Term Project

14-JAT; Alex Merino, Jorge Vucanovic, T'Avion Rodgers

Goal and Motivation

- With the implementation of some Data Structures and Algorithms we would like to build a system capable of guessing queries from partial queries, with high accuracy, and efficient CPU and Memory usage. Designing such program would provide insights into how systems of greater scale and complexity are developed.

Initial Approach

- Normal Trie
  - Each Node contains one character, one parent, one child, one sibling Node, five best guesses below
  - Used HW3 LinkedTree file
- Guessing
  - Traversed up tree to find each letter

Final Approach

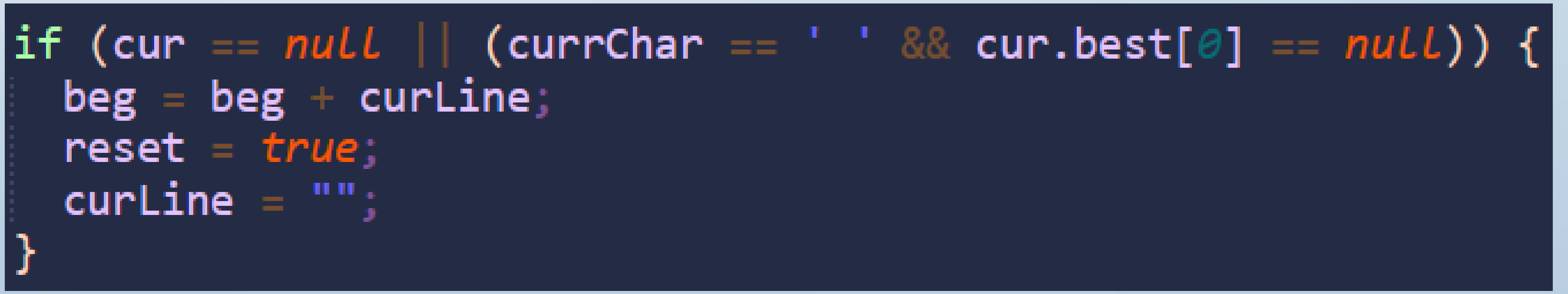
- Final Node of query contained full query to increase guessing speed
- If no more guesses then reset from root of trie and try again starting from current letter
- Sorted children of each Node by how many children each child has

Evaluation

	Algorithm Initial	Algorithm Final	Beach Initial	Beach Final
Accuracy	77.9131	80.2079	81.1291	82.5460
Time (Seconds)	1.28E-6	1.4217E-6	1.56E-6	1.683E-6
Memory (bytes)	9.06E7	9.322E7	3.37E7	3.3835E7
Score	564.2369	558.8226	909.0517	902.9463

Analysis: Accuracy

- Most accuracy lost is by queries that are brand new
- Gained accuracy by resetting search if
  - Current word is not found
  - There are no guesses and the current Node has no guesses



```
if (cur == null || (currChar == ' ' && cur.best[0] == null)) {
  beg = beg + curLine;
  reset = true;
  curLine = "";
}
```

Analysis: Speed

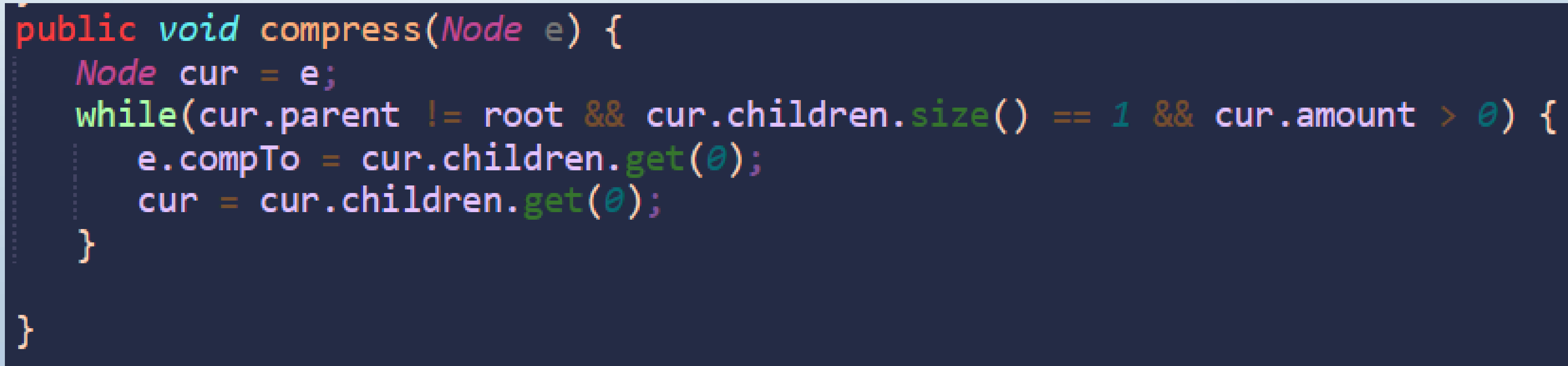
- Quicker guessing time since guesses are preloaded to each Node
  - O(1) time
- Quicker find time
  - Utilizing class traversal Node, only children of one Node need to be checked
  - O(m), worst case check every child of one Node
- Insertion:
  - O(m^2)
  - Worst case: Must go through each child Node for each letter in query

Analysis: Memory

- More memory usage:
  - Using integer values instead of short
  - Storing entire query at last Node
  - Storing 5 guesses at each node
    - Even if null, array still takes space

Analysis: Future Improvements

- Compressing Trie to save space
  - Slightly functional method but could not get implemented fully due to find() method
  - Store full queries in a different way
    - Traversing up Node takes time
    - Storing query takes space



```
public void compress(Node e) {
    Node cur = e;
    while(cur.parent != root && cur.children.size() == 1 && cur.amount > 0) {
        e.compTo = cur.children.get(0);
        cur = cur.children.get(0);
    }

}
```