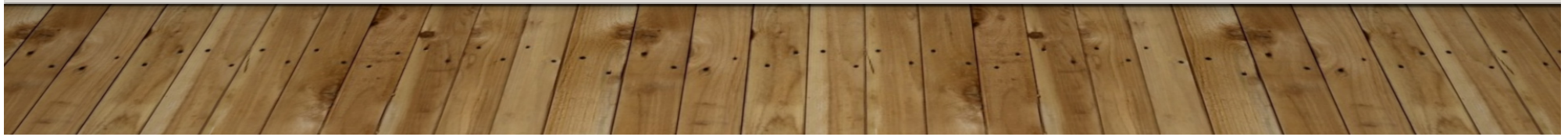
ÁRVORES TRIES

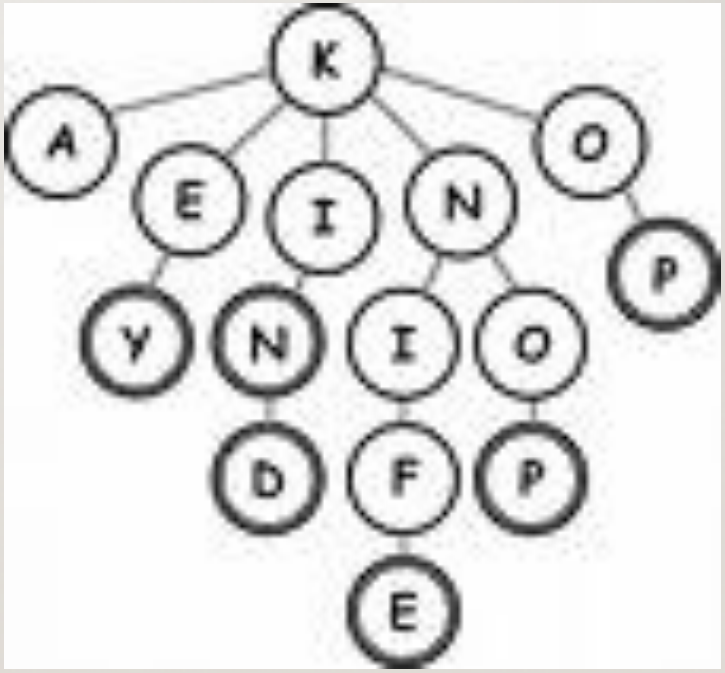


INTRODUÇÃO

- TRIE vem de RETRIEVAL – RECUPERAÇÃO
- A pronúncia: TRI ou TRAI
- É um tipo de árvore de busca como B, ISAM, quadtrees, e vermelho-preto.
- Idéia geral: usar partes das CHAVES como caminho busca
- Origem: anos 60 por Edward Fredkin

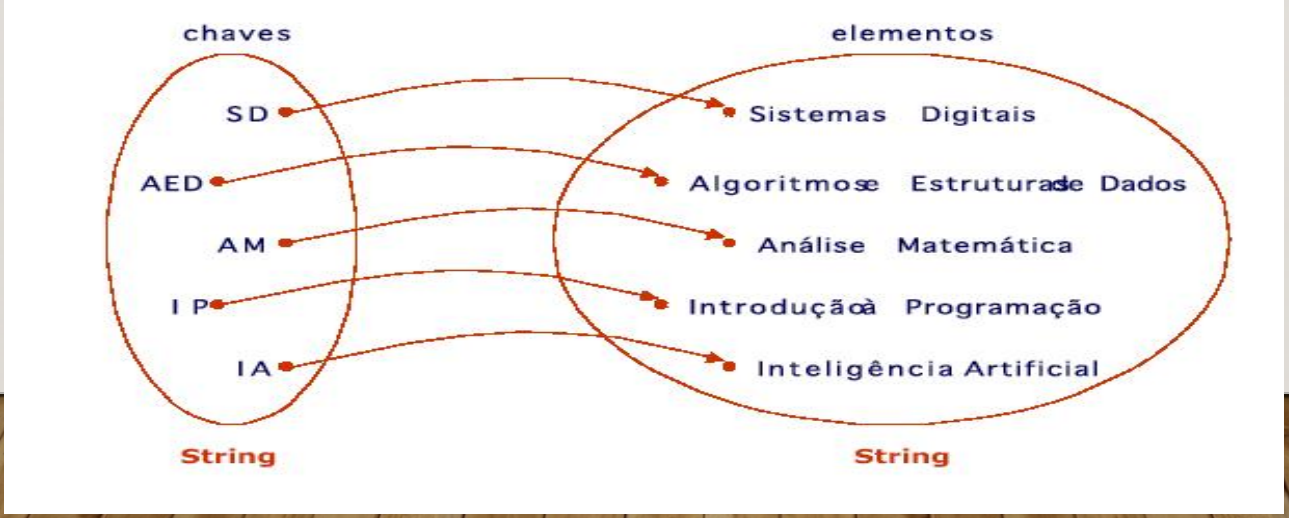
INTRODUÇÃO

Edward Fredkin (1934 -)



1. CHAVES: CARACTERÍSTICAS

- Cada chave formada por palavras sobre um alfabeto
- Palavras com tamanho variável e ilimitado
- Em geral associam-se chaves a elementos ou registros, como na tabela Hash



1. CHAVES: CARACTERÍSTICAS

- Cada chave é formada a partir de alfabeto de símbolos
- Exemplos de alfabetos: {0,1}, {A, B, C, D, E,...Z}, {0,1,2,3,4,5,...,9}
- Exemplos de chaves:
  ABABBBABABA 19034717 Maria
  01010101000000000010100000001010
- Chaves parcialmente partilhadas entre os elementos

2. TRIE: A ESTRUTURA

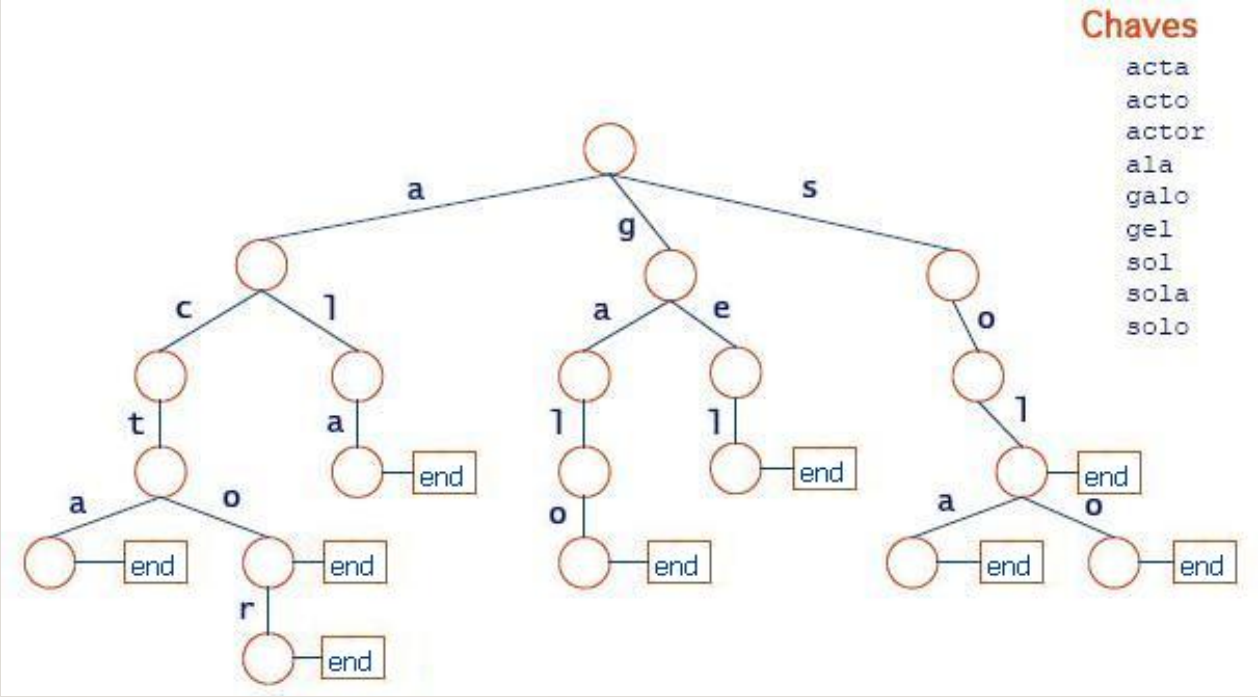
- Árvore ordenada e n-ária
- Chaves em geral caracteres
- Ao contrário da árvore de busca binária nenhum nó armazena a chave
- Chave determinada pela posição na árvore

2. TRIE: A ESTRUTURA

- Descendentes de mesmo nó com mesmo prefixo
- Raiz: cadeia vazia
- Valores ou elementos associados a folhas ou a alguns nós internos de interesse
- O caminho da raiz para qualquer outro nó é um prefixo de uma string

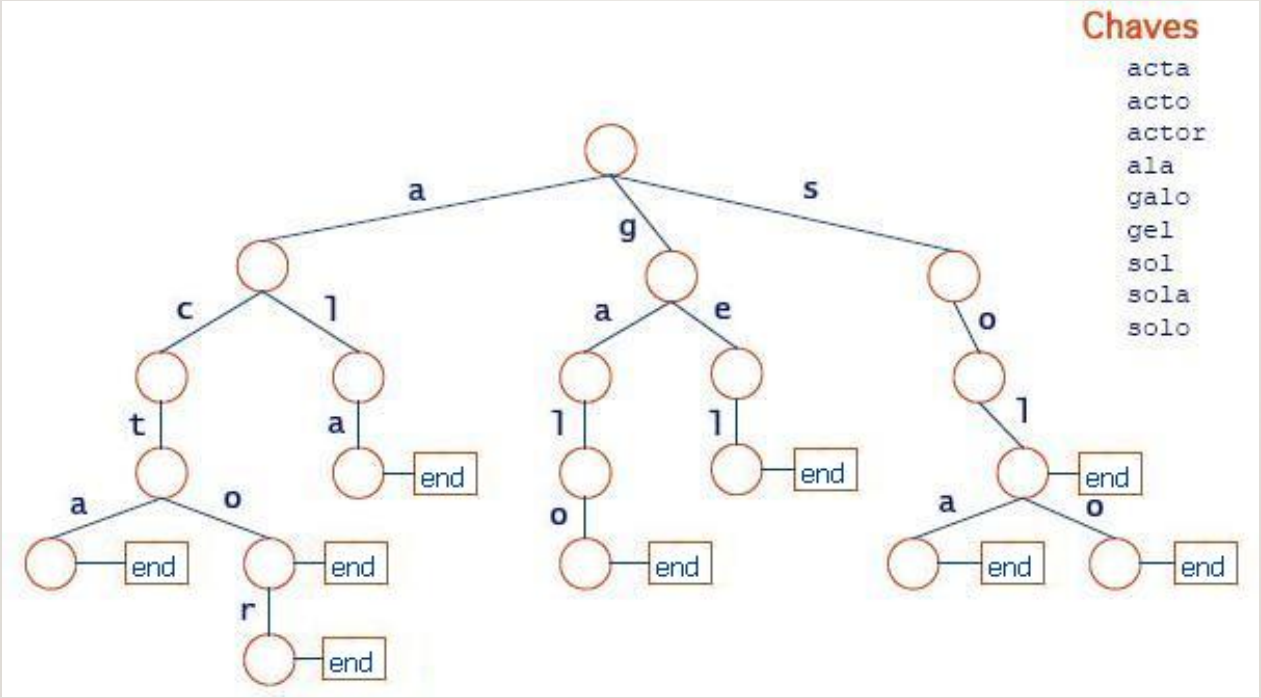
2. TRIE: A ESTRUTURA

- Árvore ordenada e n-ária



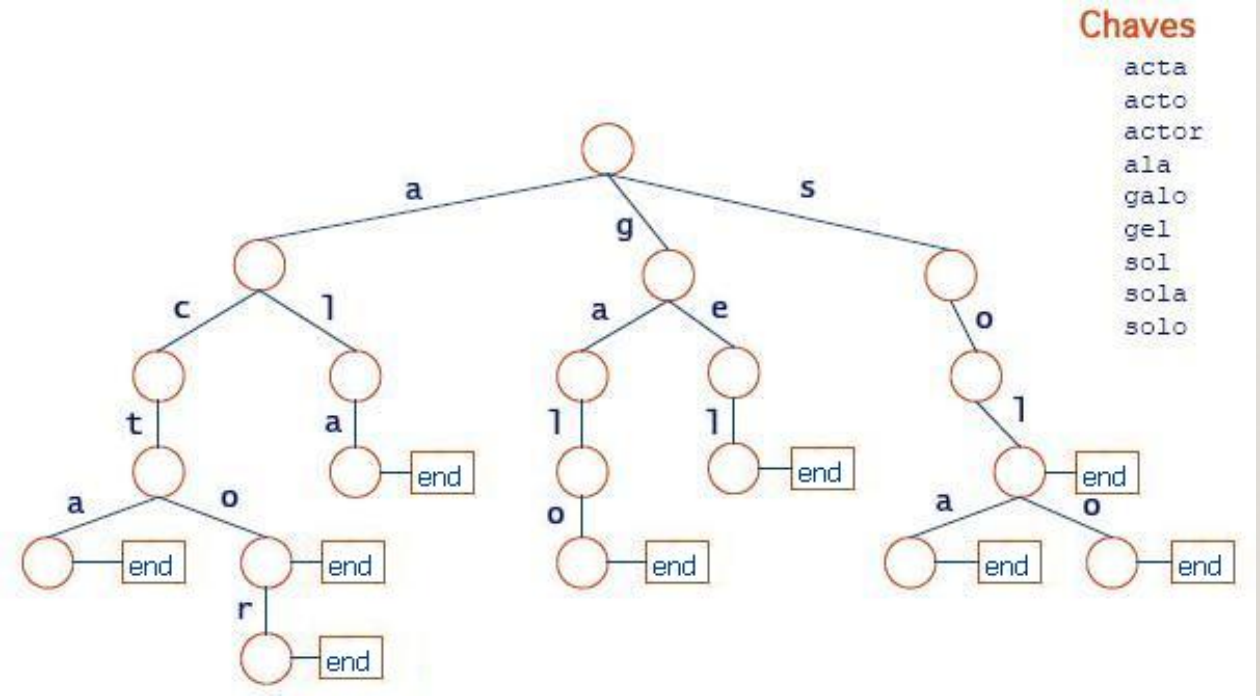
2. TRIE: A ESTRUTURA

- Chaves em geral caracteres



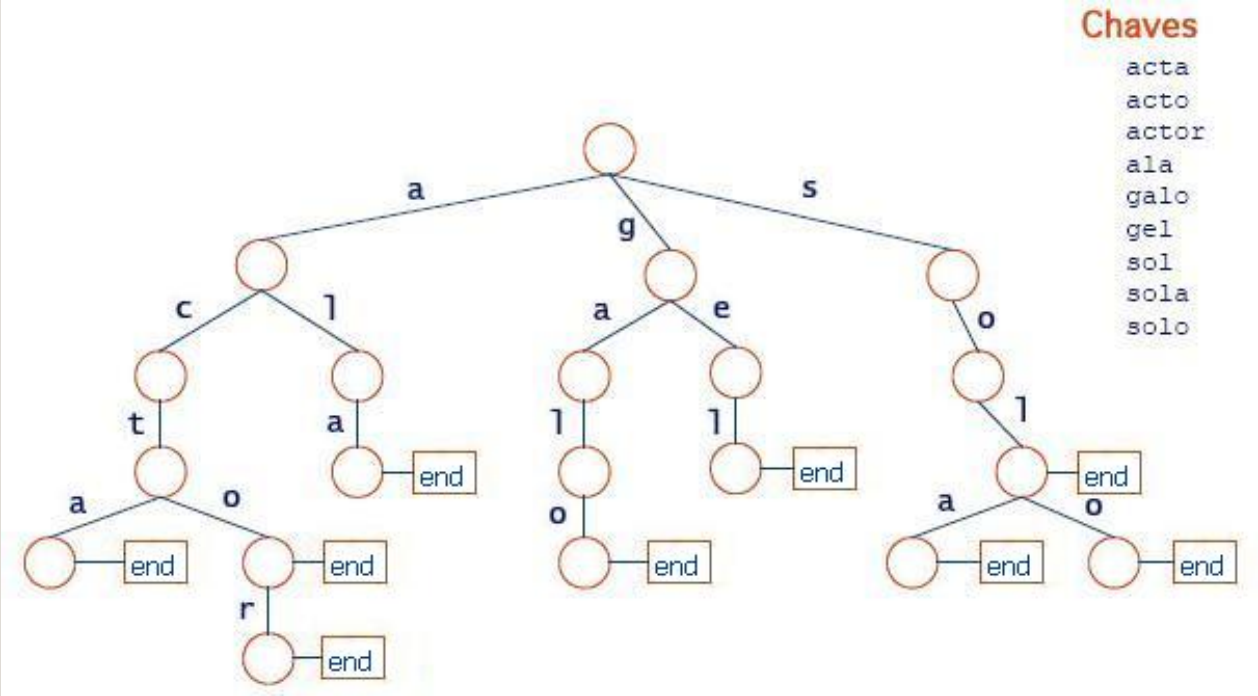
2. Trie: A estrutura

- Ao contrário da árvore de busca binária nenhum nó armazena a chave



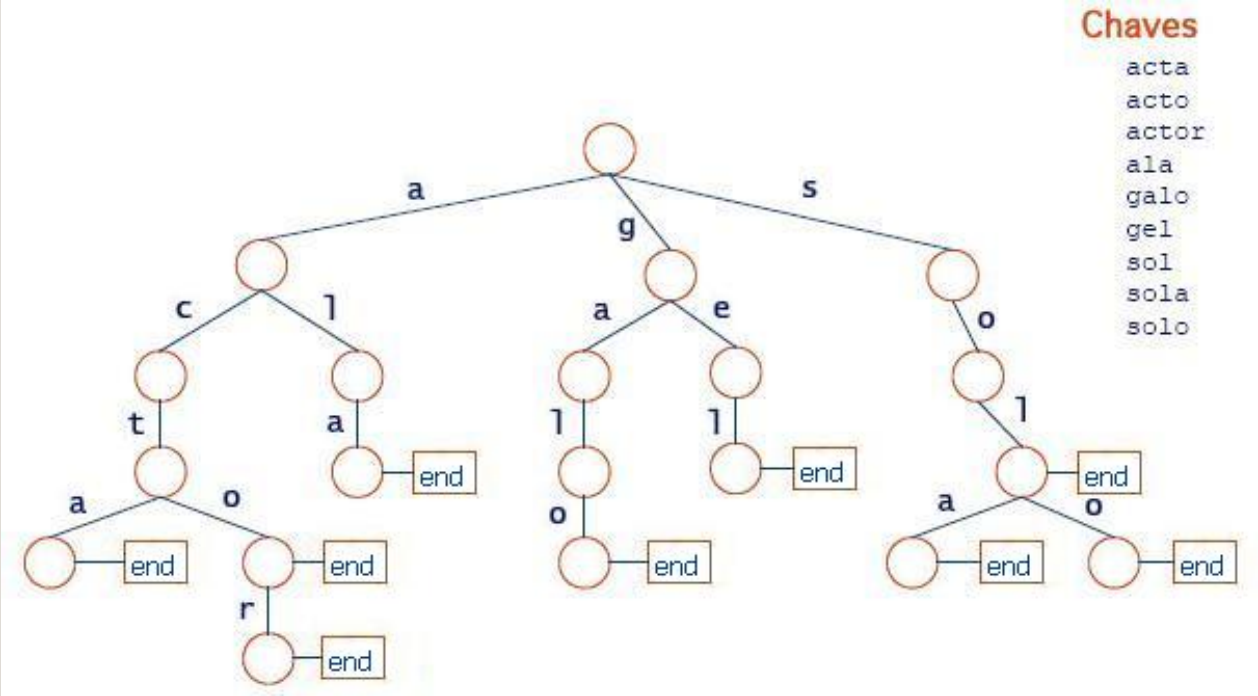
2. Trie: A estrutura

- Chave determinada pela posição na árvore



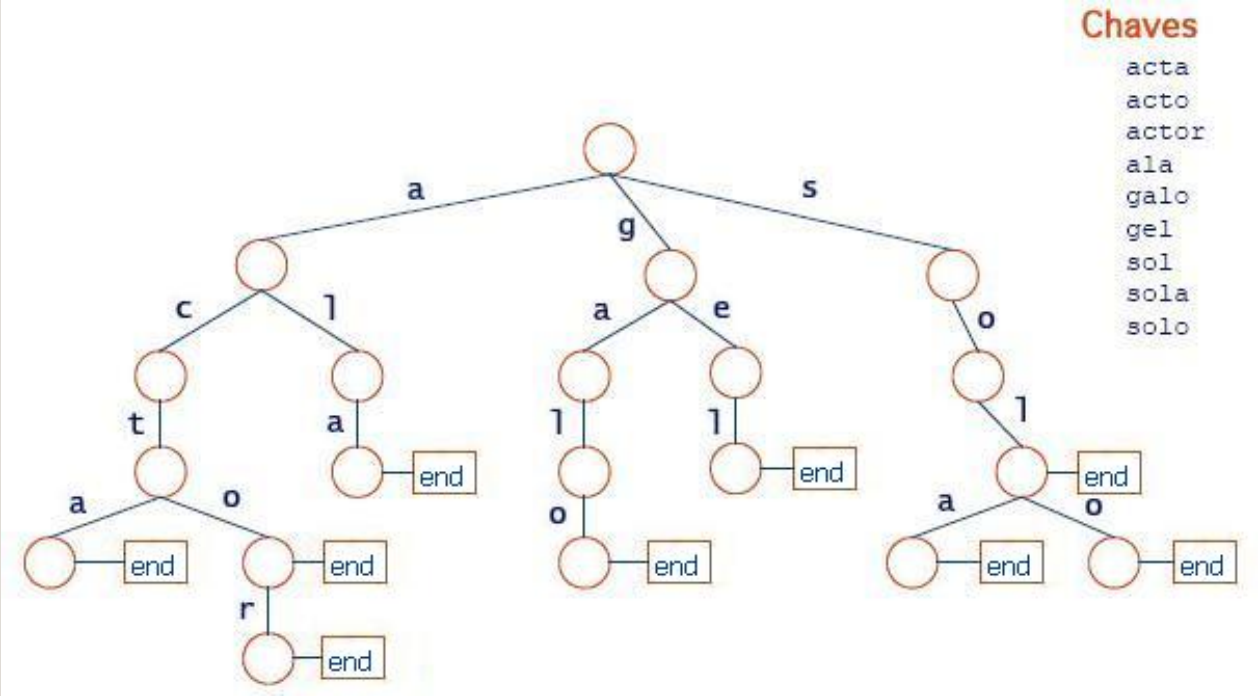
2. Trie: A estrutura

- Descendentes de mesmo nó com mesmo prefixo



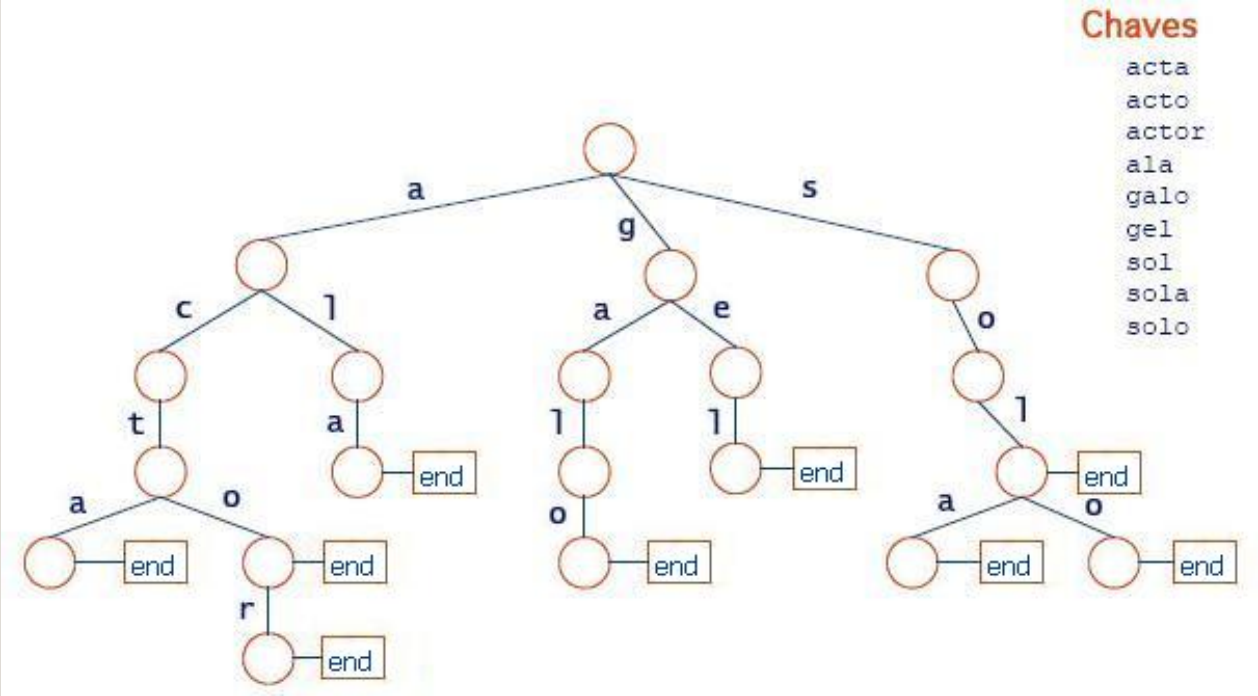
2. Trie: A estrutura

- Raiz: cadeia vazia



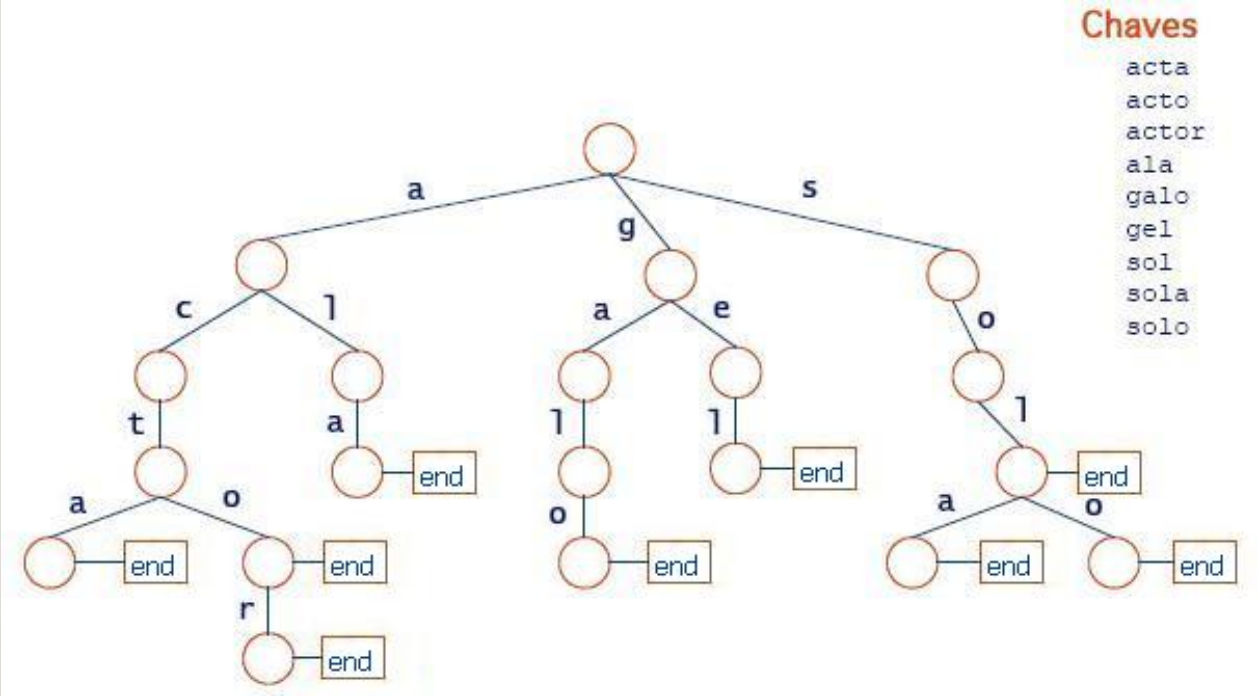
2. Trie: A estrutura

- Valores ou elementos associados a folhas ou a alguns nós internos de interesse



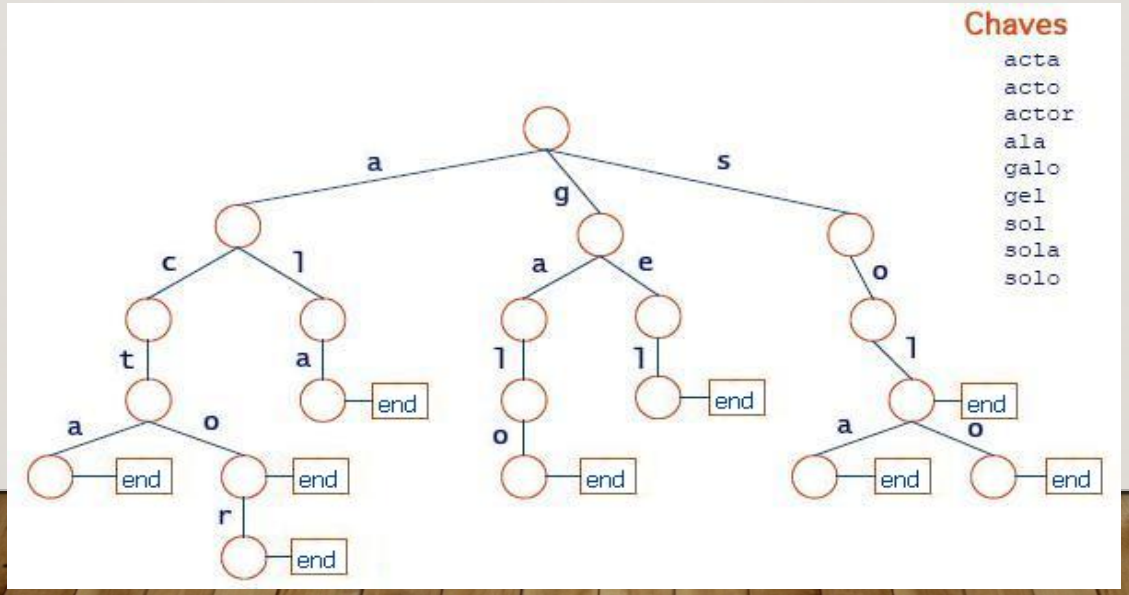
2. Trie: A estrutura

- O caminho da raiz para qualquer outro nó é um prefixo de uma string



2. Trie: A estrutura

- O grau corresponde ao tamanho do alfabeto
- A trie pode ser vista como um autômato finito
- Cada nível que se desce corresponde a avançar um elemento na chave



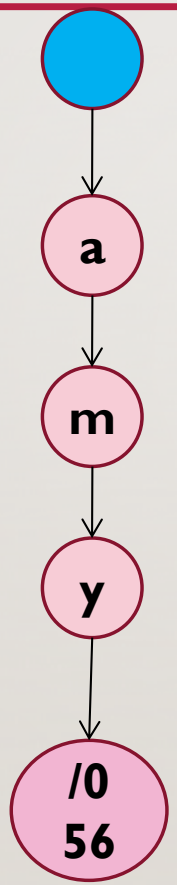
3. MONTANDO UMA ÁRVORE TRIE

- amy 56
- ann 15
- emma 30
- rob 27
- roger 52

Estes são pares que queremos colocar na árvore TRIE

3. MONTANDO UMA ÁRVORE TRIE

- amy 56



<- Nível 0 (RAIZ)

<- Nível 1

<- Nível 2

<- Nível 4

<- Nível 5

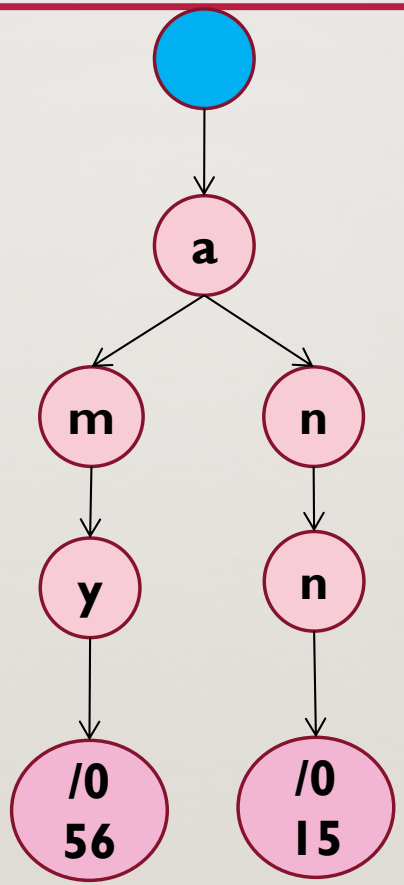
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

ann 15

3. MONTANDO UMA ÁRVORE TRIE

- ann 15



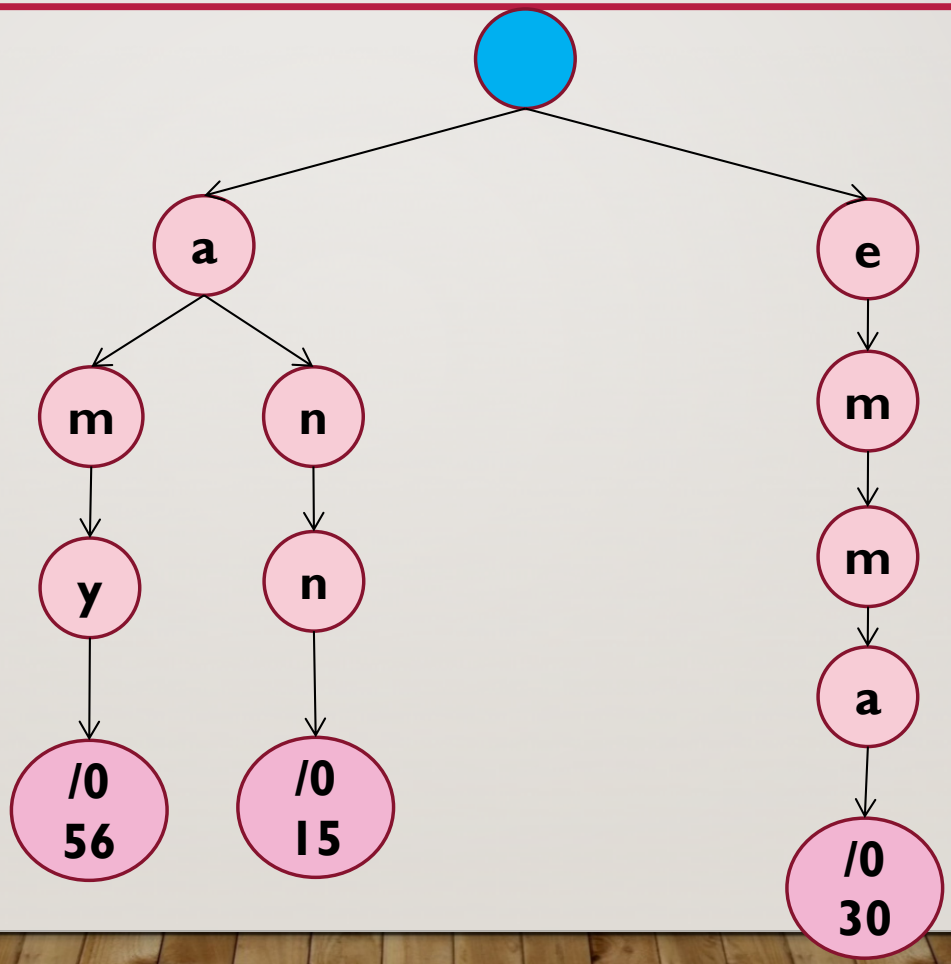
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

emma 30

3. MONTANDO UMA ÁRVORE TRIE

- emma 30



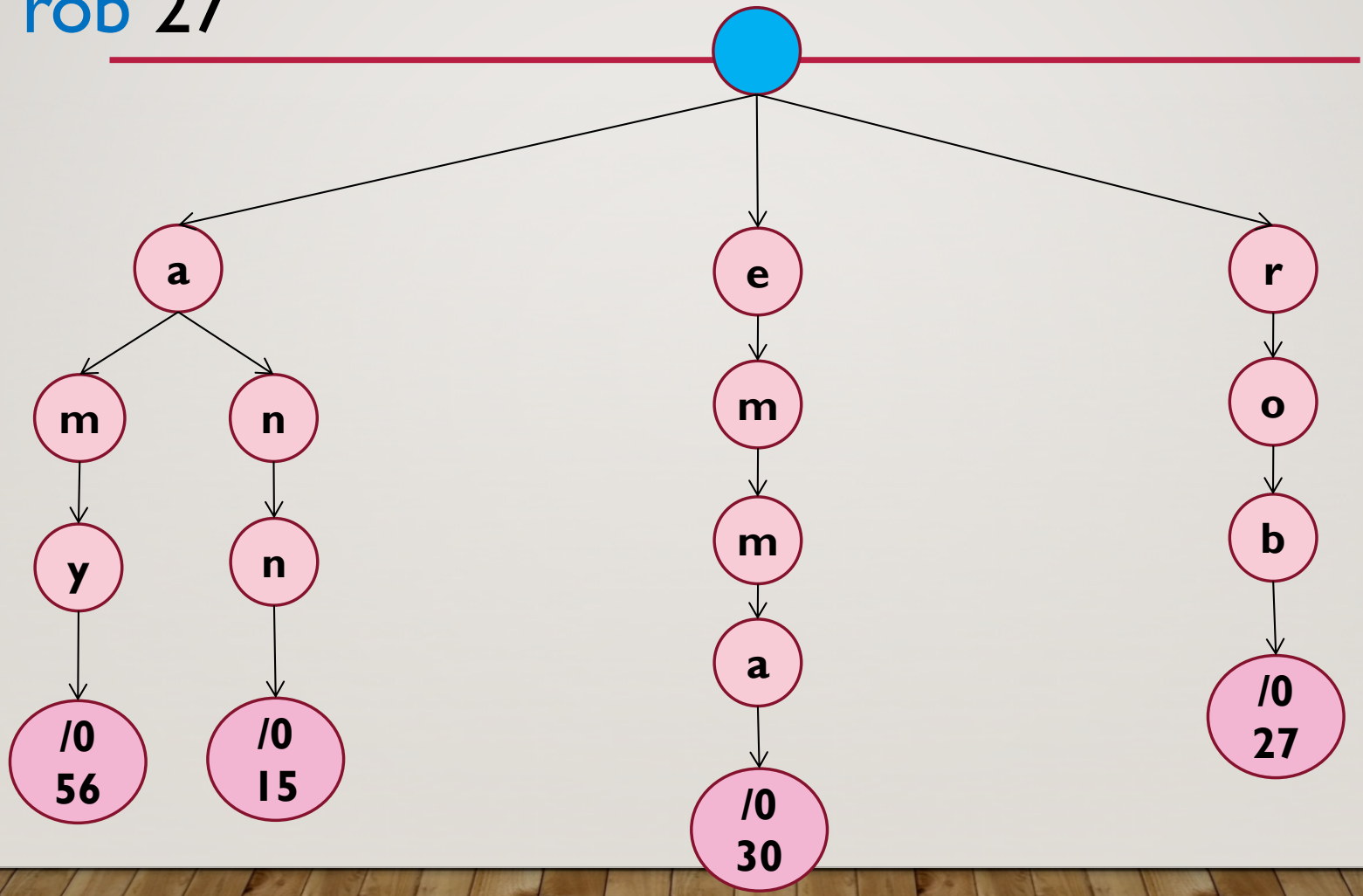
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

rob 27

3. MONTANDO UMA ÁRVORE TRIE

- rob 27



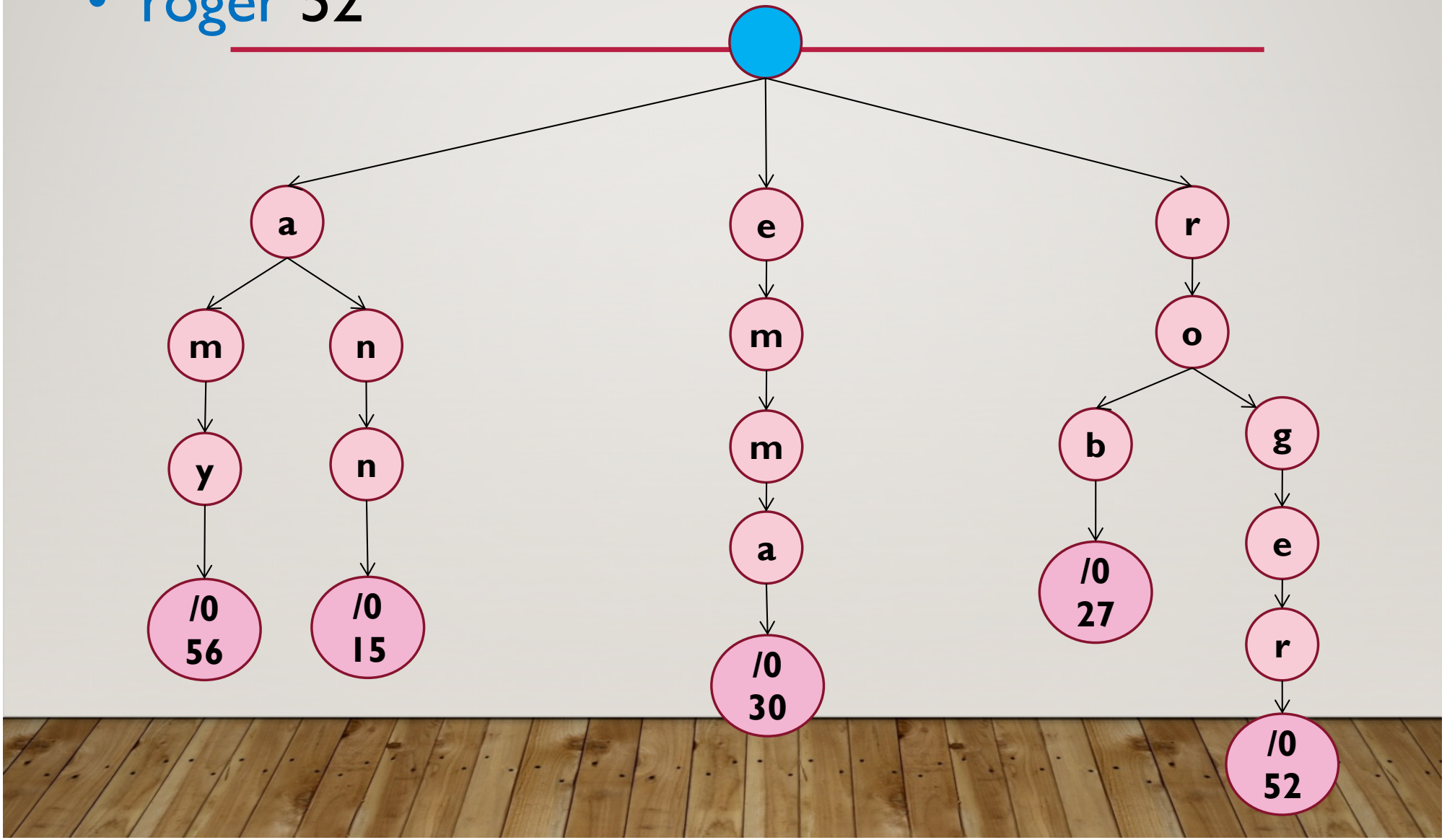
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

roger 52

3. MONTANDO UMA ÁRVORE TRIE

- roger 52



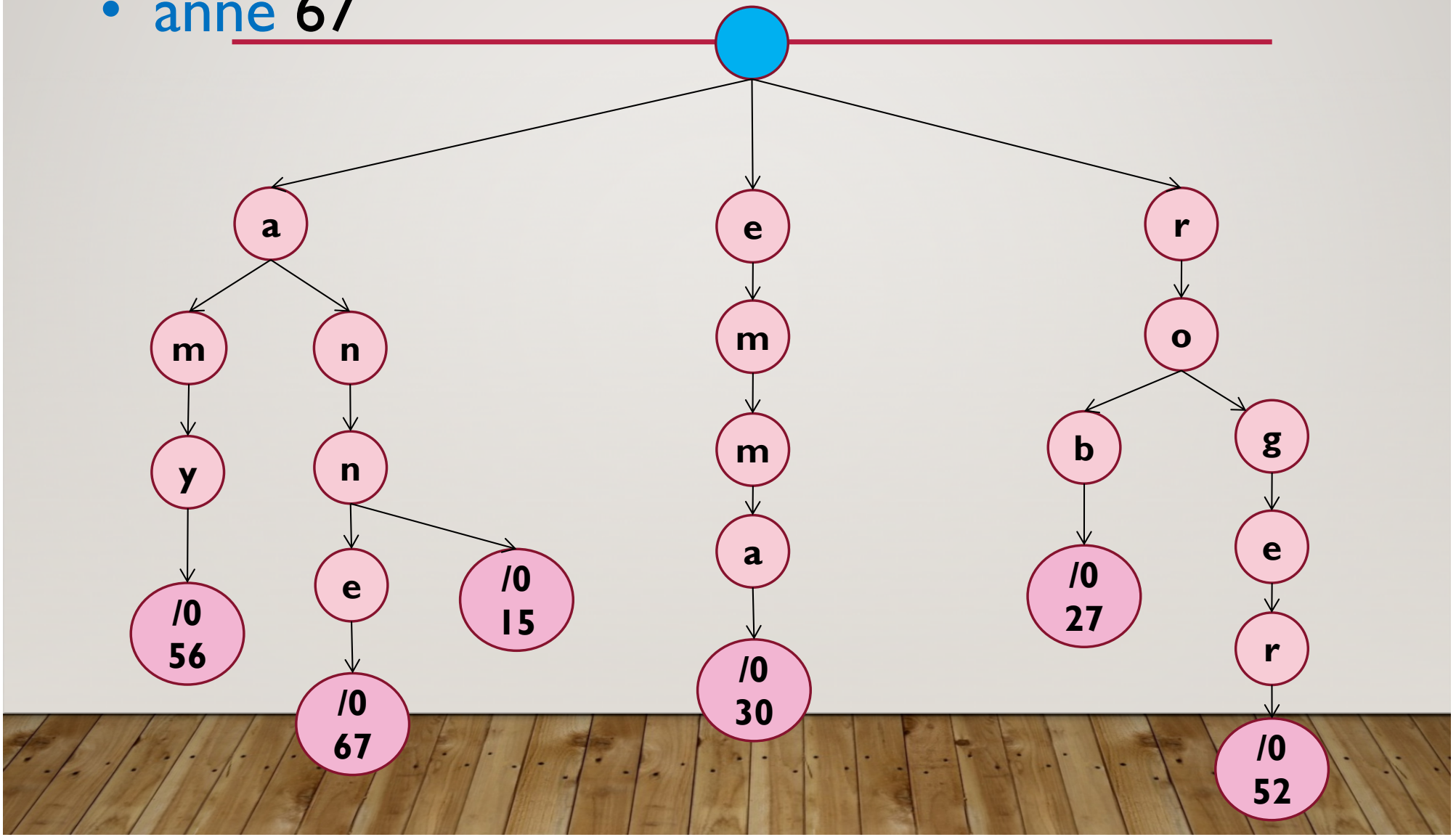
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

anne 67

3. MONTANDO UMA ÁRVORE TRIE

- anne 67



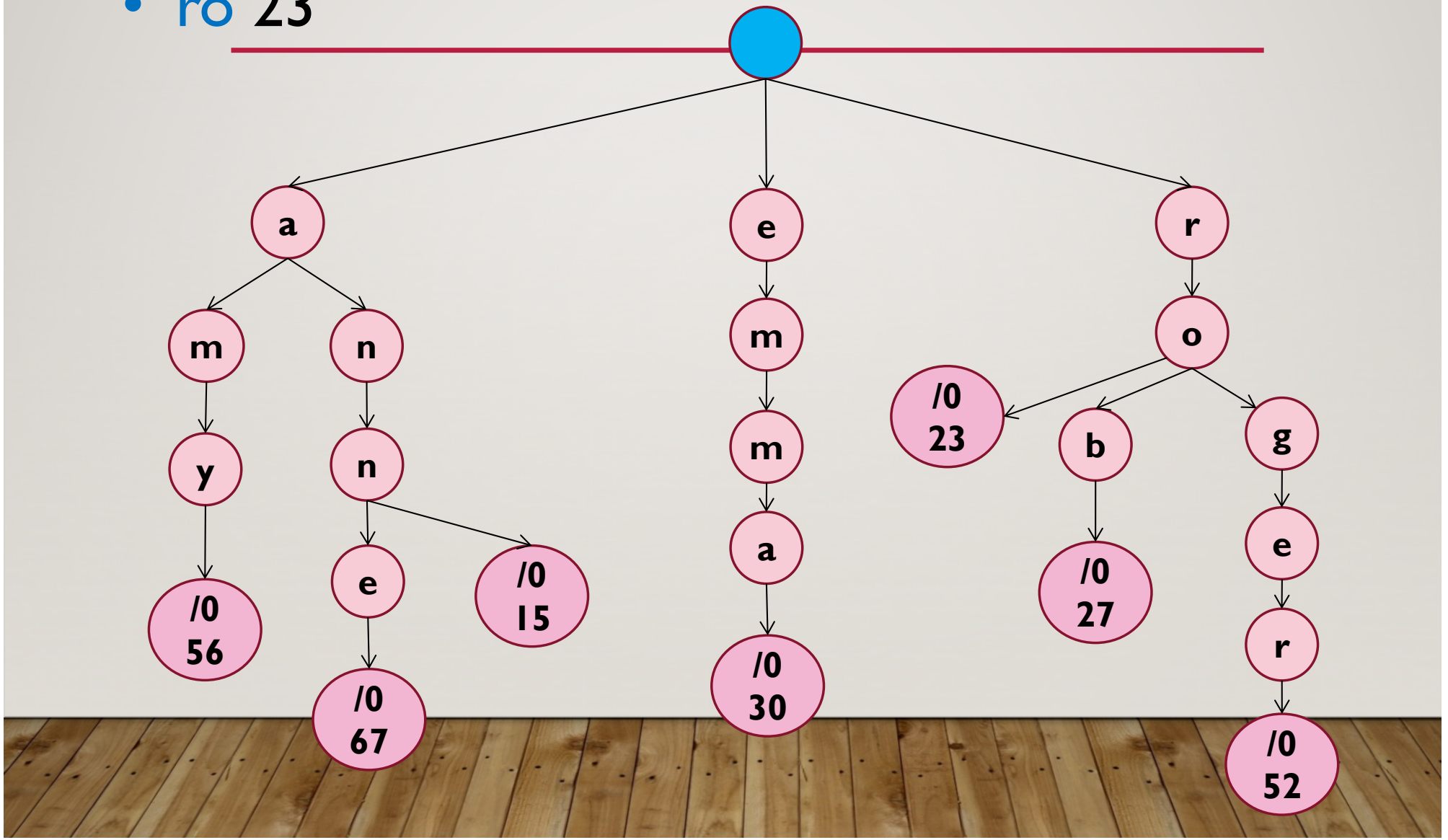
3. MONTANDO UMA ÁRVORE TRIE

- INSIRA

ro 23

3. MONTANDO UMA ÁRVORE TRIE

- ro 23



4. OPERAÇÕES EM TRIES

- INSERÇÃO
- ALGORITMO "É MEMBRO"
- REMOÇÃO

4. OPERAÇÕES EM TRIES

- INSERÇÃO

Faz-se uma busca pela palavra a ser inserida.

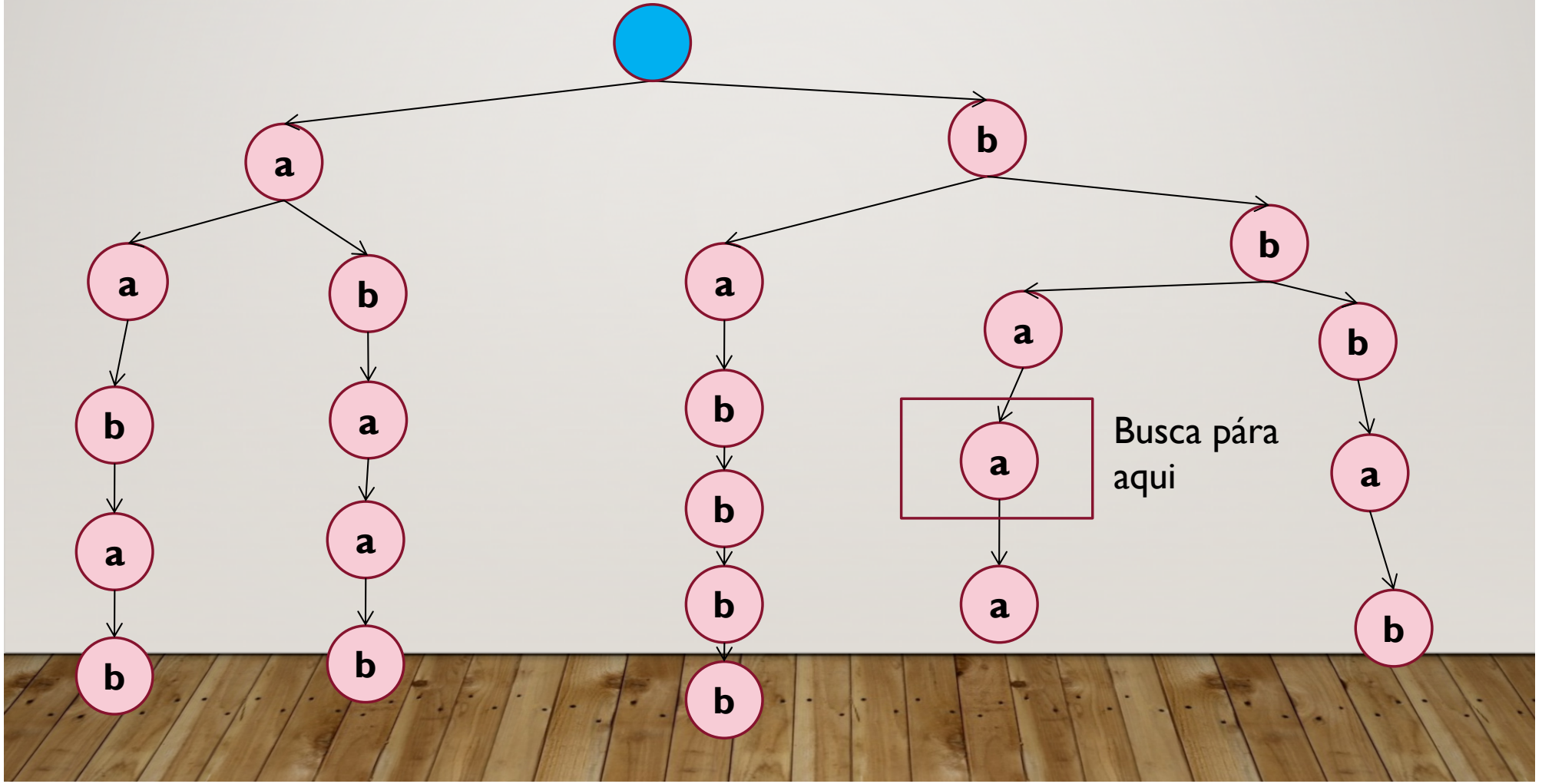
Se ela já existir na TRIE nada é feito.

Caso contrário, é recuperado o nó até onde acontece a maior substring da palavra a ser inserida.

O restante dos seus caracteres são adicionados na TRIE a partir daquele nó

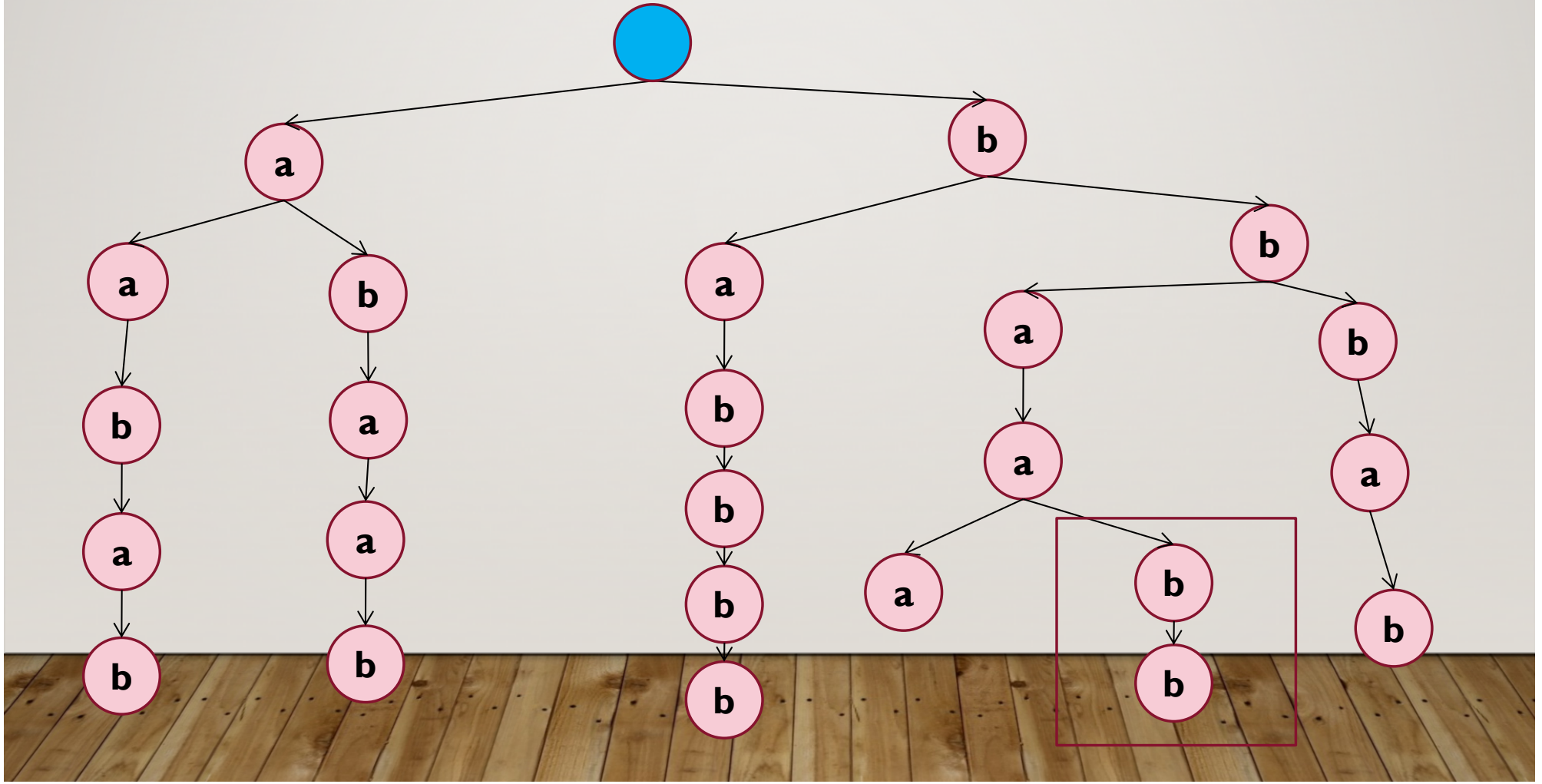
4. OPERAÇÕES EM TRIES

Inserção : bbaabb



4. OPERAÇÕES EM TRIES

Inserção : bbaabb



4. OPERAÇÕES EM TRIES

- É MEMBRO

1. Busca no nível superior o nodo que confere com o primeiro caractere (corrente) da chave
2. Se nenhum, retorna FALSO

   Senão
3. Se o caractere que confere é \0

   Retorna Verdadeiro

   Senão
4. Move para a subTRIE que confere com esse caractere
5. Avança para o próximo caractere na chave
6. Vá para passo 1

4. OPERAÇÕES EM TRIES

- REMOÇÃO

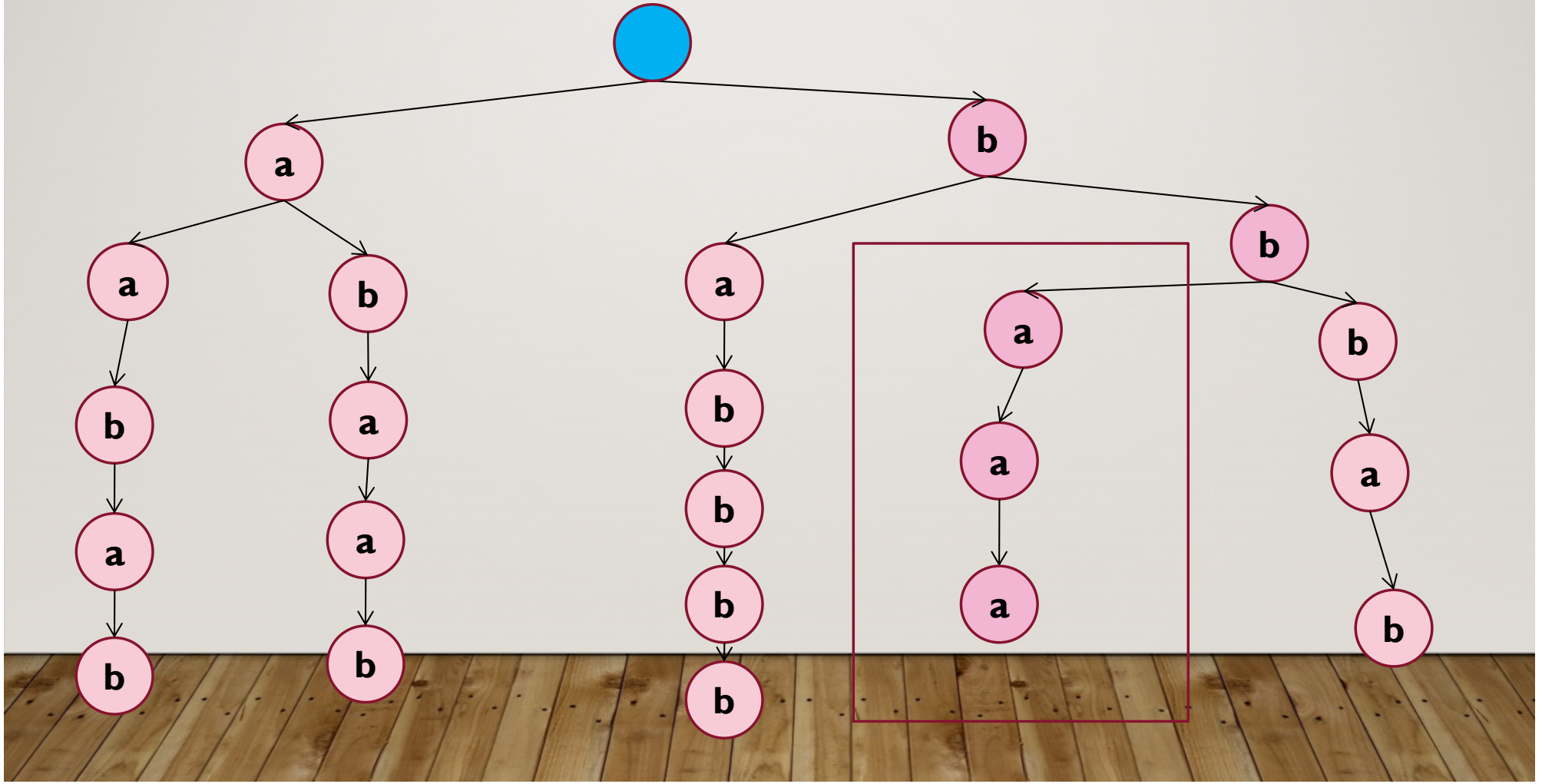
Busca-se o nó que representa o final da palavra a ser removida.

São removidos os nós que possuem apenas um filho pelo caminho ascendente.

A remoção é concluída quando se encontra um nó com mais de um filho

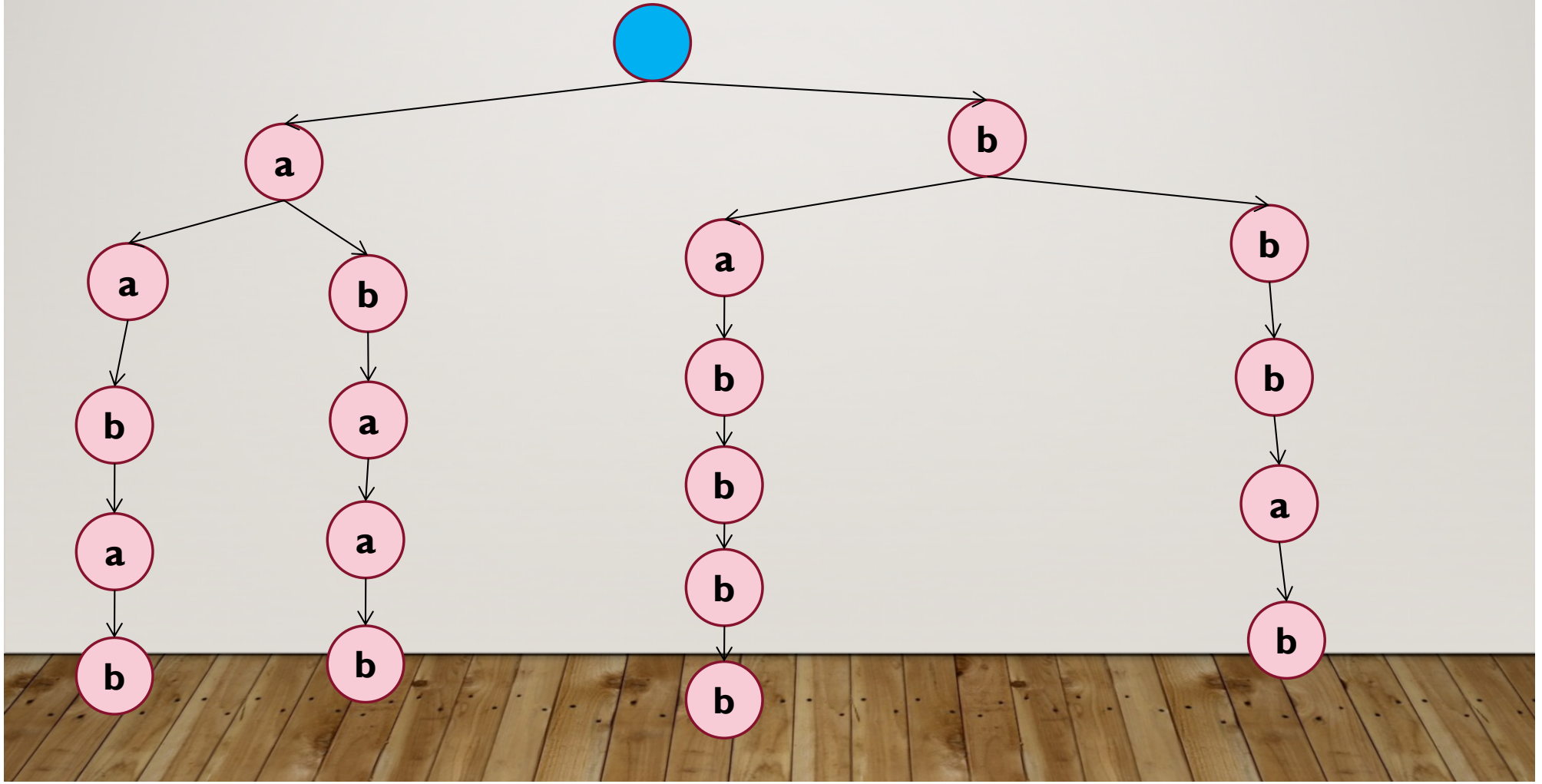
4. OPERAÇÕES EM TRIES

Remoção : bbaaa



4. OPERAÇÕES EM TRIES

Remoção : bbaaa



4. OPERAÇÕES EM TRIES

- COMPLEXIDADE

A altura da árvore é igual ao comprimento da chave mais longa

O tempo de execução das operações não depende do número de elementos da árvore

Complexidade: O (AK)

A = tamanho do alfabeto

K = tamanho da chave

A utilização de uma TRIE só compensa se o acesso aos componentes individuais das chaves for bastante rápido

Quanto maior a estrutura mais eficiente uso do espaço

Para enfrentar o desperdício de espaço com estruturas pequenas foram criadas as árvores PATRÍCIA

5. TIPOS DE TRIES

- Existem muitas variantes e tipos de TRIES
  - R-WAY
  - DST (Digital Search Tree)
  - Suffix Tree
  - Patrícia Tree
  - DAWG (Directed Acyclic Word Graph)
  - TST
  - etc

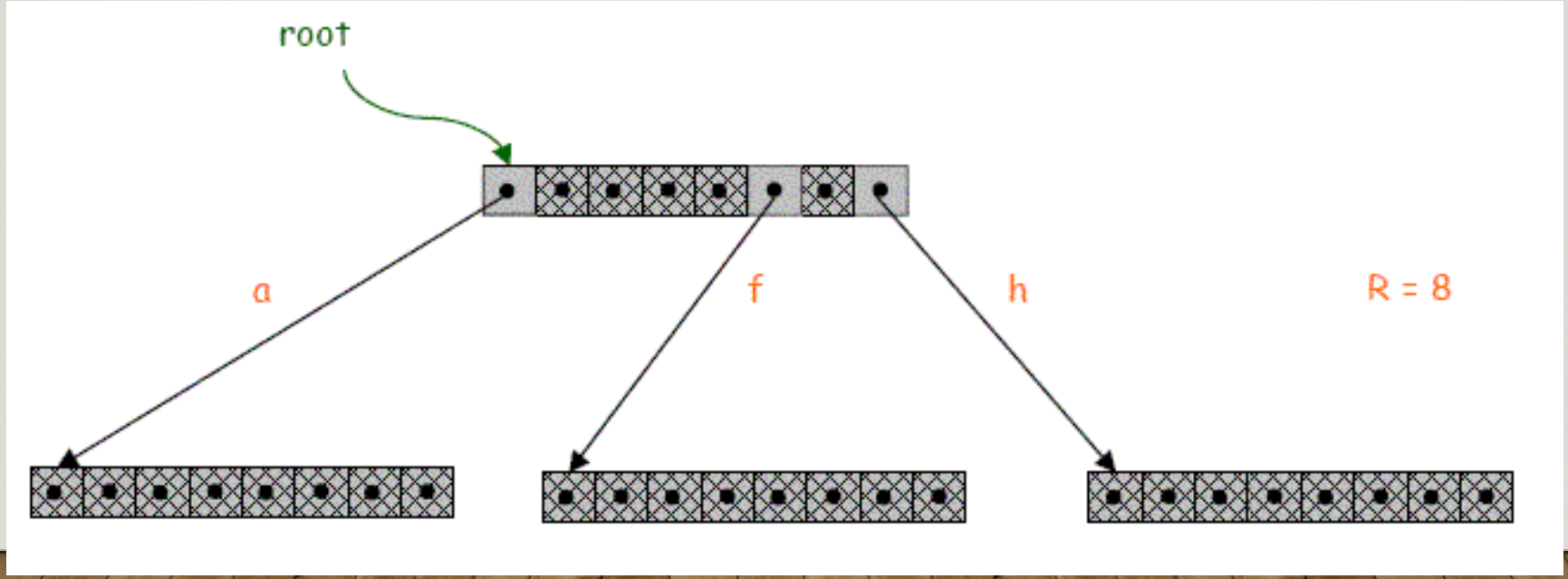
5. TIPOS DE TRIES

R-WAY

Cada nó aloca espaço para todo o alfabeto

Há desperdício de espaço.

Descendentes diretos de um pai ficam num nó só



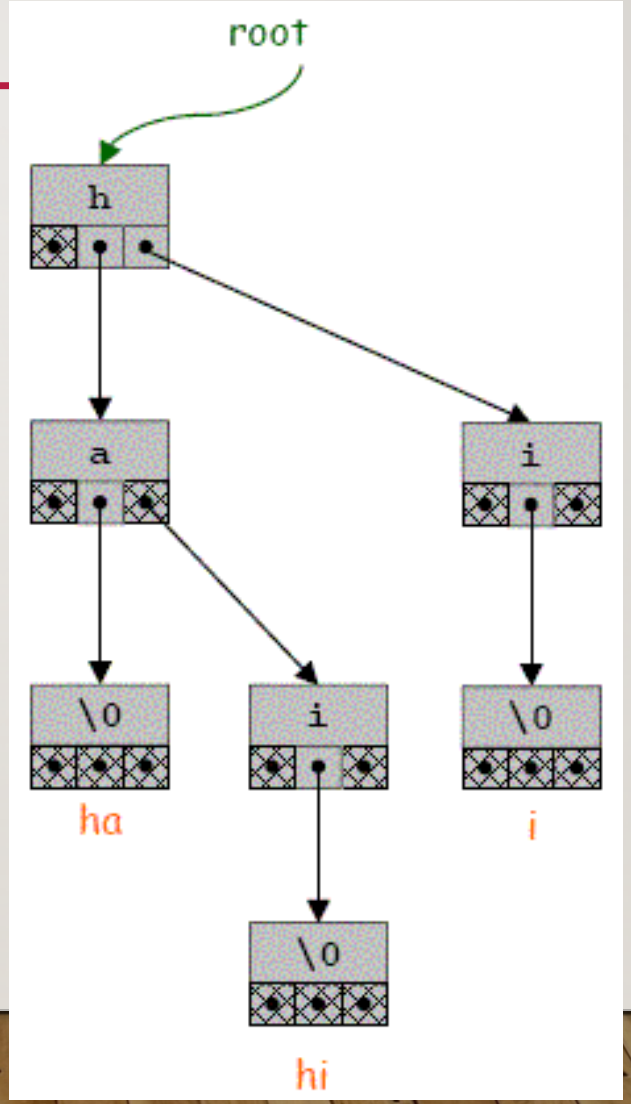
5. TIPOS DE TRIES

TST- Ternary Search Tree

Cada nó aloca três ponteiros

Centro: caractere seguinte
Filhos da esquerda e direita
: caracteres alternativos

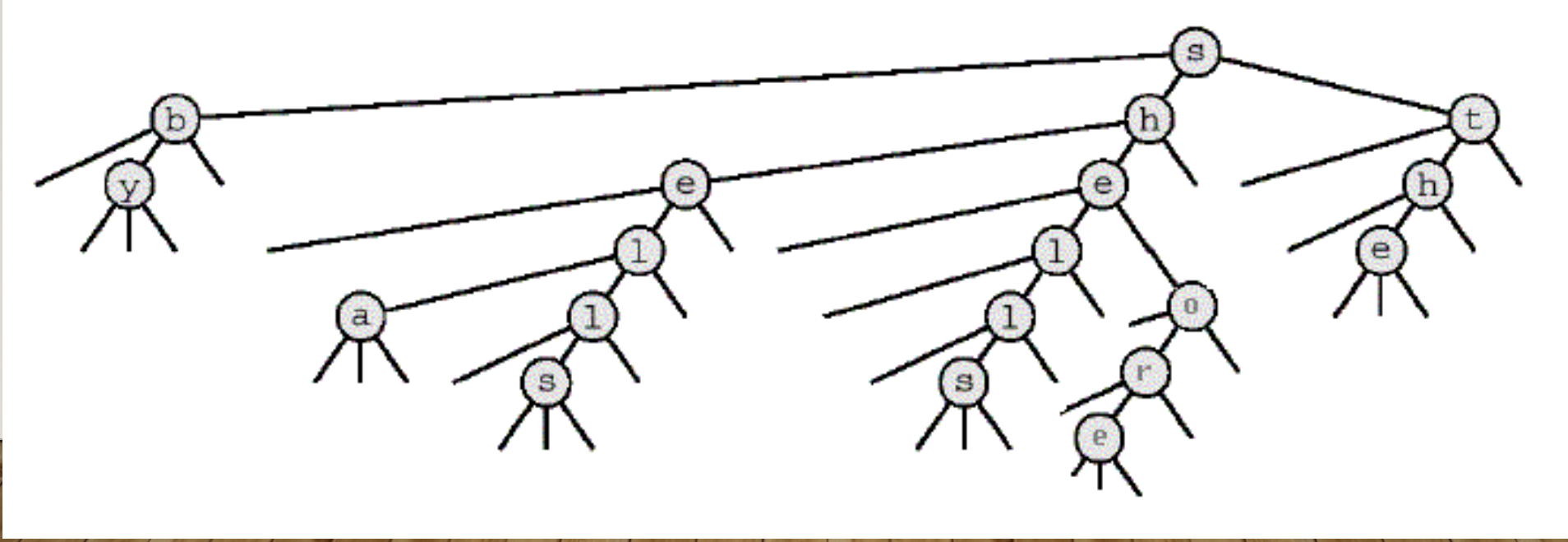
Tem desempenho melhor no que se refere a espaço.



5. TIPOS DE TRIES

TST- Ternary Search Tree

Chaves: by, sea, sells, shells, shore, the



6. VANTAGENS E DESVANTAGENS

Vantagens

Nas R-Way a busca é ramdômica em cada nível/caractere, aumentando a velocidade do acesso

Não precisa aplicar funções para converter chaves alfabéticas em numéricas

Desvantagens

Nas R-Way, nodos próximos às folhas terão subTRIES como NIL, desperdiçando espaço.

7. APLICAÇÕES DAS TRIES

Dicionários (telefone celular)

Corretores Ortográficos

Programas para compreender Linguagem Natural

Auto-preenchimento:

browsers,

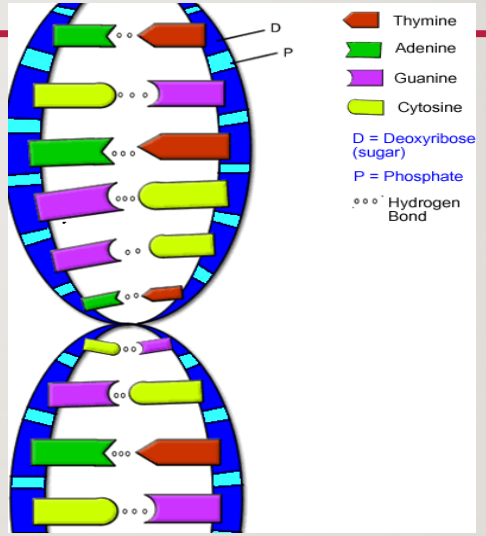
e-mail,

linguagens de programação

7. APLICAÇÕES DAS TRIES

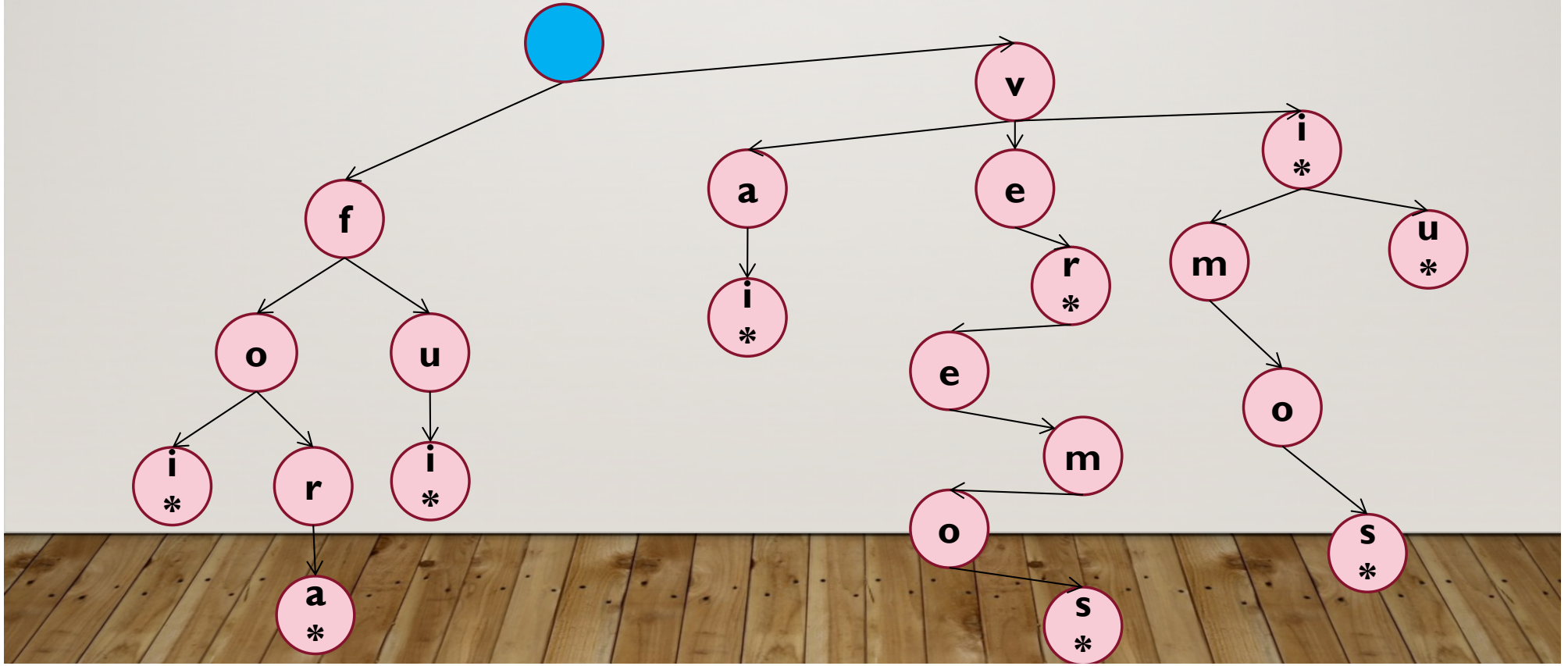
*Compressão de dados

*Biologia computacional



* Tabelas de roteamento para endereços IP
* Armazenar e consultar documentos XML
* Fundamental para o Burstsort (o método mais rápido de ordenação de strings em memória/cache)
* Tabelas de símbolos em compiladores

- Insira as chaves no is th ti fo al go pe to co to th ai of th pa numa trie inicialmente vazia. Faça um desenho da trie resultante. (Não desenhe os links de valor null.)
- Insira as chaves now is the time for all good people to come to the aid of numa trie inicialmente vazia. Faça um desenho da trie resultante. (Não desenhe os links de valor null.)
- Quais chaves/palavras estão representadas na trie abaixo?
- Insira as chaves vaidade, foice, viúva, avião, vermelho e Exclua as chaves vai, fui, vimos



8. REFERÊNCIAS

- http://www.cs.mcgill.ca/~cs251/OldCourses/1997/topic7/
- http://everything2.com/e2node/Trie
- http://www.infotem.hpg.ig.com.br/tem_progr_arvores.htm
- http://www.csse.monash.edu.au/~lloyd/tildeAlgDS/Tree/Trie/
- http://www.gpec.ucdb.br/pistori/disciplinas/ed/aulas_II/bd.htm
- http://www.cs.bu.edu/teaching/c/tree/trie/
- http://www.cs.bu.edu/teaching/c/tree/trie/
- http://www.cs.princeton.edu/courses/archive/spr04/cos226/lectures/trie.4up.pdf