

Homework 5

2.1c

Parse trees are hard to draw, but here's a derivation:

$$E \Rightarrow E+T \Rightarrow E+T+T \Rightarrow T+T+T \Rightarrow F+T+T \Rightarrow F+F+T \Rightarrow F+F+F \Rightarrow a+F+F \Rightarrow a+a+F \Rightarrow a+a+a$$

bcfgjkm

b ab, ba, aab

c a, b, aa

f false — there is no way to get from T to T by applying a single rule of the grammar

g true — you can get from T to T in 0 steps

j true — $T \Rightarrow XTX \Rightarrow XX$

k true — $T \Rightarrow XTX \Rightarrow XXX$

m all strings that don't begin and end with the same letter

2.4adf

a

$$S \rightarrow A1A1A1A \\ A \rightarrow AA \mid \epsilon \mid 0 \mid 1$$

d

$$S \rightarrow XSX \mid 0 \\ X \rightarrow 0 \mid 1$$

f

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

2.9

$$S \rightarrow MC \mid AN \\ A \rightarrow AA \mid \epsilon \mid a \\ C \rightarrow CC \mid \epsilon \mid a$$

$M \rightarrow aMb \mid \epsilon$
 $N \rightarrow bNc \mid \epsilon$

2.14

(We did this in class.)

Homework 4

Show that $L = \{w \mid w \text{ contains the same number of 0s and 1s}\}$ is nonregular.

Assume L is regular. By the Pumping Lemma, there exists a p so that for every $s \in L$ with length at least p , [the properties of the Pumping Lemma hold].

So consider $s = 0^p 1^p$. The only way of writing $s = xyz$ in accordance with the Pumping Lemma is if y contains only 0s (by property 3). But if we pump any such y up or down, the resulting string will have more/less 0s than 1s and hence will not be in L . This violates the Pumping Lemma and hence is a contradiction. Thus L cannot be regular.

Show that $L = \{0^{i^2} \mid i \geq 0\}$ is nonregular.

Assume L is regular. By the Pumping Lemma, there exists a p so that for every $s \in L$ with length at least p , [the properties of the Pumping Lemma hold].

So, consider $s = 0^{p^2}$. This is clearly in L with length at least p .

Recall that perfect squares are the sums of odd numbers (that is, $1 + 3 = 2^2$, $1 + 3 + 5 = 3^2$, $1 + 3 + 5 + 7 = 4^2$, and so on). In general, $(n + 1)^2 = n^2 + (2n + 1)$; that is, the distance from n^2 to the next greater square is $2n + 1$.

Return to our $s = 0^{p^2}$. If we write this as some xyz , y is a sequence of some 0s, say m 0s — where m is no more than p . If we now consider xy^2z , we have a string with length $p^2 + m$. Is this string in L ? It can't possibly be: the next perfect square after p^2 is $p^2 + (2p + 1)$, but m is much smaller than $2p + 1$. Thus $xy^2z \notin L$, violating the Pumping Lemma and yielding a contradiction.

Show that $L = \{w \mid w = w^R\}$.

Assume L is regular. By the Pumping Lemma, there exists a p so that for every $s \in L$ with length at least p , [the properties of the Pumping Lemma hold].

So consider $s = 0^p 10^p$. The only way of writing $s = xyz$ in accordance with the Pumping Lemma is if y contains only 0s from the first part of the string (by property 3). But if we pump any such y up or down, the resulting string will have more/less 0s in the first half than in the second half and hence will not be in L . This violates the Pumping Lemma and thus is a contradiction. Therefore L must be non-regular.

Extra Credit: Prove that if

1. $L = L_1 \cup L_2$,
2. L is non-regular, and
3. L_1 is regular,

then L_2 is nonregular.

This is a pretty straightforward proof by contradiction. Let's assume (towards an eventual contradiction) that L_2 is regular. Then what do we know about $L_1 \cup L_2$? Union is a regular operation, and regular languages are closed under the regular operations, so $L_1 \cup L_2$ must also be regular. But this contradicts the given fact that $L = L_1 \cup L_2$ is non-regular. Therefore our assumption that L_2 is regular must be wrong: L_2 is non-regular.

Homework 3

1.9 Prove that every NFA can be converted to an equivalent one that has a single accept state.

I'll just informally describe the idea here; to do this properly, the informal description should be accompanied by precise descriptions of how the 5-tuple of the NFA should be modified.

The idea is just this: take an NFA M and add a new accept state q_{new} . Then go through every accept state of M (other than q_{new}) and (1) make it non-accepting, and (2) add an ϵ -transition from that state to q_{new} . Thus, every string that could have been accepted by M can also be accepted by the new machine. Note that if M has no accept states, then there will be no transitions coming into q_{new} — it'll kinda be floating — but it will technically still be part of the new machine.

1.12b Convert NFA to equivalent DFA.

I'm not gonna draw this out online— too annoying. But it's exactly like Example 2.1; the only thing you have to be a little careful of is properly taking into account the ϵ -transition from state 1 to state 2 (but the books shows how to do this in the example).

1.13abei Give regular expressions...

a $1\Sigma^*0$

b $\Sigma^*1\Sigma^*1\Sigma^*1\Sigma^*$

e $0(\Sigma\Sigma)^* \cup 1\Sigma(\Sigma\Sigma)^*$

i $1(\Sigma 1)^* \cup (\Sigma 1)^*$

1.15bdf For each of the following languages...

b In: ab, abab. Out: abba, babab.

d In: ϵ , aaa. Out: a, aa.

f In: aba, bab. Out: ababab, abaaba.

1.16a Convert DFA to regular expression.

I'm not gonna do this one either; it's also hard to draw. But it's almost exactly like Example 1.35....

Homework 2

These are really hard to draw (still). And showing you my correct solution may not help much in determining if your solution is correct... I'll describe the idea of the solutions to 1.5, though...

1.5abd

a initial state q_0 loops on 0 and 1. there's a transition on 0 to q_1 , and another transition on 1 from q_1 to q_2 . q_2 is the only accept state.

b initial state q_0 loops on 0 and 1. then a sequence of states q_1 o q_4 with transitions from one to the next on 0,1,0,1. loop on 0 and 1 at q_4 , which is the only accept state.

d single transition on 0 from q_0 (initial state) to q_1 , which is the only accept state. no other transitions.