Department of Electronics

Data Structures and Algorithms Assessment

Predictive Text

Y3839090

January 13, 2017

Contents

1 How the program works

dummy text

1.1 How users interact with the program

dummy text

1.2 How the prediction function works

dummy text

2 Data Structures

A predictive text program needs a to be able to access a set of common words. Storing and accessing these words efficiently is one of the challenges in creating a fast predictive text engine. I decided to use a trie[2] structure.

2.1 Trie

2.1.1 What is a trie and why I've used it

The trie is a tree like data structure often used for storing strings. Each node in the trie represents a single letter in a string. Each node in a trie has a fixed number of child nodes like a binary search tree. Unlike the binary search tree the trie does not have two child nodes,it has one for each letter of the alphabet. This allows the trie nodes to not store there value because there position determines it. A node's children share a common prefix,that prefix is the value of there parent node.

There are many features of tries that make them a good choice for a predictive text system. One of them is that values can be look up by there prefixes. Predictive text systems use a partial word (a prefix) to search for possible full words. So it's important that prefix lookups are fast and efficient. [TODO:: ADD MORE STUFF HERE]

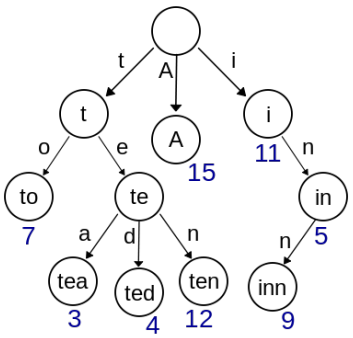

Figure 1: A trie for keys "A","to", "tea", "ted", "ten", "i", "in", and "inn"[1].

References

[1] Booyabazooka (based on PNG image by Deco). Modifications by Superm401. *Example of a trie*. [Online; accessed 13-January-2017]. 2006. URL: `https://en.wikipedia.org/wiki/File:Trie_example.svg`.

[2] Peter Brass. *Advanced Data Structures*. Cambridge University Press, 2008.