CS 21 Decidability and Tractability **Winter 2023**

Problem Set 3

Out: January 25 *Due:* **February 1**

Reminder: you are encouraged to work in groups of two or three; however you must turn in your own write-up and note with whom you worked. You may consult the course notes and the text (Sipser). The full honor code guidelines and collaboration policy can be found in the course syllabus.

Please attempt all problems. **Please turn in your solutions via Gradescope, by 1pm on the due date.**

1. Here is an informal description of a 2-Stack Nondeterministic Pushdown Automata (2-NPDA): the machine is just like an ordinary NPDA, except there is a second stack that behaves just like the first. At each step, the machine reads a symbol from the tape (possible ϵ), pops specified symbols from each of the two stacks (either may be ϵ), pushes specified symbols onto each of the two stacks (either may be ϵ), and moves into a specified state. The machine accepts if there is some computation on its input string that causes it to reach an accept state.

   (a) Give a formal definition of a 2-NPDA, and a formal definition of what it means for a 2-NPDA to accept an input string. Your definition will probably begin with something like "A 2-NPDA is a 7-tuple..."

   (b) Give an informal description of a 2-NPDA that decides the language:

   $$L = \{a^n b^n c^n : n \geq 0\}.$$

   (c) Prove that 2-NPDAs are equivalent to Turing Machines. That is, show language L is decided by a 2-NPDA if and only if it is decided by a Turing Machine.

2. Here is an informal description of a Queue Automaton: the machine is similar to an NPDA, except that the stack ("last in, first out" memory) is replaced with a queue ("first in, first out" memory). At each step, the machine reads a symbol from the tape (possible ϵ), dequeues a specified symbol (possibly ϵ) from the head of the queue, enqueues a specified symbol onto the tail of the queue (possibly ϵ), and moves into a specified state. The machine accepts if there is some computation on its input string that causes it to reach an accept state.

   (a) Give a formal definition of a Queue Automaton, and a formal definition of what it means for a Queue Automaton to accept an input string. Your definition will probably begin with something like "A Queue Automaton is a 6-tuple..."

   (b) Prove that Queue Automata are equivalent to Turing Machines. That is, show language L is decided by a Queue Automaton if and only if it is decided by a Turing Machine. Hint: repeated pairs of "dequeue x" and "enqueue x" operations cyclically shift the

queue in one direction. Can you use a sequence of cyclic shifts to replace the symbol at the head of the queue? Can you use a sequence of cyclic shifts to effect a cyclic shift in the *other* direction? For this last part you may want a queue alphabet that contains all *pairs* of symbols from the input alphabet.

3. Prove that a language L is RE if and only if it can be expressed as

$$L = \{x : \text{there exists } y \text{ for which } \langle x, y \rangle \in R\}$$

where R is a decidable language. In other words, you must prove that every language of this form is RE, and that every RE language L has a related decidable language R_L that allows it to be described in the form above.