CS 304 Theoretical Computer Science

Turing Machine Project

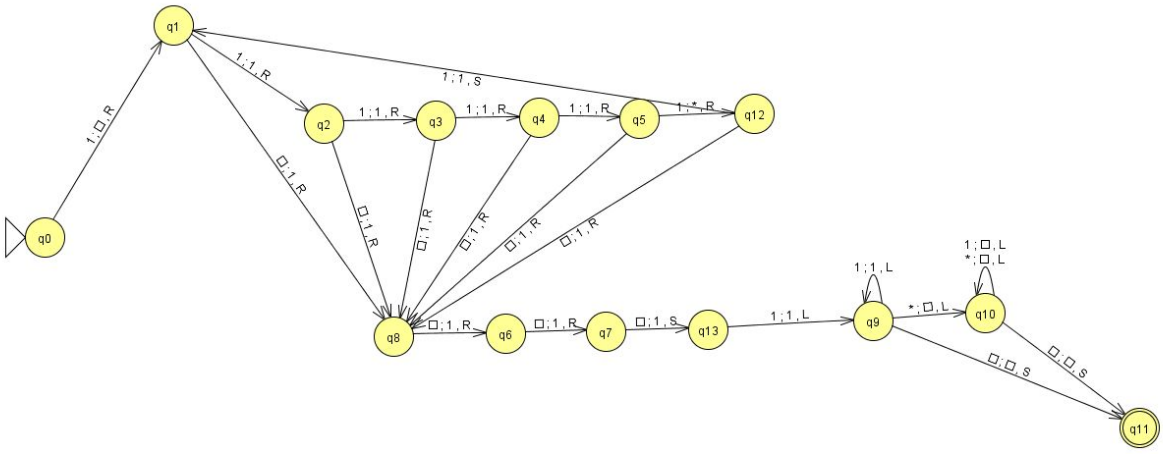
31 October 2017 / Fall 2017 / CCNY

Mohamed Sondo, Sahil Jain, Viswanath Misir

Problem Statement

Design a Turing Machine to compute the number-theoretical function f(n) = [(n mod 5) + 3].
Remember the representational 1. Using JFLAP

Our Algorithm Explained

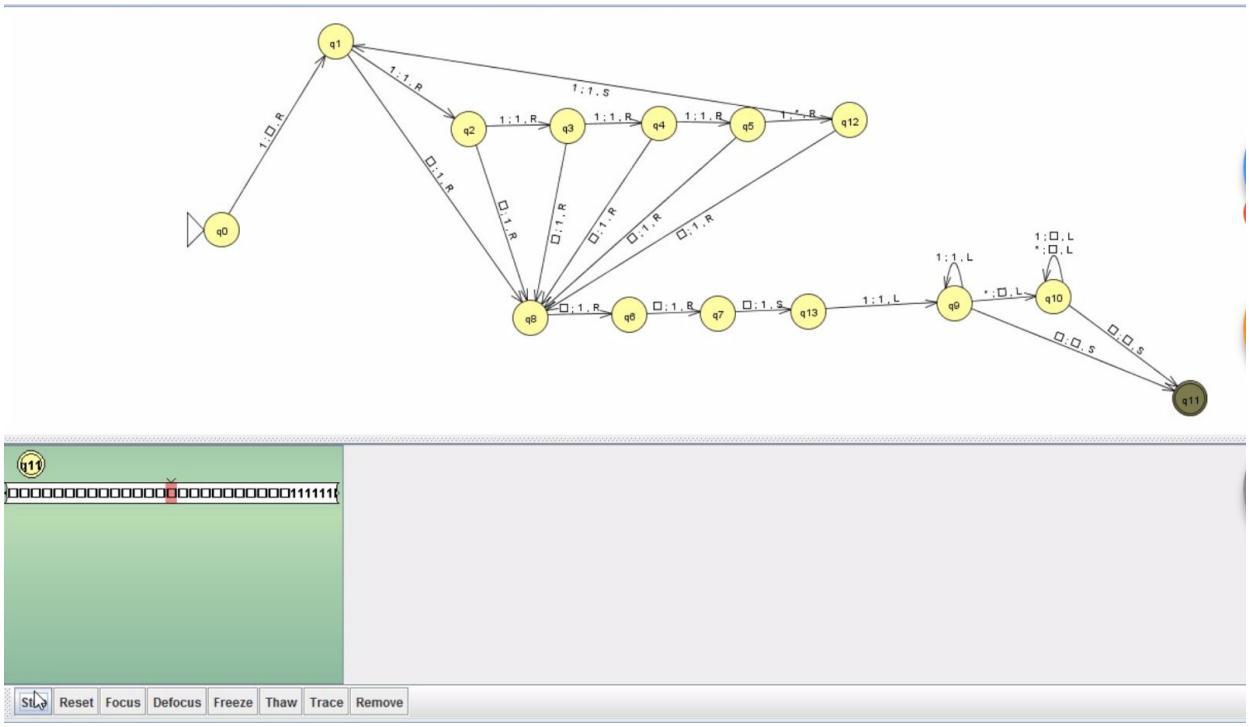


- We divided the problem into two separate pieces. One piece that gives us n%5 and the other that adds 3.
- In order to satisfy the mod 5, we needed to identify the remainder, and erase all the other numbers. In order to do so, we iterated through the

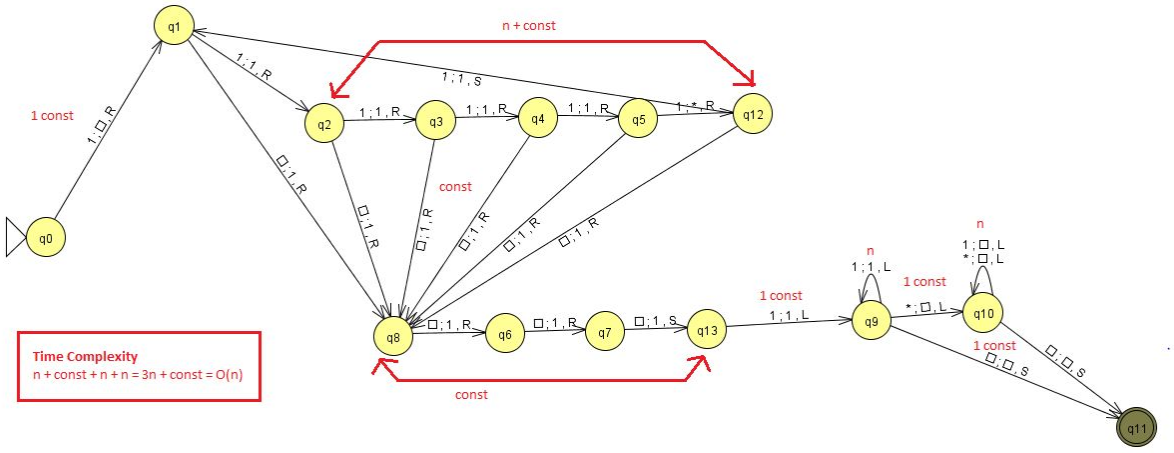
tape to the right, and for each 5th 1 symbol, we put an * in place. Therefore, any 1's after the last * will be our answer for the n%5.

- While it iterates through the strip, Q1, Q2, Q3, Q4, Q5, Q12, we have escape paths in case we reach the end of the strip denoted by existence of a Blank.
- When the TM reaches the end of the strip, all 5th occurrences of 1's are replaced with *, and the trailing 1's are our remainder. We then do the addition of 3, by moving to the right and replacing the blanks with 1 and also add back the representation 1. At this stage of the program, the answer is all the 1's to the right of the *.
- We then iterate through our strip to the left, and find the first occurence of * (leftmost * in the strip). Once we identify it, we start replacing all symbols to the left as Blanks to clean the tape, until with hit the first initial Blank. Once we've reached the initial blank, our tape is cleaned and the remaining 1's is the answer.

Success State



Time Complexity



1. Given our algorithm divides the problem into two subproblems (mod 5 and add 3), we identified the time complexities for each subproblem first.
2. For the n%5 subproblem, we step through the strip n + some const times for input n.
3. After step 2, iterated through our whole strip once and reached the end in n + const time.
4. After the n%5 subproblem is resolved, we then need to look at the time complexity of the addition.
5. The time complexity of addition is fairly simple with just constant time regardless of n.
6. At this stage, we have a delimiter (*) to our answer, and we need to go back and clean up the strip for all unwanted 1's (which is n - n%5).
7. In order to clean the strip, we move left, and each time we encounter a 1, we replace it with blank, which takes n time complexity.
8. After step 7, we've cleaned our strip, and the remaining 1's on the strip is the answer.
9. To compute time complexity of the whole problem, we add the time complexity of the subproblems, where (n + const) is time complexity for each subproblem.

   (n + const) + (n + const) = 2n + const = O(n)

Space Complexity

1. Let's Assume that for every state in the Beginning,is the space owned at that instant.
2. For the moving operation to the right, we are basically in a while loop meaning no extra space is required at this step and when we exit the while loop, seeing blank and replacing them by 1 requires a new storage (New Space) and this is done for all remaining mod operation case.
3. During Q8 where we are doing our addition step, each addition will add one 1 new Space (3 NS in total). During the last step, each time we move left, we are changing 1 to a blank space and * to a blank space and utilizing the previously Owned Space.
4. Total time Complexity: Is the sum of time complexity at each step.
5. Space Complexity: OS + 1 NS + 1 NS + 1 NS + 1 NS

   ⇒ OS + 4 NS ⇒ n + 4