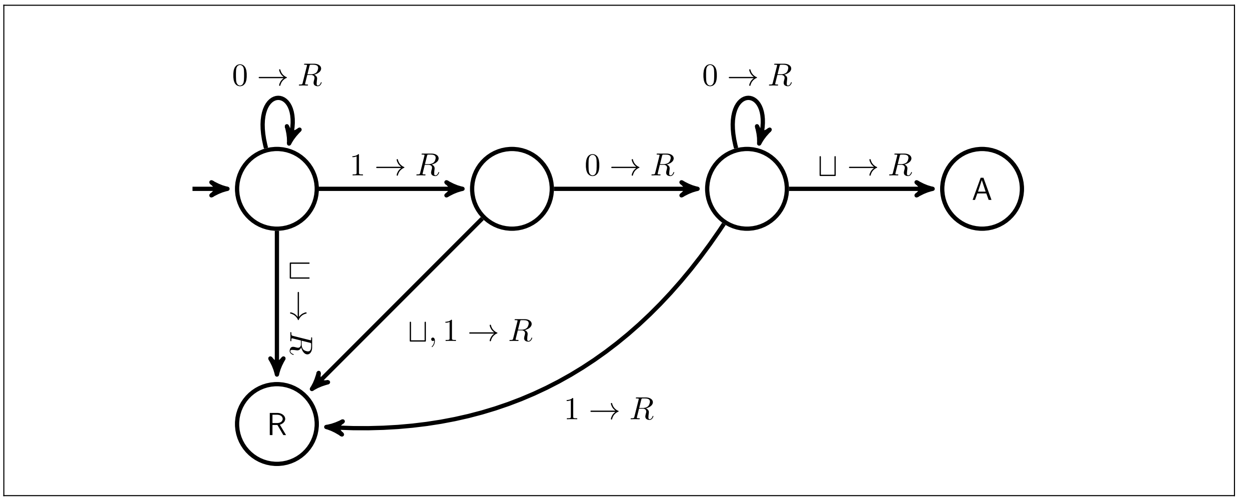
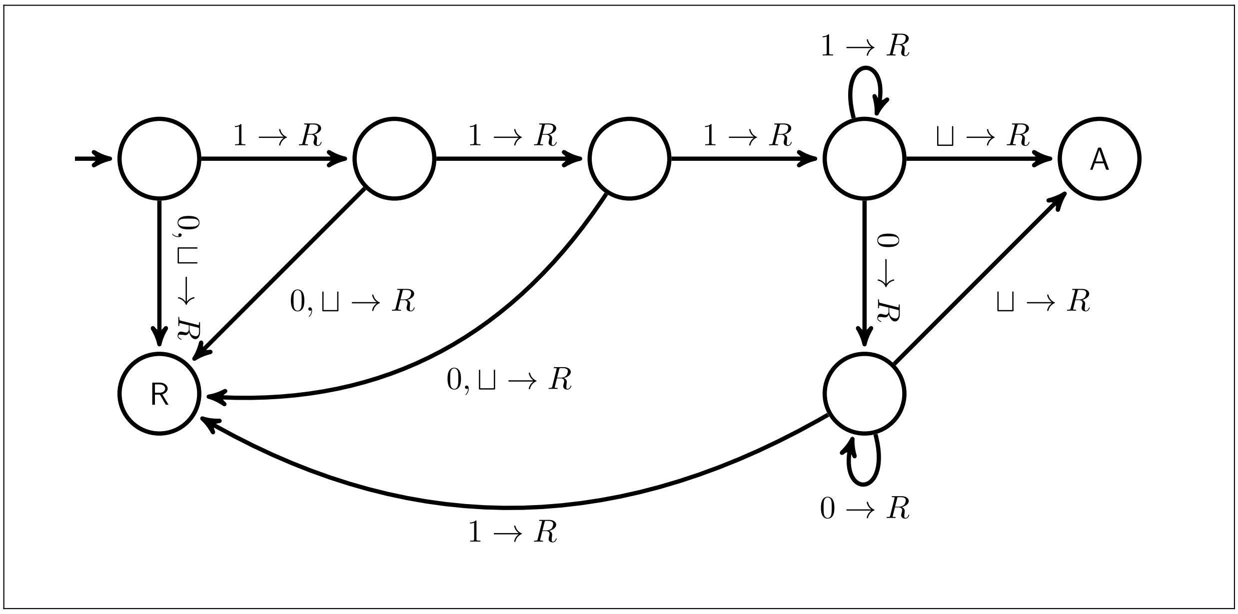
Spring 2018: COT3210–Computability and Automata
Test 04–Answers

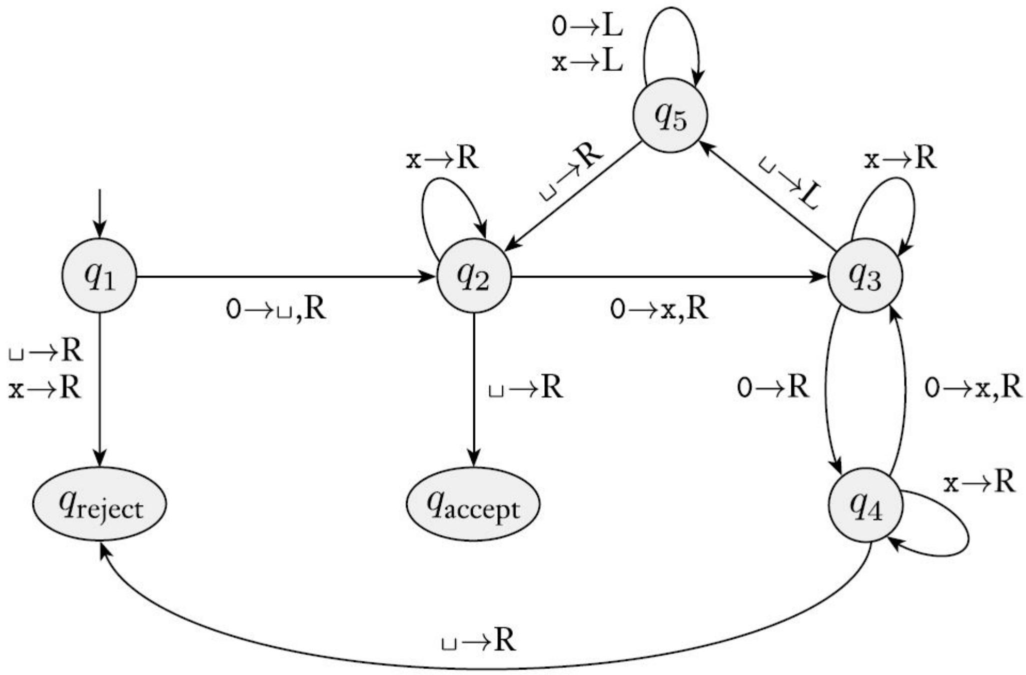
1. The required state diagram for a Turing Machine that accepts the language 0^*100^* is given below.



2. The required state diagram for a Turing Machine that accepts the language 1111^*0^* is shown below.



3. The sequence of configurations that the machine will enter when supplied with the input string 00 is given below the diagram:



$$q_1 00\sqcup;\ \sqcup q_2 0\sqcup; \sqcup x q_3 \sqcup; \sqcup q_5 x \sqcup; q_5 \sqcup x \sqcup; \sqcup q_2 x \sqcup; \sqcup x q_2 \sqcup; \sqcup x \sqcup q_{\text{accept}}$$

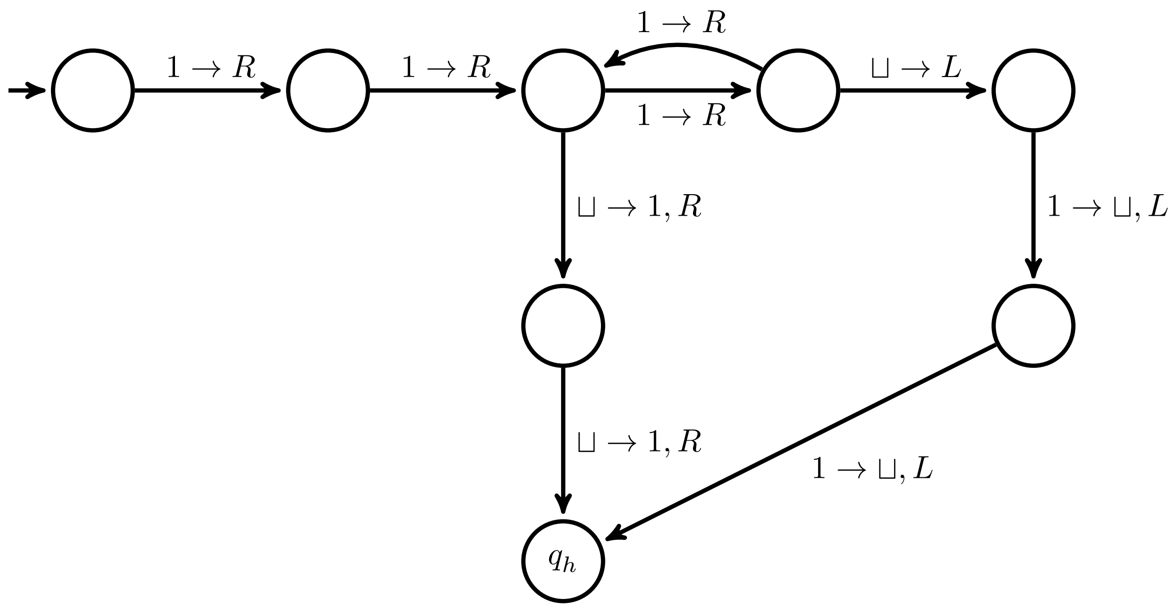
---

4. The sequence of configurations that the machine will enter when supplied with the input string 000 is shown below:

$$q_1 000\sqcup;\ \sqcup q_2 00\sqcup; \sqcup x q_3 0\sqcup; \sqcup x 0 q_4 \sqcup; \sqcup x 0 \sqcup q_{\text{reject}}$$

5. (10 pts). With $\Sigma = \{1\}$, assuming that the input consists x occurrences of 1s followed by blanks, and $x > 2$, draw the state diagram of a Turing Machine that will compute

$$f(x) = \begin{cases} x+2, & \text{if } x \text{ is even;} \\ x-2, & \text{if } x \text{ is odd.} \end{cases}$$



---

6. The required estimate in Big O notation for the number of times "Hello" is printed by the following code segment is given below in the boxes provided. Justify your answer.

$k = 2$;
while $(k < n)$ **do**
  print Hello
  $k = k^3$

The answer: $O(\log\ \log n)$

Explain your work here to receive full credit:
Initially $k = 2$ and then k takes on values of $2^1, 2^3, (2^3)^3, ((2^3)^3)^3, \cdots$ successively for each execution of the loop.
If the loop executes m times, these values are writen as

$$2^{3^0}, 2^{3^1}, \cdots, 2^{3^{m-1}}.$$

Thus, m should satisfy the condition $2^{3^{m-1}} < n \leq 2^{3^m}$. In other words, $3^{m-1} < \log_2 n$, from which we may conclude $m < \log_3 \log_2 n + 1$. Thus, the number of time Hello is printed will be $O(\log\ \log n)$.

7. Trace the execution of the following algorithm for $m = 3$ and $n = 91$. As you will see, the values of m and p increase exponentially. Therefore, you will keep them both in the form $3^{\text{something}}$ throughout your trace. For an arbitrary value of n, estimate the worst-case number of times the operation $p = p * m$ is executed, in big O notation. Explain your answer. Assume that all divisions are integer divisions. The blank trace table is given for your convenience. Use it to show the trace. Put your big O answer and the explanation in the box provided below.

```
m = 3; n = 91; p = 1;
for ( ; n > 0; n /= 2)
        if (odd(n))
            p = p * m
        m = m * m
```

m	n	p
3	91	1
3^2	45	3^1
3^4	22	3^3
3^8	11	3^3
3^{16}	5	3^{11}
3^{32}	2	3^{27}
3^{64}	1	3^{27}
3^{128}	0	3^{91}

n begins with the value of 91. It takes on successively the values $\frac{n}{2^0}, \frac{n}{2^1}, \cdots, \frac{n}{2^{k-1}}$ if the loop executes k times. Thus, in the worst-case, the statement `p = p*m` will be executed every time through the loop. In other words, it is executed k times, with $\frac{n}{2^{k-1}} \geq 1$, or $k \approx \log_2 n = O(\log n)$.

8. Show the remainder sequence generated by the modern Euclidean algorithm in the computation of gcd$(6765, 2500)$. What is the gcd computed? Compare the number of division steps needed for this computation with the worst-case number of division steps predicted by the analysis of this algorithm discussed in class.

The remainder sequence is: 6765, 2500, 1765, 735, 295, 145, 5, 0.
Thus, gcd(6765,2500) $=$ 5.
The number of actual number of division steps the algorithm performs is 6.
The maximum number of steps is estimated as $\log 6765/\log 1.618 \approx 18$.

9. The growth functions arranged in the increasing order of complexity in the Big-Oh sense are shown as below:

$$10^{100}, \quad \log\log n, \quad 3^{\log_2 n}, \quad n^{2.89}, \quad 10000000n^3 \quad 2^n, \quad 3^n, \quad n!$$