**String matching**: text $T$, pattern $P$ (substring)
find all/some occurrences of $P$ in $T$.

- obvious: $O(|T| \cdot |P|)$
- Knuth-Morris-Pratt [SICOMP 1977]: $O(|T|)$
  - preprocess $P$ into a finite automaton
- Boyer-Moore [CACM 1977]: $O(|T|)$ but sometimes even $O(|T|/|P|)$.
  - search for mismatch from back to front
  - mismatch early $\Rightarrow$ skip a lot
- Rabin-Karp [1987]: $O(|T|)$ w.h.p. (with high probability)
  - fingerprint $T[i:i+|P|]$ with $\sum_{k=1}^{i+|P|-1} T[k] \cdot x^k \bmod q$.
  - shift $i$ in $O(1)$.

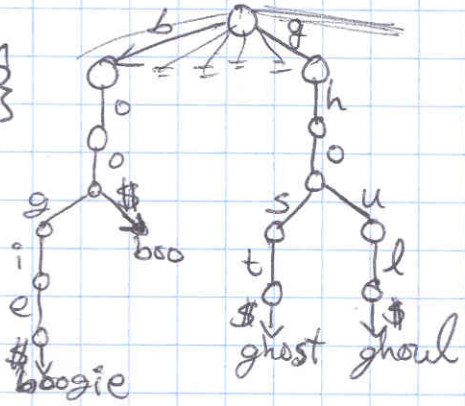
What if we can preprocess $T$?

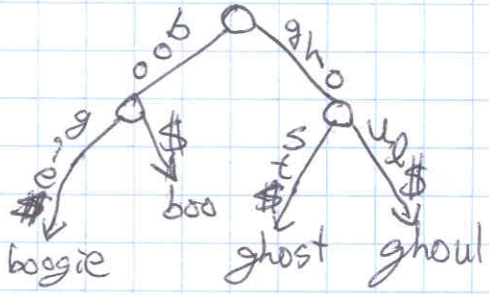
**Suffix tree** [Weiner 1973; McCreight 1976; Ukkonen 1995; Giegerich & Kurtz 1997]

- trie = tree with children pointers labeled with letters in $\Sigma$.
- trie $\leftrightarrow$ set of strings (root-to-leaf paths)
- terminate strings with \$ to allow prefixes

{ghost, ghoul, boogie, boo}

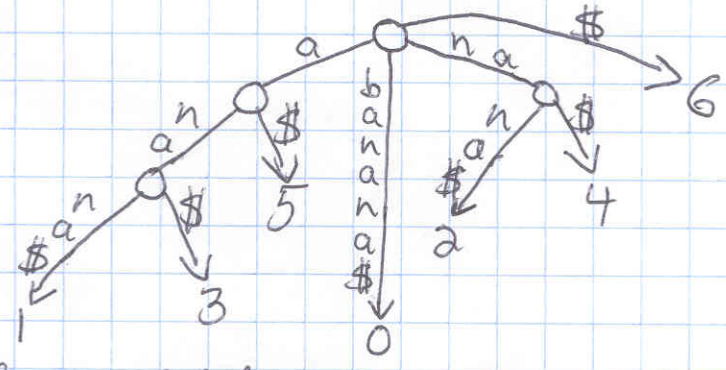
{ghost\$, ghoul\$, boogie\$, boo\$}




- compressed trie — store nonbranching paths as single labeled edges
- suffix tree/trie = compressed trie of all $|T|+1$ suffixes of $T$.

banana\$
0 1 2 3 4 5 6




- store edge labels with 2 indices into the text, or just store length & check later (& first letter) } linear space