1. What is the tightest estimate you can prove on the memory consumption of a trie built off n non-empty patterns $p_1, p_2, \ldots, p_n$ if all the patterns' lengths are bounded from above by L, and the sum of lengths of all patterns is no more than S?

1 / 1 point

- (x) $O(S)$
- ( ) $O(n+L)$
- ( ) $O(nL)$
- ( ) $O(n)$

Correct

Correct! Indeed, even if all patterns start with different characters, and so each pattern creates a whole separate branch of the tree, still the sum of lengths of those branches is no more than the total number of characters in all the patterns, which is S. You will also need some space for the root vertex of the trie, but this is just $O(1)$. You also need to store the links from the nodes of the trie to their children, but the total number of such links is also $O(S)$, and if you store them using lists of children in each node, it will take just $O(S)$ memory (if you store an array of potential children of size A, the size of the alphabet, in each node, you'll need more memory - $O(SA)$). However, if you use regular linked lists, searching in such a trie could become too slow. Instead, you can store a hash table in each node, mapping from characters to the corresponding children.

Accurate reader might object that we also need additional $O(n)$ memory to store n patterns, but if all n patterns are non-empty, then $S \geq n$, so $O(S+n) = O(S)$.

2. What is the time complexity of searching all occurrences of n patterns $p_1, p_2, \ldots, p_n$ in text T of length $|T|$ if all patterns have length at most L and the sum of their lengths is at most S?

1 / 1 point

- ( ) $O(S)$
- (x) $O(|T|L)$
- ( ) $O(|T|S)$
- ( ) $O(L)$

Correct

Correct! Indeed, for each of the $|T|$ starting positions in the text T, in the worst case you would need to traverse the trie from its root to the deepest node, and any such path has length at most L.

3. What is the smallest possible number of nodes in a trie built off n patterns $p_1, p_2, \ldots, p_n$ if all the patterns have the same length $L > 0$?

1 / 1 point

- (●) $L + 1$
- ( ) n
- ( ) nL
- ( ) L

Correct
Correct! If all the patterns are the same, you will need exactly $L + 1$ nodes: the root node and L more nodes to store each character of the first pattern. All the other patterns will use the same nodes.

4. If you take all the suffixes of a string S of length L and build a regular trie off those suffixes as patterns, what is the maximum possible number of nodes in such trie?

1 / 1 point

- (●) $\frac{L(L+1)}{2} + 1$
- ( ) $L + 1$
- ( ) $L^2 + 1$

Correct
Correct! Indeed, if all the characters in S are different, all the suffixes will start with different characters, so each of them will require a whole separate branch in the trie. The longest suffix will generate a branch of length L, the next one will generate a branch of length $L - 1$, and so on, down to length 1 for the shortest suffix. The total number of nodes will be $1 + L + (L - 1) + \cdots + 1 = \frac{L(L+1)}{2} + 1$.

5. What is the smallest possible number of nodes in a suffix tree of string S with length L?

1 / 1 point

- ( ) L
- ( ) $\frac{L(L+1)}{2} + 1$
- (●) $L + 1$

Correct
Correct! If the string S is $aaa \ldots aaa$ - consists of many letters a - then all the suffixes of this string are also of the form $aaa \ldots aaa$, so they will all fit into the single branch of the suffix tree, this branch will contain the root node and L more nodes for the longest suffix of the string S.