CSCI 3333 Homework AC2: $O(1)$-Time Autocomplete

1 Introduction

Homework hwAC1 asked you to implement autocomplete using a balanced binary search tree. This led to roughly $O(\log(n))$ worst-case running times, which is pretty good! But if you're a fast typer and you've ever used autocomplete on a modern phone, you know that any latency whatsoever is obnoxious.

The goal of this assignment is to speed up autocomplete by implementing the same `Autocompleter` abstract data type (i.e. public methods) using a different data structure. The goal will be operations that all run in $O(1)$ worst-case running time![1] To achieve this, a *trie* (augmented with the most-frequent `Entry`s in each subtree) will be used.

2 Instructions

The following files have been given to you:

1. A C++ header file (`autocompleter.h`) declaring the Autocompleter class.
2. A C++ source file (`main.cpp`) containing a `main` function with tests.
3. A text file (`words2.txt`) containing 300000 common English words and their frequencies.[2]

Create a new C++ source file named `autocompleter.cpp` that implements the class declared in `autocompleter.h`, so that `autocompleter.cpp` and the provided files compile into a program that runs with no failed tests. Submit the source file `autocompleter.cpp`.

[1] Assuming that all strings have constant length.

[2] Source: `http://norvig.com/ngrams/count_1w.txt`.

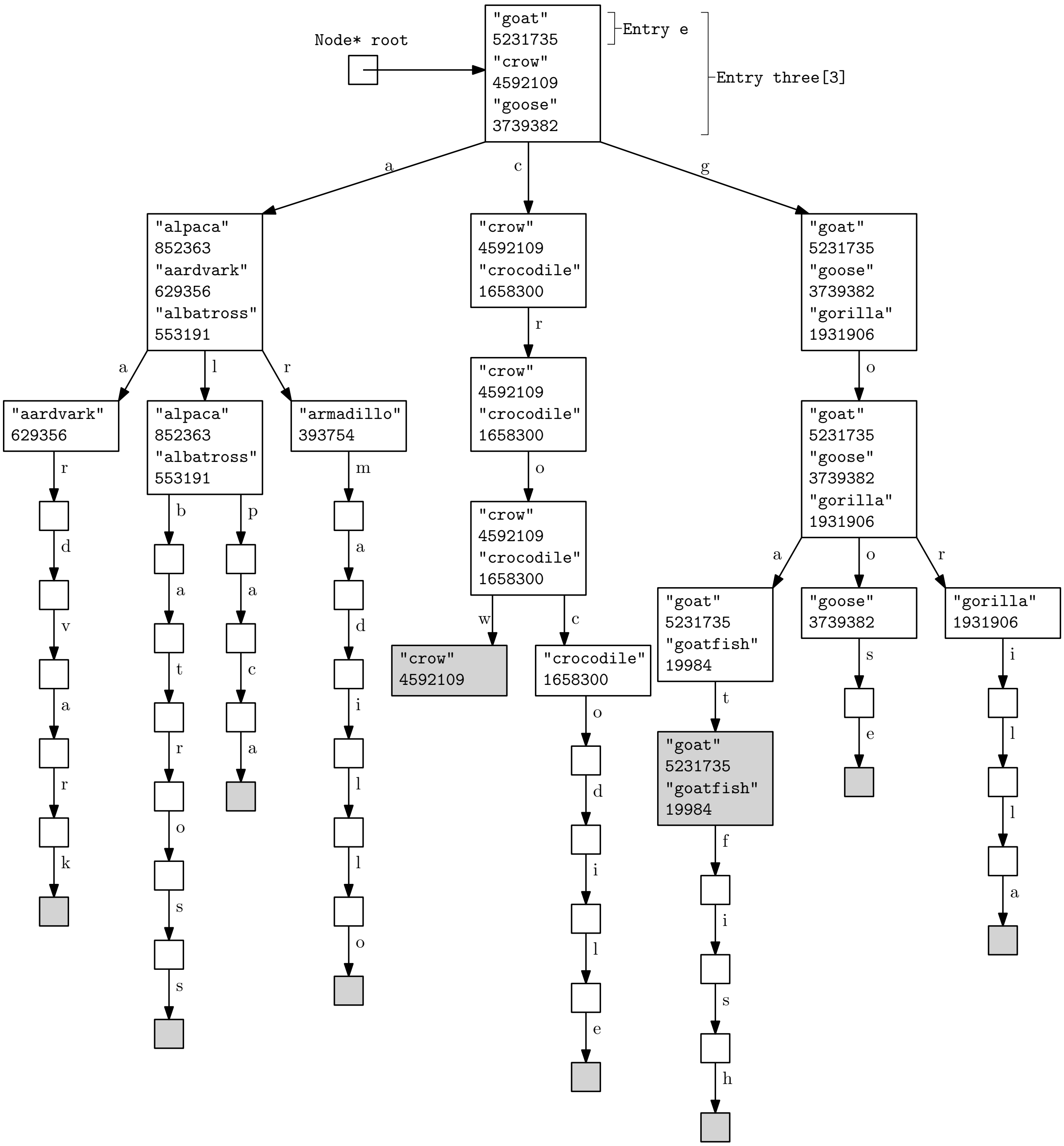

Figure 1: An Autocompleter for a set of 10 words.