Collaborating on homework is encouraged, but you must write your own solutions in your own words and list your collaborators. Copying someone else's solution will be considered plagiarism and may result in failing the whole course.

Please answer clearly and concisely. Explain your answers. Unexplained answers will get lower scores or even no credits.

(1) (30 points) Consider the following context-free grammar G:

$$T \to \texttt{()} \mid \texttt{(}T\texttt{\#)} \mid \texttt{(\#}T\texttt{)}$$

It generates expressions like `()`, `(()#)`, `(#(()#))` and so on.

  (a) Every partially completed rule of the form $A \to \alpha \bullet \beta$ is known as an *item*. Write all items in the grammar G and construct an NFA for the valid item updates.
  (b) Convert the NFA to a DFA. Which of the states are shift states and which are reduce states? Are there any conflicts?
  (c) Using the DFA, show an execution of the LR(0) parsing algorithm on the input

$$\texttt{((\#())\#)}$$

  Show the stack of states, stack of processed input, and remaining input throughout the execution.
  (d) Now consider the following context-free grammar G':

$$T \to \texttt{()} \mid \texttt{()}T$$

  Show that G' is not an LR(0) grammar by giving the DFA of valid item updates.

(2) (25 points) In this problem, you will design a Turing machine for the following language. Briefly explain how your Turing machine works (insufficient explanation may get zero points).

$$L = \{\texttt{a}^n \texttt{b}^n \texttt{c}^n \mid n \geqslant 0\}.$$

The input alphabet is $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{c}\}$. Give both a high-level description and a state diagram of your Turing machine.

(3) (30 points) A **Turing machine with doubly infinite tape** is similar to an ordinary Turing machine, but its tape is infinite to the left as well as to the right. The tape is initially filled with blanks except for the portion that contains the input. Computation is defined as usual except that the head never encounters an end to the tape as it moves leftward.
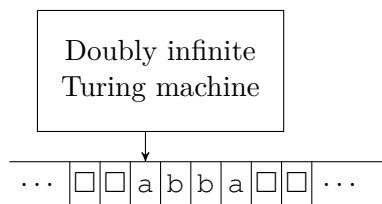
You will argue that this type of Turing machine is equivalent to the usual one-side-unbounded Turing machine.

(a) Write a formal definition of a doubly infinite tape Turing machine. A formal definition of an automaton will look like page 17 of Lecture 14 or page 9/slide 7 of Lecture 10.

(b) Show how to simulate a usual Turing machine on a doubly infinite tape Turing machine. You need to specify

- how the tape of the doubly infinite Turing machine will be used to represent the usual Turing machine.
- how the doubly infinite Turing machine tape should be set up initially;
- what the doubly infinite Turing machine should do when the usual Turing machine performs a transition (you may specify in 1-2 sentences the general idea, omitting the tedious details);
- what the doubly infinite Turing machine should do when the usual Turing machine accepts/rejects.

(c) Show how to simulate a doubly infinite Turing machine on a usual Turing machine. Again you should specify simulation details similar to those in part (b). *Hint:* You may want to simulate the doubly infinite Turing machine with some machine M that is not the usual Turing machine, but can itself be simulated by a usual Turing machine.

(4) (15 points) The Church–Turing Thesis is often quoted as the claim that Turing machines are a universal model of computation: Any computation that can be performed on any computer we will ever build can also be done on a Turing machine. Here are some possible objections to the Church–Turing Thesis. For each of these objections, say if you think it is reasonable or not, and explain why. **You only need to answer three out of four objections below**; you can choose any three to answer.

You won't be graded based on whether your answer is "right" or "wrong", but based on how well you explain your answer. We expect your answer to each objection to be about 3–6 sentences long. Try not to give an exceedingly long answer. For this question, you are encouraged to research on the Internet for supporting arguments, and cite appropriately.

(a) Suppose I want to know what is the smallest country in the world. In real life, I would use Google, type in "smallest country", and I find out the answer after a few clicks. But I cannot do this on a Turing Machine. How do I even connect a Turing Machine to the Internet? Since there are computations we can do in real life but not on a Turing Machine, the Church–Turing thesis is false.

(b) Modern machines may be equipped with sensors that receive analog signals, such as object detectors on a self-driving car. Analog signals and computation about them are not captured by a Turing machine.

(c) Some programming languages can handle infinite objects, such as infinite list in Lisp or Haskell. But every object in a Turing machine is finite. How could a Turing machine represent these infinite objects?

(d) Humans can also be modeled as computers: We take inputs from the environment (by seeing, hearing, touching) and produce outputs (via speaking and gestures). If the Church–Turing thesis is true, then any task that humans can do can also be done on a Turing Machine, and so on any machine. But there are tasks that humans are better at than machines: Learning foreign languages, identifying objects in images,

winning basketball games, and so on. Therefore the Church–Turing Thesis cannot be true.