CS 208 : Automata Theory and Logic March 17, 2024

Lecture - 26

Topic: PDA expressive power

Scribed by: Busa S N V Aditya (22B1024)
Checked and compiled by:

1 Section title

In previous class, we have discussed that for a Push Down Automata [PDA] which accepts by "Final state" , we can show a PDA which accepts the same language, but by "empty stack". Now we will try to show the converse (i.e. For a Push Down Automata [PDA] which accepts by "empty stack" , we can show a PDA which accepts the same language, but by "Final states")

Proof :

Similar to the previous proof where we used an alphabet not present in the language to ensure the stack doesn't empty before we reach Final states, here also we use such an auxiliary alphabet 'X'.

We have two things to take care of, to ensure correctness of our proof :

1. Whenever stack gets emptied, we have to redirect that into an Final state.
2. The Final state should not be achieved by any word which doesn't empty the stack.

Now we do changes to existing Automata

1. Place an alphabet (say 'X') (which doesn't appear in the language) at bottom of the stack (even below initial alphabet)
2. Place an edge of the format $(\epsilon, X|X)$ from all the nodes in the existing automata to a new node which we define (this is also going to be the "Final State")

With the above modifications, we can be sure that whenever stack gets emptied (Not in real sense, because we will still have X at the bottom) it gets redirected to Final state and if it is not empty then X won't be seen at the top of the stack (since it is not present in the language of the original automata)

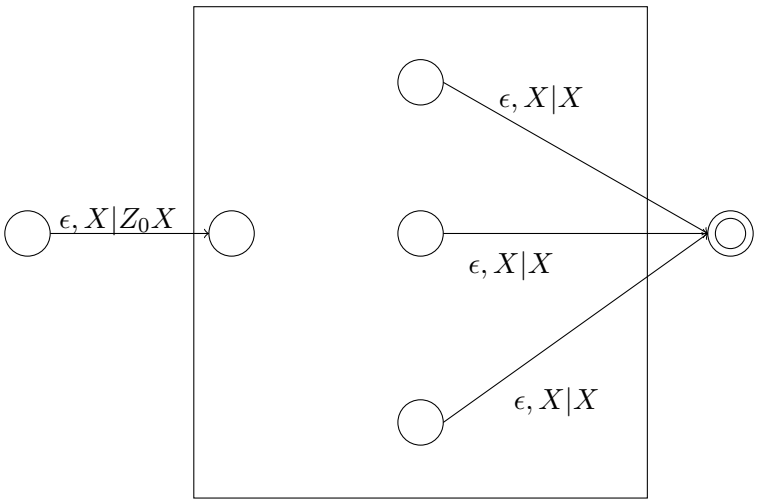

Figure 1: Converting Acceptance by empty stack to Acceptance by Final state

2 Non-Deterministic vs. Deterministic

Terminology : Languages accepted by Push Down Automata are called *Context-free languages* (we shall come back to why it is called this name)

We have shown that DFA,NFA and ϵ-NFA are all equally expressive(i.e. every language which can be accepted by some DFA, an NFA and ϵ-NFA can be constructed and similarly other way rounds) and we have called such languages regular languages.

So we wish to investigate the same here,

By definition, every language accepted by DPDA is going to be accepted by NPDA(i.e. DPDA $\subseteq$ NPDA). So we just want to see is this subset relation an equality one or is it a strict subset.

Although the proper mathematical proof is rigourous enough to be excluded from here. We shall intuitively try to understand why DPDA is a strict subset of NPDA (i.e. there can be a language which is accepted by NPDA and there doesn't exist any DPDA for it) Consider a language L,

$$L = \{0^n 1^{i+j} 0^k \mid (n < i), (j < k)\} \cup \{0^n 1^{i+j} 0^k \mid (n > i), (j > k)\}$$

Say the former language is L_1 and the latter is L_2

The DPDA for L_1 could be "just j = 0 and maximize i" because whatever maybe j , i
[j = 0 and i = i+j] is also an accepted word

The DPDA for L_2 could be "just i = 0 and maximize j" because whatever maybe i , j
[i = 0 and j = i+j] is also an accepted word

But in case of their union, We can not predict when to switch from i to j, but through NPDA , we can always use ϵ edges to jump from L_1 to L_2 anytime and check acceptancy.

3 Context Free Languages

Consider a word of length n, and let's say at two positions i and j (i $<$ j) the stack size is same (say size is k) , and also between i and j the stack size never dropped below k, By these two conditions we can make a statement,

The word between positions i and j doesn't depend on the part of stack below size k (i.e. **"Context Free"**), We can also replace the part of word between i and j with some other word which also will be processed in the same way as current word(i.e. leaves the stack the same as it got)