# Homework 5: Non-CFLs and Turing Machines

Theory of Computing (CSE 30151), Spring 2024

Due: 2024-03-22 5pm

## Instructions

- Create a PDF file (or files) containing your solutions. You can write your solutions by hand, but please scan them into a PDF.
- Please name your PDF file(s) as follows to ensure that the graders give you credit for all of your work:
  - If you're making a complete submission, name it `netid-hw5.pdf`, where *netid* is replaced with your NetID.
  - If you're submitting some problems now and want to submit other problems later, name it `netid-hw5-part123.pdf`, where `123` is replaced with the problem number(s) you are submitting at this time.
- Submit your PDF file(s) in Canvas.

## Problems (10 points each)

1. **Non-closure properties of CFLs**

   (a) [Exercise 2.2a] Use the languages

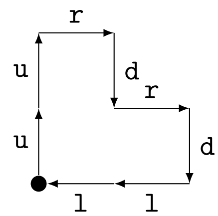
   $$A = \{\mathtt{a}^m\mathtt{b}^n\mathtt{c}^n \mid m, n \geq 0\}$$
   $$B = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^m \mid m, n \geq 0\}$$

   to prove that context-free languages are *not* closed under intersection.

   (b) [Exercise 2.2b] Use Problem 1a and DeMorgan's law to prove that context-free languages are *not* closed under complementation.

2. **There and back again.** Imagine a robot turtle that you can give instructions `u` (go up 1 cm), `d` (go down 1 cm), `l` (go left 1 cm), `r` (go right 1 cm). A program is a string of instructions.

   Let $C$ be the set of programs that make the turtle return to its starting point. For example, `uurdrdll` is in $C$, as shown in this picture:




(a) Prove that $C$ is not context-free.

(b) Write a **formal description** of a Turing machine that decides $C$.

3. **Turing closure properties.** Let $\Sigma = \{\texttt{0}, \texttt{1}\}$. Recall in HW2 we defined

$$\text{STRETCH}(w_1 w_2 \cdots w_n) = w_1 w_1 w_2 w_2 \cdots w_{n-1} w_{n-1} w_n w_n$$

for any string $w = w_1 \cdots w_n \in \Sigma^*$. This induces an operation on languages,

$$\text{STRETCH}(L) = \{\text{STRETCH}(w) \mid w \in L\}.$$

(a) Write an **implementation-level** description of a Turing machine $S$ that, on input $v \in \Sigma^*$, decides whether $v = \text{STRETCH}(u)$ for some $u$. Moreover, if $S$ accepts $v$, then when it halts, the contents of the tape should be $u$. For example, if the input is `001100`, $S$ should accept and the final contents of the tape should be `010`. But if the input is `001101`, $S$ should reject.

(b) Prove that if $L$ is a Turing-decidable language over $\Sigma$, then STRETCH($L$) is also Turing-decidable. You should let $M$ be a Turing machine that decides $L$, then use your answer to 3a to give an **implementation-level** description of a Turing machine that decides STRETCH($L$). One of the lines of your description can be "Simulate $M$."

(c) Prove that if $L$ is a Turing-recognizable language, then STRETCH($L$) is also Turing-recognizable.