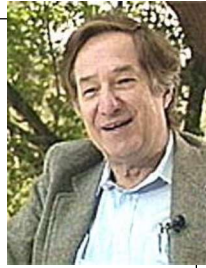


## Topic 24

### Tries

“In 1959, (Edward) Fredkin recommended that BBN (Bolt, Beranek and Newman, now BBN Technologies) purchase the very first PDP-1 to support research projects at BBN. ***The PDP-1 came with no software whatsoever.***



Fredkin wrote a PDP-1 assembler called FRAP (Free of Rules Assembly Program);”

Tries were first described by René de la Briandais in *File searching using variable length keys*.

## Clicker 1

► How would you pronounce “Trie”

- A. “tree”
- B. “tri – ee”
- C. “try”
- D. “tiara”
- E. something else

CS314

Tries

2

## Tries aka Prefix Trees

- Pronunciation:
- From retrieval
- Name coined by Computer Scientist Edward Fredkin
- **Retrieval** so “tree”
- ... but that is very confusing so most people pronounce it “try”

CS314

Tries

3

## Predictive Text and AutoComplete

- Search engines and texting applications guess what you want after typing only a few characters

Hel

hello  
hellboy  
hello fresh  
helen keller  
helena christensen  
hello may  
hell or high water  
hello neighbor  
helzberg  
help synonym

## AutoComplete

- ▶ So do other programs such as IDEs

```
String name = "Kelly J";
```

```
name.s
```

```
while  
S  
to  
i  
substring(int beginIndex, int endIndex) : String - String - 0.11%  
split(String regex) : String[] - String  
split(String regex, int limit) : String[] - String  
startsWith(String prefix) : boolean - String  
startsWith(String prefix, int toffset) : boolean - String  
subSequence(int beginIndex, int endIndex) : CharSequence - String  
substring(int beginIndex) : String - String
```

CS314

Tries

5

## Searching a Dictionary

- ▶ How?
- ▶ Could search a set for all values that start with the given prefix.
- ▶ Naively  $O(N)$  (search the whole data structure).
- ▶ Could improve if possible to do a binary search for prefix and then localize search to that location.

CS314

Tries

6

## Tries

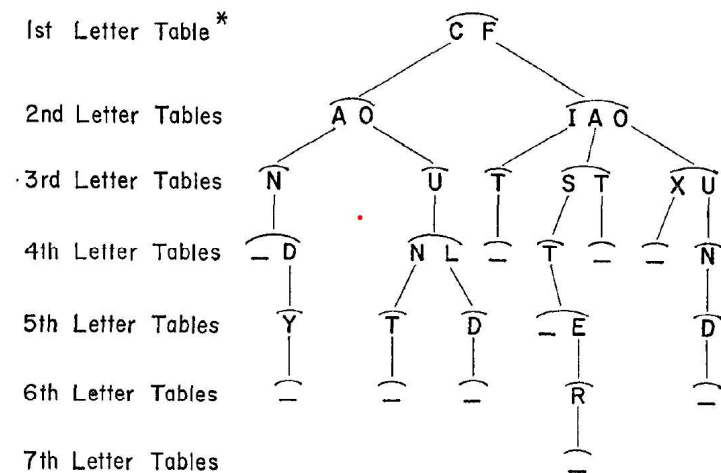
- ▶ A general tree (more than 2 children possible)
- ▶ Root node (or possibly a list of root nodes)
- ▶ Nodes can have many children
  - not a binary tree
- ▶ In simplest form each node stores a character and a data structure (list?) to refer to its children
- ▶ "Stores" all the words or phrases in a dictionary.
- ▶ How?

CS314

Tries

7

## René de la Briandais Original Paper



\*All entries of any one table are covered by a single arc (—).

Fig. 1—Formation of a set of tables.

CS314

Tries

8



?????

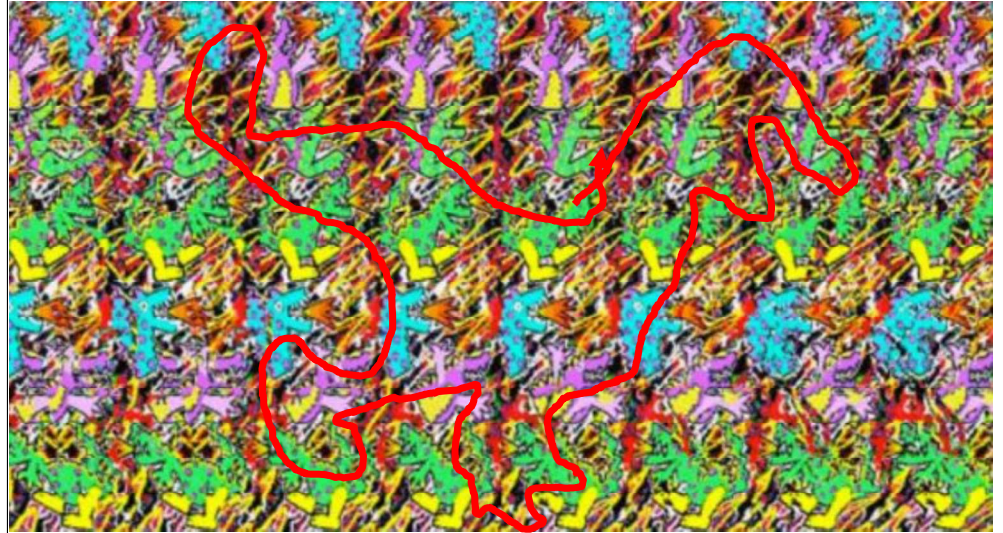


CS314

Tries

9

?????



Picture of a Dinosaur

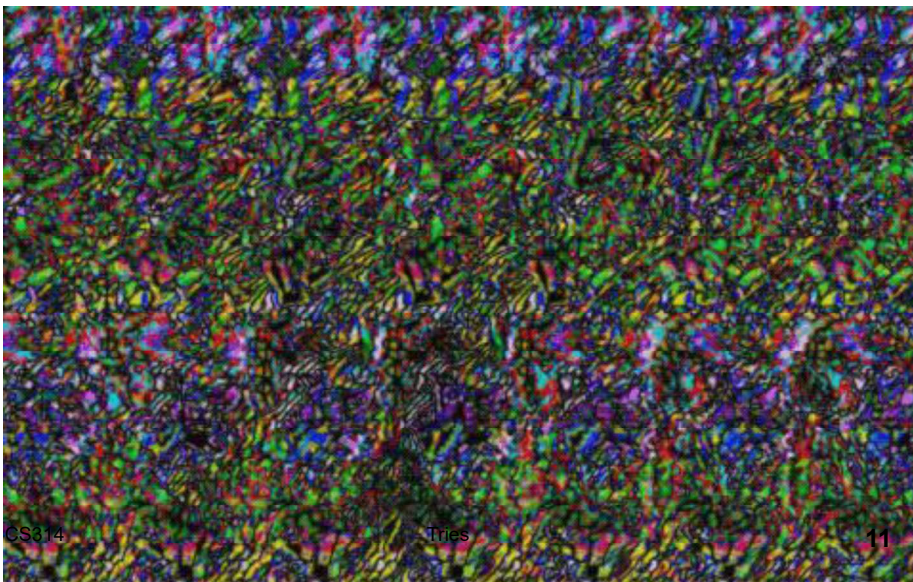
CS314

Tries

10

Fall 2022 - Ryan P.

Created with Procreate: <https://procreate.art/>

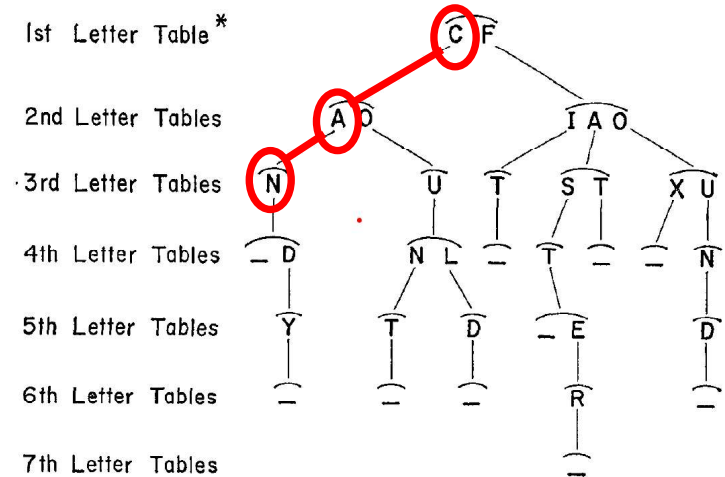


CS314

Tries

11

Can



\*All entries of any one table are covered by a single arc (—).

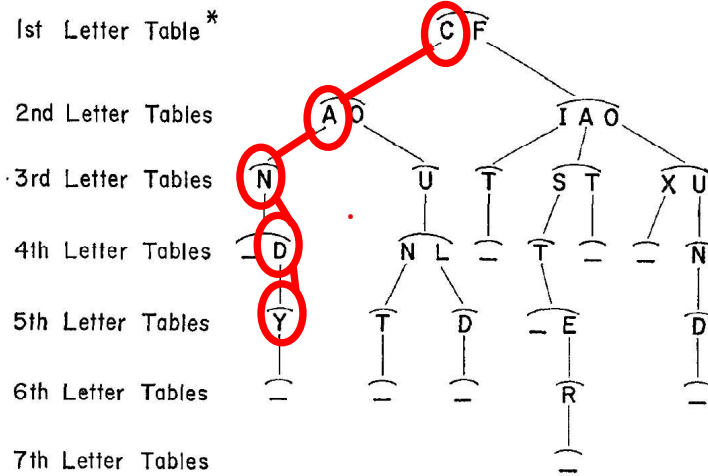
Fig. 1—Formation of a set of tables.

CS314

Tries

12

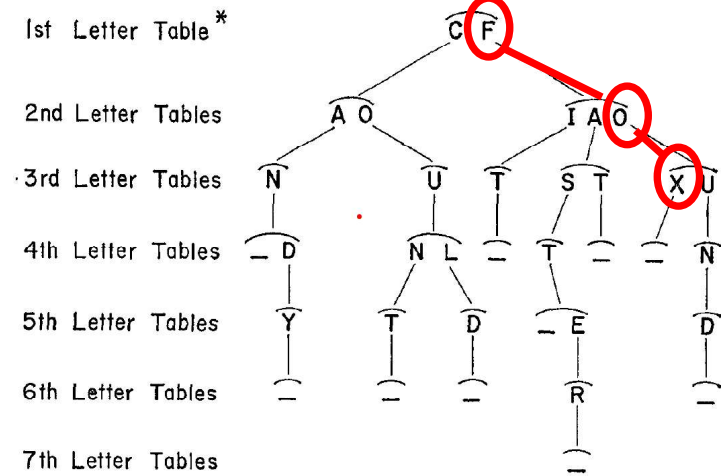
## Candy



\*All entries of any one table are covered by a single arc (—).

Fig. 1—Formation of a set of tables.

## Fox



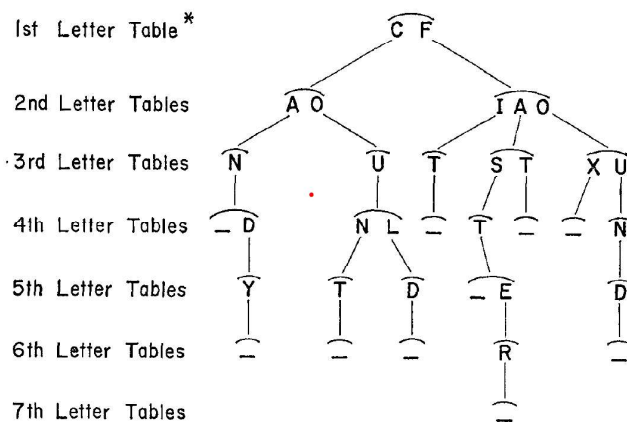
\*All entries of any one table are covered by a single arc (—).

Fig. 1—Formation of a set of tables.

## Clicker 2

► Is “fast” in the dictionary represented by this Trie?

- A. No
- B. Yes
- C. It depends



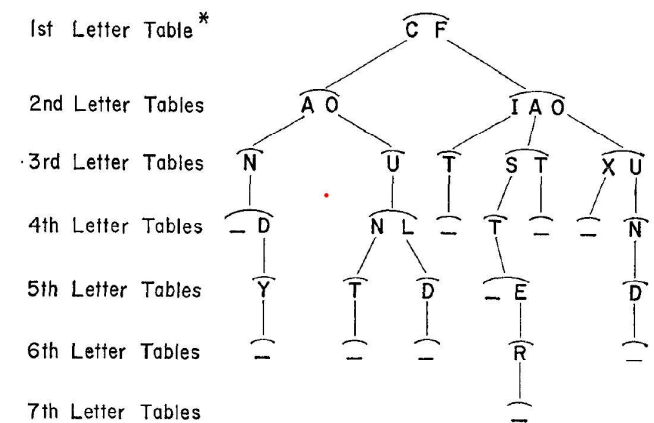
\*All entries of any one table are covered by a single arc (—).

Fig. 1—Formation of a set of tables.

## Clicker 3

► Is “fist” in the dictionary represented by this Trie?

- A. No
- B. Yes
- C. It depends



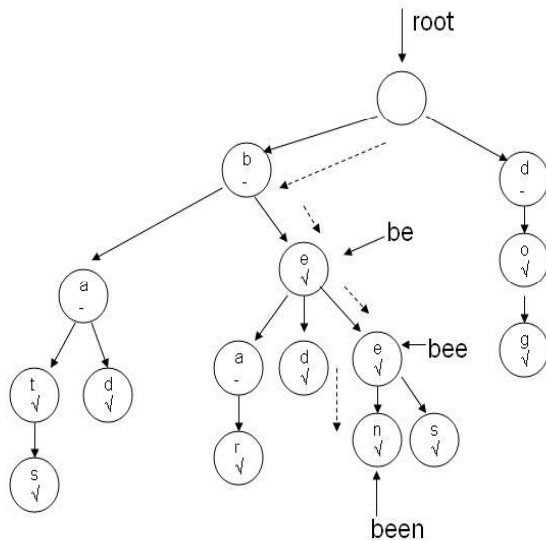
\*All entries of any one table are covered by a single arc (—).

Fig. 1—Formation of a set of tables.



## Tries

- ▶ Another example of a Trie
- ▶ Each node stores:
  - A char
  - A boolean indicating if the string ending at that node is a word
  - A list of children



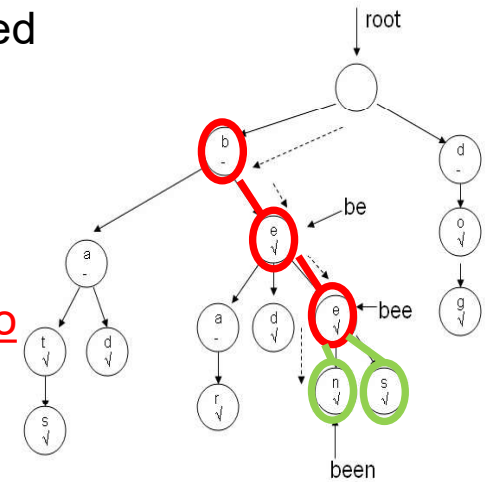
CS314

Tries

17

## Predictive Text and AutoComplete

- ▶ As characters are entered we descend the Trie
- ▶ ... and from the current node ...
- ▶ ... we can descend to terminators and leaves to see all possible words based on current prefix
- ▶ b, e, e -> bee, been, bees



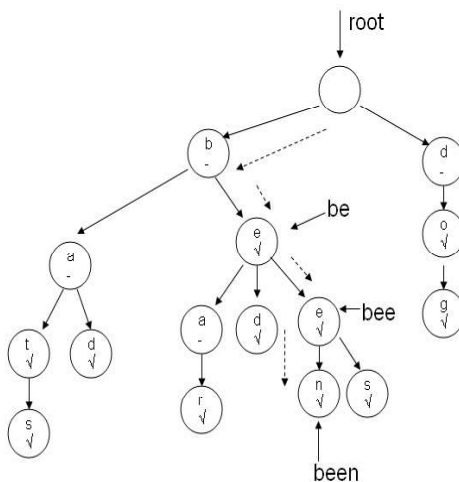
CS314

Tries

18

## Tries

- ▶ Stores words and phrases.
  - other values possible, but typically Strings
- ▶ The whole word or phrase is not actually stored in a single node.
- ▶ ... rather the path in the tree represents the word.



## Implementing a Trie

```
public class Trie {

    private TNode root;
    private int size; // number of words
    private int numNodes;

    public Trie() {
        root = new TNode();
        numNodes = 1;
    }
}
```

CS314

Tries

20

## TNode Class

```
private static class TNode {  
    private boolean word;  
    private char ch;  
    private LinkedList<TNode> children;
```

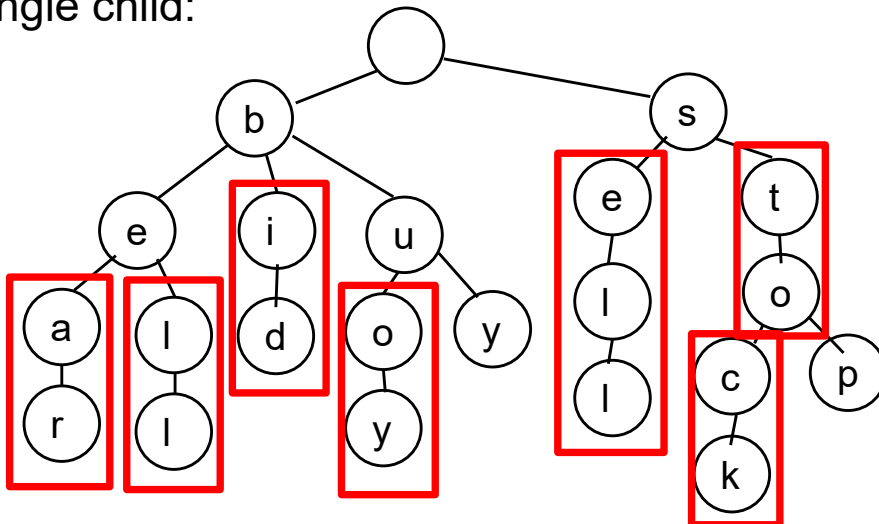
- Basic implementation uses a LinkedList of TNode objects for children
- Other options?
  - ArrayList?
  - Something more exotic?

## Basic Operations

- Adding a word to the Trie
- Getting all words with given prefix
- Demo in IDE

## Compressed Tries

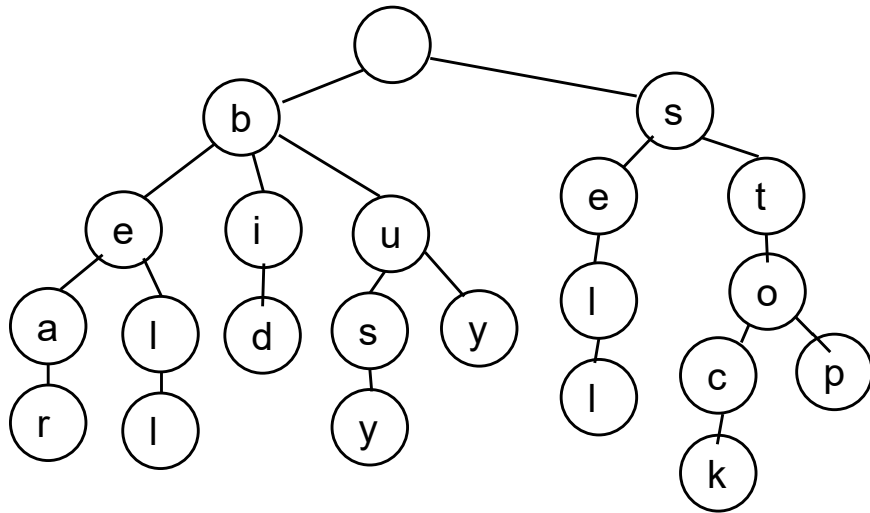
- Some words, especially long ones, lead to a chain of nodes with single child, followed by single child:



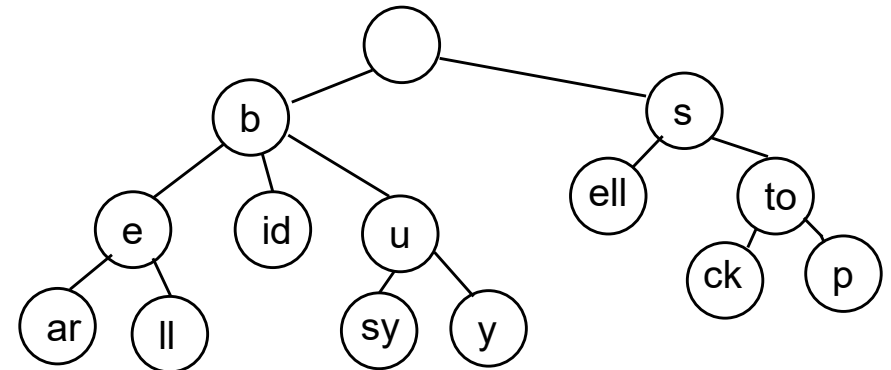
## Compressed Trie

- Reduce number of nodes, by having nodes store Strings
- A chain of single child followed by single child (followed by single child ... ) is compressed to a single node with that String
- Does not have to be a chain that terminates in a leaf node
  - Can be an internal chain of nodes

## Original, Uncompressed



## Compressed Version



8 fewer nodes compared to uncompressed version  
s – t – o – c – k