**Maher Salman**

This report compares two algorithms implemented to perform string matching: one utilizing MCS trees and the other is naive string matching algorithm.

**Algorithms**

**1. MCS-based Algorithm**

**a. Generating Lossy MCSs:**

* **Description:** The algorithm generates binary strings of length N containing at least N\_2​ ones. From these binary strings, it extracts the first Nsovp ones to form the lossy MCSs.
* **Complexity:** The complexity is O(2^N \* N) due to generating all possible binary strings of length N and filtering them.

**b. Building the MCS Tree:**

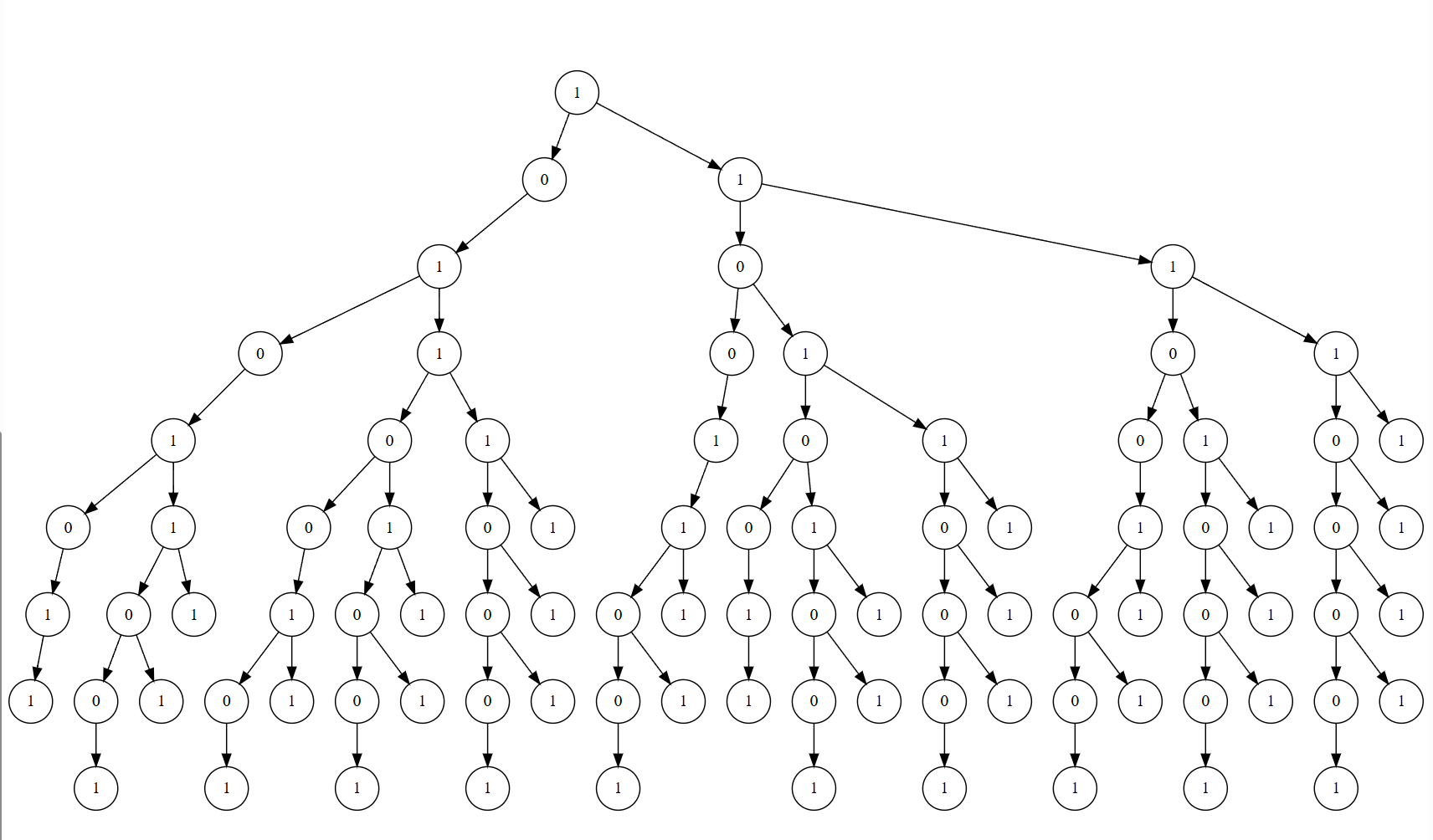
* **Description:** The algorithm inserts each lossy MCS into a binary tree, where '0' and '1' determine the left and right branches, respectively.
* **Complexity:** Inserting each MCS into the tree takes O(N) time, leading to a total complexity of O(L⋅N) . where L is the number of lossy MCSs.

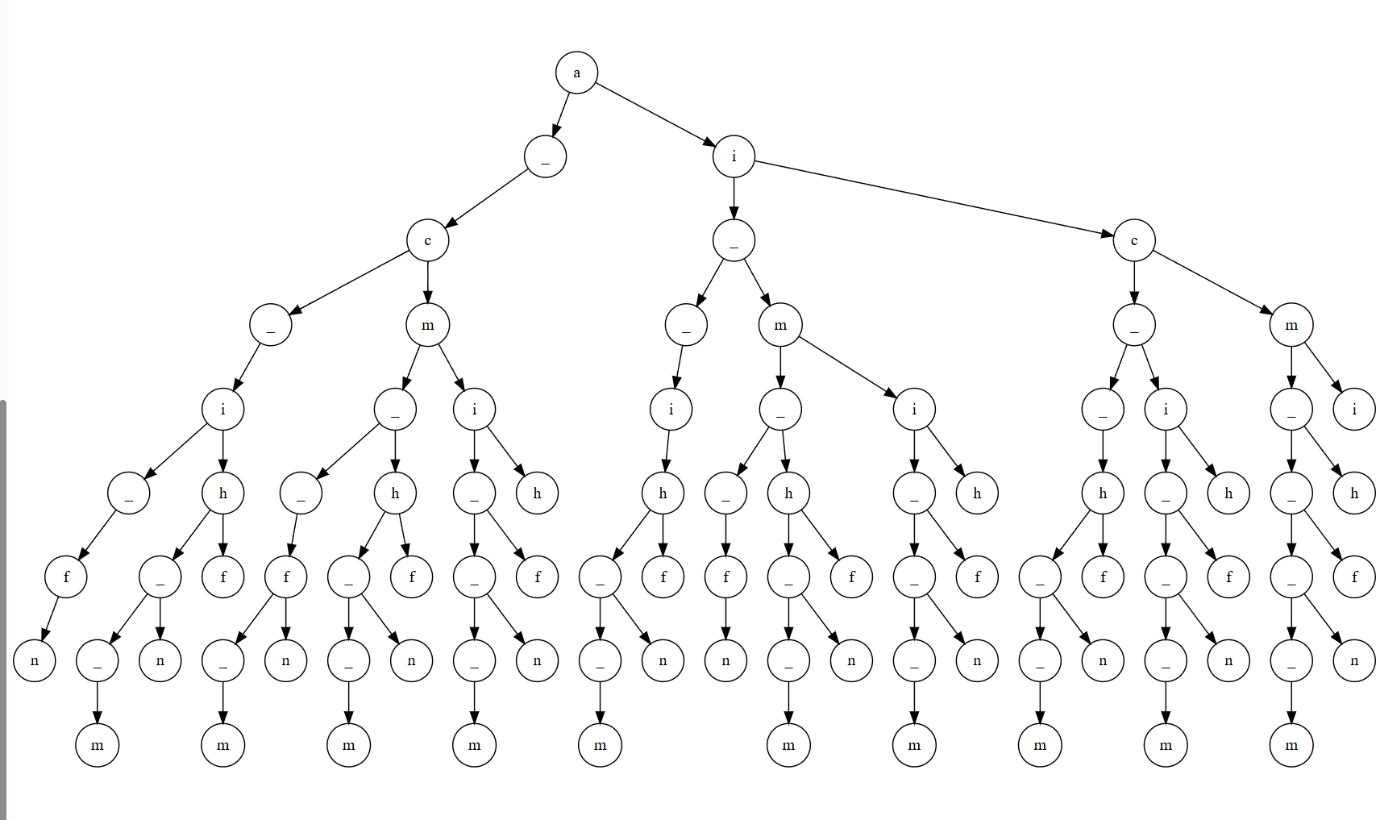
**c. Text Tree Generation:**

* **Description:** For a given text, the algorithm generates a series of trees based on the structure of the MCS tree, where each tree represents a substring of the text.
* **Complexity:** This step involves creating a tree for each substring of length N, resulting in O(T⋅N) complexity, where T is the text length.

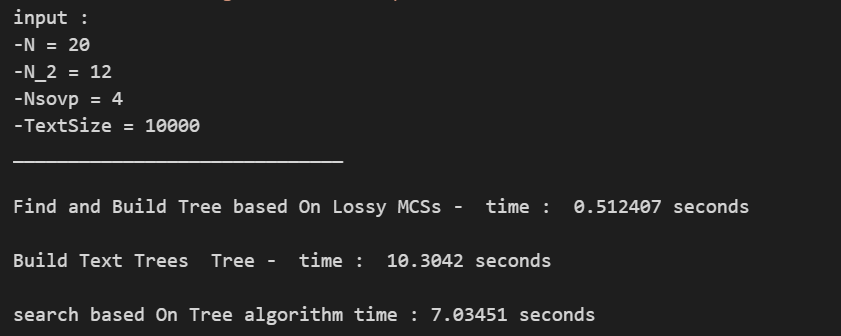
**Our Algorithm :**

**First We Have build our MCS tree , for example i get this :**



Then I build for each char for the text a graph according to the text and mcs tree to be look like 

**Then we make the search on string using the trees.**

****