Tries

- Standard Tries
- Compressed Tries
- Suffix Tries
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 $S = \{\text{bear, bell, bid, bull, buy, sell, stock, stop}\}$

• 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,
Applications of Tries

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

Each operation is performed by tracing a path in the trie starting at the root.
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 \textit{index ranges} at the nodes.

$$
\begin{array}{c|cccc}
S[0] = & s & e & e \\
S[1] = & b & e & a & r \\
S[2] = & s & e & l & l \\
S[3] = & s & t & o & c & k \\
\hline
S[4] = & b & u & l & l \\
S[5] = & b & u & y \\
S[6] = & b & i & d \\
S[7] = & h & e & a & r \\
S[8] = & b & e & l & l \\
S[9] = & s & t & o & p \\
\end{array}
$$
Insertion and Deletion into/from a Compressed Trie

search stops here

insert(bbaaabb)
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.iitd.ac.in is 103.27.9.24
- Use nslookup on Unix to find out IP addresses.
- An organization uses a subset of IP addresses with the same prefix, e.g., IITD uses 103.27.*.*, Yale uses 130.132.*.*.
- 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 103.27. should be forwarded to the IITD gateway router.
- Routers use tries on the alphabet 0,1 to do prefix matching.