# MA2C03: TUTORIAL 18 PROBLEMS
# COUNTABILITY OF SETS AND TURING MACHINES

1) Prove that the language generated by a regular expression is countable. Give an example of a regular expression that generates a finite language and another example of a regular expression that generates a countably infinite language. Justify your answers.

2) (a) Consider the language over the binary alphabet $A = \{0,1\}$ given by $L = \{0^m 1^{2m} \mid m \in \mathbb{N}\}$. Write down the algorithm of a Turing machine that recognizes $L$. Process the following strings according to your algorithm: $\epsilon$, 01, 011, and 010.

(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) By definition, a set is countable, if it is finite or countably infinite. A regular expression is built up from $\emptyset$, $\epsilon$, and the letters of the alphabet $A$ via the Kleene star $*$, concatenation, and union. The Kleene star makes a countably infinite set out of a finite one. Concatenation gives a set whose size matches the size of the biggest set in the concatenation. In other words, the concatenation of strings from two finite sets will yield a finite set. The concatenation of strings from a finite set with a countably infinite set will yield a countably infinite set, whereas the concatenation of strings from two countably infinite sets yields a countably infinite set. Union behaves just like concatenation. Therefore, from a finite set via the Kleene star, union, and concatenation, we can only obtain a finite set or a countably infinite set. This concludes our proof. To give the required examples, let us consider the binary alphabet $A = \{0,1\}$. The regular expression $\{01\} \cup \{11\}$ yields a regular language with two elements, whereas the regular expression $0^* \cup 1^*$ gives the regular language consisting of all strings of just 0's and all strings of just 1's, which is countably infinite as the sequence of strings $\epsilon$, 0, 00, 000, etc. is inside this language.

2)

(a) Here is the algorithm for recognising $L = \{0^m 1^{2m} : m \in \mathbb{N}\}$.

(1) If there is a blank in the first cell, ACCEPT. If there is anything else, apart from 0, then REJECT.

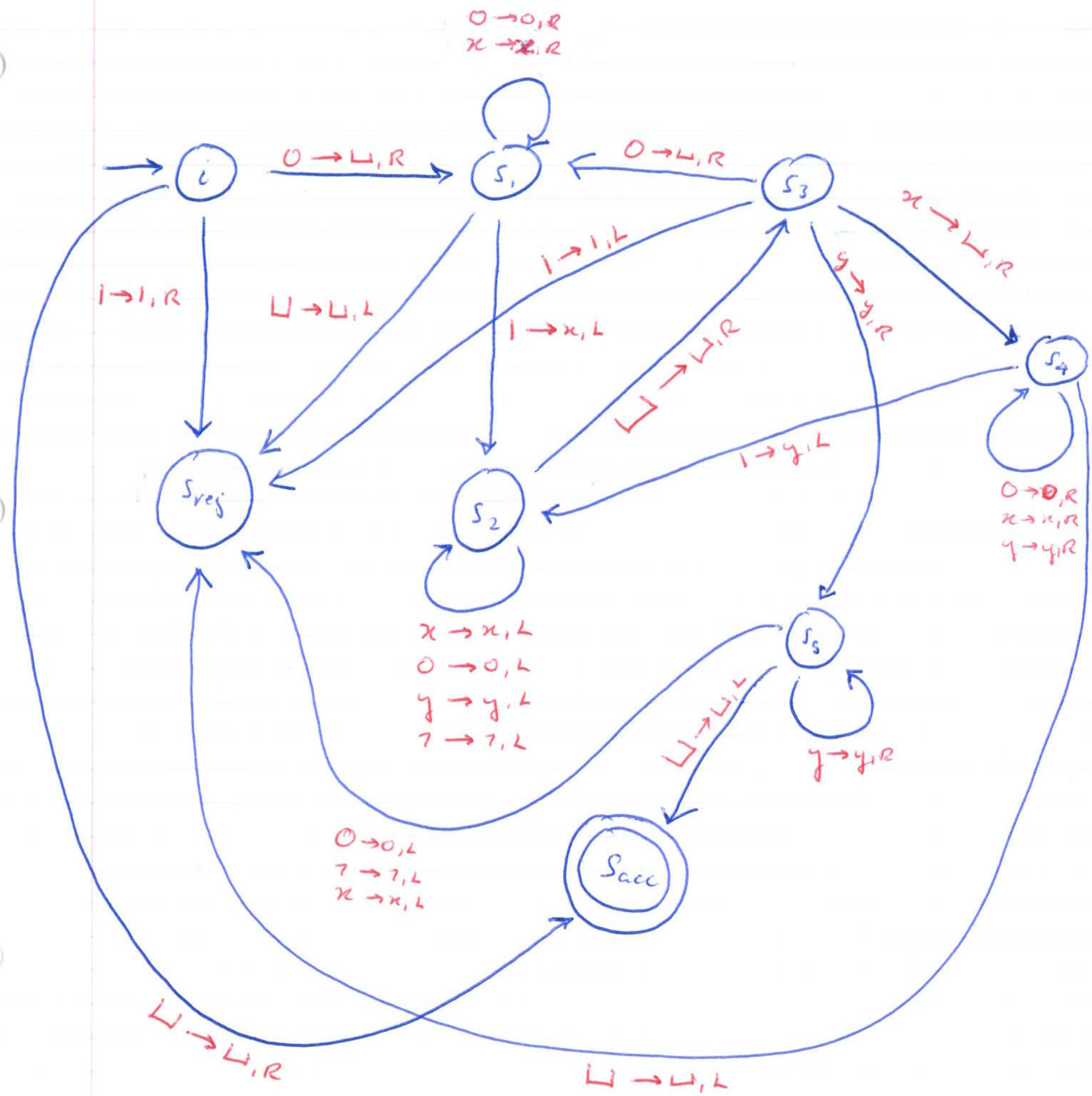
(2) If 0 is in the current cell, delete it, then move right to the first 1.
(3) If there is no first 1, REJECT. Otherwise change 1 to $x$.
(4) Move to the leftmost non blank symbol. If 0, go to step 2. If 1, REJECT. If $x$, go to step 5. If $y$, go to step 6.
(5) Delete $x$, move right to the nearest 1. If none, REJECT. Otherwise change it to $y$ and go to step 4.
(6) Move right to the rightmost non blank character. If anything but $y$ is found, REJECT. Otherwise, ACCEPT.

Here is how the following strings are treated:
- $\epsilon$ is accepted immediately.
- $01 \to \textvisiblespace 1 \to \textvisiblespace x \to \textvisiblespace\textvisiblespace \to$ REJECT.
- $011 \to \textvisiblespace 11 \to \textvisiblespace x1 \to \textvisiblespace\textvisiblespace 1 \to \textvisiblespace\textvisiblespace y \to$ ACCEPT.
- $010 \to \textvisiblespace 10 \to \textvisiblespace x0 \to \textvisiblespace\textvisiblespace 0 \to$ REJECT.

(b) The transition diagram for

$$T = (\{i, s_1, s_2, s_3, s_4, s_5, s_{\text{acc}}, s_{\text{rej}}\}, \{0, 1\}, \{0, 1, x, y, \textvisiblespace\}, t, i, s_{\text{acc}}, s_{\text{rej}})$$

is at the end of the solution set, along with an example accepted string.




E.g

| | |
|---|---|
| 0 0 1 1 1 1 | $i$ |
| ⊔ 0 1 1 1 1 | $s_1$ |
| 0 x 1 1 1 | $s_2$ |
| 0 x 1 1 1 | $s_3$ |
| ⊔ x 1 1 1 | $s_1$ |
| x x 1 1 | $s_2$ |
| x x 1 1 | $s_3$ |
| ⊔ x 1 1 | $s_4$ |
| x y 1 | $s_2$ |
| x y 1 | $s_3$ |
| ⊔ y 1 | $s_4$ |
| y y | $s_2$ |
| y y | $s_3$ |
| | $s_{acc}$ ! |