

Advanced Topics in Data Structures

By Eric Schles

Recap

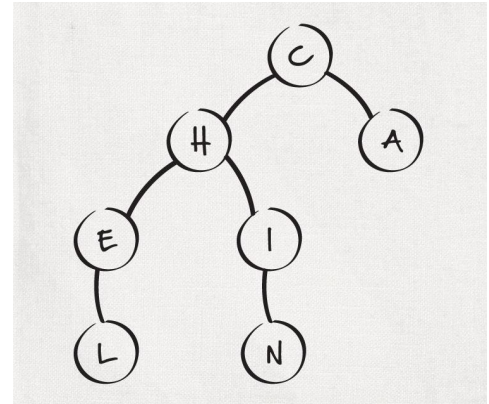
In part one I looked at running time analysis. The question I was trying to answer was, how efficiently can we access our data.

Now I will turn to the far harder question of, how do we efficiently store our data.

Enter the Trie

A trie is nothing more than a purposely unbalanced tree. The goal of a trie is to conserve as much memory as possible.

Most common use:
auto-complete of words



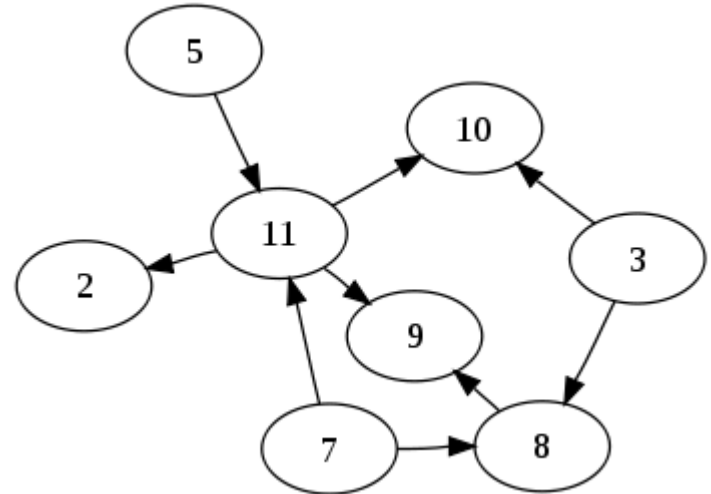
DAWG: Directed Acyclic Word Graph

A directed acyclic word graph is a special case of a directed acyclic graph (for words).

Example DAG:

Rules:

- 1) No Cycles
- 2) Each node points at, at most one other node



Comparing Tries and DAWGs

Tries:

- only one path to a given node
- easy to code

DAWGs:

- multiple paths to a node -> more space efficient than Tries
- harder to code

Demo

Installation:

```
sudo pip install DAWG
```

```
sudo pip install biopython
```

Demo continued!

[demo]

Reference

<http://kmike.ru/python-data-structures/>

--a litany of advanced data structures in python

<http://www.toptal.com/java/the-trie-a-neglected-data-structure>

--why tries are awesome

http://en.wikipedia.org/wiki/Deterministic_acyclic_finite_state_automaton

--in depth discussion of the difference between DAWGs and Tries