Trie data structures

08 November 2020 11:54

- Keys of Varying length
- Based on multilevel branching and searching is based on a portion of key and not on the whole.
- Also called as Lexicographic search trees.
- Trie originated from the word "retrieval"

Definition:

A trie of order 'm' can be empty. If non empty, then it consists of an ordered sequence of exactly 'm' tries of order 'm'. The branching at any level is determined by a portion and not by the whole key.

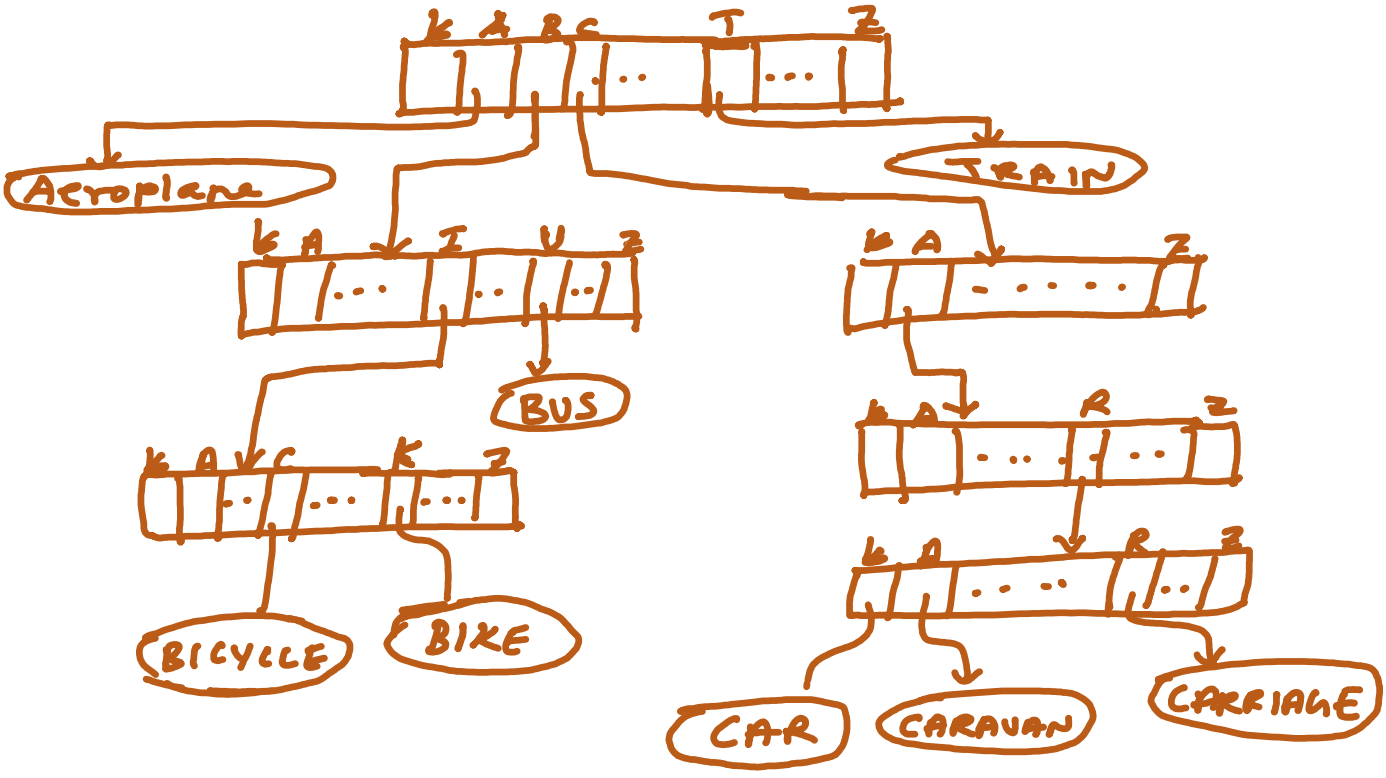
- Tries have two types of node structures.

  Branch node: Collection of links pointing to branches or information node.

  Information node: Holds the key, that is to be stored in the trie.

Example:

Trie of alphabetical keys.



Searching

- To Search a key 'K' in a tree 'T', we begin at the root node which is a branch node.
- Key $K \longrightarrow k_1 k_2 \cdots k_n$ (seq. of characters)

→ First character 'k_1' is extracted from key and the link field of 'k_1' is identified.

→ If $Link(T_1, k_1) = nil$, search is unsuccessful

else

$Link(T_1, k_1)$ points to next branch node or information node

- If the information node is equal to 'K' then Search is successful

else

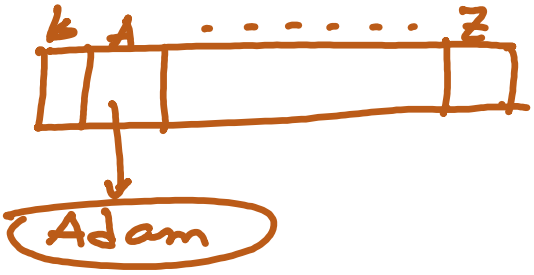
extract k_2 and repeat the

Process until Key is found or unsuccessful Search.

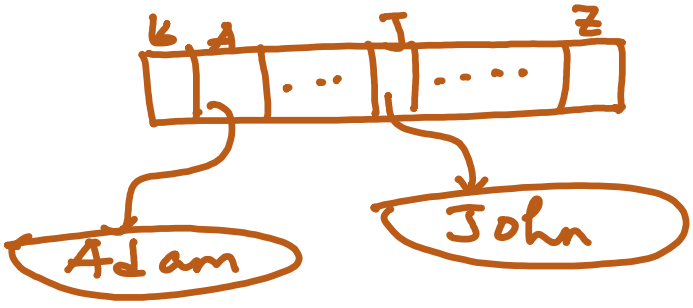
Insertion : Example

Create a trie by inserting Adam, John, Johnson, Joseph.

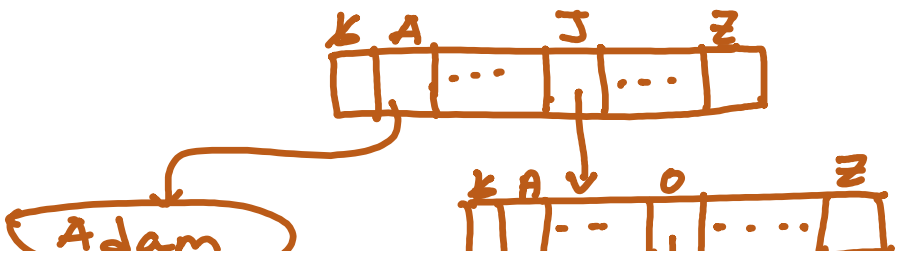
1) Insert : Adam

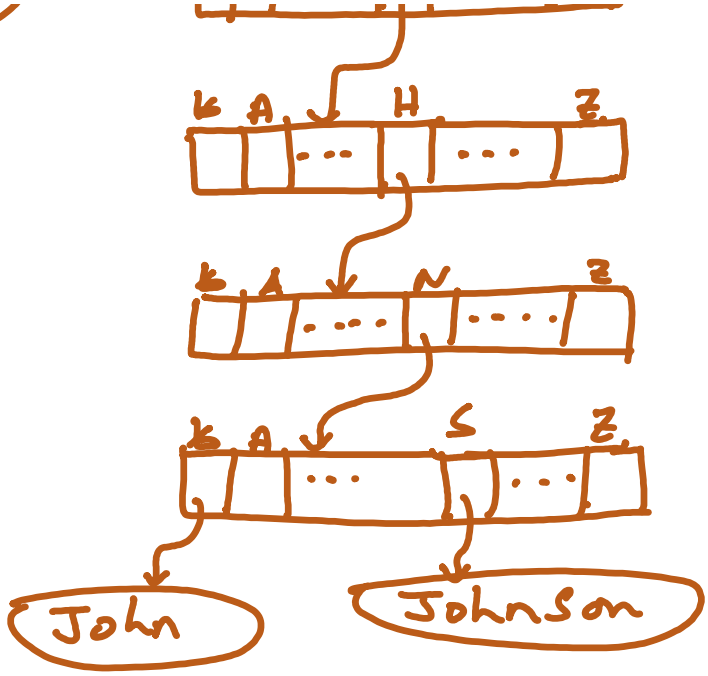


2) Insert : John

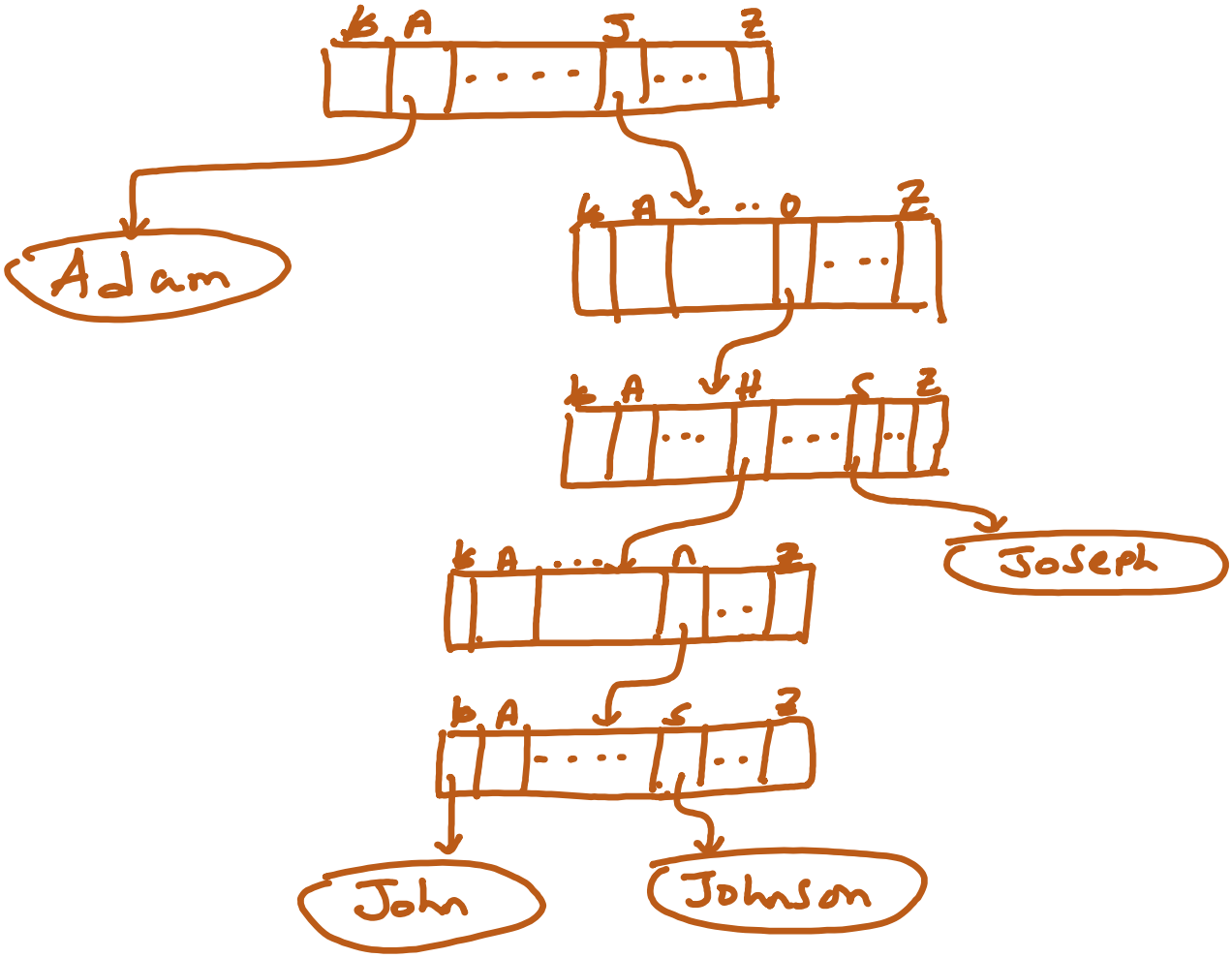


3) Insert : Johnson





(iv) Joseph



Deletion:

For deleting a key 'k', search the information node I holding k, then delete it. Check whether the branch node to which node I is linked accomodates other information nodes. If there are more than one information node links to the concerned branch or if there is atleast on link field to another branch node or both then the deletion is done by deleting the information node holding the key. Suppose after deletion of node I, if the branch node contains one key, then we delete the branch node and push node I to higher level.

Example:

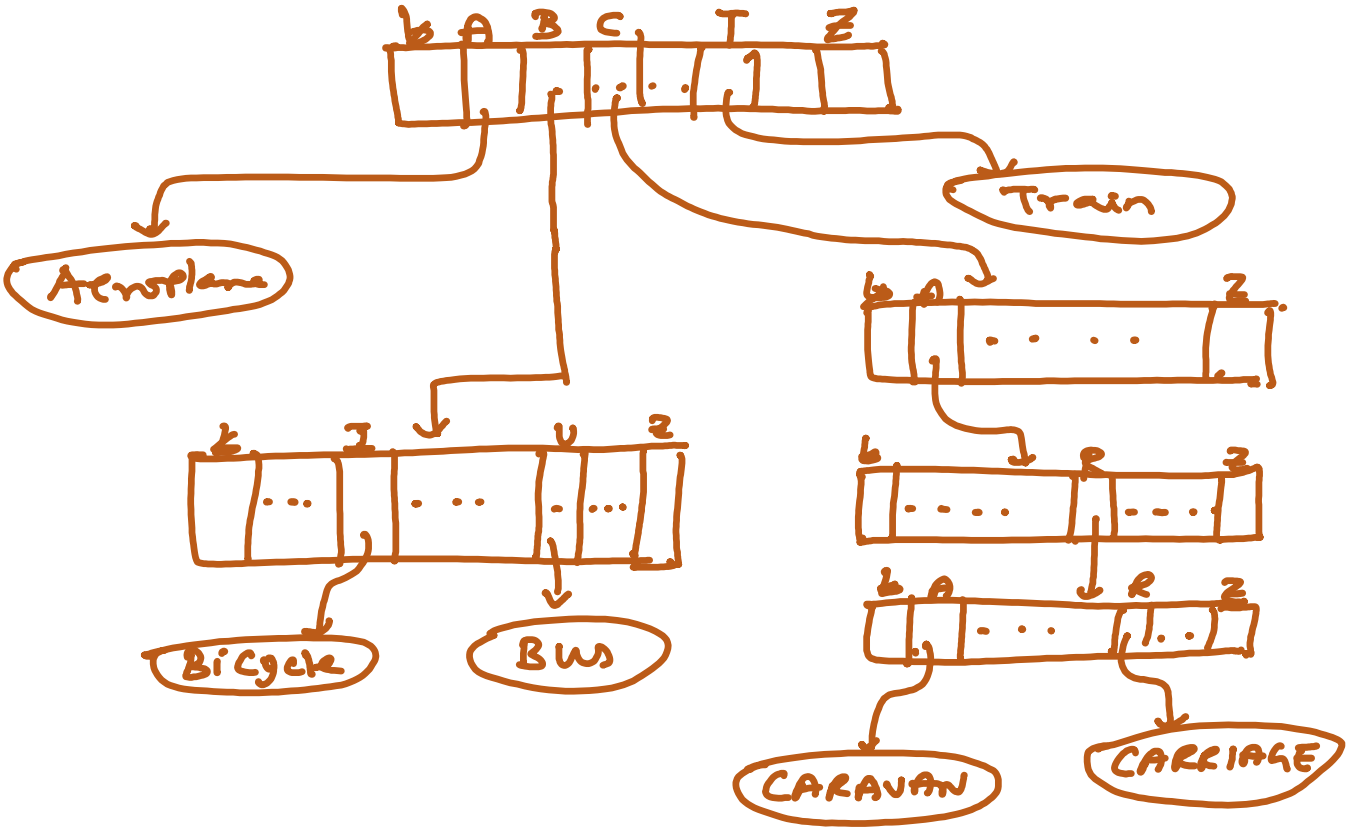
Delete car, bike in first example.

Delete car:

Simply delete, more than one information node is available.

Delete Bike

Deletion of Bike. Left with one information node. Hence delete that branch node. Move up and accomodate the key. Result in,



Performance

Performance of trie is determined by the length of the key. For example if the length of the key of a trie is equal to 7, then the trie can represent $(26)^7$ keys.

For the maximum length of Uniform Prefixes within the keys being 6 can retrieve keys in atmost 7 comparisons.

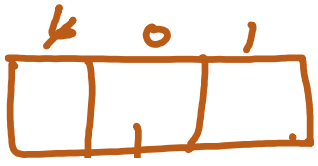
In case of BST, we need, $\log_2(26)^7 \simeq 33$ Comparisons.

Example:

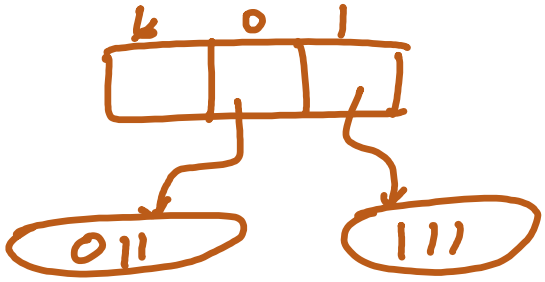
Construct a binary trie by inserting 011, 111, 101, 001.

Ans

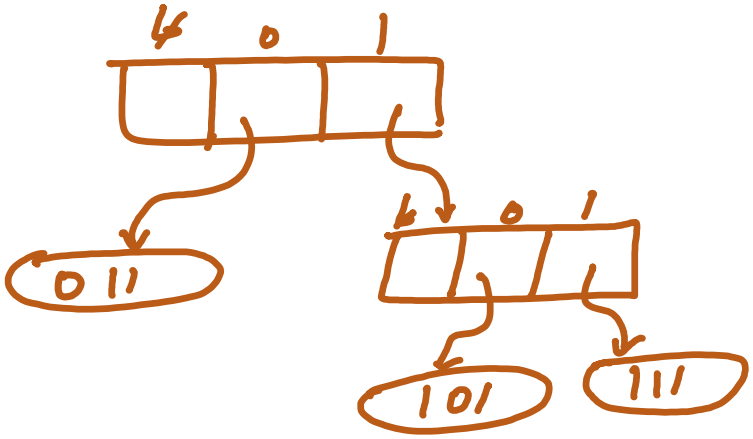
Insert 011



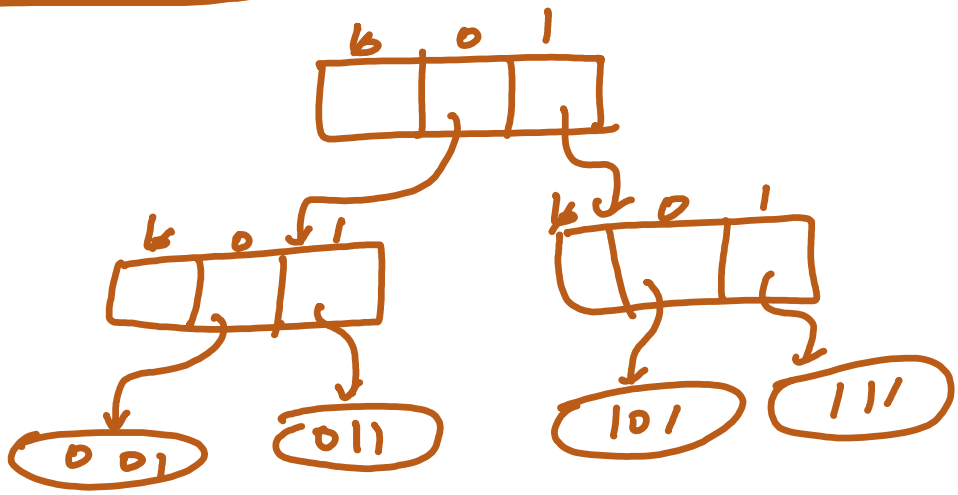
Insert 111



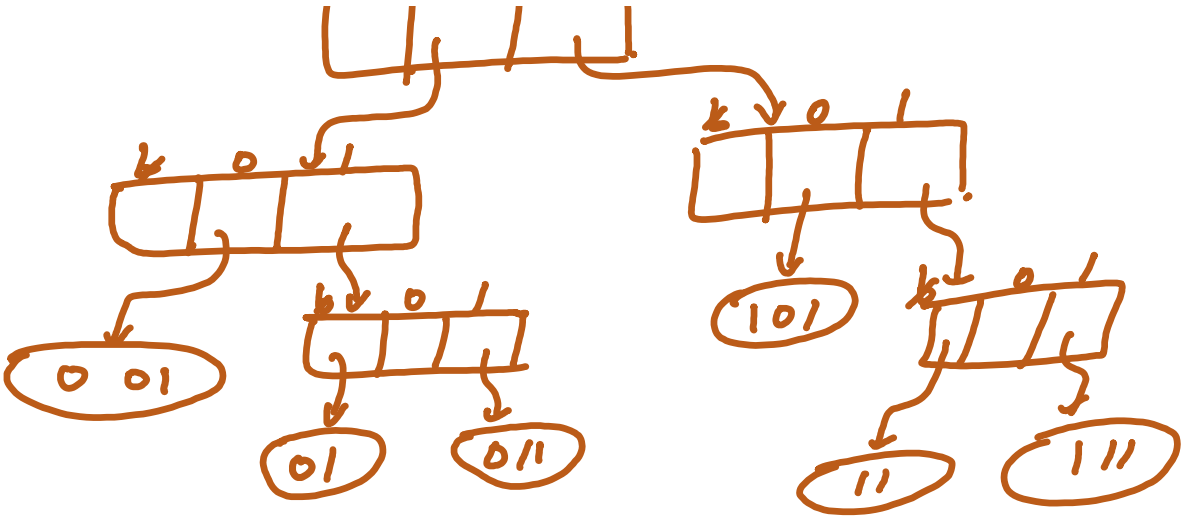
Insert 101



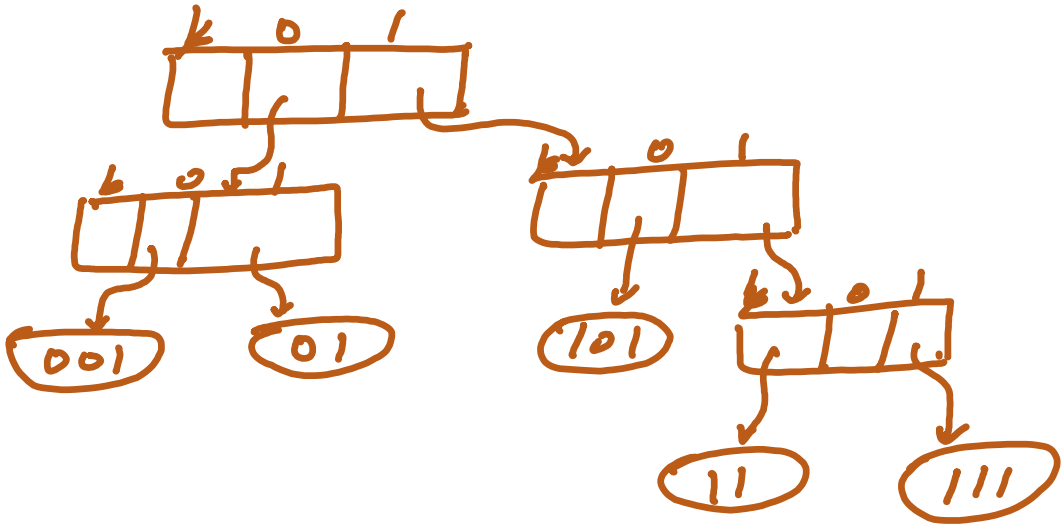
Insert 001



Insert 01, 11 in above



Delete 011



Delete 001