Cpt S 450 Homework #6 Solutions

2.

Procedure Subseq($\mathcal{M}$)

Given a finite automaton $\mathcal{M}$, this procedure is to construct and return a finite automaton $\mathcal{M}'$ that accepts the language of all subsequences γ of all words accepted by $\mathcal{M}$. That is, γ is accepted by $\mathcal{M}'$ iff, for some β accepted by $\mathcal{M}$, γ is a subsequence of β.

$\mathcal{M}'$ is constructed as follows. On an input word γ, $\mathcal{M}'$ runs $\mathcal{M}$ from the initial state in $\mathcal{M}$. Whenever $\mathcal{M}$ reads an input symbol, $\mathcal{M}'$ guesses one of the following two ways to move:

- $\mathcal{M}'$ guesses a symbol and feed this symbol to $\mathcal{M}$ (i.e., $\mathcal{M}$ reads this guessed symbol);
- $\mathcal{M}'$ reads a symbol from γ and feed this symbol to $\mathcal{M}$.

$\mathcal{M}'$ accepts when, at some moment, $\mathcal{M}'$ enters final satte and $\mathcal{M}'$ has read the entire input word γ.

(****) How many states in $\mathcal{M}'$? it is $O(|\mathcal{M}|)$, where $|\mathcal{M}|$ is the number of states in $\mathcal{M}$.

Algorithm to compute the D

We define two finite automata M_1' and M_2' as follows. M_1' accepts the set of all subsequences in all $\alpha \in L_1$. M_2' accepts the set of all subsequences in all $\beta \in L_2$. You need to figure out the following. The D that we are looking for is exactly the maximal length of all words γ that is accepted by M_1' and is accepted by M_2'.

Hence, the algorithm of computing the D has the following four steps:

step1. Construct M_1'. We let M_1 be a finite automaton that accepts the language L_1. Then, M_1' is constructed as Subseq(M_1).

step2. Construct M_2'. We let M_2 be a finite automaton that accepts the language L_2. Then, M_2' is constructed as Subseq(M_2).

step3. Construct a finite automaton M that accepts the intersection of the language accepted by M_1' and the language accepted by M_2'. This is standard in cs317.

step4. Now, we eliminate all the Λ-transitions in M (using a standard algorithm in cs317). There are two cases to consider: 1. there is a cycle on a path from the initial state to the final state in the graph of M. In this case,

M accepts an infinite language. So, the $D = \infty$. (How to detect whether there is such loop? figure it out or Google) 2. there is no such cycle (i.e., the transition graph of M is a DAG). In this case, M accepts only a finite language. The D is exactly the length of a longest word accepted by M, or the length of a longest path in the DAG. Using a breadth first search, you may find the length of a longest path. Try Google(DAG, longest path).

1. We use the solutions to the previous problem.

Let M_α (resp. M_β) denote an NFA that accepts the language that contains word α (resp. β) only. The number of the states in M_α (resp. M_β) is bounded by the length of α (resp. β), which is denoted by n (resp. m)..

We then construct M_1 (resp. M_2) as Subseq(M_α) (resp. Subseq(M_β)). That is, M_1 (resp. M_2) accepts the set of all subsequences in α (resp. β). Notice that the state number in M_1 (resp. M_2) is $O(n)$ (resp. $O(m)$).

Let M_{12} be an NFA that accepts "all strings that do not contain abb". We use k to denote the state number in M_{12}.

We now, using the standard cartesian product construction, to construct an NFA M (by intersectiing M_1, M_2, and M_{12}) to accepts the language $L(M_1) \cap L(M_{12}) \cap L(M_2)$. How many states in M? bounded by $O(nmk)$.

M accepts a finite language. Now, in order to answer the question, one need only find the length of a longest path from the initial state to a final state in M (its transition graph represented as a DAG). Complexity? $O(n^2m^2k^2)$.