Data Structures

Trie Implementation

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

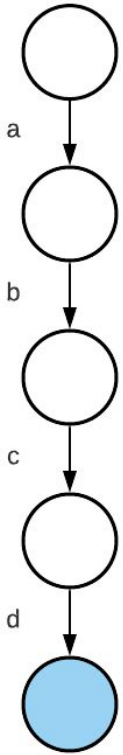
Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Representation

- A fast-way to create & access your M children is to create array of pointers
- Then each array index represent an edge.
- Initially all are null (memset), indicating no children

```
 7  class trie {
 8  private:
 9      static const int MAX_CHAR = 26;
10      trie* child[MAX_CHAR];
11      bool isLeaf {};
12
13  public:
14      trie() {
15          // set an array to 0s. Here pointers to null
16          memset(child, 0, sizeof(child));
17      }
```

Insertion



- To insert we go with the string letter by letter, create edges if it doesn't exist. Mark the last node as leaf
- Observe: char - '0' converts a character to [0-25] index

```
void insert(string str, int idx = 0) {
    if (idx == (int) str.size())
        isLeaf = 1;
    else {
        int cur = str[idx] - 'a';
        if (child[cur] == 0)
            child[cur] = new trie();
        child[cur]->insert(str, idx + 1);
    }
}
```

Does a complete word exist?

- Iterate tell you finish tracing the whole word
- Its last node must be marked as isLeaf = true

```
bool word_exist(string str, int idx = 0) {
    if (idx == (int) str.size())
        return isLeaf;  // there is a string marked here

    int cur = str[idx] - 'a';
    if (!child[cur])
        return false;   // such path don't exist

    return child[cur]->word_exist(str, idx + 1);
}
```

Does a prefix exist?

- Same logic, but once ended return true
- Clearly, all the functions are O(L) time complexity
  - All can be rewritten iteratively to avoid extra O(L) for memory auxiliary space

```
bool prefix_exist(string str, int idx = 0) {
    if (idx == (int) str.size())
        return true;    // all subword covered

    int cur = str[idx] - 'a';
    if (!child[cur])
        return false;   // such path don't exist

    return child[cur]->prefix_exist(str, idx + 1);
}
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."