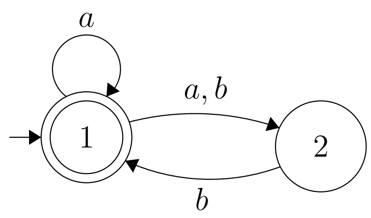
CITS2211 Discrete Structures
Week 11 Exercises – Regular expressions and regular languages & PDAs

2022

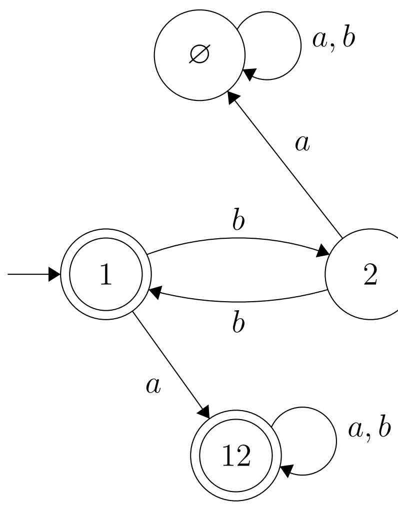
Topics: Regular Expressions, Regular Languages, The pumping lemma for regular languages, Context-free languages, Context-free grammars, Push-down automata

1 FSM Revision

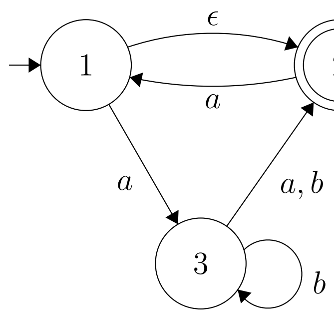
1. *[Source: Sipser 1.16]* Convert the following nondeterministic FSM (NFSM) to an equivalent deterministic finite automata (DFSM).



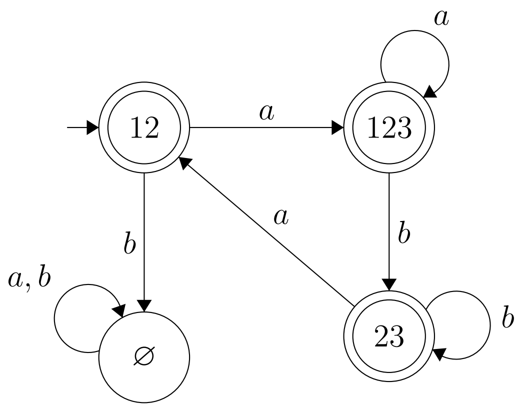
SOLUTION:



2. *[Source: Sipser 1.16]* Convert the following nondeterministic FSM (NFSM) to an equivalent deterministic finite automata (DFSM).



Solution:



How does this work?

When considering the starting state, note that since 1 can reach 1 or 2 by epsilon only, the initial state is given as the set state 1,2 (we lose the original start state 1). Neither 2 or 3 have any epsilon transitions leaving them. Technically this is called epsilon closure of the states. Here is the state transition table starting from the starting state 1,2.

Solution: (solution, cont'd)

START	INP	END	COMMENT
1,2	a	1,2,3	
1,2	b	∅	neither 1 or 2 have a transition for b in the NFSM
1,2,3	a	1,2,3	1a3, 3a2, 2a1, 2ae2 in the NFSM
1,2,3	b	2,3	1 and 2 have no b transition but 3b2 and 3b3 in the NFSM
2,3	a	1,2	2a1, 2ae2, 3a2
2,3	b	2,3	2 has no b, but 3b3 and 3b2
∅	a,b	∅	This is the non-accept state with no way out

Note that any set of states with a 2 in it is an accepting state of the DFSM because 2 is accepting in the NFSM. So 1,2 1,2,3 and 2,3 are all accepting states.

2 Regular expressions and languages

1. Prove that if A is a regular set with alphabet I, then the language defined by taking the set difference $I^* - A$ is also regular.

SOLUTION:

$I^* - A$ is the set of all possible words from the alphabet except those strings in A.

Suppose we have a DFSM M that recognises A. So it has accepting states for every string in A. (We include all the error states in M.)

Now create an FSM P with the same structure as M but with the accepting and non accepting states swapped - That is, $F_P = Q - F_M$.
(F_P is the set of all accepting states of P, accordingly F_M is the set of all accepting states of M, and Q is the set of all states.)

The new FSM P will recognise $I^* - A$. Therefore, since we have created a FSM to recognise the language $I^* - A$, that language is regular. QED

2. Simplify the following regular expression as much as possible

$$(((\mathtt{a}^*)^*)^*)^*(\epsilon + \mathtt{b})\mathtt{c}(\mathtt{c} + (\epsilon + \epsilon))^*$$

Explain your reasons for each simplification step.

SOLUTION:

$(((a^*)^*)^*)^*$ is a^{****} which is the same as a^* (that is, a repeated zero or more times).

$(\epsilon + \epsilon)$ is the same as ϵ: that is, the empty symbol.

So the regular expression simplifies to $a^*(\epsilon + b)c(c + \epsilon)^*$

And $(c + \epsilon)^*$ is the same as c^*, so it simplifies further to: $a^*(\epsilon + b)cc^*$.

Note that cc^* generates one or more cs while c^* generates zero or more, so you can't simplify the two cs any more.

Finally we have the simplified expression:

$a^*(b + \epsilon)cc^*$

3 PDAs

1. State the Pumping Lemma for Regular Languages. Write out your answer in a way that helps you to remember the lemma.

SOLUTION:

Theorem: If L is a regular language then

$\exists$ an integer p called the *pumping length* of L such that
$\forall$ words $w \in L$ where $|w| > p$
$\exists$ an expression $w = xyz$ where

(a) $\forall\ i \geq 0.\ xy^iz \in L$
(b) $|y| \geq 1$
(c) $|xy| \leq p$

Note the $i \geq 1$ version is OK too.

2. Use the Pumping Lemma for Regular Languages to prove that the language of all binary strings that have equal numbers of 0s and 1s is *not* regular.

SOLUTION:

Suppose L is regular.

Let the pumping length be p.

Choose $w = 0^p1^p$. Clearly $w \in L$ because it has p 0s and p 1s.

For any adversary choice of $xyz = w$, both x and y can only contain 0s since $|xy| \leq p$ (constraint (c)).

Let $x = 0^m$, $y = 0^n$ for some $m + n \leq p$.

The pumping lemma states that $xyyz \in L$ but $xyyz \notin L$ because $xyyz$ contains more 0s than 1s or if $i = 0$ then xz contains fewer 0s than 1s.

We have derived a contradiction. Therefore L is not regular. QED

3. Describe the error in the following "proof" that 0^*1^* is *not* a regular language. Note that there is an error because 0^*1^* *is* a regular language.

The proof is by contradiction. Assume that $L = 0^*1^*$ is regular and p is the pumping length for L given by the pumping lemma. Choose w to be the string 0^p1^p. You know that $w \in L$ but w can not be pumped, since any $xyyz$ will have more 0s than 1s. Thus you have a contradiction so 0^*1^* is not regular.

SOLUTION: The error is in the statement *w can not be pumped, since any xyyz will have more 0s than 1s.*

The chosen string 0^p1^p has equal numbers of 0s and 1s and is in the language. If pumped then it loses the equal number of 0 and 1 property, *but* the resulting string *is* in the language L because it still has all 0s before 1s.

We need to find a string that *can not* be pumped in order to show L is not regular.

4. For the language $L = \{a^ib^jc^k \mid i, j, k \geq 0 \wedge (i = 1 \rightarrow j = k)\}$

   (a) show that L is not regular
   Hint: Try to use Kleene's theorem and the pigeonhole principle instead of the pumping lemma in this case.

   (b) show that $w = a^ib^jc^k$ satisfies the pumping lemma conditions (for some i,j,k).
   Challenge: You can show that all words $w = a^ib^jc^k \in L$ with $|w| > 2$ satisfy the pumping lemma conditions.

(c) explain why parts a) and b) do not contradict the pumping lemma

SOLUTION: a) We can show that L is not regular by using an idea from the proof of Kleene's theorem:

Assume L is regular and so, by Kleene's Theorem there is an FSM which recognizes L. Say that M has p states. Now consider M with inputs

$$a, ab, abb, ..., ab^i, ..., ab^p.$$

As M only has p states the pigeonhole principle implies that at least two of these strings leave M in the same state. Say ab^v and ab^w for $v < w$ being the first two. Now ab^vc^v is accepted so also ab^wc^v will be. Contradiction.

b) We are asked to show that L satisfies the pumping lemma conditions for some words w. For example, let p be the pumping length and $w = a^pb^jc^k$. Then for any xyz we have x and y contain only as and that $xy^mz \in L$ for any $m \geq 0$ since the string still separates all its as, bs and cs.

Additional notes:
Even though we were not asked to do this in the question, we can actually show that all conditions of the pumping lemma hold:
Let $p = 2$ be the pumping length. We now need to prove that all words $w \in L$ of length at least 3 ($|w| > p$) can be pumped. So let w be an arbitrary word from the language L with at least three symbols. There are three cases to consider:

Case 1: $w = a^1b^jc^k$ (so $i = 1$).
We know that in this case there are equally many bs and cs ($j = k$). We can choose $x = \epsilon$, $y = a$ and $z = b^kc^k$ such that $w = xyz$. For this representation of w the three conditions from the pumping lemma hold:

- $|y| = |a| = 1 \geq 1$,
- $|xy| = |a| = 1 \leq p$,
- we can repeat the 'a' zero or more times and the string would still be in the language, i.e. $\forall\, n \geq 0.\ xy^nz \in L$.

Case 2: $w = a^2b^jc^k$ (so $i = 2$).
In this case we cannot assume equally many bs and cs but we can choose $x = \epsilon$, $y = aa$ and $z = b^jc^k$ to cover the rest of the word such that $w = xyz$. For this representation of w the three conditions from the pumping lemma hold:

- $|y| = |aa| = 2 \geq 1$,
- $|xy| = |aa| = 2 \leq p$,
- we can repeat the 'aa' zero or more times and the string still separates all its as, bs and cs. Additionally, there cannot be a single 'a' at the start of the string as we would always get an even number of 'a's. It holds that $\forall\, n \geq 0.\ xy^nz \in L$.

Case 3: $w = a^ib^jc^k$ with $i \neq 1$ and $i \neq 2$.
Again, we cannot assume equally many bs and cs but we know there must be none or at least three as. We choose $x = \epsilon$, and y to be the first symbol of the word, and z to cover the rest of the word such that $w = xyz$. For this representation of w the three conditions from the pumping lemma hold:

- $|y| = 1 \geq 1$,
- $|xy| = |y| = 1 \leq p$,
- since y only contains one symbol, we can repeat 'y' zero or more times and the string still separates all its as, bs and cs. Additionally there cannot be a single 'a' at the start of the string since either the first symbol was not an $a4$, which would mean there were no as or there were three as, then $y = a$ followed by two as (this is because $i \neq 1$ and $i \neq 2$). So $\forall\ n \geq 0.\ xy^n z \in L$.

c) There is no contradiction because pumping lemma is of the form $R \rightarrow P$ (i.e. if language is regular, then all words of the language can be pumped) but if $R = false$ then we can't say anything about the truth of P.

Note that if we only show that some of the words 'can be pumped', then we have not even shown that the conditions of the pumping lemma hold.

5. Describe a grammar that generates all binary strings that have equal numbers of 0s and 1s.

SOLUTION:

This is version of the balanced brackets language. So a possible grammar is

$$S \rightarrow \epsilon \mid 0S1 \mid 1S0 \mid SS$$

6. Design a pushdown automata (PDA) and draw the state machine diagram for the language of all binary strings that have equal numbers of 0s and 1s.

SOLUTION:

Strategy:

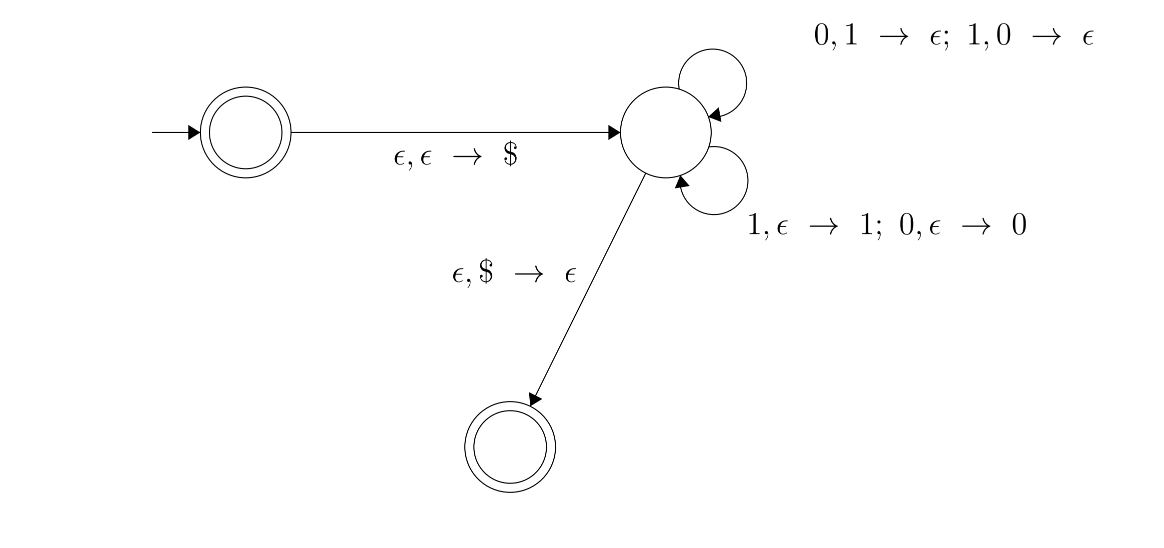
Put the \$ symbol on the stack.

If input 1 when top of stack is 0 then we have a match, so pop the 0.
If input 0 when top of stack is 1 then we have a match, so pop the 1.

Non-deterministically, if input 0 then push 0 onto the stack guessing that the top of the stack is also 0.
Non-deterministically, if input 1 then push 1 onto the stack guessing that the top of the stack is also 1.

If the stack is finished (\$) and there is no more input then accept the string.

Pushdown automata:



7. Define a grammar that generates all binary strings with more 0s than 1s.

SOLUTION:

Idea: start with a 0 and then build around it. The scaffolding around the initial 0 has either equal 0s and 1s or just 0, so there will always be at least one more 0 than 1s.

Grammar:

$$S \to A0A$$

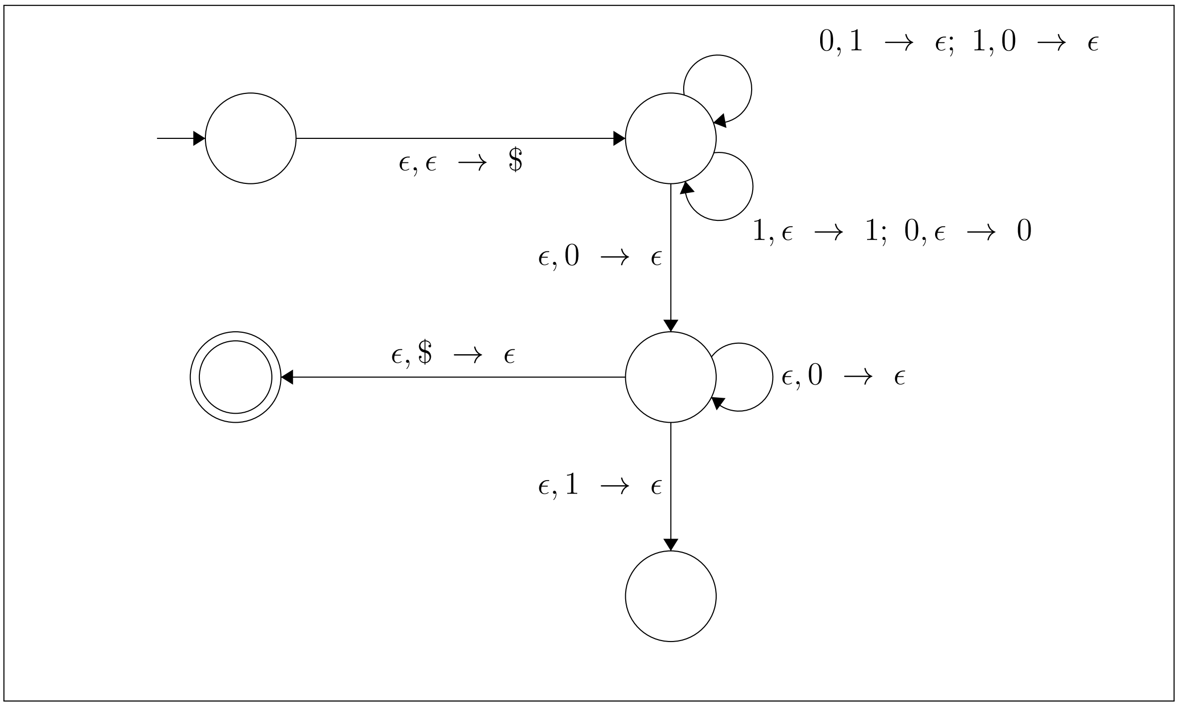
$$A \to AA \mid 1A0 \mid 0A1 \mid 0 \mid \epsilon$$

8. Design a pushdown automata (PDA) and draw the state machine diagram for the language of all binary strings with more 0s than 1s.

SOLUTION:

Strategy:

This is similar to the equal number of 0s and 1s machine in the last question. But now, guess when you reach the end of the inputs and move to a state for checking that there are only 0s left on the stack by popping them all off.


$0, 1 \rightarrow \epsilon;\ 1, 0 \rightarrow \epsilon$
$\epsilon, \epsilon \rightarrow \$$
$1, \epsilon \rightarrow 1;\ 0, \epsilon \rightarrow 0$
$\epsilon, 0 \rightarrow \epsilon$
$\epsilon, \$ \rightarrow \epsilon$
$\epsilon, 0 \rightarrow \epsilon$
$\epsilon, 1 \rightarrow \epsilon$