MAU22C00: ASSIGNMENT 6 SOLUTIONS

1) (20 points)

(a) Let L be the language over the alphabet $A = \{a, l, p\}$ consisting of all words containing both a and l. Write down the algorithm of a Turing machine that decides L. Process the following strings according to your algorithm: p, al, pap, pla, and $aapppla$.

(b) Write down the transition diagram of the Turing machine from part (a) carefully labelling the initial state, the accept state, the reject state, and all the transitions specified in your algorithm.

Solution: 1)

(a) Here is the algorithm for deciding L, which is actually a regular language:

(1) If there is a blank in the first cell, then REJECT. If p is in the first cell, then move right. If a is in the first cell, then move right and go to step 3. If l is in the first cell, then move right and go to step 4.

(2) If a is in the current cell, then move right and go to step 3. If l is in the current cell, then move right and go to step 4. If there is a blank in the current cell, then REJECT.

(3) If a or p is in the current cell, then move right and repeat this step. If l is in the current cell, then ACCEPT. If there is a blank in the current cell, then REJECT.

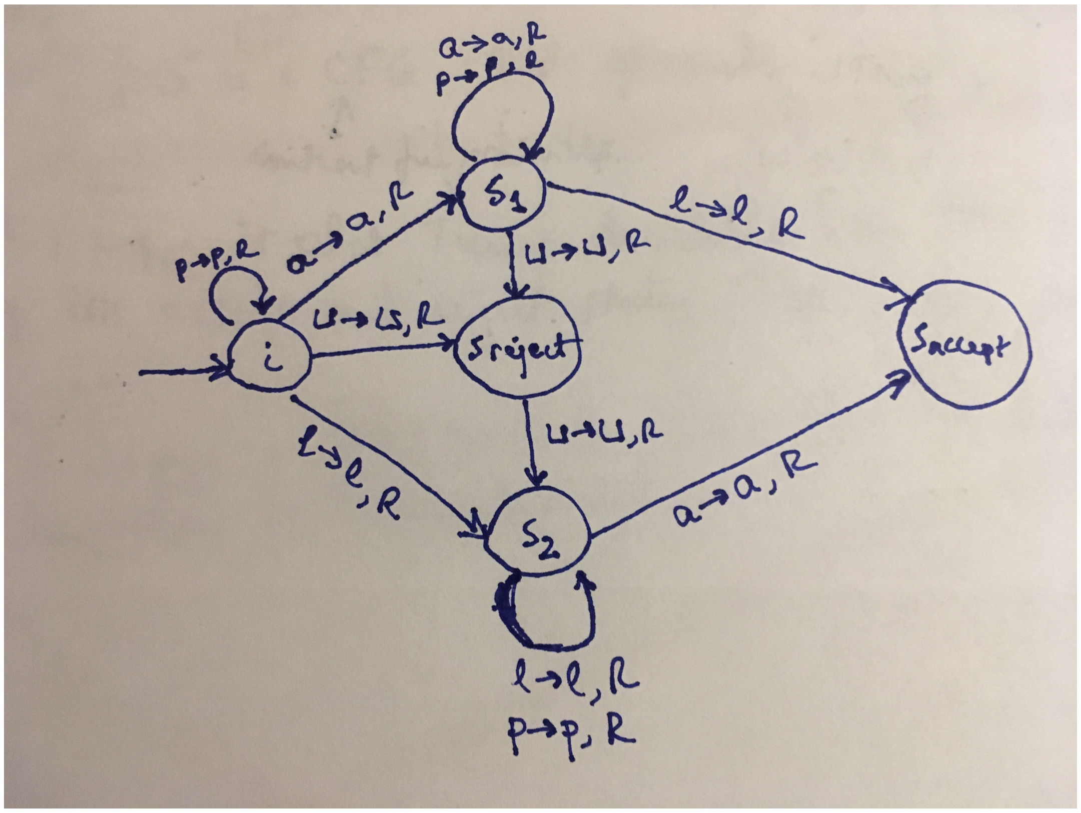
(4) If l or p is in the current cell, then move right and repeat this step. If a is in the current cell, then ACCEPT. If there is a blank in the current cell, then REJECT.

We will use a dot over a character to denote where the tape head is located. Here is how the following strings are treated:

- $\dot{p}\llcorner\!\lrcorner \rightarrow p\dot{\llcorner\!\lrcorner} \rightarrow$ REJECT at step 2.
- $\dot{a}l\llcorner\!\lrcorner \rightarrow a\dot{l}\llcorner\!\lrcorner \rightarrow$ ACCEPT at step 3.
- $\dot{p}ap\llcorner\!\lrcorner \rightarrow p\dot{a}p\llcorner\!\lrcorner \rightarrow pa\dot{p}\llcorner\!\lrcorner \rightarrow pap\dot{\llcorner\!\lrcorner} \rightarrow$ REJECT at step 3.
- $\dot{p}la\llcorner\!\lrcorner \rightarrow p\dot{l}a\llcorner\!\lrcorner \rightarrow pl\dot{a}\llcorner\!\lrcorner \rightarrow$ ACCEPT at step 4.
- $\dot{a}apppla\llcorner\!\lrcorner \rightarrow a\dot{a}pppla\llcorner\!\lrcorner \rightarrow aa\dot{p}ppla\llcorner\!\lrcorner \rightarrow aap\dot{p}pla\llcorner\!\lrcorner \rightarrow aapp\dot{p}la\llcorner\!\lrcorner \rightarrow$
  $aappp\dot{l}a\llcorner\!\lrcorner \rightarrow$ ACCEPT at step 3.

(b) Here is the transition diagram for

$T = (\{i, s_1, s_2, s_{\text{accept}}, s_{\text{reject}}\}, \{a, l, p\}, \{a, l, p, \text{␣}\}, t, i, s_{\text{accept}}, s_{\text{reject}})$:



Grading Rubric & Remarks: 6 marks for the algorithm, 4 marks for processing the strings and 10 marks for (b). If your algorithm from (a) is incorrect, but your transition diagram in (b) is faithful to what you wrote, no additional marks will be taken off.

For (b), large, clear, well labelled diagrams are a necessity. If it cannot be read, it cannot be awarded marks! Correct, consistent notation is also required.

2) (10 points) Let the alphabet $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Write down the algorithm of an enumerator that prints out EXACTLY ONCE every string in the language

$$L = \{3m + 1 \mid m \in \mathbb{N}\}$$

that is EVEN.

Solution: Let us order the input tape as $\{w_1, w_2, w_3, \dots\} = \{0, 1, 2, 3, 4, \dots\}$. Note that for $3m+1$ to be even, m has to be odd so that $3m$ is odd and

$3m+1$ is the sum of two odd numbers, hence even. Since m is odd, we can represent m as $2n+1$ for $n \in \mathbb{N}$. We plug $2n+1$ into $3m+1$ to obtain $3(2n+1)+1 = 6n+3+1 = 6n+4$.

```
$E =$ Given the input tape $\{w_1, w_2, w_3, \dots\}$
    (1) If $i = 6n+5$ for $n \in \mathbb{N}$, then print $w_i$.
```

We have $i = 6n+5$ instead of $i = 6n+4$ because $i = w_i + 1$, so $w_i = i - 1$.

Grading Rubric: 10 points; other correct solutions exist besides this one.

3) (20 points)

(a) The emptiness testing problem for phrase structure grammars (PSG's) is given by the language

$$E_{PSG} = \{\langle G \rangle \;\big|\; G \text{ is a phrase structure grammar and } L(G) = \emptyset\}.$$

Is it possible to modify the marking argument used in lecture to prove that the emptiness testing problem for context-free grammars E_{CFG} is Turing-decidable in order to prove that E_{PSG} is also Turing-decidable? Justify your answer.

(b) Is E_{PSG} a finite set, a countably infinite set, or an uncountably infinite set? Justify your answer. You may assume without proof the fact that if G is a phrase structure grammar, then $L(G)$ is a Turing-recognisable language.

Solution: 3 (a) In lecture, we proved that the emptiness testing problem for context-free grammars (CFG's) given by

$$E_{CFG} = \{\langle G \rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$$

is Turing-decidable by defining the following Turing machine:

```
$M =$ on input $\langle G \rangle$, where $G$ is a CFG:
    1. Mark all terminal symbols in $G$.
    2. Repeat until no new variable get marked:
    3. Mark any non-terminal $\langle T \rangle$ if $G$ contains a production rule
       $\langle T \rangle \to u_1 \cdots u_k$, and each symbol (terminal or non-terminal)
       $u_1, \dots, u_k$ has already been marked.
    4. If start symbol $\langle S \rangle$ is not marked, accept; otherwise, reject.
```

As we can see from step 4, if $\langle S \rangle$ is marked, then the context free grammar will end up generating at least one string as all terminals have already been marked in step 1. Therefore, $L(G) \neq \emptyset$, and we reject G. For a phrase structure grammar, each production rule has the form $v_1 \cdots v_s \to u_1 \cdots u_k$, where $v_1, \dots, v_s$ consists of terminals and non-terminals and is such that there is at least one non-terminal among them. If we could modify the above argument, then step 3 would read:

3. Mark any terminals and non-terminals $v_1, \dots, v_s$ if the phrase structure grammar G contains a production rule $v_1 \cdots v_s \to u_1 \cdots u_k$, and each symbol (terminal or non-terminal) $u_1, \dots, u_k$ has already been marked.

The issue is that we could have the following valid phrase structure grammar:
(1) $\langle S \rangle 0 \langle S \rangle \to 10100$.
(2) $\langle S \rangle \to 0 \langle S \rangle$.

Rule (2) produces $0^p \langle S \rangle$ for $p \geq 1$, and rule (1) never gets to be applied as we never obtain any string with a substring given by the left-hand side of rule (1), namely $\langle S \rangle 0 \langle S \rangle$. As a result, this phrase structure grammar produces the empty language, hence it should be accepted by our Turing machine with that modified step 3. Obviously, the right-hand side of rule (1) only consists of terminals, so by the modified step 3, we would mark all terminals and non-terminals on the left-hand side of rule (1) including the start symbol $\langle S \rangle$. This phrase structure grammar would thus be incorrectly rejected at step 4. Therefore, the marking argument used for CFG's cannot work for PSG's. In fact, E_{PSG} is not Turing-decidable, but it is via far more advanced methods than presented in this module that this assertion is proven.

(b) Given that if G is a phrase structure grammar $L(G)$ is a Turing-recognisable language, there are at most as many languages generated by phrase structure grammars as there are Turing-recognisable languages. We proved in lecture that the set of Turing-recognisable languages is countably infinite. Therefore, E_{PSG} is a subset of a countably infinite set, hence either finite or countably infinite. To rule out that it could be finite, we notice that for any $n \in \mathbb{N}^*$, we can write down a phrase structure grammar G_n with n production rules such that $L(G_n) = \emptyset$. Since $\mathbb{N}^*$ is countably infinite and $\{G_1, G_2, \dots\} \subset E_{PSG}$, we conclude that E_{PSG} must be countably infinite.

Grading Rubric: 10 points per part. Everyone got full marks for part (a) due to the confusion created by the original statement. For part (b), proving E_{PSG} is the subset of a countably infinite set is worth 5 points. The rest of the argument is worth the remaining 5 points.