Home Work 3 Re-Submission

Francisco J. Díaz Riollano
Student ID: 802-15-2172

October 16, 2018

```
procedure NAIVE(On input S[1,..m])
    n ← |T|
    m ← |S|
    for s=0 to n-m do
        if P[1,..m] = T[s+1...s+m] then
            print("pattern found")
```

Question 1

The string matching problem is defined as: "Given a text $T = T_1 ... T_n$ which is stored as array $T = T[1, ..., n]$, and a pattern $P = P_1 ... P_m = P[1...m]$ with $m < n$, where both are strings over the same alphabet Σ; decide whether S is a substring of T.

Algorithm 1 is the so-called naive-pattern finding algorithm. Use Algorithm 1 to construct a Finite State. Automata(deterministic or non-deterministic) for solving the matching problem.

Let M $= (Q, \Sigma, \delta, q_0, F)$ be 5-tuple NFSA where:
Q is a finite set of states, $\{t_1, t_2, ..., t_{|T|}\}$.
Σ is a finite set if input symbols
δ is a state transition function from $Q \times \Sigma \rightarrow Q$
$q_0 \in Q$ is the initial state
$F \subseteq Q$ Q is the set of final states
Let $\Sigma = \{s_1, s_2, ..., s_m\}$ for the alphabet of unique symbols of the text.

High Level Description of the non-deterministic finite state automata:
M =: "On input P[1,...,m]" , where the P is the pattern

1) In the initial state of the automata has empty transitions to every state in the NFSA. The idea behind doing this is that the machine will evaluate every possible subtext. If one of them accepts it means that a pattern has been found.Else the pattern was not found in the text. In essence we are doing an exhaustive search within the automata.

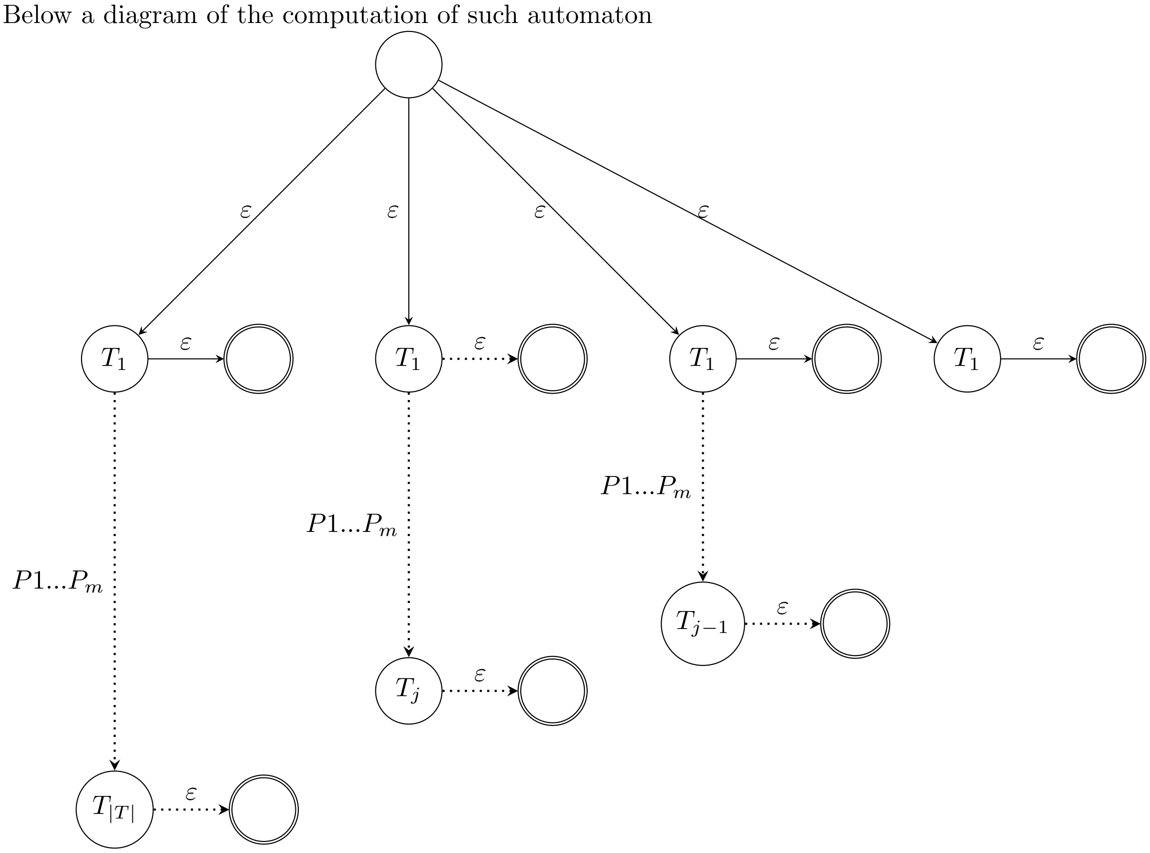
2) Every state in the NFSA will aslo have an empty transition to an accept state.

3) The automata then reads the first symbol, if there is match between the string that was read and the corresponding string symbol for the text, then the automata transitions to the next state and so on. Else this part or in any other state of the automata does not match then it does not proceed to the next state, thus 'rejecting this subtext', it may continue in other parts. 4) The machine will continue to the next state if and only if the two strings match or accepting

if the input string has been read completely

5) If none of the subtexts(with their corresponding states), does not match with the input pattern. Then the machine rejects the string.

Below a diagram of the computation of such automaton



Question 2

Algorithm 1 returns a result in the time proportional to $O(|T||S|)$. Discuss the computation time of your automaton.

In the worst case the automata will have $O(|T||S|)$ complexity. We know this because the tree diagram of the Automata reads at most $|T|$. For each of these readings there are $|S|$ possible states to consider or that it compares.Thus the automata has the same time complexity as the algorithm.

In tree diagram below explores all of the possible text inputs from T_1 all the way to T_n. For the corresponding pattern we induce a empty transition to the acceptance state if the pattern.

We consider every possible comparison up to P_m. Note that the last P_m is not necessarily the only comparison we do of P_m, there may be multiple comparisons in between the tree diagram.