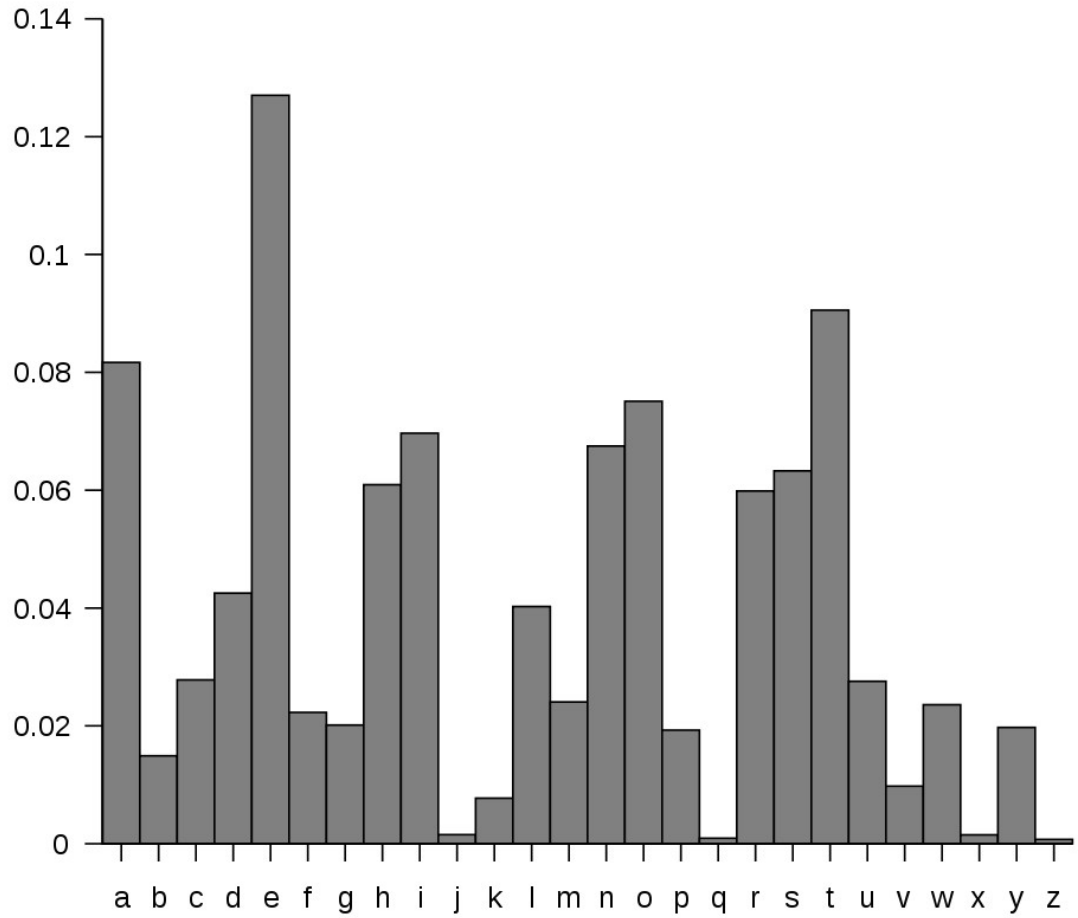
TRIES

CMPT 381

Outline

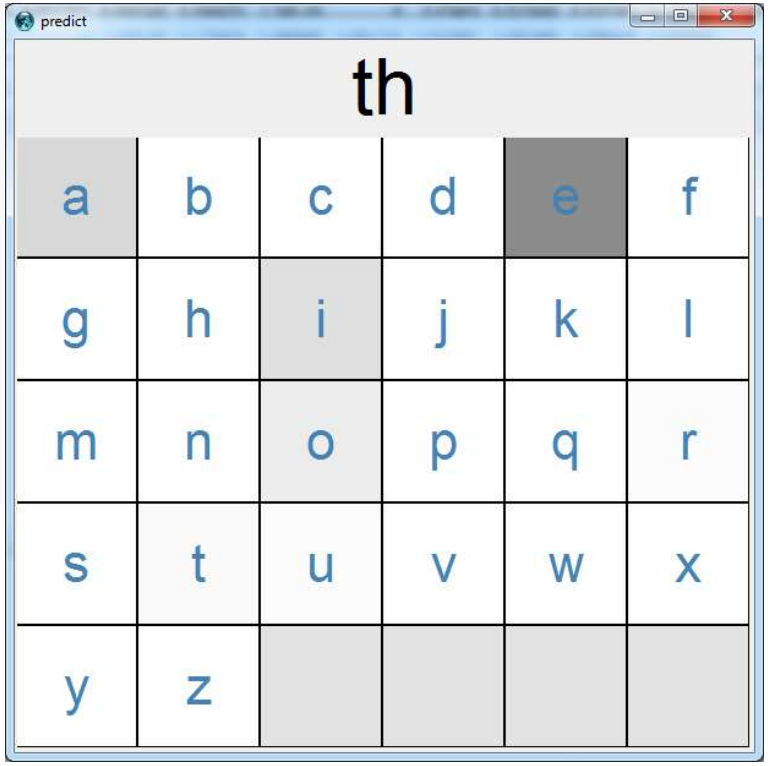
- Review: modeling text
- Tries
  - Operations on tries
  - Prediction using tries
  - Comparison of filtering and tries

Modeling text: letter frequencies

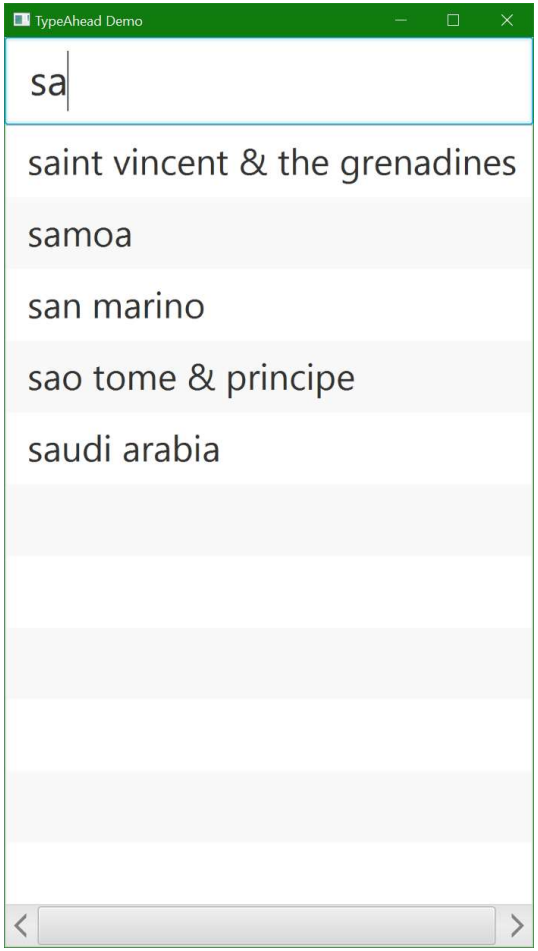


Modeling text: bigrams

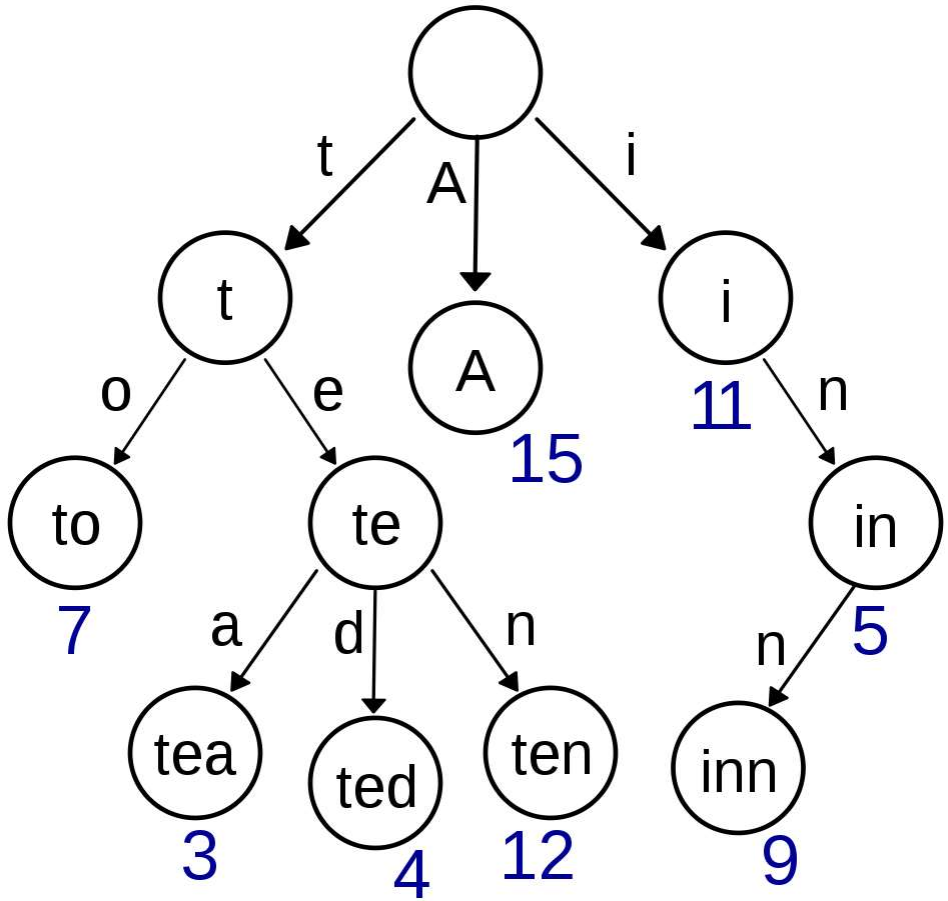
th 1.52%	en 0.55%	ng 0.18%
he 1.28%	ed 0.53%	of 0.16%
in 0.94%	to 0.52%	al 0.09%
er 0.94%	it 0.50%	de 0.09%
an 0.82%	ou 0.50%	se 0.08%
re 0.68%	ea 0.47%	le 0.08%
nd 0.63%	hi 0.46%	sa 0.06%
at 0.59%	is 0.46%	si 0.05%
on 0.57%	or 0.43%	ar 0.04%
nt 0.56%	ti 0.34%	ve 0.04%
ha 0.56%	as 0.33%	ra 0.04%
es 0.56%	te 0.27%	ld 0.02%
st 0.55%	et 0.19%	ur 0.02%



Type-ahead: filtering



Modeling text: tries



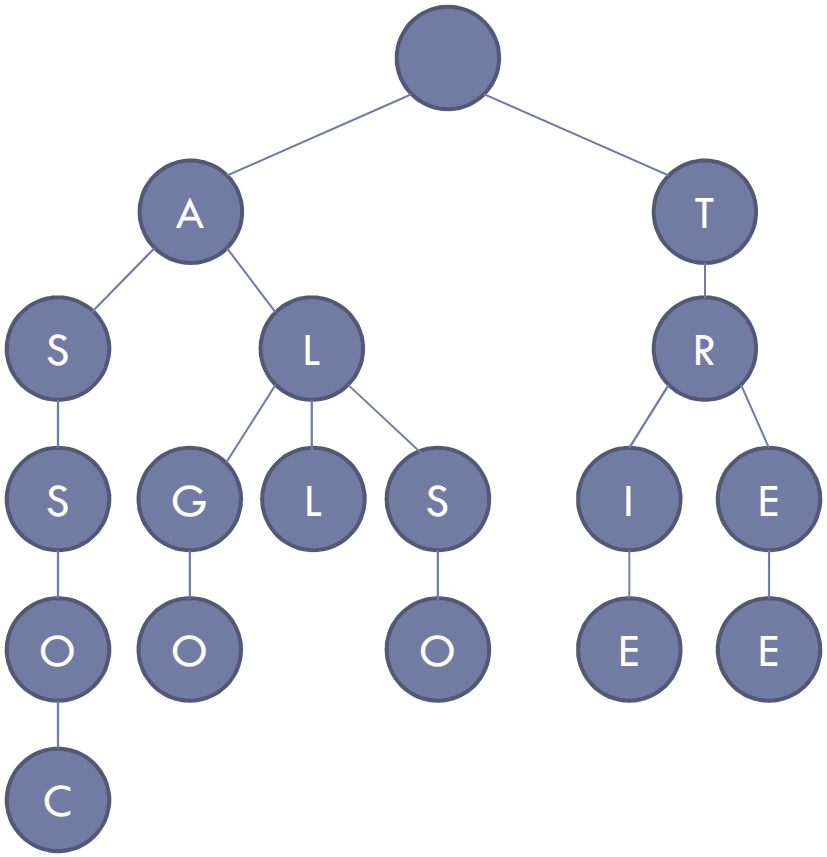
Tries

- Trie (or "prefix tree")
  - A tree where nodes have up to n children
    - where n is the size of the alphabet
  - All children of a node have a common prefix
  - Origin: "trie" from "re***trie***val" (inventor says pronunciation is "tree" but I will say "try")
- Useful for storing strings and searching for strings
  - Dictionaries in a spelling checker
  - Natural-language understanding
  - Text prediction

Trie example

TREE
TRIE
ALGO
ASSOC
ALL
ALSO

- add symbol and word counts



Operations on tries

- Insertion:
  - Traverse trie using characters of the word until:
    - Reach a leaf node (and some characters remain):
      - insert remaining characters
    - Reach end of word:
      - increment frequency count at that node
- Search:
  - Traverse trie using characters of the search string until:
    - Reach a leaf node (and some characters remain):
      - return false
    - Reach end of search string:
      - return true (and optionally the frequency count)

Operations on tries

- Deletion:
  - If word is in trie:
    - Traverse trie using characters of the word until:
      - A node's count is 1:
        - Remove this node
      - Reach a leaf node:
        - Decrement frequency count

Prediction using tries

- Based on a prefix (e.g., “sen”)
- Traverse trie, using prefix, arriving at node N
- All words descending from N are candidates
  - “senator”
  - “sense”
  - “sensible”
- Use depth-first search from N

Comparison to filtering

- Time complexity:
  - Trie insert and search: $O(L)$
    - where L is the length of the word
  - Filtering: $O(n)$
    - where n is the length of the list
- Space complexity
  - Trie: all common prefixes are shared
  - Filtering: no sharing