Search

Radix search trie (RST)

R-way trie (RT)

De la Briandias trie (DLB)

Binary search tree (BST)

- Left branch is less than
- Right branch is larger than

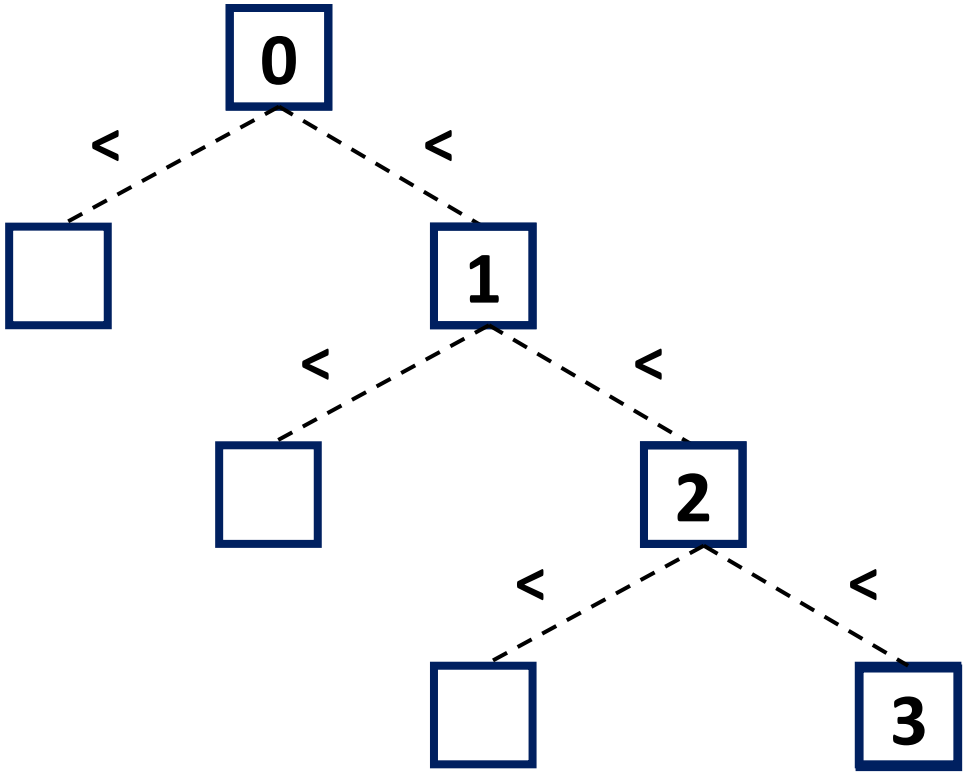
Create BST (with 0, 1, 2, 3 in order)

0

1

2

3



Can we do better?

Radix search trie

- Using binary representation of each value, and not the value itself (like in BST)

- Create an RST with 0, 1, 2, 3 (in order)

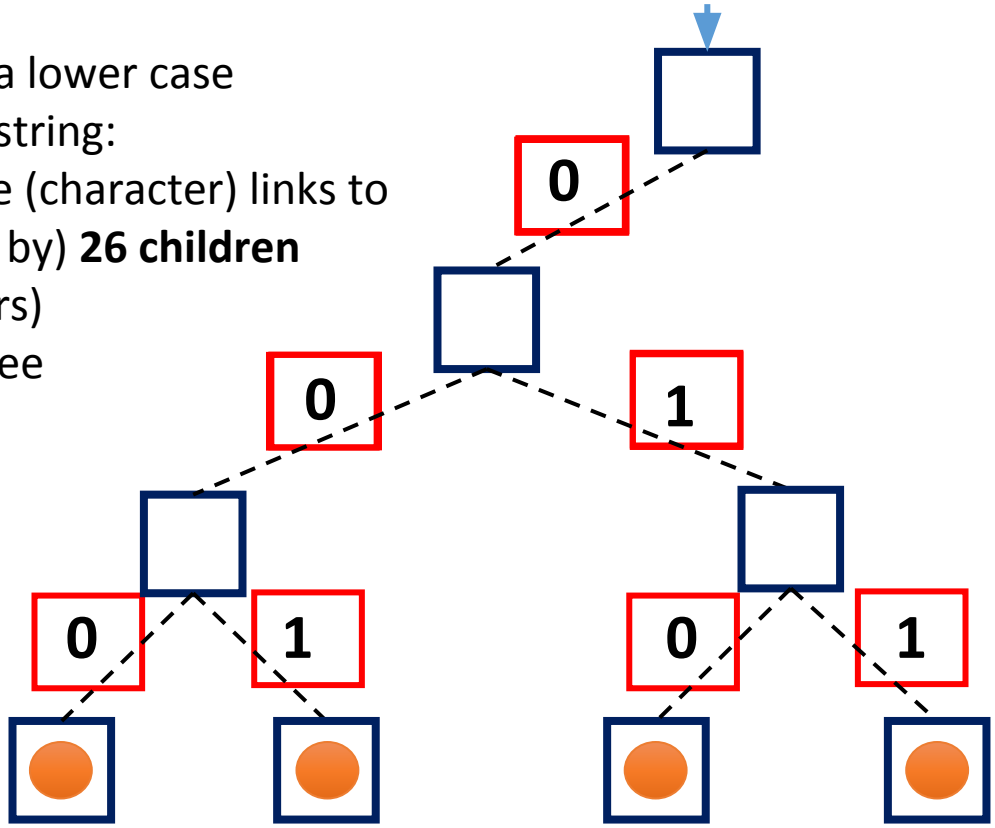
Create RST

Key	Value
000	0
001	1
010	2
011	3

Binary representation

Consider a lower case alphabet string:
each node (character) links to (followed by) **26 children** (characters)
26-way tree

"binary" or 2-way tree
each node links to 2 children



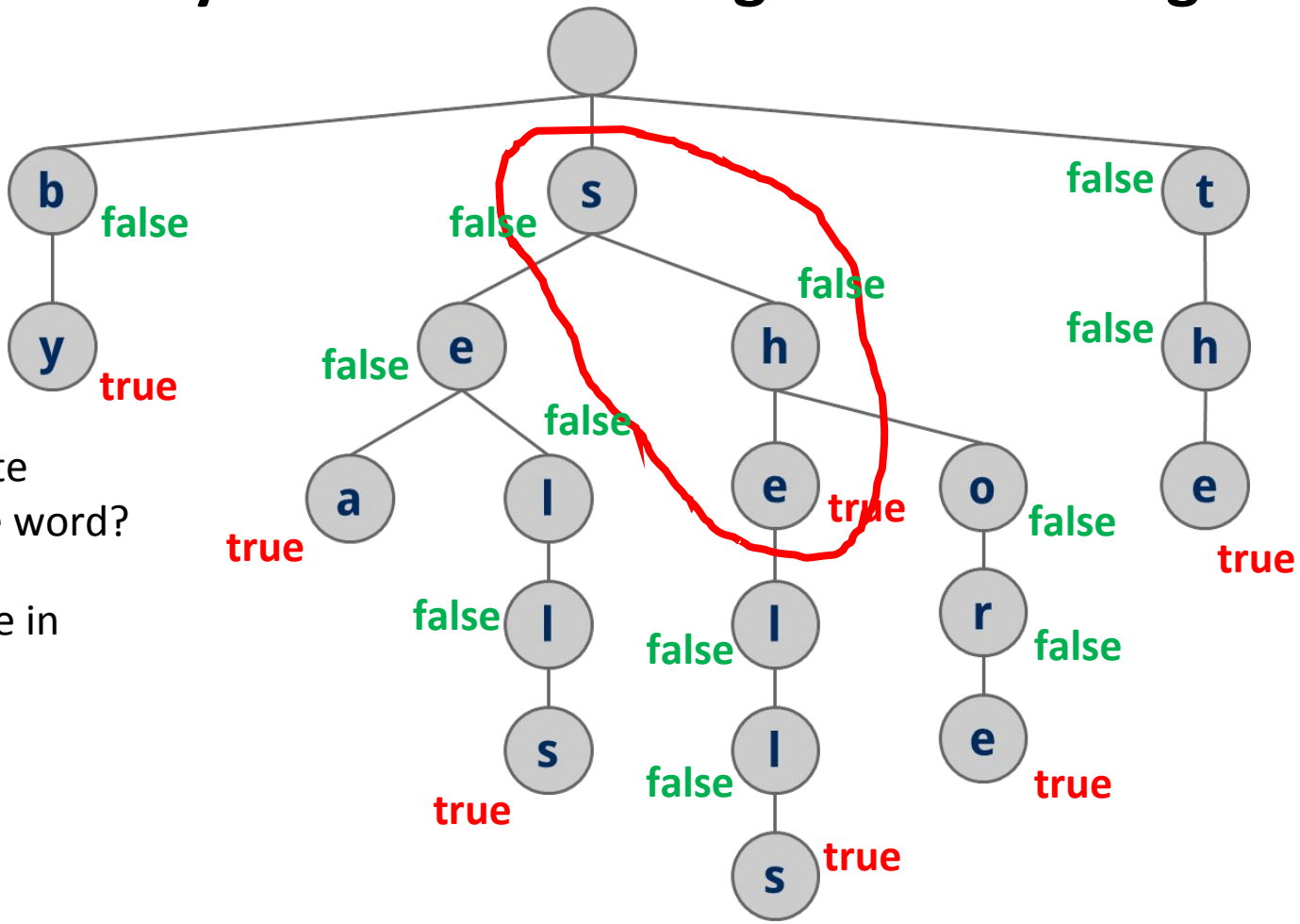
Worst case is bound by the binary representation length, not by the number of elements as in the BST

Can it be applied to strings?

R-way trie (lecture example)

she, sells, sea, shells, by, the, sea, shore

Worst case bounds by the character length of the string



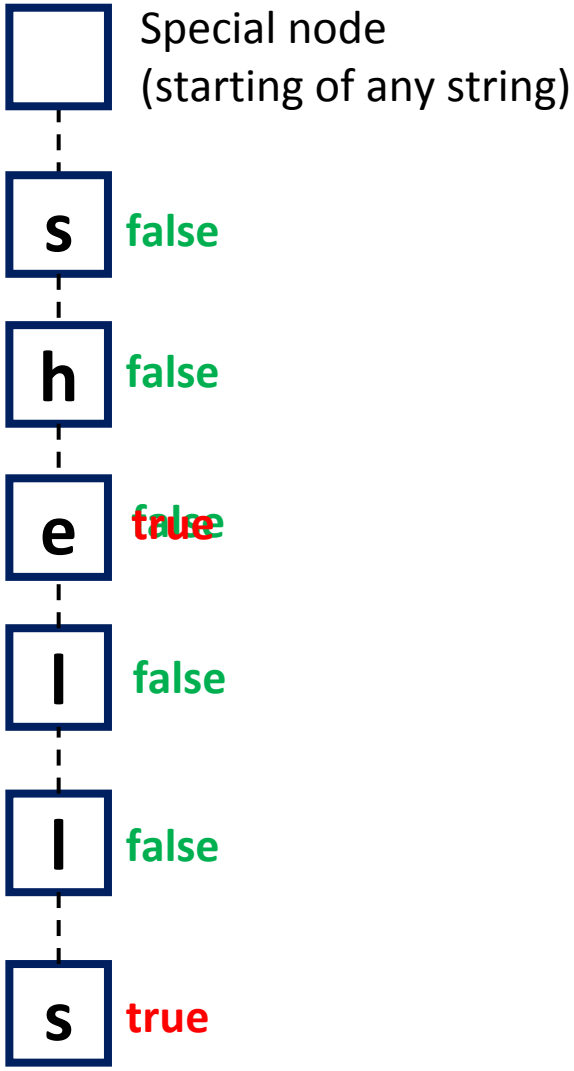
How can we indicate "she" is a complete word?

Using a flag variable in each node?

Create R-way trie

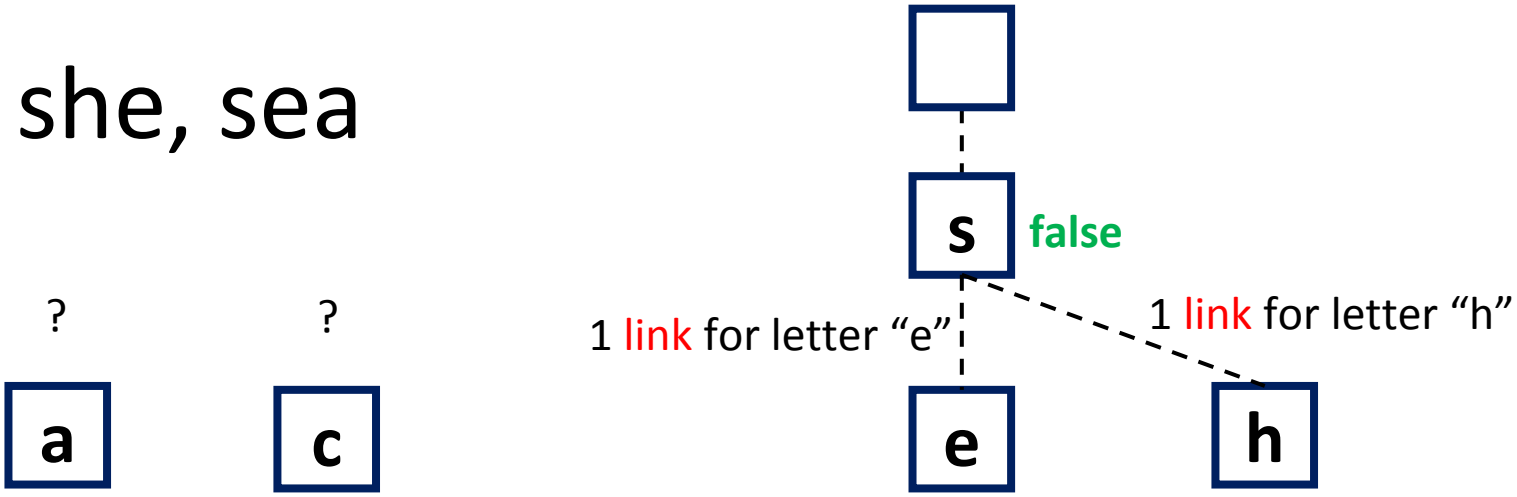
- shells, she

s h e l l s



The ugly truth

she, sea



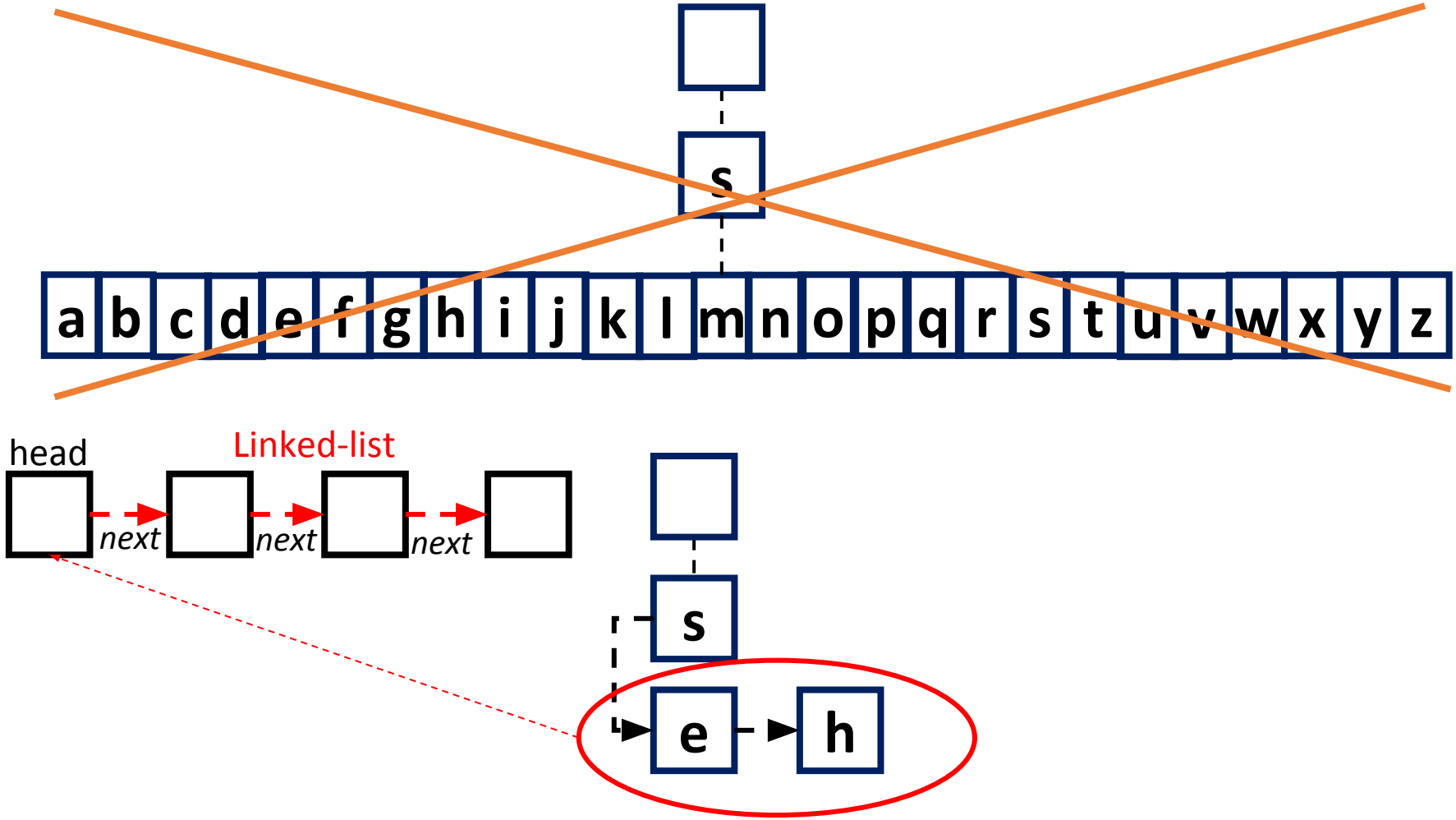
1 node = 26 links + 1 flag variable

The same prefix? Impossible combinations?

Can we do it better?

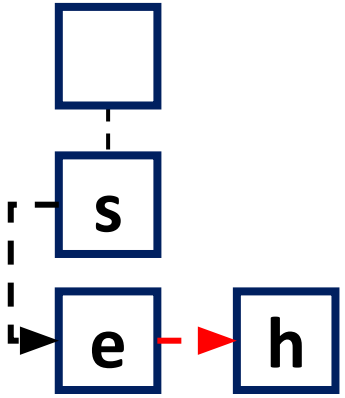
De la Briandais (DLB)

- Replace the fixed link array by a flexible linked list



Trade off

- Save a lot of space, especially when the real case has sparse strings
- Increase searching time. Why?
  - R-way trie: Directly go to a child in the **array**
  - DLB: linearly go the child in the linked list

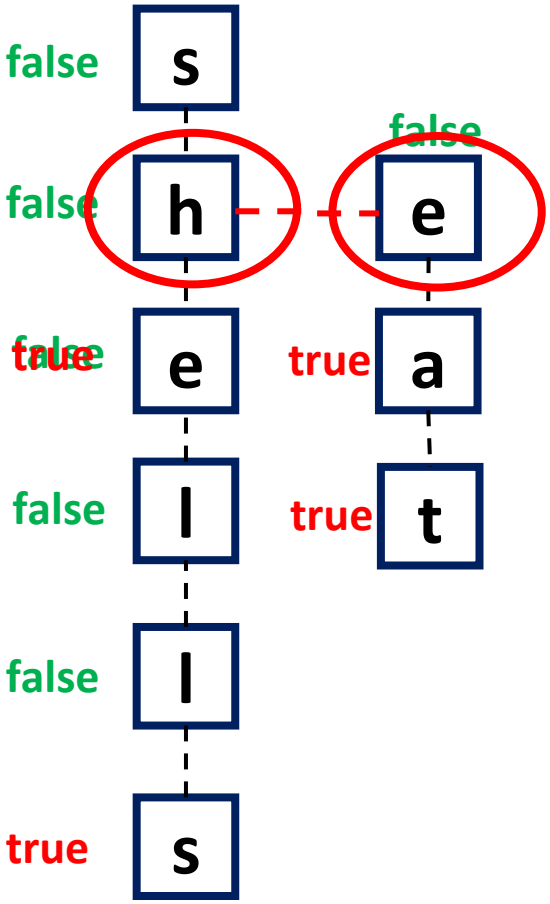


Create a DLB

- shells, she, sea, seat

s h e l l s

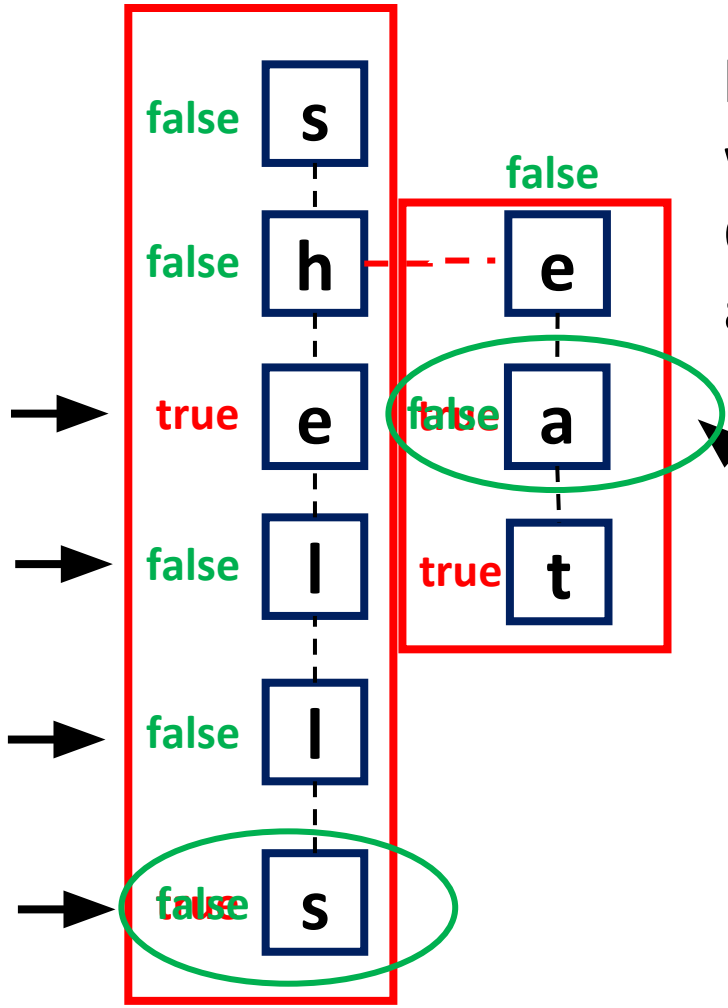
s e a t



Delete a word in DLB

shells

sea



If the path does not belong to other words, remove. Otherwise, leave it alone.

A true node
stop

False, no children
delete

False, no children
delete

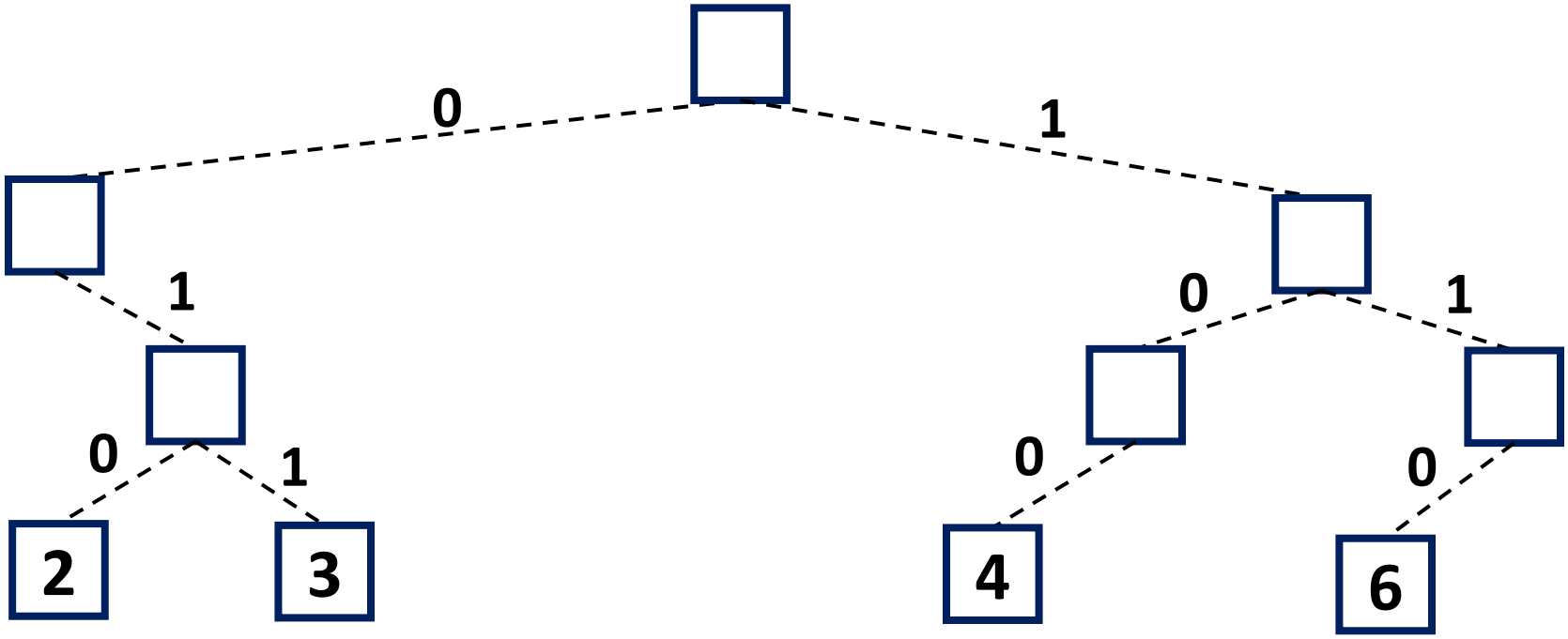
Change to false
no children: *delete*

Change to false, has a child
stop

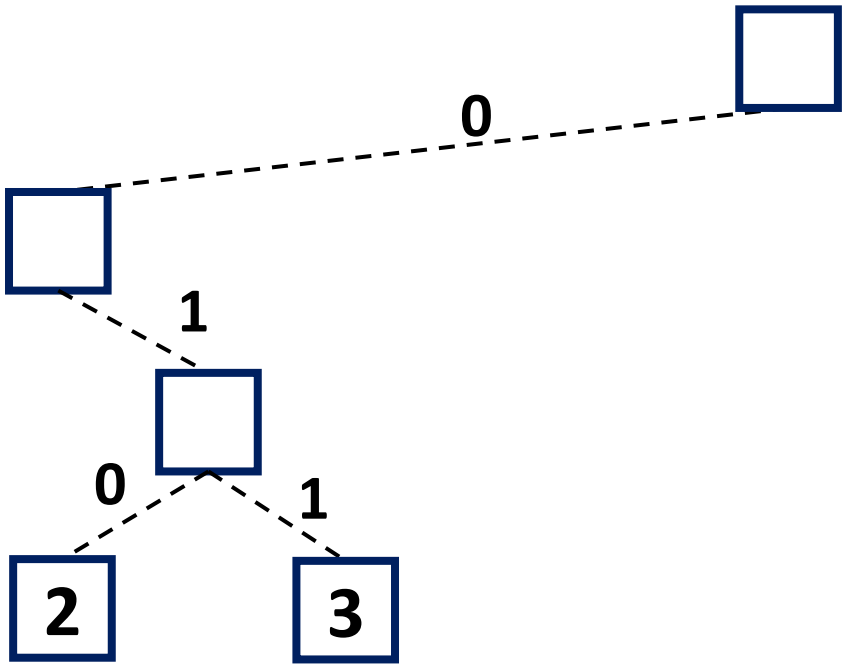
Exercises (on paper)

- RST:
  - Create a tree for: 2, 3, 4, 6
  - Delete values: 4, 6
- DLB:
  - Create a tree for: baby, bad, bank, box, dad, dance
  - Delete words: bad, bank, dance

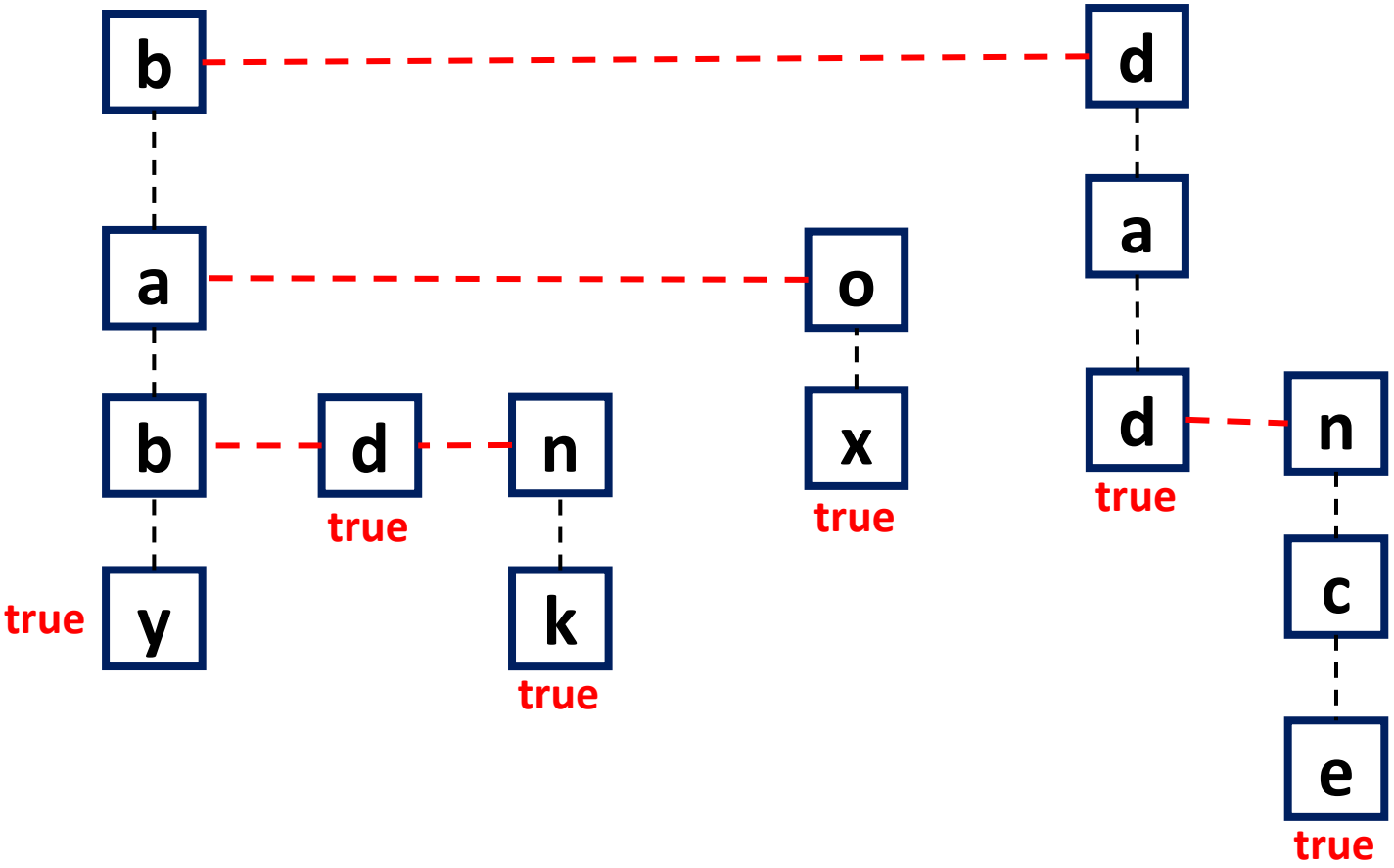
Exercises RST (creation)



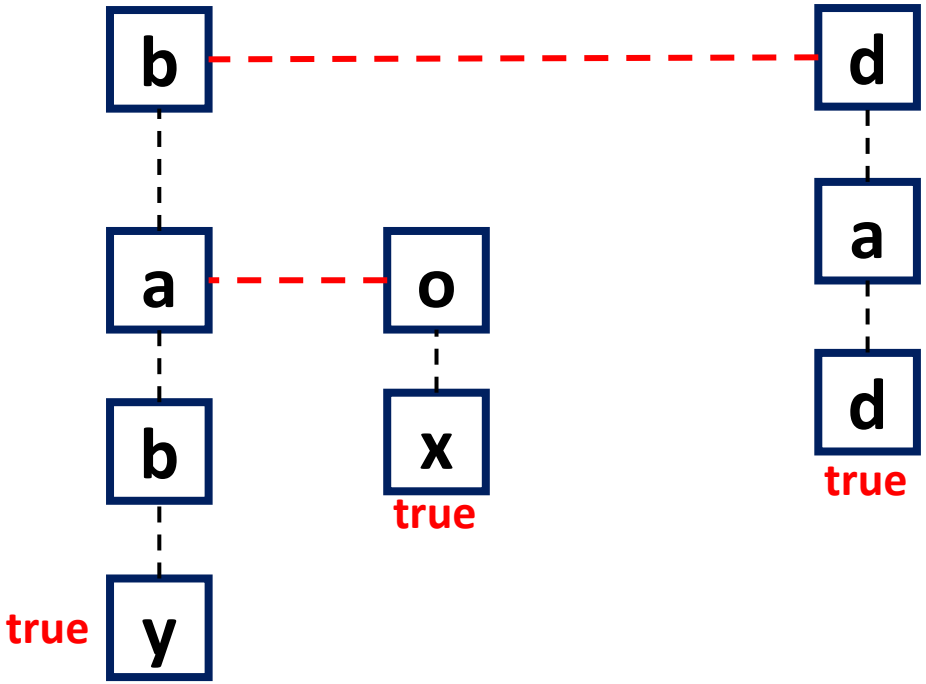
Exercises RST (deletion)



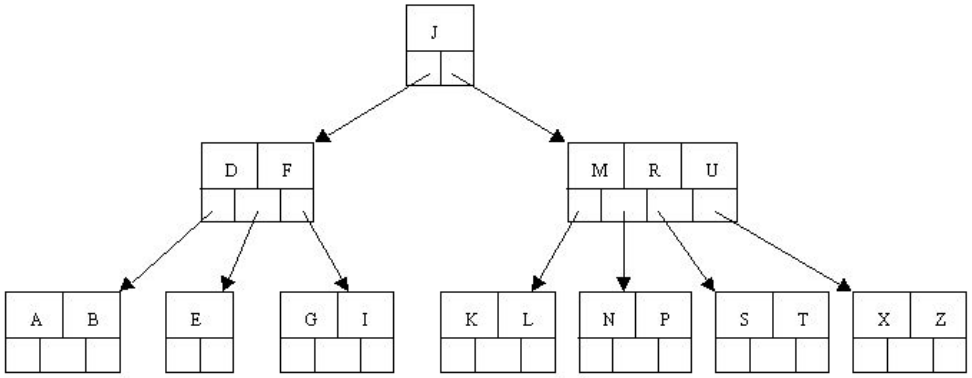
Exercise DLB (creation)



Exercise DLB (deletion)



B-Tree



- **Self-balancing** tree data structure that keeps data **sorted** and allows searches, sequential access, insertions, and deletions in **logarithmic time**.
- Each node can have **more** than two children
- **Max Degree** defines max number of children each node can have

B-Tree Insertion

1. Find the node to insert (starting from the root)
2. If there is room in the node, just insert in that node
3. Otherwise the node is full, evenly split it into two nodes so:
    1. A single median is chosen from among the leaf's elements and the new element.
    2. Values less than the median are put in the new left node and values greater than the median are put in the new right node, with the median acting as a separation value.
    3. The separation value is inserted in the node's parent, which may cause it to be split, and so on. If the node has no parent (i.e., the node was the root), create a new root above this node (increasing the height of the tree).

B-Tree Deletion

1. **Find and delete** the value
    i. If the value is not in a leaf node, you need to find a replacement…
2. **Rebalance** the tree
    i. Is there a sibling node with more than minimum keys?
        1. If so rotate right/left accordingly
    ii. If not, need to merge with the left or right sibling

B-Tree Example

https://www.cs.usfca.edu/~galles/visualization/BTree.html

Inserts:

77 33 11 99 88 44 22 66 15 50 60 63 64 100

Deletions:

99 77 50 33 44 64 66