Ordem qualquer)

insert S



E



A



R



C



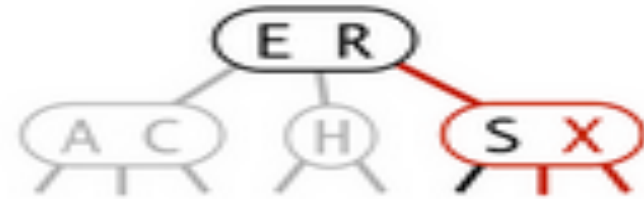
H



Árvore 2-3

➤ Construção da árvore: **passo a passo**
(Ordem qualquer)

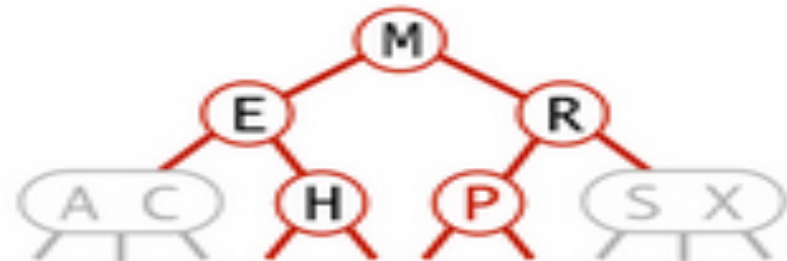
X



M



P



L



standard indexing client

Árvore 2-3 de Busca (2-3 Search Tree)

- ▶ Ver o vídeo com ilustrações de operações de busca e inserção sobre uma árvore 2-3:

- algs4.cs.princeton.edu/lectures/33Demo23Tree.mov

Atividade de participação

- 1) Inserir as seguintes chaves em uma Árvore de Busca 2-3:
 - a) JROBUSCAEXMPL
 - b) GBFIEXAMPLSRCHU

- 2) Inserir as seguintes chaves em uma Árvore de Busca Rubro-Negra:
 - a) JRBUSCAEXMPLO
 - b) GBUFIEXAMPLSRCH

