Tries

Joshua Ganschow - DSU Computer Club

March 5, 2025

Introduction

- **Definition:** A trie (or prefix tree) is a search tree data structure that is used to store and retrieve strings.
- **Motivation:** They are particularly useful for things like autocomplete or searching within a database.

Trie Structure and Properties

- **Nodes and Edges:** Each node represents a character and a node's children represent characters that occur after the parent node in a string.
- **Key Properties:** Children nodes that represent the next characters, and something to mark that a node is the end of a string.

Key Terms and Definitions

- **Node:** The basic unit of a trie representing a single character. Nodes are linked level-by-level to form keys.
- **Key:** A string or sequence of characters stored in the trie.
- **Prefix:** A substring that represents the initial segment of a key.
- **Child/Parent:** Relationship between nodes; a parent node connects to one or more child nodes, forming the path of a key.
- **End-of-Word Marker:** A flag within a node indicating that the node represents the final character of a valid key.

Basic Operations

- **Insertion:** traverse starting at the root node, adding children as necessary to represent what needs to be inserted.
- **Search:** traverse starting at the root node, keep going until you either hit a dead end (next character in string we're searching for is not child of current node) or we've gotten through the entire string and the current node is marked as an end node.
- **Deletion:** (assuming key does exist in trie) traverse starting at the root node until we reach the end of the key we're trying to delete. When we reach the end, unmark current node as an end node (effectively removing the key from the trie).

Time and Space Complexity

- **Time Complexity:** O(m) for all operations where m is the length of the keys.
- **Space Complexity:** Worst case of O(m*n) where n is number of keys and m is length of each key. In practice, it will be better than this because common prefixes will be shared by keys in the trie.
- **Comparison to hash table:**
  1. Hash tables are hypothetically faster for searching with an average time complexity of O(1) but could be slightly less optimal with a bad hashing function.
  2. Requires less space than the hash table because the hash table will require extra space for the table and handling collisions.

Code and Demonstration

I've written a simple python program that implements a trie, initializes it with 10,000 common English words and has a simple GUI to show autocomplete options.

Discussion and Q&A

Any questions?

Conclusion

Thanks for coming!