CS375: Logic and Theory of Computing

Fuhua (Frank) Cheng

Department of Computer Science

University of Kentucky

Table of Contents:

Table of Contents (conti):

- **Weeks 12-13: Propositional Logic (Chapter 6), Predicate Logic (Chapter 7), Computational Logic (Chapter 9), Algebraic Structures (Chapter 10)**

8. Turing Machines and Equivalent Models – The Church-Turing Thesis

Goal: to show you construction of Turing Machines that can perform:

1. the addition function,
2. the subtraction function,
3. the multiplication function, and
4. the division function

for unary numbers

TM for the **addition function** for the unary number system

Notations:

The unary number is made up of only one character, i.e. The number 5 can be written in unary number system as 11111. In this TM, we are going to perform the addition of two unary numbers.

For example:

2 + 3

i.e., 11 + 111 = 11111

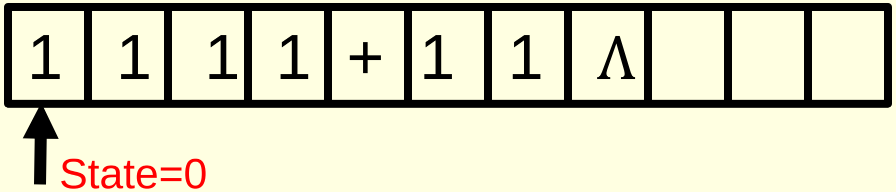
Solution:

If you observe this process of addition, you will find the resemblance with **string concatenation** function.

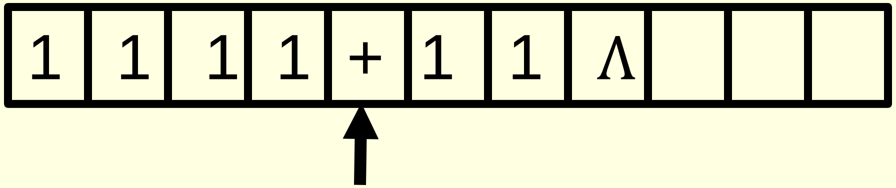
In this case, we simply replace ‘+’ by ‘1’ and move right to search for the right most ‘1’ and convert that ‘1’ to ‘Λ’.

Input: 4+2

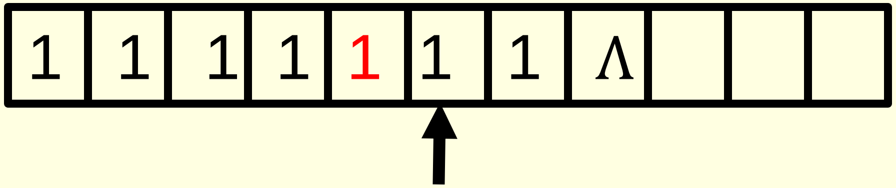
The simulation for 1111+11Λ can be shown as below:



Move right up to the + sign:

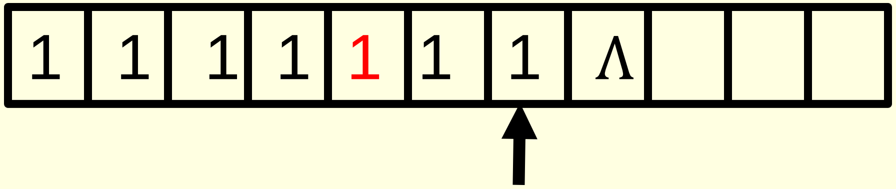


Convert + to 1 and move right:

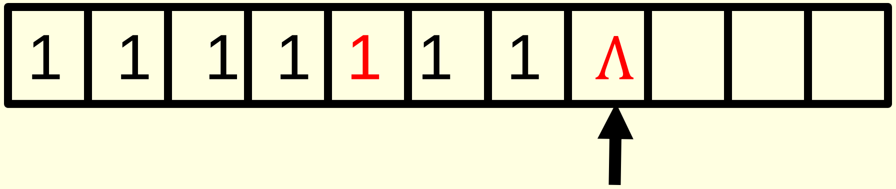


Conti.

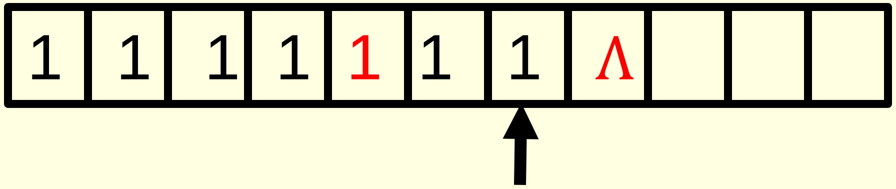
Keep moving right:



Until we reach Λ:

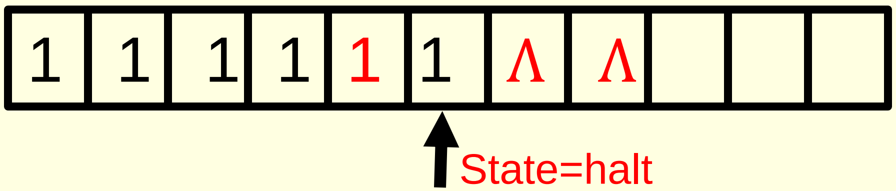


Move left :

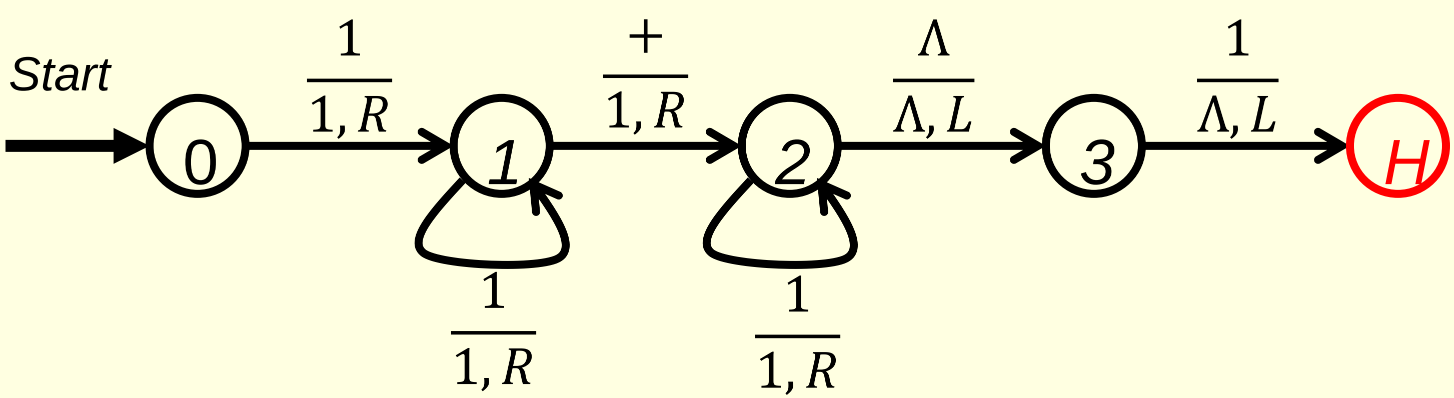


Conti.

Convert 1 to Λ, move left and stop :

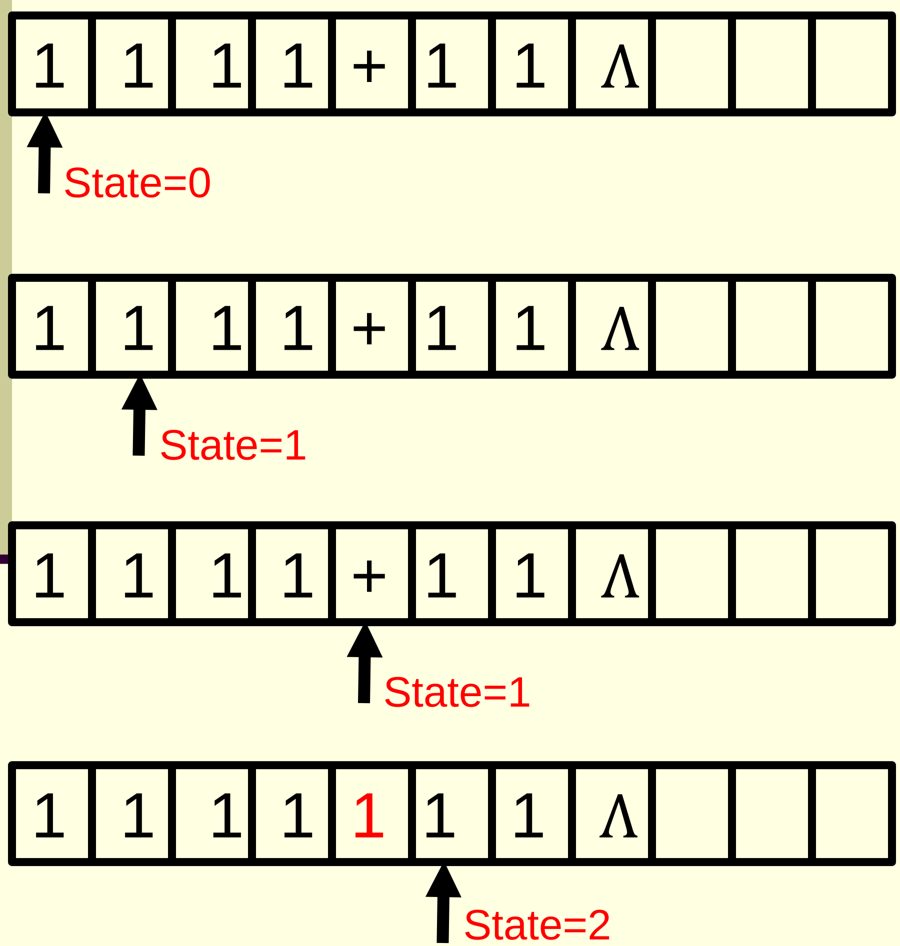


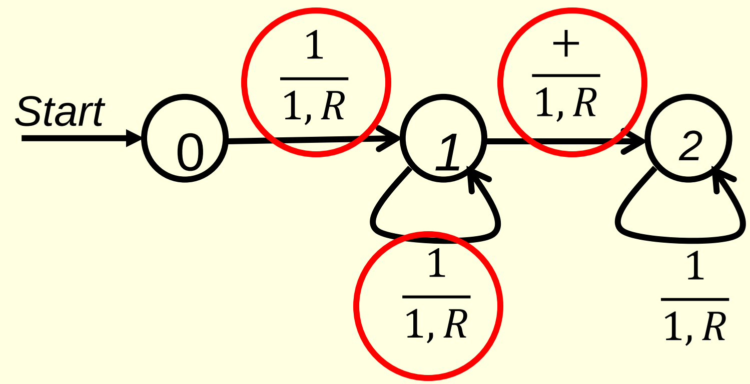
Such a TM would look like the one below :



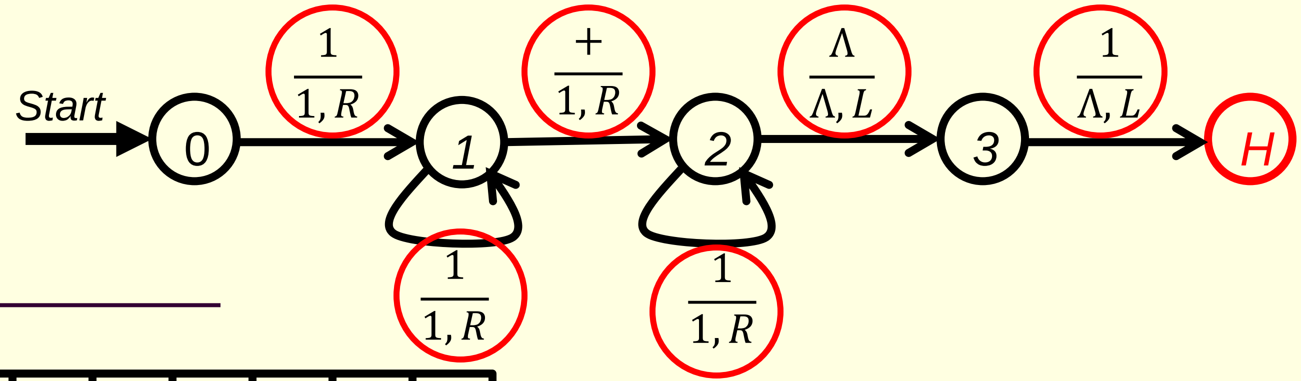
Input: 4+2

The simulation for 111+11Λ can be shown as below:





Conti.



1	1	1	1	1	1	1	Λ			

State=2

1	1	1	1	1	1	1	Λ			

State=2

1	1	1	1	1	1	1	Λ			

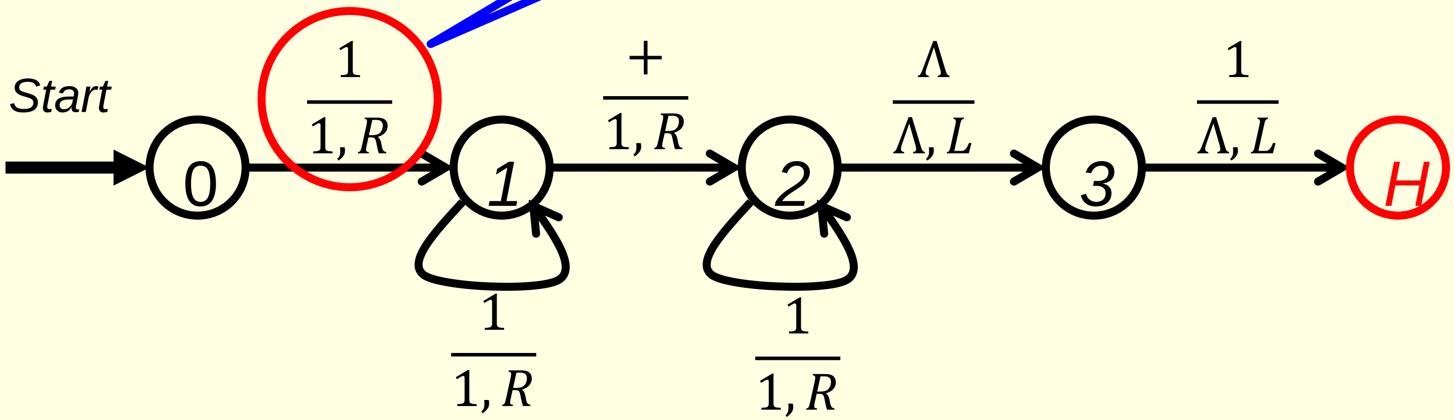
State=3

1	1	1	1	1	1	Λ	Λ			

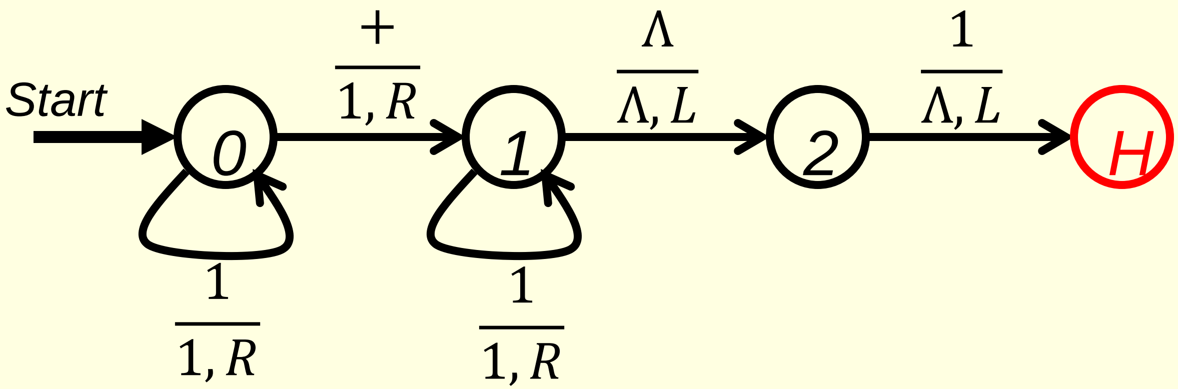
State=halt

Note:

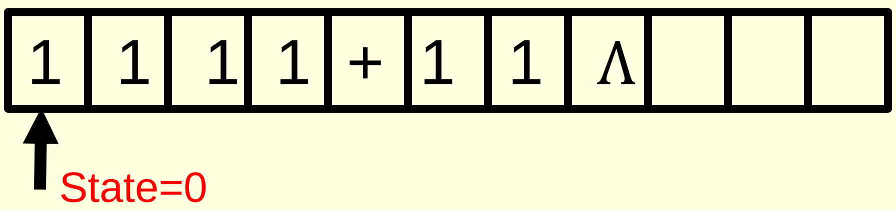
This $\frac{1}{1,R}$ is to prevent having a '+' in the left most cell



Would the following TM work as well?

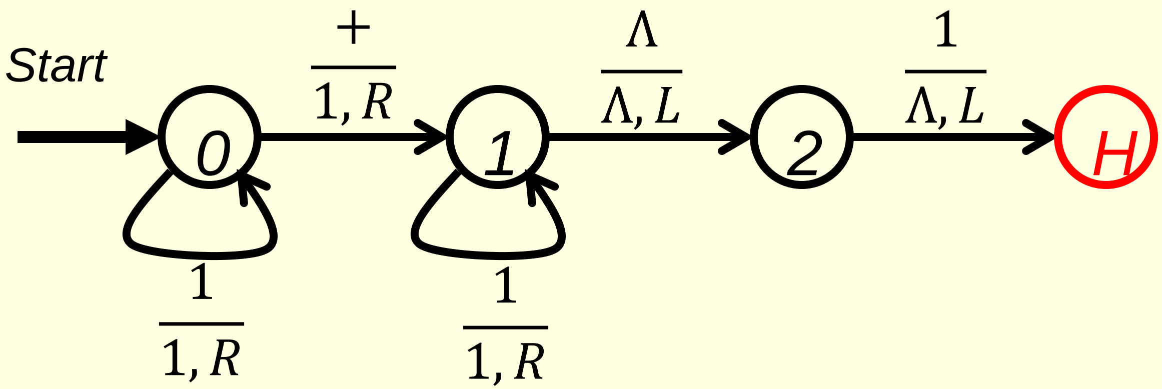


Can it handle the following input?

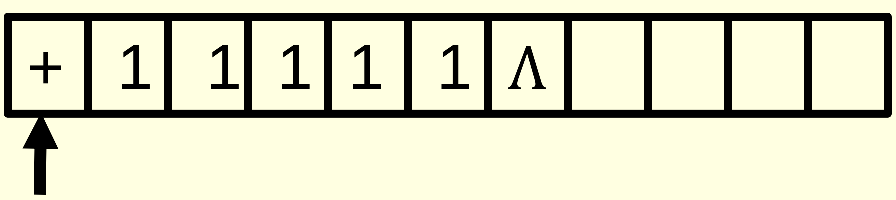


YES

Would the following TM work?



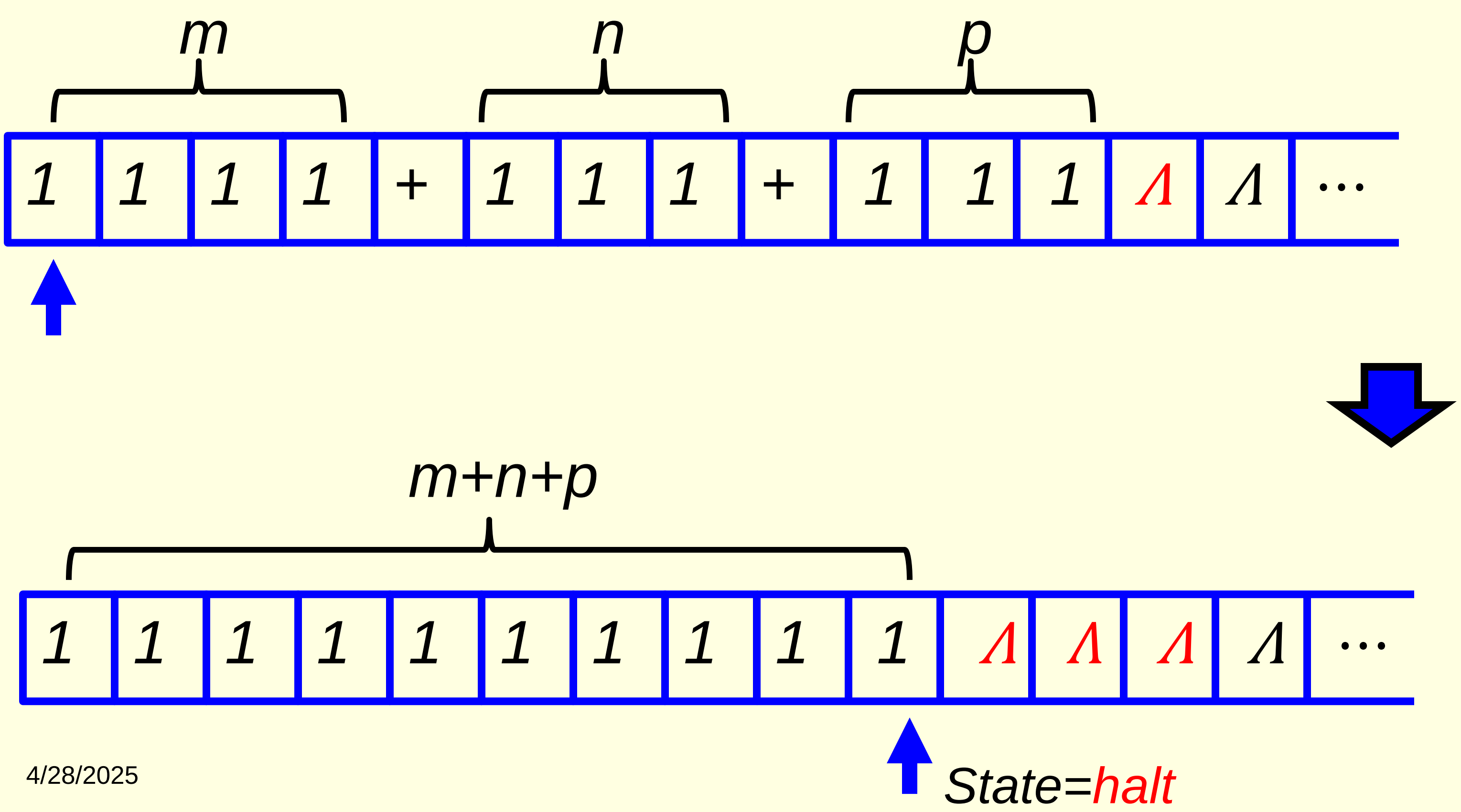
What if the input is like the one below?



This input would be rejected by the TM given in slide 9

Question:

Can the above TM be modified to do addition of three numbers, four numbers, ..., n numbers in unary form directly?



TM for the **subtraction function** for the unary number system

Notations and Assumption:

For example:

4 - 2

i.e., 1111 - 11 = 11

Develop a TM for the subtraction of two unary numbers f(a-b) = c where ‘a’ is greater than or equal to ‘b’.

Solution:

If there are n 1’s in the unary representation of *b* and m 1’s in the unary representation of *a*, the process is to reduce n 1’s from the unary representation of *a*.

The reduction will be perform from the right side of the unary representation of *a*.

Input: 4-2

So the input tape will look like:

1	1	1	1	-	1	1	Λ			

State=0

Move right to the '–' symbol:

1	1	1	1	-	1	1	Λ			

Move right and convert 1 to '*' as:

1	1	1	1	-	*	1	Λ			

Move left:

1	1	1	1	-	*	1	Λ			

Conti.

Again move left:

1	1	1	1	-	*	1	Λ			

Convert 1 to '*' and move right:

1	1	1	*	-	*	1	Λ			

Keep moving right until a '1' is reached:

1	1	1	*	-	*	1	Λ			

Conti.

Convert 1 to * and move left:

1	1	1	*	-	*	*	Λ			

Keep moving left until a 1 is reached:

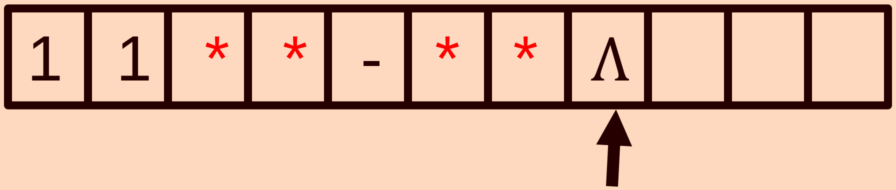
1	1	1	*	-	*	*	Λ			

Convert 1 to * and move right

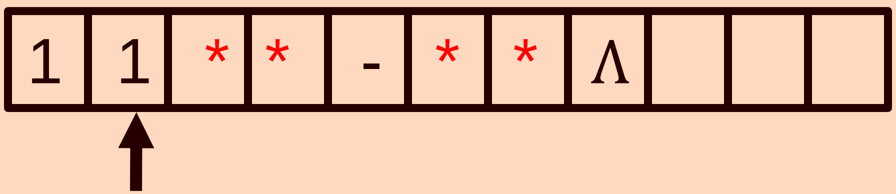
1	1	*	*	-	*	*	Λ			

Conti.

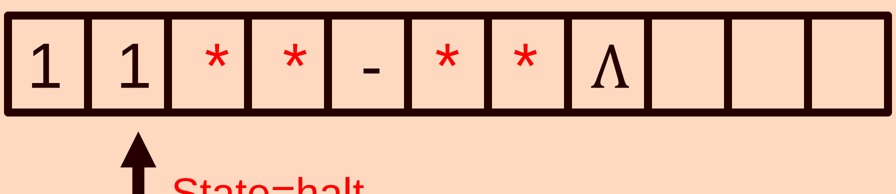
Keep moving, ignore all *’s and ‘–’ until we reach an ‘1’ or a ‘Λ’



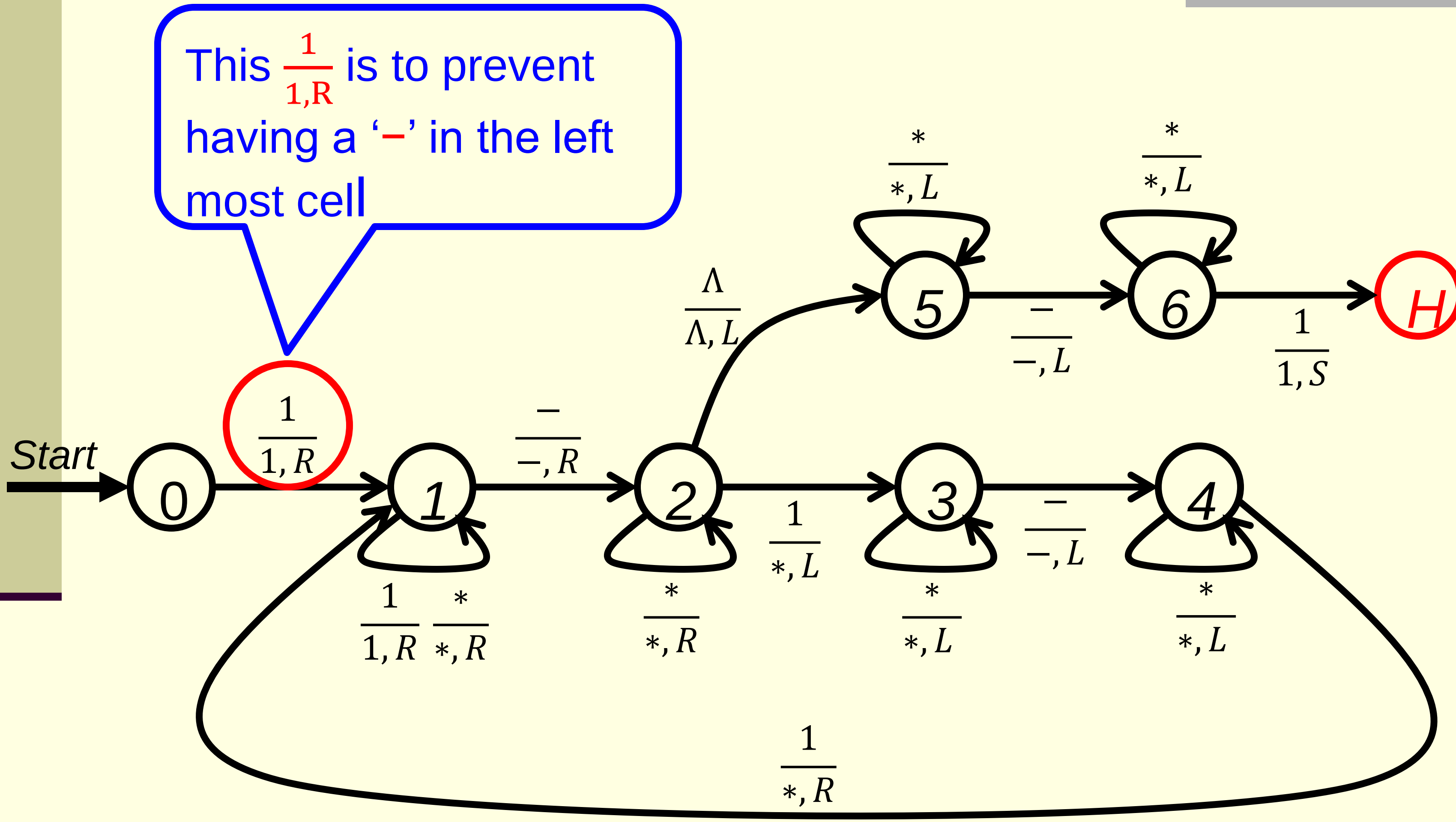
Find a ‘Λ’. Turn left, ignore *’s and the ‘–’, until an ‘1’ is reached



Change the state to ‘halt’ and stop.

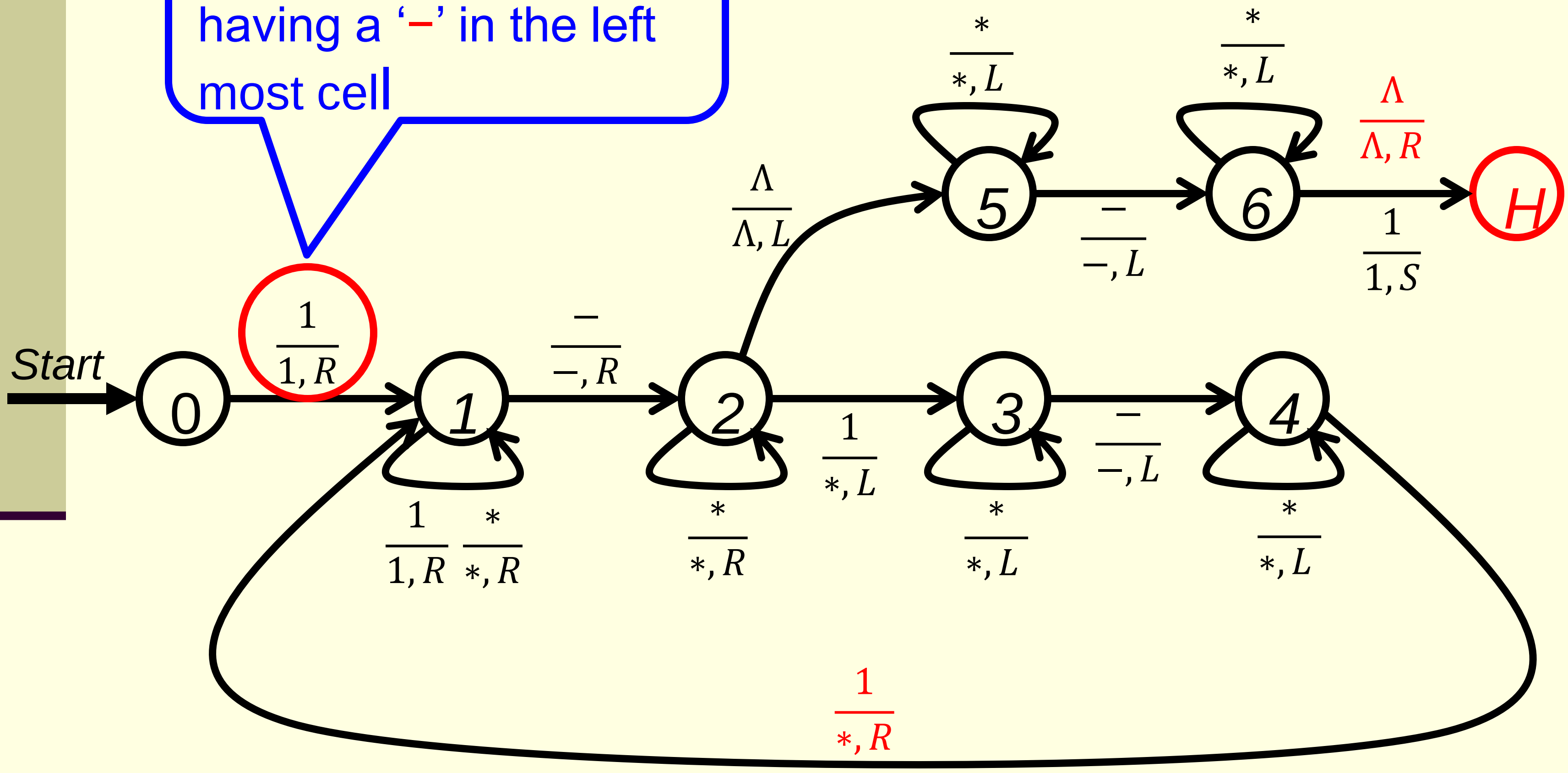


The TM will look like the one below (when n < m):

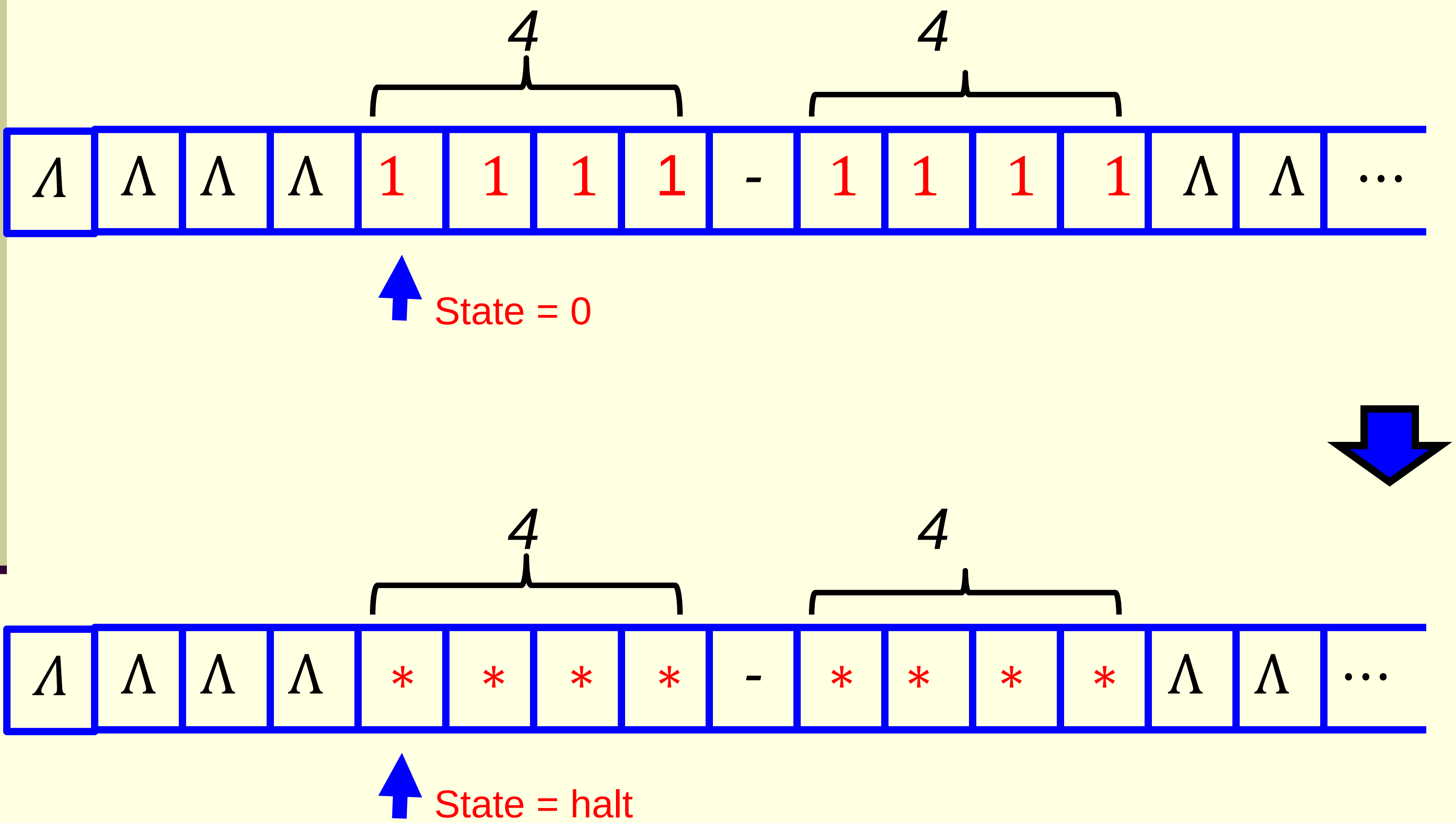


The TM will look like the one below (when n <= m):

This $\frac{1}{1,R}$ is to prevent having a '–' in the left most cell

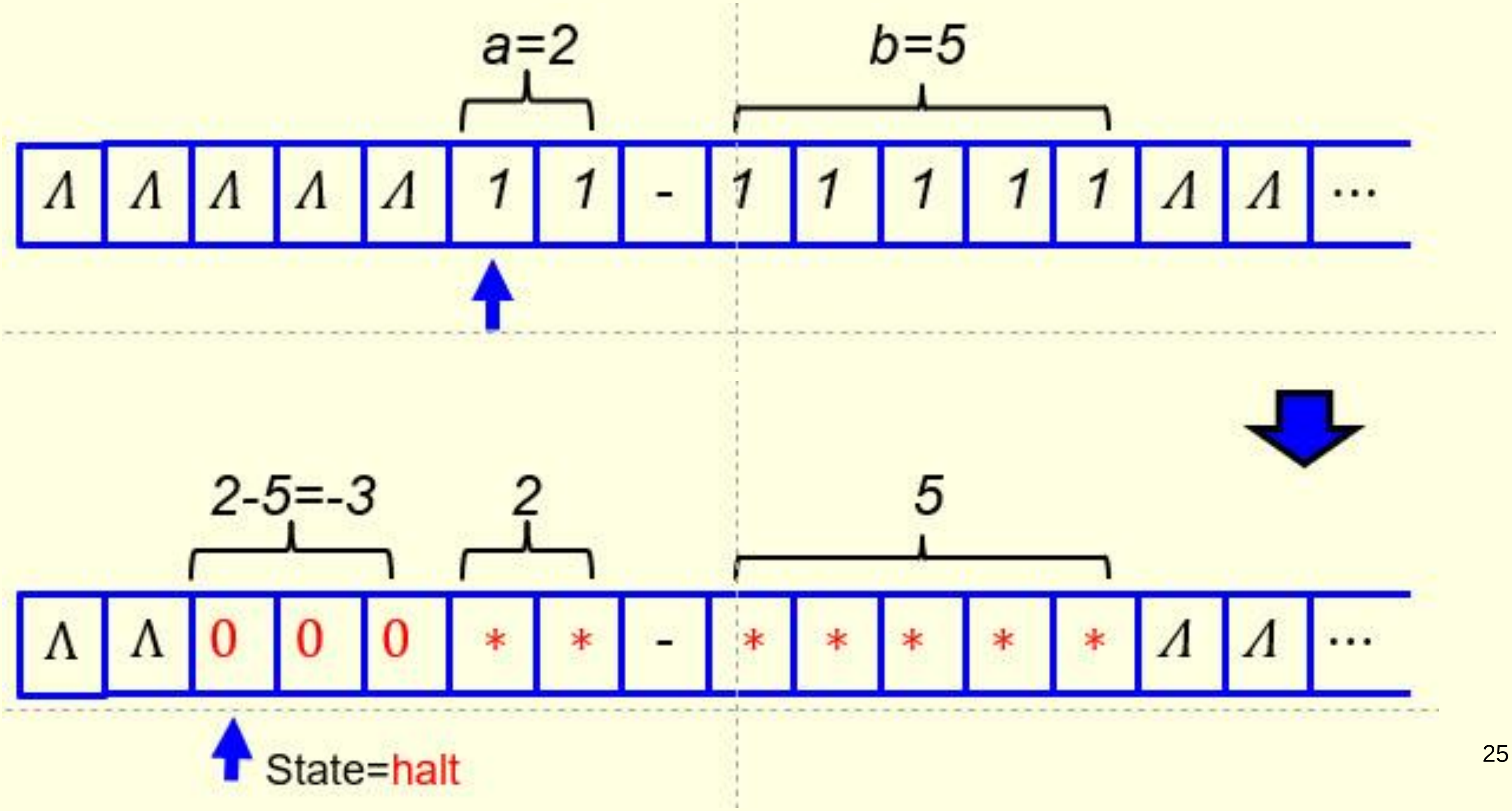


Note that in case *m=n*, the read/write head, at the end, will point at the left most '*'

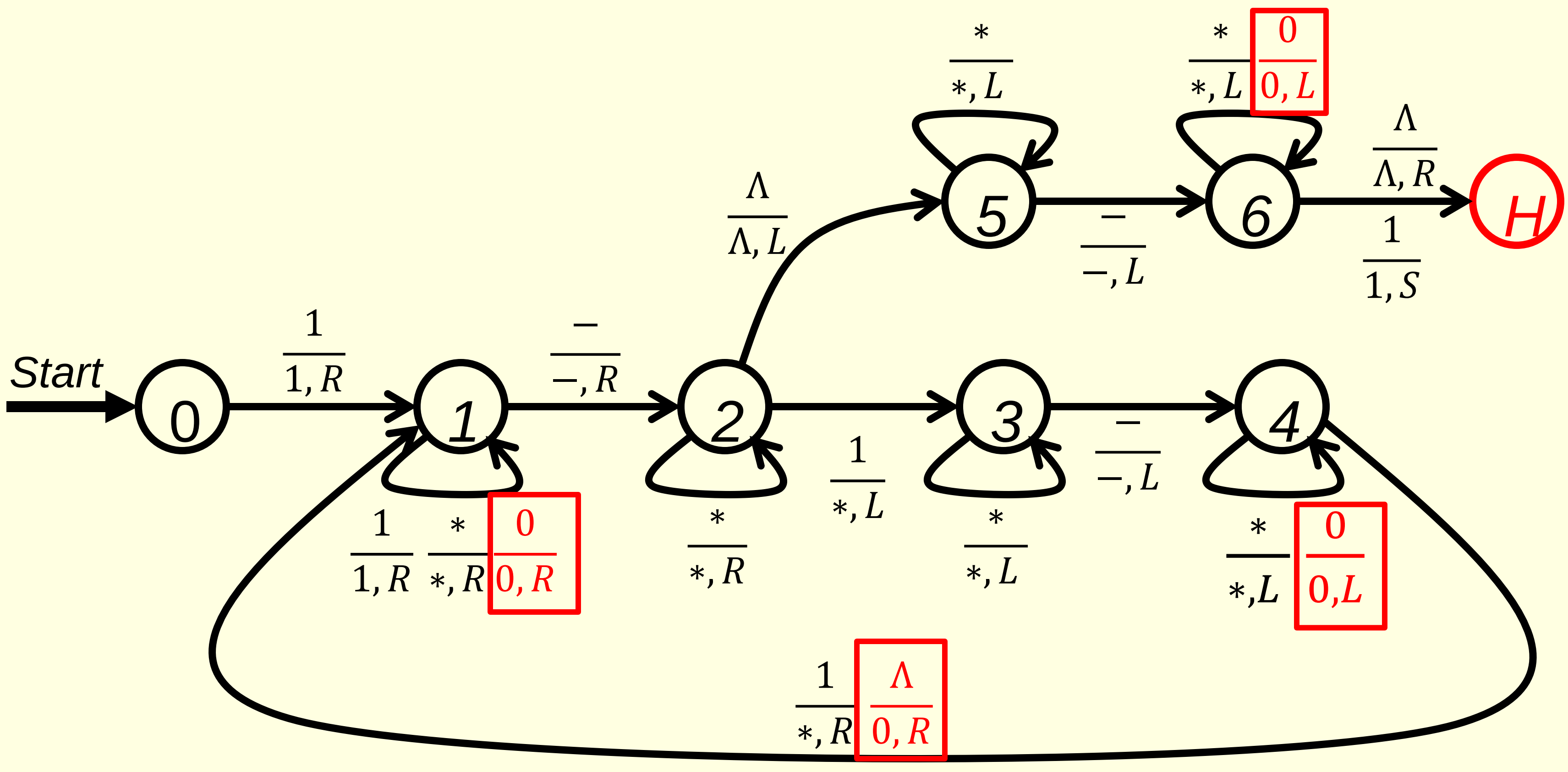


Question 1:

Can the above TM be modified to do subtraction $f(m-n)=c$ even when n is bigger than m?

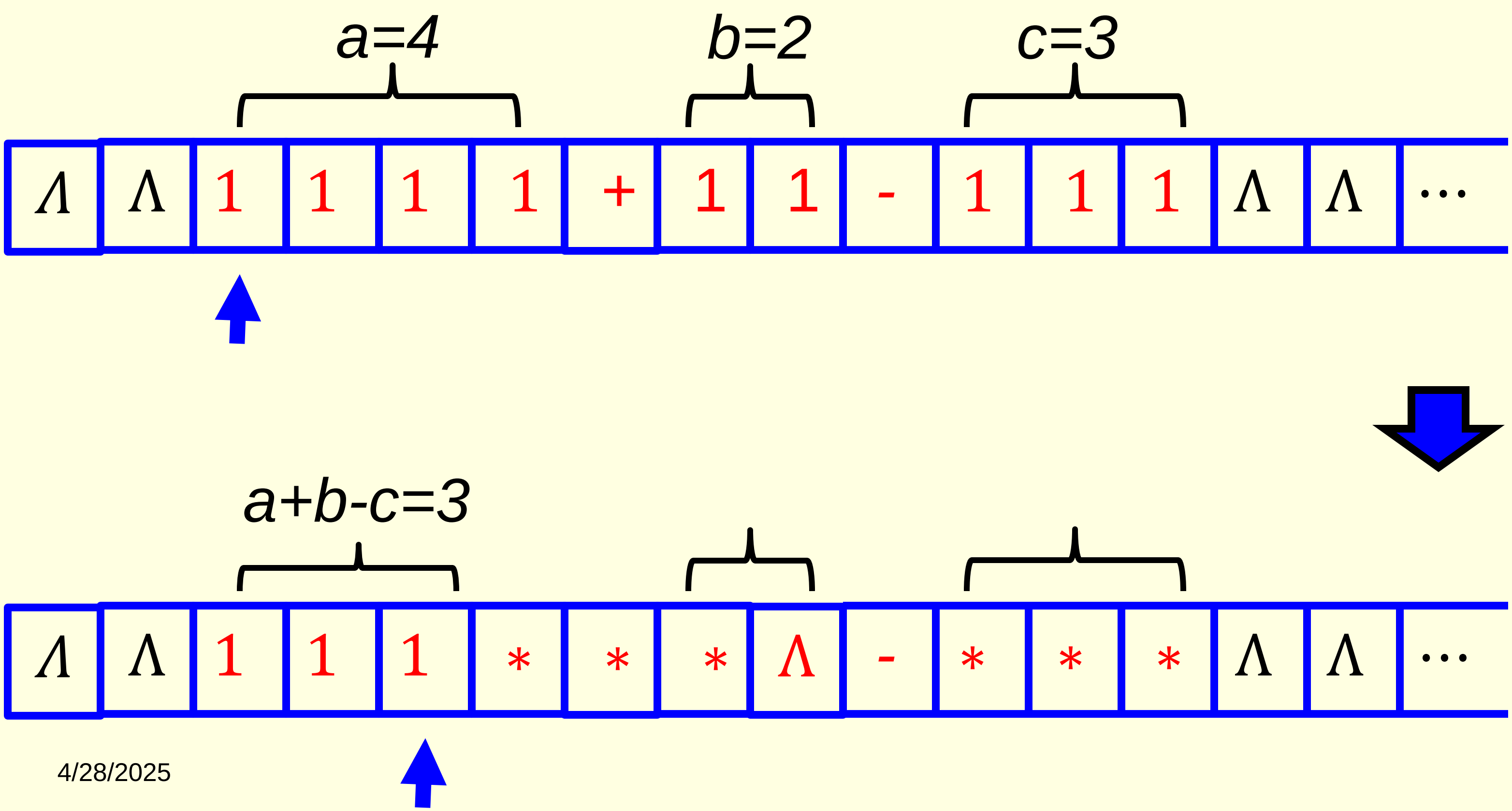


The TM will look like the one below (when m < n):



Question 2:

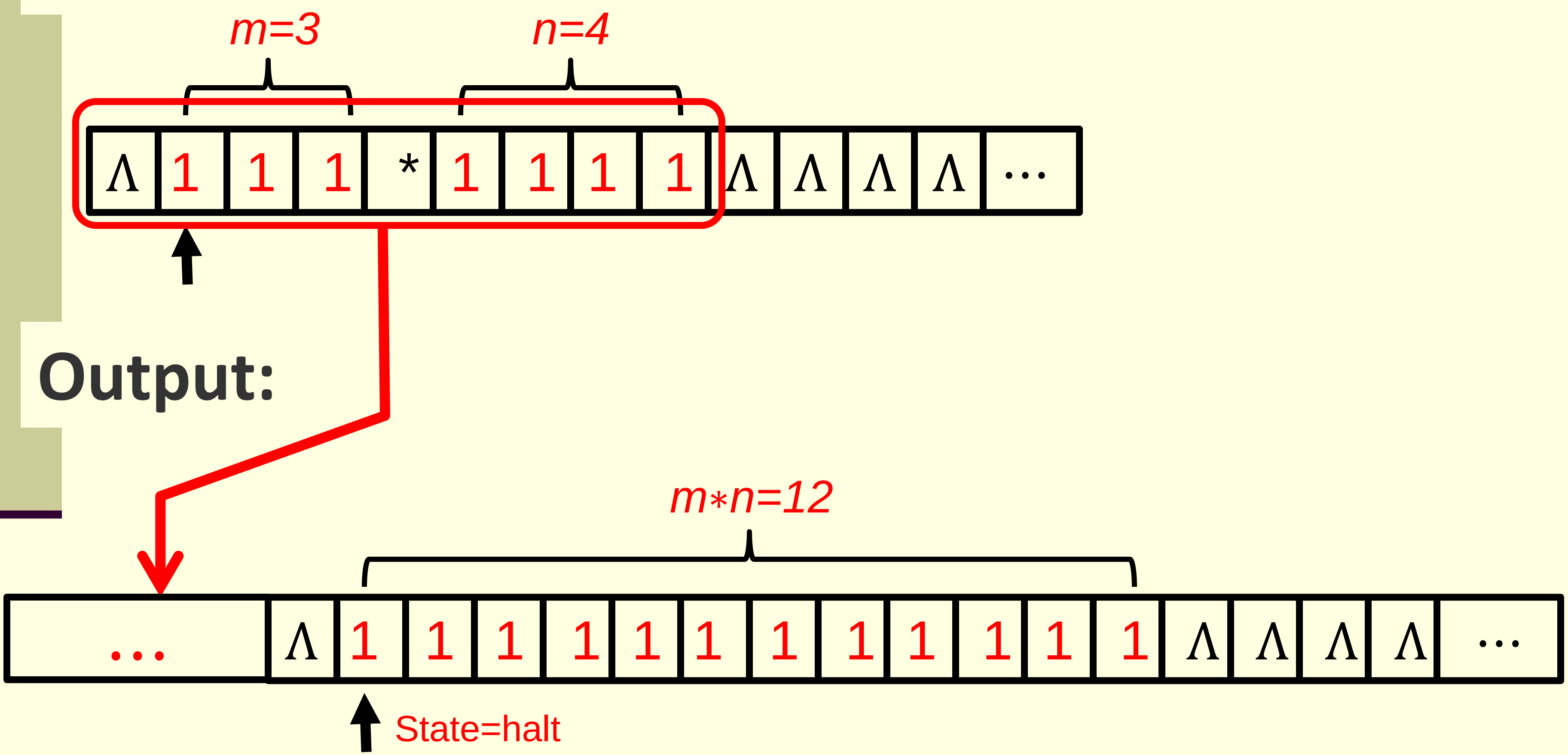
Given three non-zero positive unary numbers *a*, *b* and *c*, can a TM be built to carry out this function $f(a+b-c)$?



TM for the **multiplication function** for the unary number system

Input:

Output:



Basic concept: repeatedly attach *m* 1's to the end of the 2^{nd} string until *n* iterations have been done

What is multiplication?

Extended addition. Why?

$$4 \times 5 = \left.\begin{array}{r} 4 \\ 4 \\ 4 \\ 4 \\ +\ 4 \\ \hline \end{array}\right\} \begin{array}{c} 5 \\ \text{copies} \end{array} = \left.\begin{array}{r} 1\,1\,1\,1 \\ 1\,1\,1\,1 \\ 1\,1\,1\,1 \\ 1\,1\,1\,1 \\ +\ 1\,1\,1\,1 \\ \hline \end{array}\right\} \begin{array}{c} 5 \\ \text{copies} \end{array}$$

Basic concept: repeatedly attach *m* 1's to the end of the 2^{nd} string until *n* iterations have been done

What is multiplication?

Extended addition. Why?

$$
\begin{array}{r}
1\,1\,1\,1 \\
\boxed{1\,1\,1\,1} \\
1\,1\,1\,1 \\
1\,1\,1\,1 \\
+\ 1\,1\,1\,1 \\
\hline
\end{array}
\quad = \quad
\begin{array}{l}
\ \ 1\,1\,1\,1\,1\,1\,1\,1 \\
\ \ \boxed{1\,1\,1\,1} \\
\ \ 1\,1\,1\,1 \\
+\ 1\,1\,1\,1 \\
\hline
\end{array}
\quad =
$$

Basic concept: repeatedly attach *m* 1’s to the end of the 2nd string until *n* iterations have been done

What is multiplication?

Extended addition. Why?

1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1

\+ 1 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Implementation:

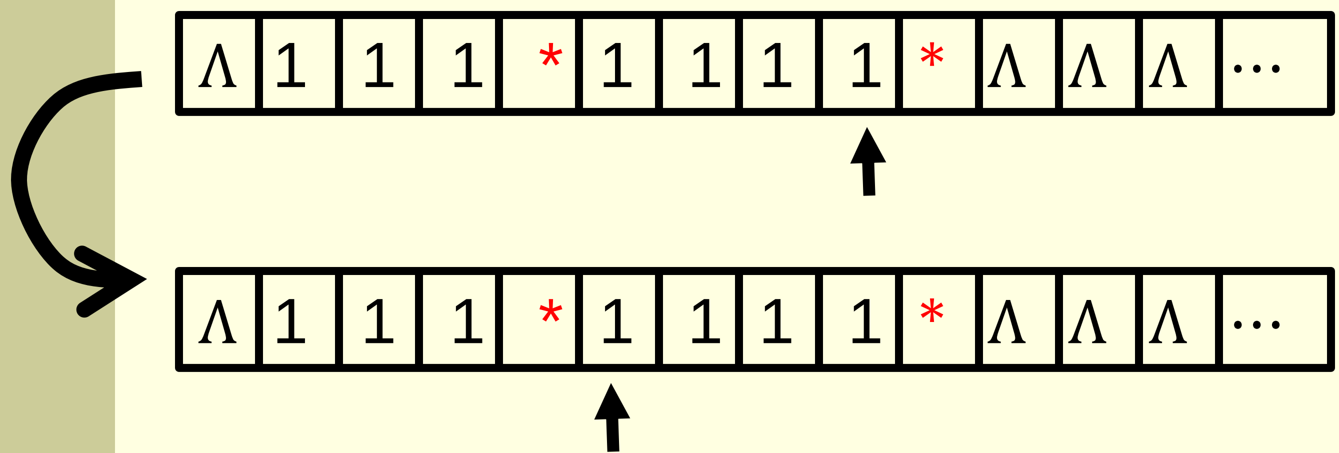
Input:

$$\Lambda\ \overbrace{1\ 1\ 1}^{m}\ *\ \overbrace{1\ 1\ 1\ 1}^{n}\ \Lambda\ \Lambda\ \Lambda\ \Lambda\ \cdots$$

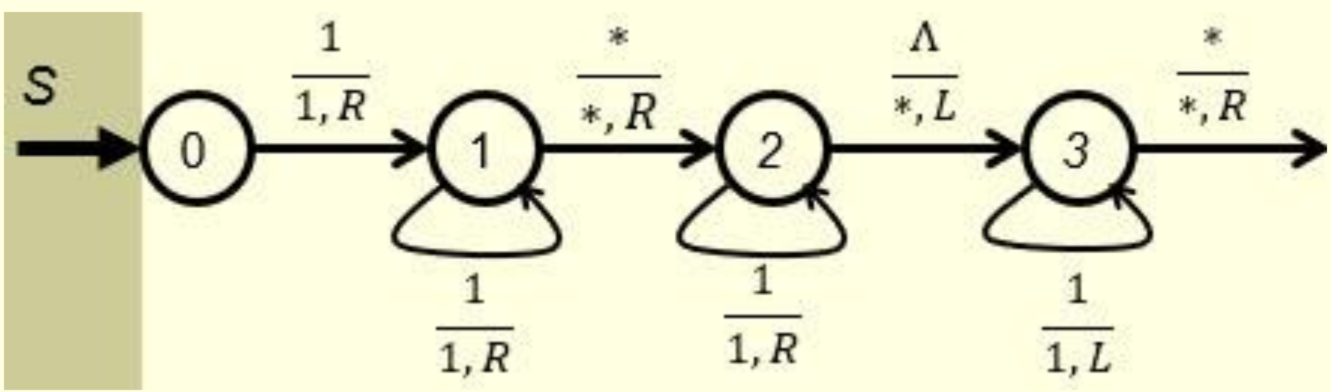
Step 1: move right, ignore all the 1's and *, to find the first Λ, convert it to * and turn left

$$\Lambda\ 1\ 1\ 1\ *\ 1\ 1\ 1\ 1\ *\ \Lambda\ \Lambda\ \Lambda\ \cdots$$

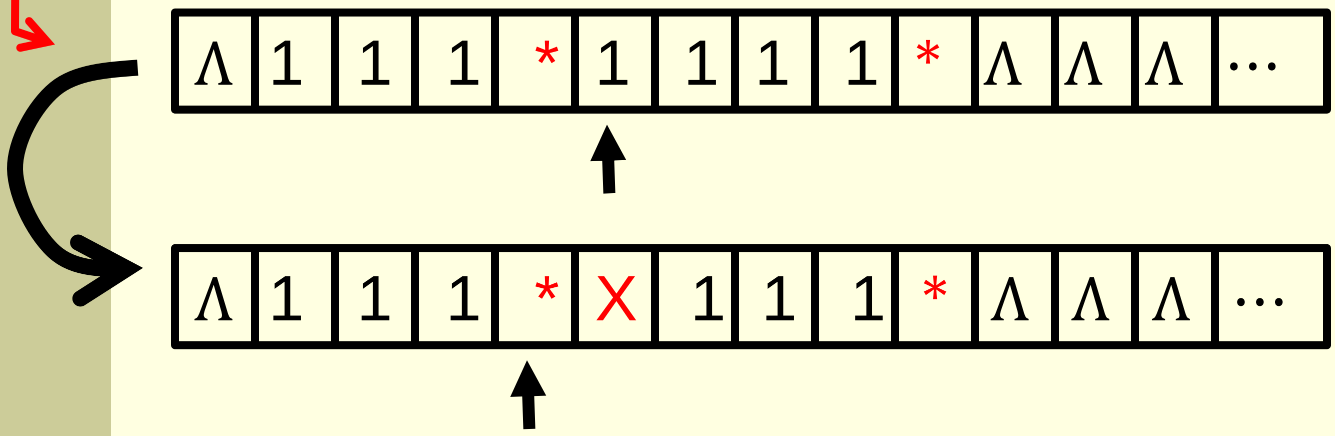
Step 2: move left, ignore all the 1’s, until a * is found, then turn right



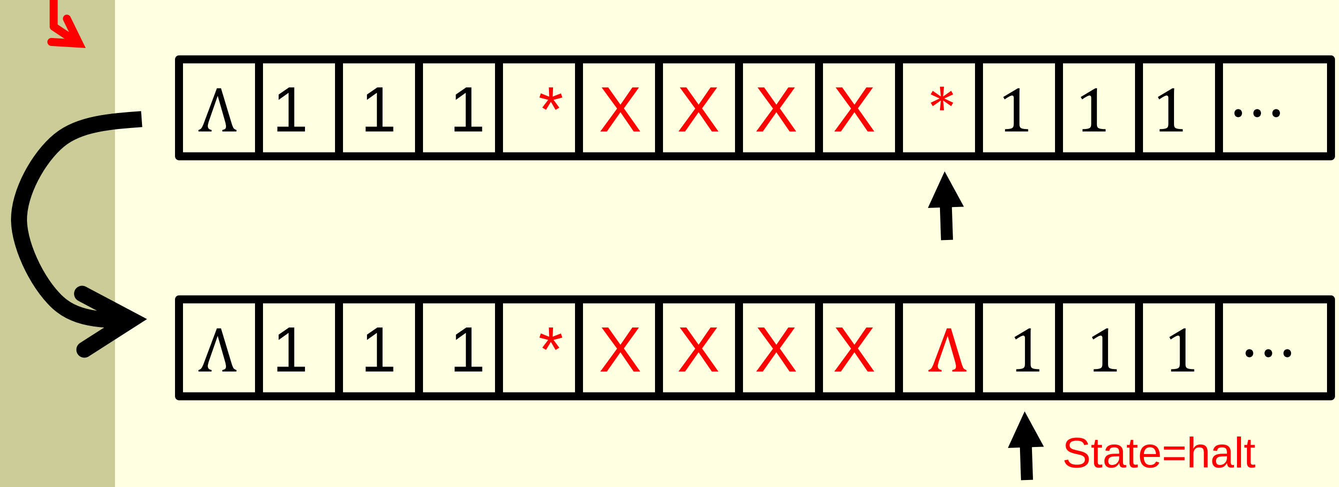
The portion of the TM for step 1 and step 2:



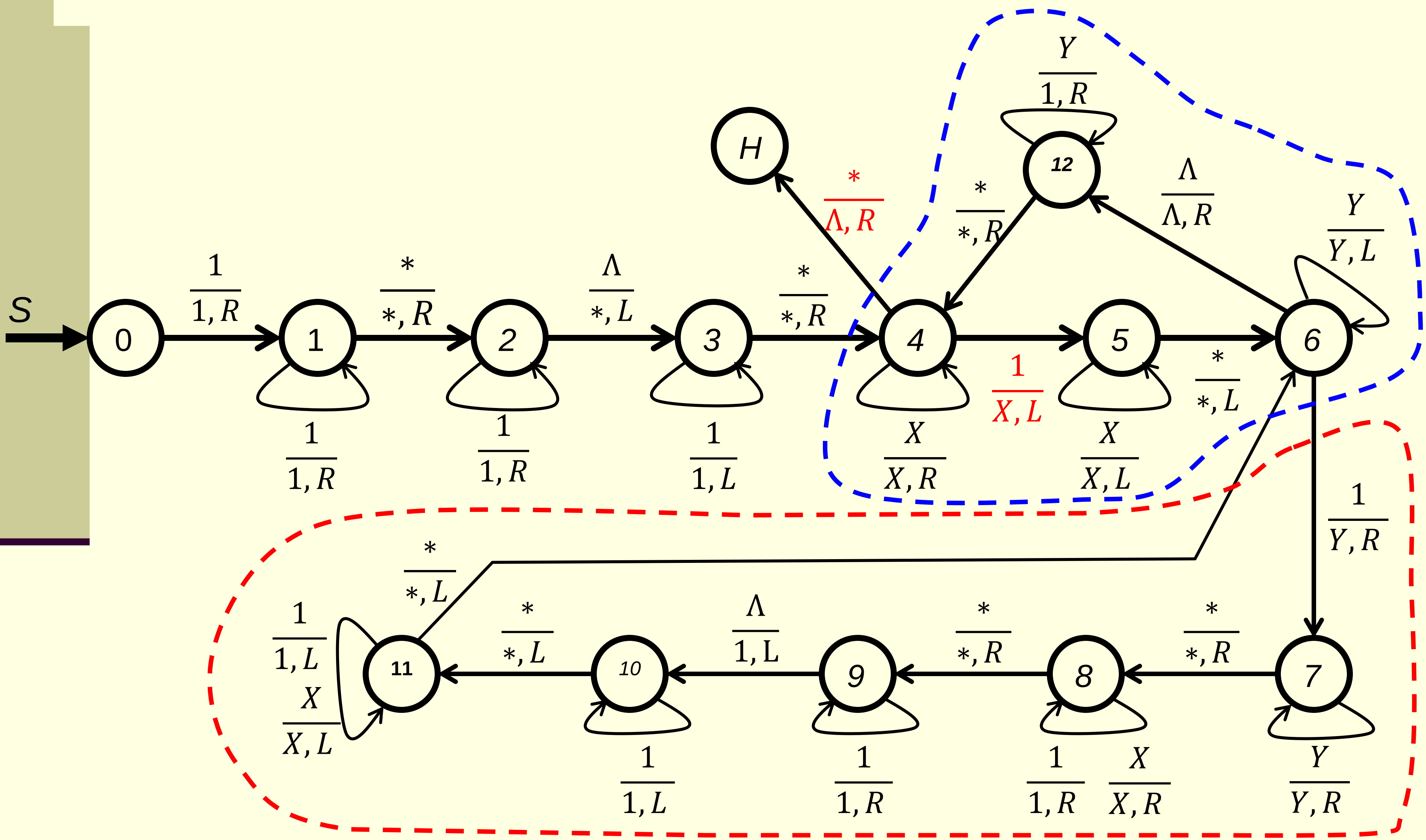
Step 3: move right, mark the first 1 reached with an ‘X’ and turn left. If no ‘1’ found but ‘*’ is reached, convert it to ‘Λ’, move one unit to the right and stop.



Step 3: move right, mark the first 1 reached with an ‘X’ and turn left. If no ‘1’ found but ‘*’ is reached, convert it to ‘Λ’, move one unit to the right and stop.



The TM will look like as follows :

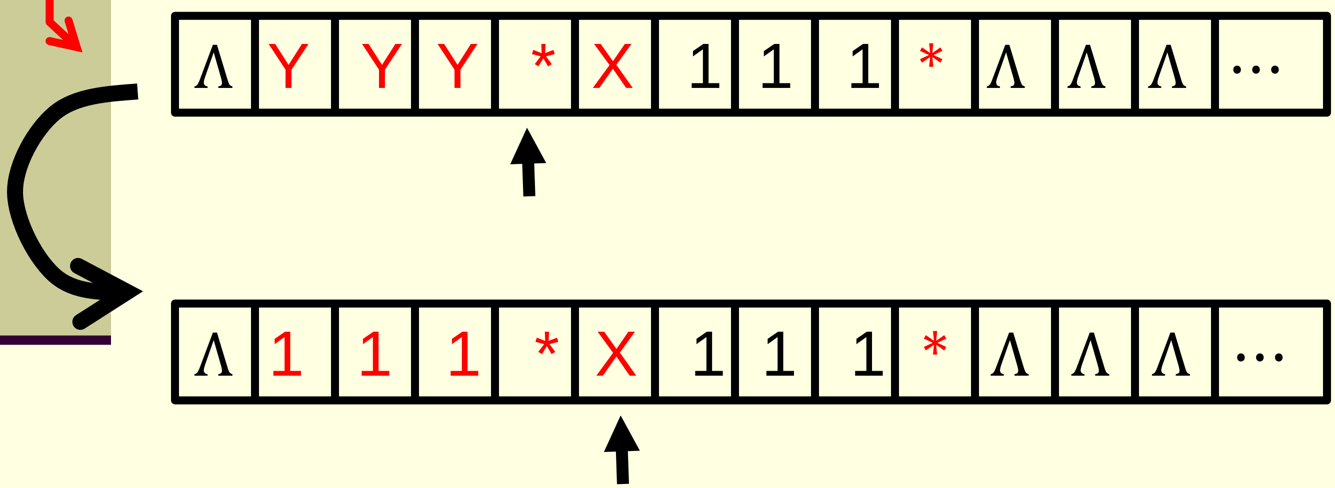


Step 4: move left, mark the first 1 reached with an ‘Y’ and turn right. If no ‘1’ found but a ‘Λ’ is reached, keep that ‘Λ’, turn right and change all the Y’s to 1, when we reach the ‘*’, keep that ‘*’, move one unit to the right and go to Step 3.

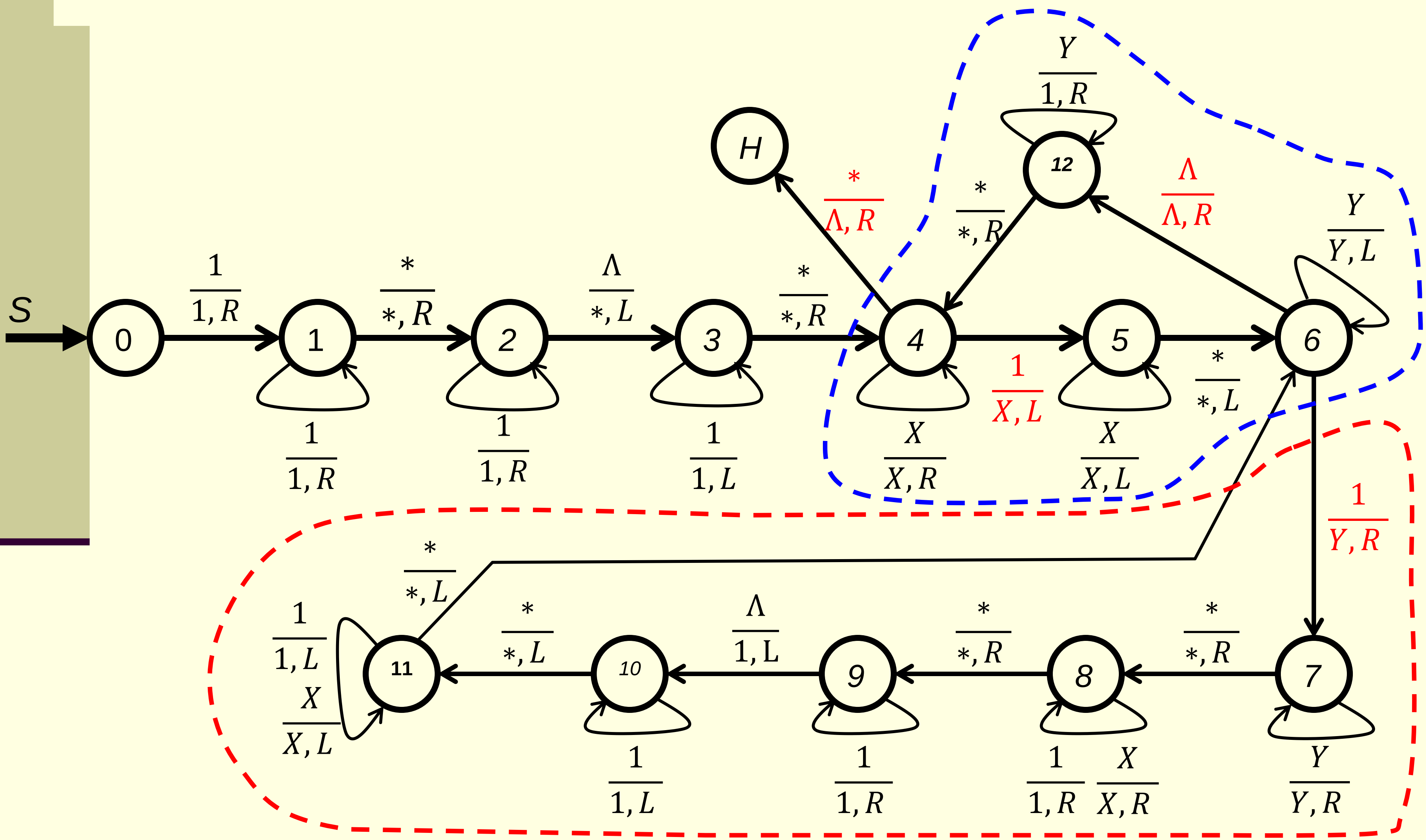
Λ	1	1	1	*	X	1	1	1	*	Λ	Λ	Λ	…

Λ	1	1	Y	*	X	1	1	1	*	Λ	Λ	Λ	…

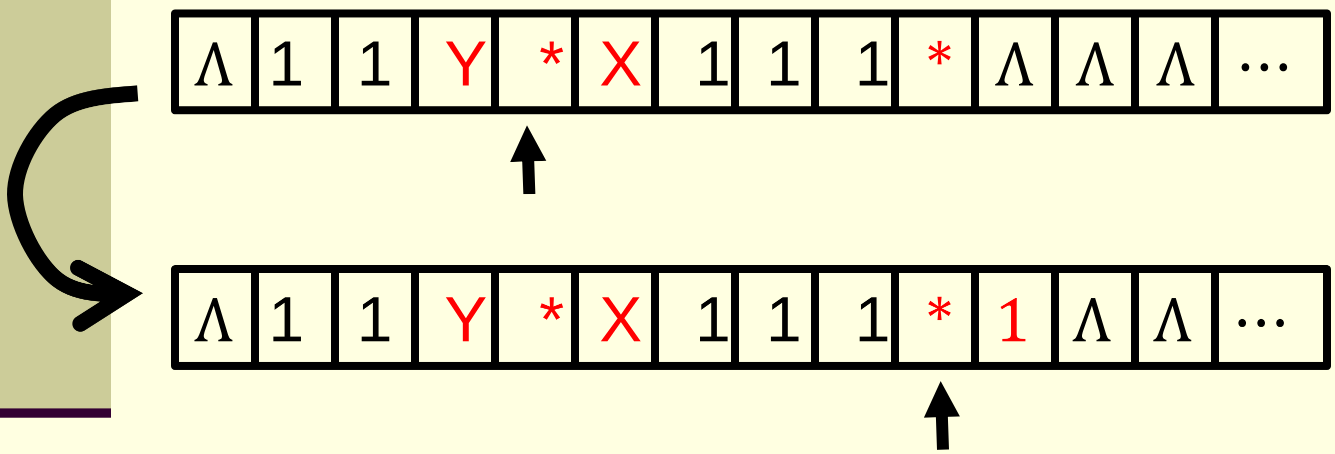
Step 4: move left, mark the first 1 reached with an ‘Y’ and turn right. If no ‘1’ found but a ‘Λ’ is reached, keep that ‘Λ’, turn right and change all the Y’s to 1, when we reach the ‘*’, keep that ‘*’, move one unit to the right and go to Step 3.



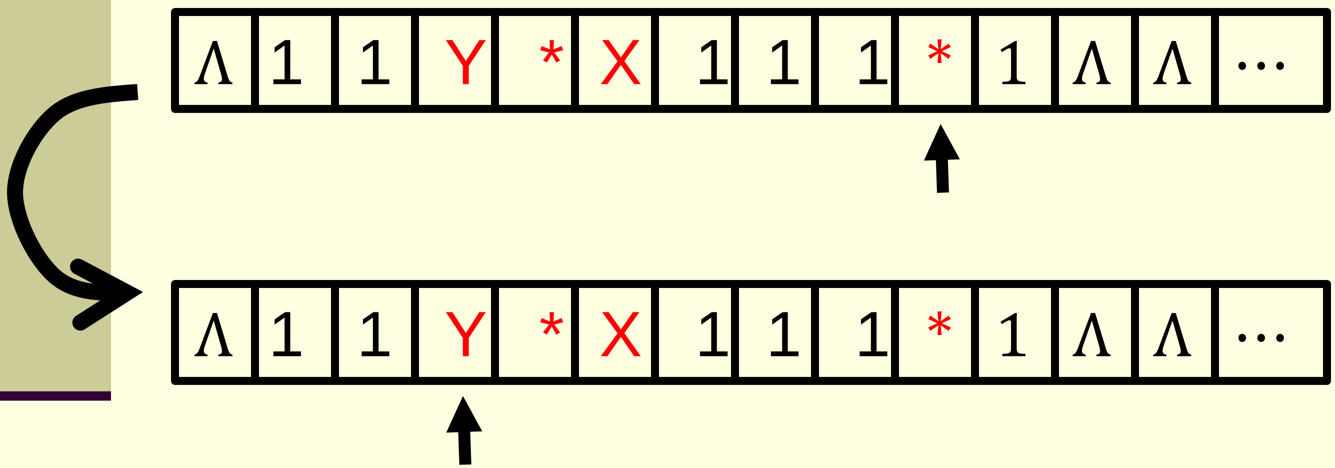
The TM will look like as follows :



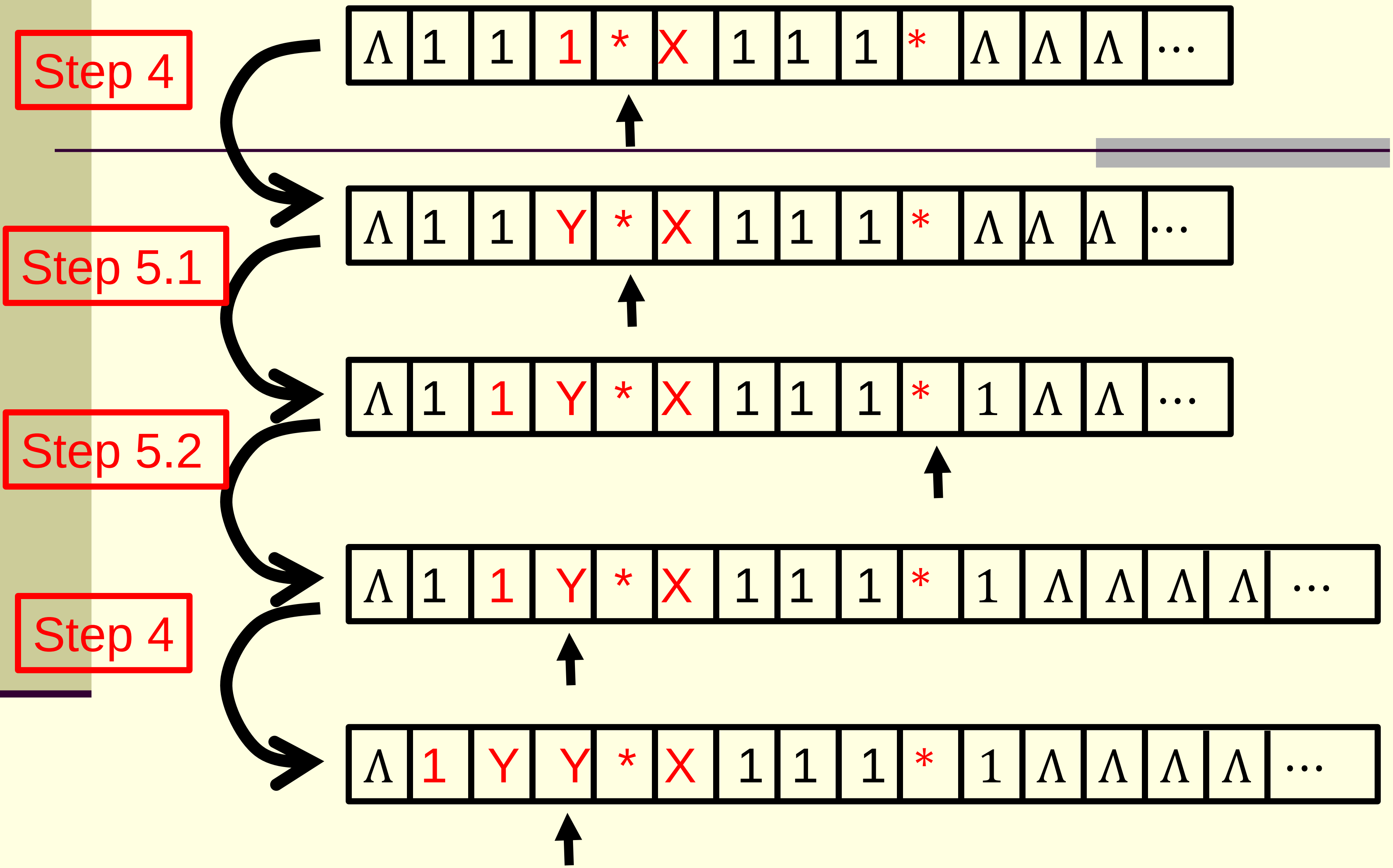
Step 5.1: move right, ignore all the Y's, the first '*' , all the X's, all the 1's, the second '*', all the 1's until we reach a 'Λ'. Convert that Λ to a 1 and then turn left.

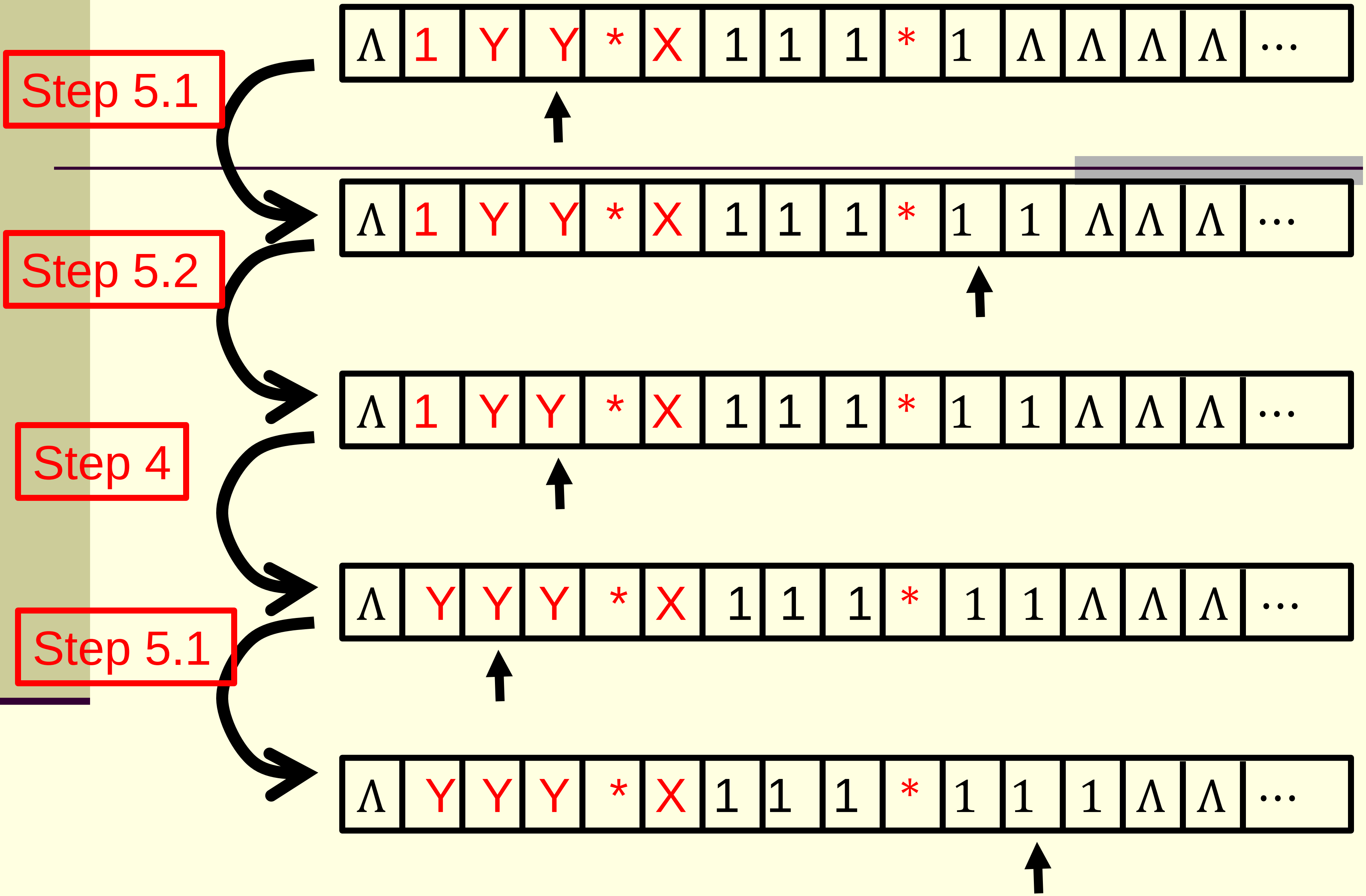


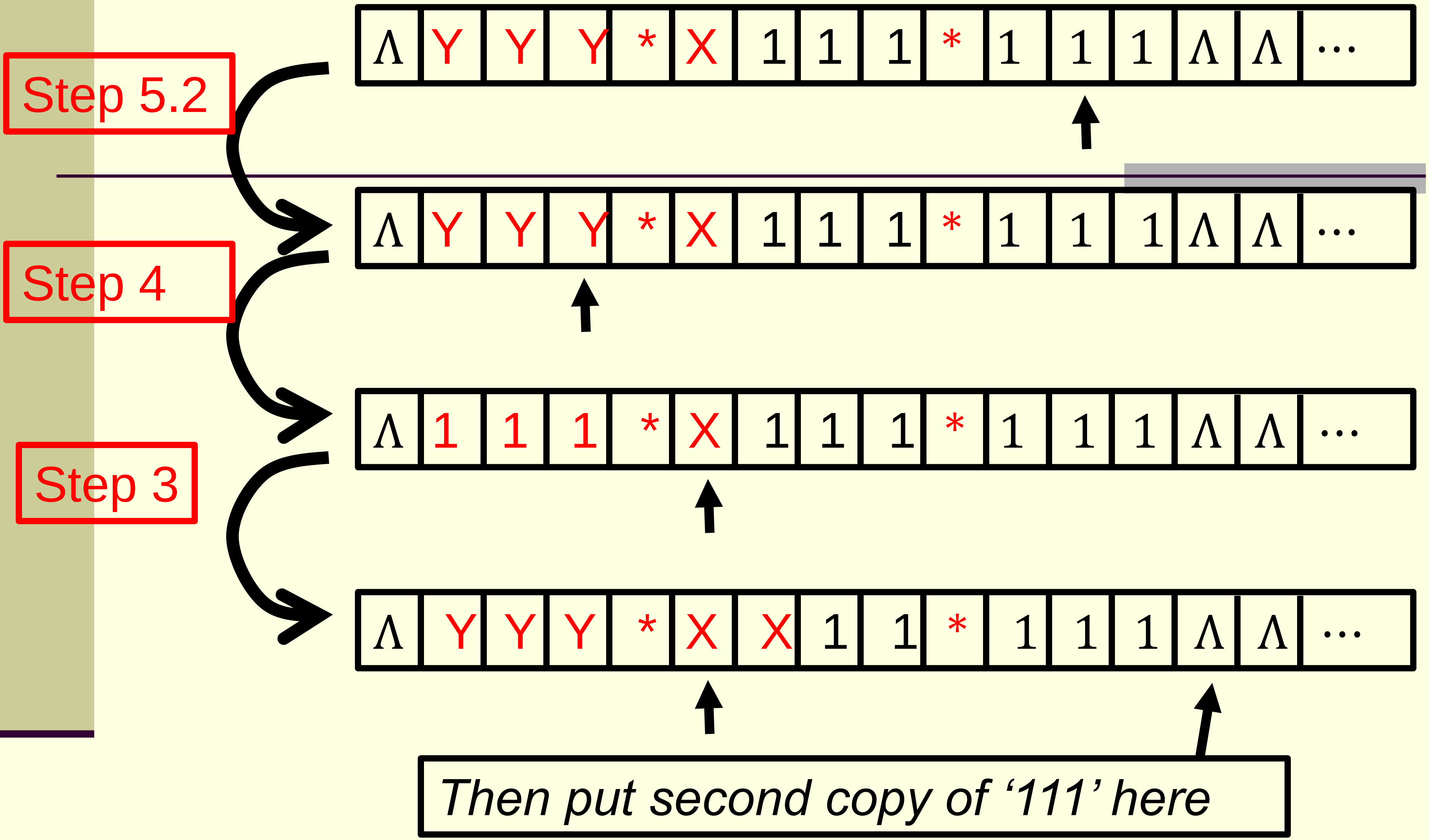
Step 5.2: Move left, ignore the first ‘*’, all the 1’s, all the X’s, until the second ‘*’ is reached. Keep that ‘*’ and move one unit to the left and go to Step 4.



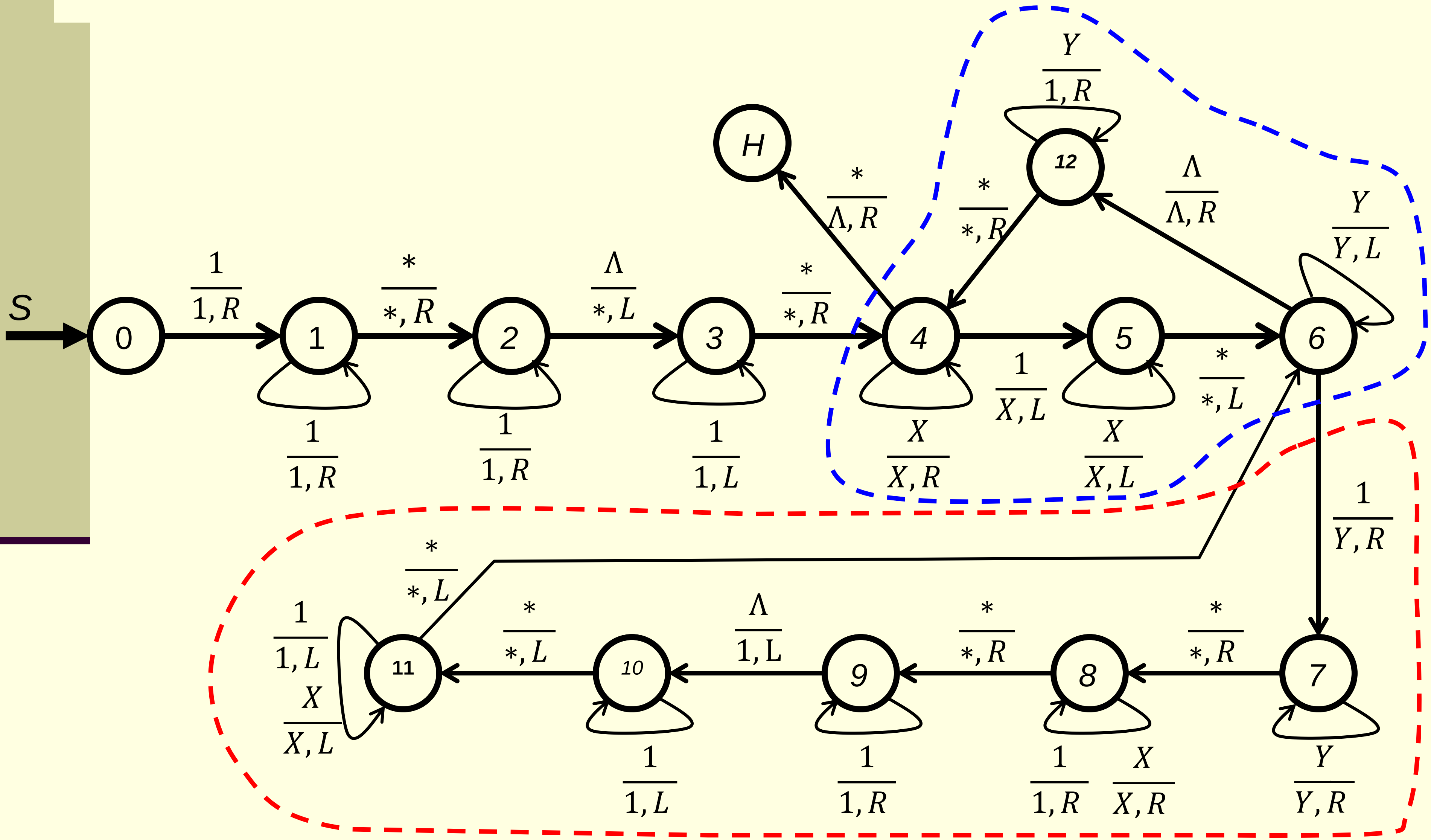
From slide 33




Λ 1 Y Y * X 1 1 1 * 1 Λ Λ Λ Λ …
Step 5.1
Λ 1 Y Y * X 1 1 1 * 1 1 Λ Λ Λ …
Step 5.2
Λ 1 Y Y * X 1 1 1 * 1 1 Λ Λ Λ …
Step 4
Λ Y Y Y * X 1 1 1 * 1 1 Λ Λ Λ …
Step 5.1
Λ Y Y Y * X 1 1 1 * 1 1 1 Λ Λ …

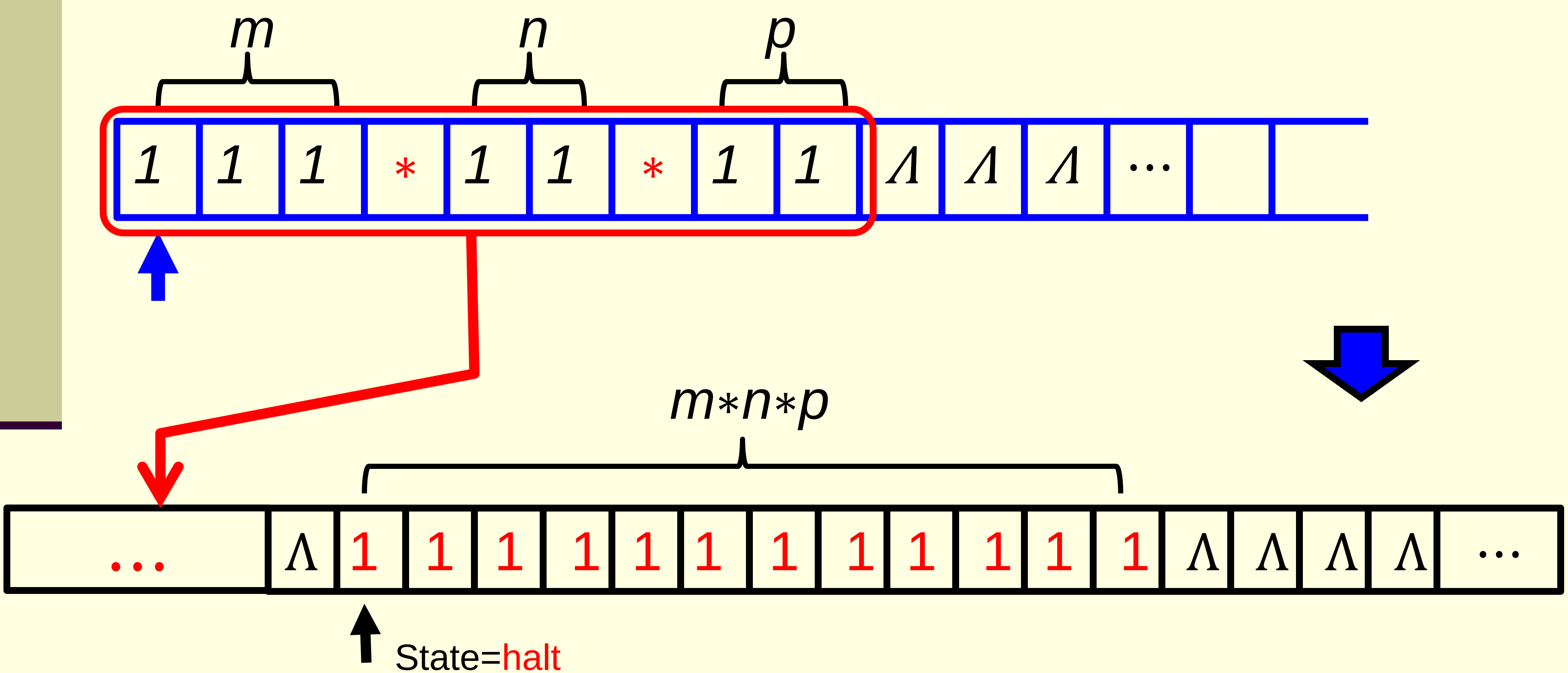

Step 5.2
Λ Y Y Y * X 1 1 1 * 1 1 1 Λ Λ …
Step 4
Λ Y Y Y * X 1 1 1 * 1 1 1 Λ Λ …
Step 3
Λ 1 1 1 * X 1 1 1 * 1 1 1 Λ Λ …
Λ Y Y Y * X X 1 1 * 1 1 1 Λ Λ …
Then put second copy of ‘111’ here

The TM will look like as follows :



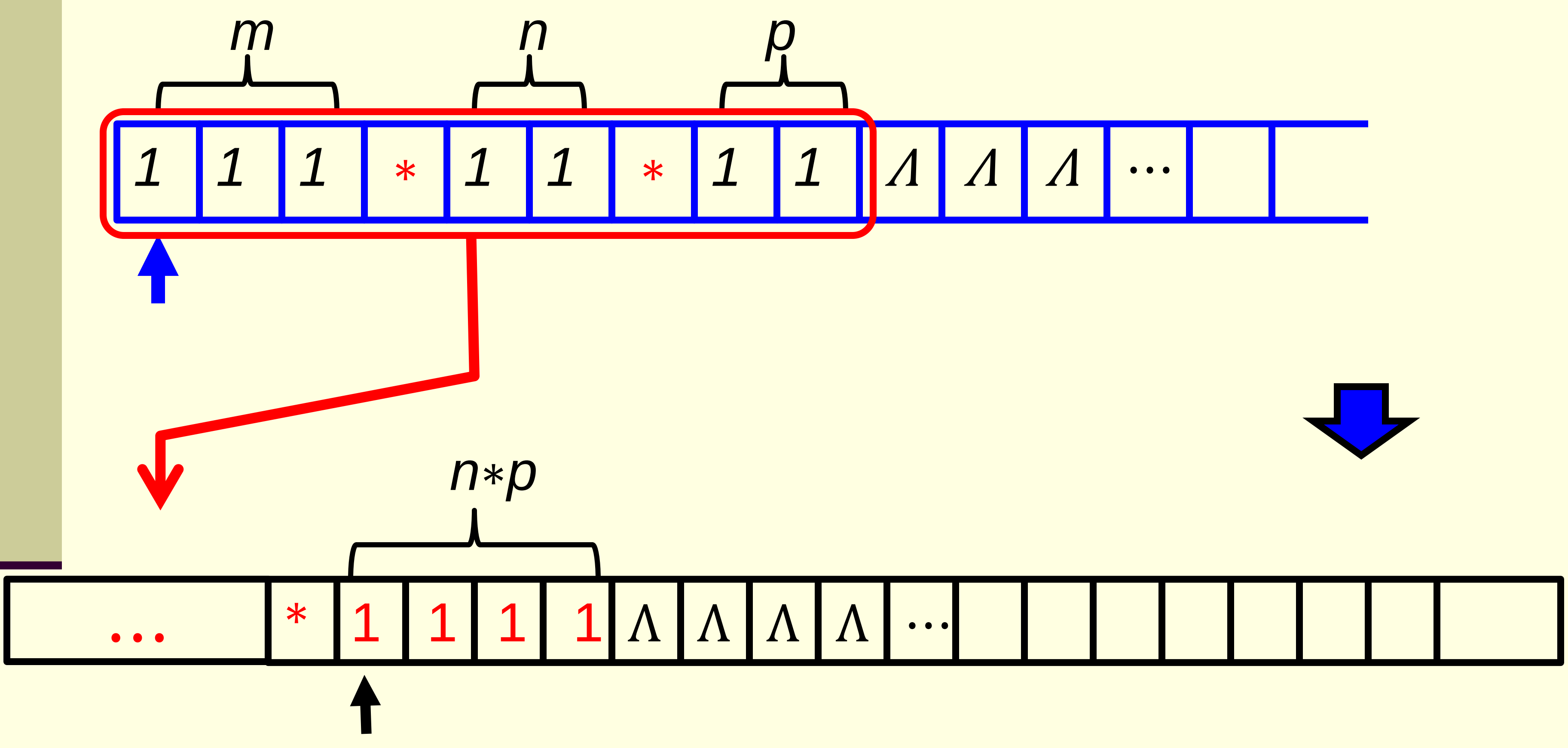
Question:

Can the above TM be modified to do multiplication of three numbers, four numbers, ..., n numbers in unary form directly?



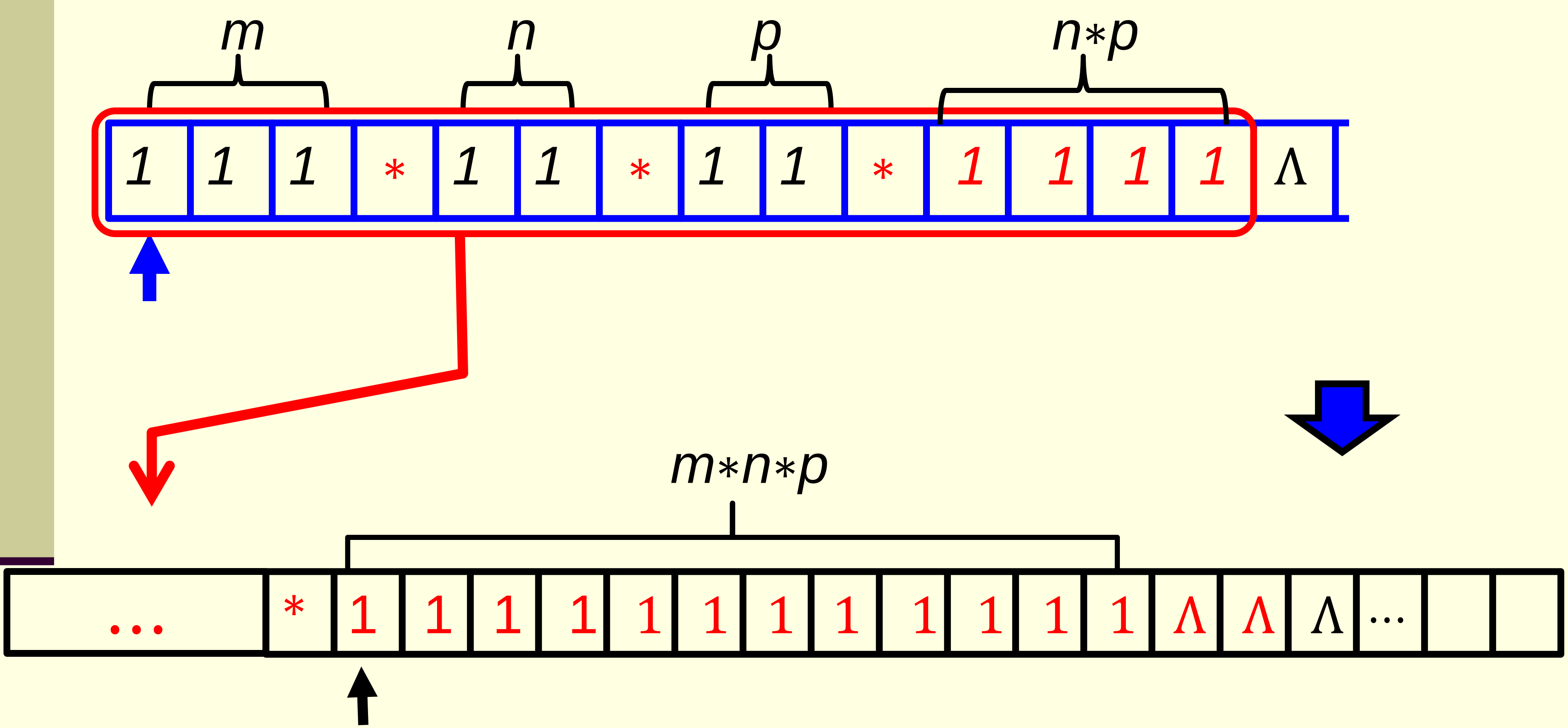
Solution:

Step 1



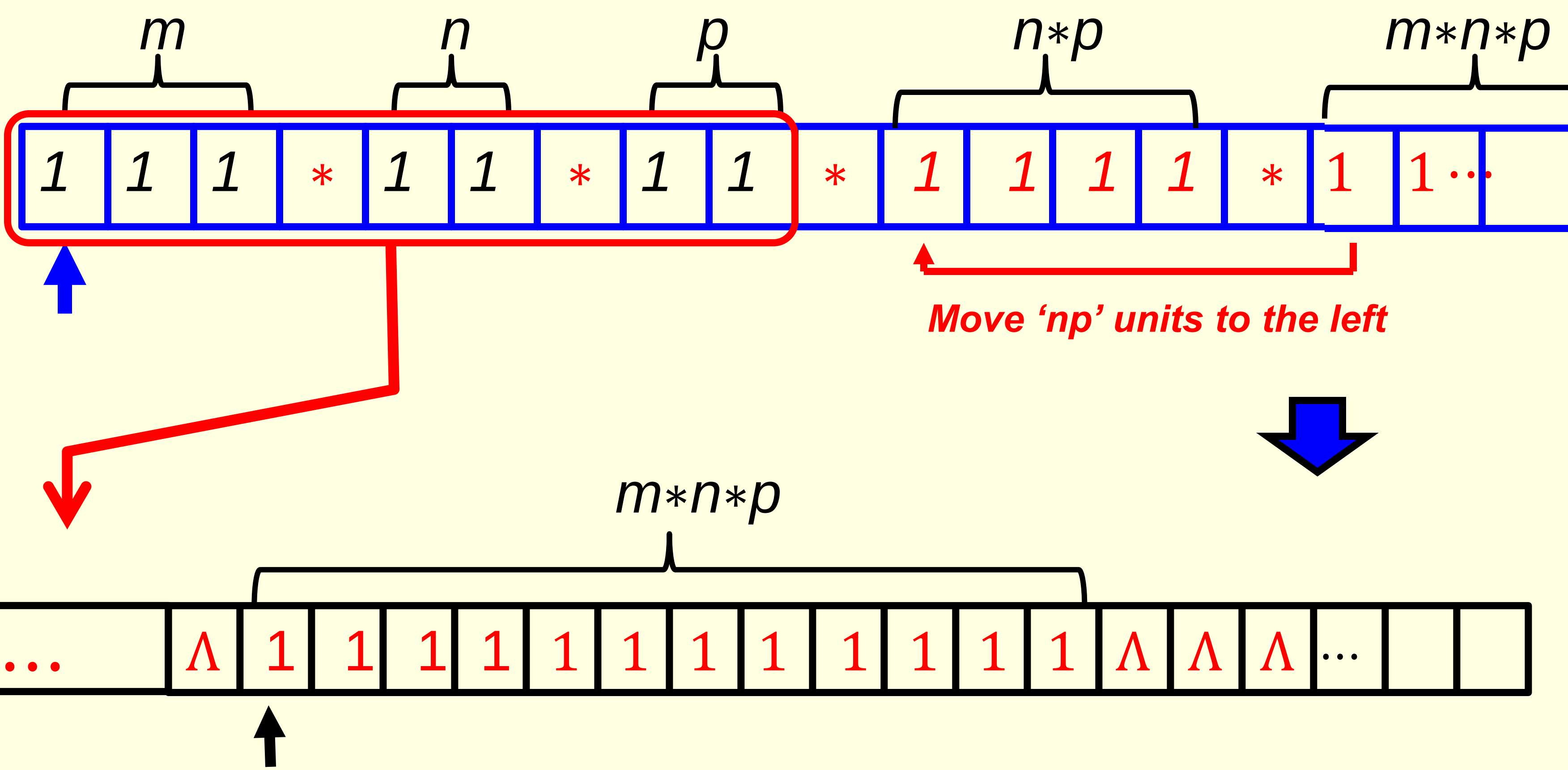
Solution:

Step 2



Solution:

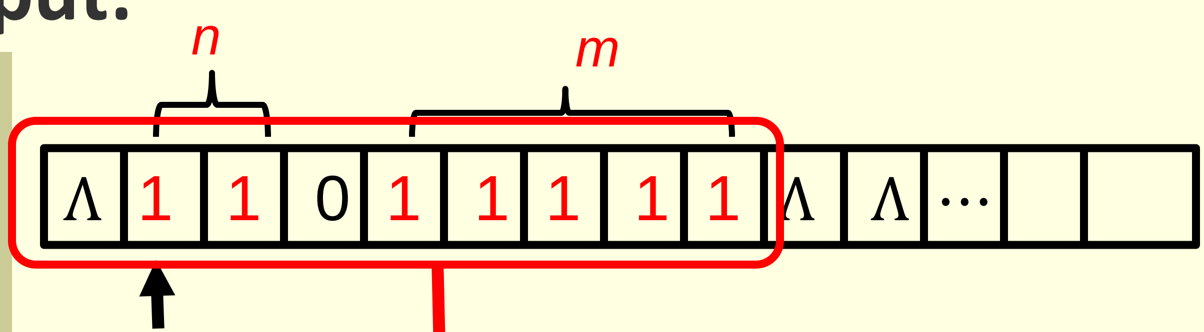
Step 3



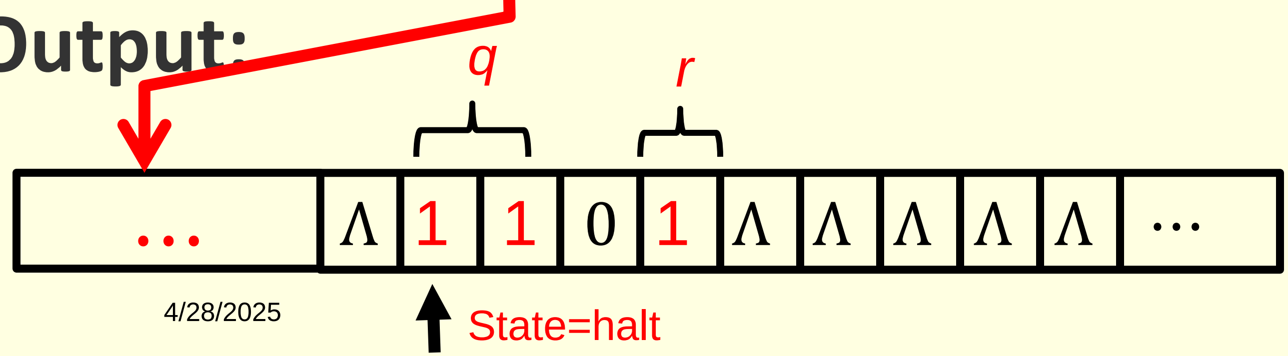
TM for the **division function** for the unary number system

Develop a TM for the division of two unary numbers m and n such that f(m/n) = q + r where q is the quotient and r is the remainder.

Input:



Output:



Basic concept: divide the unary representation of *m* into subgroups of length *n*; the number of subgroups is the quotient and the excess part is the remainder.

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n=4}$ $\qquad$ $\overbrace{1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1}^{m=10}$

Q = Λ

R = Λ

If *n <= (current) m* do

- create a subgroup of length *n*
- add 1 to Q
- erase this subgroup from *m*

Otherwise, set the current *m* to be R

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n}$ $\qquad$ $\overbrace{1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1}^{m}$

Q = Λ

R = Λ

current m

4 <= 10, so

(Increment Q by 1)

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n}$ $\qquad$ $\overbrace{\boxed{1\ \ 1\ \ 1\ \ 1}\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1}^{m}$

(Create a subgroup of length 4)

Q = 1

R = Λ

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n}$ $\qquad$ $\overbrace{\boxed{*\ \ *\ \ *\ \ *}\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1}^{m}$

(erase this subgroup from m)

Q = 1

R = Λ

$\overbrace{1\ 1\ 1\ 1}^{n}$ $\qquad$ $\overbrace{\boxed{*\ *\ *\ *}\ 1\ 1\ 1\ 1\ 1\ 1}^{m}$

Q = 1

R = Λ

current m

4 <= 6, so

(Increment Q by 1)

$\overbrace{1\ 1\ 1\ 1}^{n}$ $\qquad$ $\overbrace{*\ *\ *\ *\ \boxed{1\ 1\ 1\ 1}\ 1\ 1}^{m}$

(Create a subgroup of length 4)

Q = 2

R = Λ

$\overbrace{1\ 1\ 1\ 1}^{n}$ $\qquad$ $\overbrace{*\ *\ *\ *\ \boxed{*\ *\ *\ *}\ 1\ 1}^{m}$

(erase this subgroup from m)

Q = 2

R = Λ

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n}$ $\qquad$ $\overbrace{\boxed{*\ *\ *\ *\ \ *\ *\ *\ *}\ 1\ \ 1}^{m}$

$\boxed{\begin{array}{l} Q = 2 \\ R = \Lambda \end{array}}$

current m

4 > 2, so

$\overbrace{1\ \ 1\ \ 1\ \ 1}^{n}$ $\qquad$ $\overbrace{*\ *\ *\ *\ \ *\ *\ *\ *\ \boxed{1\ \ 1}}^{m}$

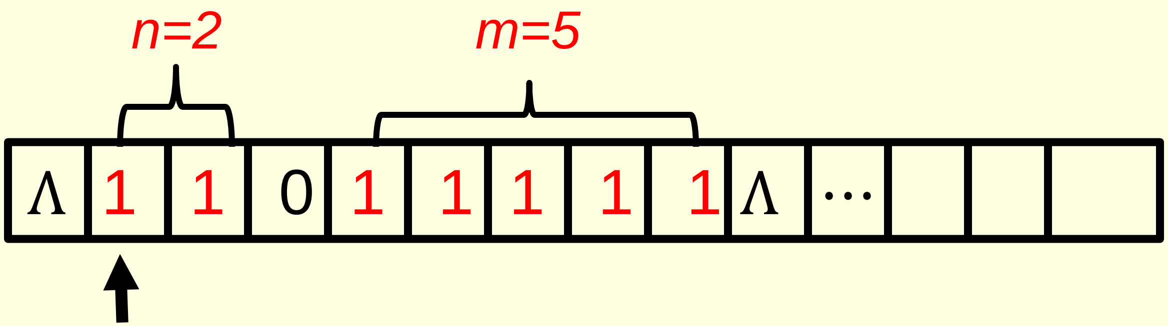
$\boxed{\begin{array}{l} Q = 2 \\ R = 2 \end{array}}$

(set this part as R)

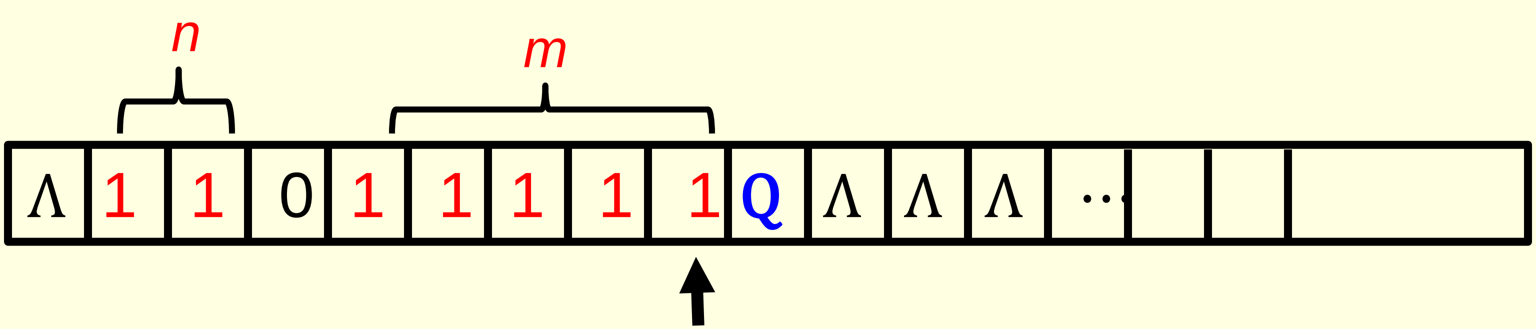
And then stop

Implementation:

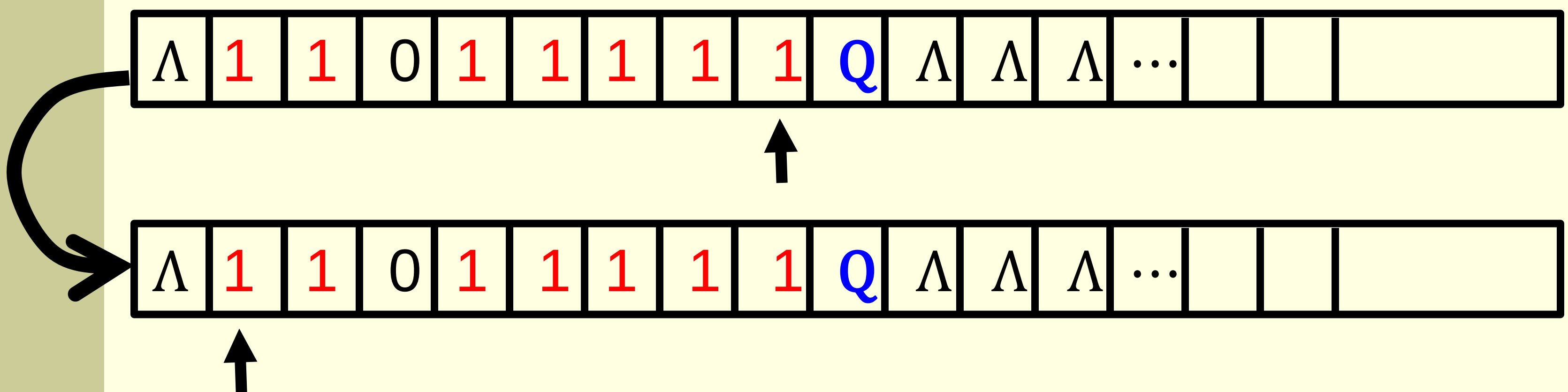
Input:



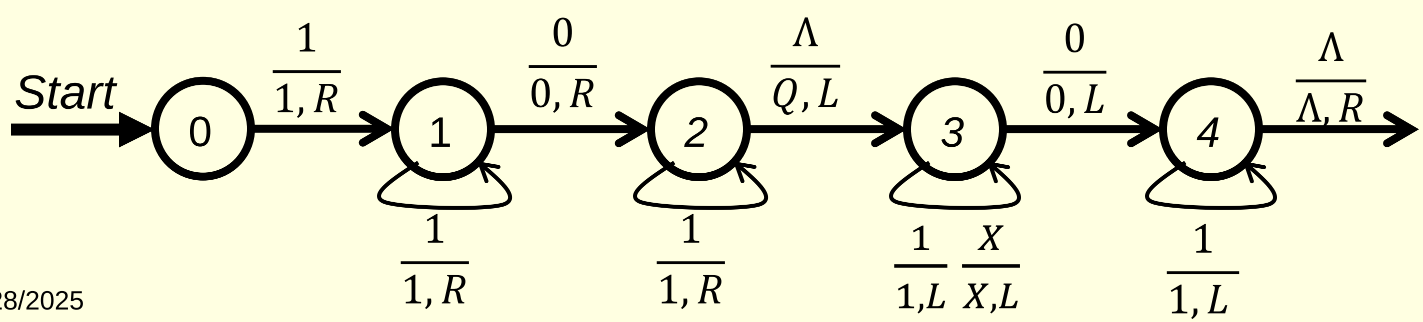
Step 1.1: move right, ignore all the 1's and 0, to find the first Λ, convert it to Q and turn left



Step 1.2: move left, ignore all the 1’s, X’s and 0 until a Λ is found, then turn right



The portion of the TM for Step 1.1 and Step 1.2:



Step 2 (test if n > (current) m)

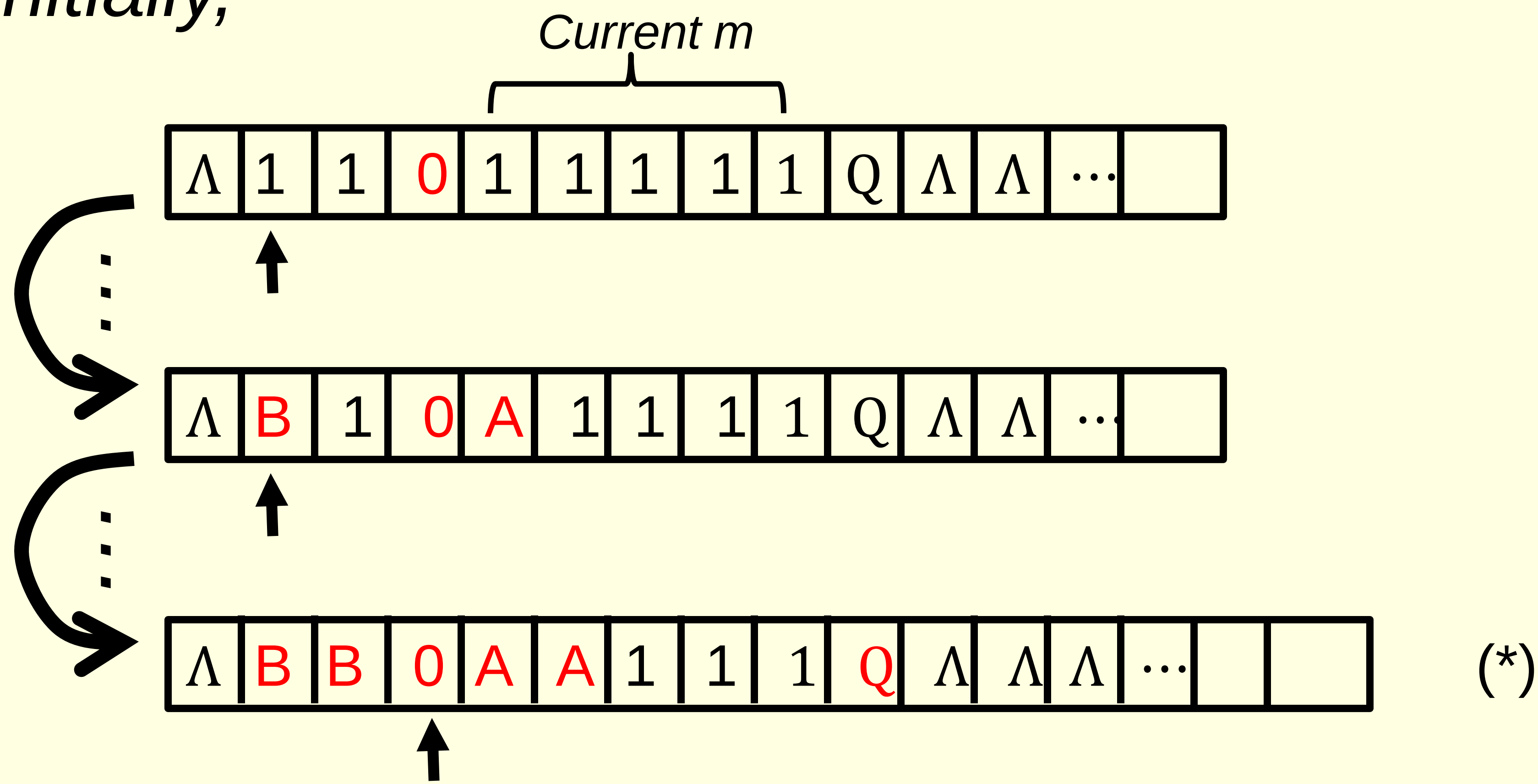
Repeat the following process: mark a 1 in n with a B then mark an available 1 in m with an A.

If all the available 1's in m are marked before all the 1's in n are marked (i.e., n > (current) m), then go to Step 4 (to format an output and then stop).

Otherwise (i.e., n <= (current) m), go to Step 3.

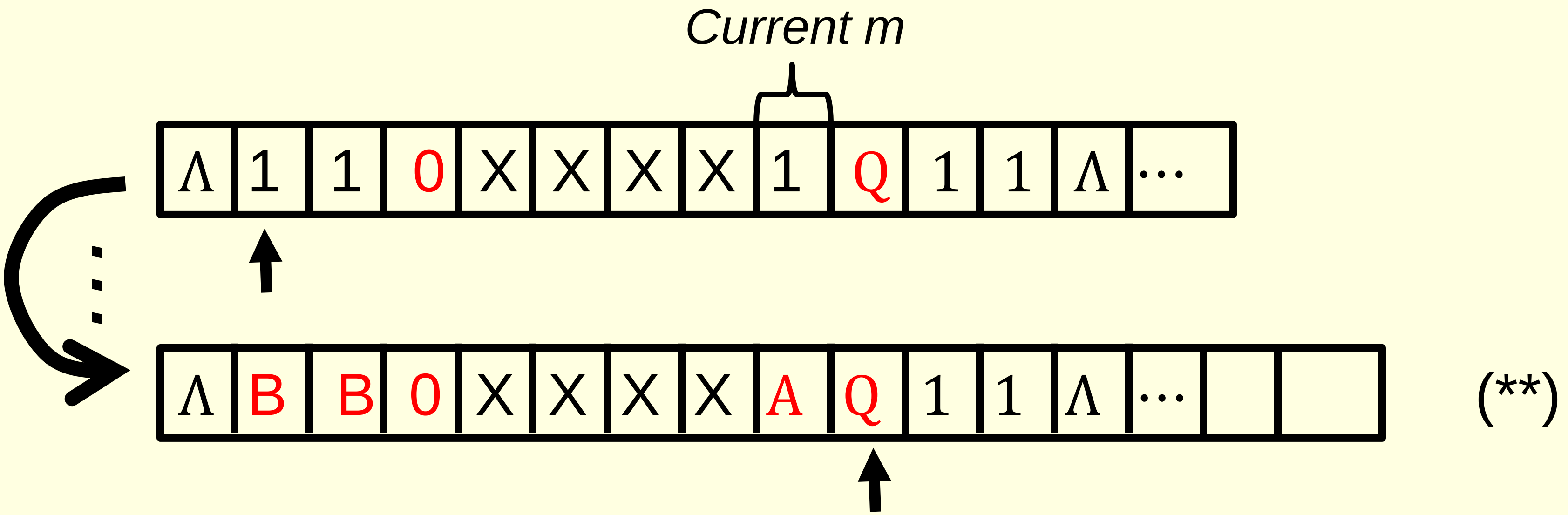
i.e., there are still some unmarked 1's in m after marking all the 1's in n (see the example in the next slide)

Initially,



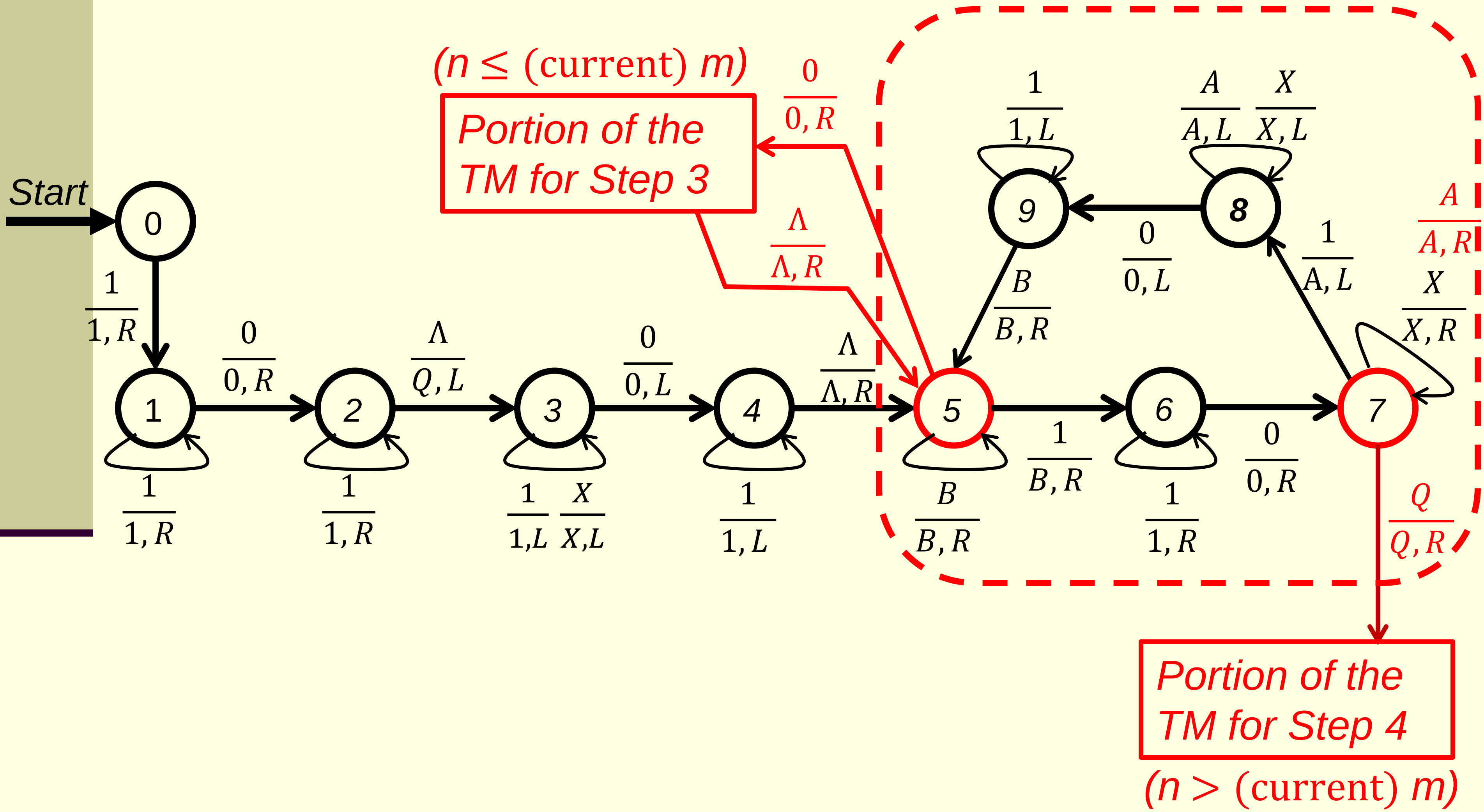
No more 1's in n to mark, but there are still some unmarked 1's in m (i.e., n <= current m), so go to Step 3

After a while,



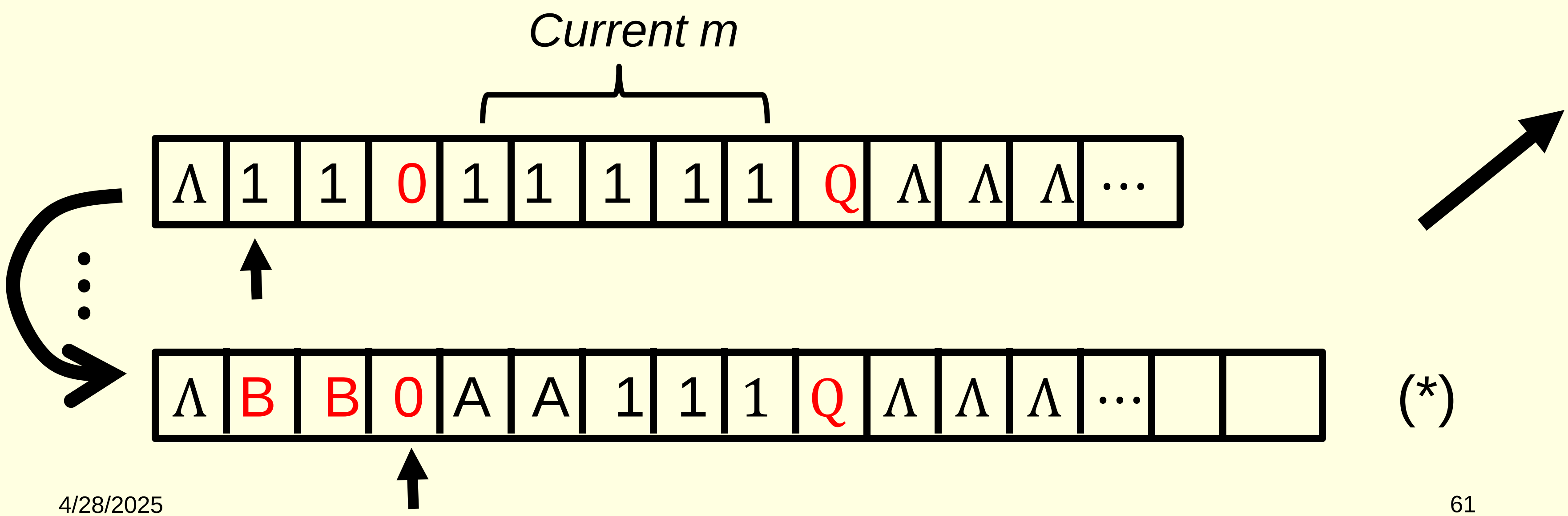
No available 1 in current m to mark/match for the 1 just marked in n (i.e., n > (current) m), so go to Step 4.

The portion of the TM for Step 1 and Step 2 is as follows :



Step 3: This step marks n 1's of m with X's (so the next 'current m' has n less available 1's than the current 'current m') and adds 1 to the quotient and then go to Step 2 (to perform the next (n <= (current) m) test).

From (*) in Step 2 :



Λ	B	B	0	A	A	1	1	1	Q	Λ	Λ	Λ	···		

(*)

Λ	B	B	0	X	X	1	1	1	Q	Λ	Λ	Λ	···		

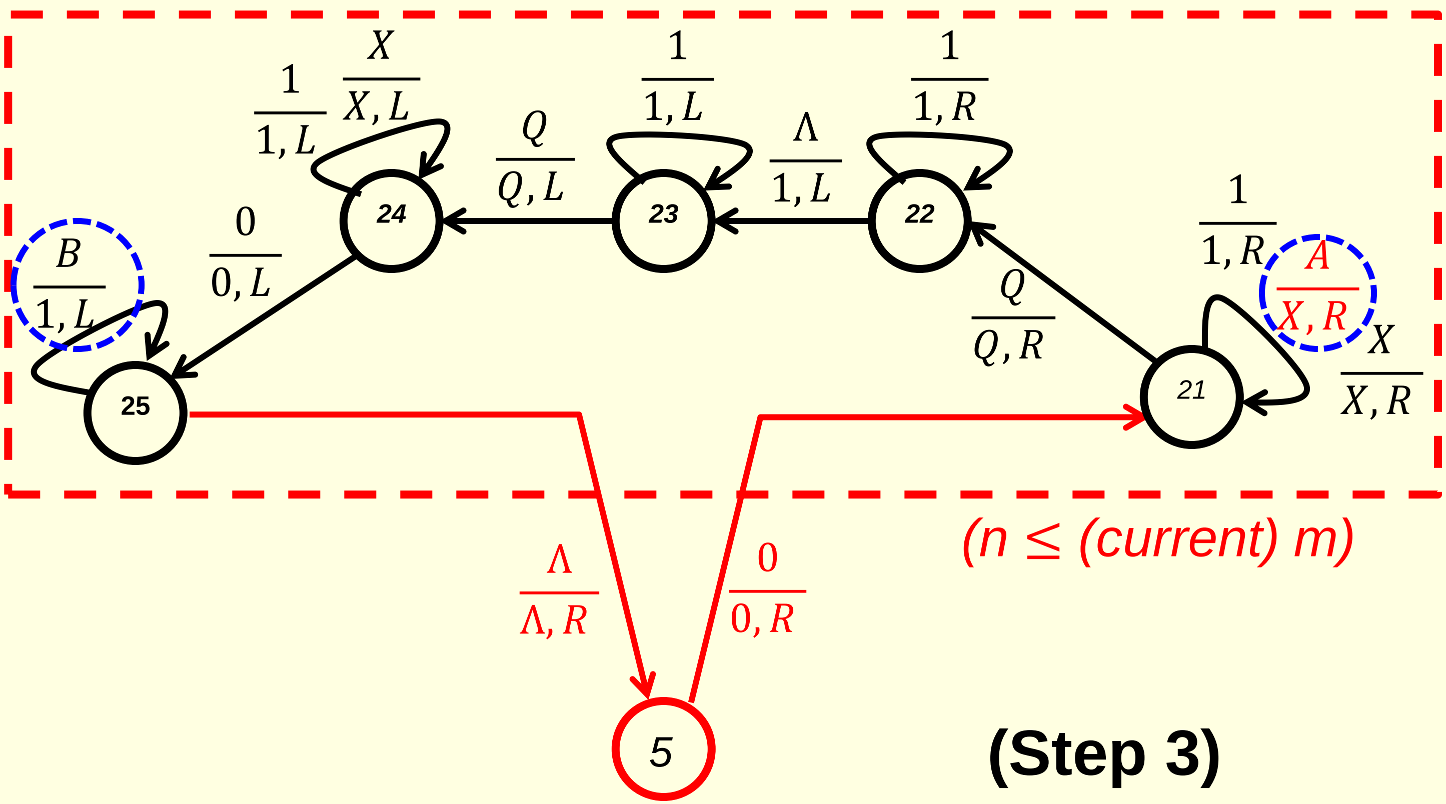
Λ	B	B	0	X	X	1	1	1	Q	1	Λ	Λ	···		

Λ	B	B	0	X	X	1	1	1	Q	1	Λ	Λ	···		

Λ	1	1	0	X	X	1	1	1	Q	1	Λ	Λ	···		

Then go to Step 2

The portion of the TM that does Step 3 :

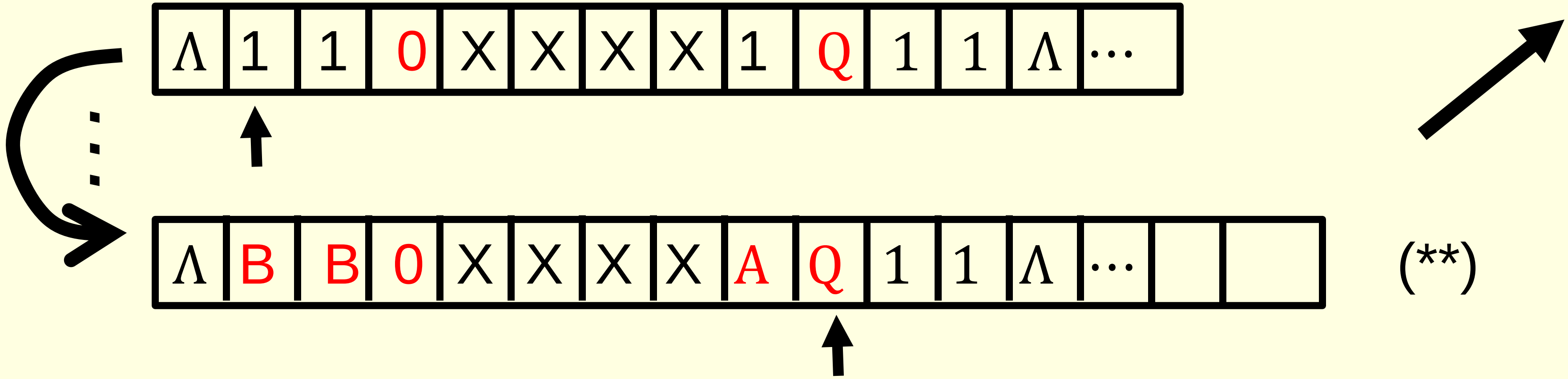


(Step 3)

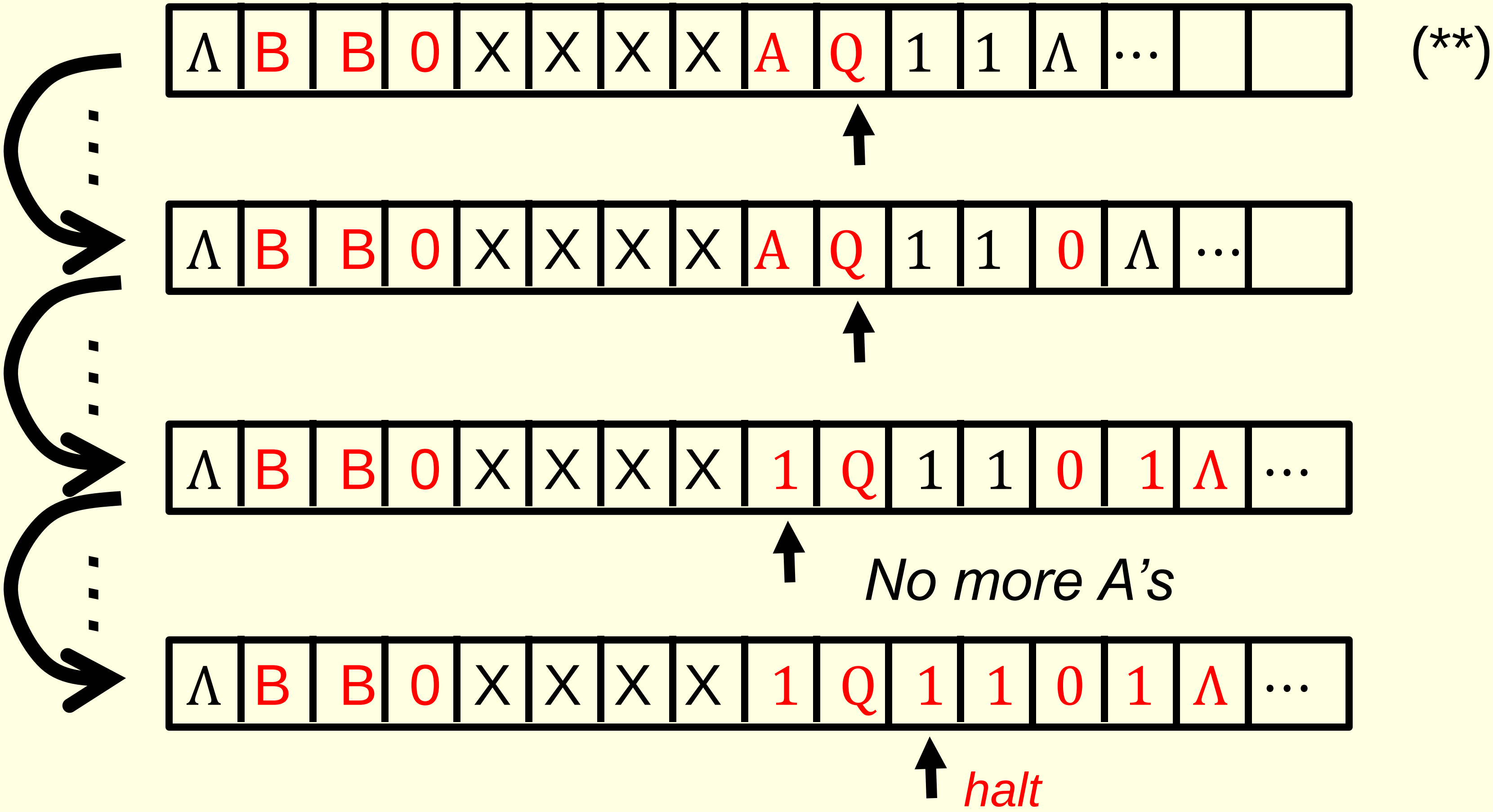
Step 4:

This step formats the output by using the number of 1’s put behind Q as the quotient and the number of available 1’s in current m as the remainder, then terminates.

From (**) in Step 2 (slide 58):



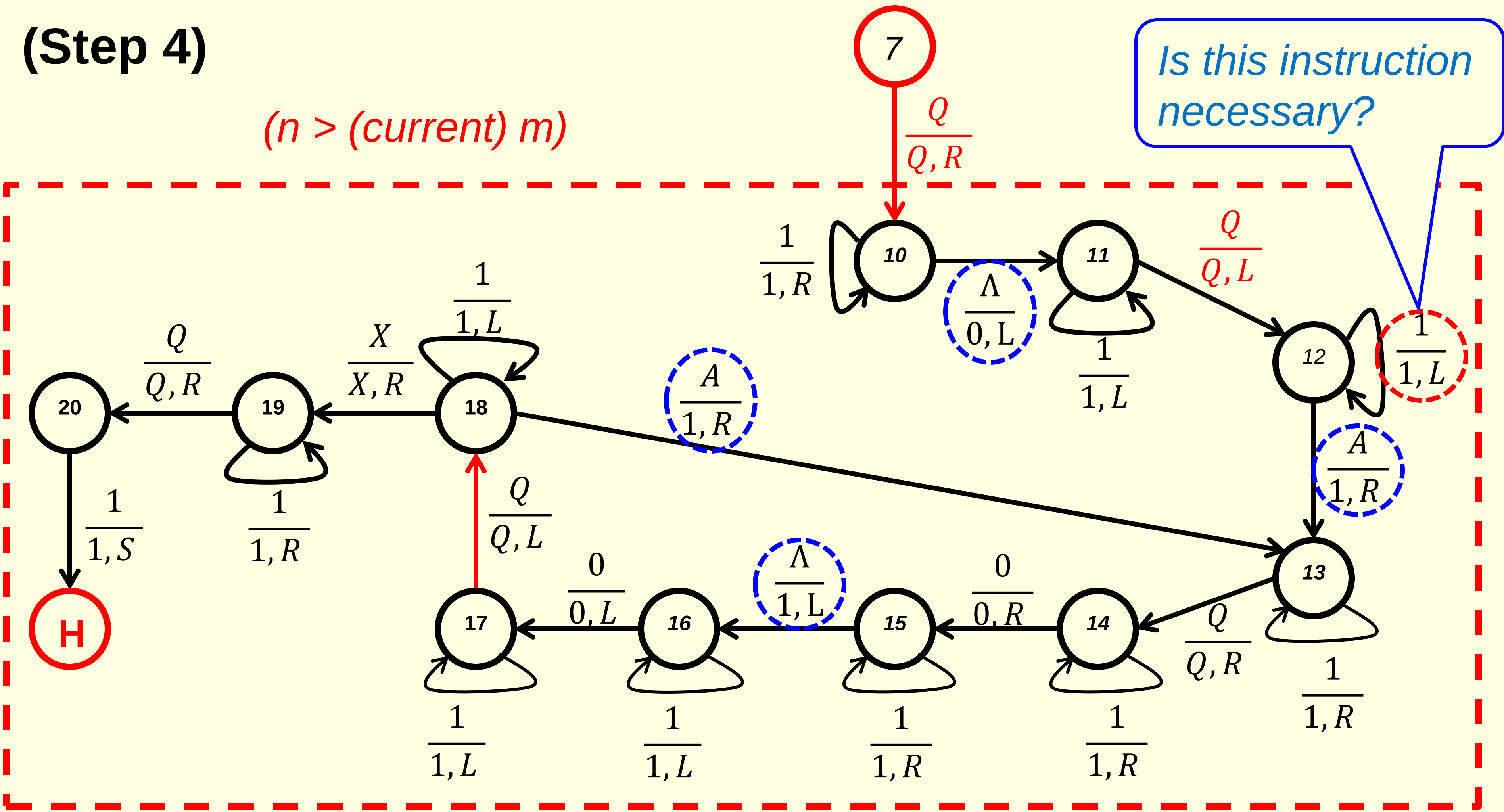
(**)


Λ B B 0 X X X X A Q 1 1 Λ ···
(**)
Λ B B 0 X X X X A Q 1 1 0 Λ ···
Λ B B 0 X X X X 1 Q 1 1 0 1 Λ ···
No more A's
Λ B B 0 X X X X 1 Q 1 1 0 1 Λ ···
halt

The portion of the TM that performs Step 4 :

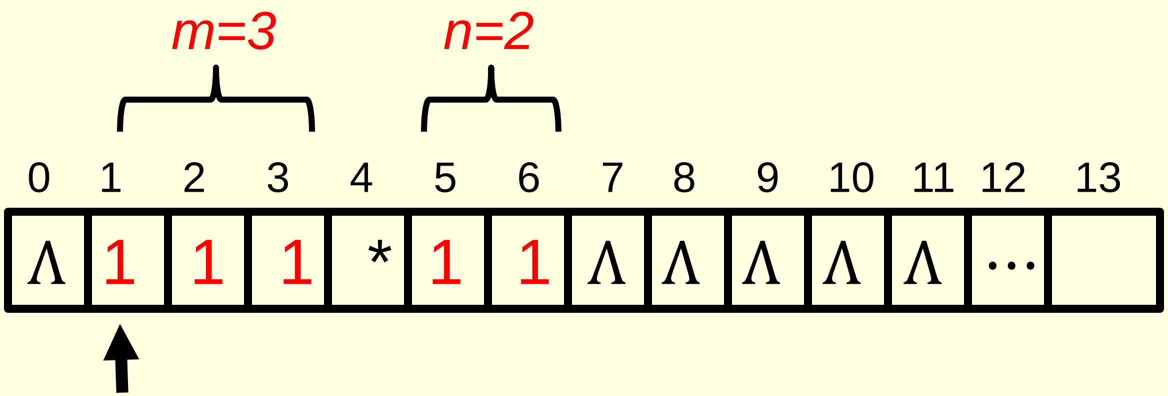
(Step 4)

(n > (current) m)



End of Turing Machines II

Input for question 1 of HW10 – 2025s



Question 2 of HW10-2025s

Input: