**ALGORITHMS & DATA STRUCTURES**

ADDITIONAL INFORMATION FOR LAB 18: COMPRESSED TRIES, PART 2

WEEK OF 18TH MARCH 2019

1. **BIG PICTURE**

The query method is a recursive function that receives as an input argument the string s. If the string is contained in the compressed trie, the query method must return TRUE. Otherwise, it must return FALSE.

The method has 3 base cases (3 cases where it returns either FALSE or TRUE):

**Base case 1:** The compressed trie is empty. In that case, it must return FALSE

**Base case 2:** The whole string s is contained in the first level of the compressed trie. In that case, it returns TRUE

**Base case 3:** After recursively checking the whole compressed trie, the string was not found. It must return FALSE

The method must be recursively called when a part of the string (not the whole string, as base case 2 deals with this situation) is found in the first level of the compressed trie. If this is the case (one part of the string is found in the first level of the compressed trie), then the query method must be called again. This new call must work on the next level of the compressed trie and must receive as an input argument the remaining of the string.

The pseudocode for this would be:

|  |
| --- |
| function query(s)  **if** (compressed trie is empty) **then** //base case 1  return false  **end if**  **if** (s+‘{‘ **end for ifturn .query(s.substring(0,i))rie) then hrough the compressed trie.ompressed trie), then the query method must be c end for ifturn .query(s.substring(0,i))rie) then hrough the compressed trie.ompressed trie), then the query method must be c** in first level of compressed trie) **then** //base case 2  return true  **end if**  **for** s.length() ≥ i > 0 **do**  **if** (s.substring(0,i) in first level of compressed trie) **then**  **return** next\_level.query(s.substring(0,i))  **end if**  **end for**  **return false**  **end function** |

Let’s look at the execution of this code for a simple example. Consider the compressed trie of Figure 1.



**Figure 1.** Compressed trie storing the words “moon”, “monk”, “mop”, “money” and “monkey”.

Assume that the method query is called with the string “moon” as input argument. Then:

* The first base case is evaluated. The compressed trie is not empty. Thus, the instructions of the first if are not executed.
* The second base case is evaluated. The string “moon{“ is not in the first level. Thus, the instructions of the second if are not executed.
* We enter the for loop: s.length() is 4.
  + For i=4, the substring is “moon”, this is not in the first level of the trie. Go for the second iteration.
  + For i=3, the substring “moo” is not in the first level of the trie. Go for the third iteration.
  + For i=2, the substring “mo” is in the first level of the trie. Thus, the method query is called again for the next level of the trie, using the string “on” as input argument.
  + Recursive call query(“on”)
    - The first base case is evaluated. The compressed trie is not empty. Thus, the instructions of the first if are not executed.
    - The second base case is evaluated. The string “on{“ is found in this level. Thus TRUE is returned
  + TRUE is returned