HomeWork 3
Computer Theory

Peter Keres

May 2020

1 Q1

1.1 Question 1

1.1.1 Stage1:

Check that your first input char is a 0 and replace it with a *.
if not reject

1.1.2 Stage2:

now scan past all the following zeros until a 1 is found and replace it with a *.
if a 2 is found reject

1.1.3 Stage3:

now scan past all the following 1's until a 2 is found and replace it with a *.
if a 0 is found reject

1.1.4 Stage4:

now scan back to the beginning until you find the 0's, once there find the first *.
scan until the next 0 and replace with a *

1.1.5 Stage5:

repeat stages 2-4 until there are no more 0's at stage 4.
once this happens, scan the input for any character other then a *.
if found, reject

1.2 Question 2

1.2.1 Stage 1 only takes bigO(1)

because you are only reading 1 input and placing a character

1.2.2 Stages 2-4 take bigO(N)

Because you might have to go over the whole input to replace 1 set of 0,1,2 with a *

1.2.3 Stages 5 takes bigO(M)

Because stages 2-4 need to be repeated any number of times for the whole input

1.2.4 Answer bigO(MN)

so this leaves us with O(1) O(n) O(m)
we drop the 1 because it doesn't matter
left with bigO(MN)

1.3 Question 3

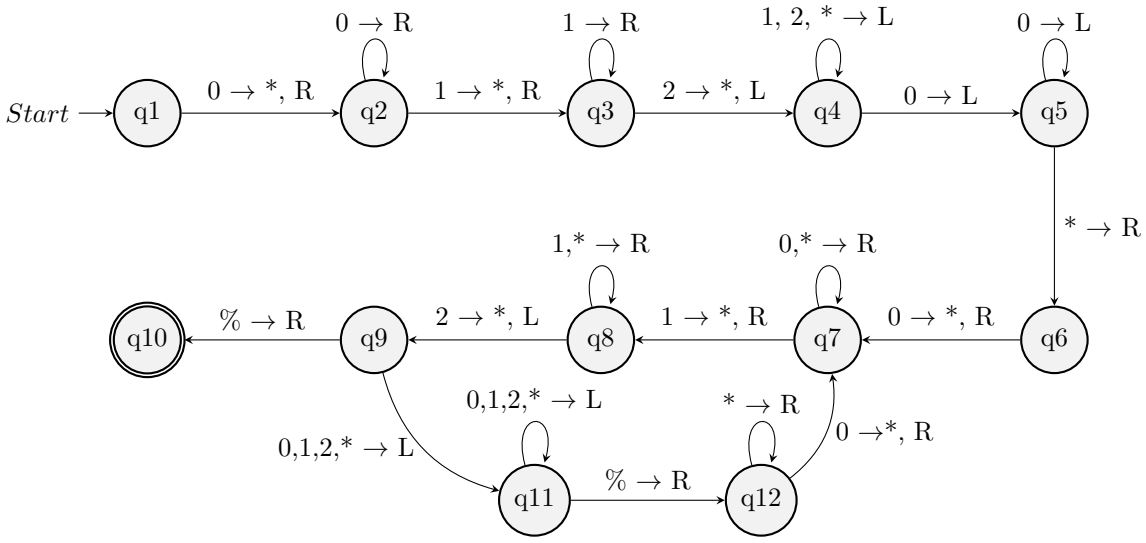

Figure 1: Q1 TM diagram

State	Read	Next	Write	Move
q1	0	q2	*	R
q2	0	q2	-	R
q2	1	q3	*	R
q3	1	q3	-	R
q3	2	q4	*	L
q4	1,2,*	q4	-	L
q4	0	q5	-	L
q5	0	q5	-	L
q5	*	q6	-	R
q6	0	q7	*	R
q7	0	q7	-	R
q7	1	q8	*	R
q8	1	q8	-	R
q8	2	q9	*	R
q9	%	q10	*	R
q9	0,1,2,*	q11	-	L
q11	0,1,2,*	q11	-	L
q11	%	q12	-	R
q12	*	q12	-	R
q12	0	q7	*	R

Table 1: Q1 transition table

2 Q2

2.1 Question 1

2.1.1 Stage1:

scan the input until you find a 1 and replace it with a *.
if non is found accept

2.1.2 Stage2:

now scan the front part of the input looking for 2 0's and replace them with a *.
if none is found scan the whole input for 2 0's and replace them with a *
if none is found, reject

2.1.3 Stage3:

Now scan the input for a 1 and replace it with a *.
if none found accept

2.1.4 Stage4:

now scan the front part of the input looking for 0 and replace it with a *.
if none is found scan the whole input for 0 and replace it with a *
if none is found, reject

2.1.5 Stage5:

repeat stages 3-4

2.2 Question 2

2.2.1 Stage1 takes bigO(n)

because it will have to look over the whole input once to find a 1.

2.2.2 Stage2 takes bigO(NlogN)

because in the worst case, when you find a 1 you are in the middle.
once in the middle you have to go all the way left. which is 1/2 of —w— or logN.
if its not found here, you have to go over the right side of input in-case its in the last overall slot of the input. making this move bigO(N)
Theses 2 moves would create a possible search of NlogN.

2.2.3 Stage 3 takes bigO(N)

because it will have to look over the hwole input once to find a 1.

2.2.4 Stage 4 takes bigO(NlogN)

because in the worst case, when you find a 1 you are in the middle.
once in the middle you have to go all the way left. which is 1/2 of —w— or logN.
if its not found here, you have to go over the right side of input in-case its in the last overall slot of the input. making this move bigO(N)
Theses 2 moves would create a possible search of NlogN.

2.2.5 Stage 5 takes bigO(M)

because it has to do stages 3-4 M amount of times for the number of 1's in the input

2.3 Answer bigO(M NlogN)

so we over all have (N NlogN N NlogN M)
we drop stage 1 and 3 from complexly because if it ends here, its not worst case. leaving (NlogN NlogN M).
we drop stage 2 because its the same actions of stage 4 so its not needed in the final complexity. leaving (NlogN M).
Therefore we are left bigO(M NlogN).
because we must do the action of finding and replacing 1's and 0's with *'s (NlogN). the number of times as the input(M).

2.4 Question 3

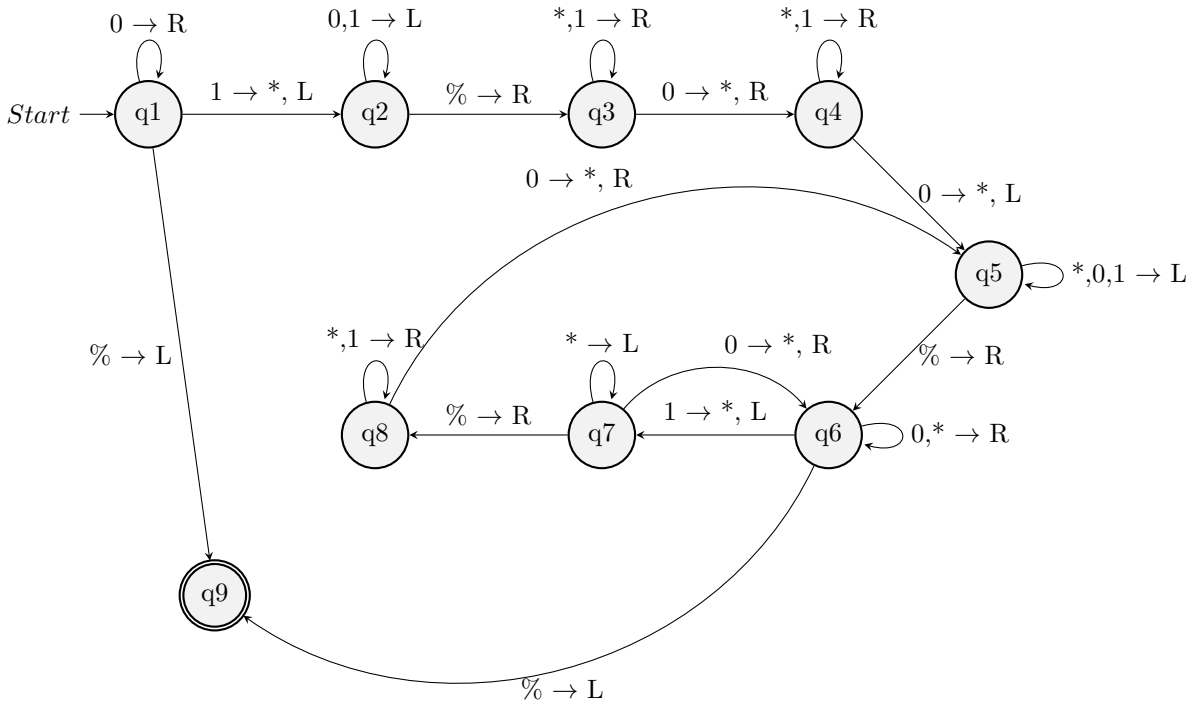

Figure 2: Q2 TM diagram

State	Read	Next	Write	Move
q1	0	q1	-	R
q1	1	q2	*	L
q1	%	q9	-	L
q2	0,1	q2	-	L
q2	%	q3	-	R
q3	*,1	q3	-	R
q3	0	q4	*	R
q4	*,1	q4	-	R
q4	0	q5	*	L
q5	*,0,1	q5	-	L
q5	%	q6	-	R
q6	0,*	q6	-	R
q6	1	q7	*	L
q6	%	q9	-	L
q7	*	q7	-	L
q7	0	q6	*	R
q7	%	q8	-	R
q8	*,1	q8	-	R
q8	0	q5	*	R

Table 2: Q2 transition table