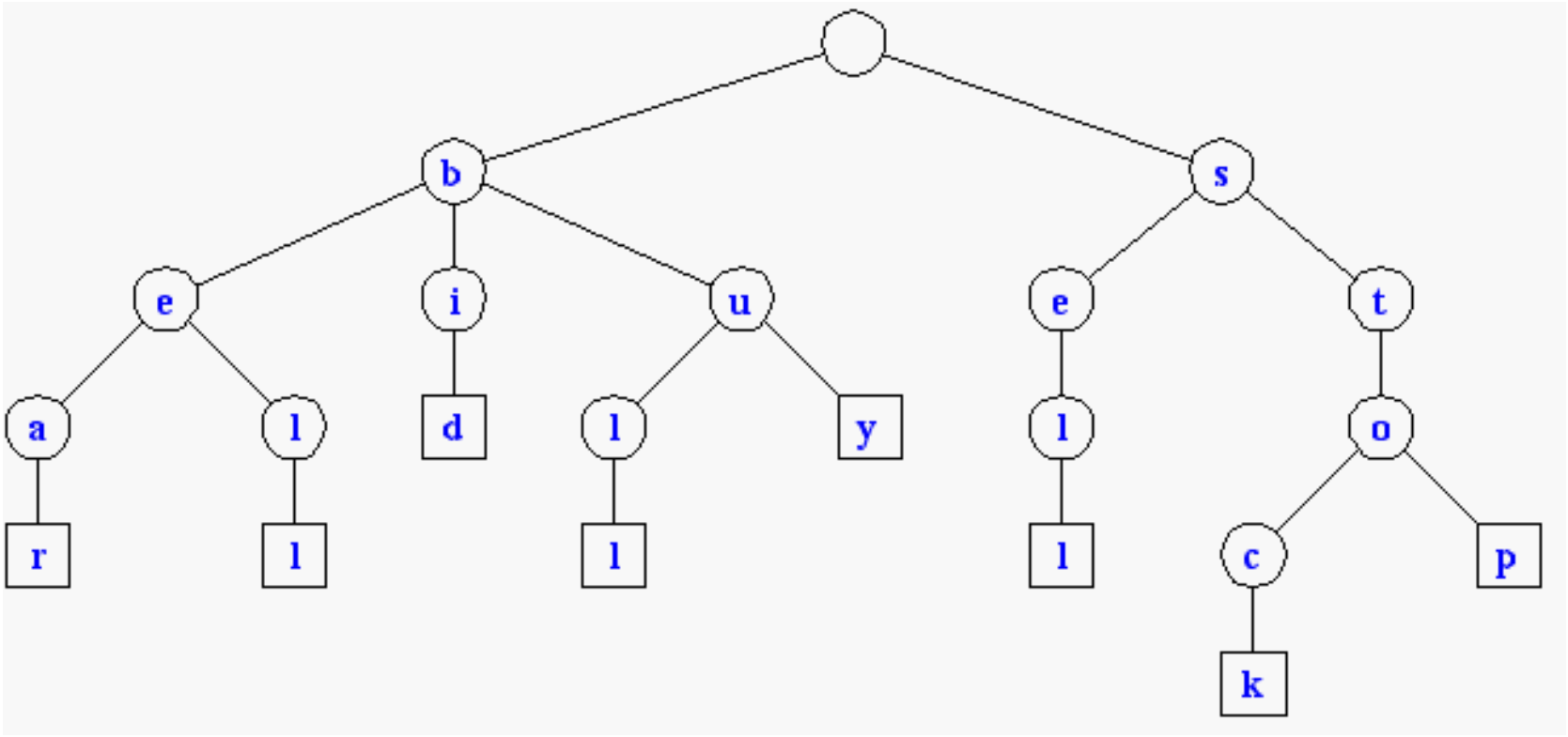
# Advanced Data Structure and Algorithm

Tries and Suffix Trees

# Tries

- Standard Tries
- Compressed Tries
- Suffix Tries



# Text Processing

- Preprocessing the pattern speeds up pattern matching queries
- After preprocessing the pattern in time proportional to the pattern length, the Boyer-Moore algorithm searches an arbitrary English text in (average) time **proportional to the text length**
- If the text is large, immutable and searched for often (e.g., works by Shakespeare), we may want to preprocess the text instead of the pattern in order to perform pattern matching queries in time ***proportional to the pattern length***.
- Tradeoffs in text searching

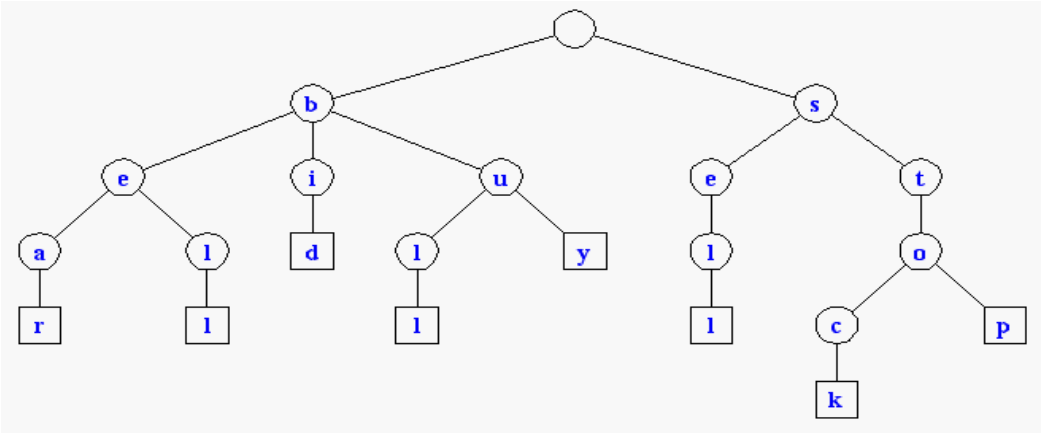
| | Preprocess Pattern | Preprocess Text | Space | Search Time |
|---|---|---|---|---|
| **Brute Force** | | | O(1) | O(mn) |
| **Boyer Moore** | O(m+d) | | O(d) | O(n) * |
| **Suffix Trie** | | O(n) | O(n) | O(m) |

n = text size
m = pattern size

* on average

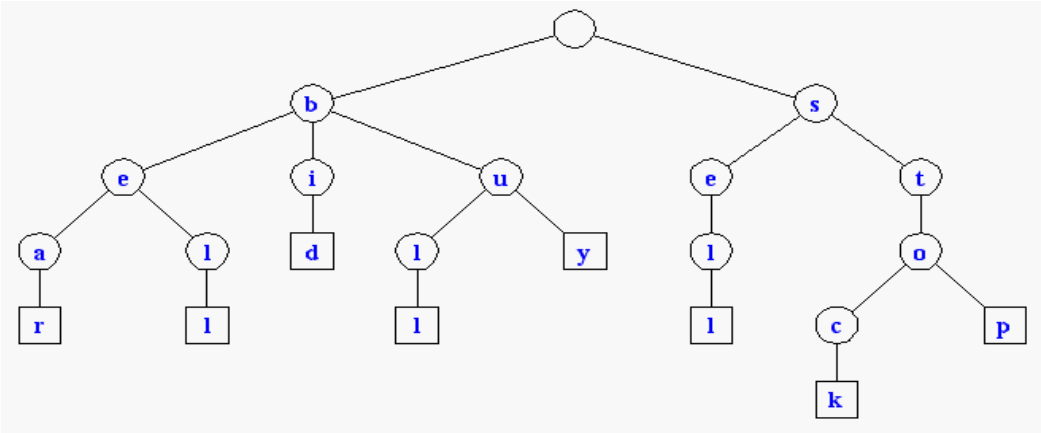
# Standard Tries

- The ***standard trie*** for a set of strings S is an ordered tree such that:
  - each node but the root is labeled with a character
  - the children of a node are alphabetically ordered
  - the paths from the external nodes to the root yield the strings of S
- Example: standard trie for the set of strings
  S = { bear, bell, bid, bull, buy, sell, stock, stop }



# Standard Tries

- The ***standard trie*** for a set of strings S is an ordered tree such that:
  - each node but the root is labeled with a character
  - the children of a node are alphabetically ordered
  - the paths from the external nodes to the root yield the strings of S
- Example: standard trie for the set of strings S = { bear, bell, bid, bull, buy, sell, stock, stop }




•A standard trie uses O(n) space. Operations (find, insert, remove) take time O(dm) each, where:

-n = total size of the strings in S,

-m =size of the string parameter of the operation

-d =alphabet size (i.e., max number of possible child of a node),

# Applications of Tries

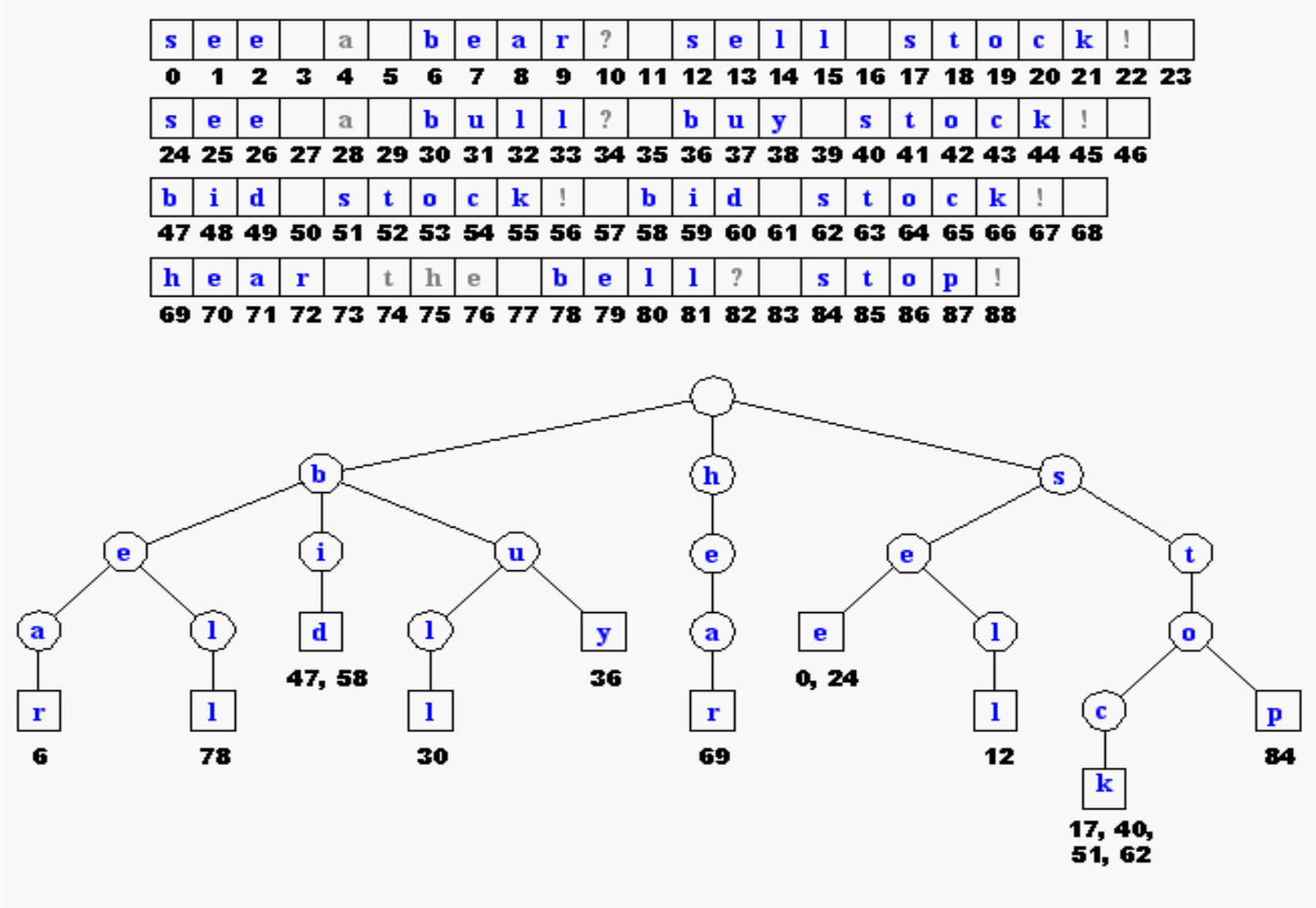
- A standard trie supports the following operations on a preprocessed text in time O(m), where m = |X|

  -***word matching***: find the first occurence of word X in the text

  -***prefix matching***: find the first occurrence of the longest prefix of word X in the text

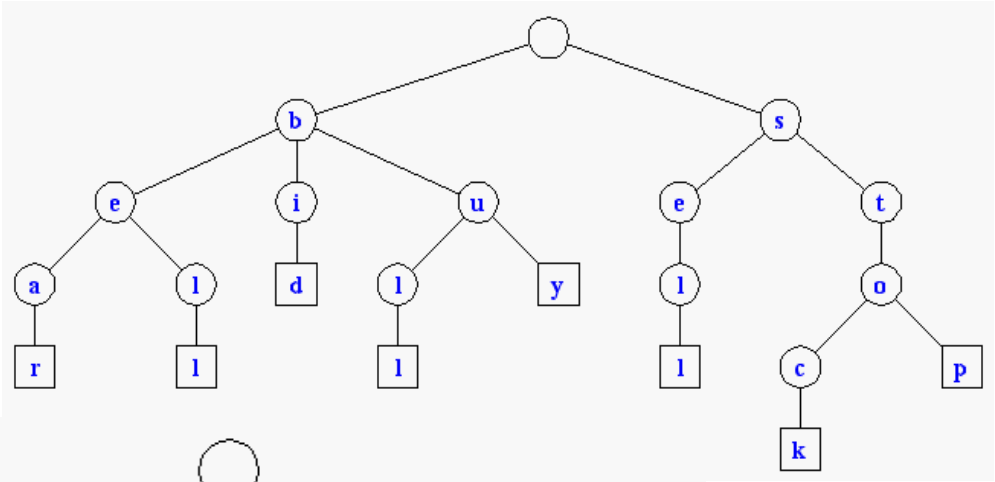
# Applications of Tries

- Each operation is performed by tracing a path in the trie starting at the root

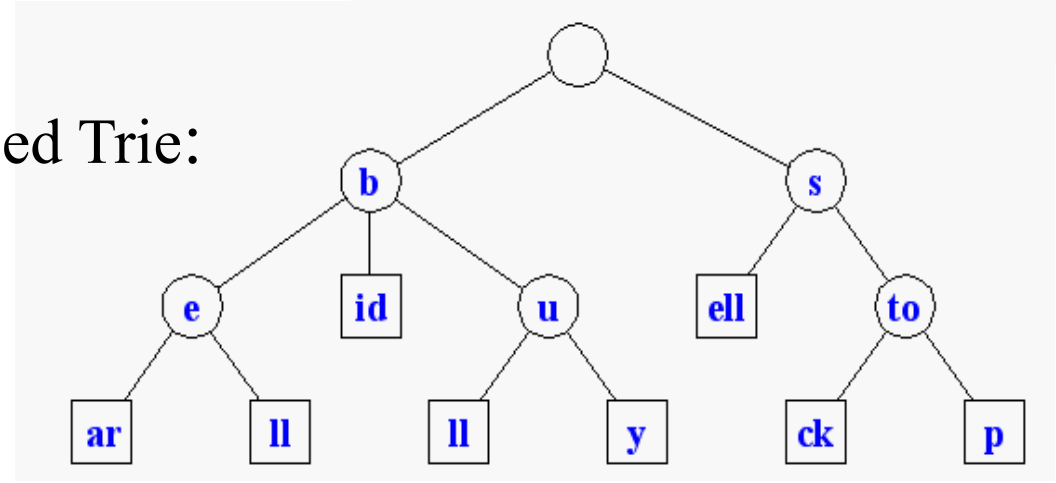


# Compressed Tries

Standard Trie:



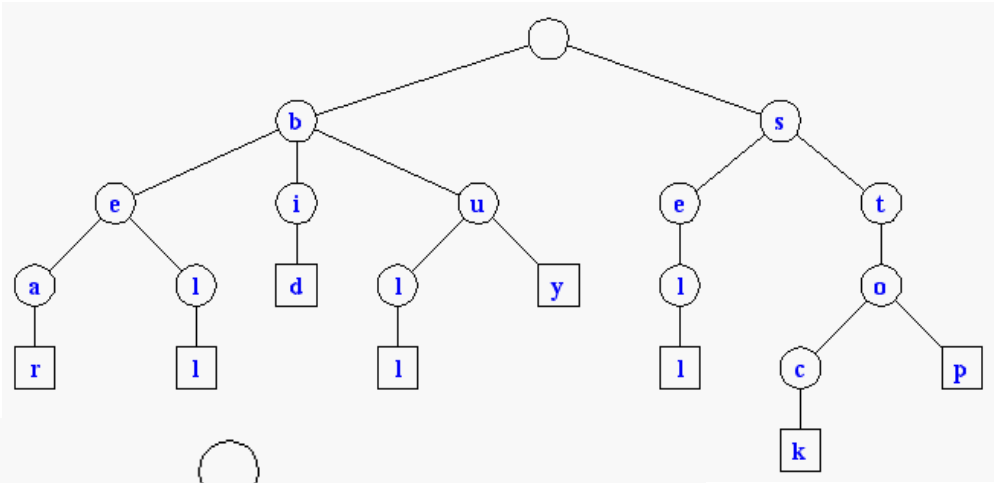

Compressed Trie:



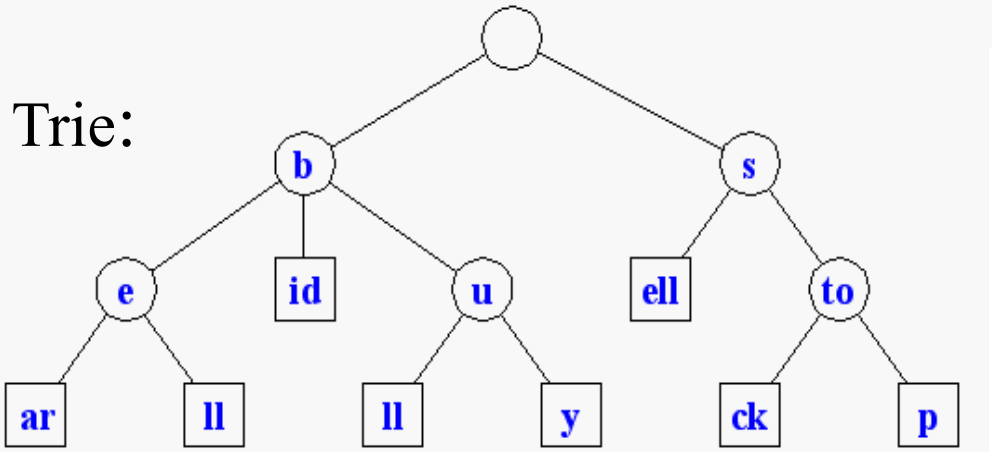
# Compressed Tries

- Trie with nodes of degree at least 2
- Obtained from standard trie by compressing chains of ***redundant nodes***

Standard Trie:

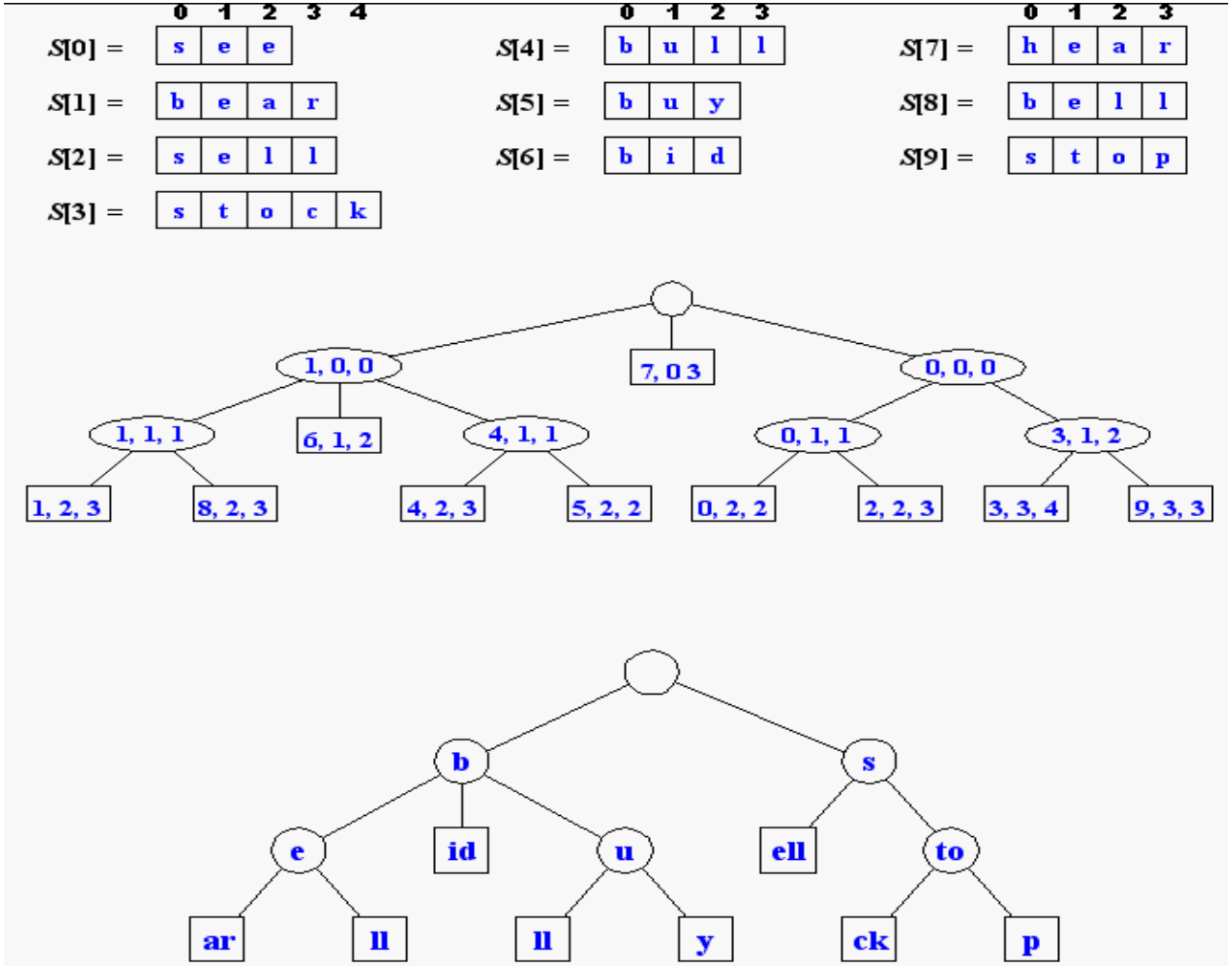



Compressed Trie:

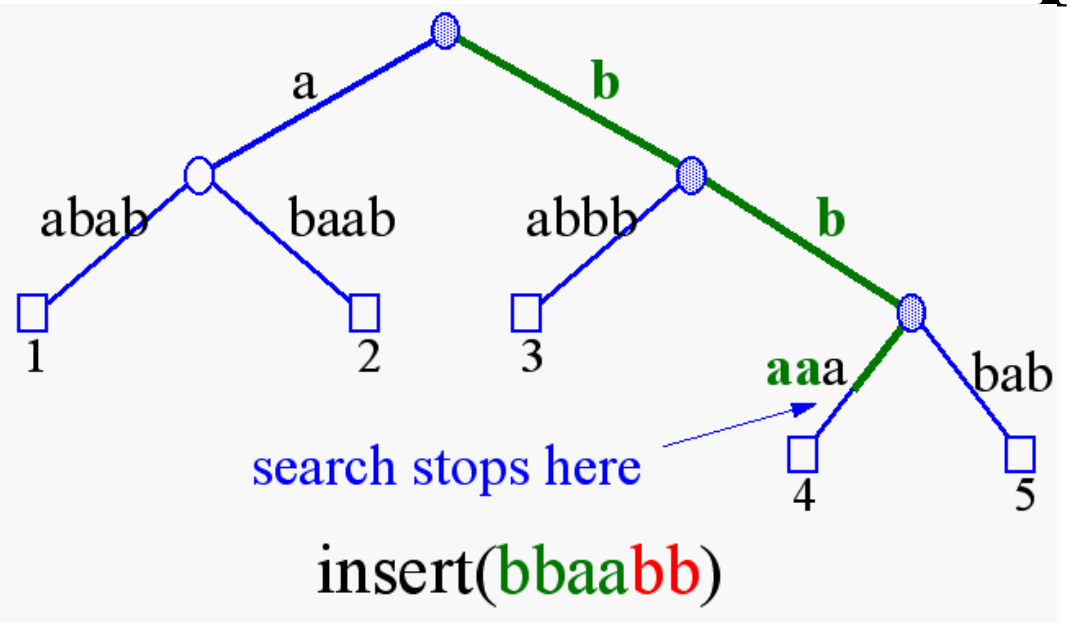


# Compact Storage of Compressed Tries

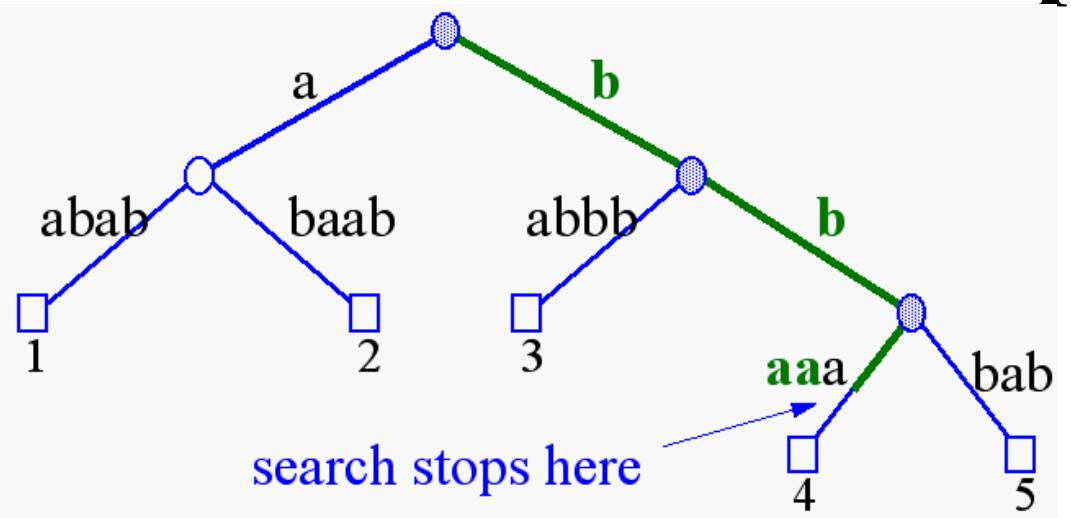
- A compressed trie can be stored in space O(s), where s = |S|, by using O(1) space ***index ranges*** at the nodes



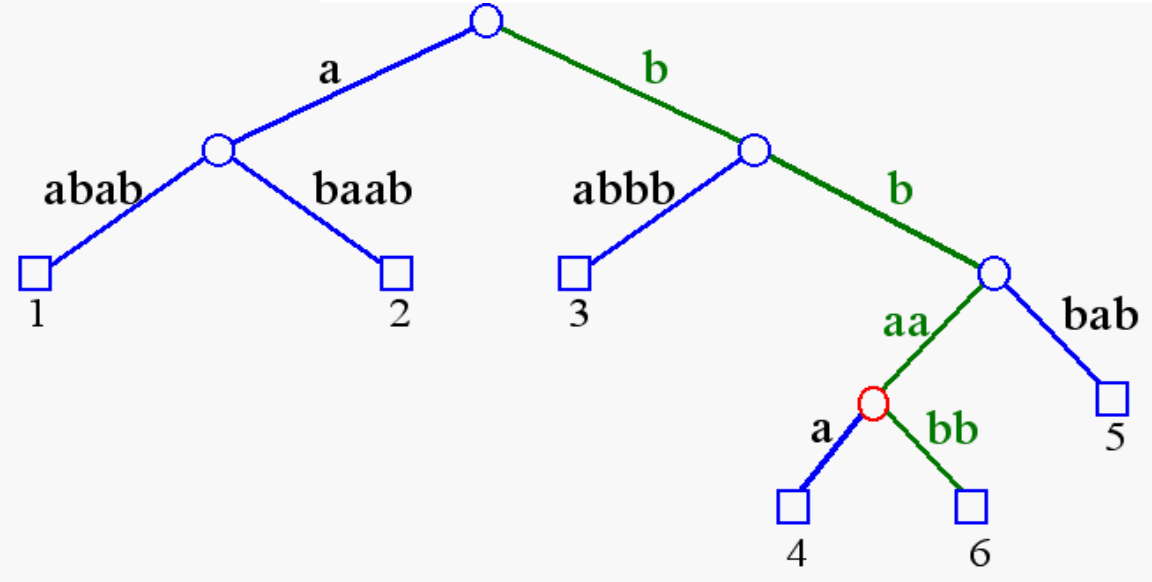
# Insertion and Deletion into/from a Compressed Trie



# Insertion and Deletion into/from a Compressed Trie



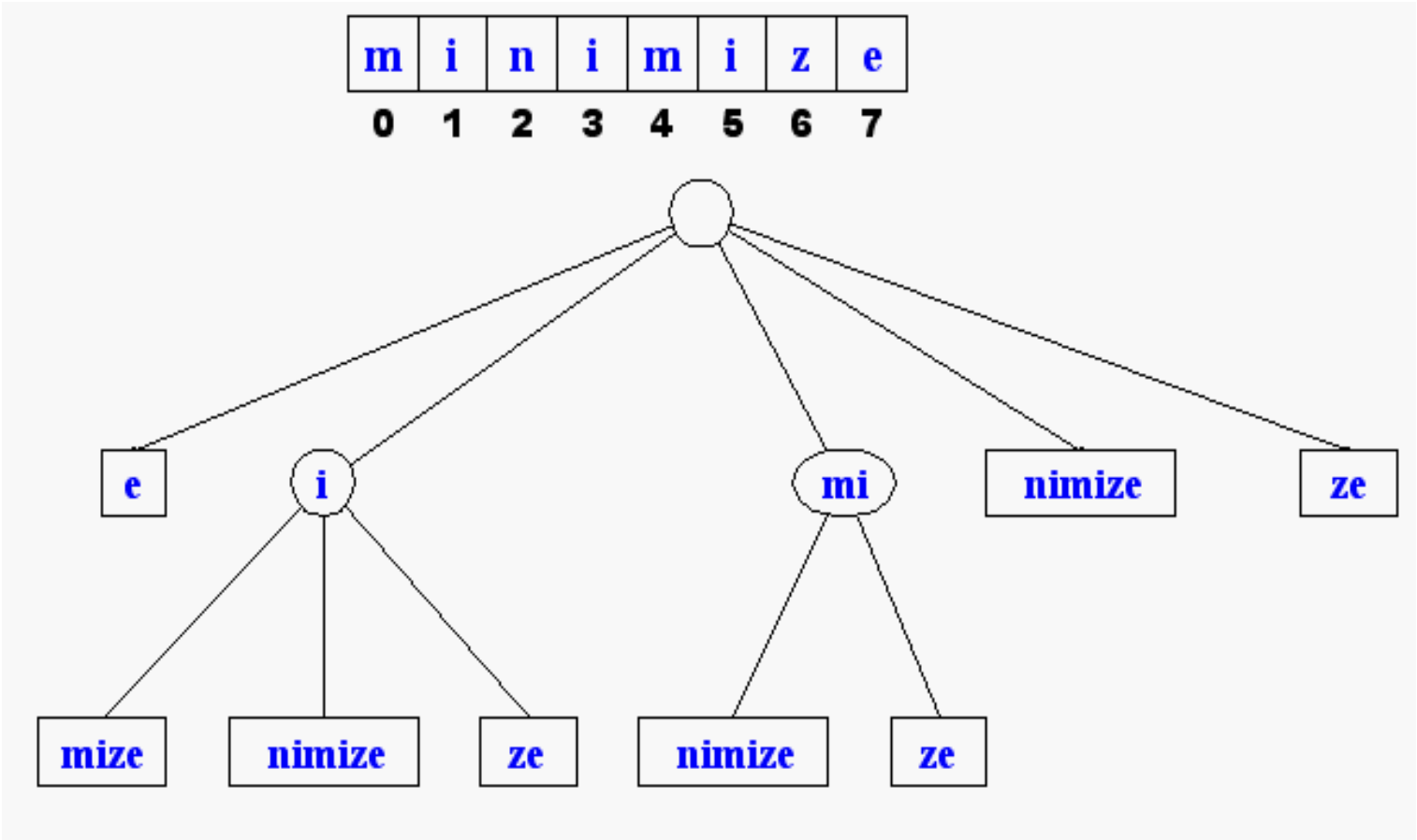

insert(bbaabb)



# Suffix Tries

- A ***suffix trie*** is a compressed trie for all the suffixes of a text

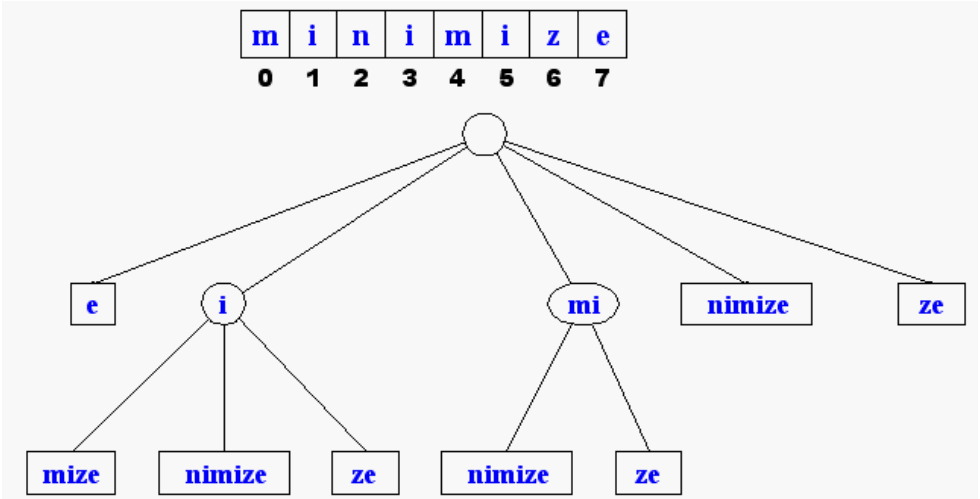
Example:



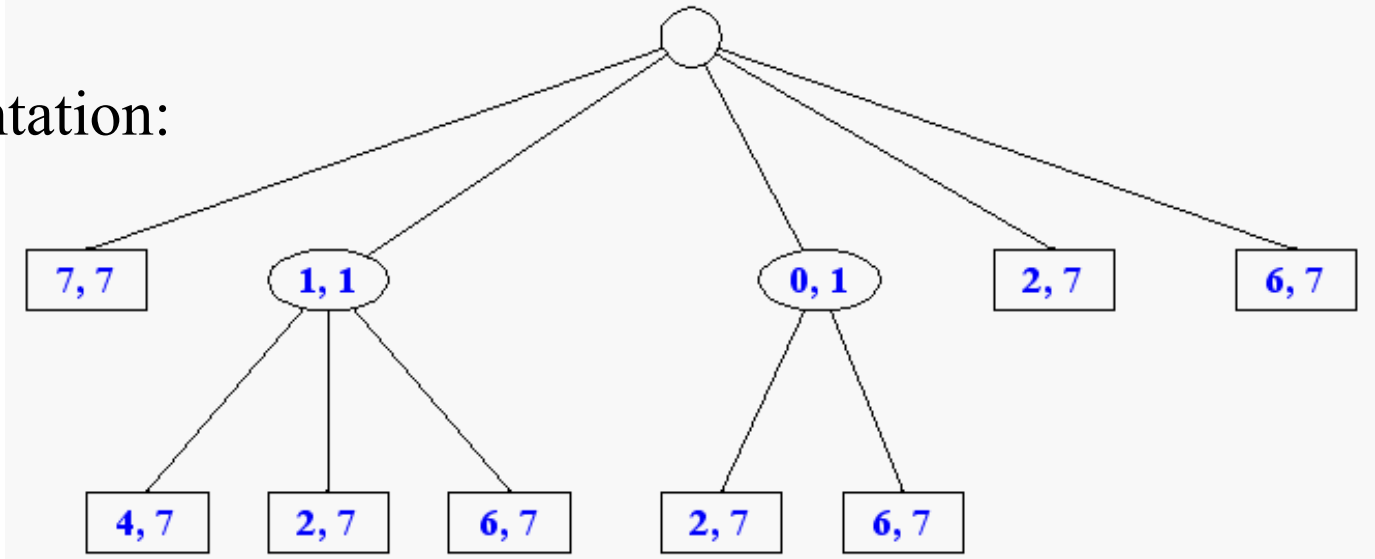
# Suffix Tries

- A ***suffix trie*** is a compressed trie for all the suffixes of a text

Example:




Compact representation:



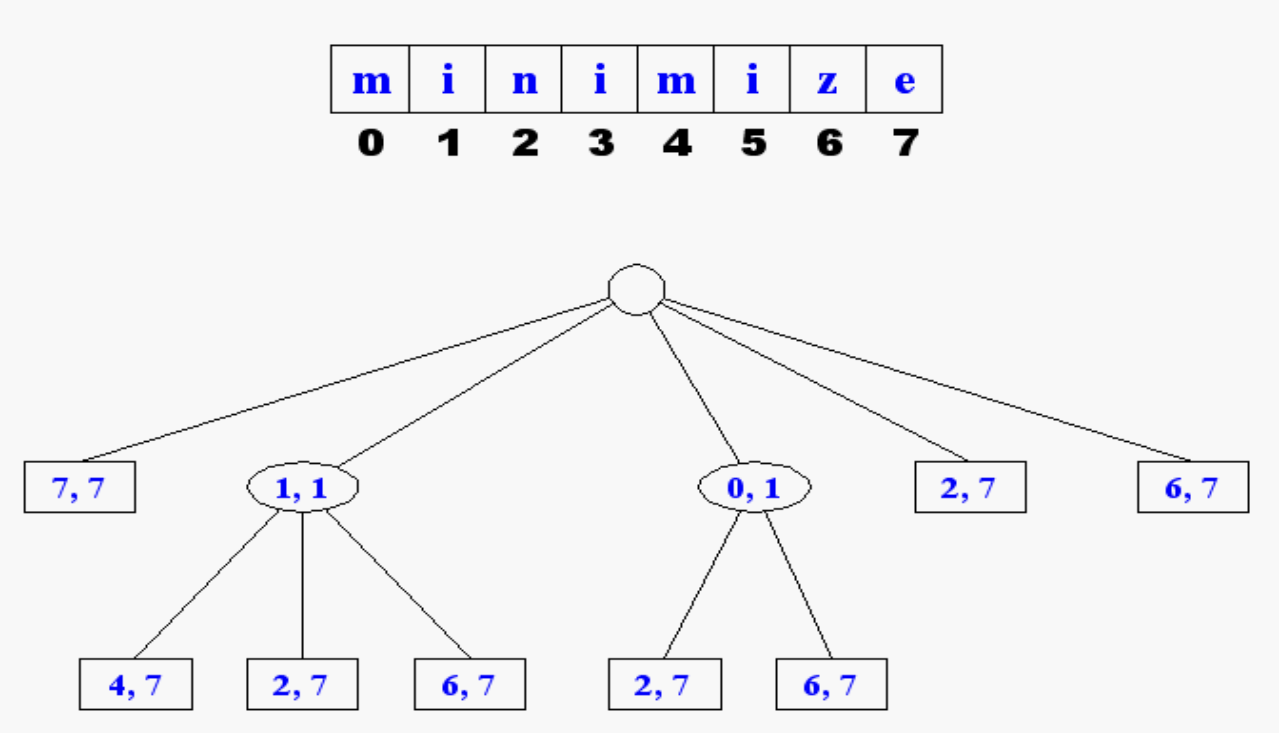
# Properties of Suffix Tries

- The ***suffix trie*** for a text X of size ***n*** from an alphabet of size ***d***

  -stores all the ***n(n-1)/2 suffixes*** of X in O(n) space

  -supports arbitrary ***pattern matching*** and prefix matching queries in ***O(dm) time***, where m is the length of the pattern

  -can be constructed in ***O(dn) time***



# Tries and Web Search Engines

- The ***index of a search engine*** (collection of all searchable words) is stored into a compressed trie
- Each leaf of the trie is associated with a word and has a list of pages (URLs) containing that word, called ***occurrence list***
- The trie is kept in internal memory
- The occurrence lists are kept in external memory and are ranked by relevance
- Boolean queries for sets of words (e.g., Java and coffee) correspond to set operations (e.g., intersection) on the occurrence lists
- Additional ***information retrieval*** techniques are used, such as
  - stopword elimination (e.g., ignore “the” “a” “is”)
  - stemming (e.g., identify “add” “adding” “added”)
  - link analysis (recognize authoritative pages)

# Tries and Internet Routers

- Computers on the internet (hosts) are identified by a unique 32-bit IP (***internet protocol***) address, usually written in "dotted-quad-decimal" notation
- E.g., www.iiits.ac.in is 116.206.105.72
- Use nslookup on Unix to find out IP addresses
- An organization uses a subset of IP addresses with the same prefix, e.g., 116.206.*.*, 128.132.*.*
- Data is sent to a host by fragmenting it into packets. Each packet carries the IP address of its destination.
- The internet whose nodes are ***routers***, and whose edges are communication links.
- A router forwards packets to its neighbors using IP ***prefix matching*** rules. E.g., a packet with IP prefix 116.206. should be forwarded to the IIITS gateway router.
- Routers use tries on the alphabet 0,1 to do prefix matching.