COMS 461 - Midterm 2 Review

1. Create a context free grammar that generates the following language.

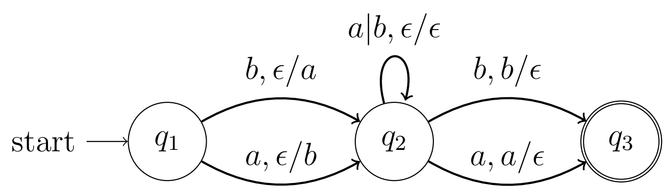
$$L = \{w \in \{a,b\}^* : w \text{ starts and ends with different symbols}\}.$$

> **Solution:**
>
> $$S \to aXb \mid bXa$$
> $$X \to aX \mid bX \mid \epsilon$$

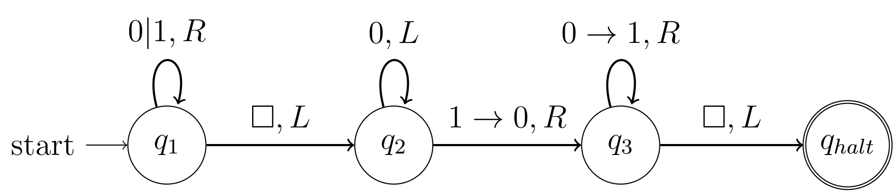
2. Draw a state diagram for a nondeterministic pushdown automata (NPDA) that recognizes

$$L = \{w \in \{a,b\}^* : w \text{ starts and ends with different symbols}\}.$$

> **Solution:**



3. Consider the Turing machine with state diagram shown below.



(a) What will this TM output if the input tape initially contains the string `101000` and the head is initially pointing at the left-most digit? For your answer, write down the final tape contents and indicate the position of the head when the TM halts.

> **Solution:** The tape contains `100111` with head pointed at the rightmost 1.

(b) This TM corresponds to a simple function on binary numbers. What is that function?

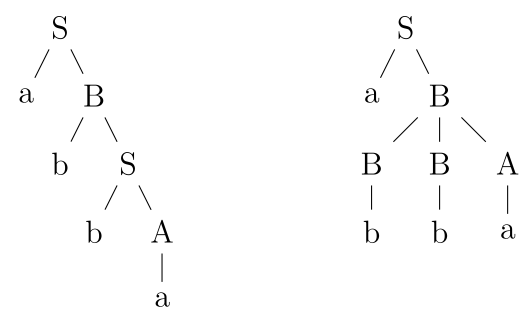
> **Solution:** Binary decrement by 1 function.

4. Consider the context free grammar below.

$$\begin{aligned} S &\to aB \mid bA \\ A &\to a \mid aS \mid AAB \\ B &\to b \mid bS \mid BBA \end{aligned}$$

This grammar generates the language of all strings in $\{a,b\}^*$ with an equal number of a's and b's. Prove that this grammar is ambiguous by finding two different derivations of the string $abba$. Draw parse trees for the two different derivations that makes it clear that they are different.

Solution:



5. The following statements are all false. For each one, explain why it is false.

(a) The is an uncountable number of Turing machines that can be defined with a given input alphabet Σ and tape alphabet Γ.

Solution: The number of Turing machines is countable, since for every number of states $|Q|$, there is only a finite number of transition functions

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$$

(b) For any function $f : \{0,1\}^* \to \{0,1\}$, you can always find a Turing machine that accepts a string $w \in \{0,1\}^*$ if and only if $f(w) = 1$, but the Turing machine might loop forever on w if $f(w) = 0$.

Solution: Let $L = \{w \in \{0,1\}^* : f(w) = 1\}$. Then L is a language which might not be Turing computable. If it is not Turing computable, then there won't be any Turing machine that accepts it, let alone decides it.

(c) All Turing decidable languages are regular.

Solution: There are lots of examples of Turing decidable languages that are not regular. Here is one:

$$L = \{a^n b^n : n \in \mathbb{N}\}.$$

It's very easy to make a TM that decides this language, but it is not regular by the pumping lemma.

6. Let

$$L_1 = \{a^n\, b\, a^m\, b\, a^n \;:\; m, n \in \mathbb{N}\}$$

and let

$$L_2 = \{a^n\, b\, a^n\, b\, a^n \;:\; n \in \mathbb{N}\}.$$

(a) Prove that L_1 is context free.

Solution: The following CFG generates L_1:

$$\begin{aligned} S &\to aSa \mid B \\ B &\to bAb \\ A &\to aA \mid \epsilon \end{aligned}$$

(b) Use the pumping lemma to show that L_2 is not context free.

Solution: Suppose that L_2 is context free. Then it has a pumping length p, and any string in L_2 with more than p symbols contains substrings wxy such that at least one of w and y is not empty, $|wxy| \leq p$, and w and y can be pumped. Consider the string $s = a^p b a^p b a^p \in L_2$. Since every string in L_2 contains exactly two b's, we know that neither of the pumping substrings w and y can contain a b. Also, since the length of wxy is at most p, it can only contain a's from at most two of the three a^p substrings. Therefore, if you try to pump w and y, the three a-portions of the string separated by the b's will no longer have the same length, which contradicts the pumping lemma for context-free languages. So we conclude that L_2 is not context-free. $\square$

7. Describe a Turing machine that accepts the language

$$L = \{w \in \{a, b\}^* : w \text{ has a different number of a's and b's}\}.$$

Hint: Use a 2-tape Turing machine.

Solution: Here is one way to do it. Let the first tape store the input w and the second tape has a single # symbol at the start position. We'll read w from left to right.

Step 1. Read the first character of w, if it is a, move one space to the right on tape 2 and if it is a b, then move one space to the left on tape 2.

Step 2. Repeat until you reach a blank on tape 1. Then accept w if the current position on tape 2 is not #, otherwise reject.