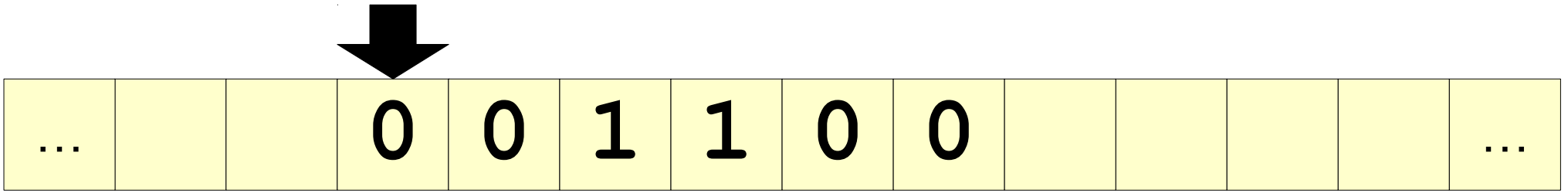
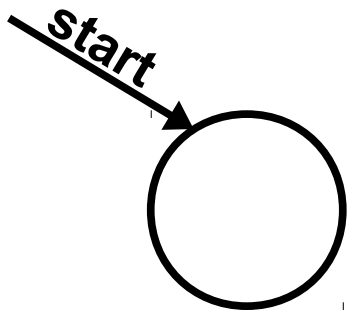
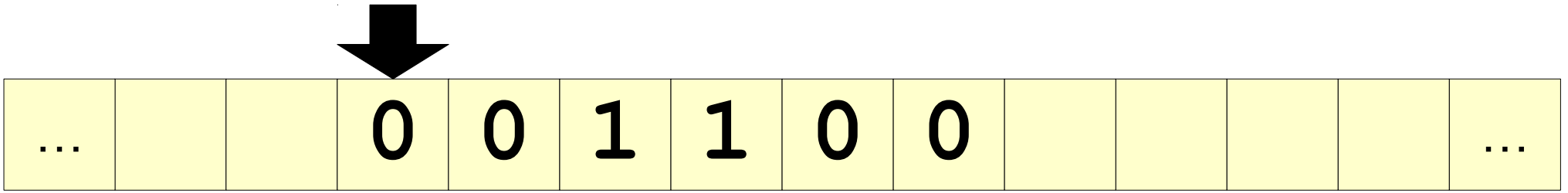
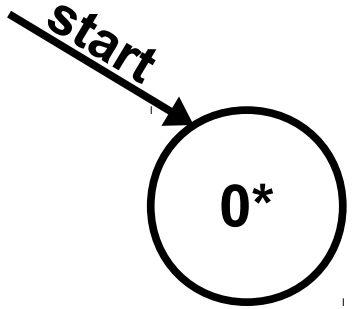
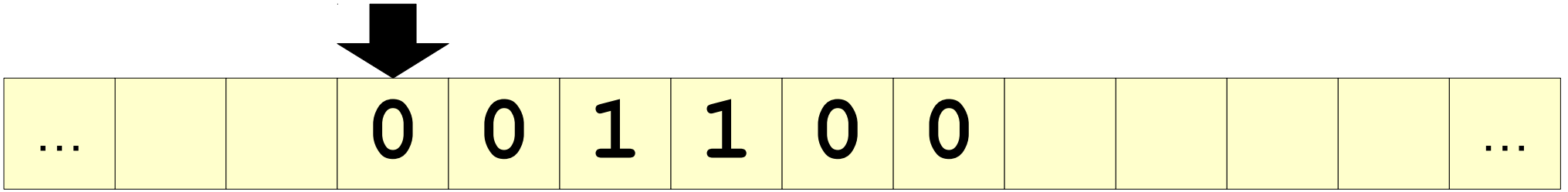
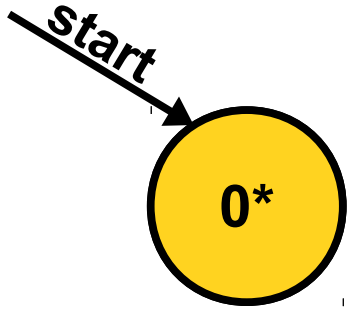

...
0 0 1 1 0 0
...

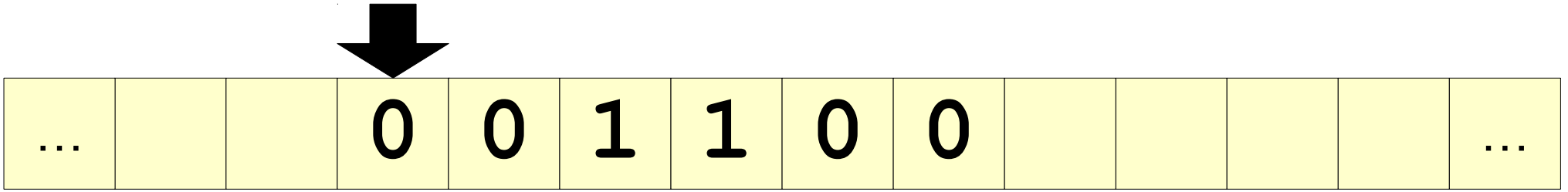

start

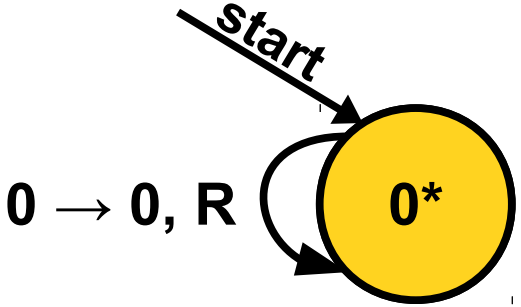

...
0 0 1 1 0 0
...

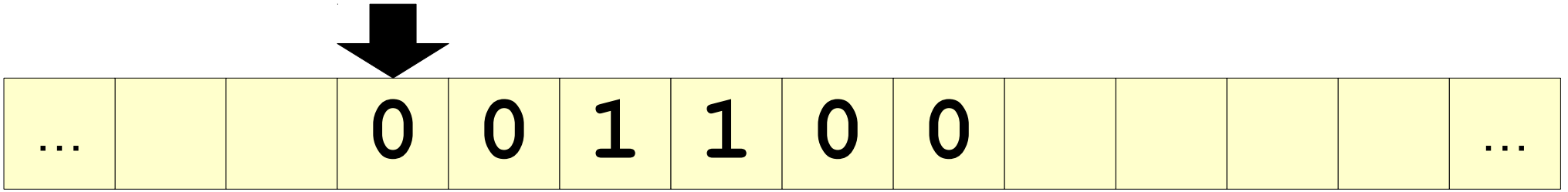

start
0*

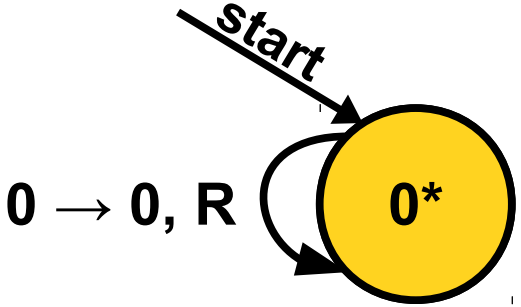

…
0 0 1 1 0 0
…

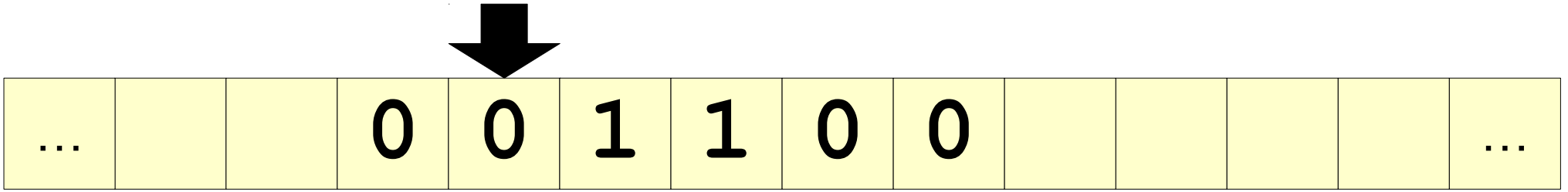

start
0*

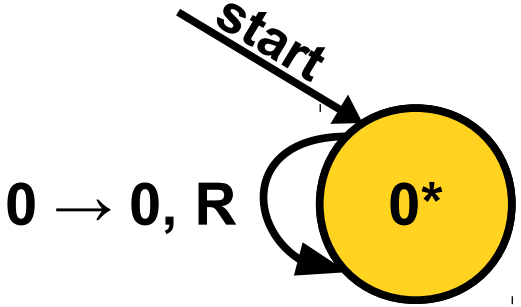

...
0 0 1 1 0 0
...

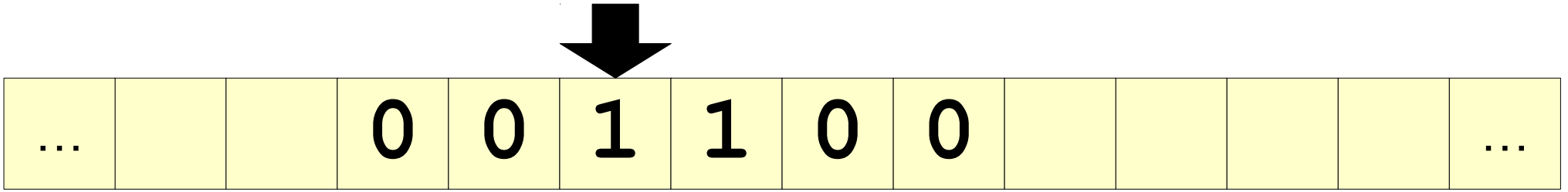

start
0 → 0, R
0*

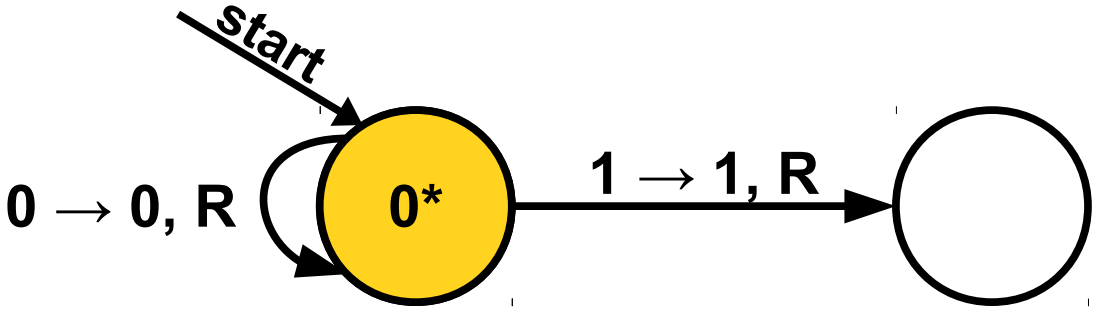

…
0 0 1 1 0 0
…

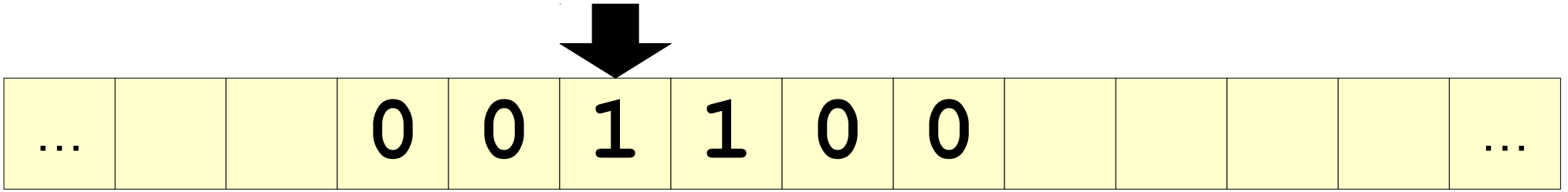

start
0 → 0, R
0*

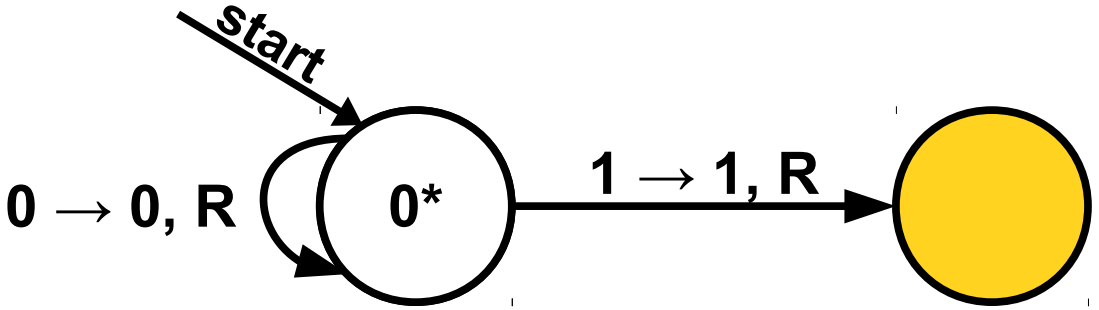

…
0 0 1 1 0 0
…

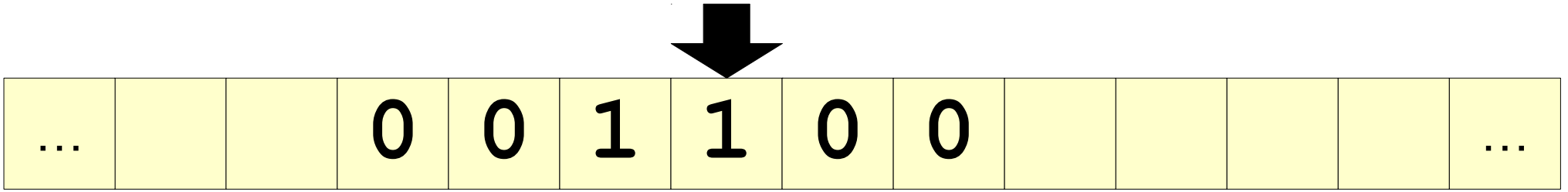

start
0 → 0, R
0*

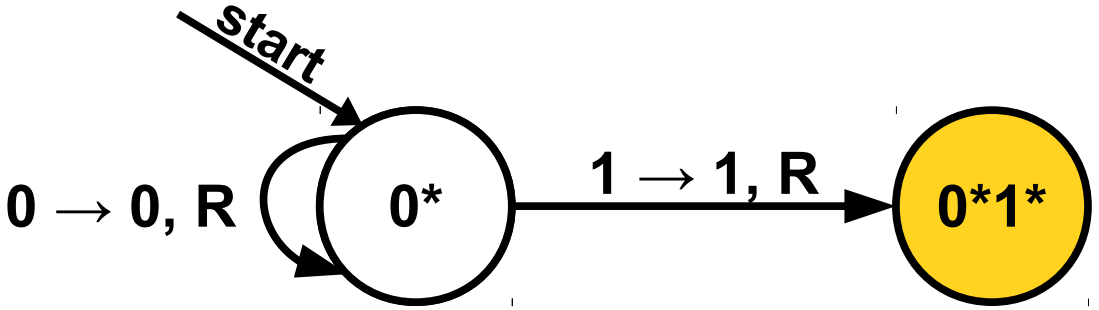

…
0 0 1 1 0 0
…

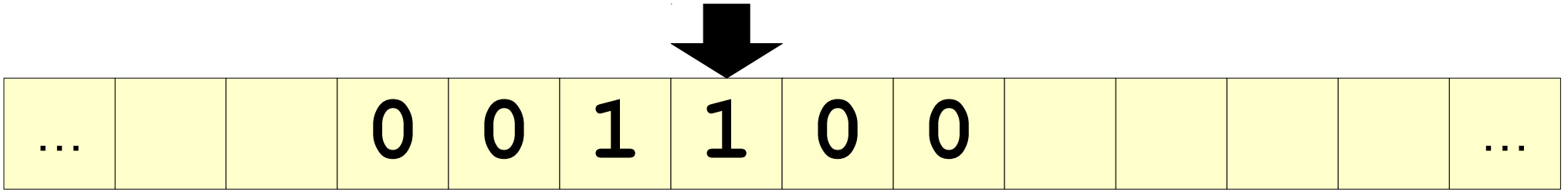

start
0 → 0, R
0*
1 → 1, R

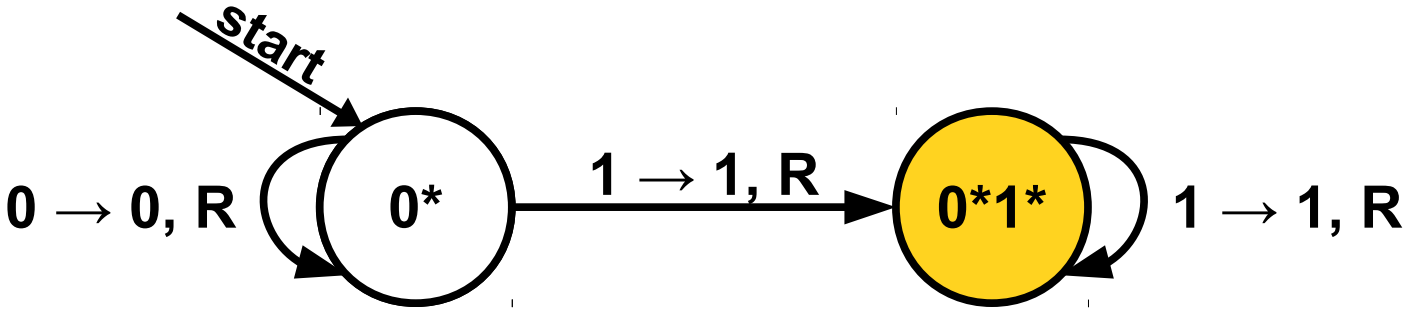

…
0 0 1 1 0 0
…

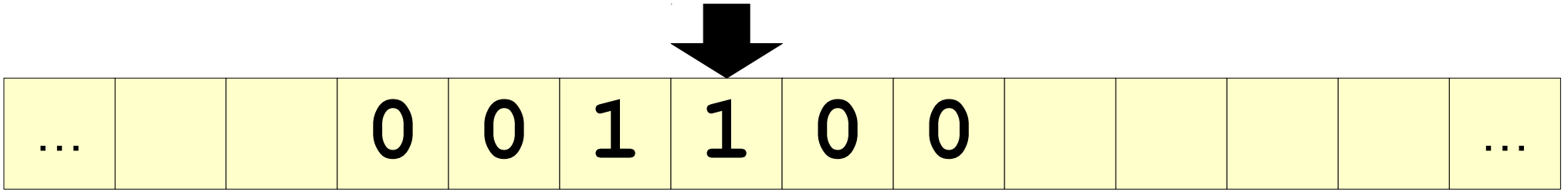

start
0 → 0, R
0*
1 → 1, R

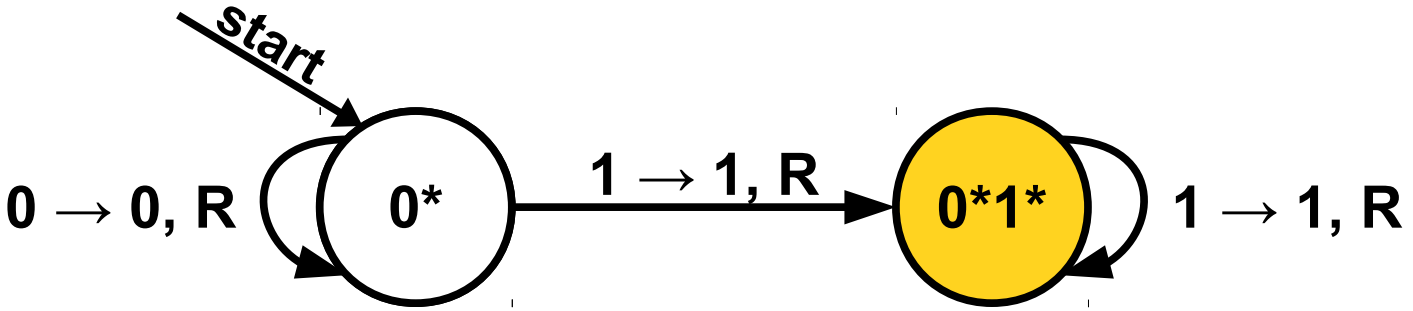

…
0 0 1 1 0 0
…

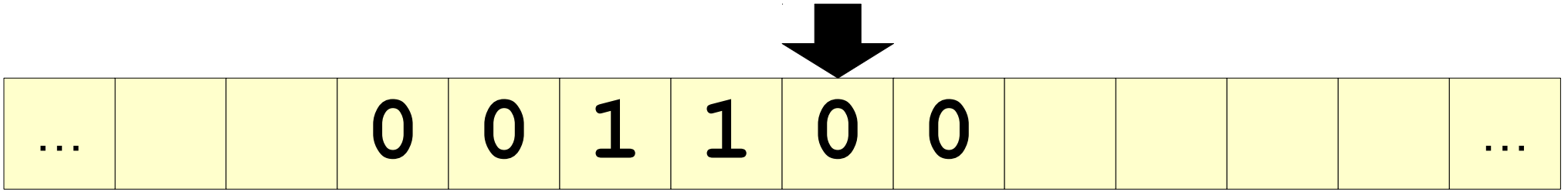

start
0 → 0, R
0*
1 → 1, R
0*1*

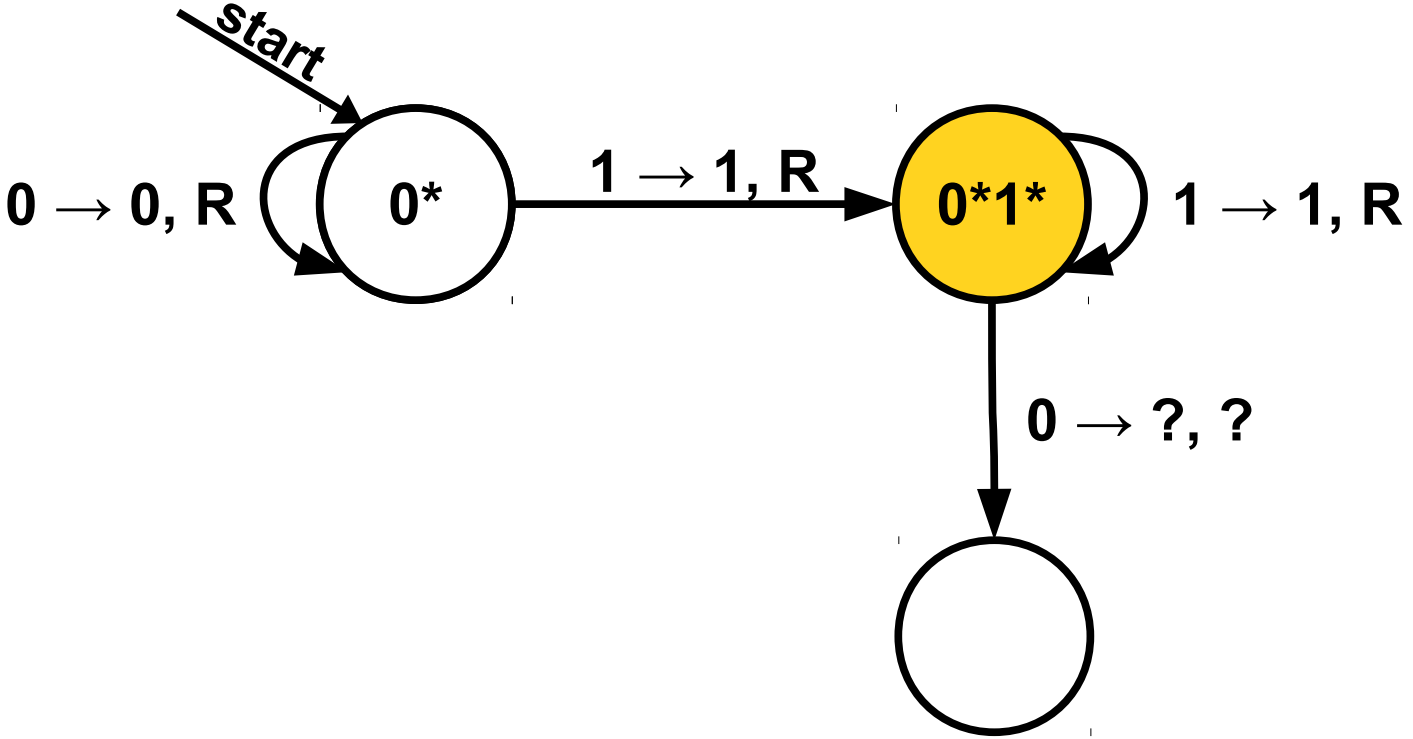

…
0 0 1 1 0 0
…

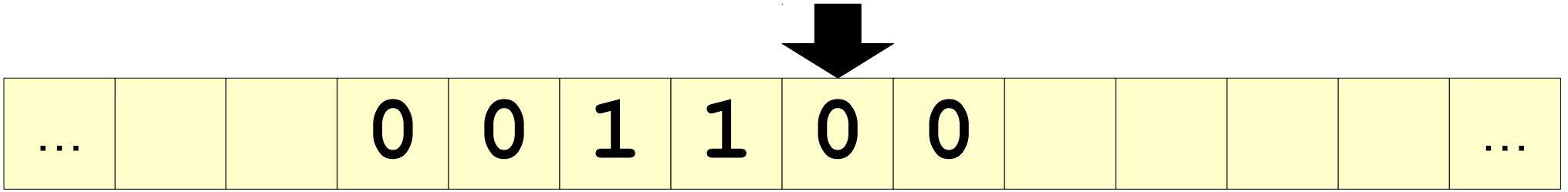

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R


…
0 0 1 1 0 0
…


start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R

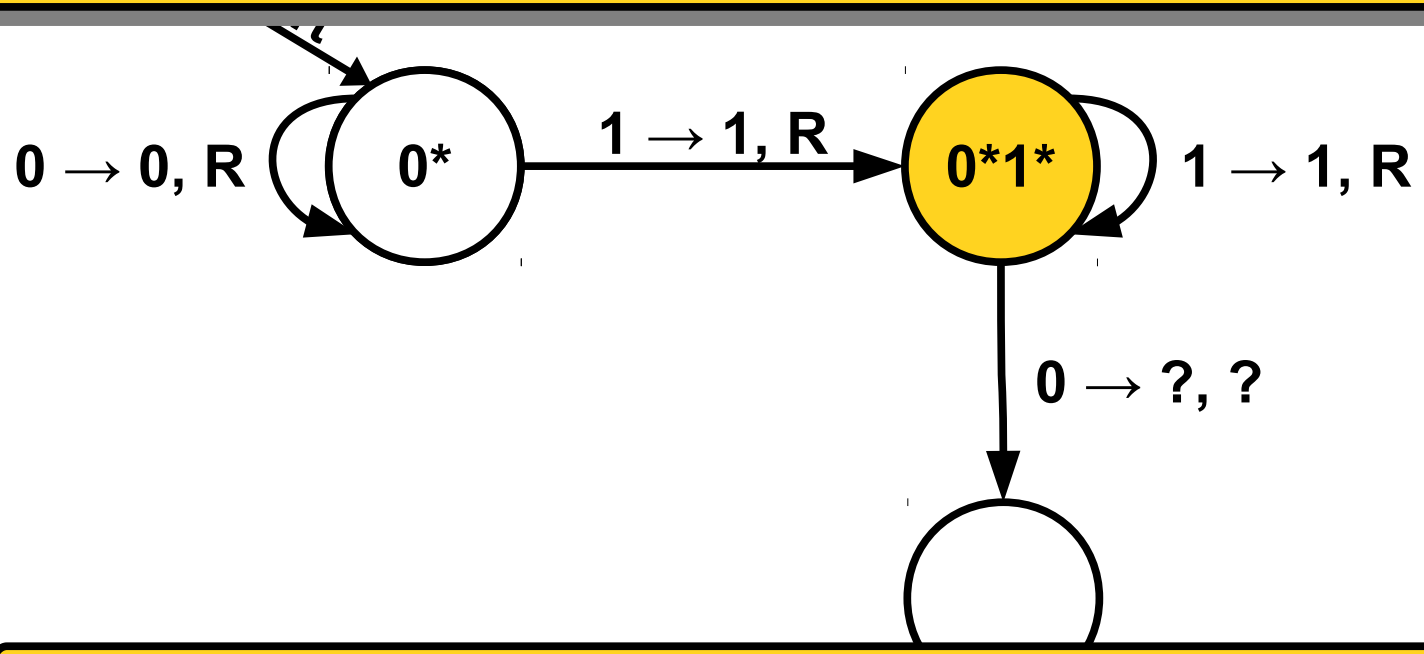

…
0 0 1 1 0 0
…


start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → ?, ?

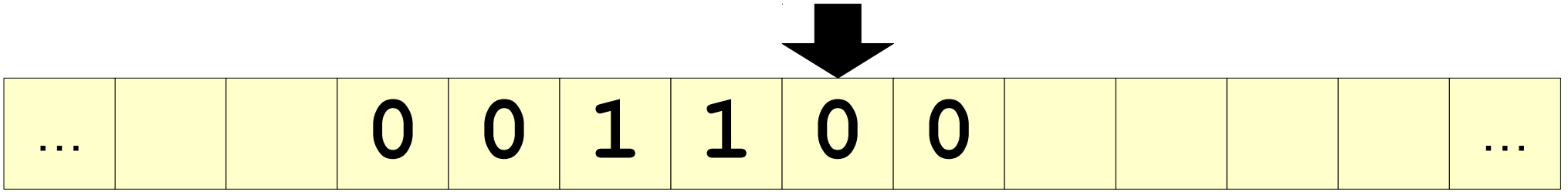

…
0 0 1 1 0 0
…

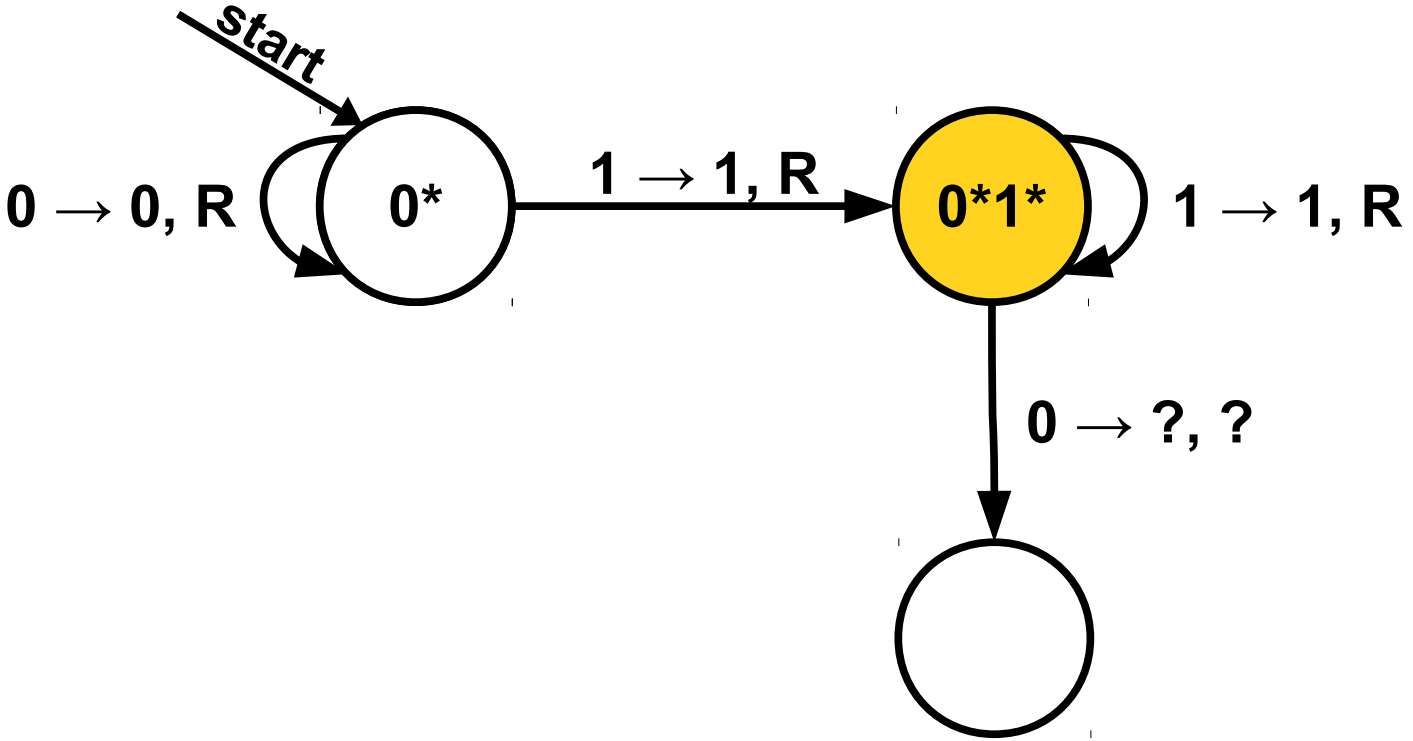
Based on we want this TM to do, what should this transition say?

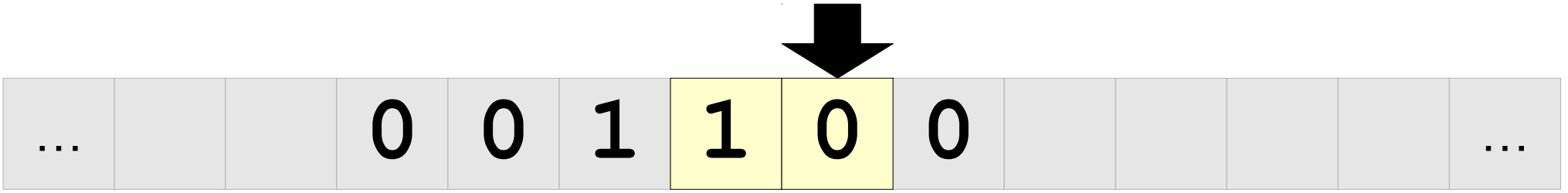
A. **0 → 0**, R
B. **0 → 1**, R
C. **0 → 0**, L
D. **0 → 1**, L
E. None of these, or two or more of these.

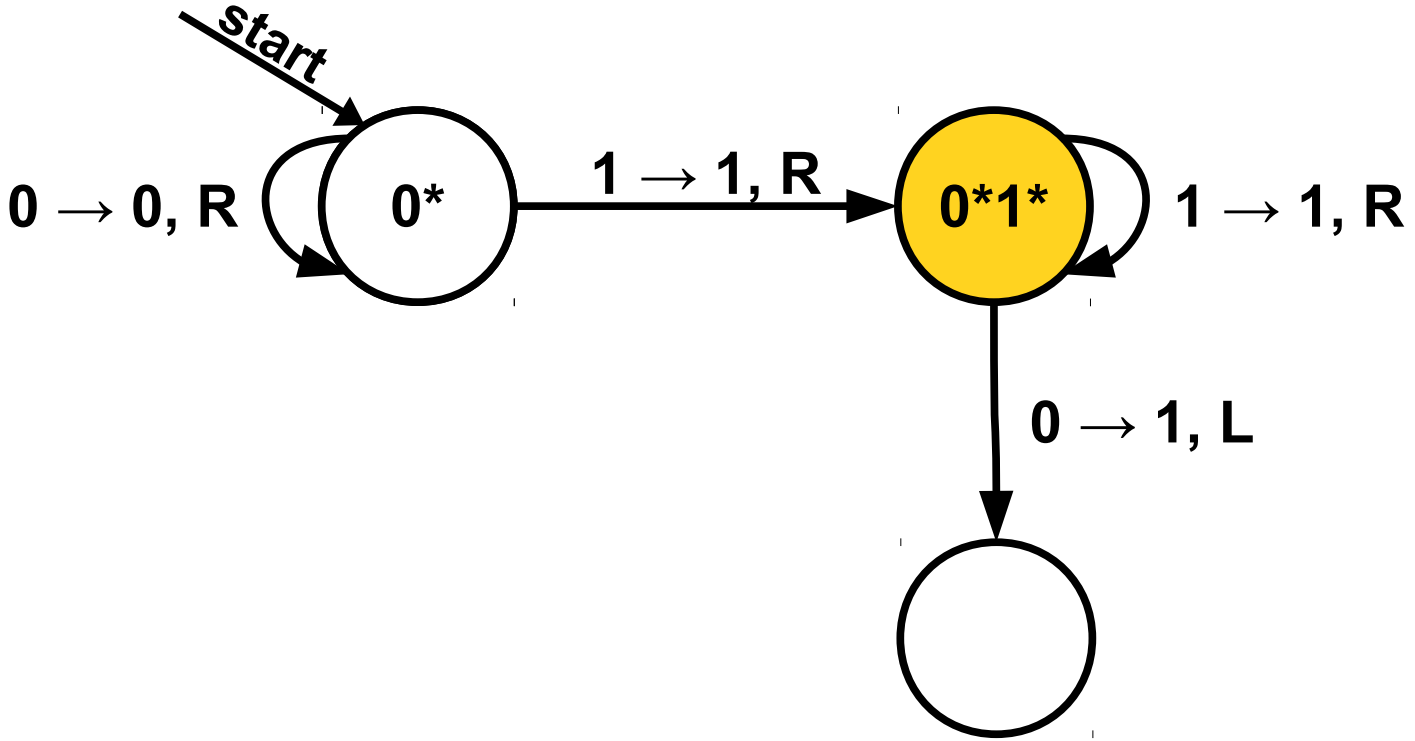


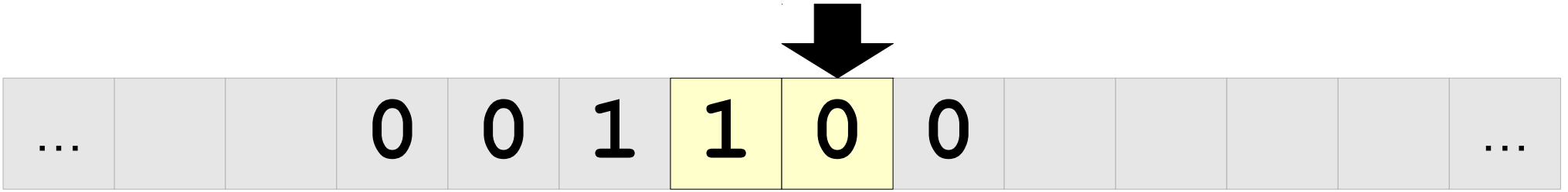
Answer at **PollEv.com/cs103** or
text **CS103** to **22333** once to join, then ***A***, ***B***, ***C***, ***D***, or ***E***.

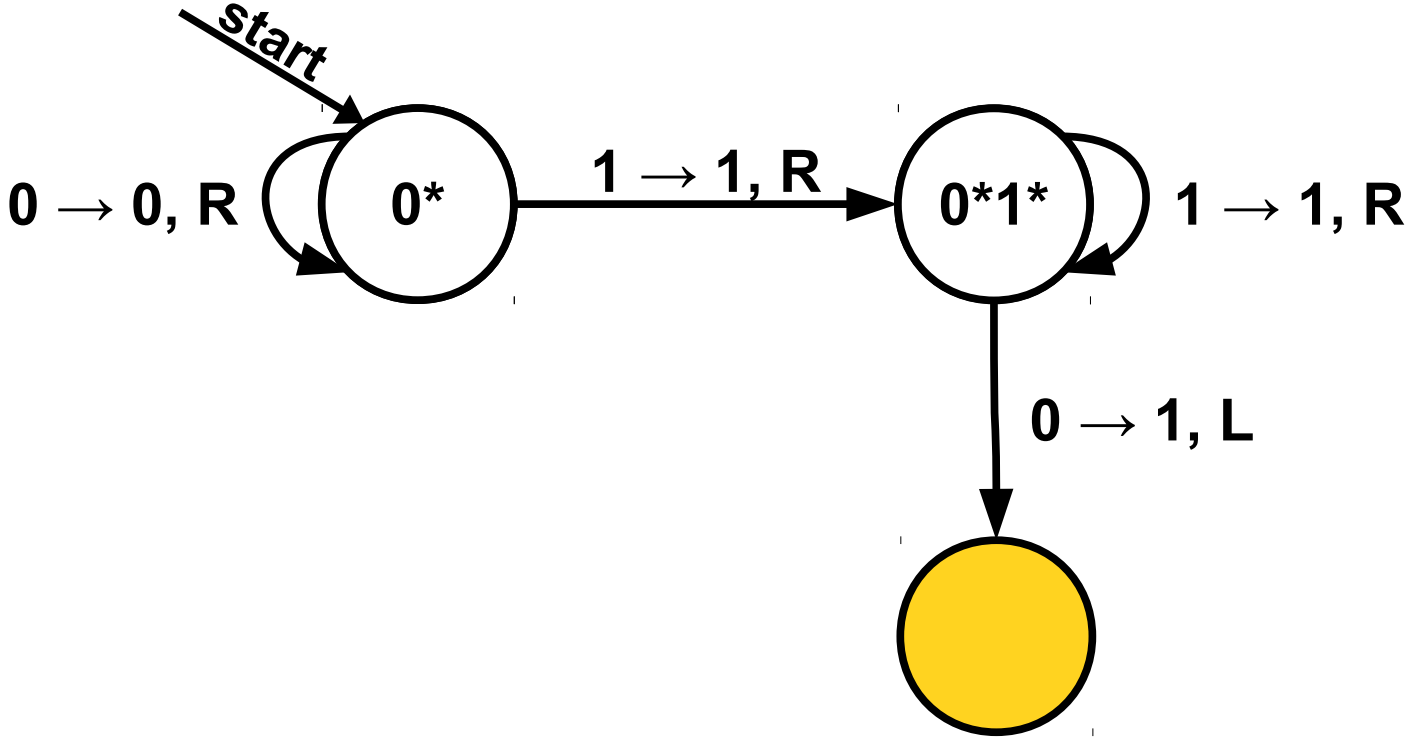


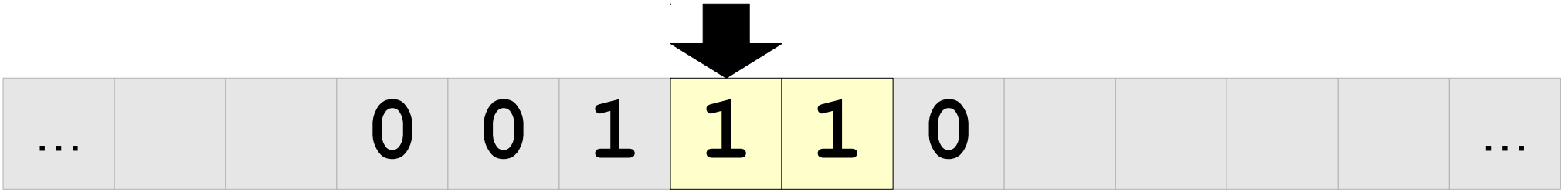

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → ?, ?

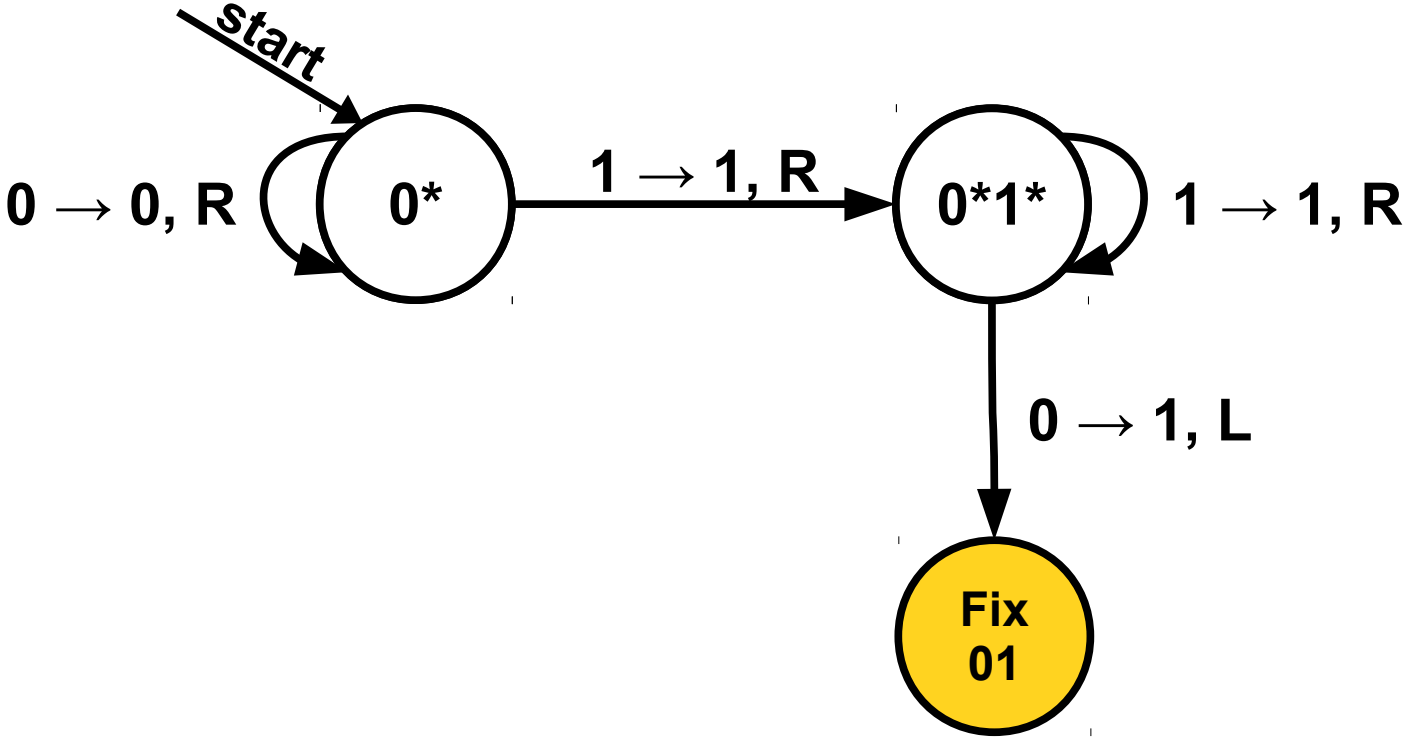

…
0 0 1 1 0 0
…

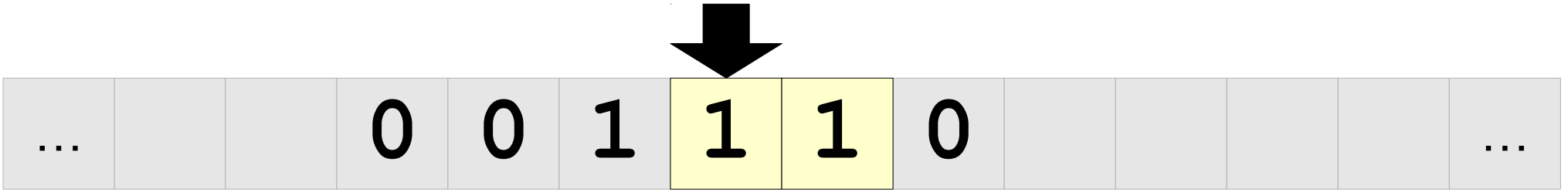

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L

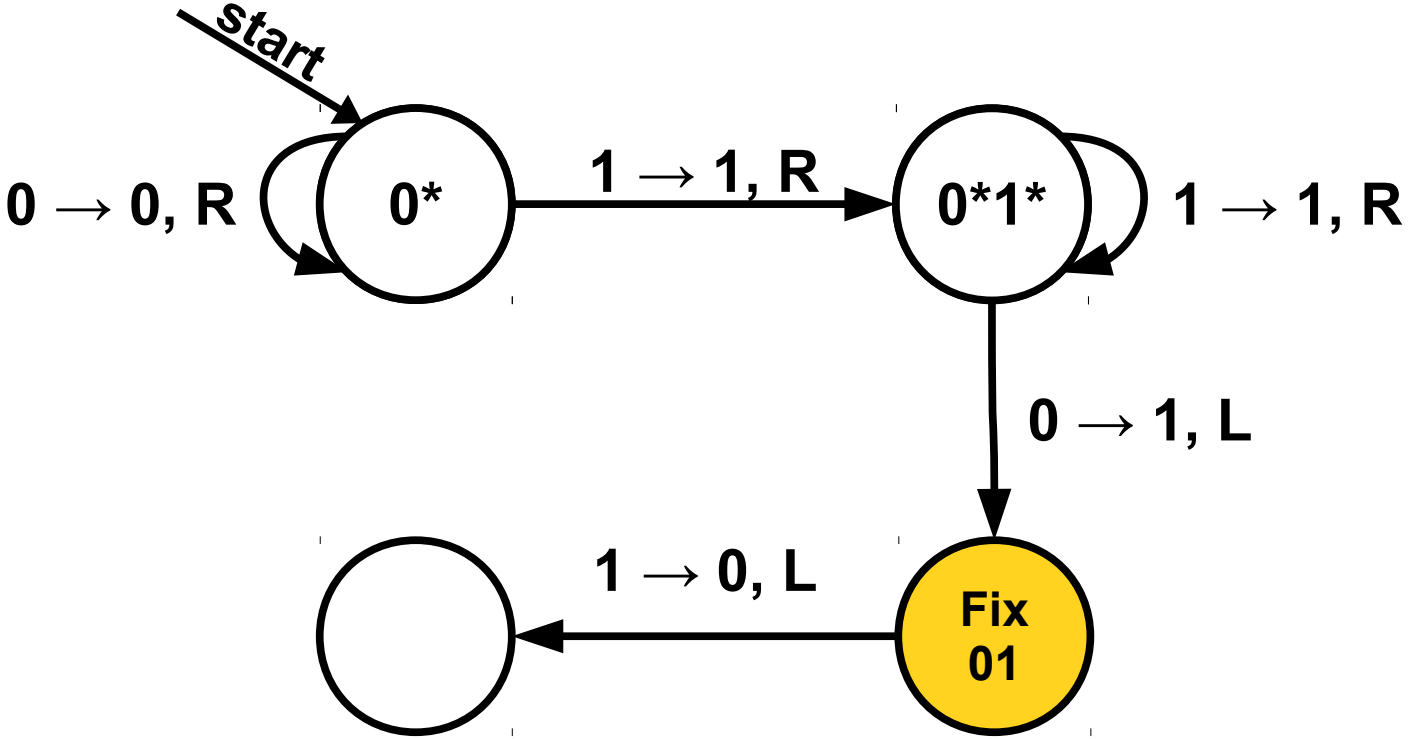

…
0 0 1 1 0 0
…

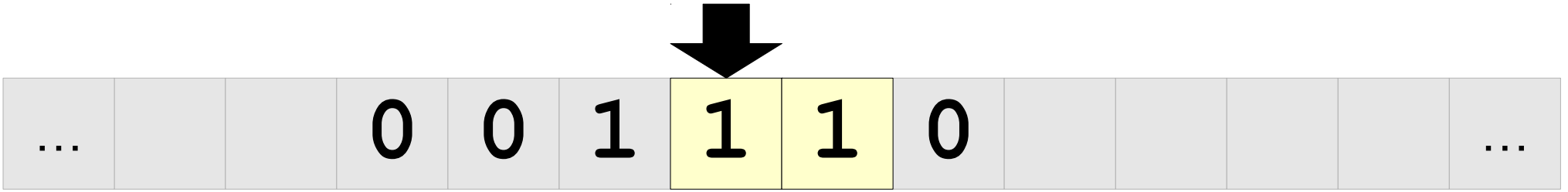

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L

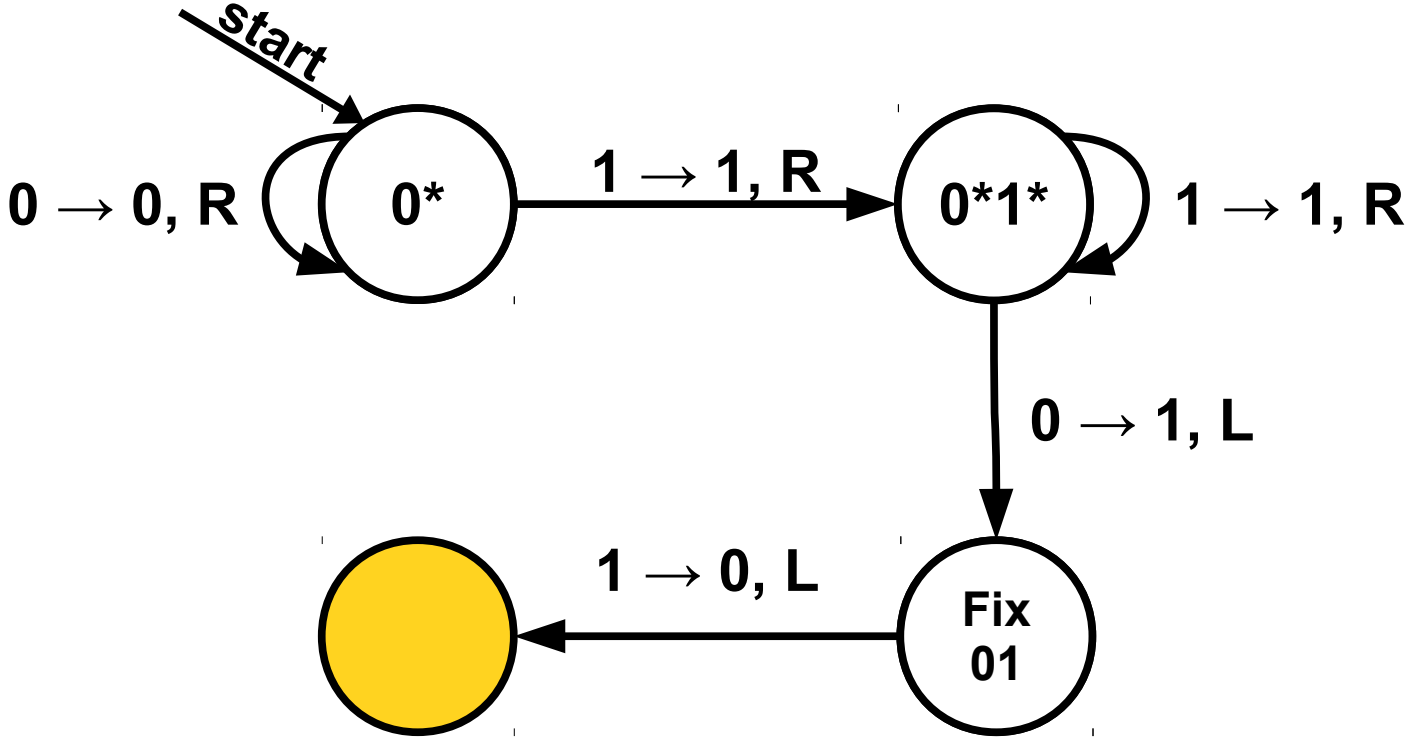

…
0 0 1 1 1 0
…

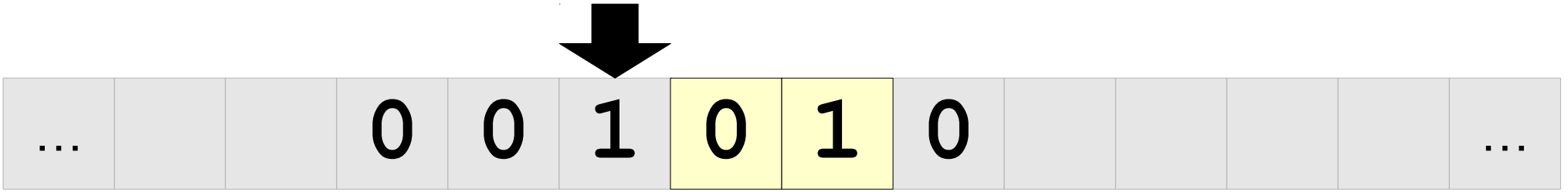

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
Fix
01

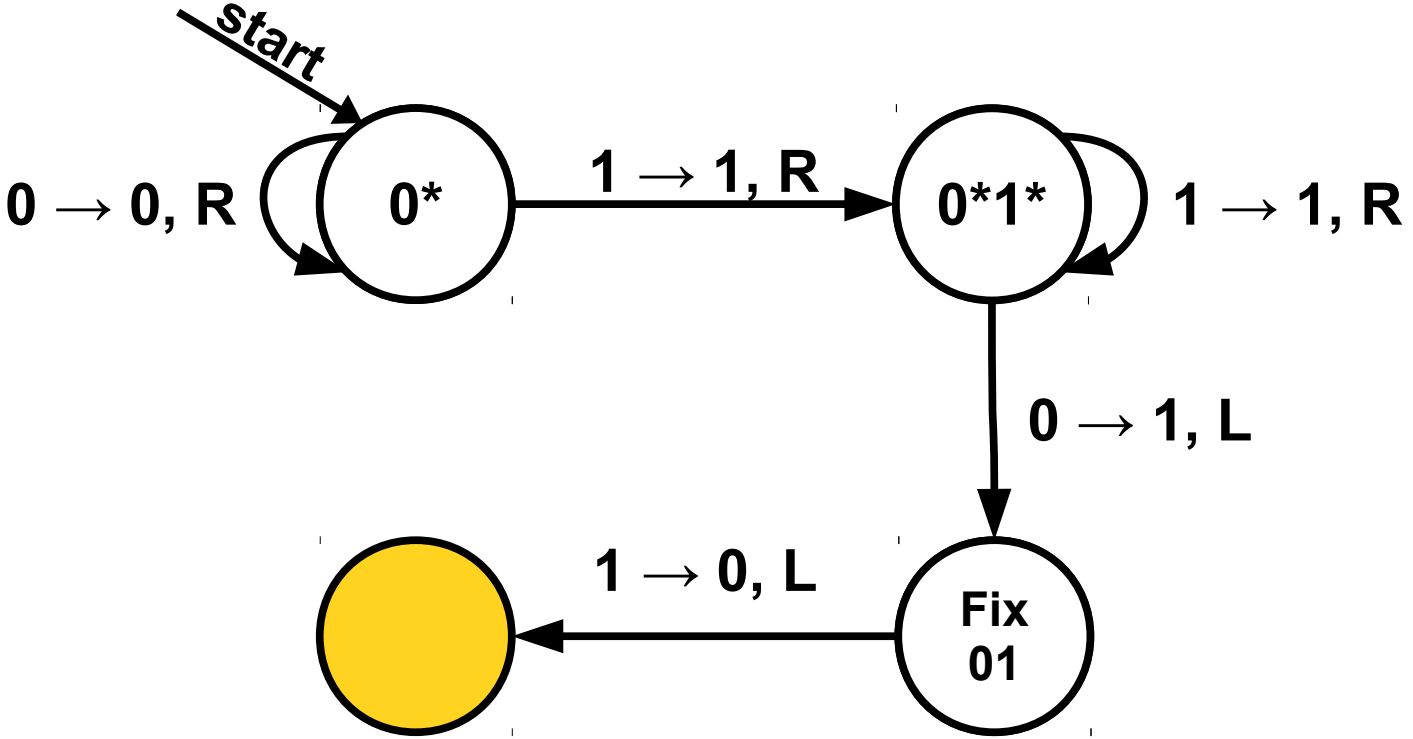

…
0 0 1 1 1 0
…

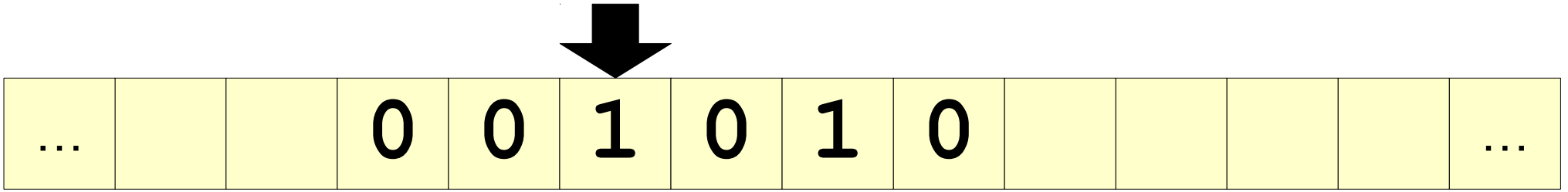

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
1 → 0, L
Fix
01

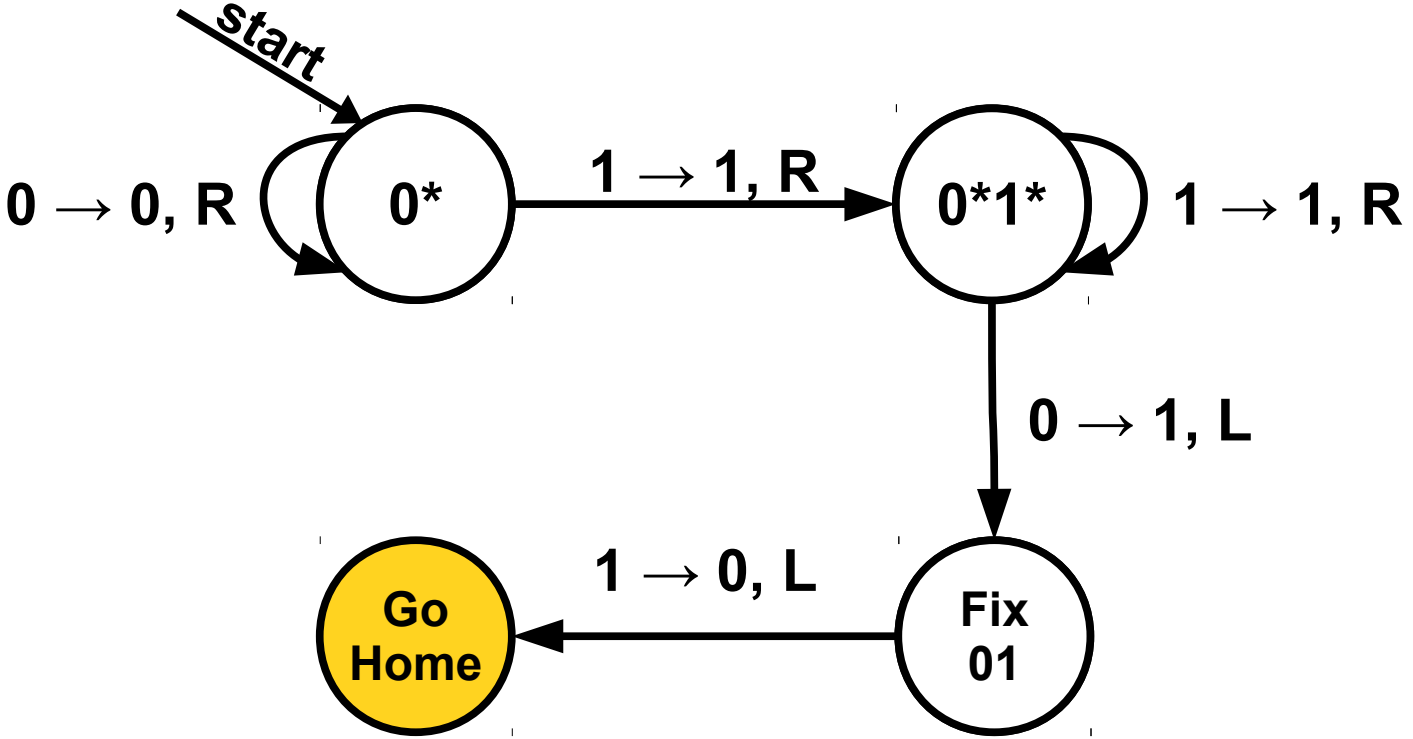

…
0 0 1 1 1 0
…

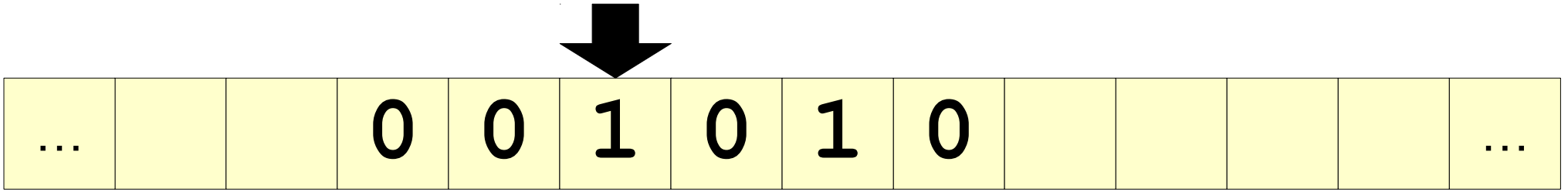

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
1 → 0, L
Fix
01

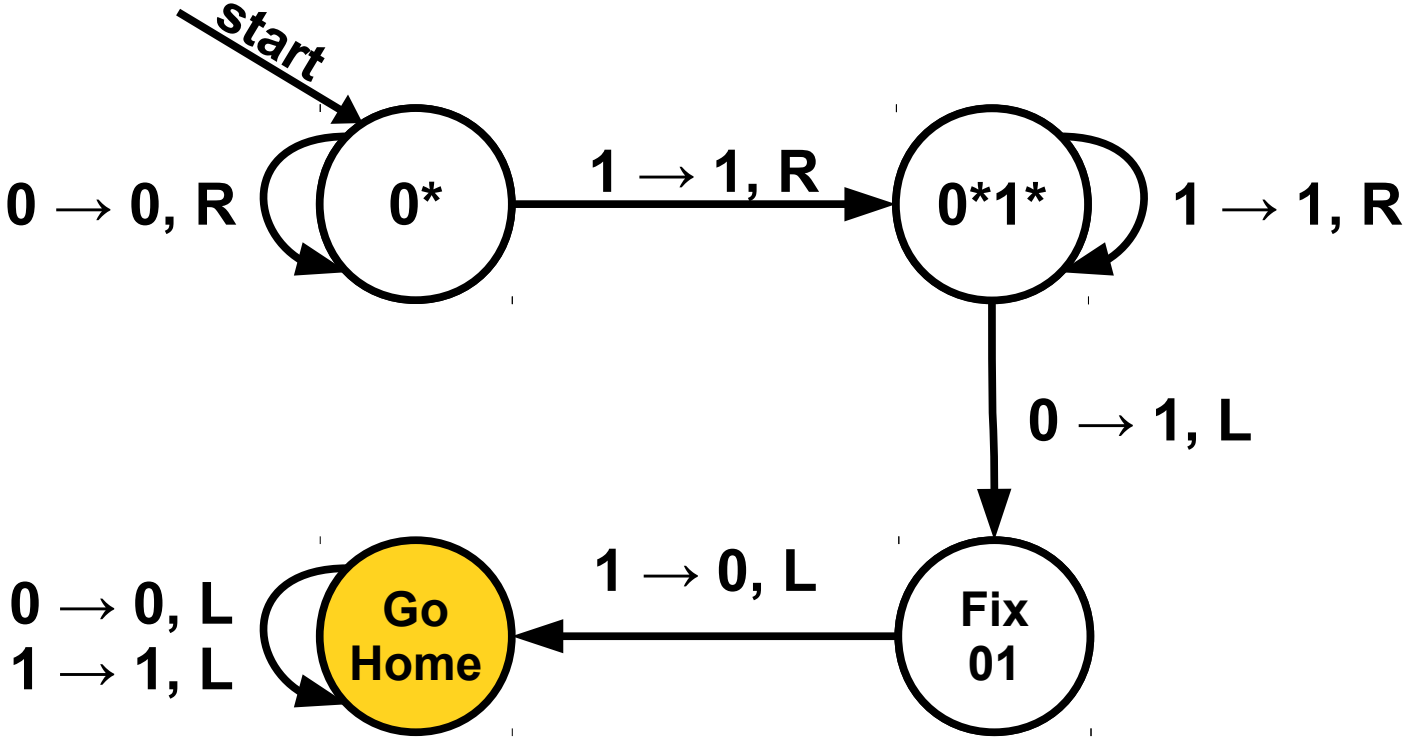

…
0 0 1 0 1 0
…

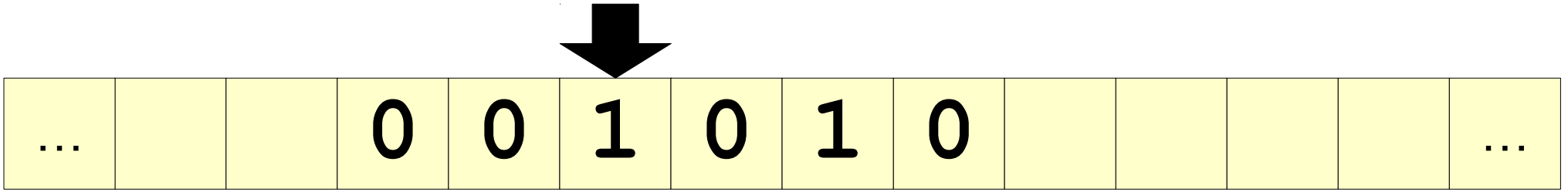

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
1 → 0, L
Fix
01

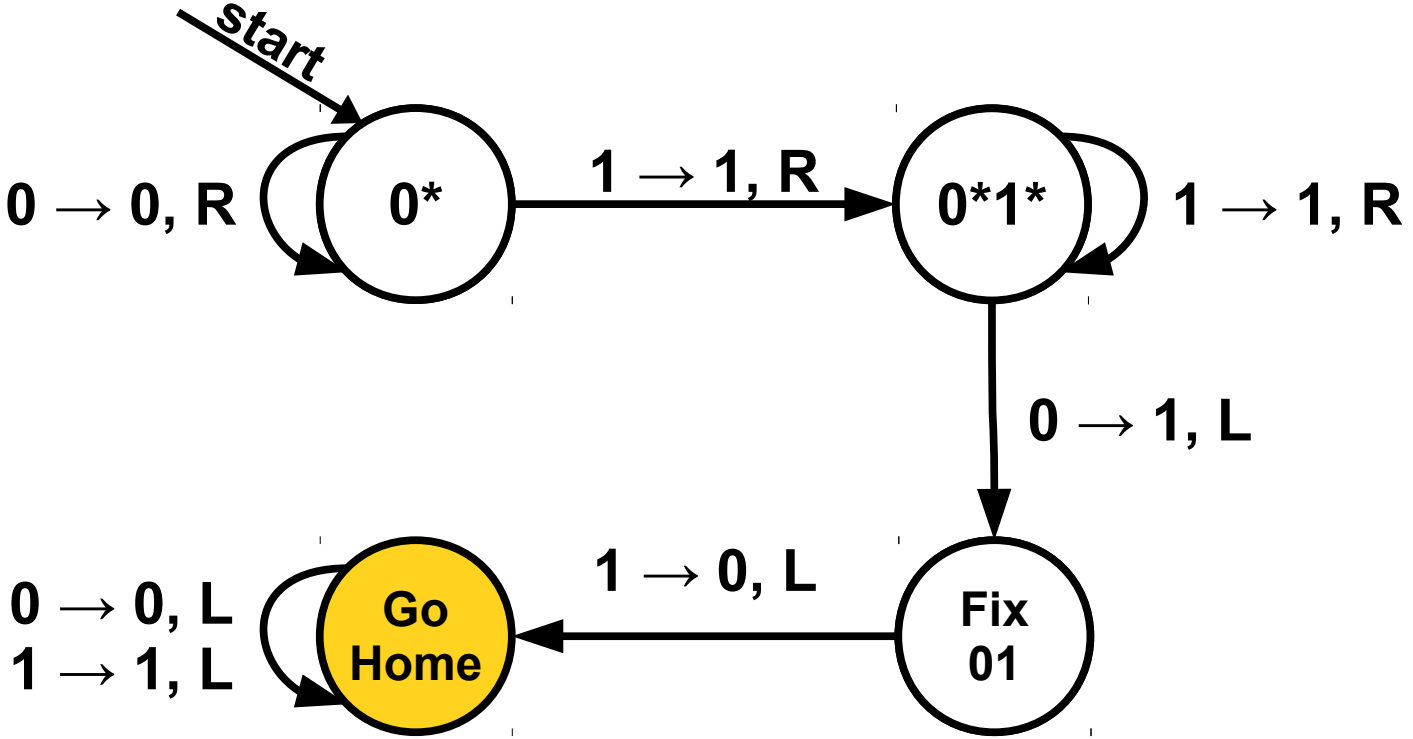

…
0 0 1 0 1 0
…

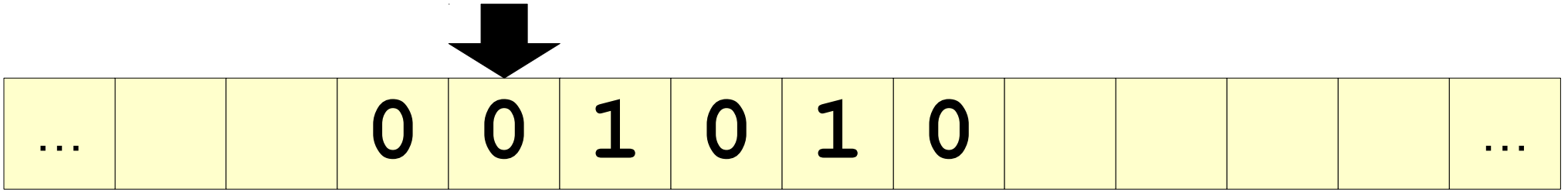

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
Go Home
1 → 0, L
Fix 01

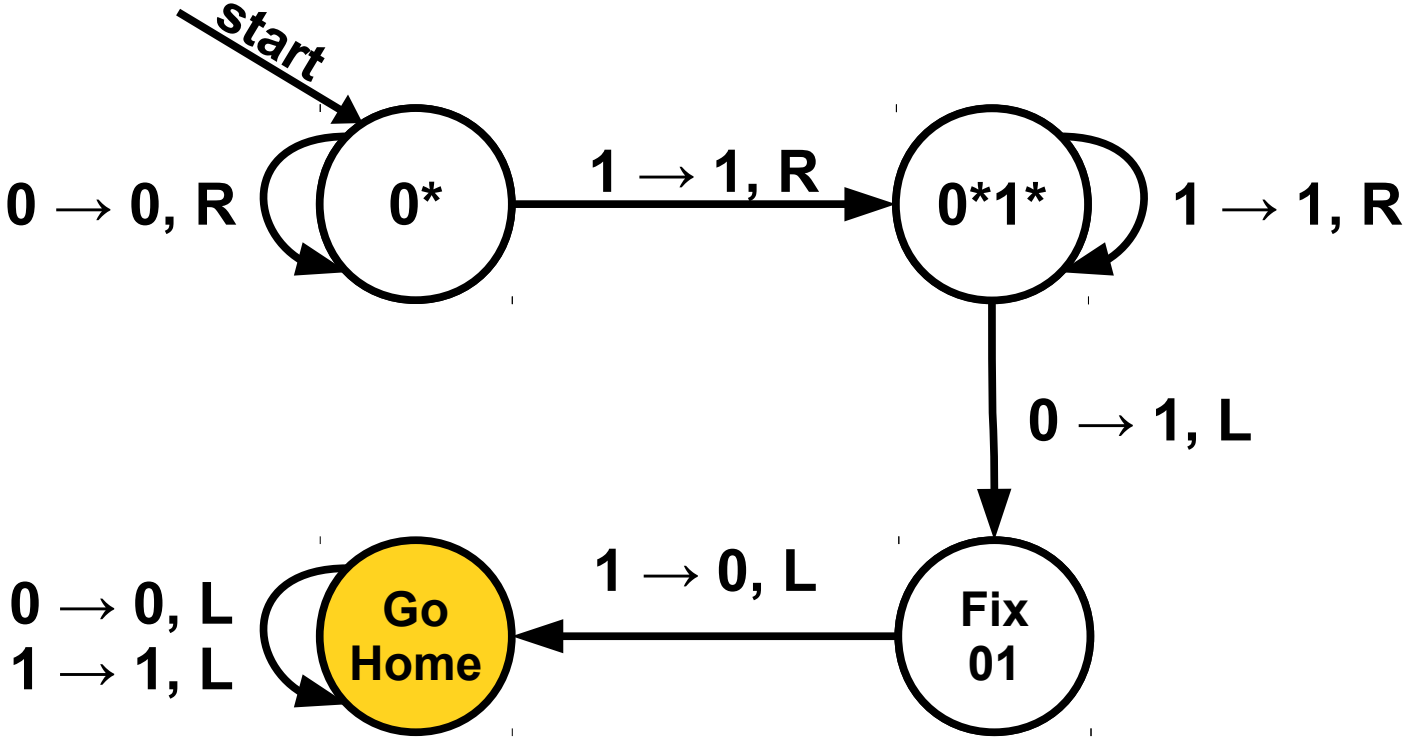

…
0 0 1 0 1 0
…

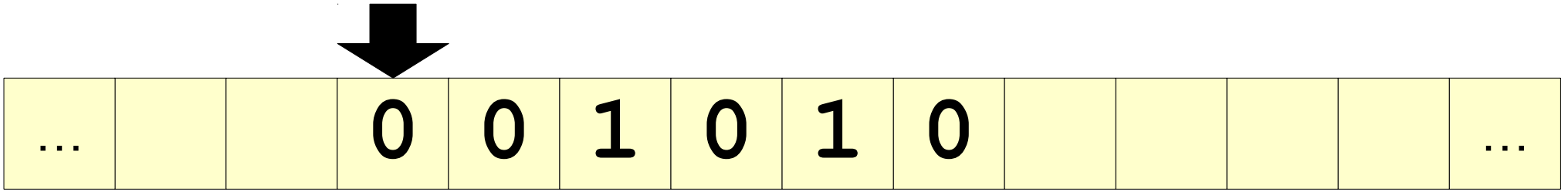

start
0*
0 → 0, R
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
Fix
01
1 → 0, L
Go
Home
0 → 0, L
1 → 1, L

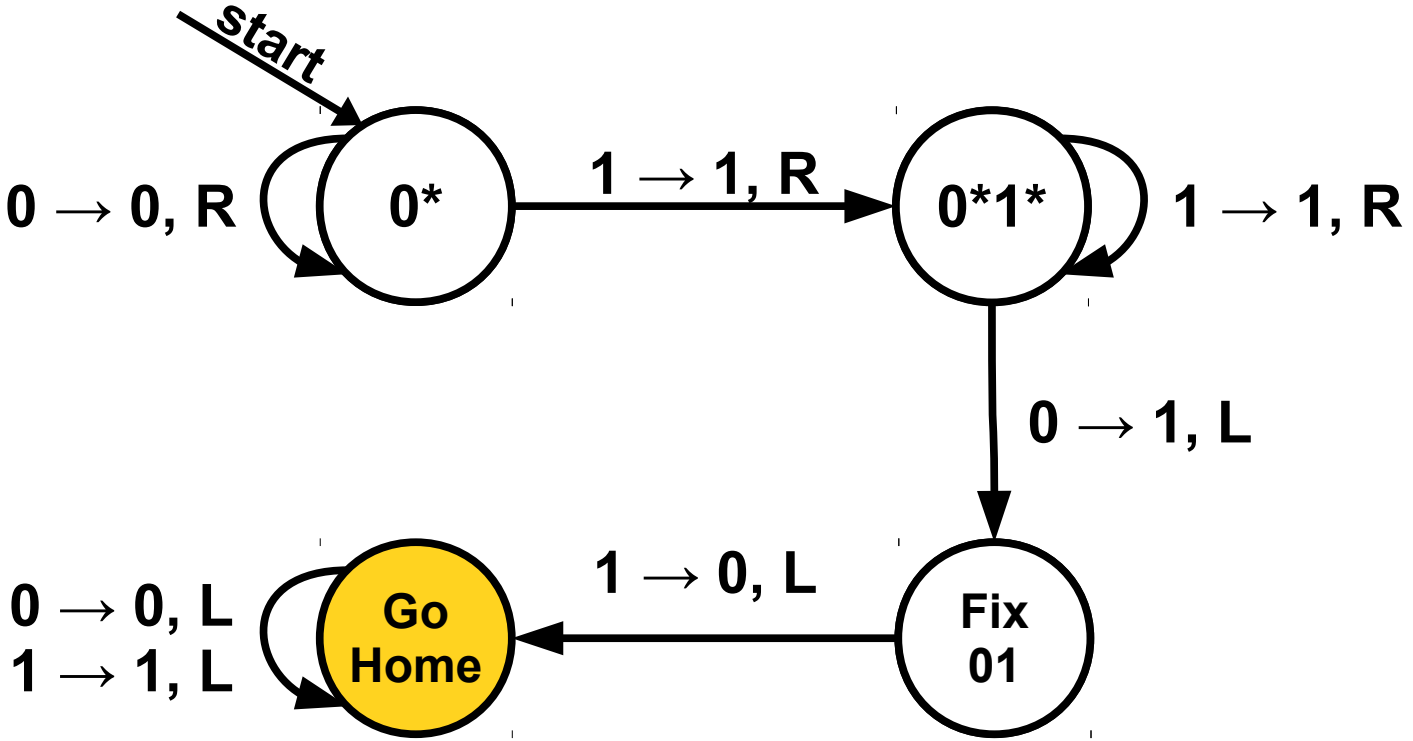

…
0 0 1 0 1 0
…

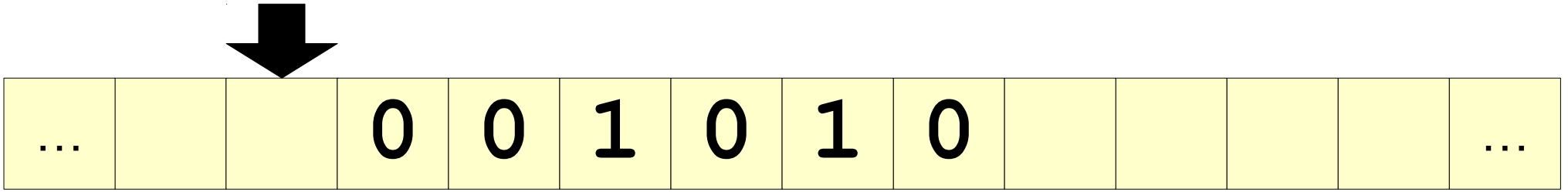

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

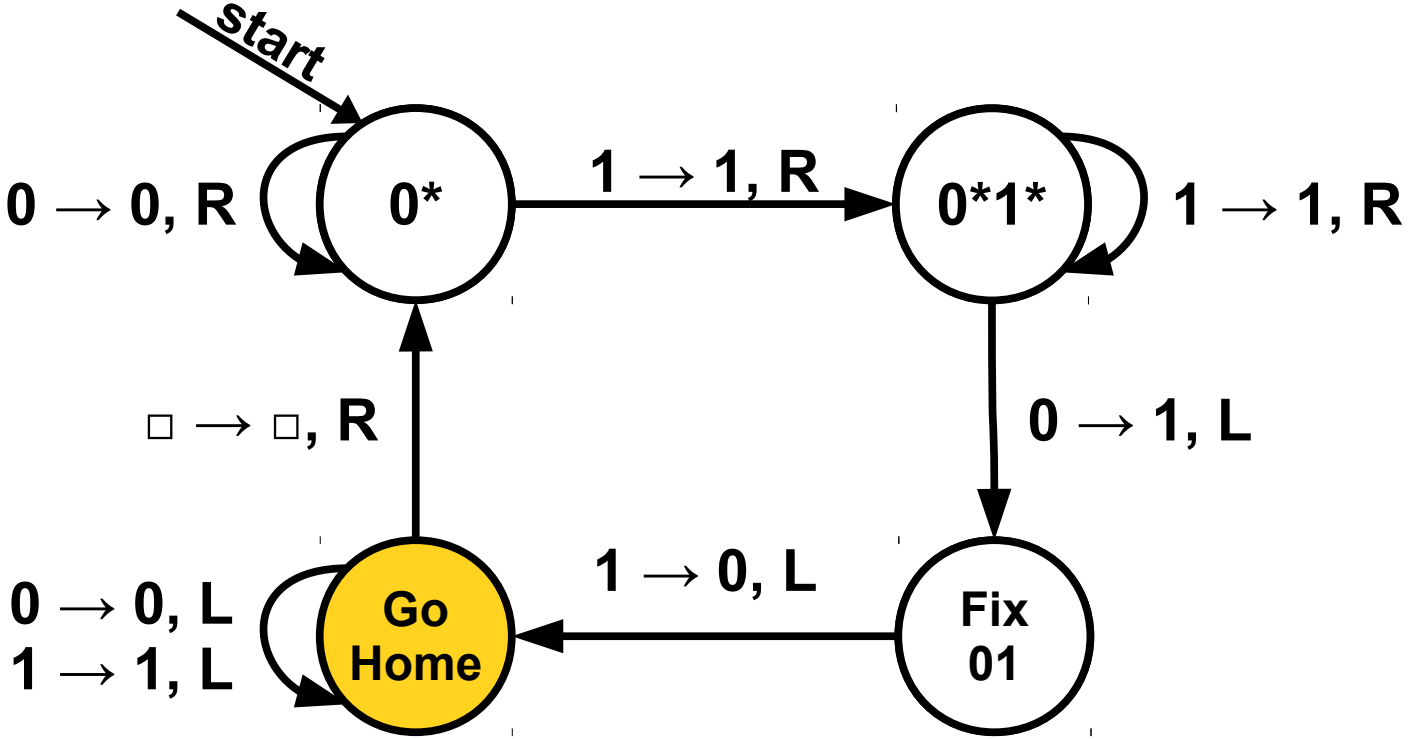

…
0 0 1 0 1 0
…

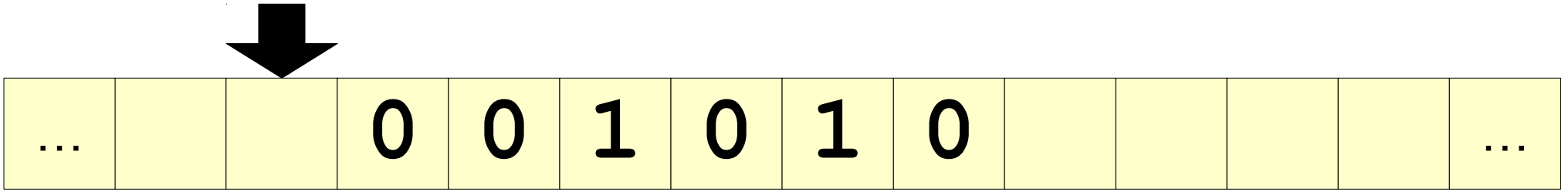

start
0*
0 → 0, R
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
Fix
01
1 → 0, L
Go
Home
0 → 0, L
1 → 1, L

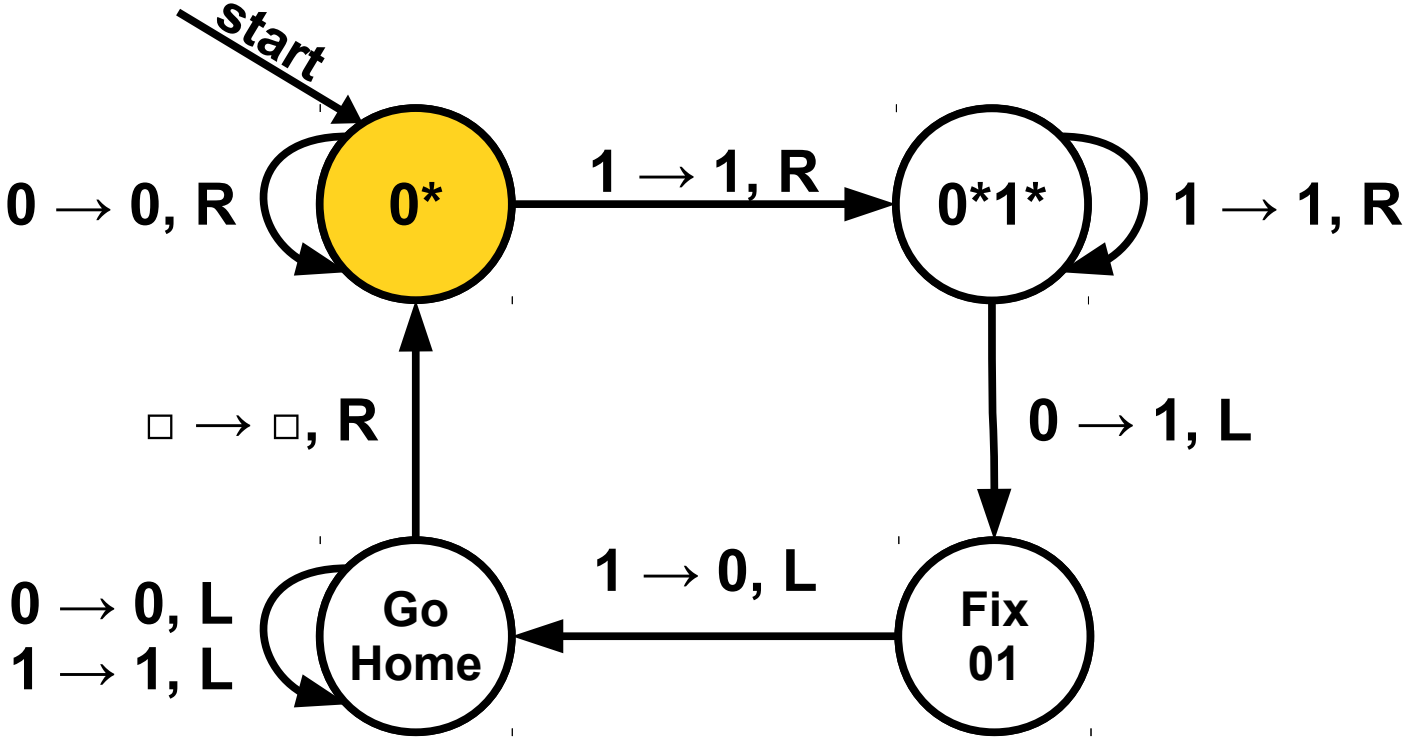

…
0 0 1 0 1 0
…

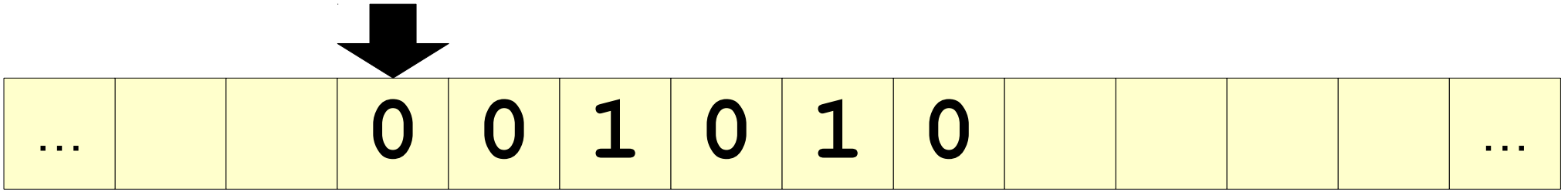

start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

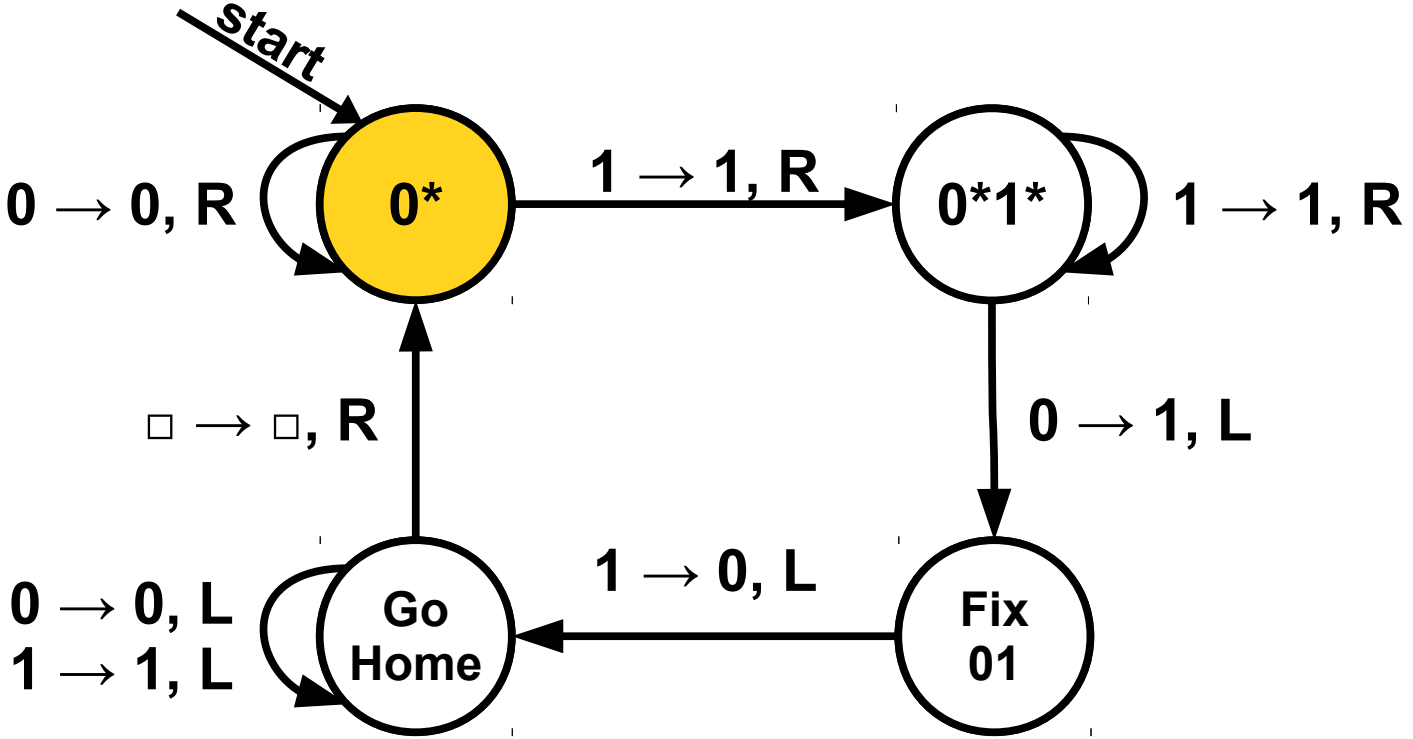

…
0 0 1 0 1 0
…

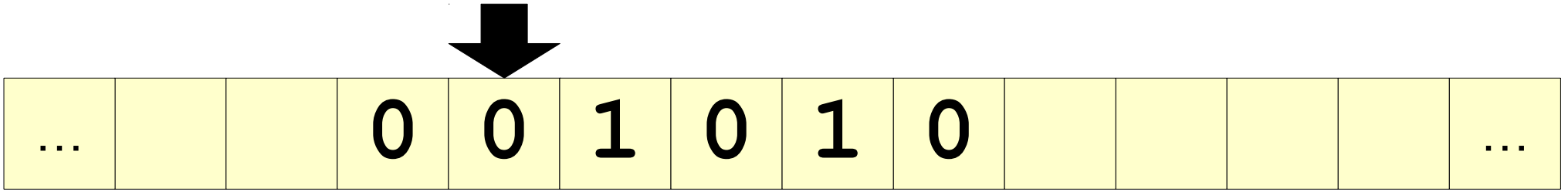

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

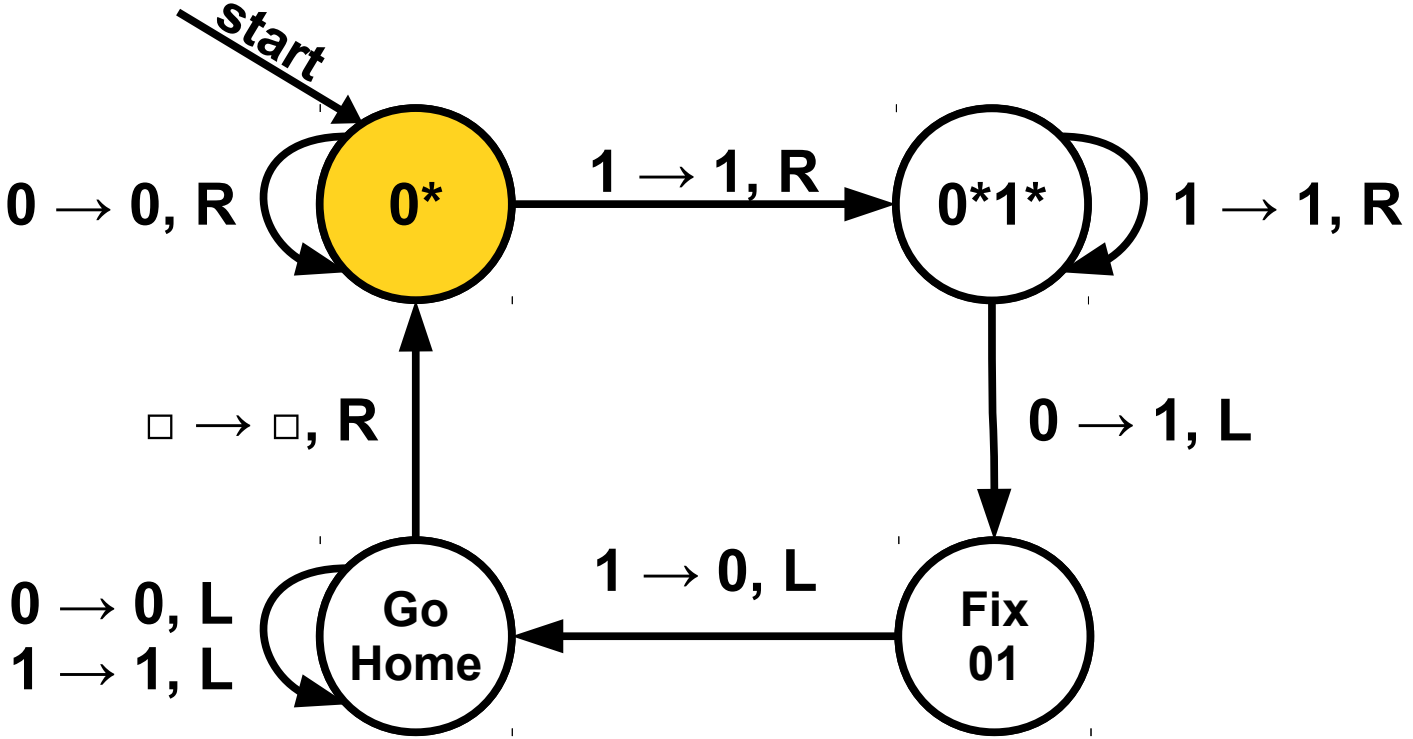

…
0 0 1 0 1 0
…

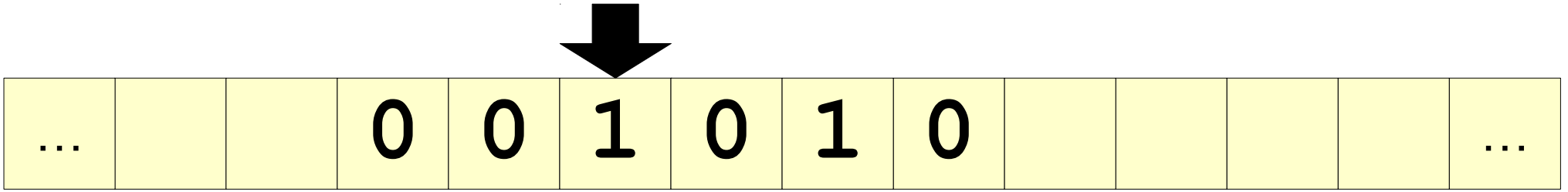

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

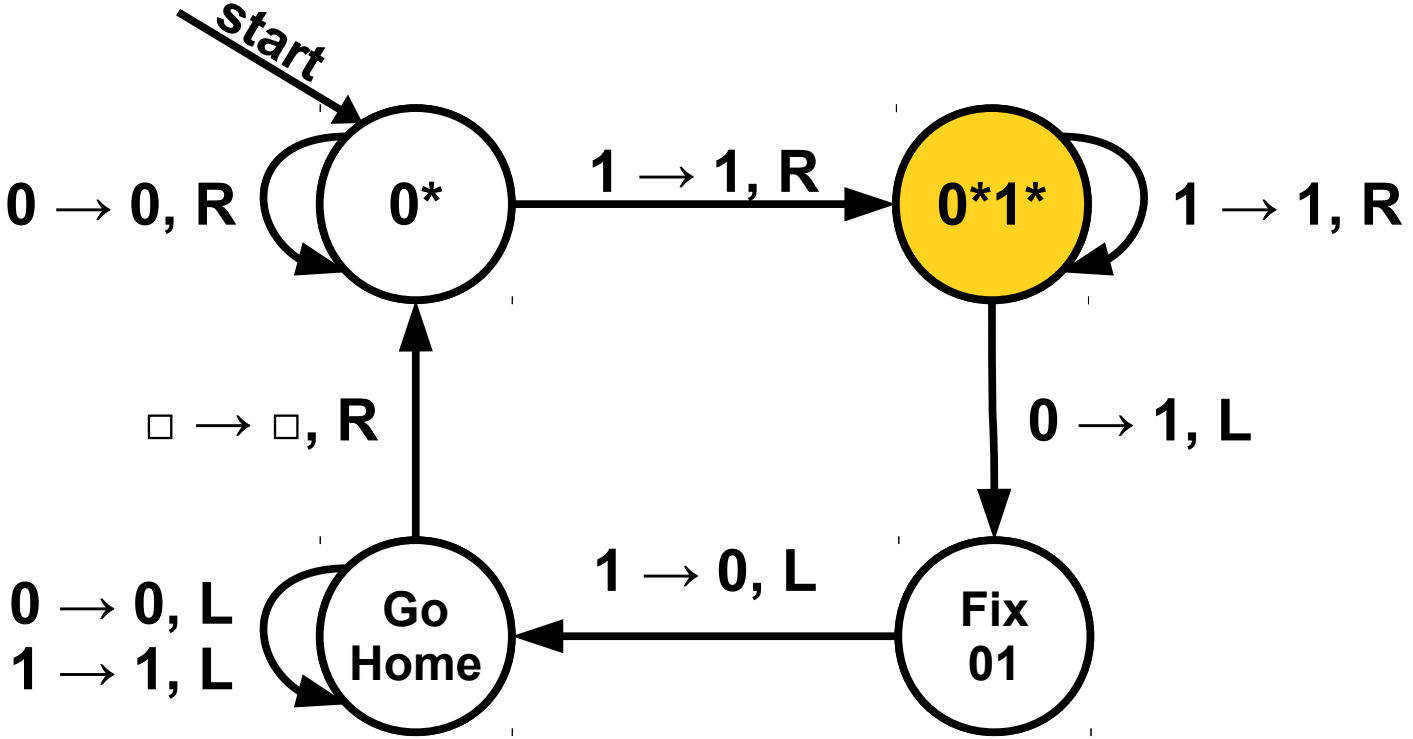

…
0 0 1 0 1 0
…

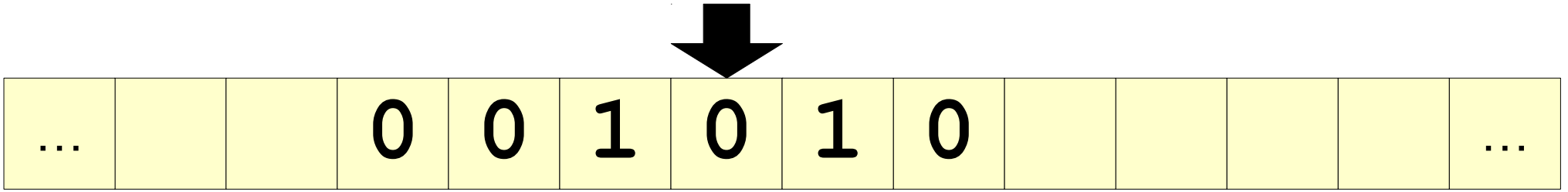

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

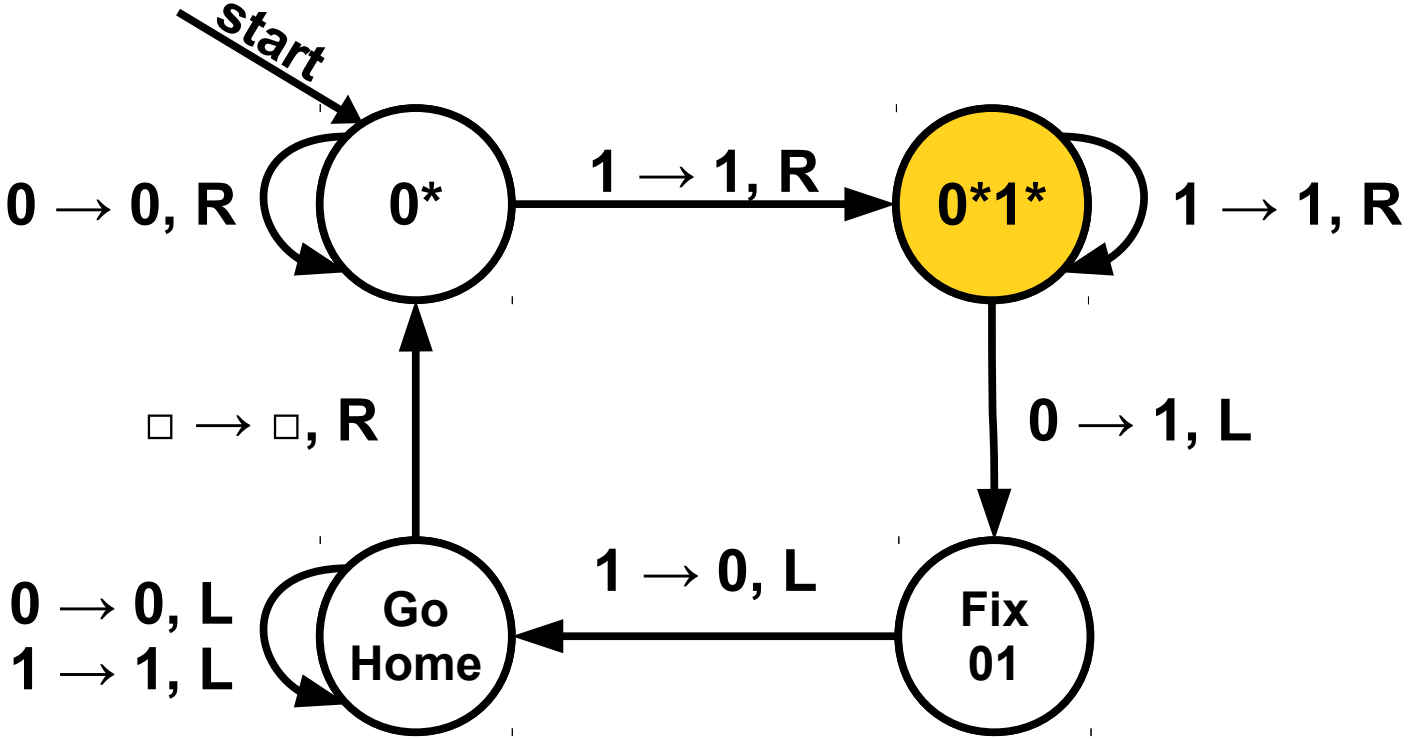

…
0 0 1 0 1 0
…

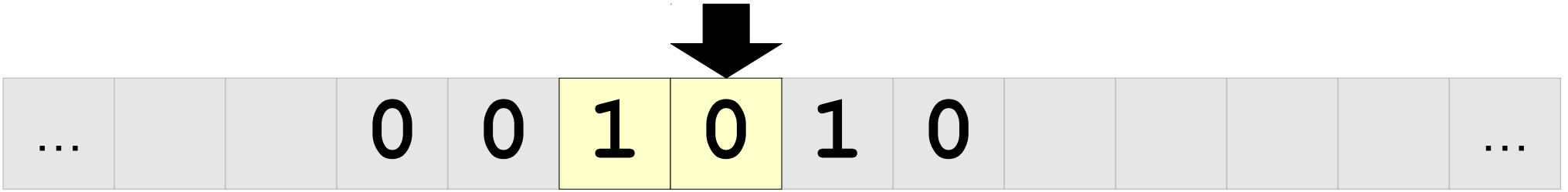

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

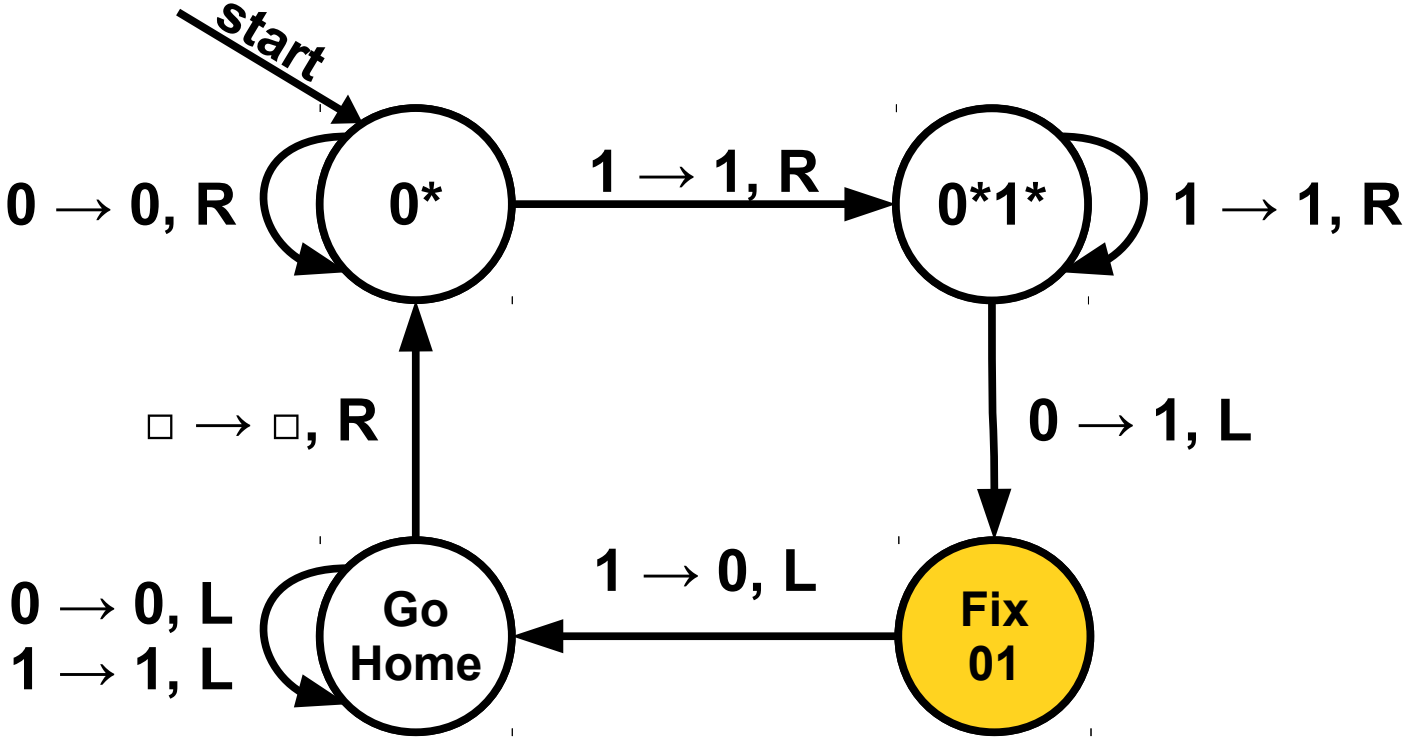

…
0 0 1 0 1 0
…

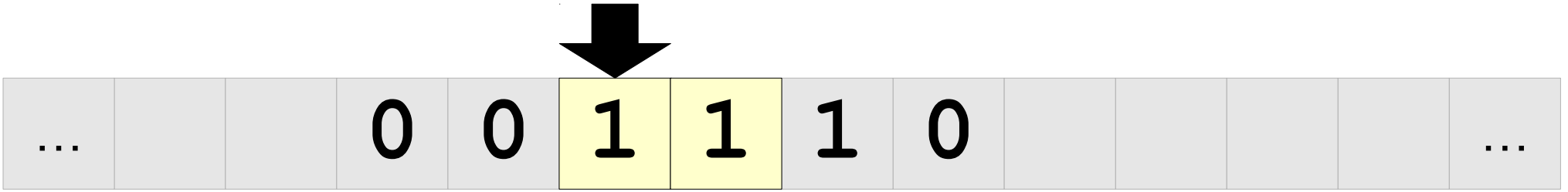

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

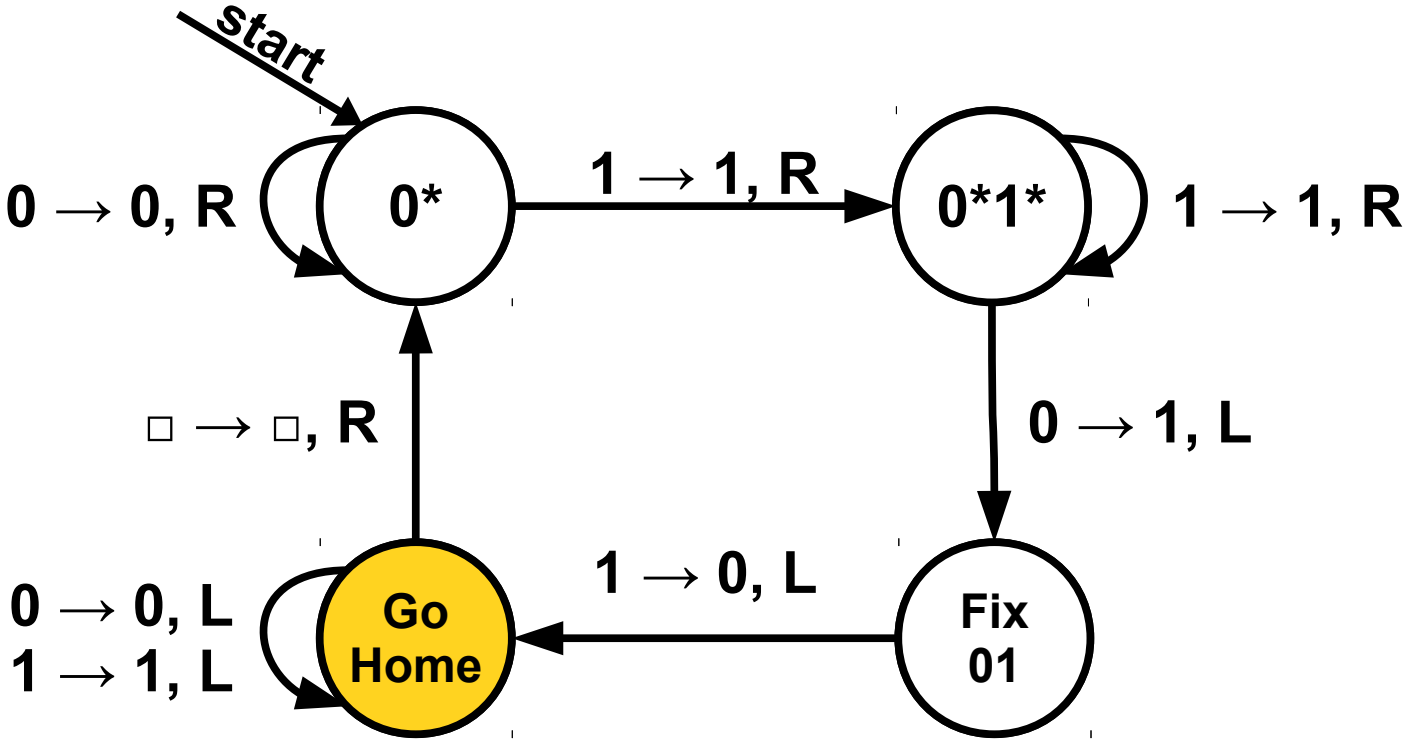

…
0 0 1 0 1 0
…

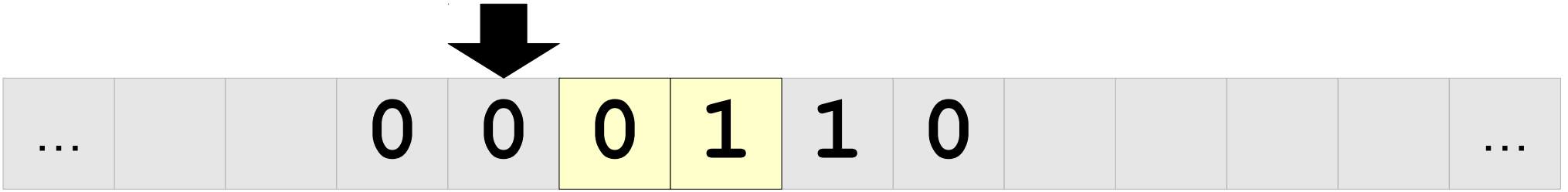

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

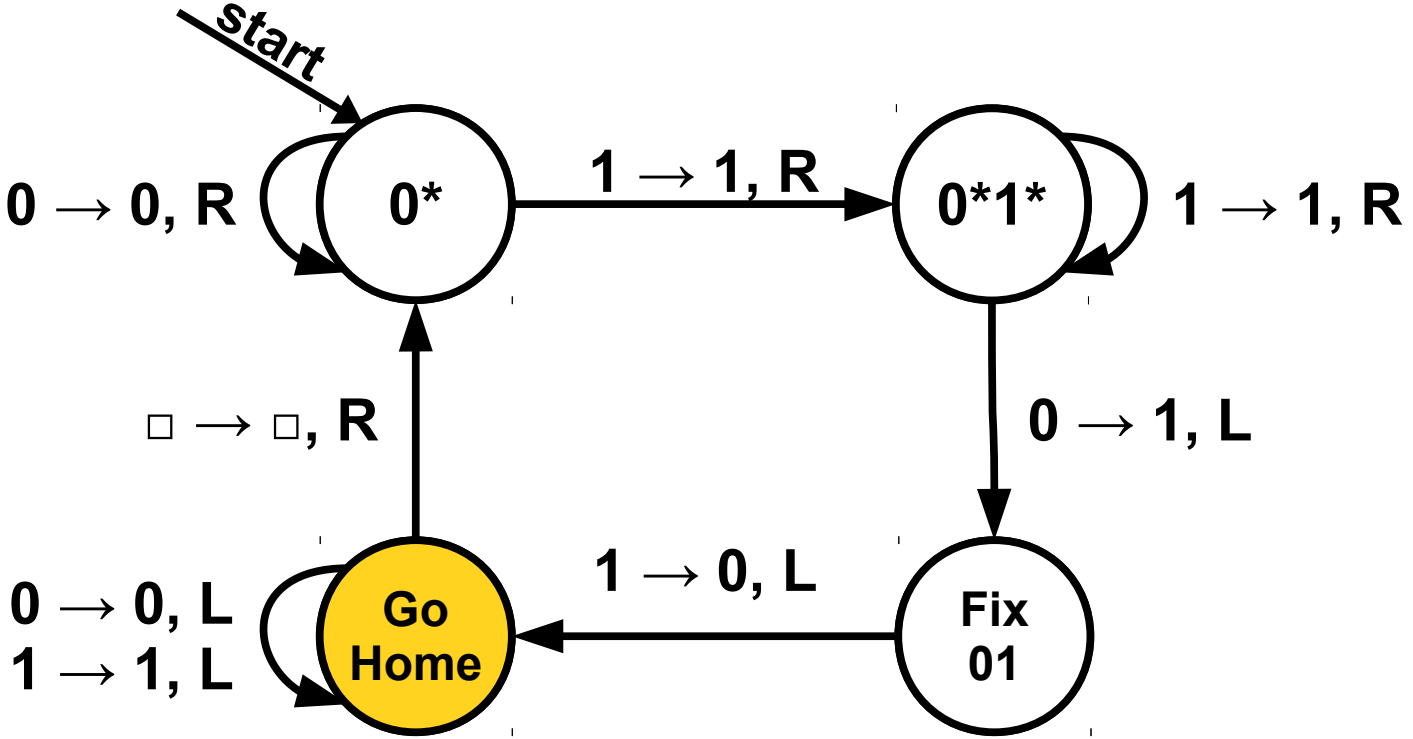

…
0 0 1 0 1 0
…

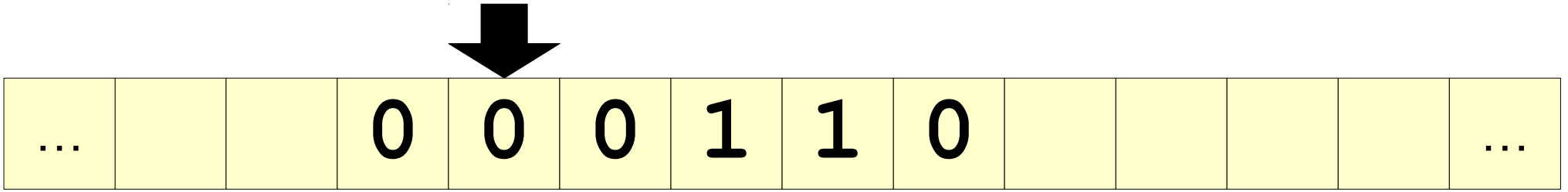

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

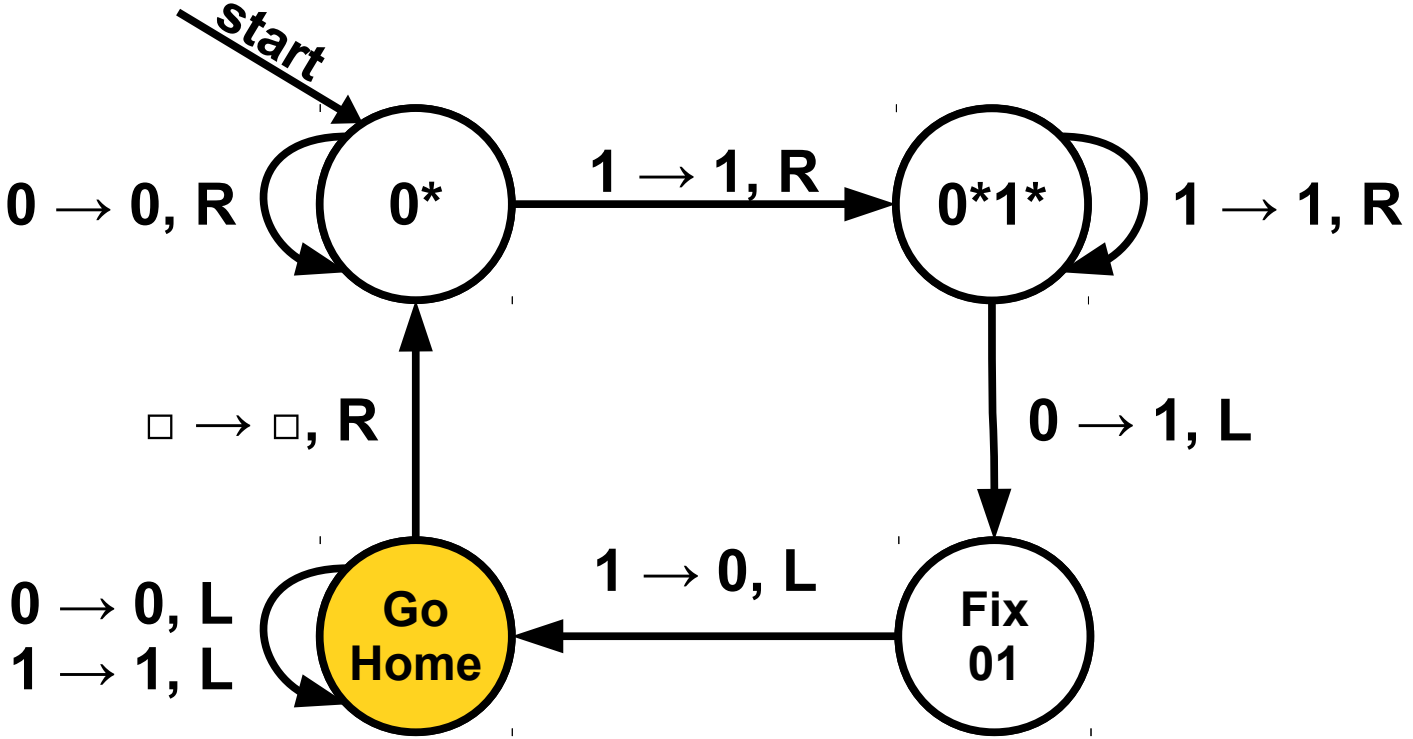

…
0 0 1 1 1 0
…

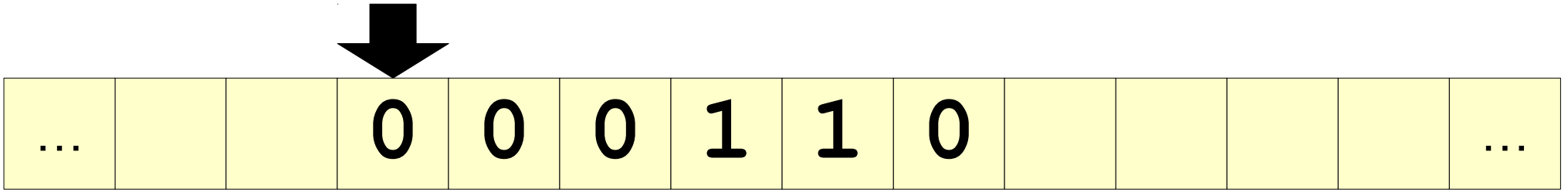

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

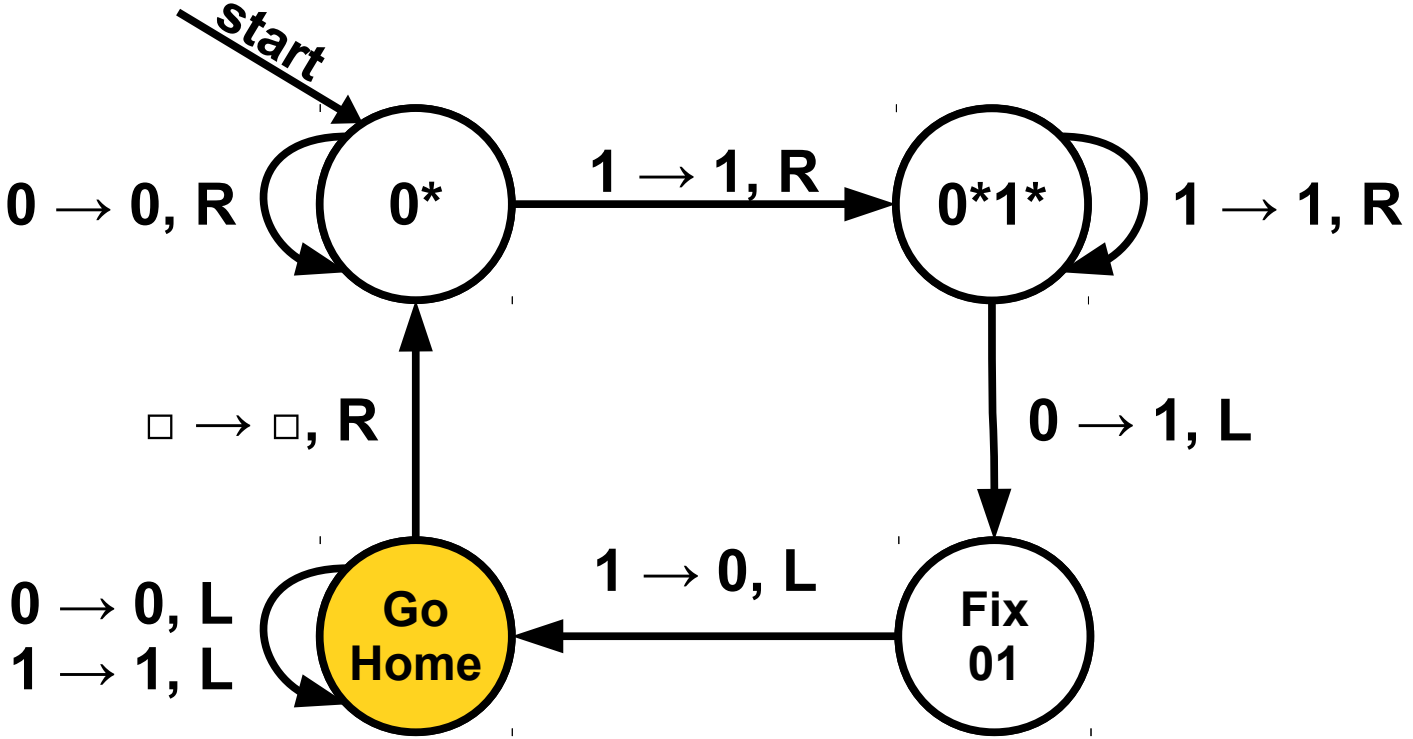

…
0 0 0 1 1 0
…

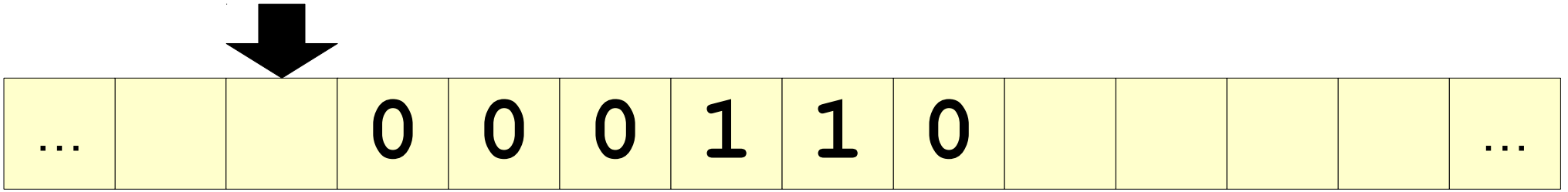

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

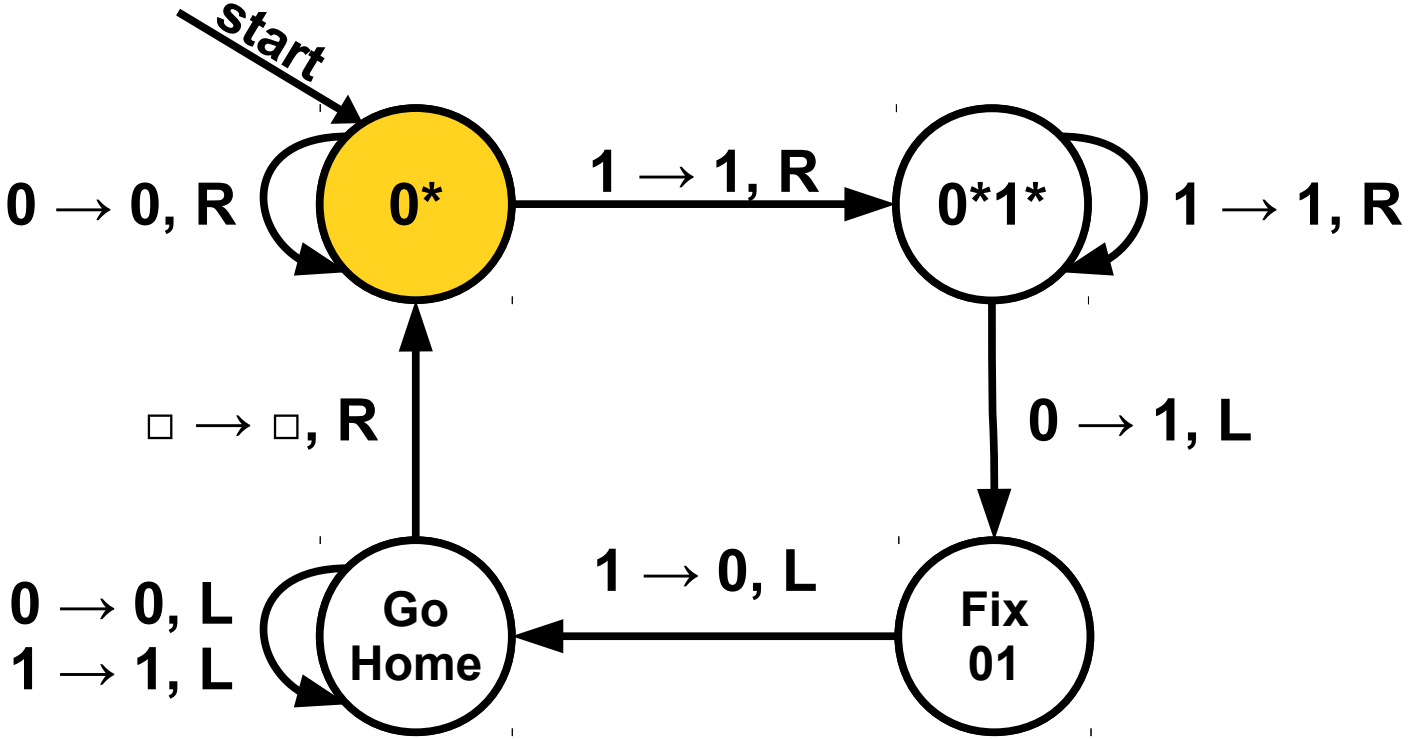

…
0 0 0 1 1 0
…

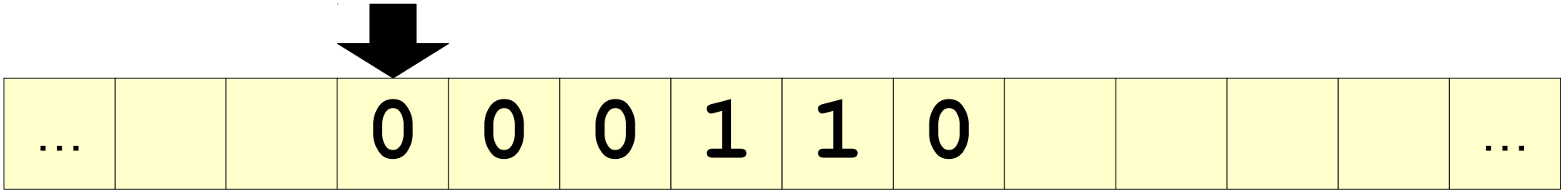

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

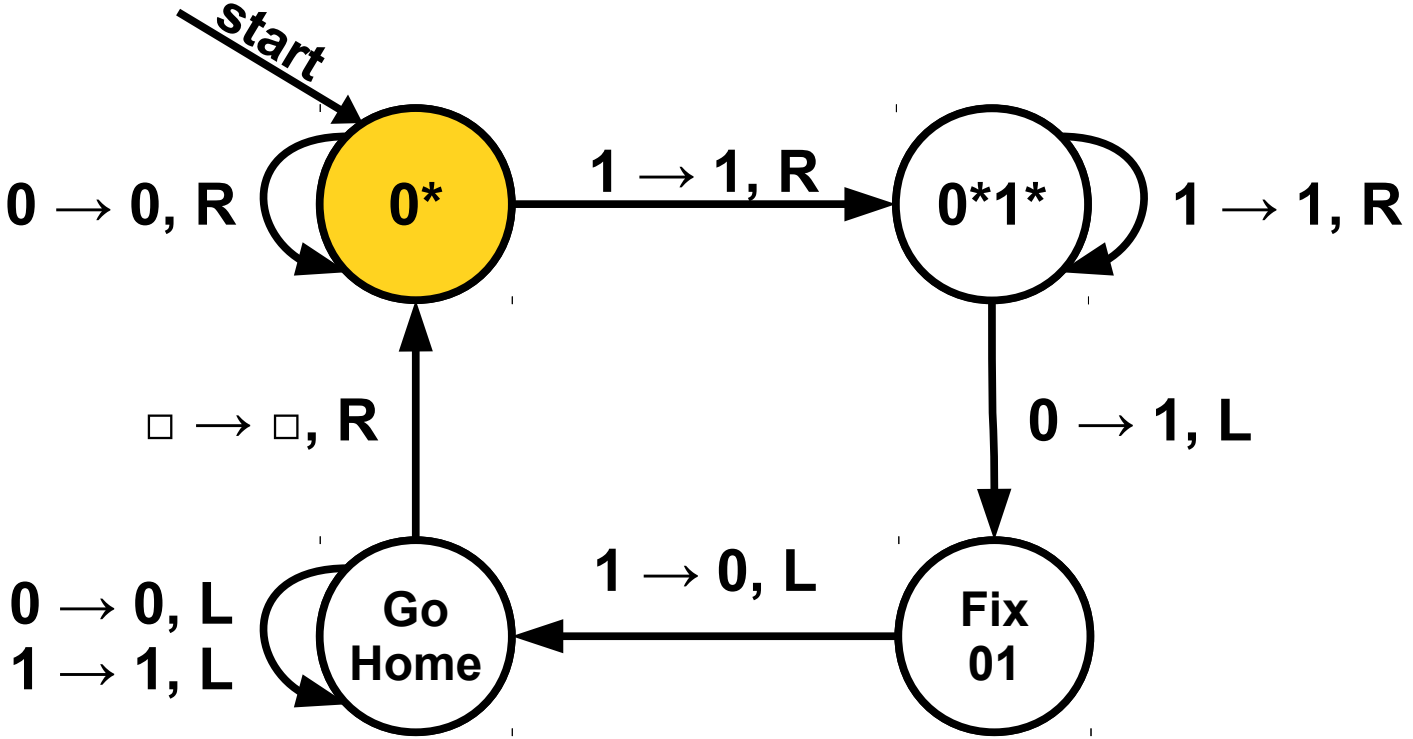

…
0 0 0 1 1 0
…

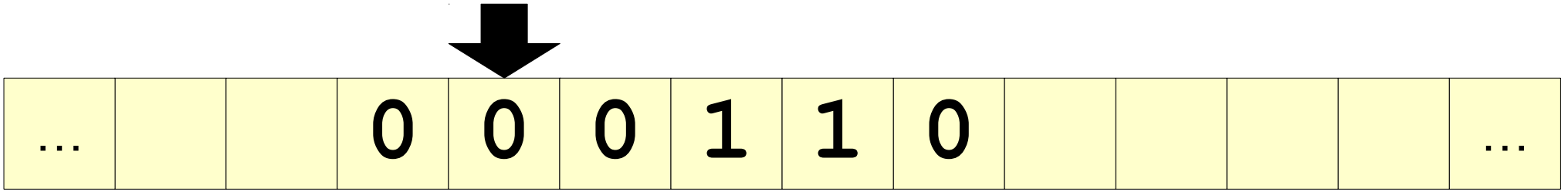

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

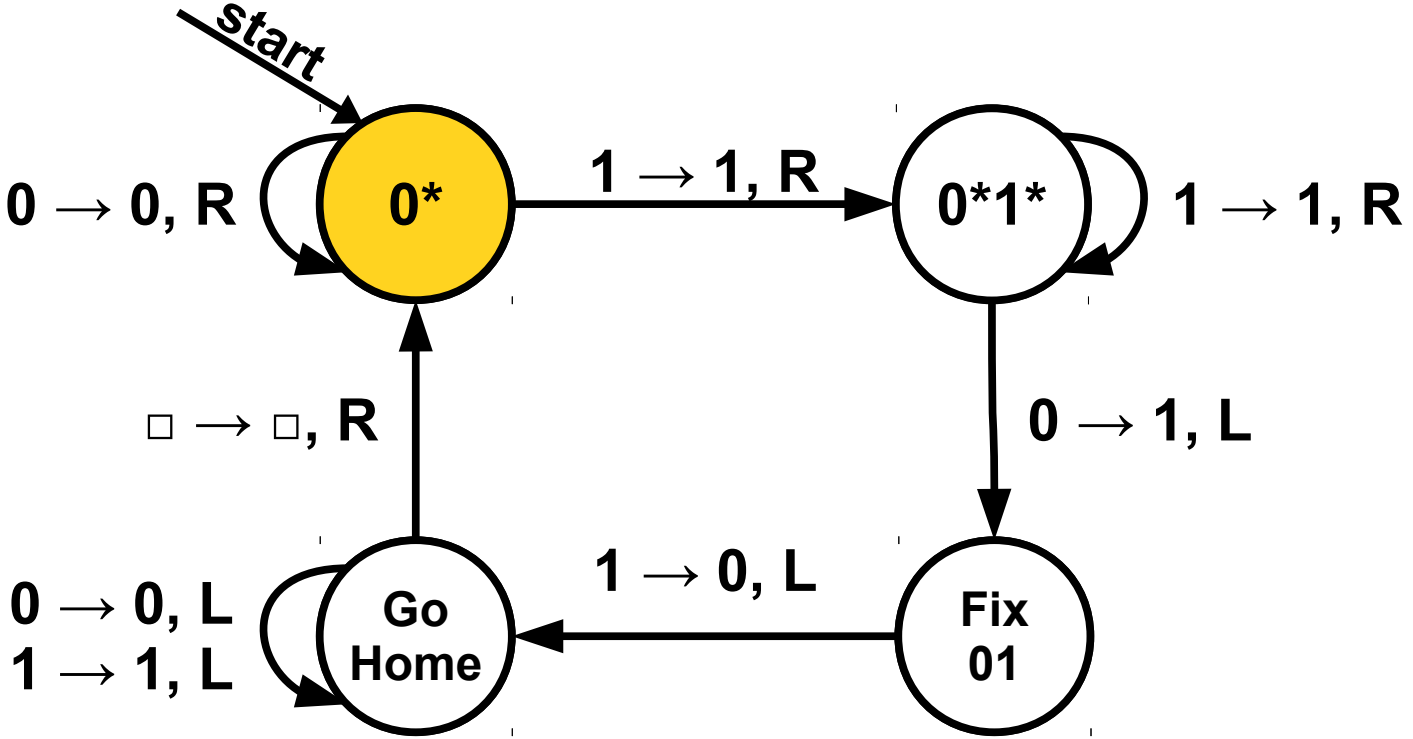

…
0 0 0 1 1 0
…

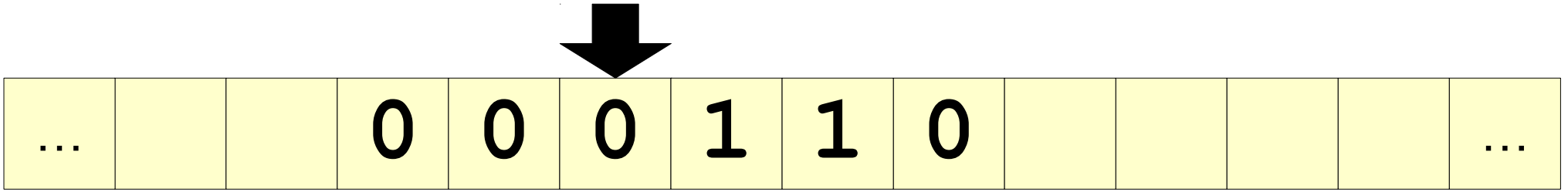

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

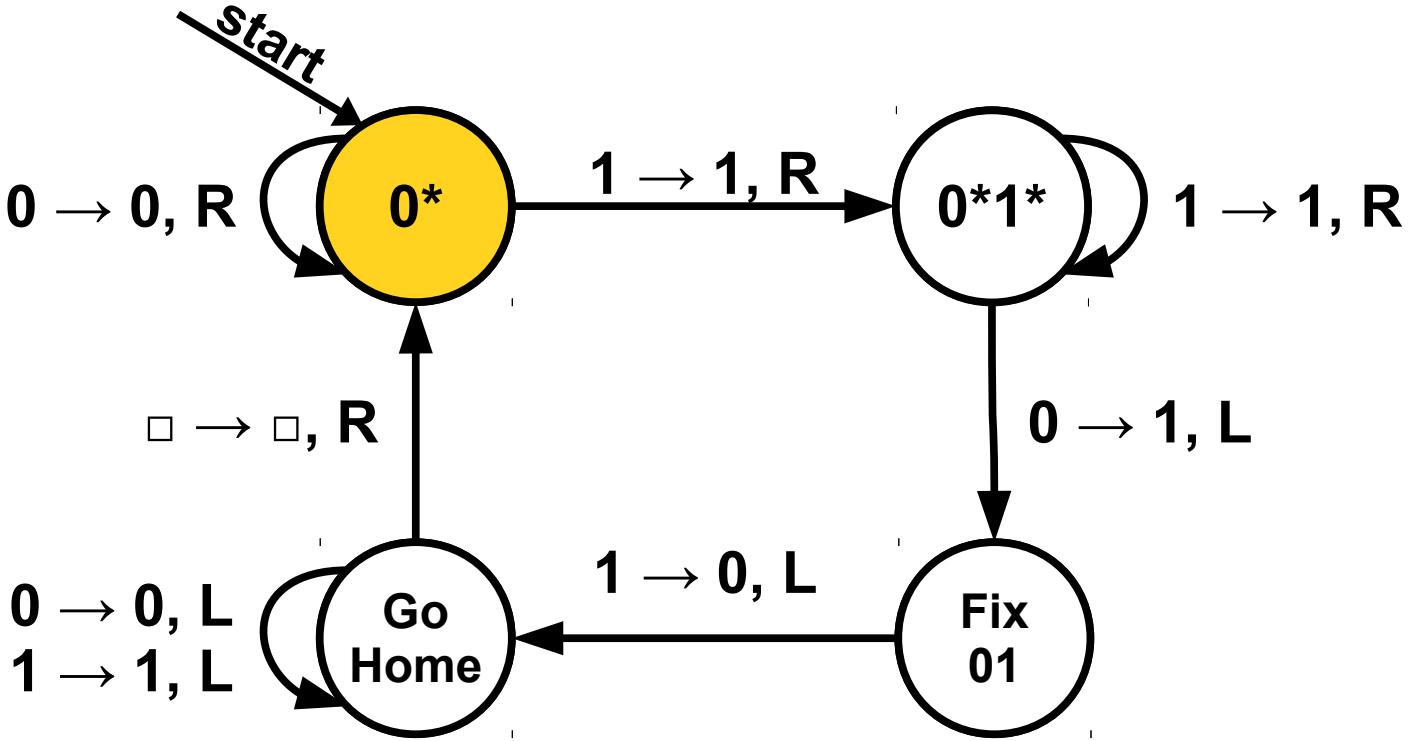

…
0 0 0 1 1 0
…

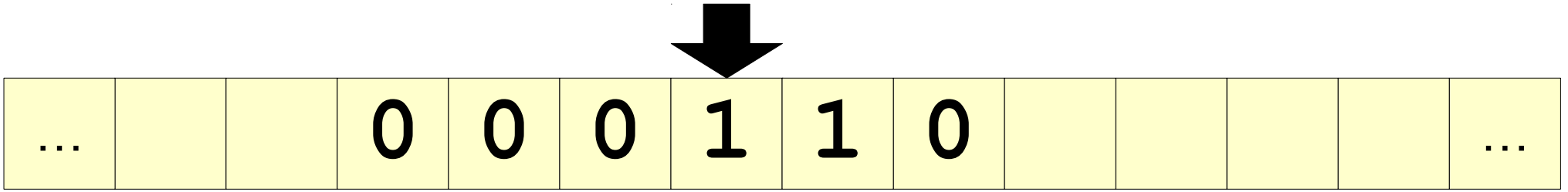

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

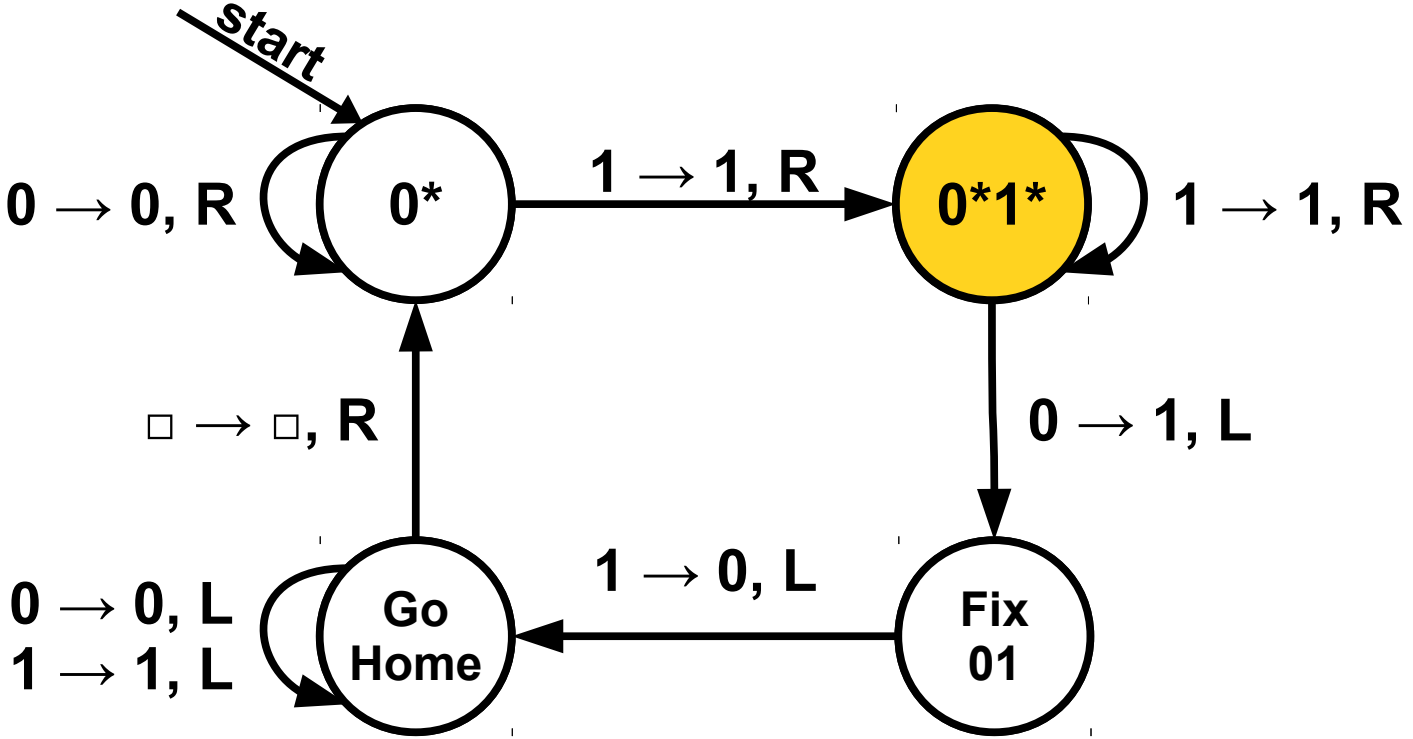

…
0 0 0 1 1 0
…

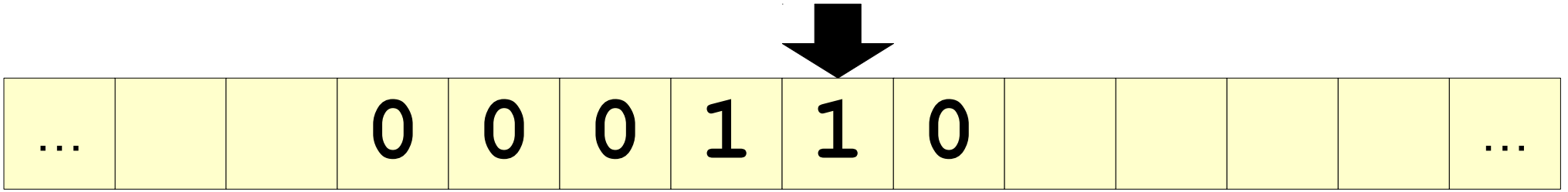

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

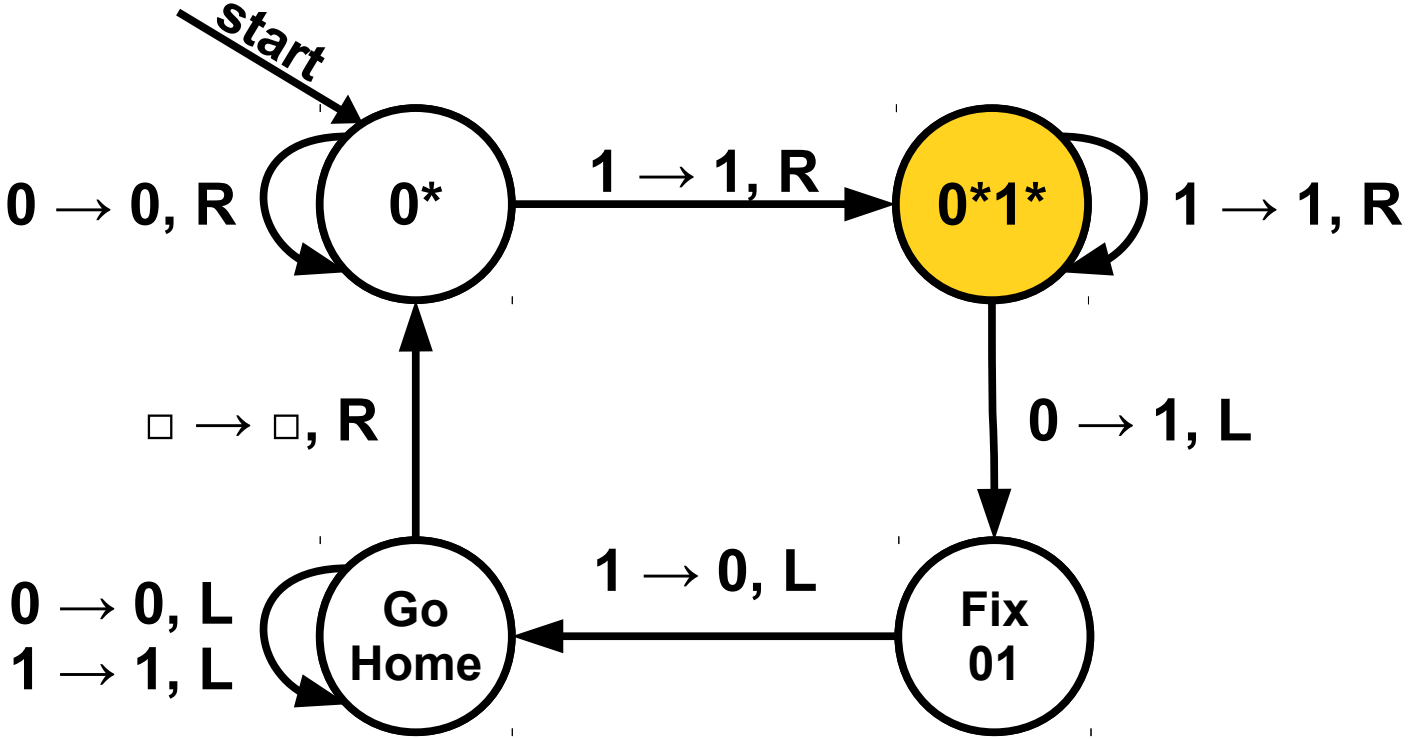

…
0 0 0 1 1 0
…

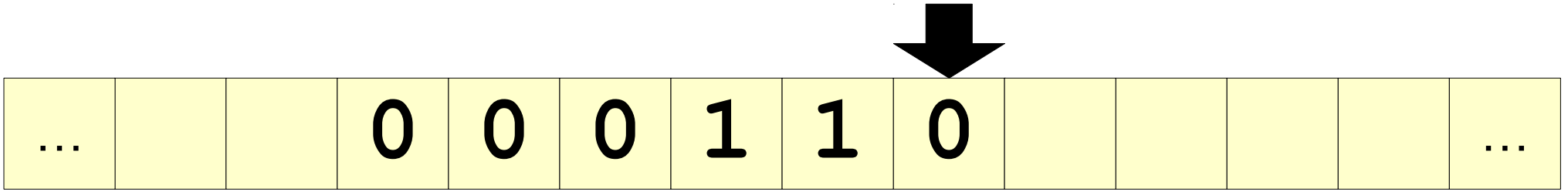

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

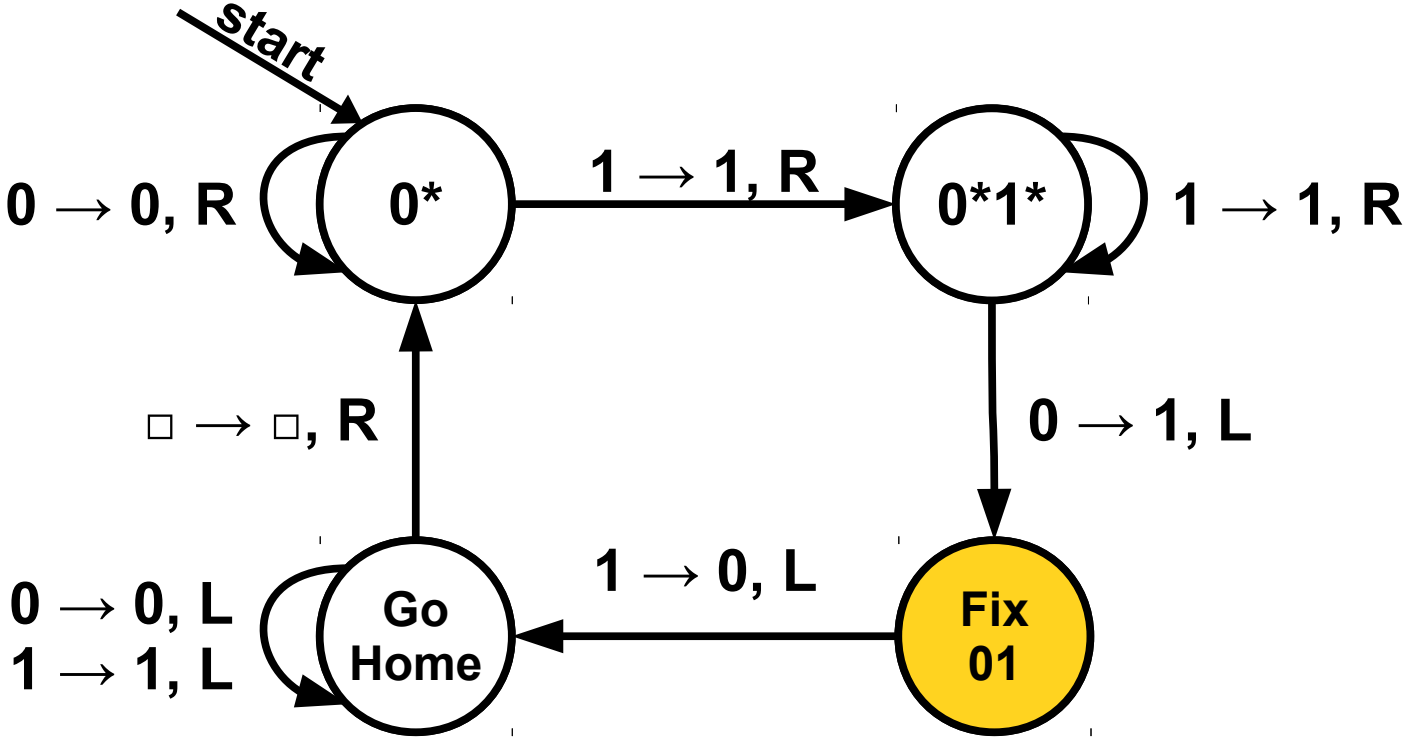

…
0 0 0 1 1 0
…

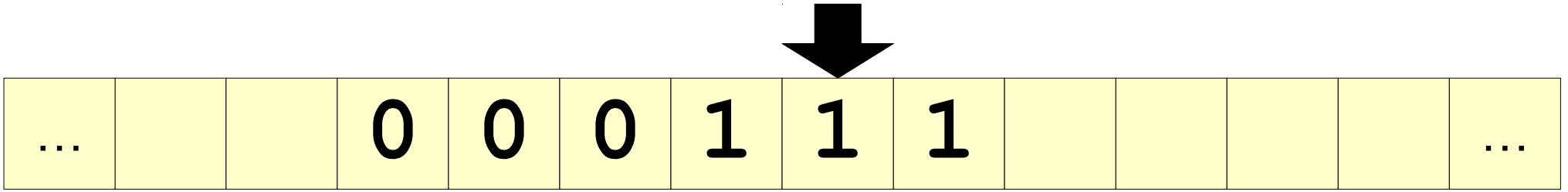

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

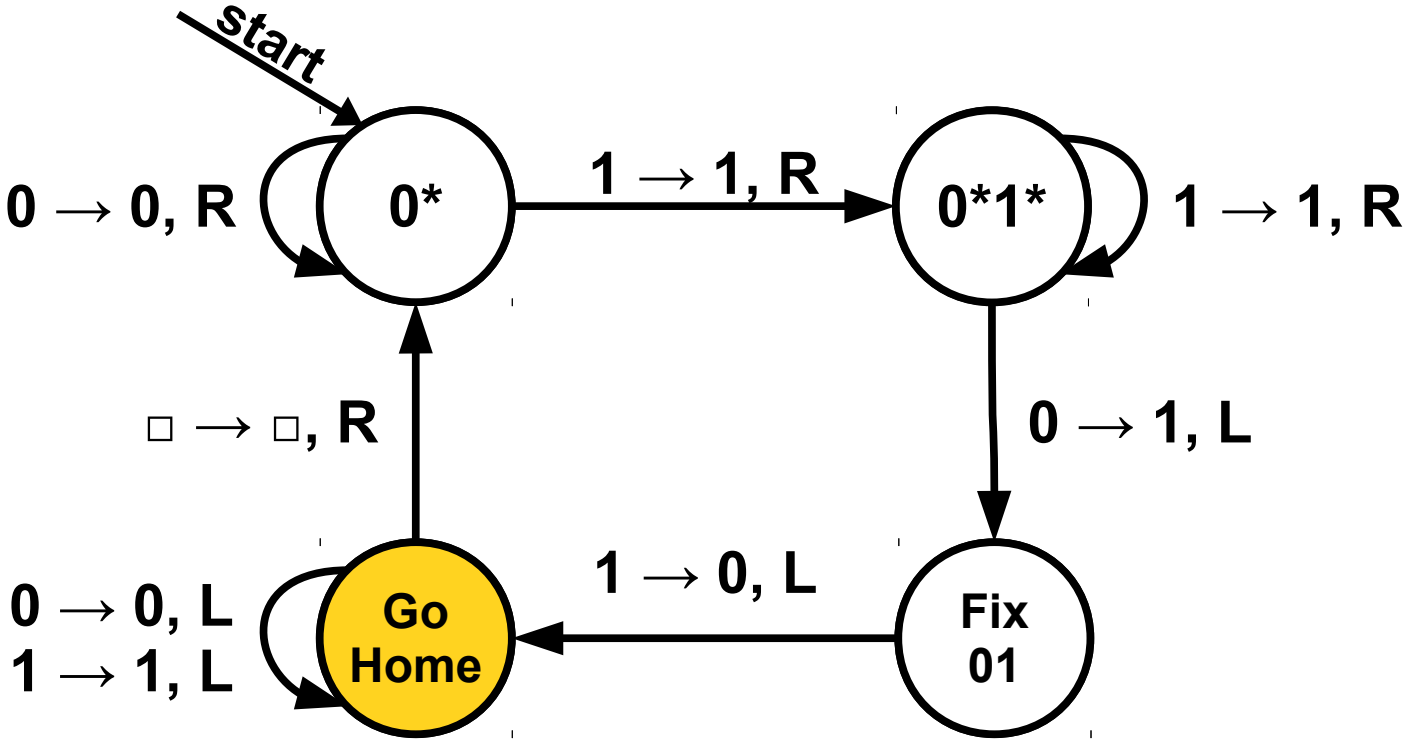

…
0 0 0 1 1 0
…

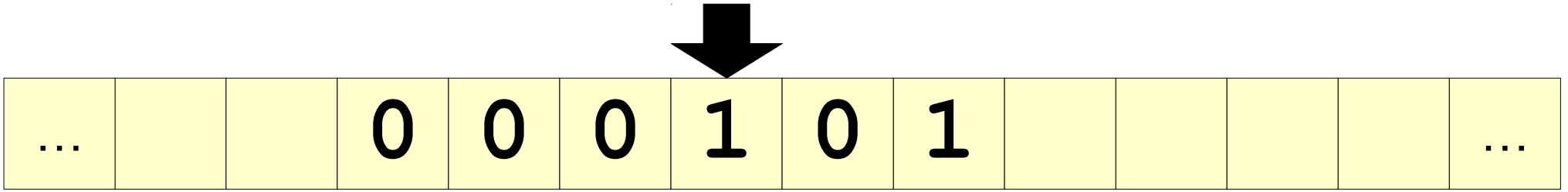

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

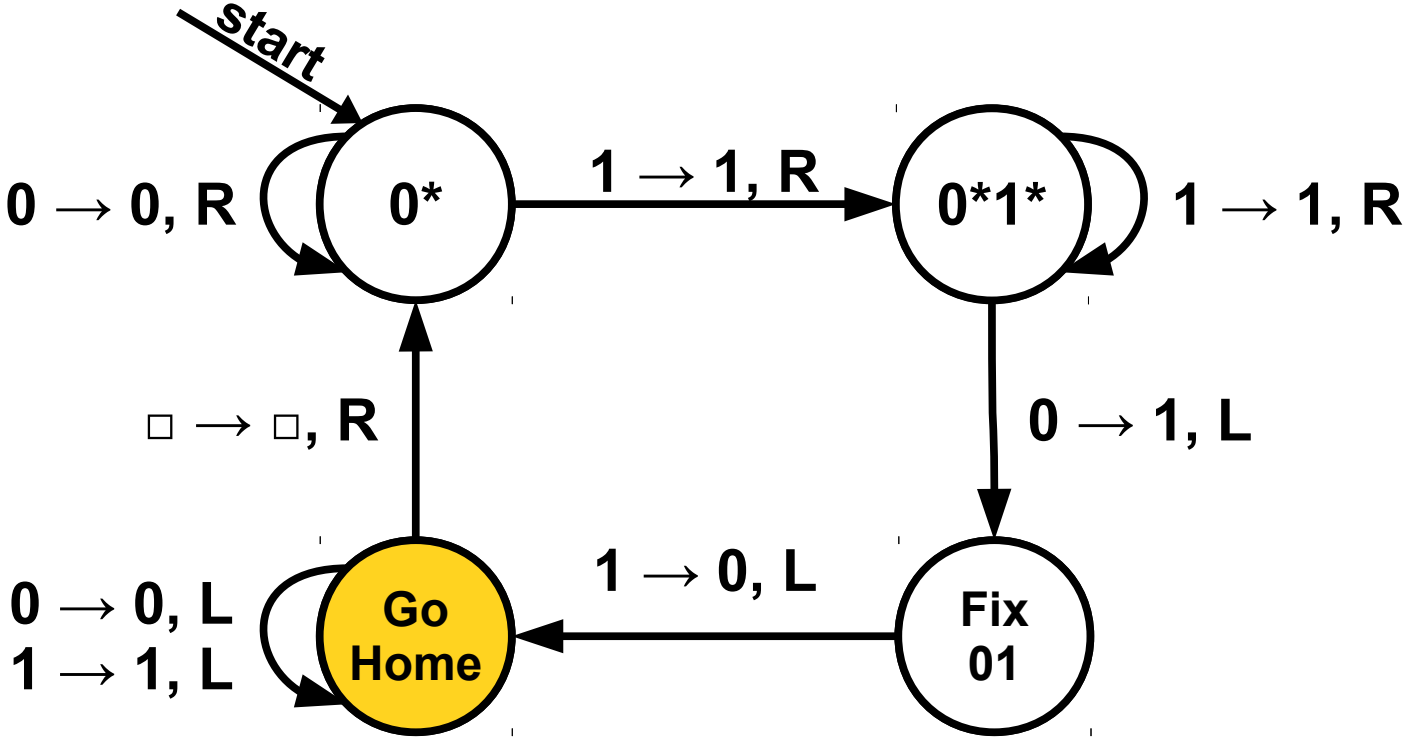

…
0 0 0 1 1 0
…

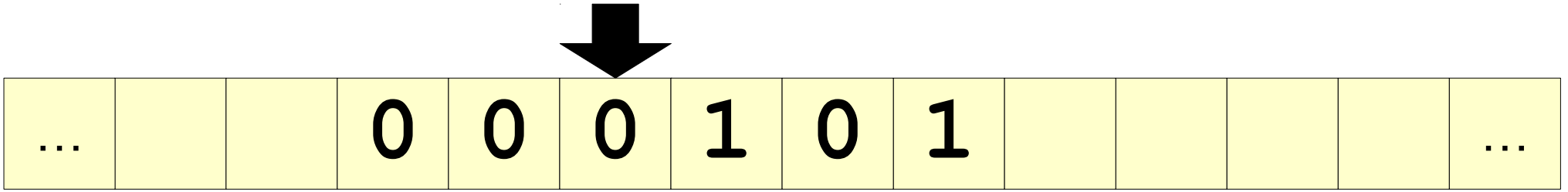

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

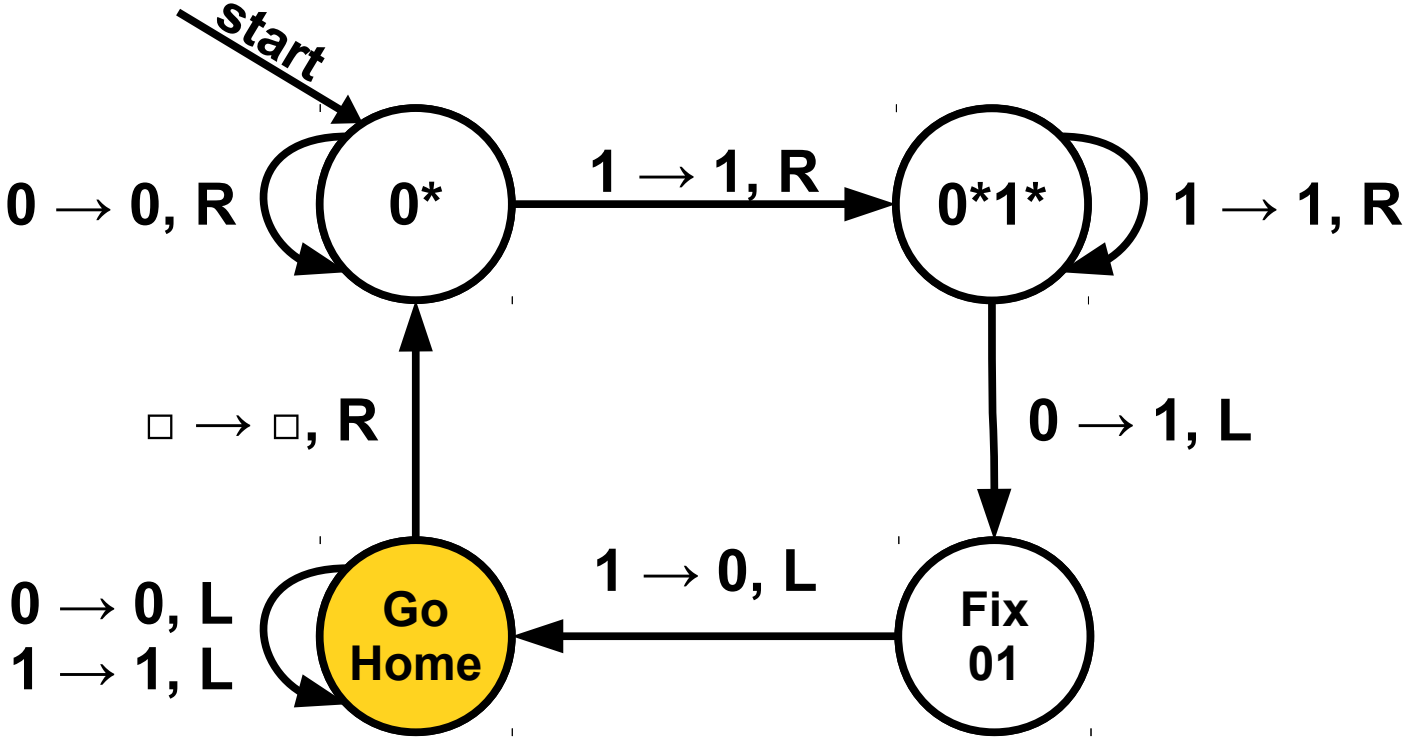

…
0 0 0 1 1 1
…

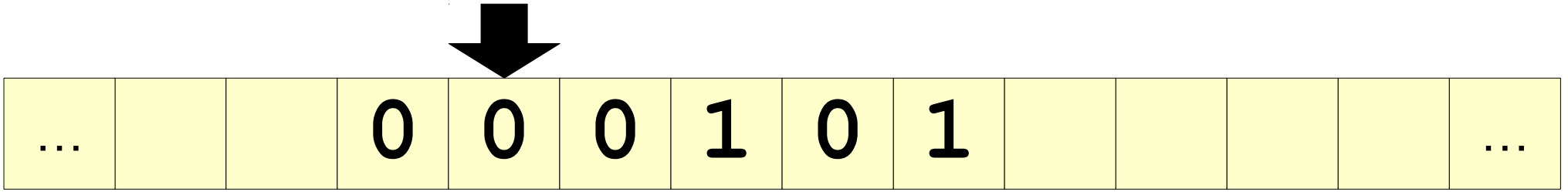

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

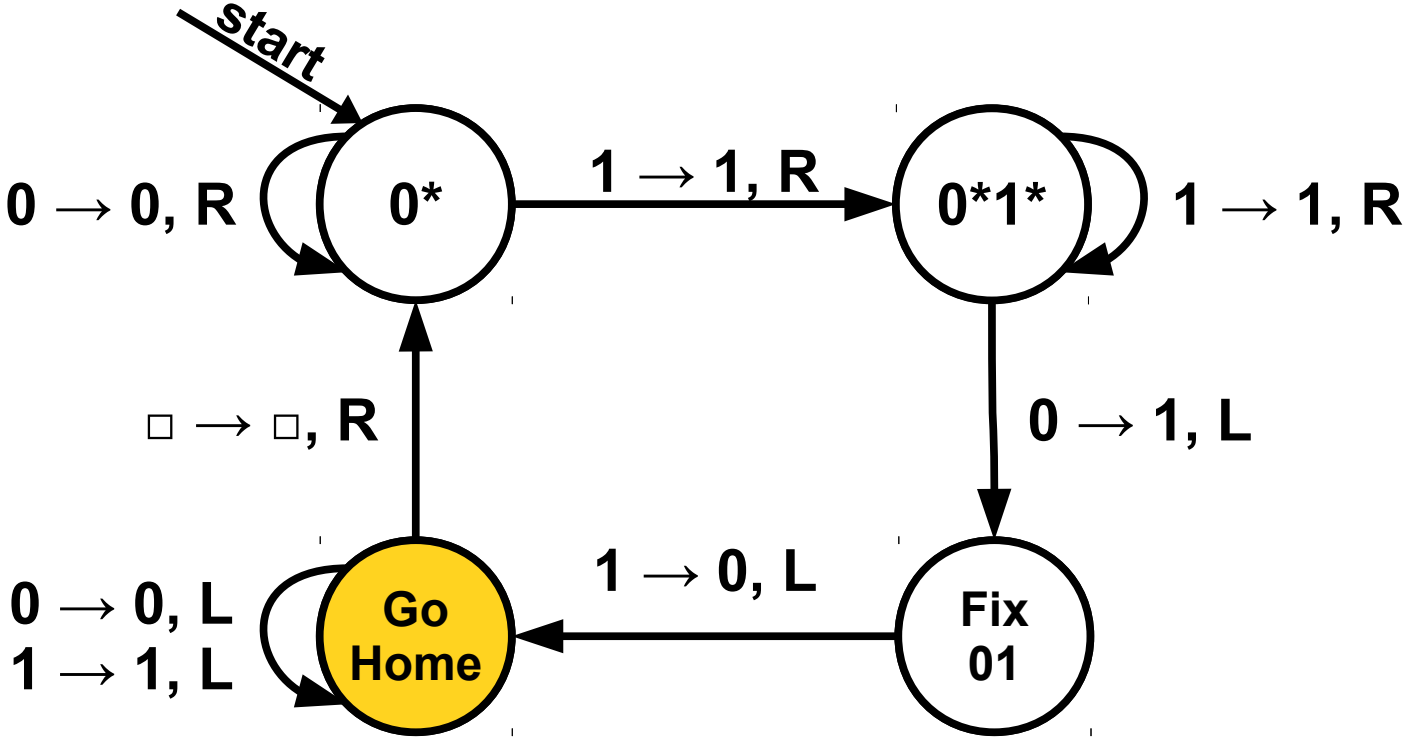

…
0 0 0 1 0 1
…

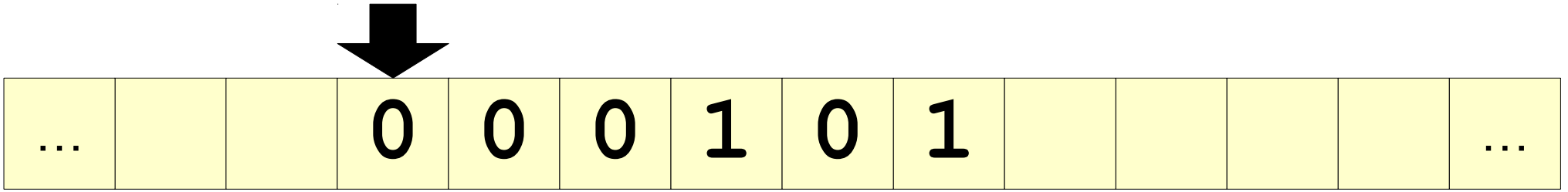

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

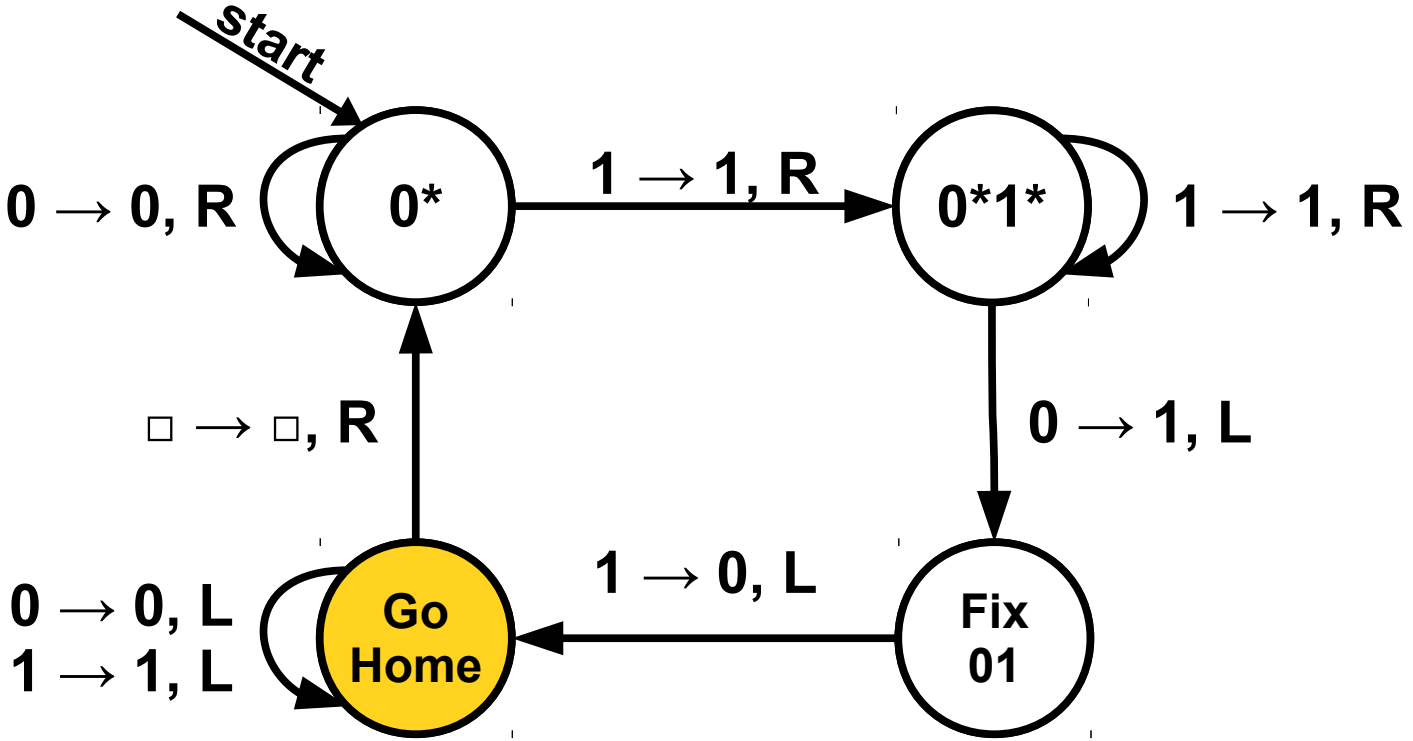

…
0 0 0 1 0 1
…

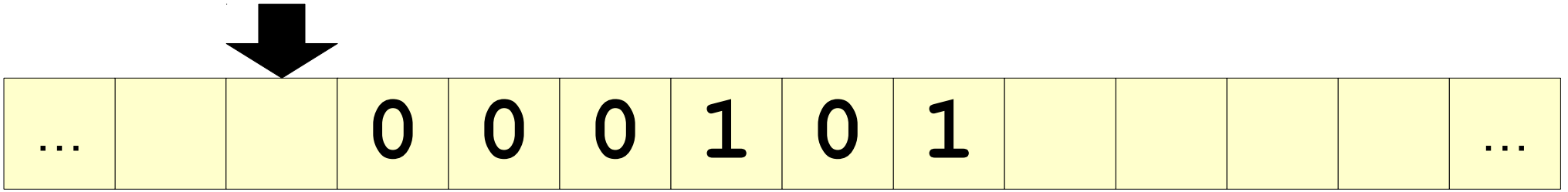

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

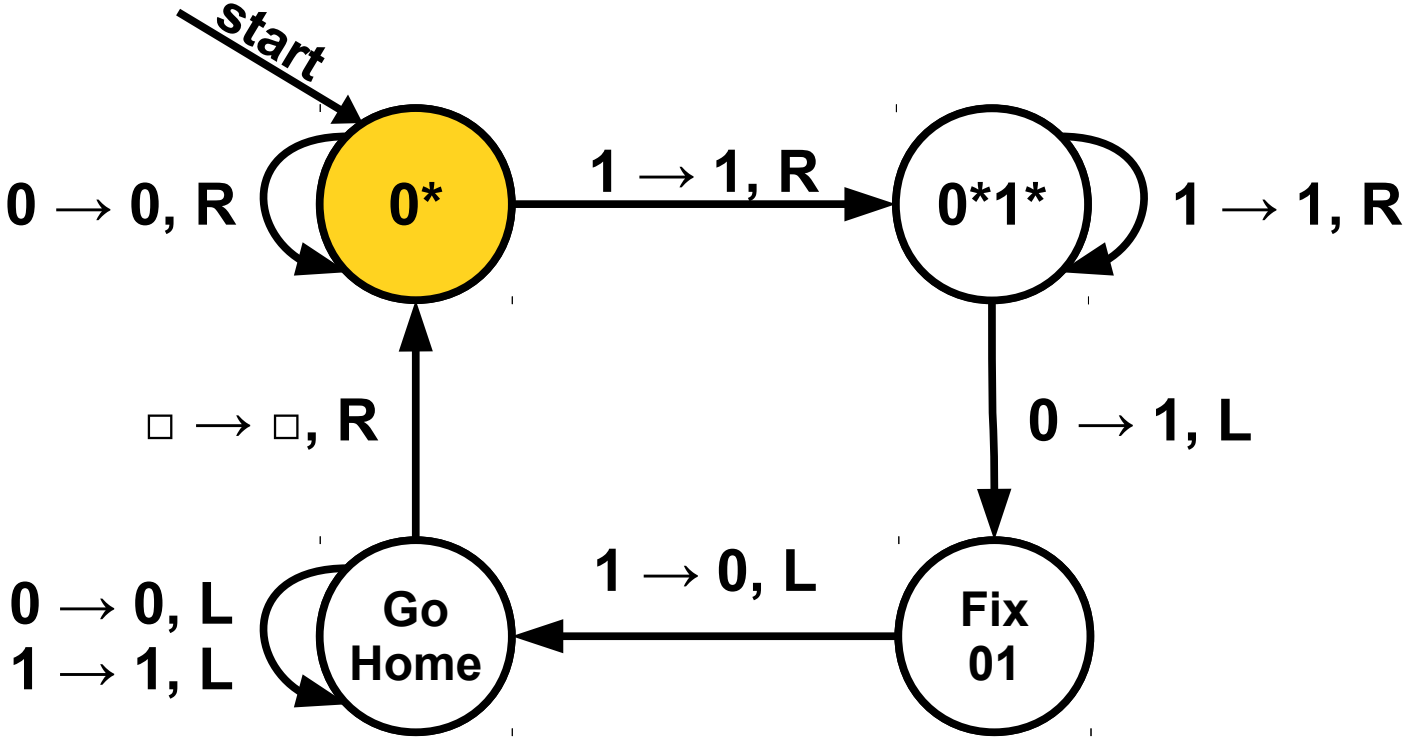

…
0 0 0 1 0 1
…

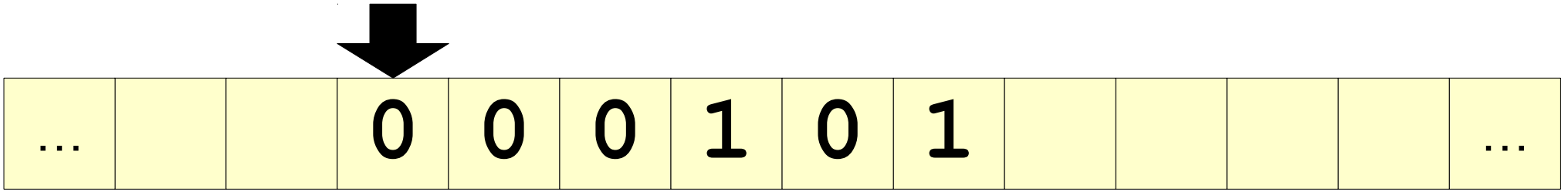

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

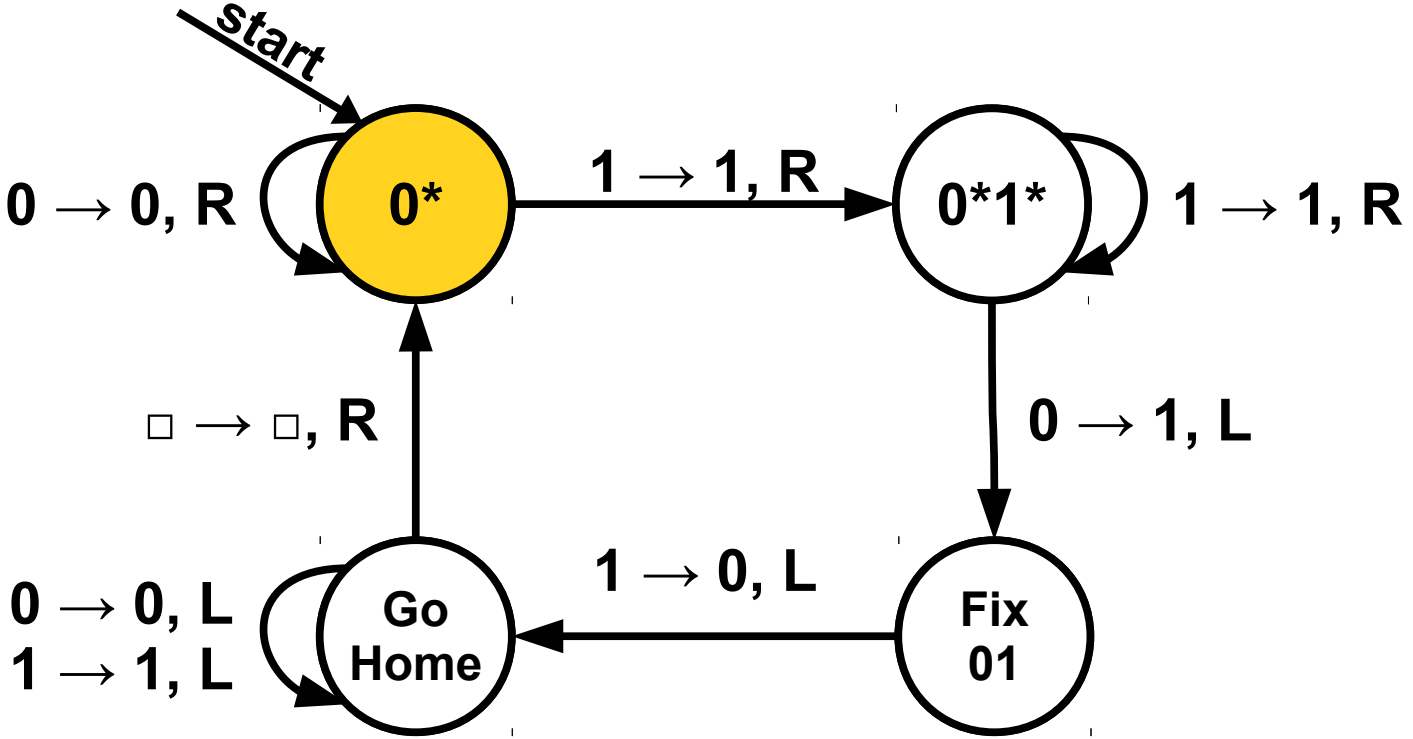

…
0 0 0 1 0 1
…

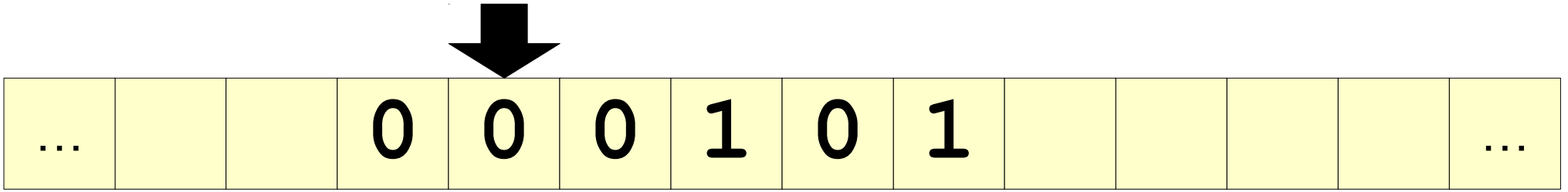

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

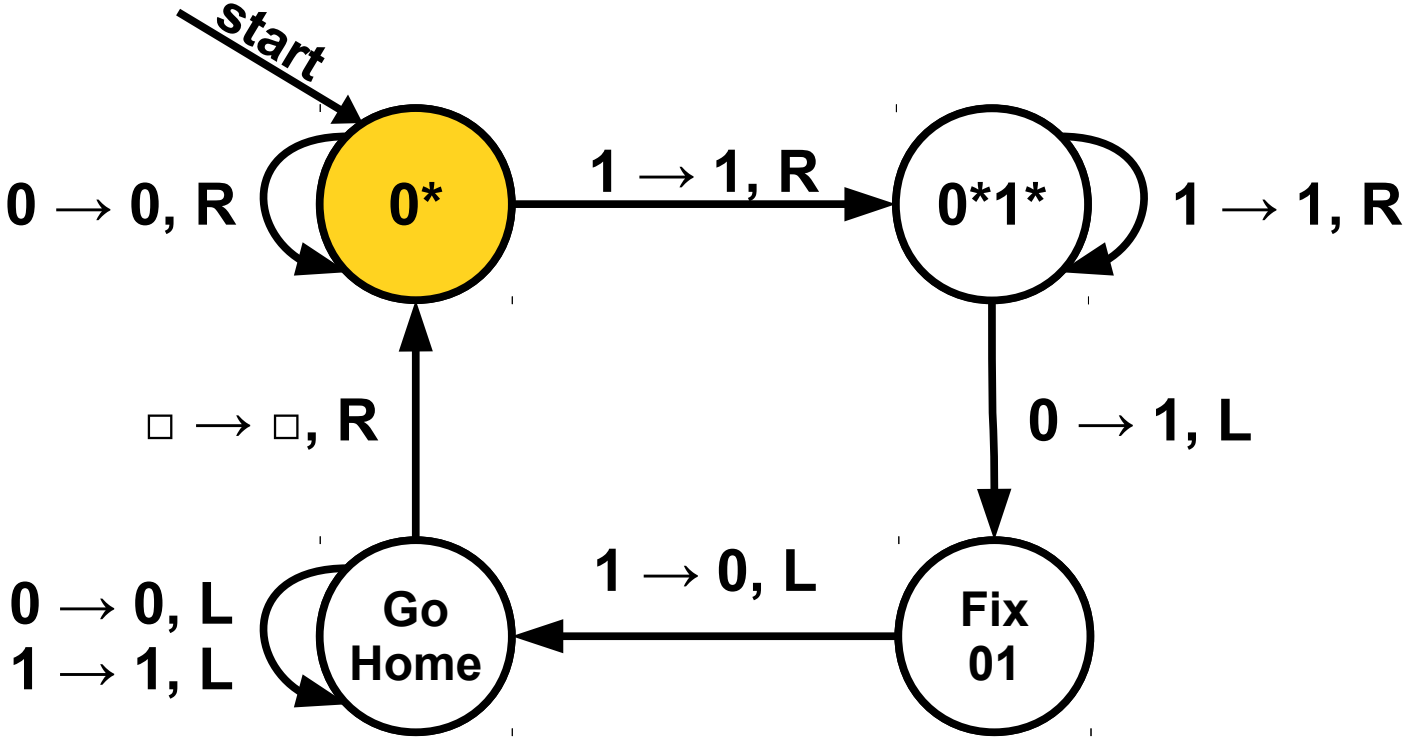

…
0 0 0 1 0 1
…

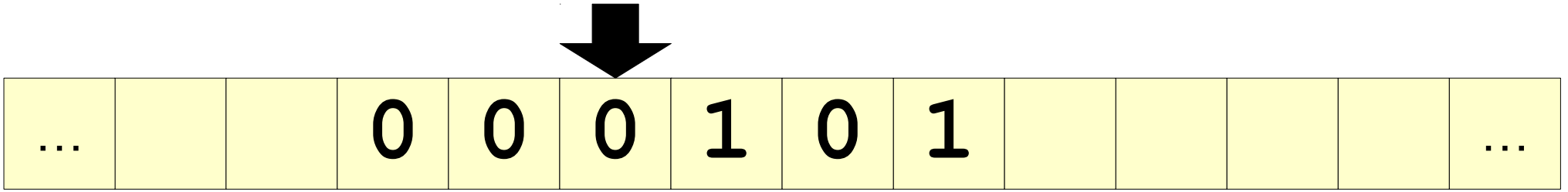

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

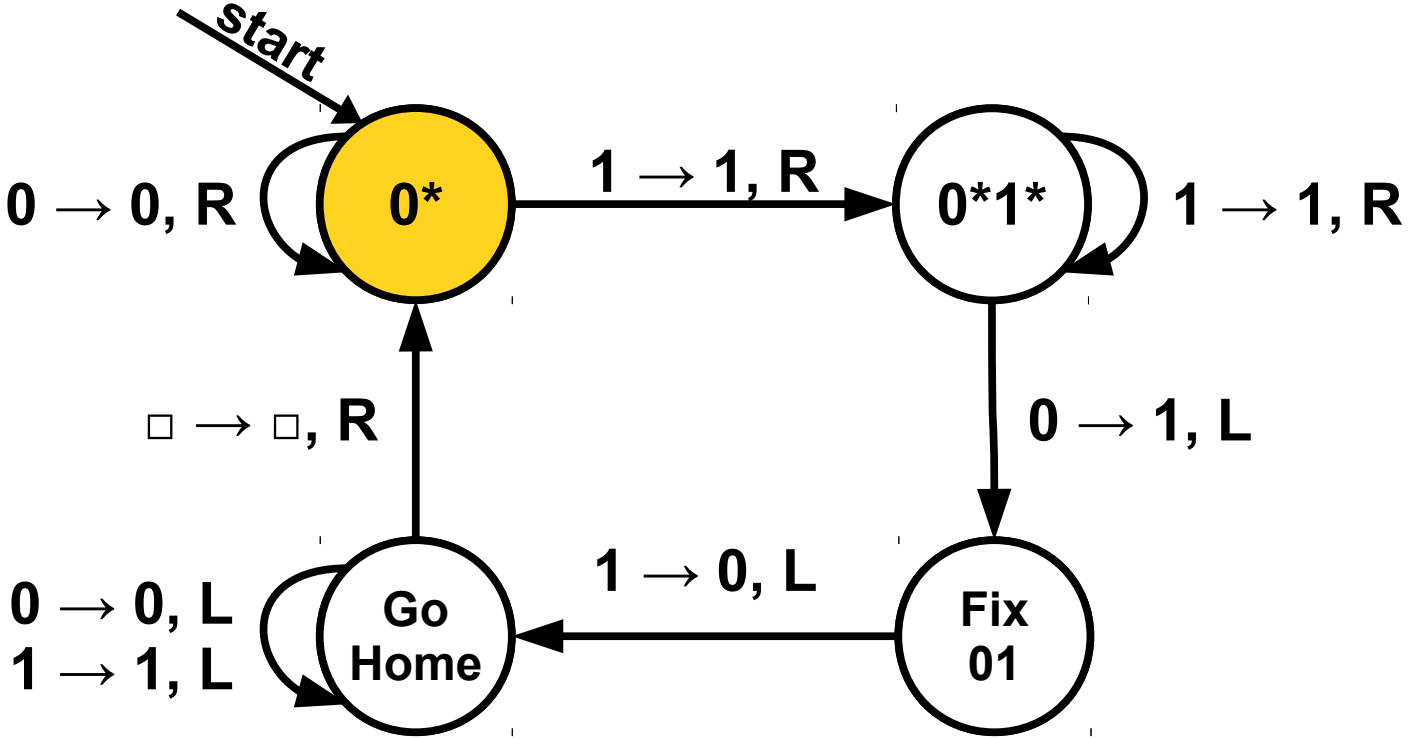

…
0 0 0 1 0 1
…

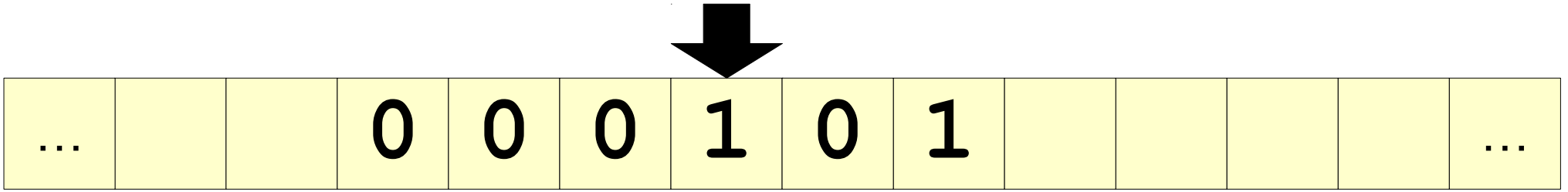

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

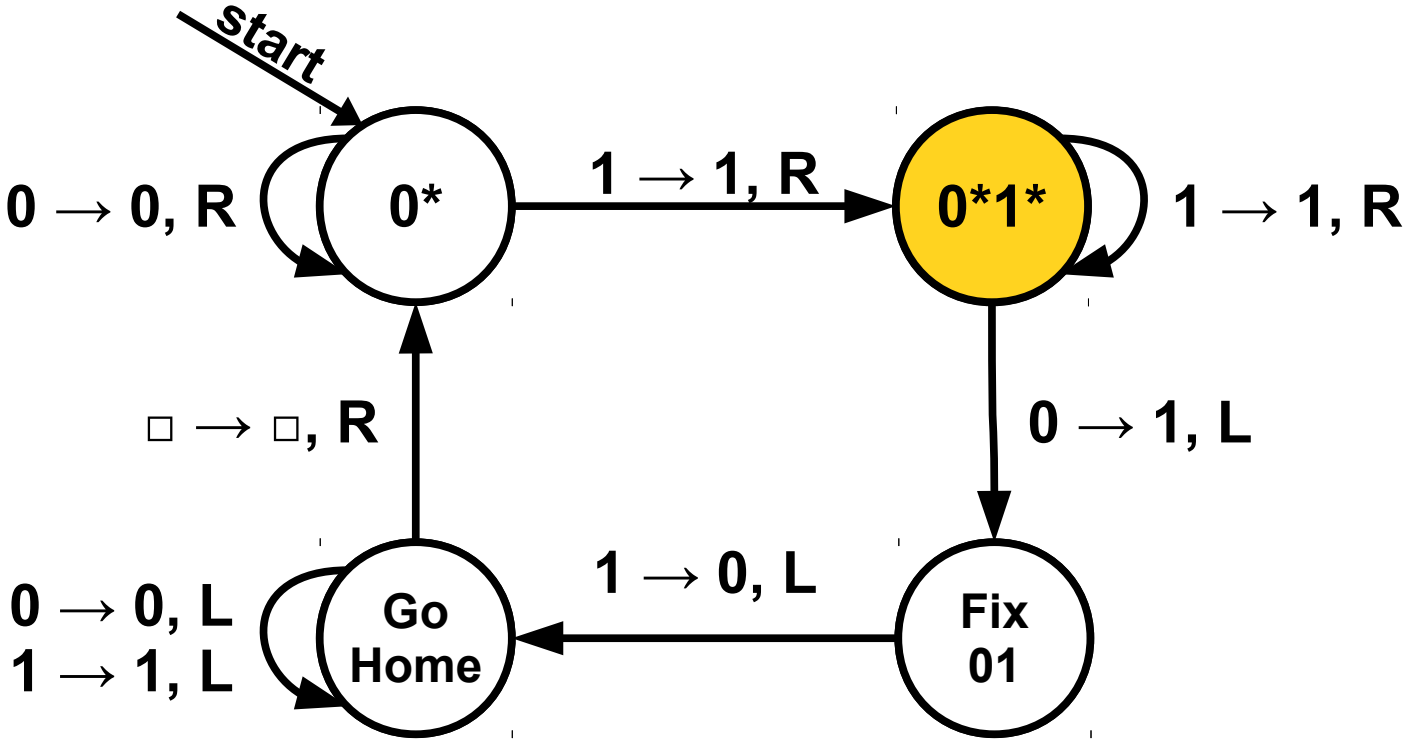

…
0 0 0 1 0 1
…

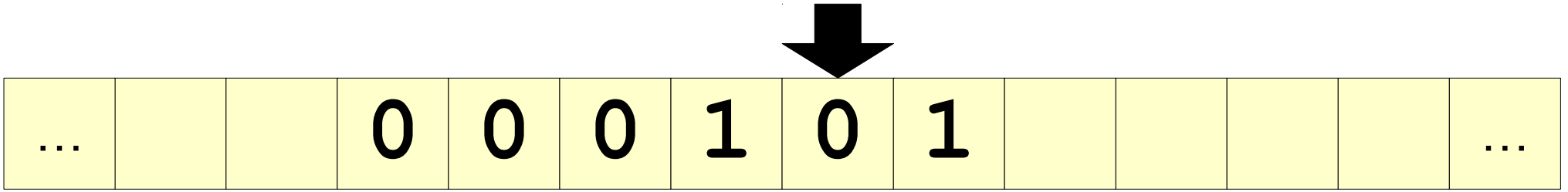

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

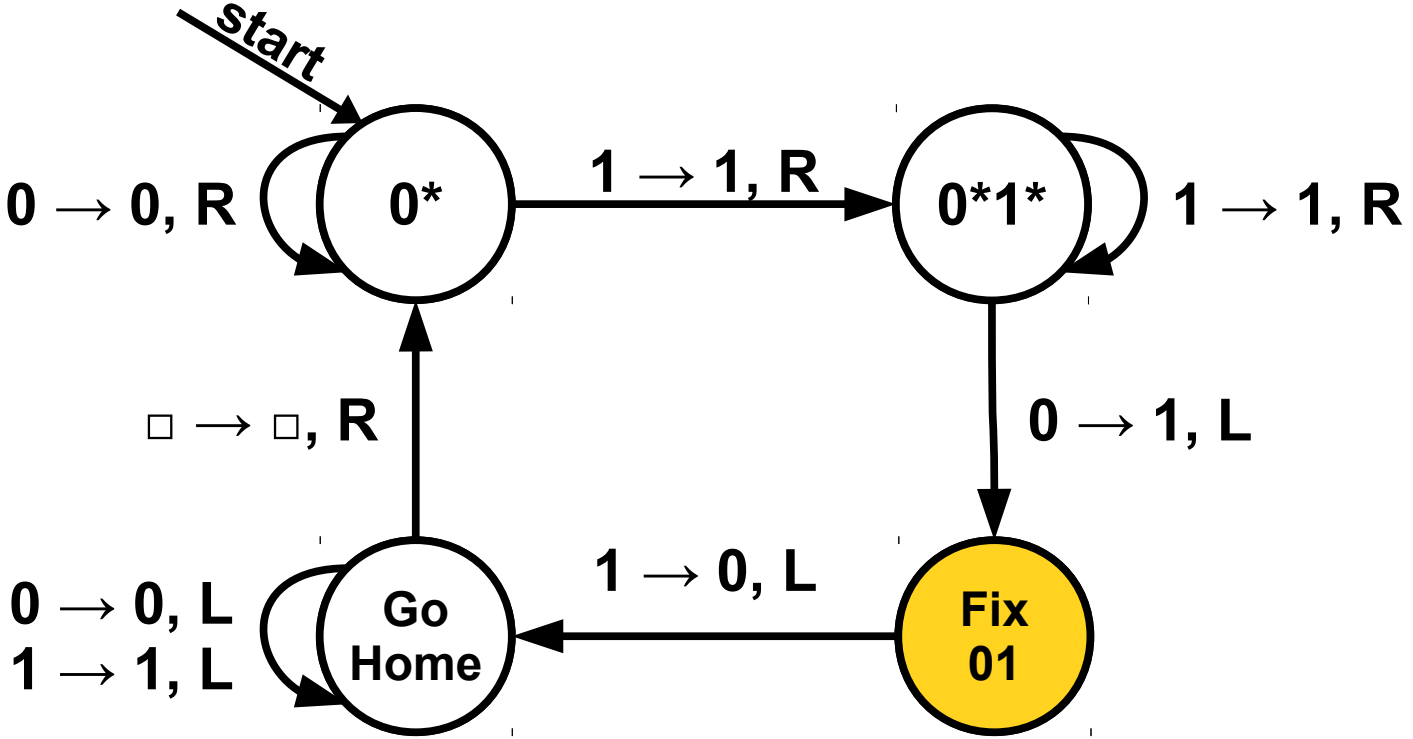

…
0 0 0 1 0 1
…

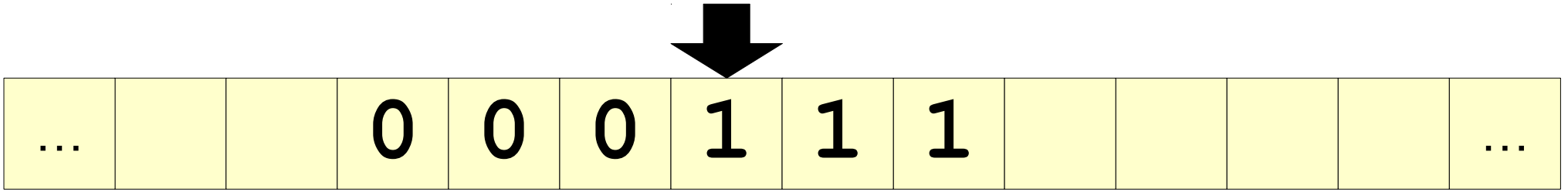

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

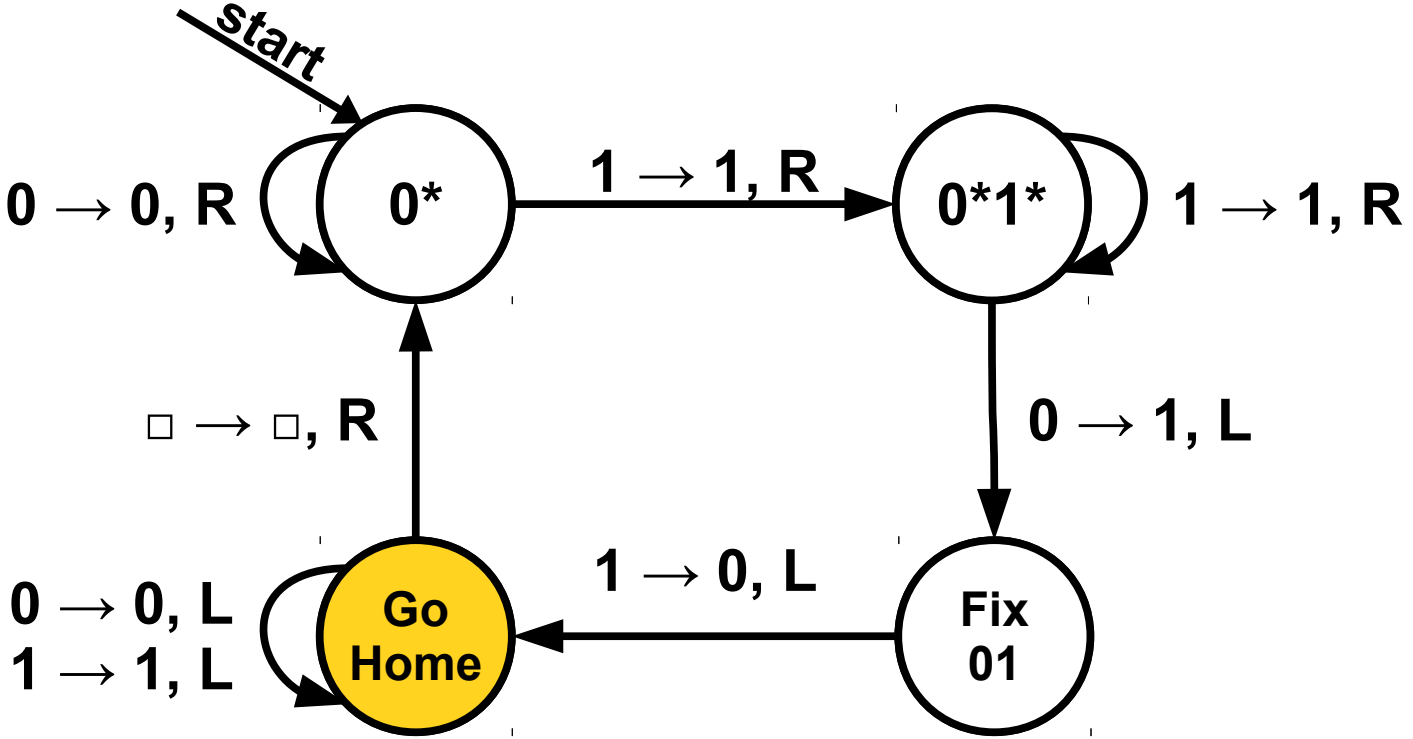

…
0 0 0 1 0 1
…

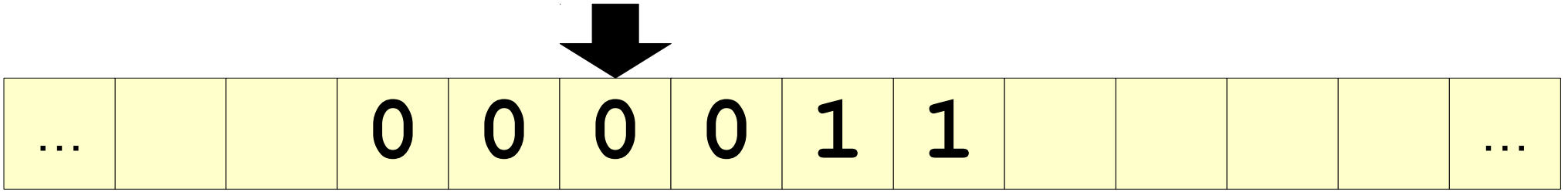

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

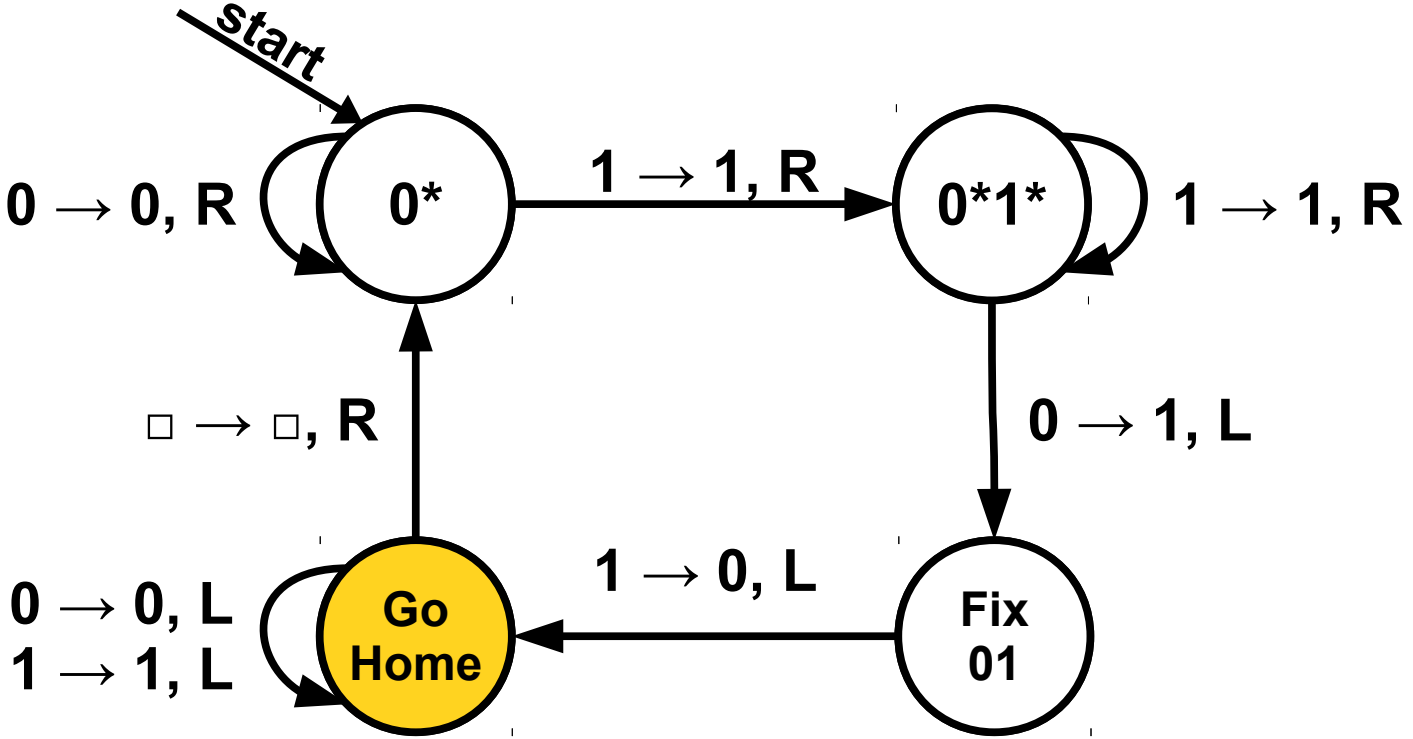

…
0 0 0 1 0 1
…

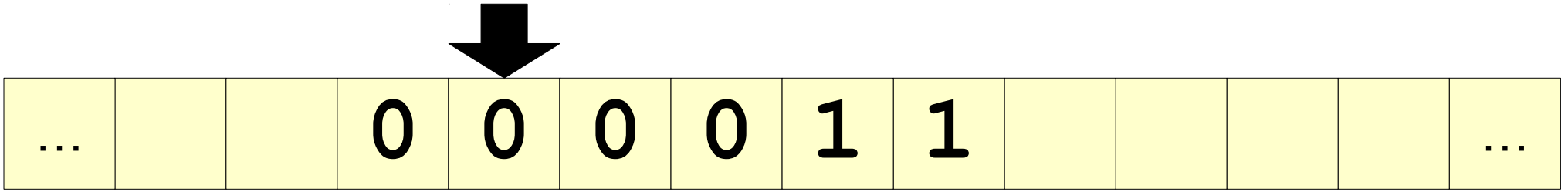

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

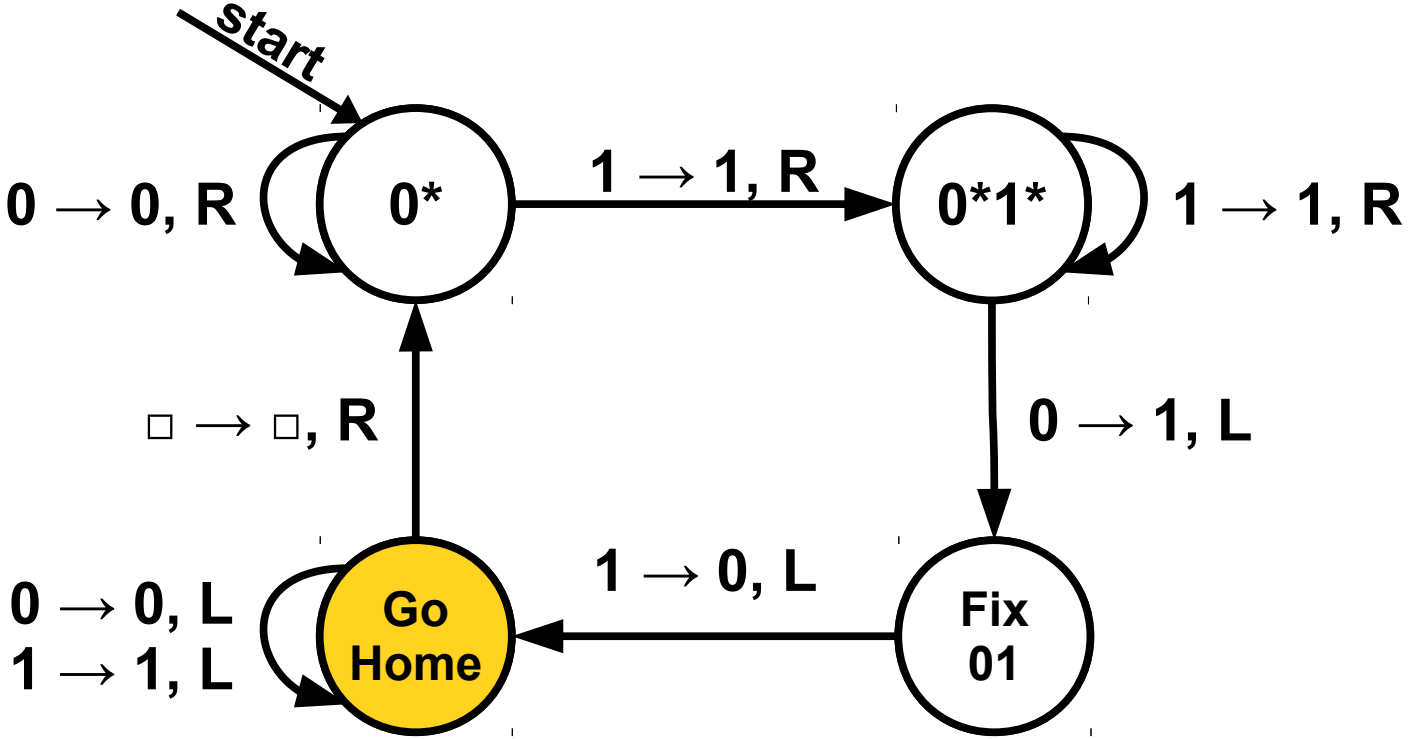

…
0 0 0 1 1 1
…

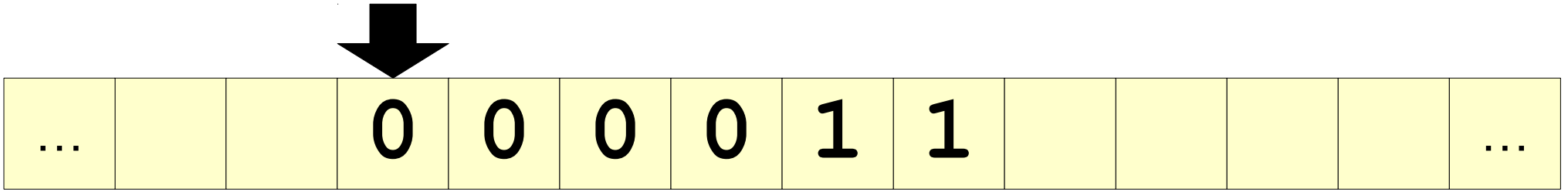

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

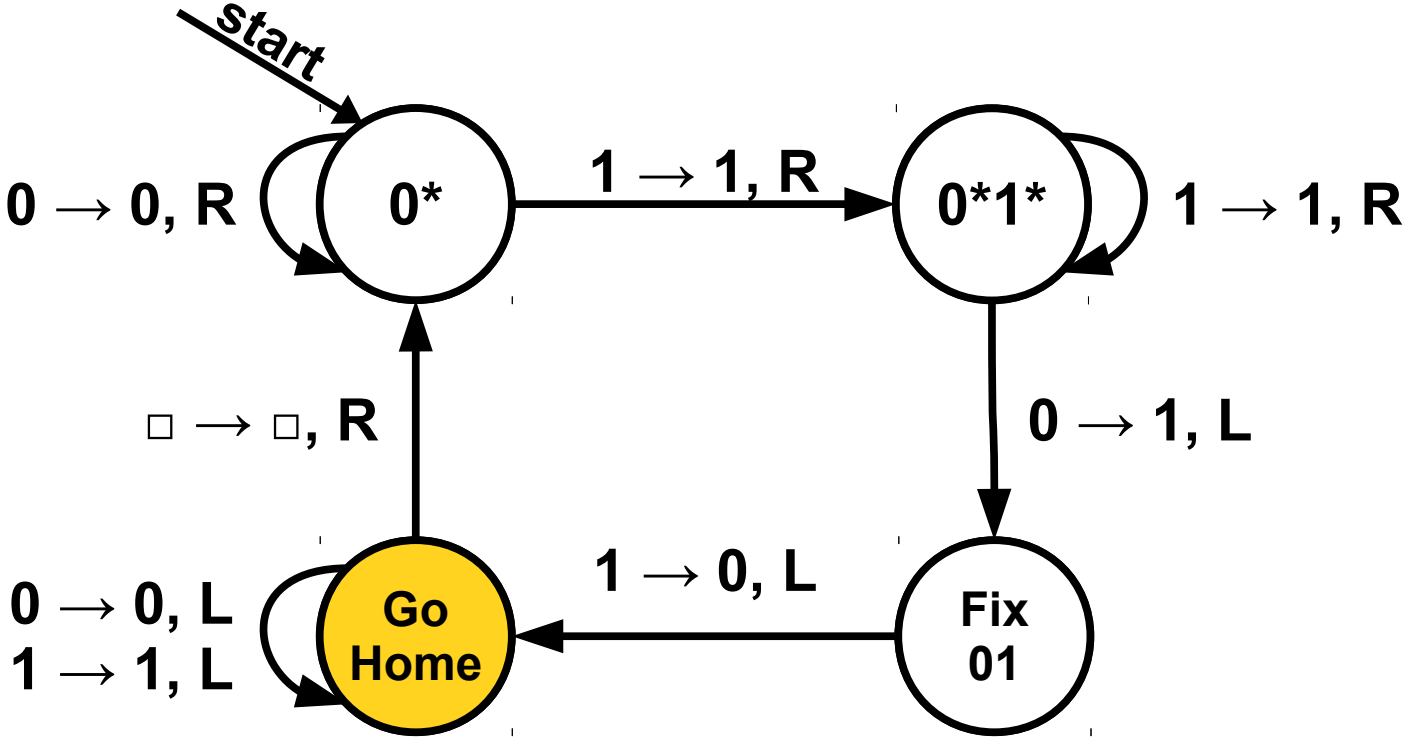

…
0 0 0 0 1 1
…

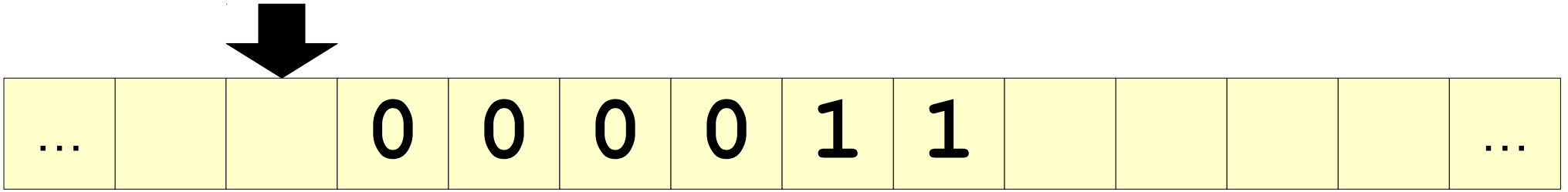

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

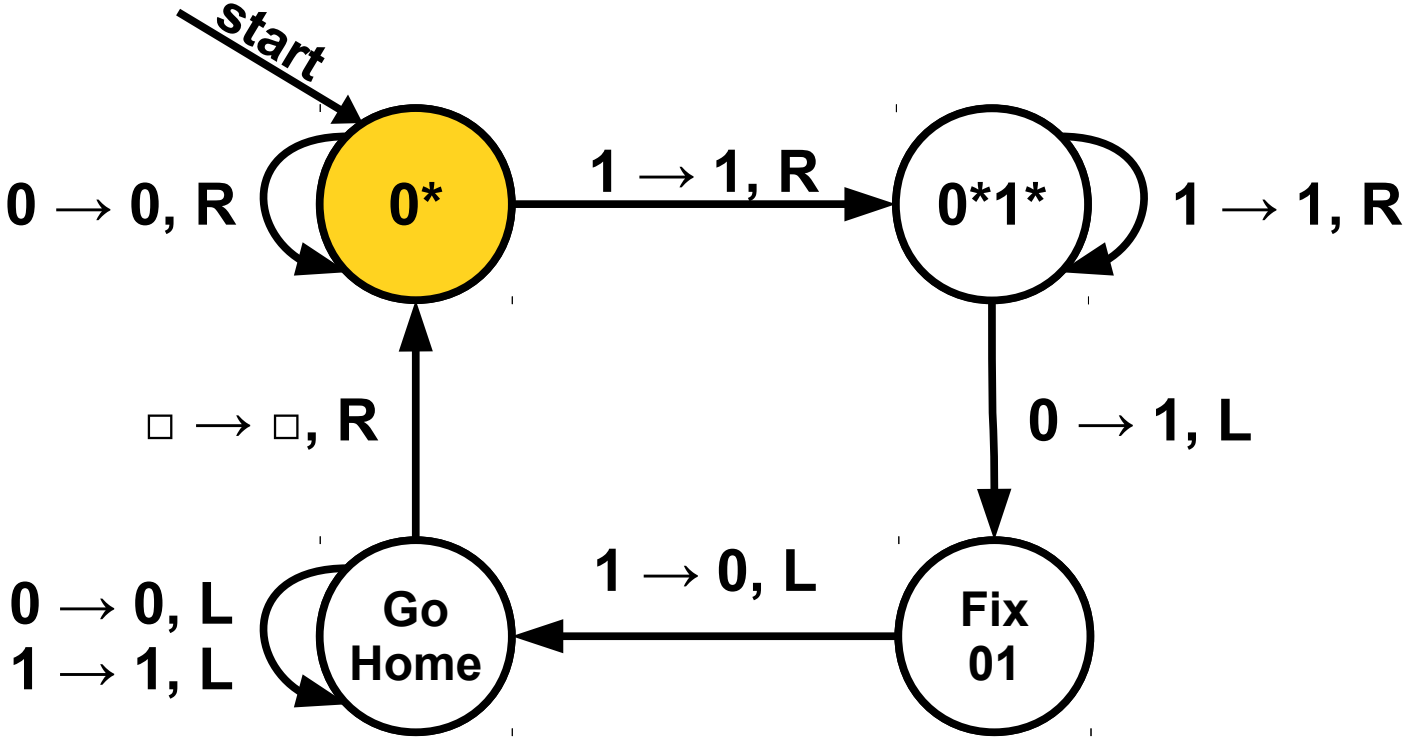

…
0 0 0 0 1 1
…

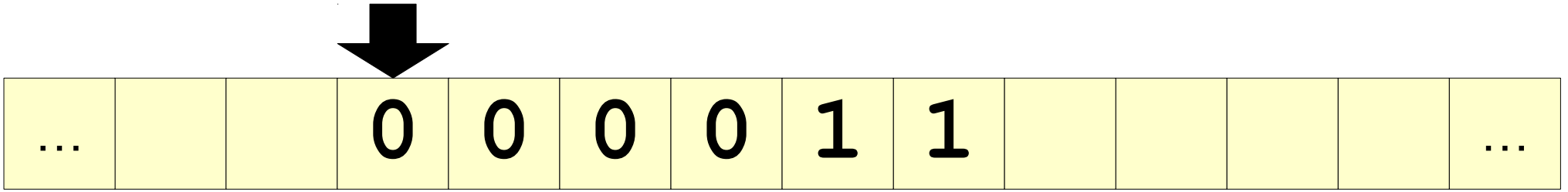

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

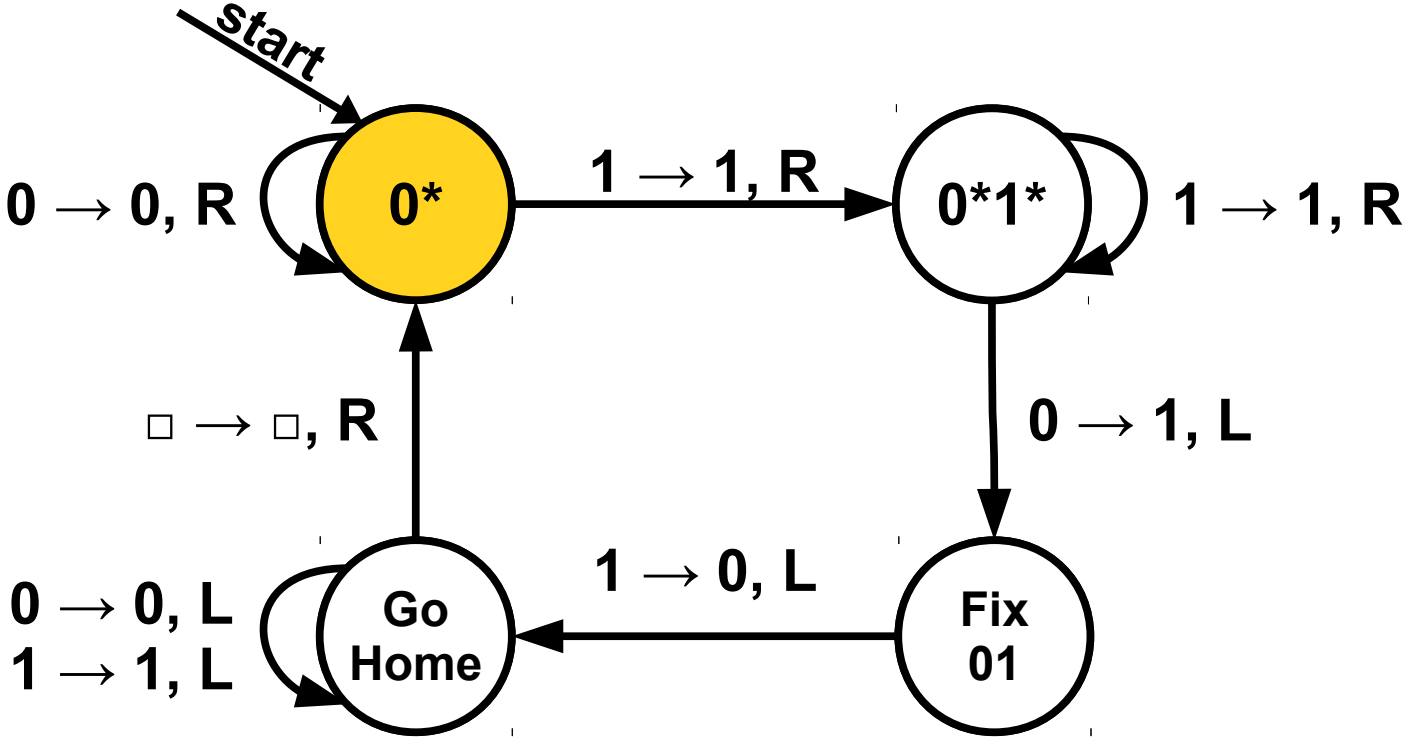

…
0 0 0 0 1 1
…

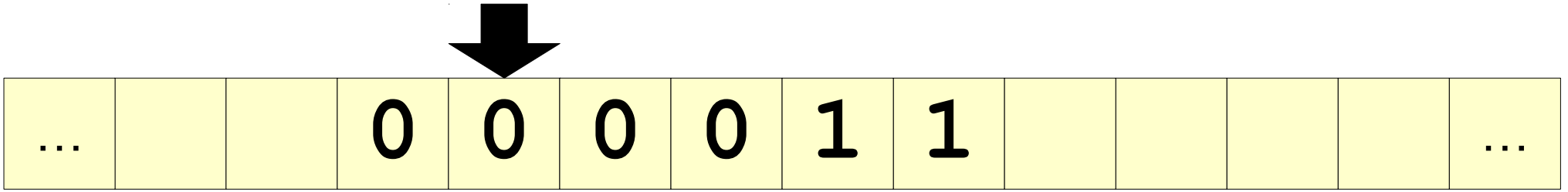

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

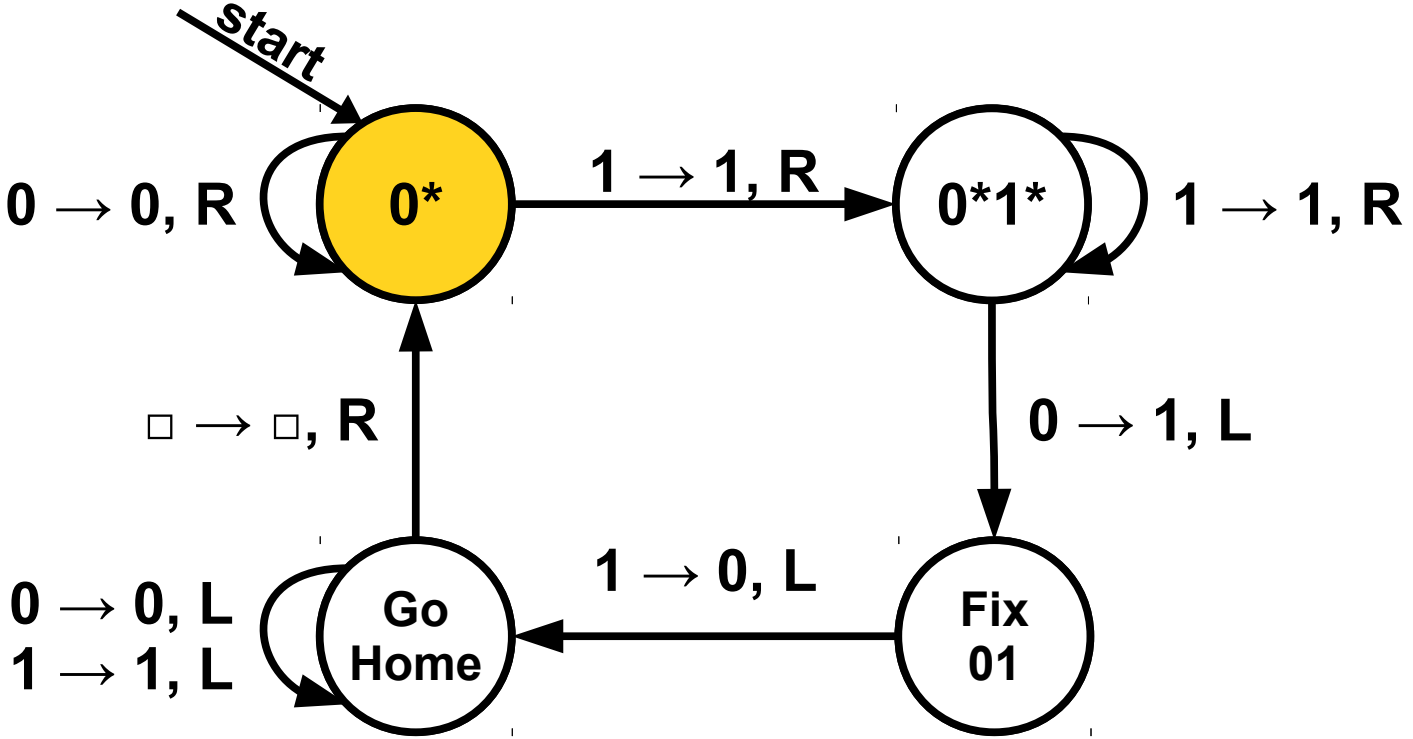

…
0 0 0 0 1 1
…

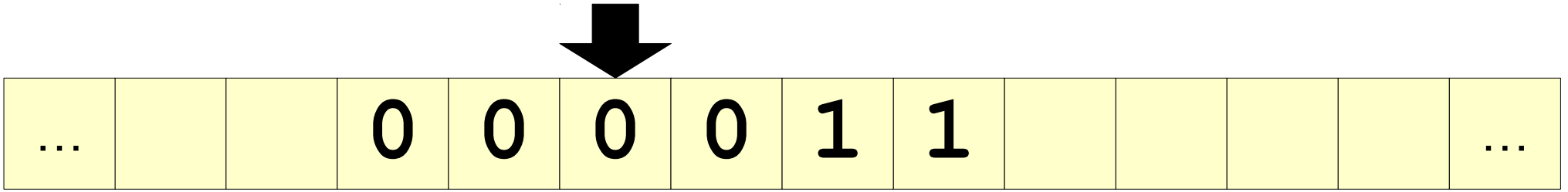

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

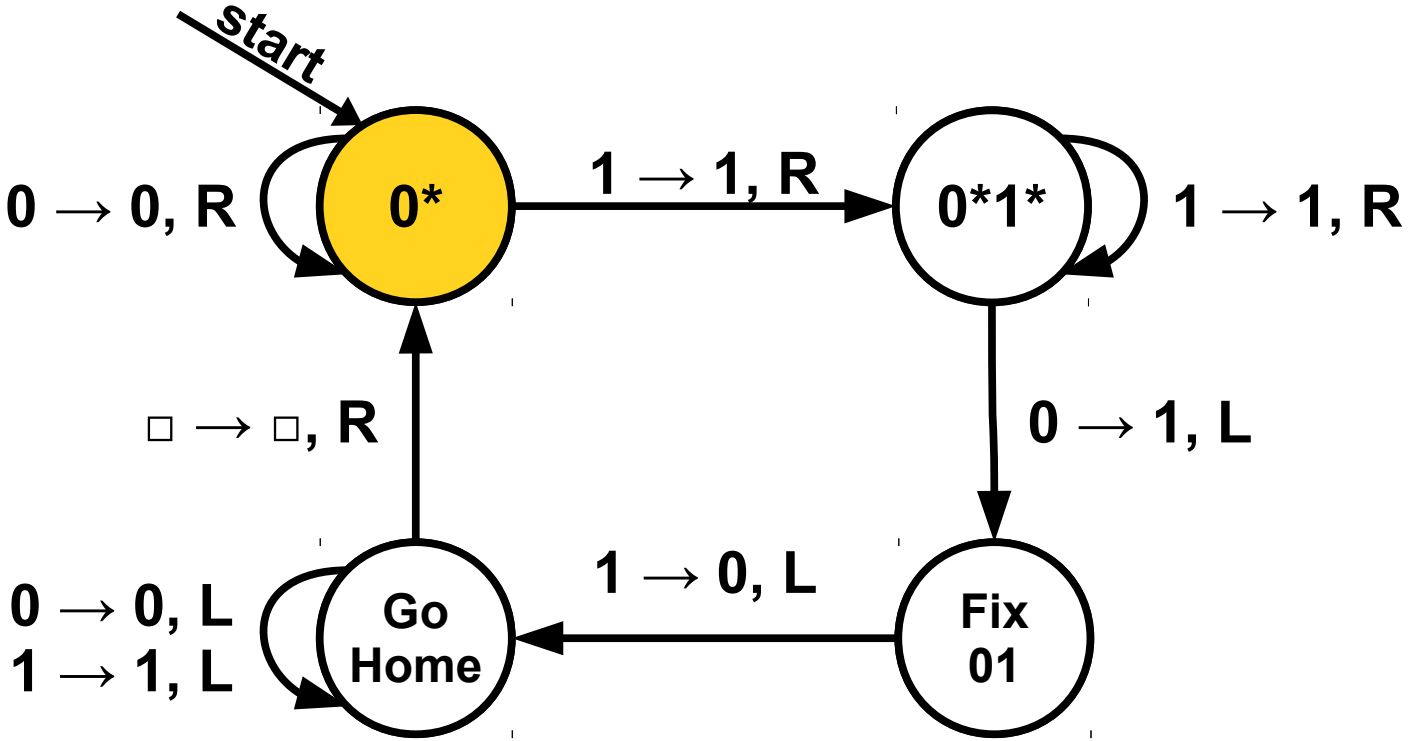

…
0 0 0 0 1 1
…

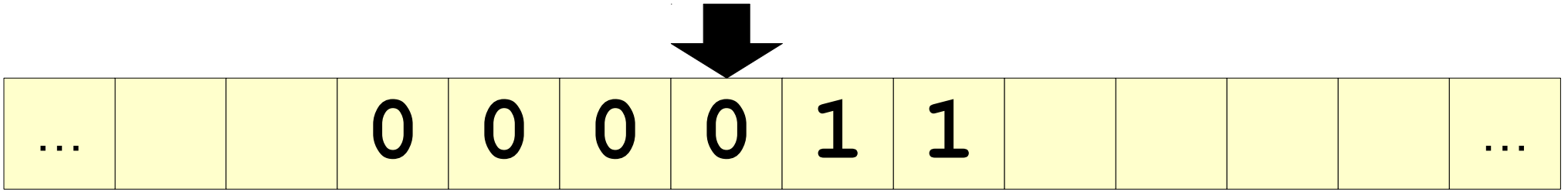

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

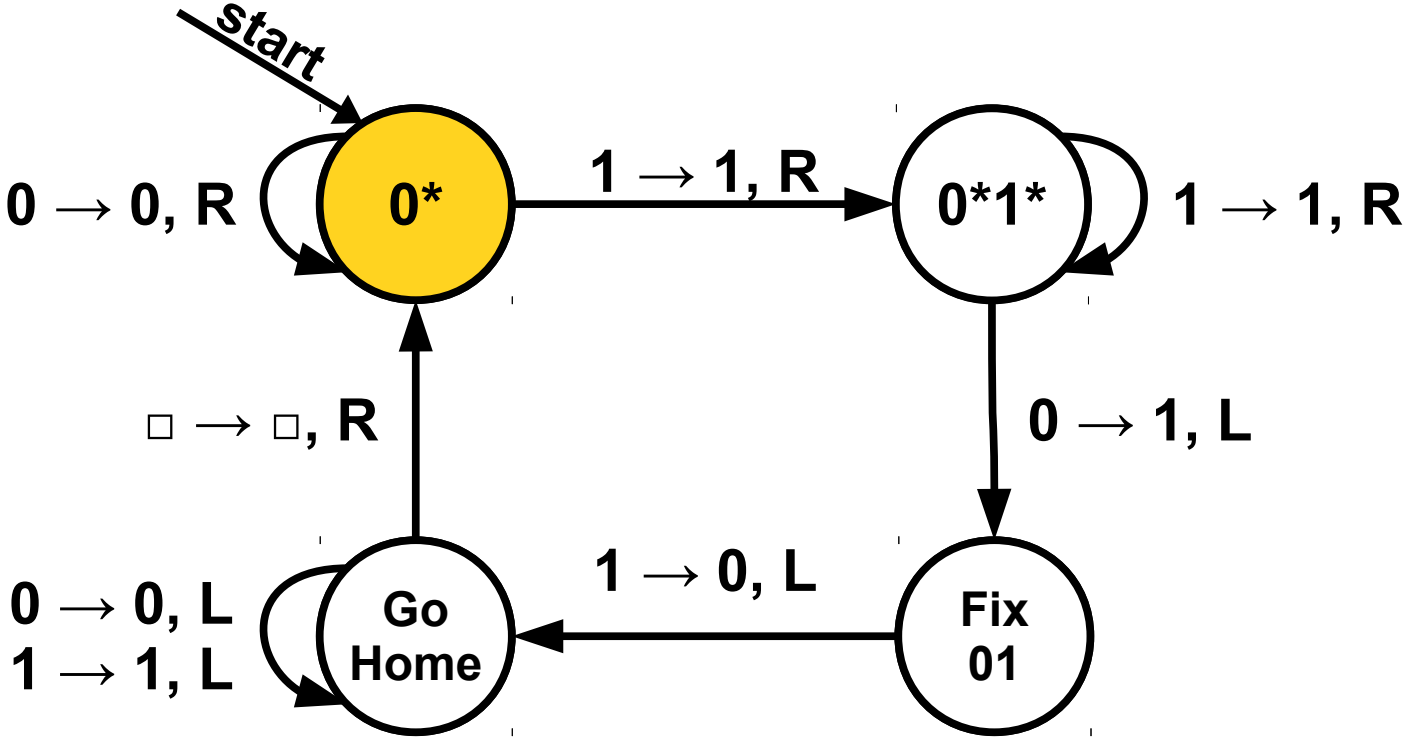

…
0 0 0 0 1 1
…

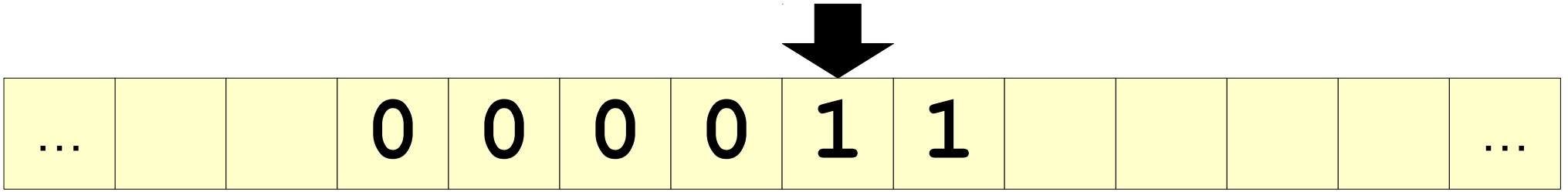

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

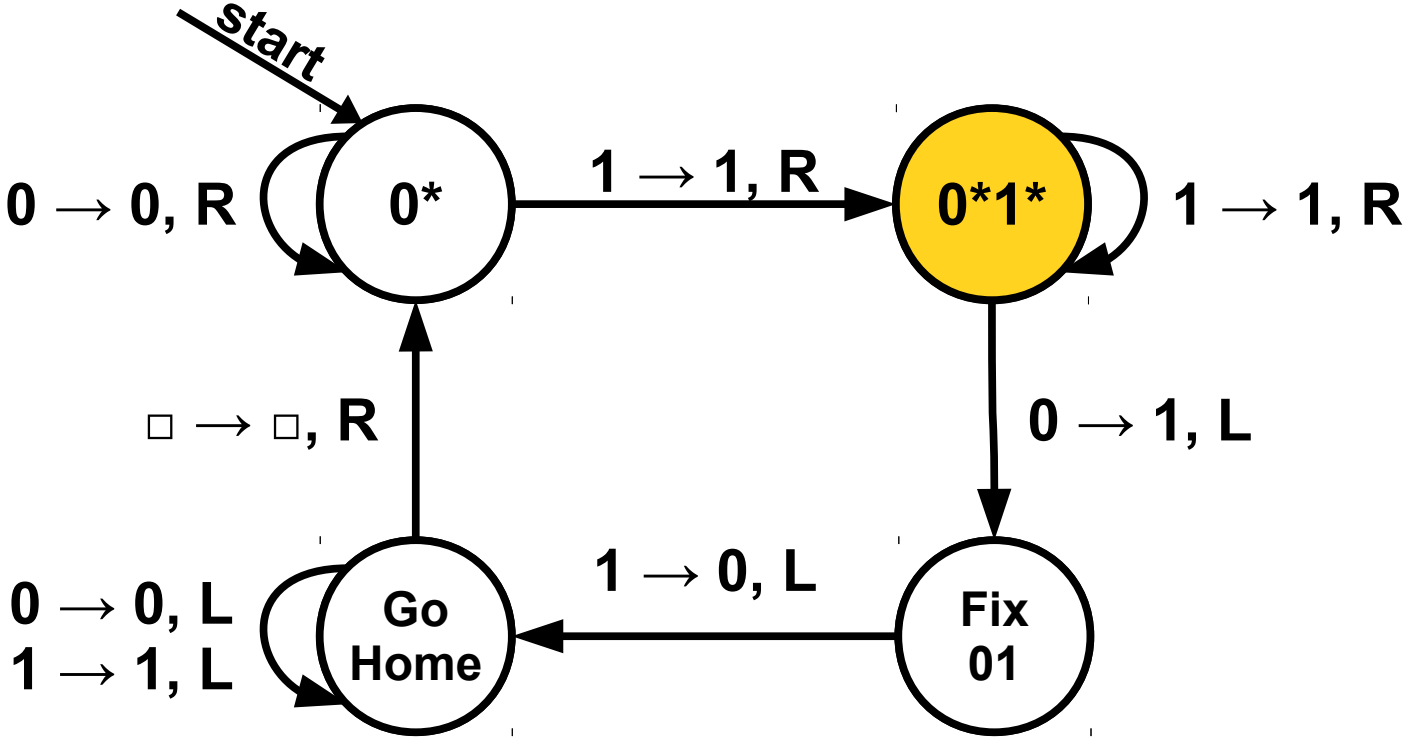

…
0 0 0 0 1 1
…

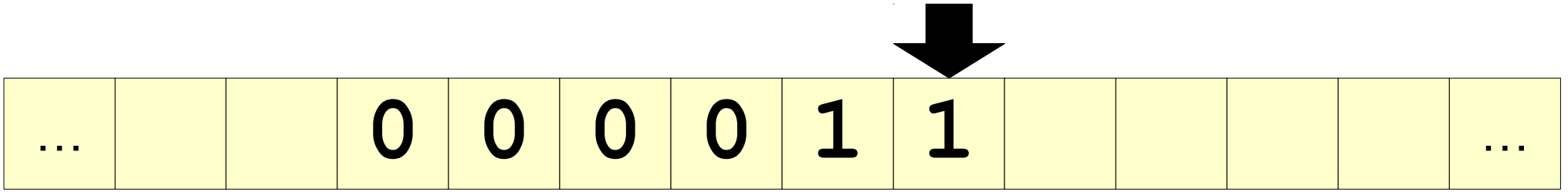

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

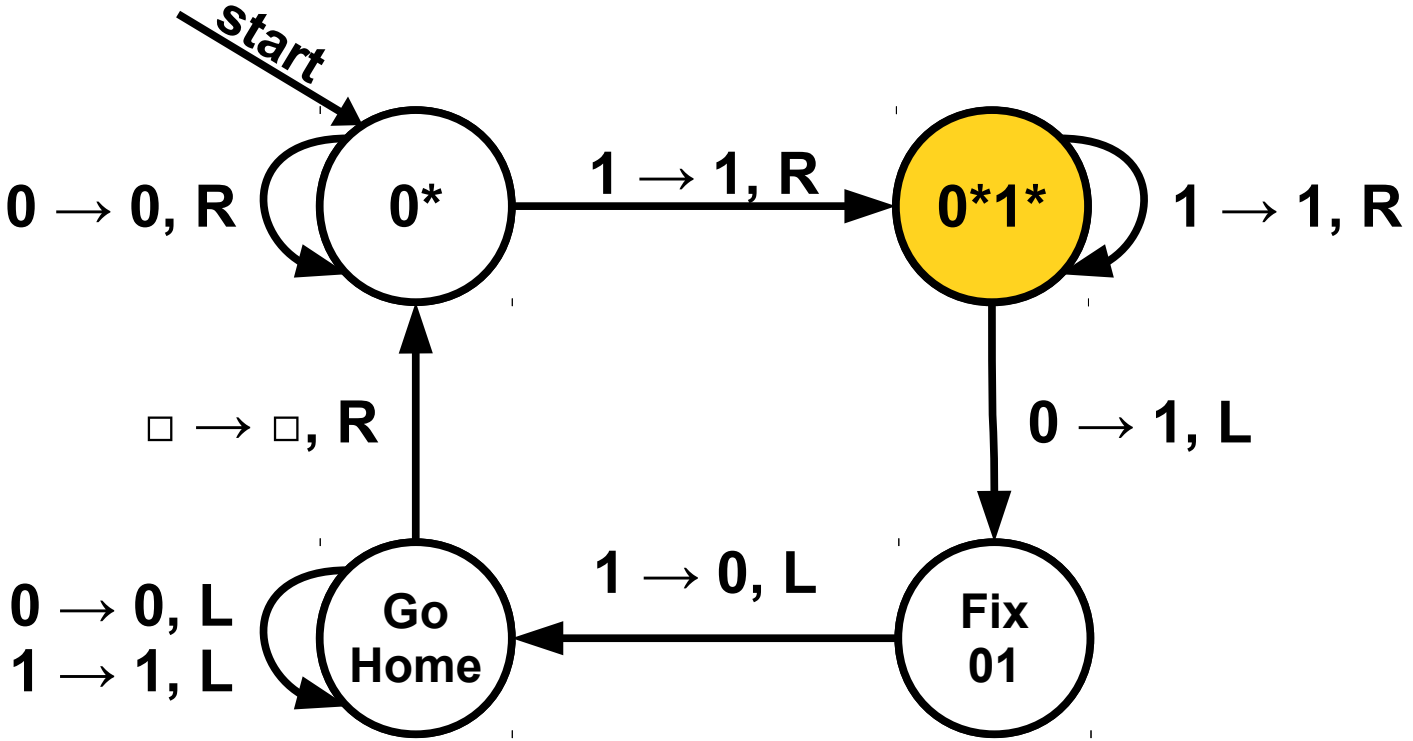

…
0 0 0 0 1 1
…

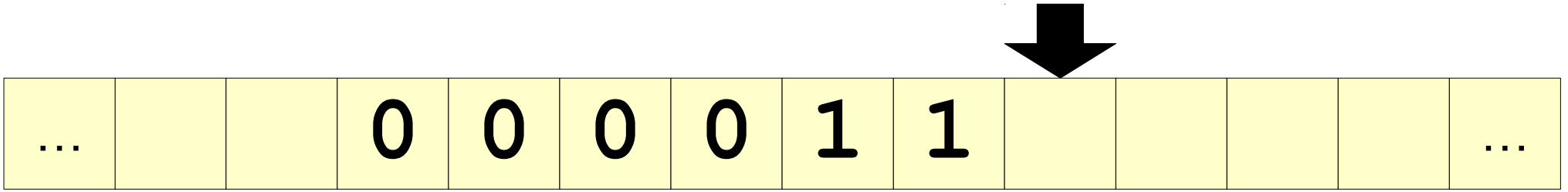

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L


…
0 0 0 0 1 1
…

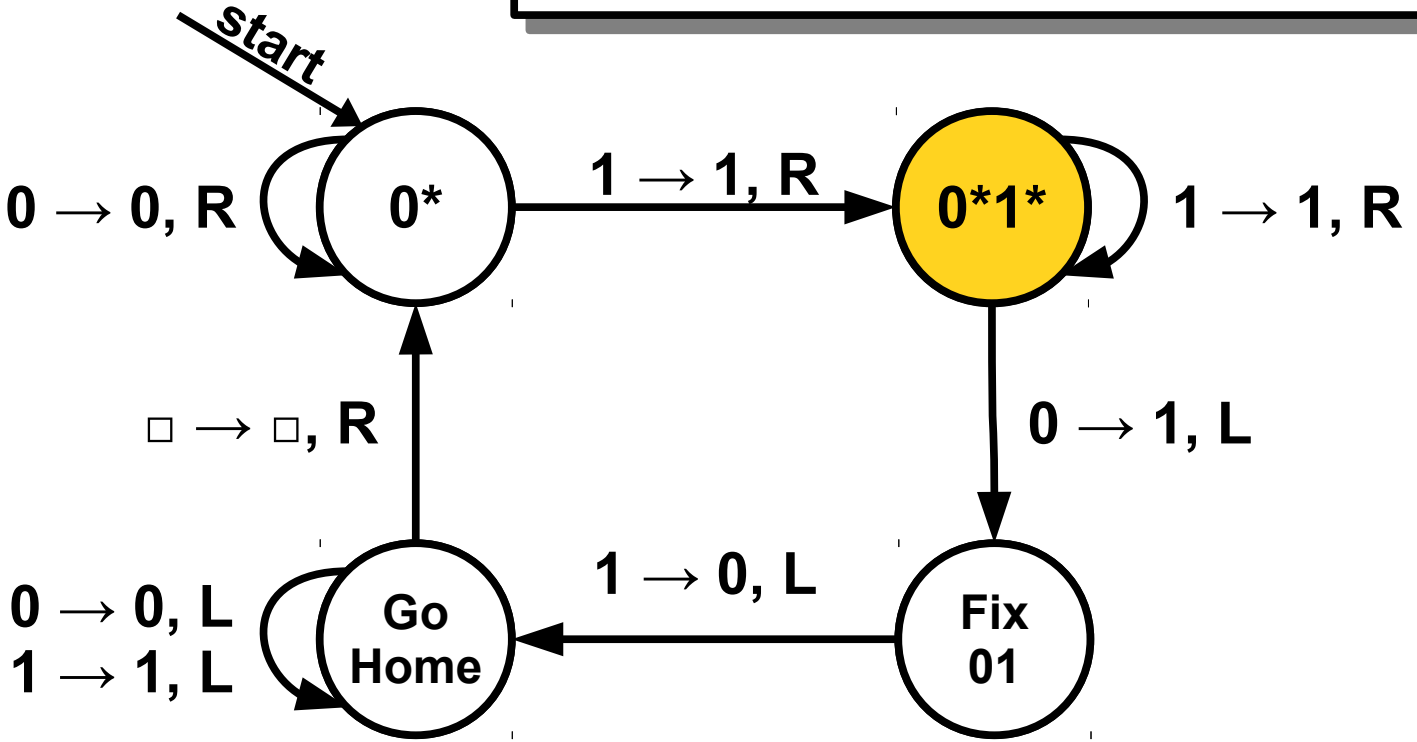

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

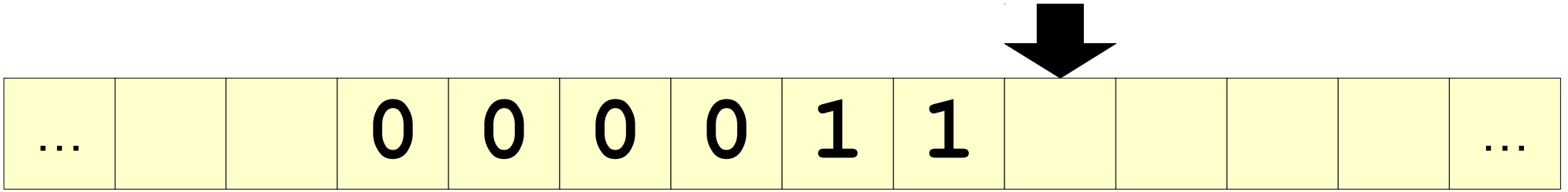

…
0 0 0 0 1 1
…

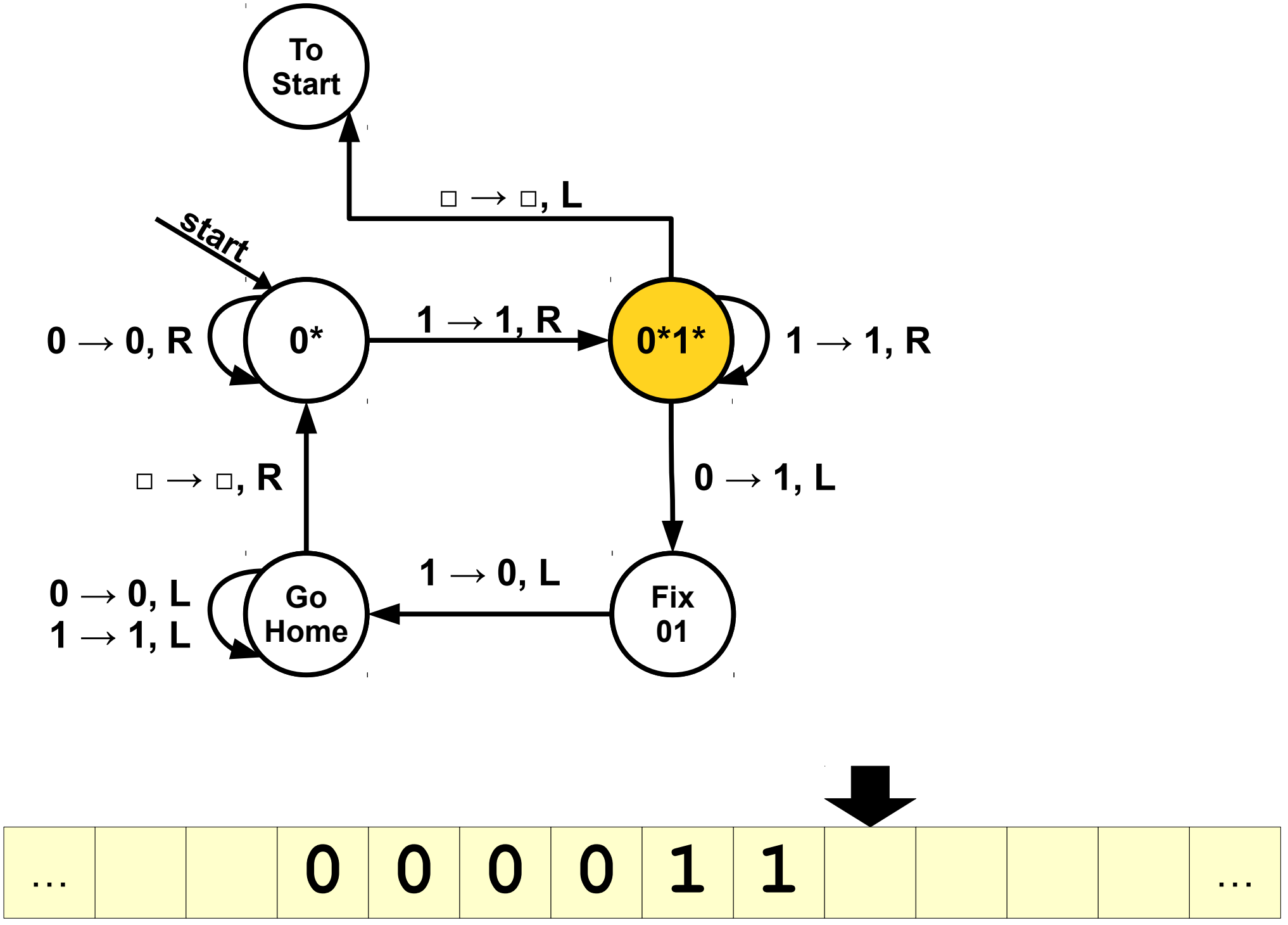

start
0*
0*1*
Go Home
Fix 01
0 → 0, R
1 → 1, R
1 → 1, R
□ → □, R
0 → 1, L
1 → 0, L
0 → 0, L
1 → 1, L

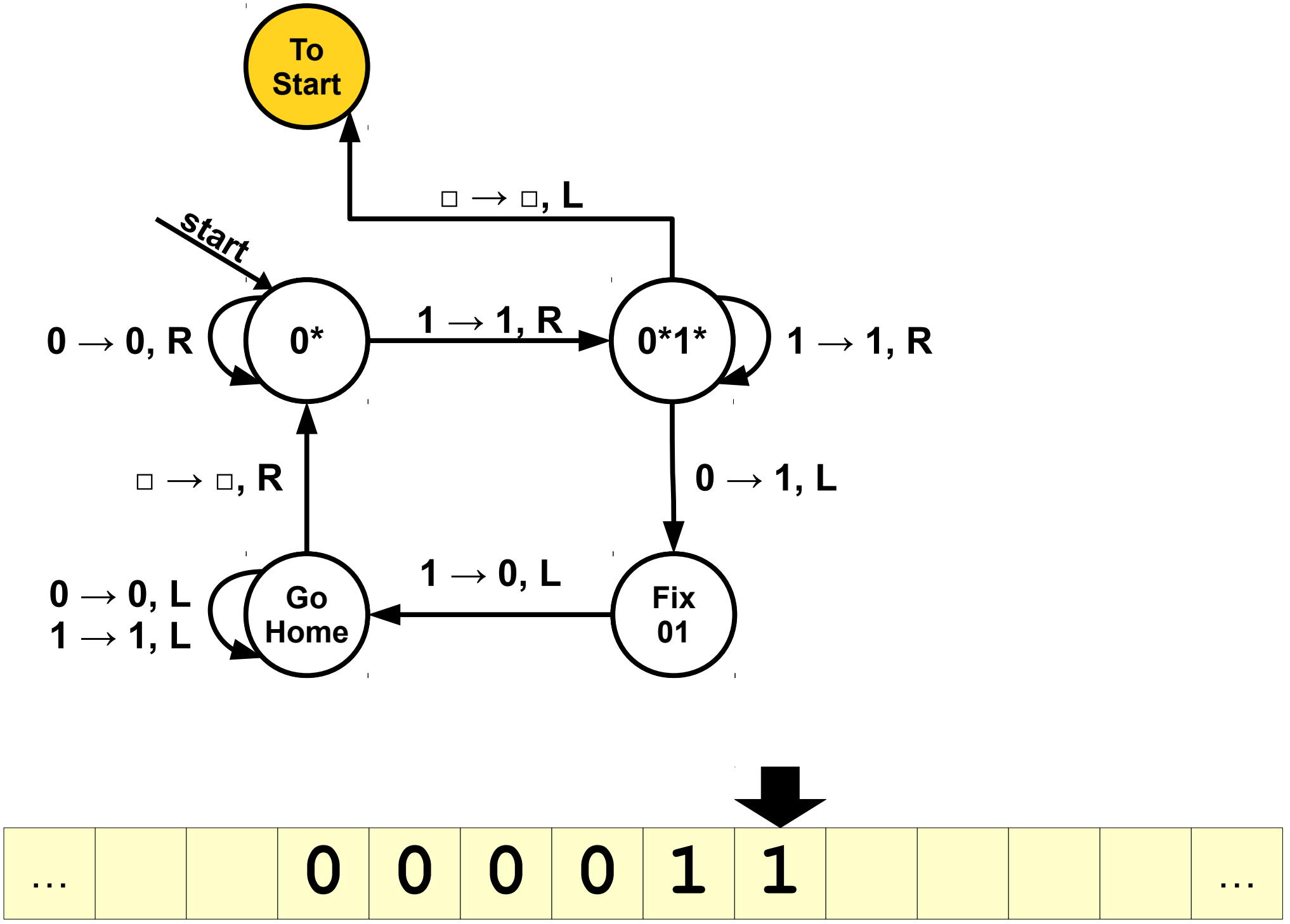

…
0 0 0 0 1 1
…

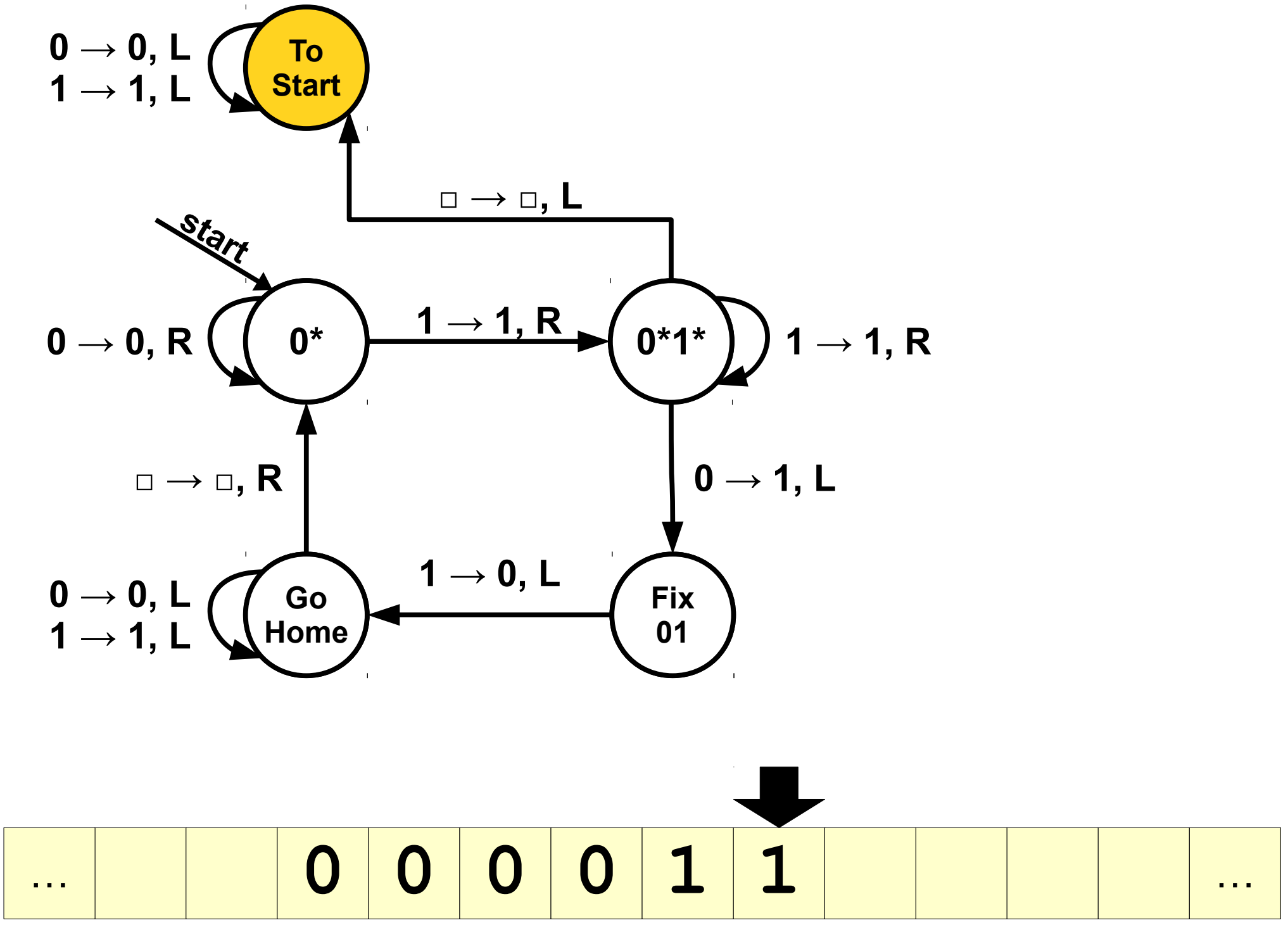
Our ultimate goal here was to sort everything so we could hand it off to the machine to check for 0^n1^n. Let's rewind the tape head back to the start.

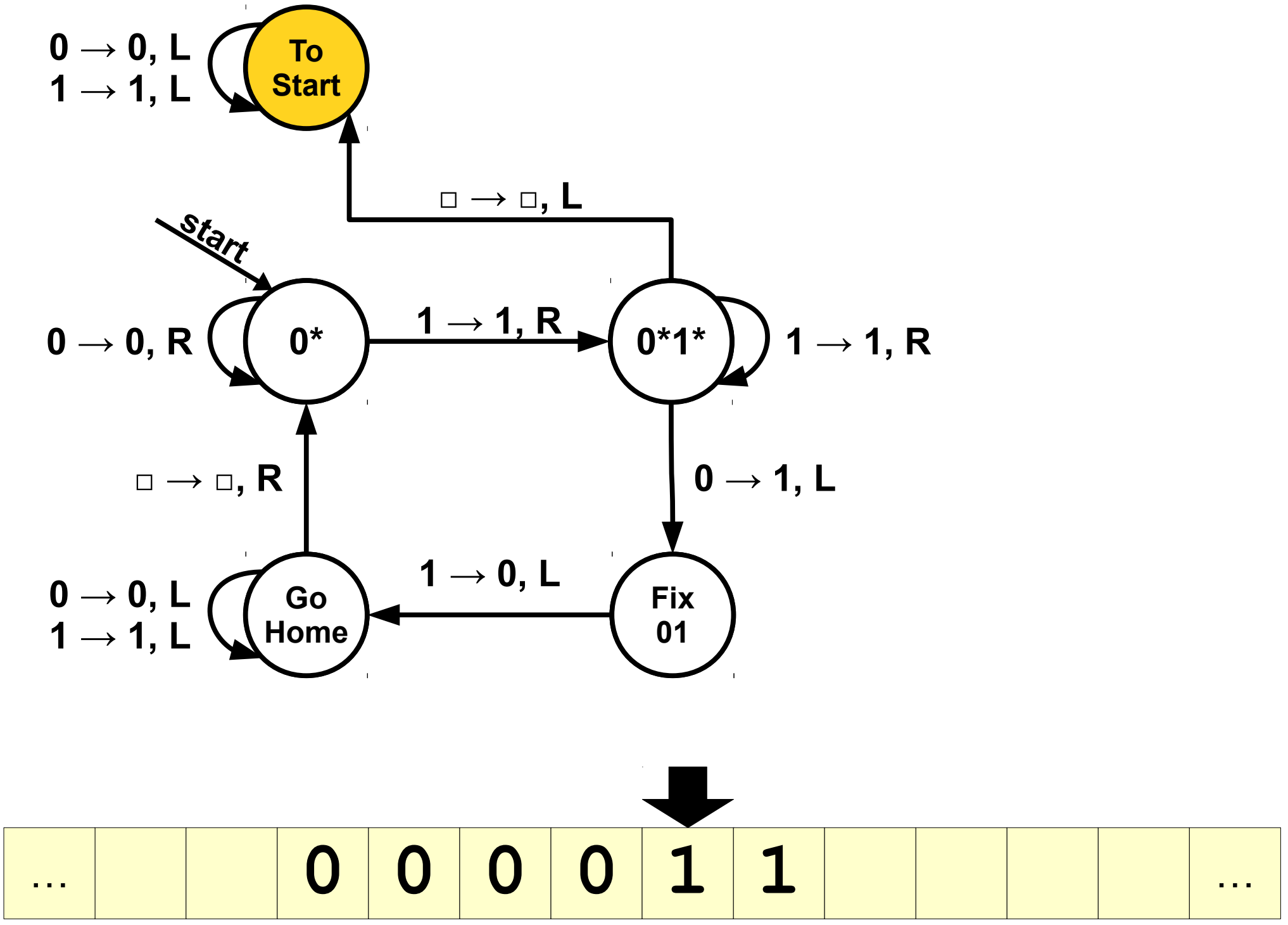


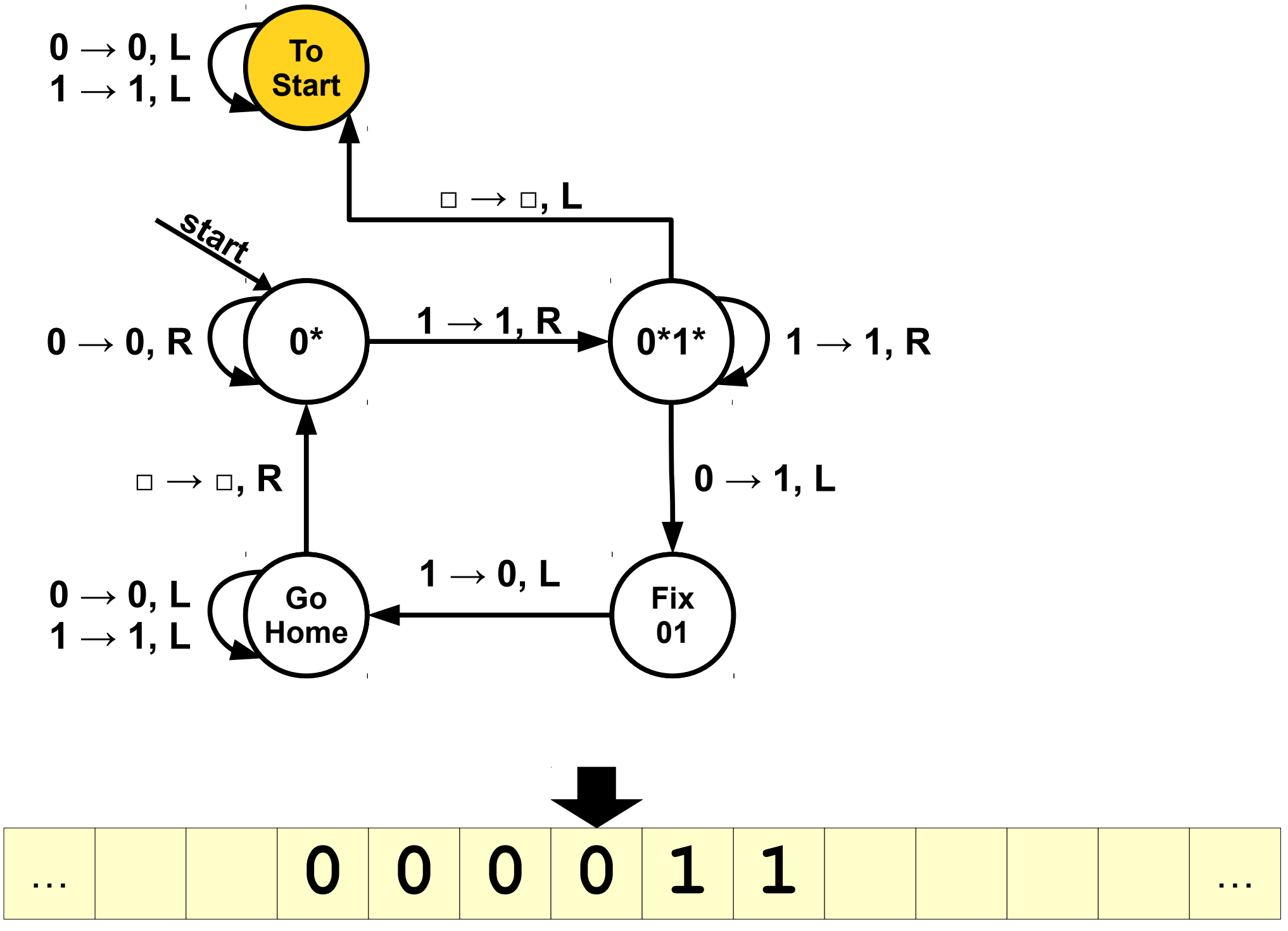


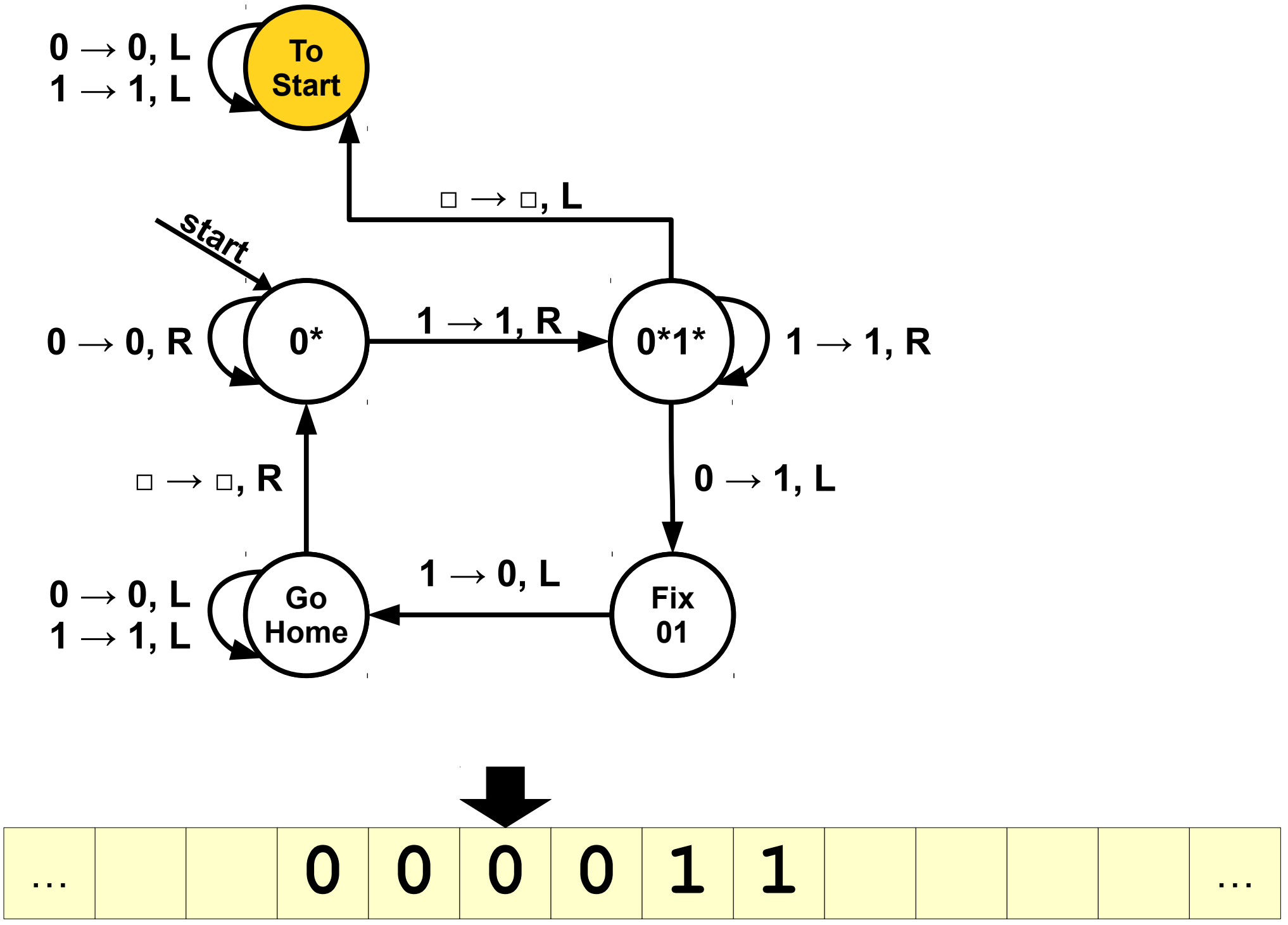

To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

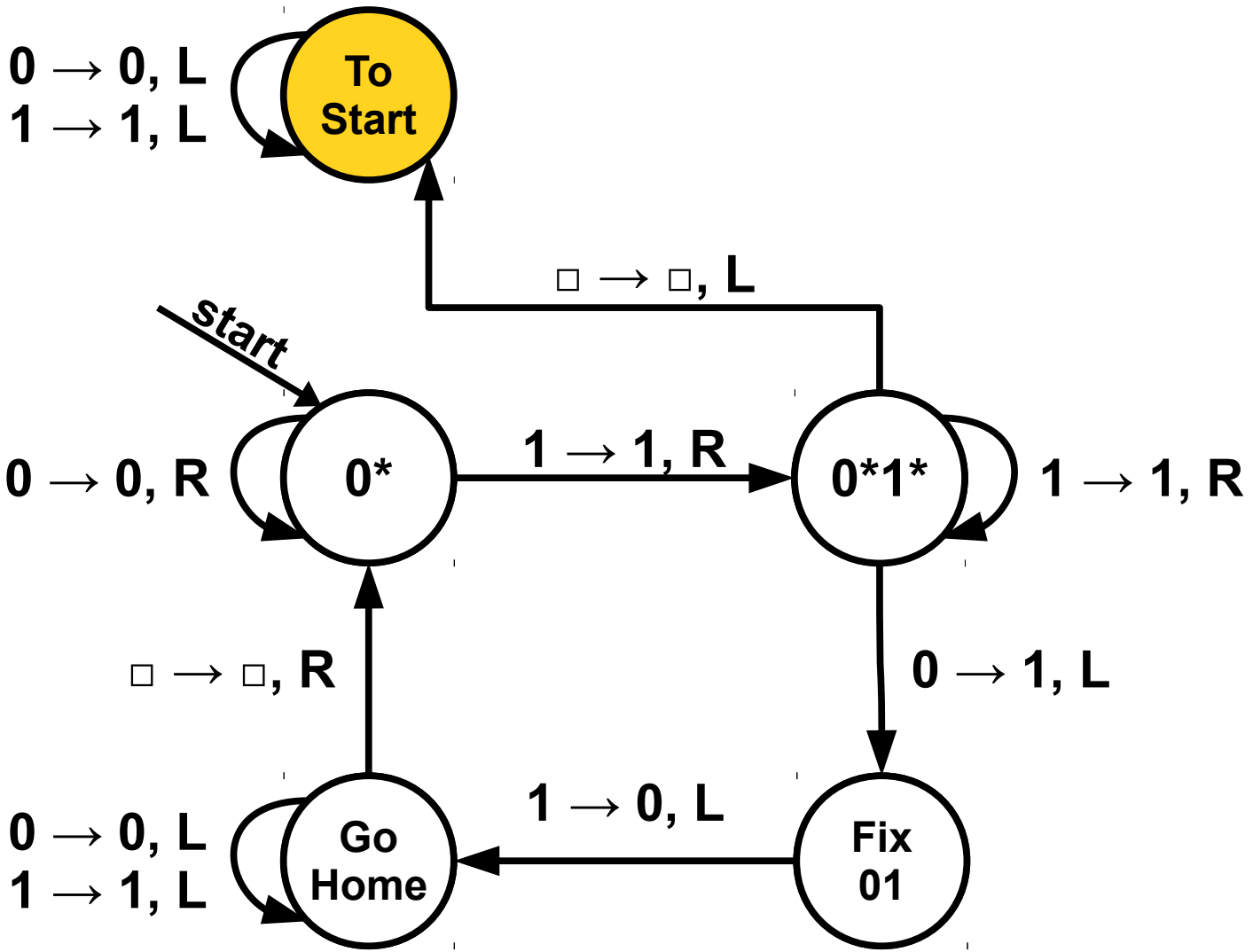

To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

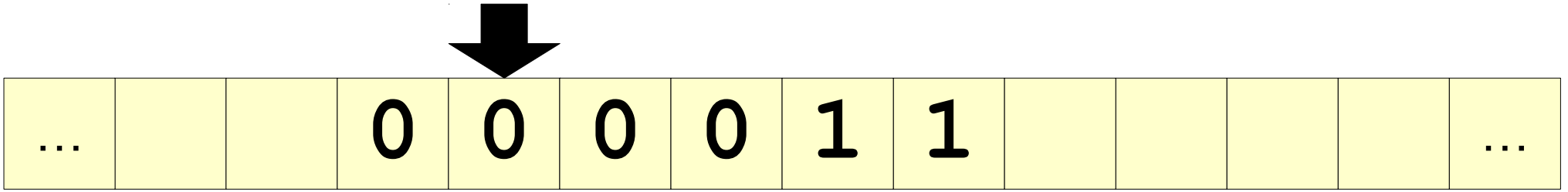

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

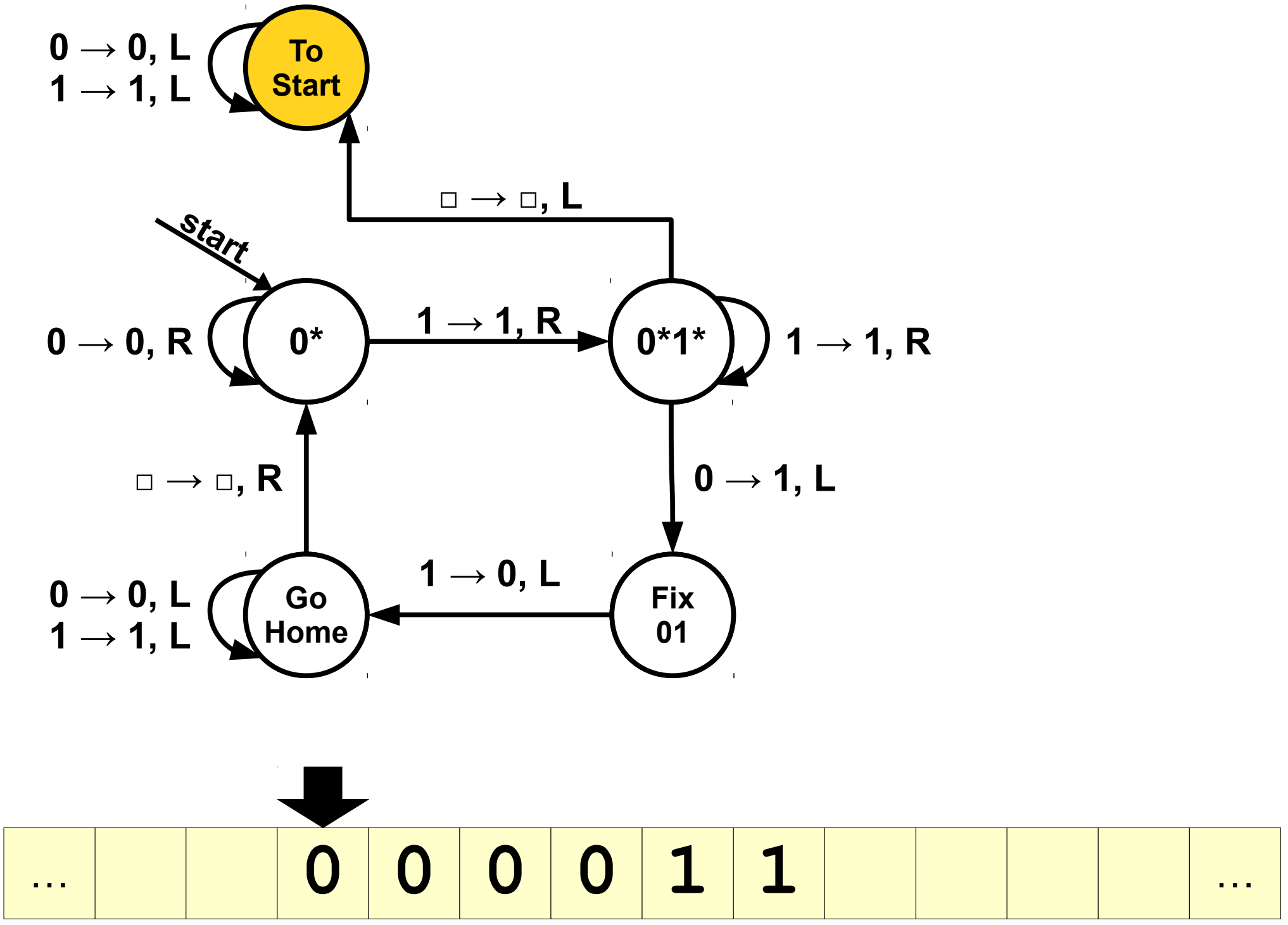

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

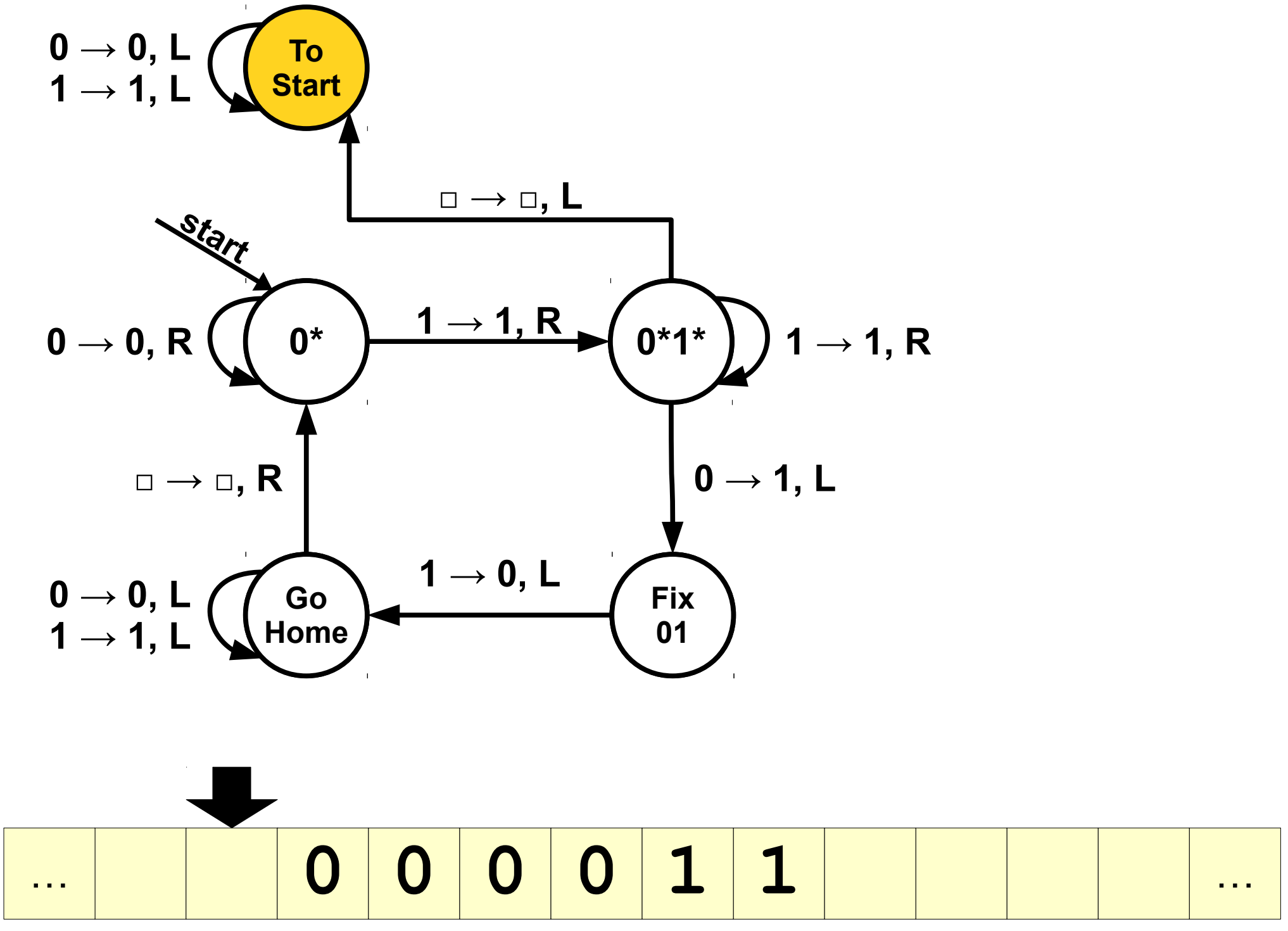

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

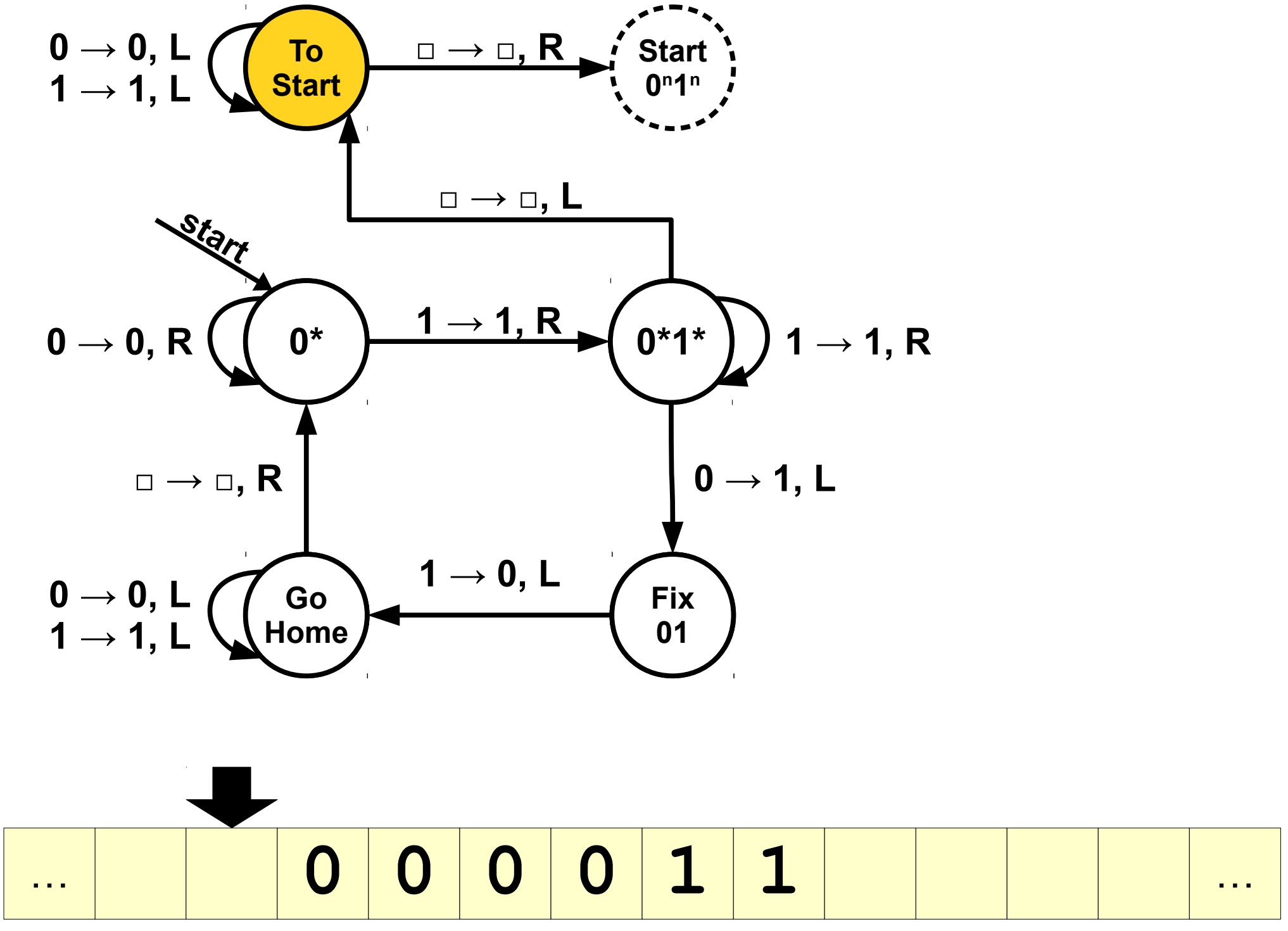

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

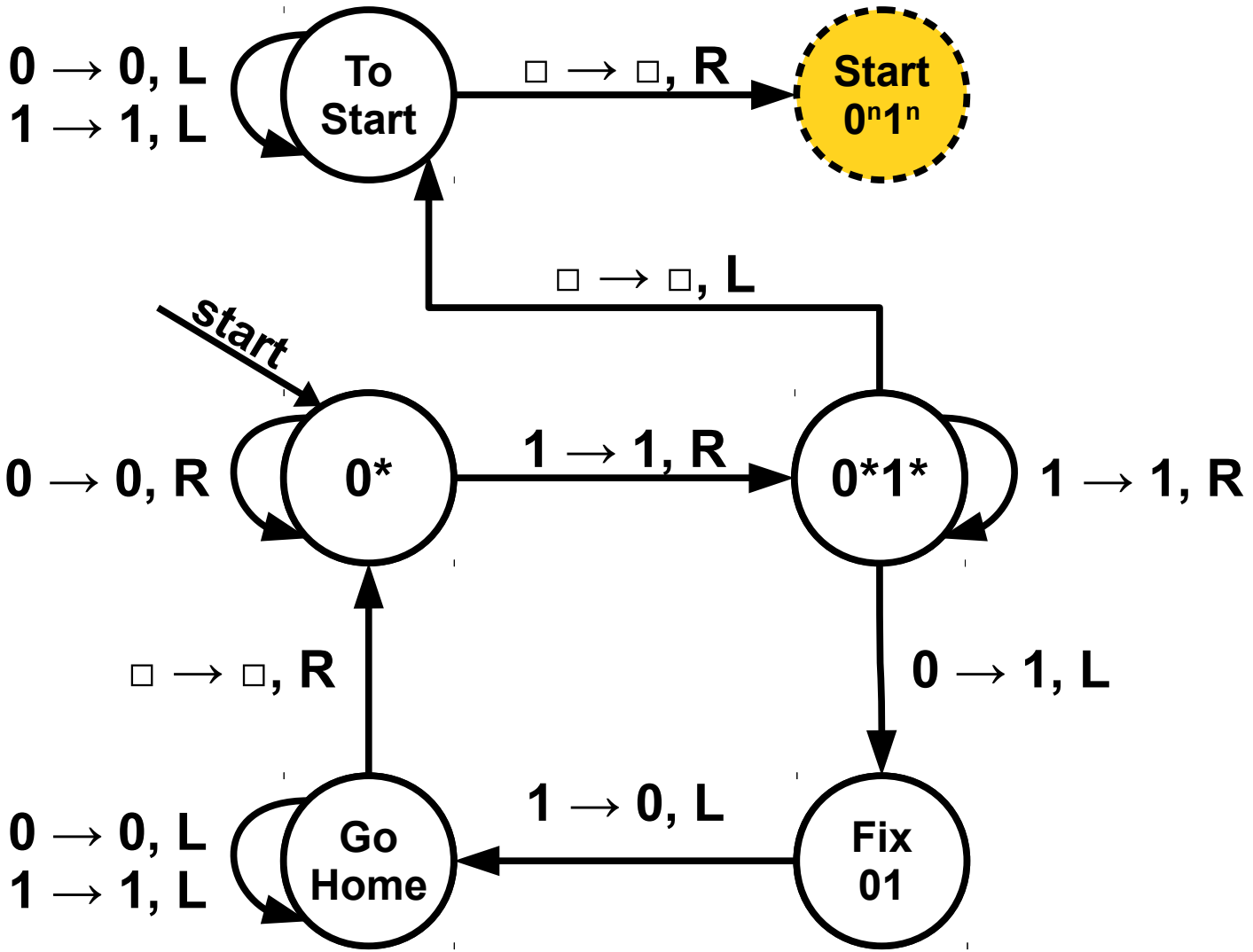

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

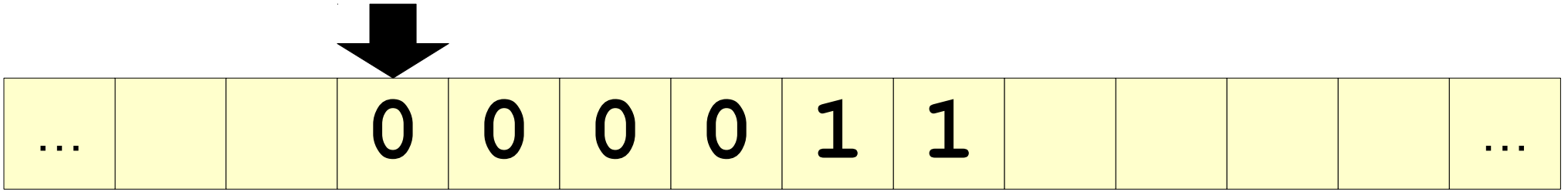

…
0 0 0 0 1 1
…

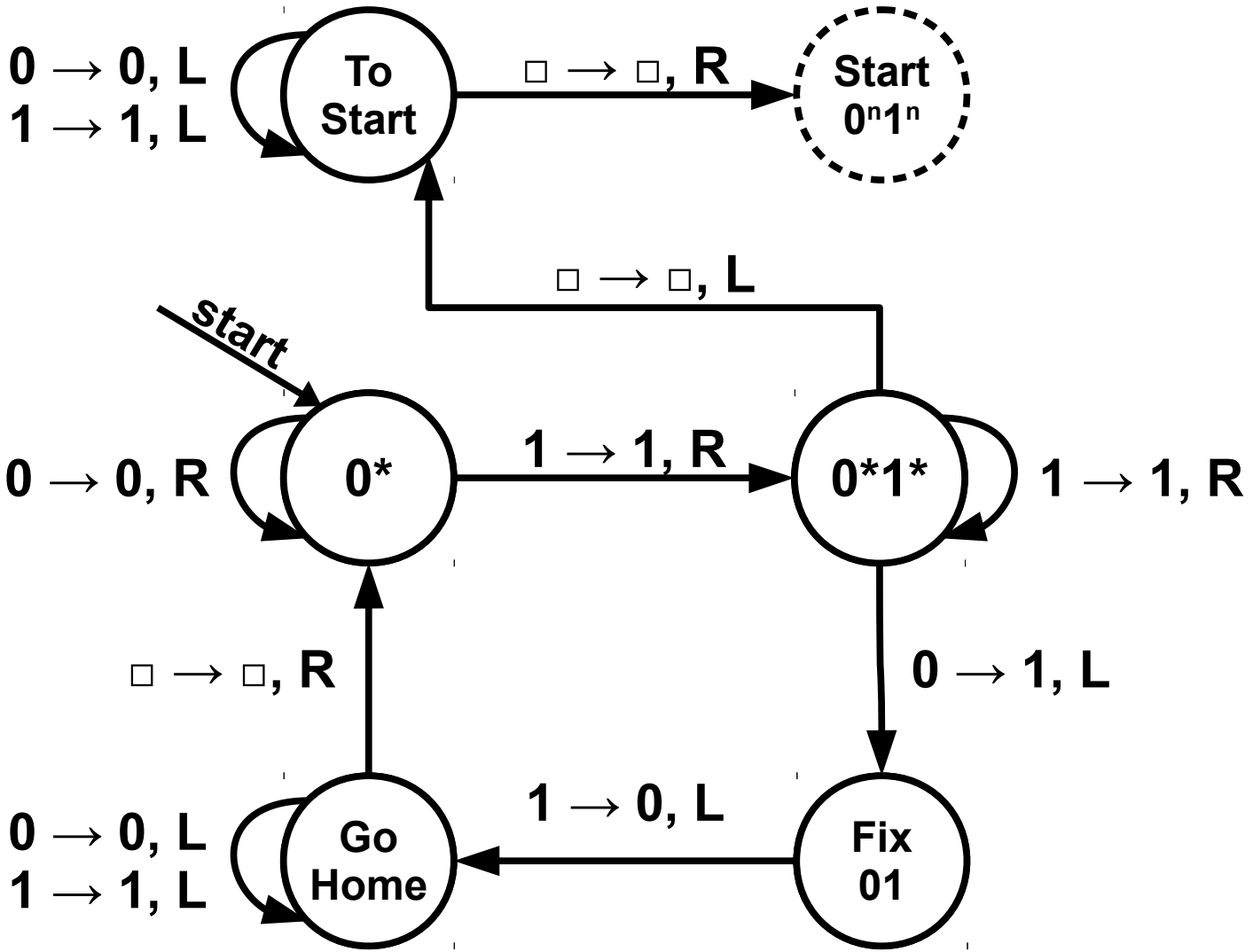

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

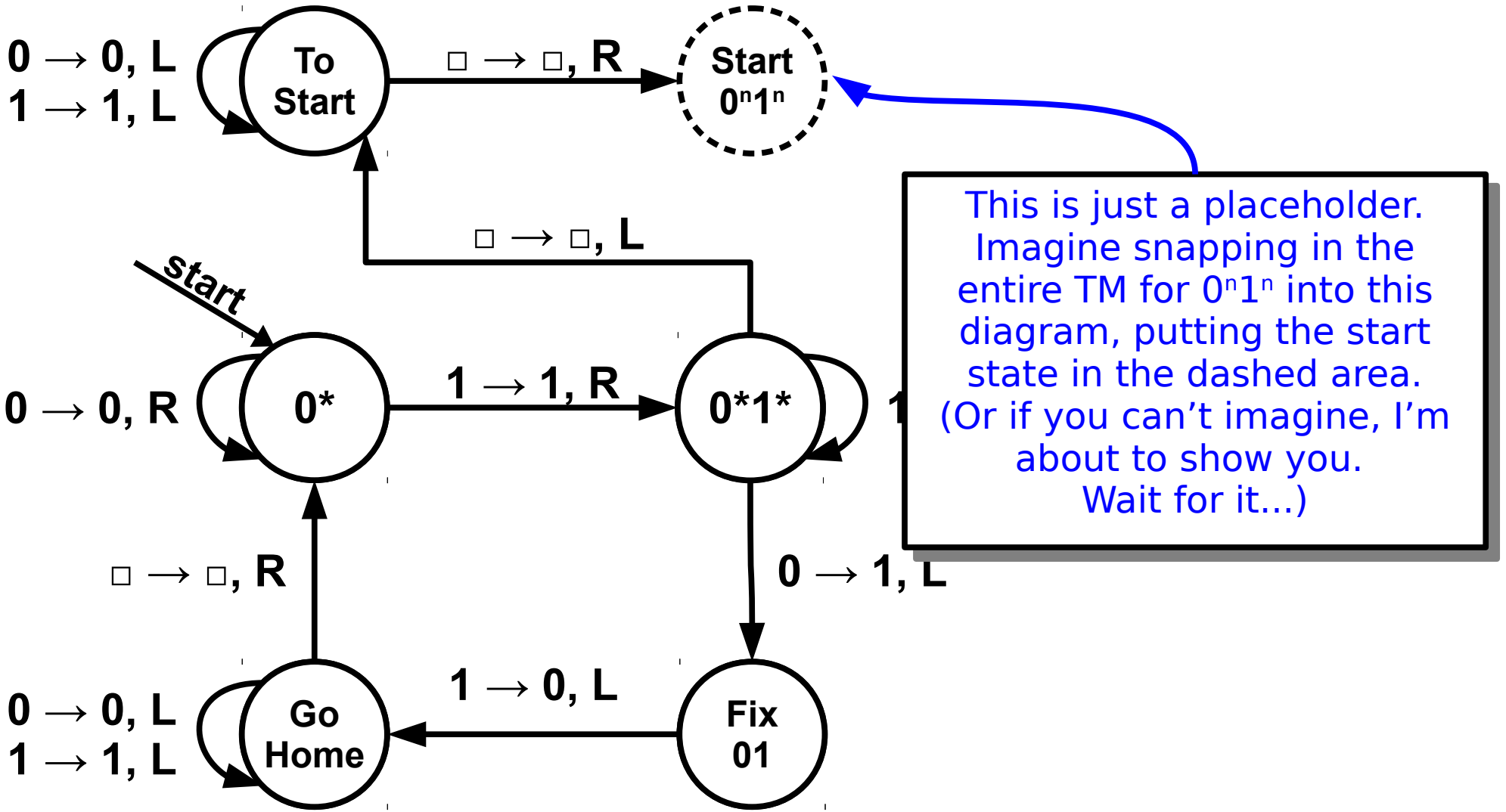

0 → 0, L
1 → 1, L
To Start
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

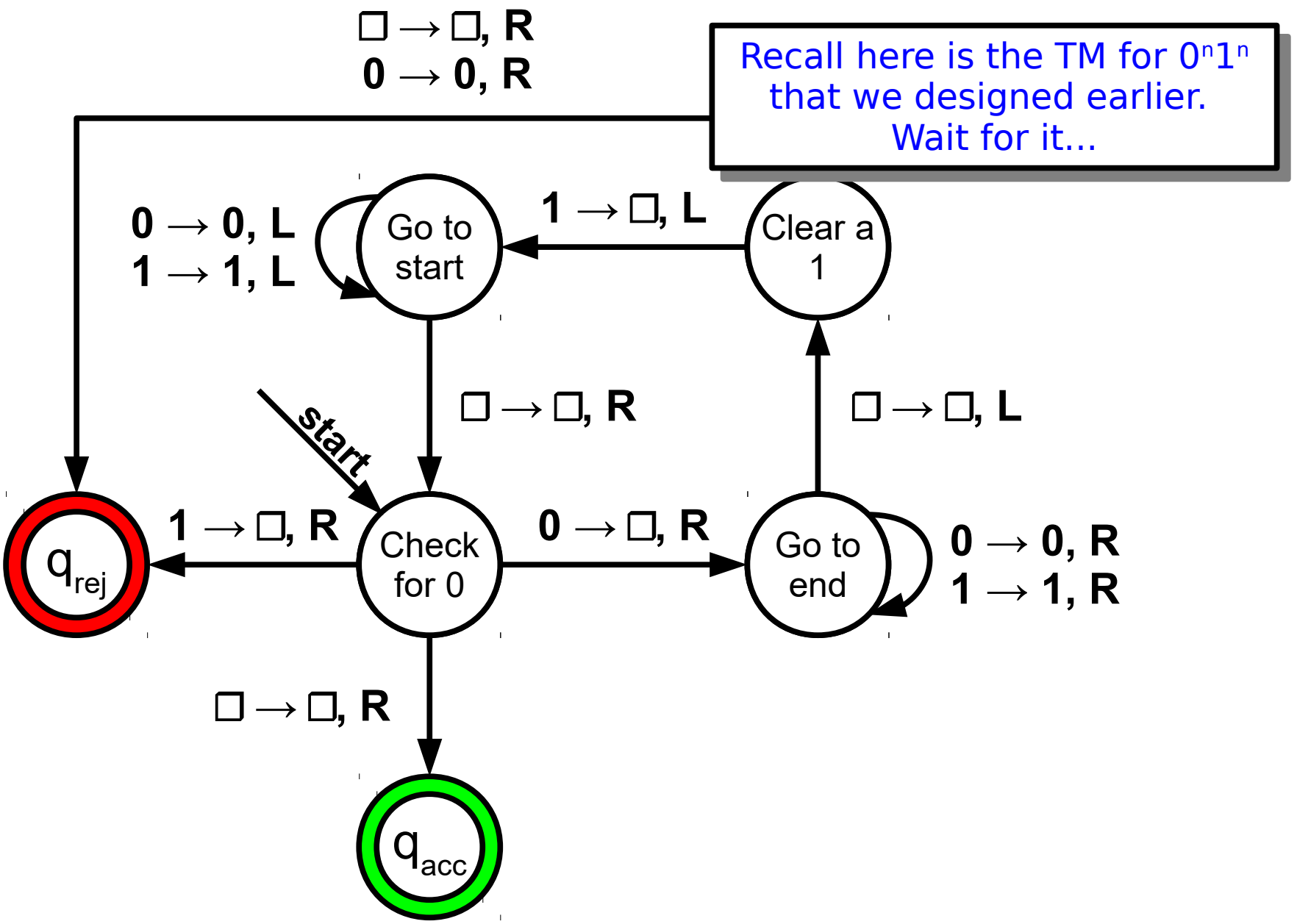

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0 0 0 0 1 1
…

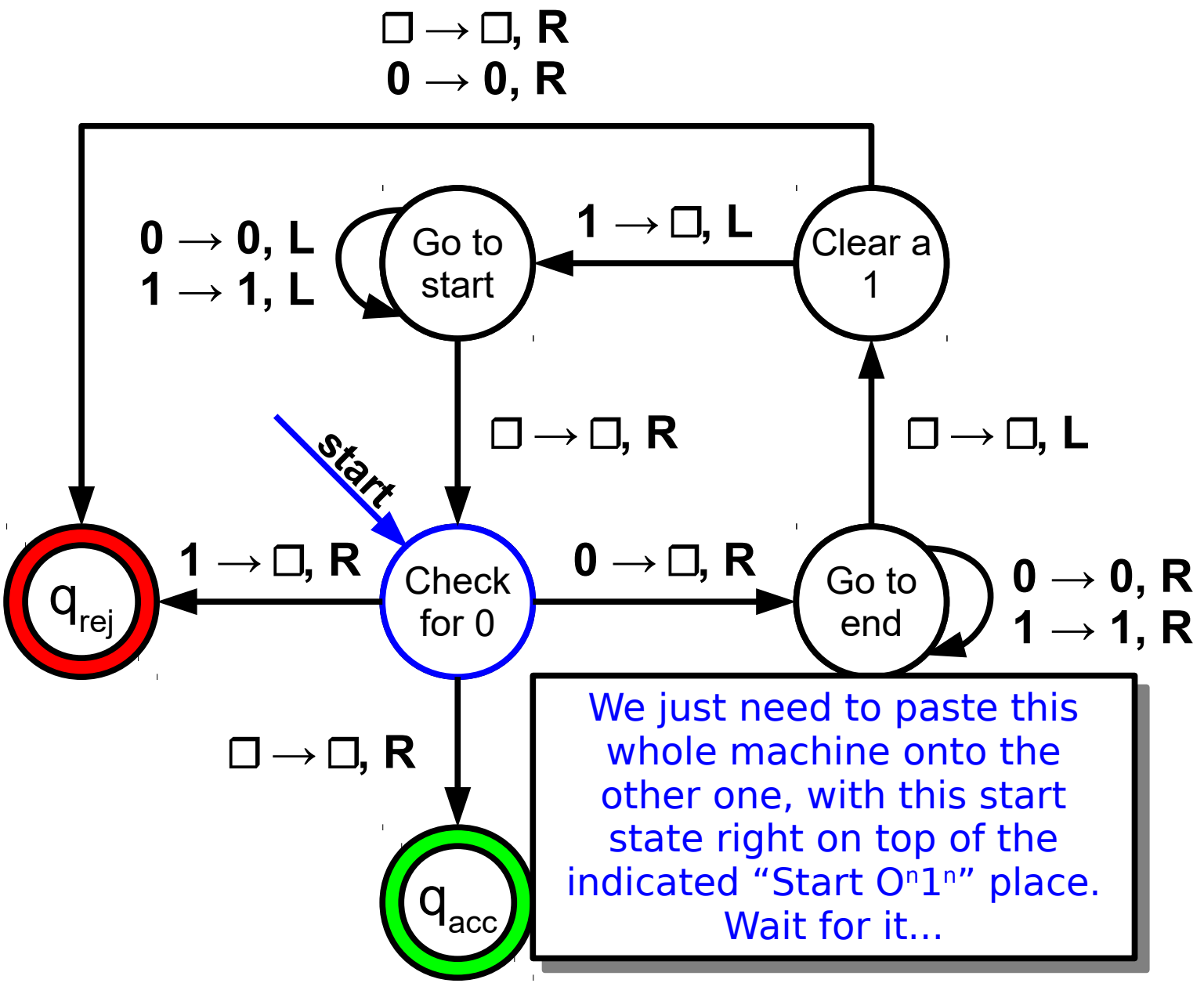

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01


…
0 0 0 0 1 1
…

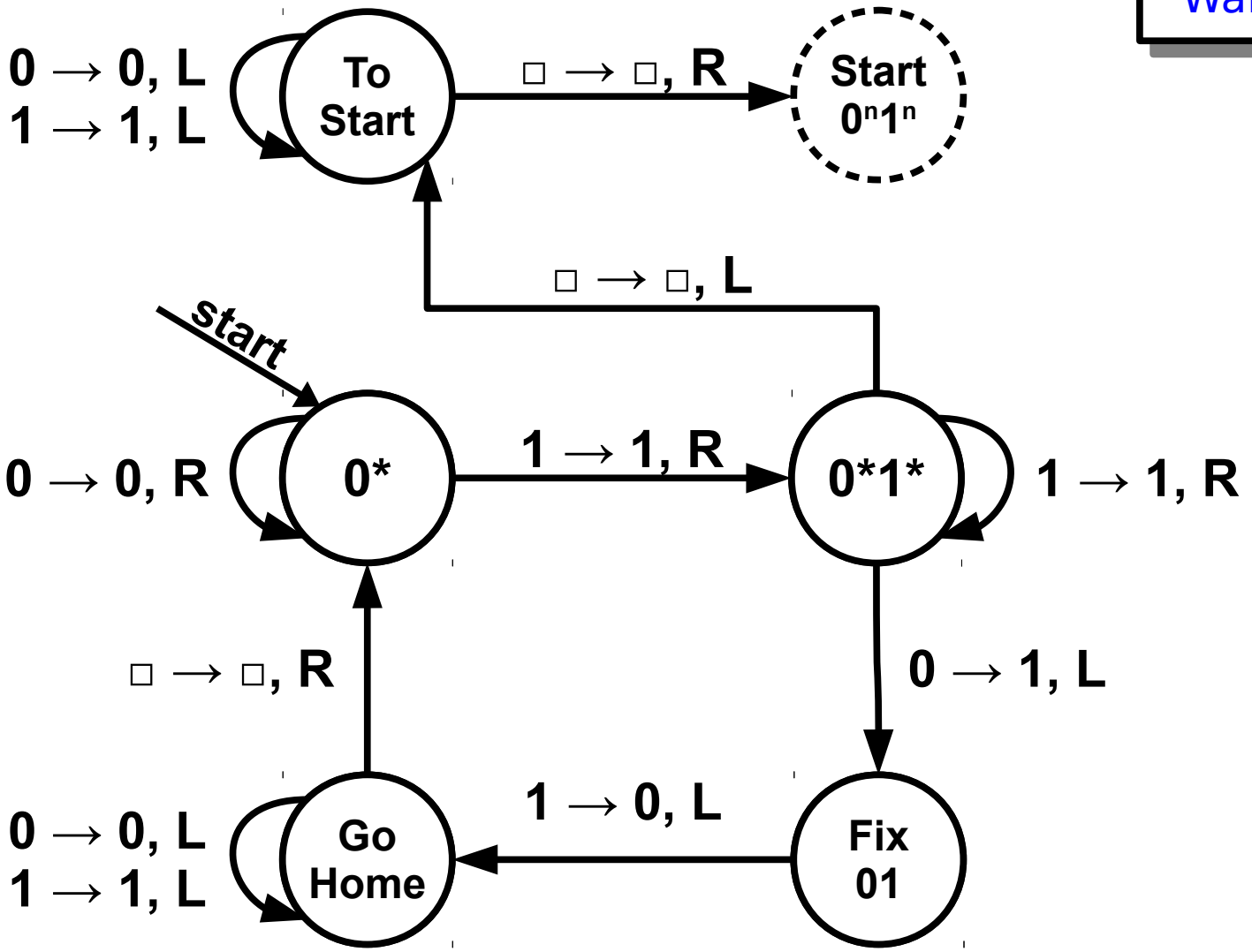

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0^n1^n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

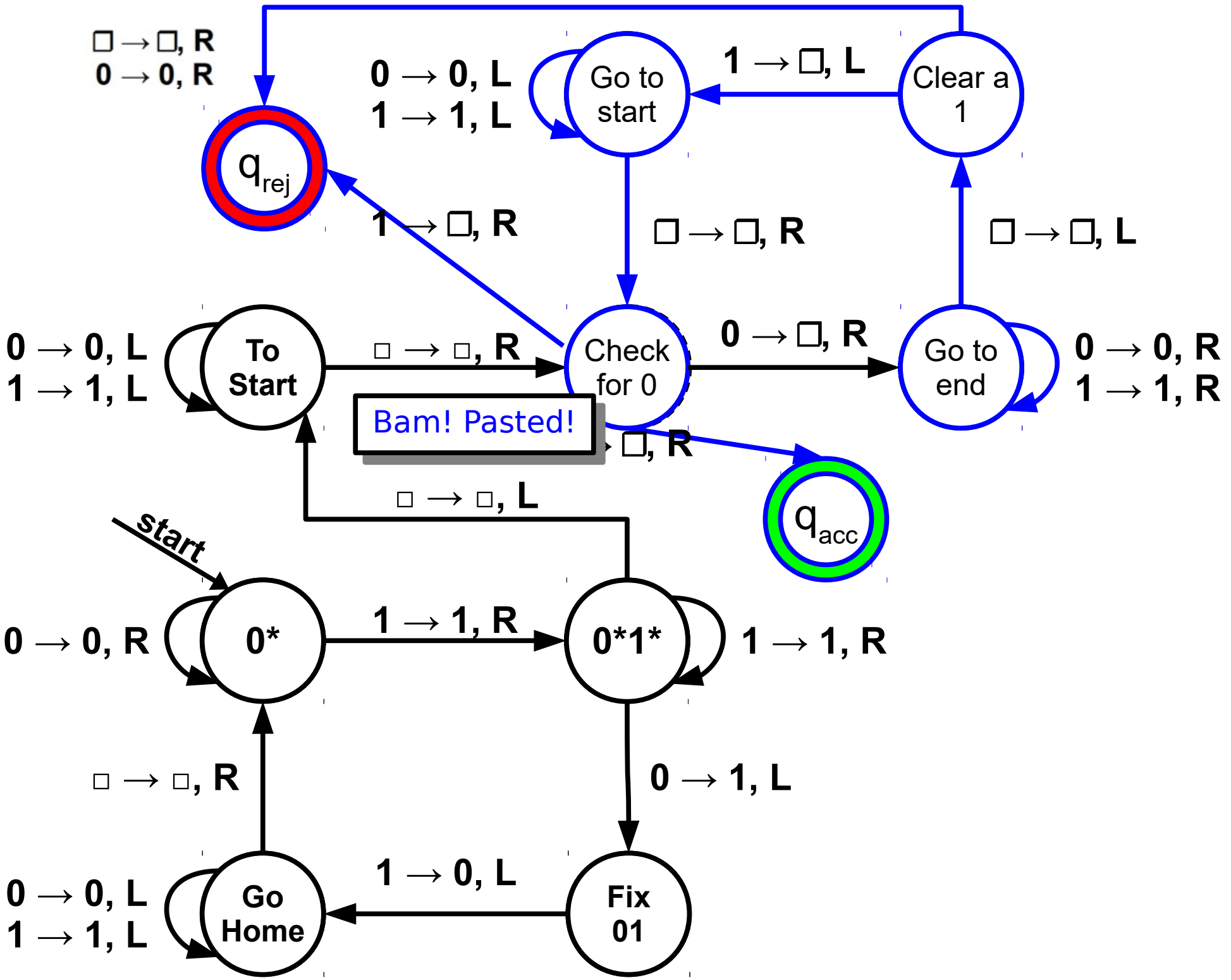

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0ⁿ1ⁿ
This is just a placeholder.
Imagine snapping in the entire TM for 0ⁿ1ⁿ into this diagram, putting the start state in the dashed area.
(Or if you can't imagine, I'm about to show you.
Wait for it...)
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

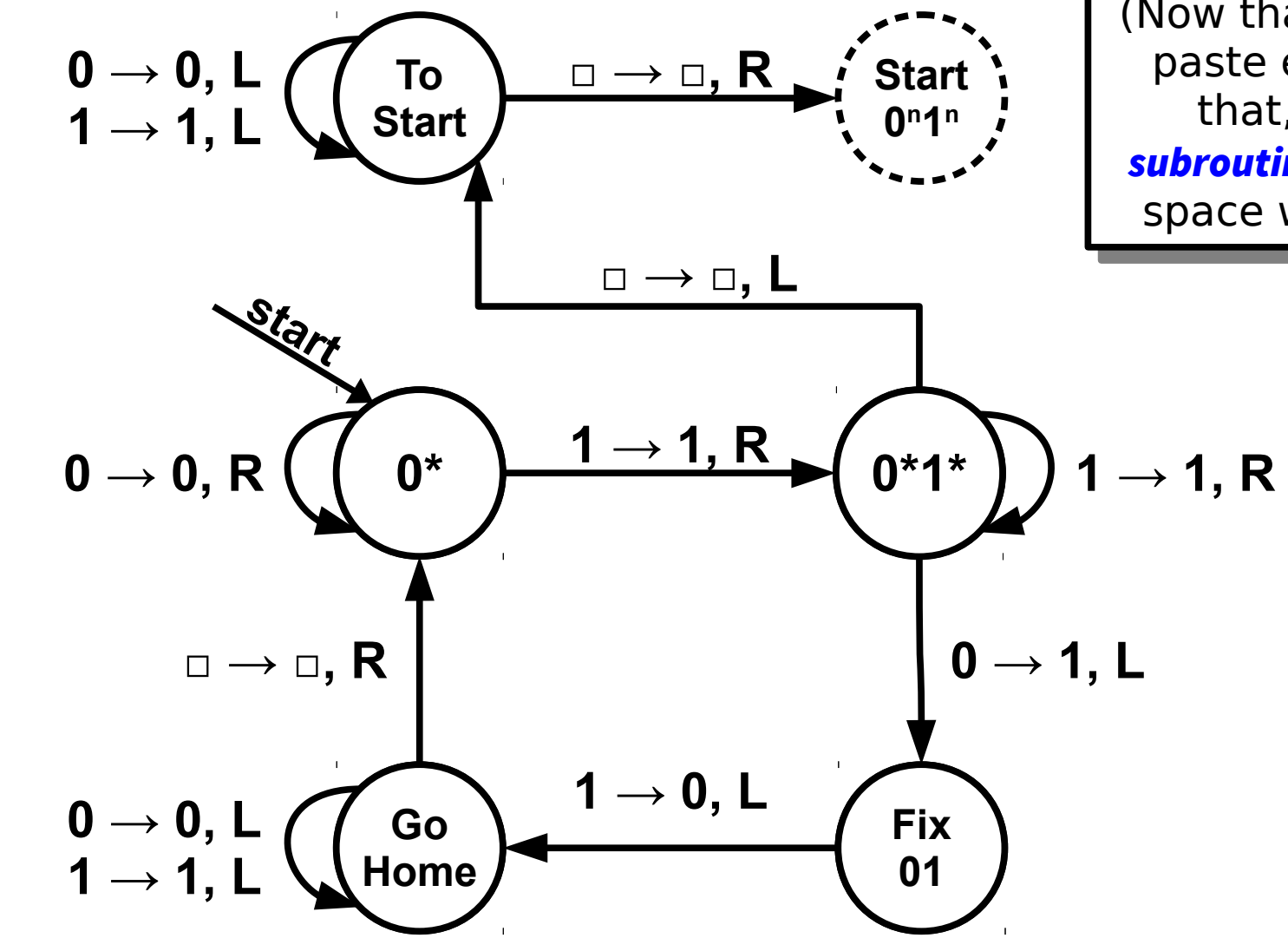

□ → □, R
0 → 0, R
Recall here is the TM for 0ⁿ1ⁿ
that we designed earlier.
Wait for it...
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
□ → □, R
□ → □, L
start
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc


□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
□ → □, R
□ → □, L
start
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
We just need to paste this whole machine onto the other one, with this start state right on top of the indicated “Start O^n1^n” place. Wait for it...

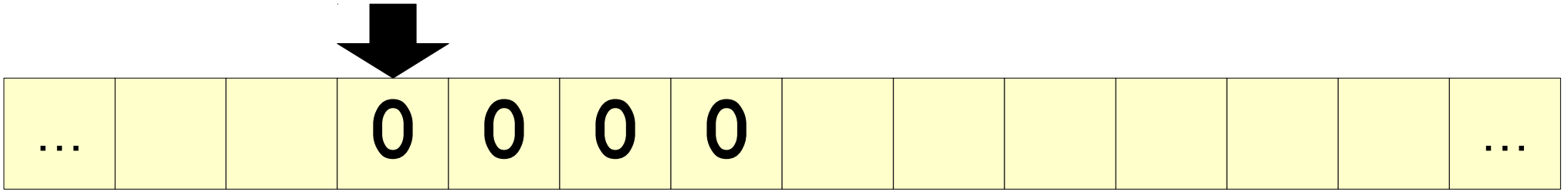
Wait for it...

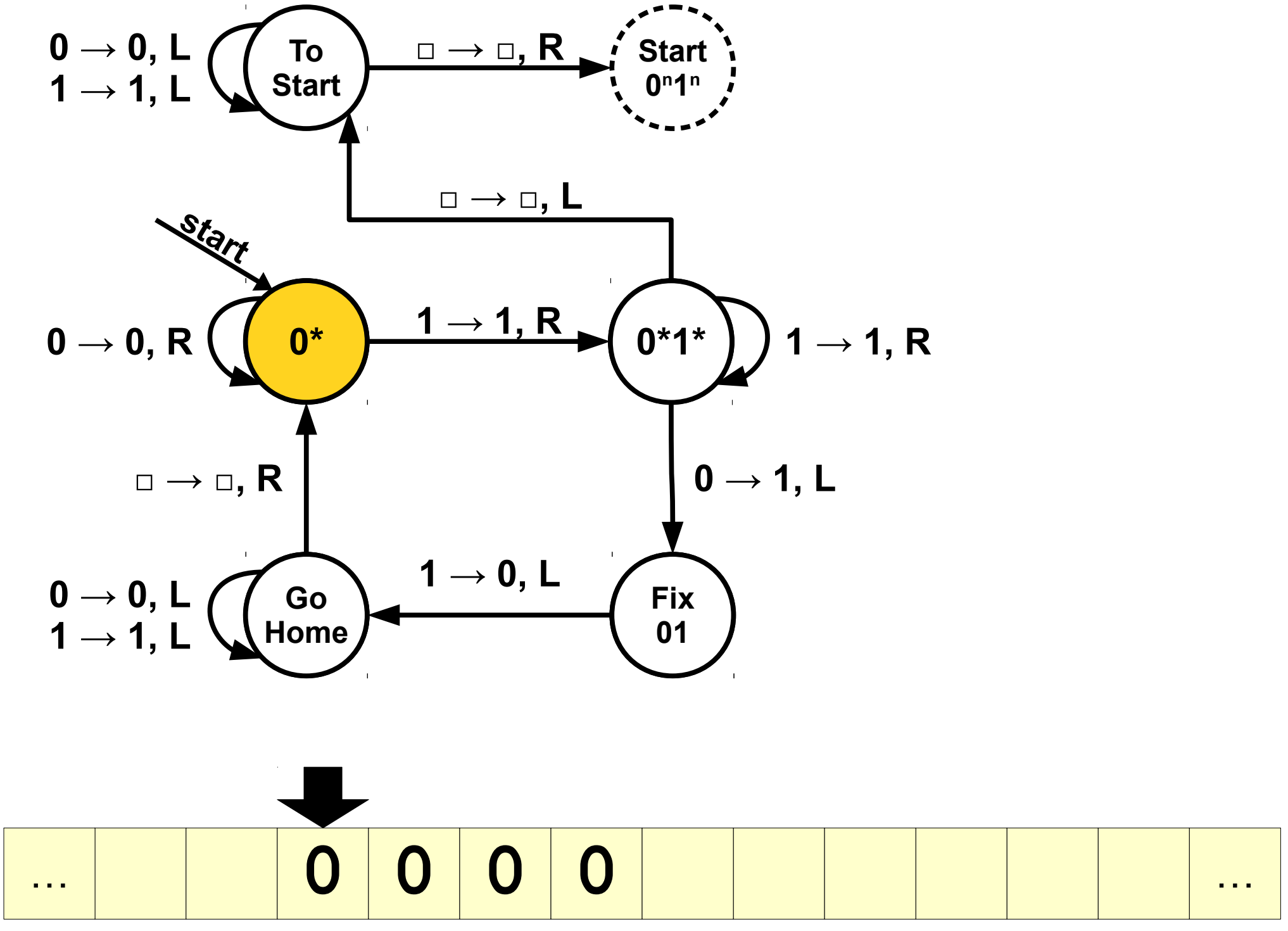

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0ⁿ1ⁿ
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

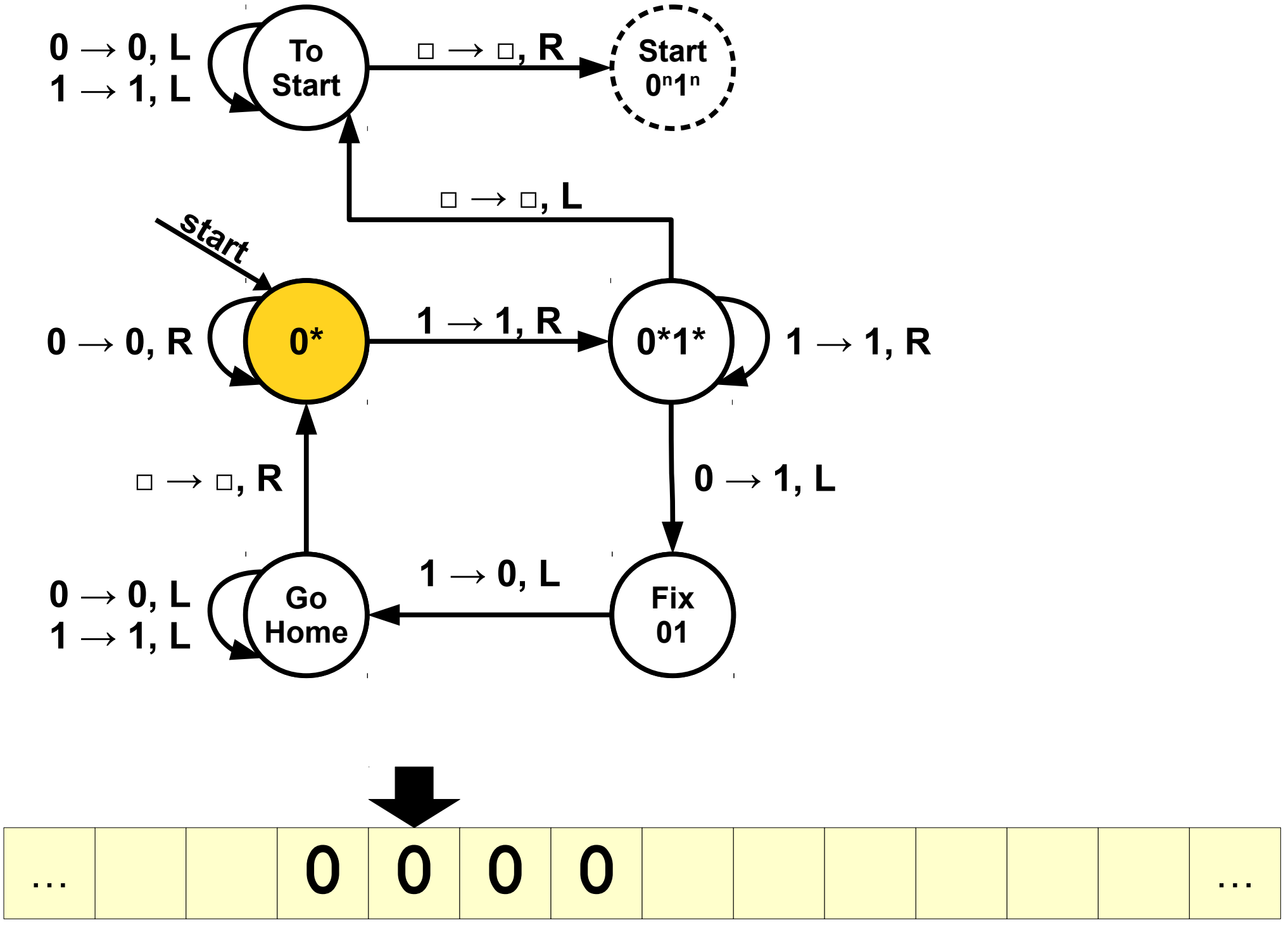

□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
q_rej
1 → □, R
□ → □, R
□ → □, L
0 → 0, L
1 → 1, L
To Start
□ → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
Bam! Pasted!
□ → □, R
q_acc
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01

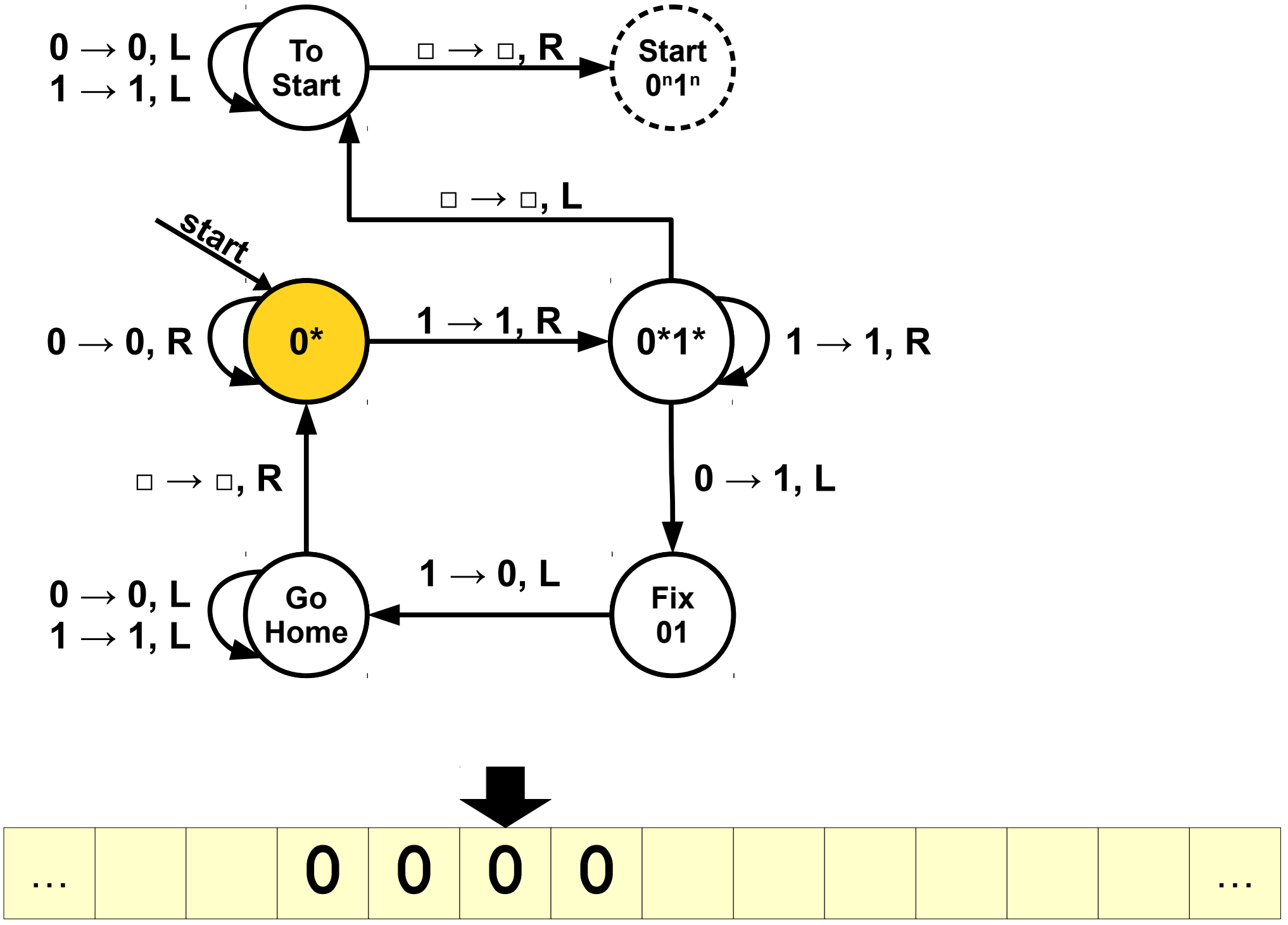


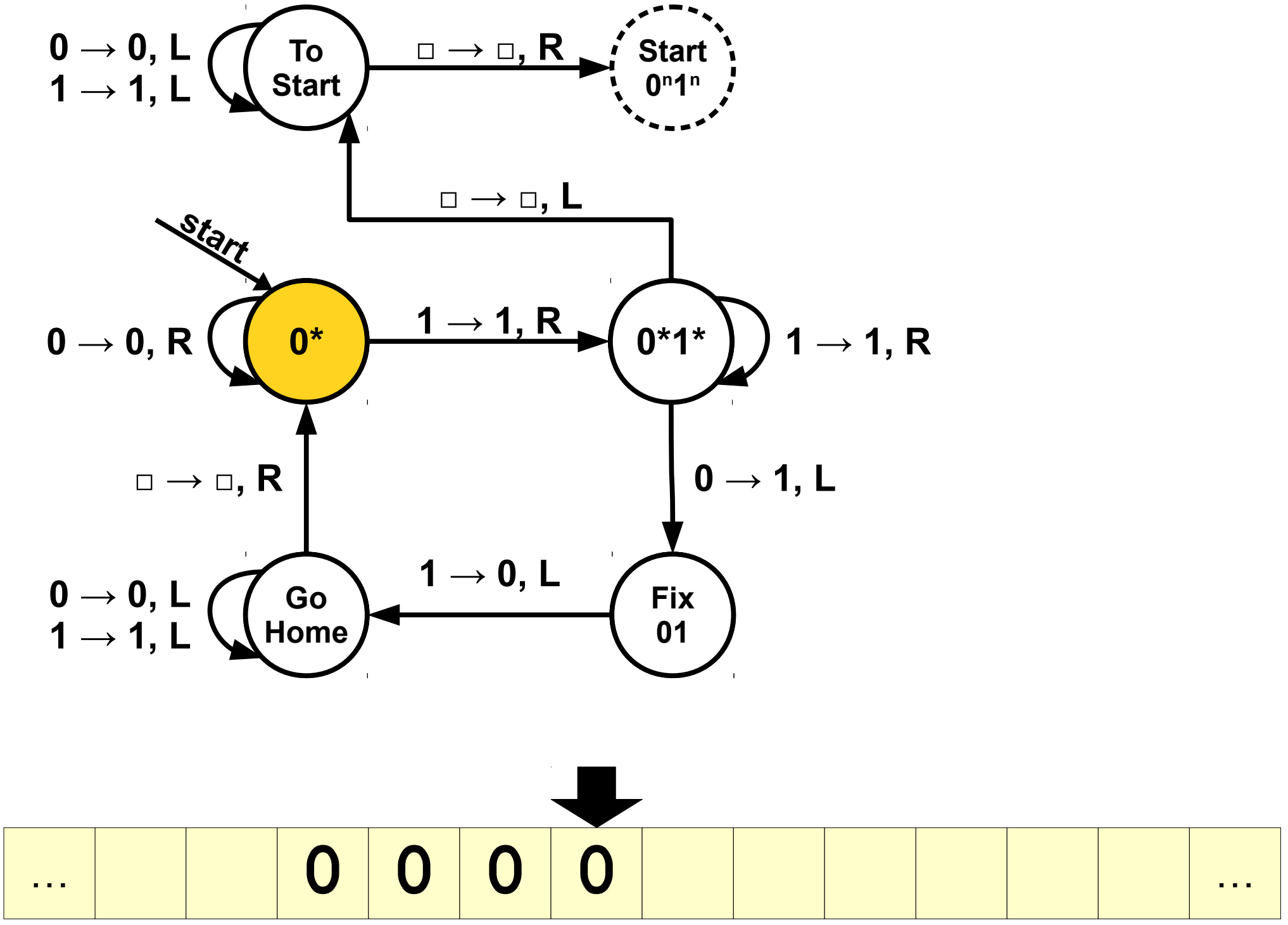
(Now that we know we can paste entire TMs on like that, using them as ***subroutines***, for the sake of space we won't show it.)

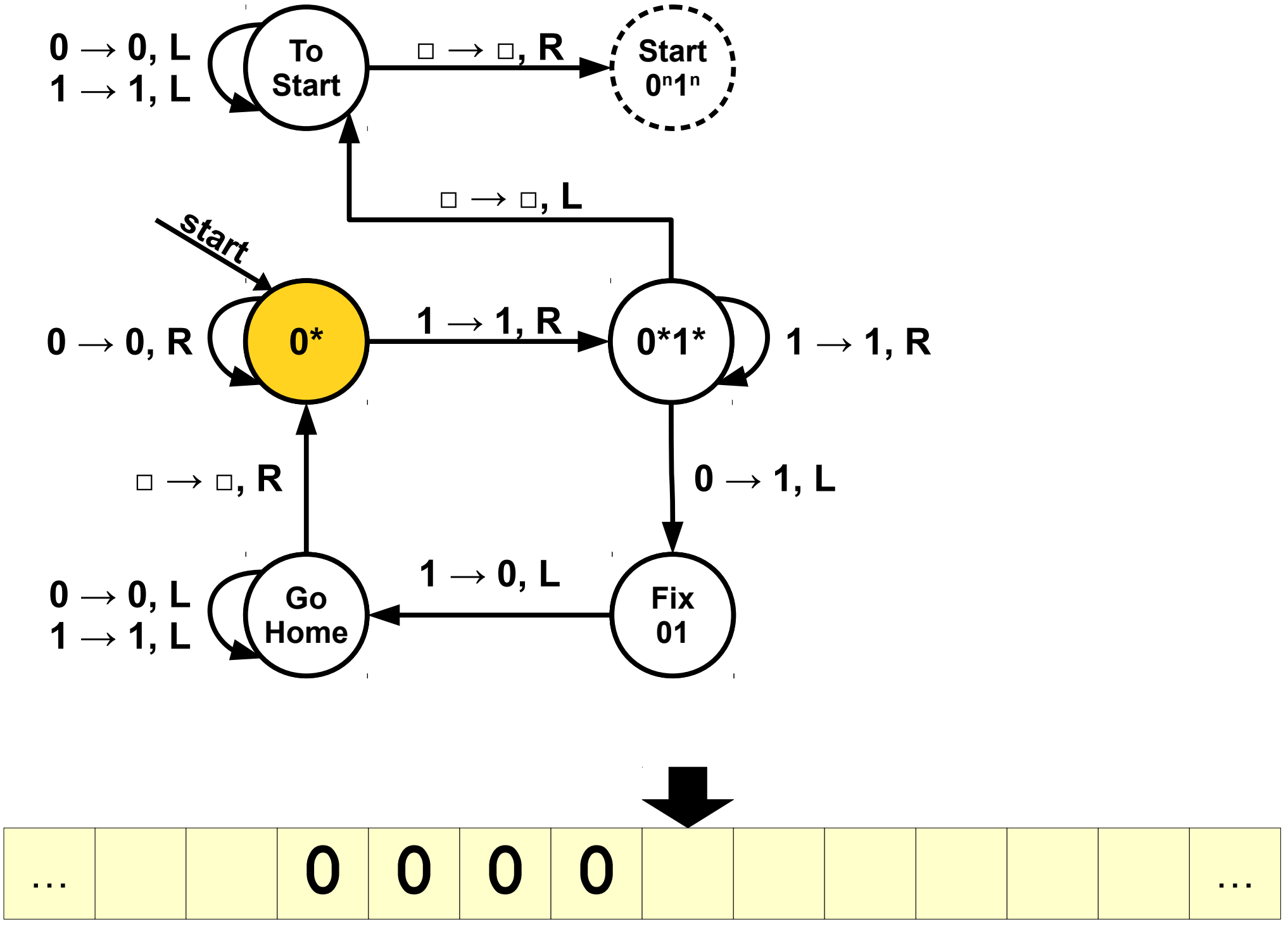


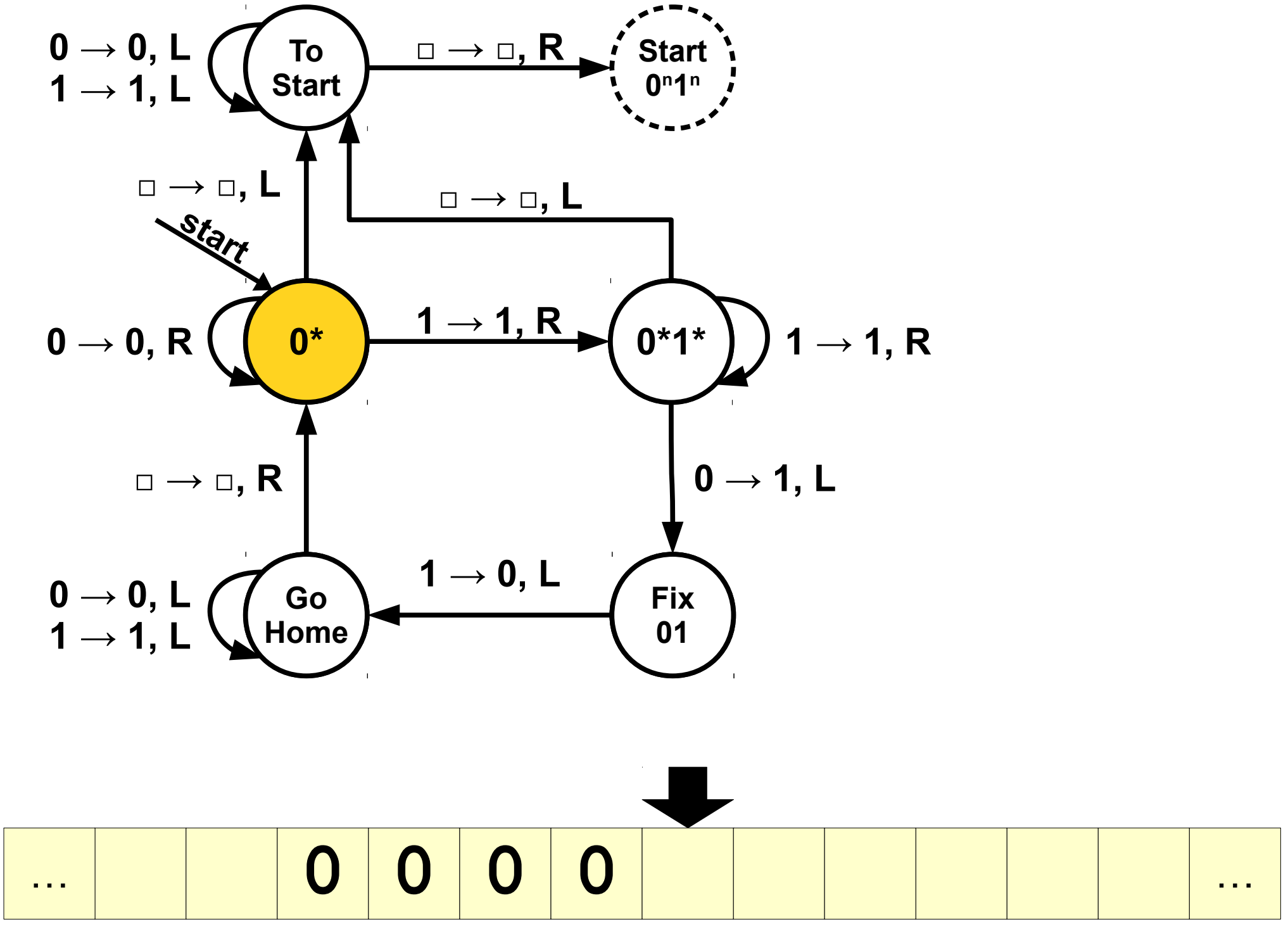

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

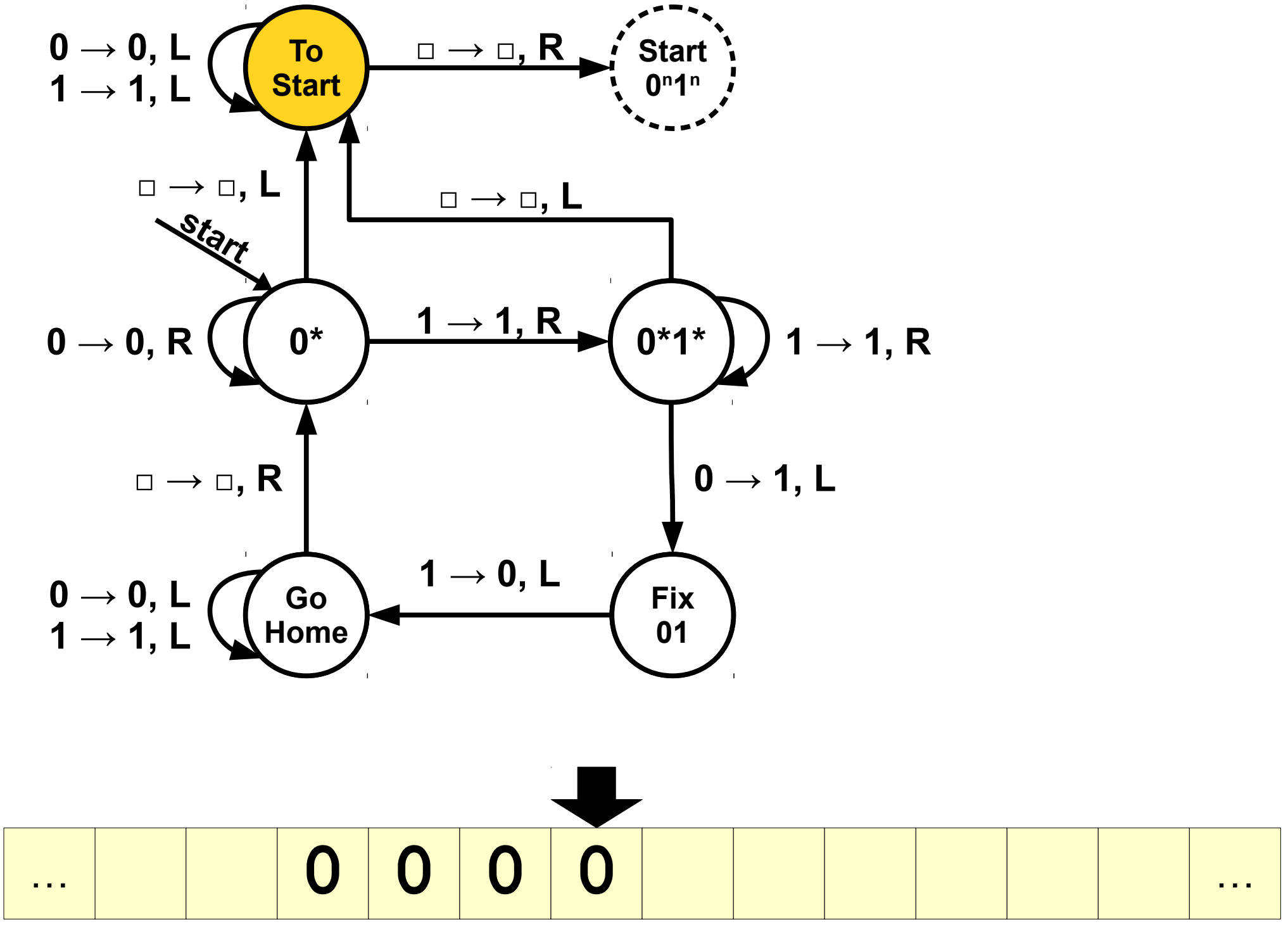

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

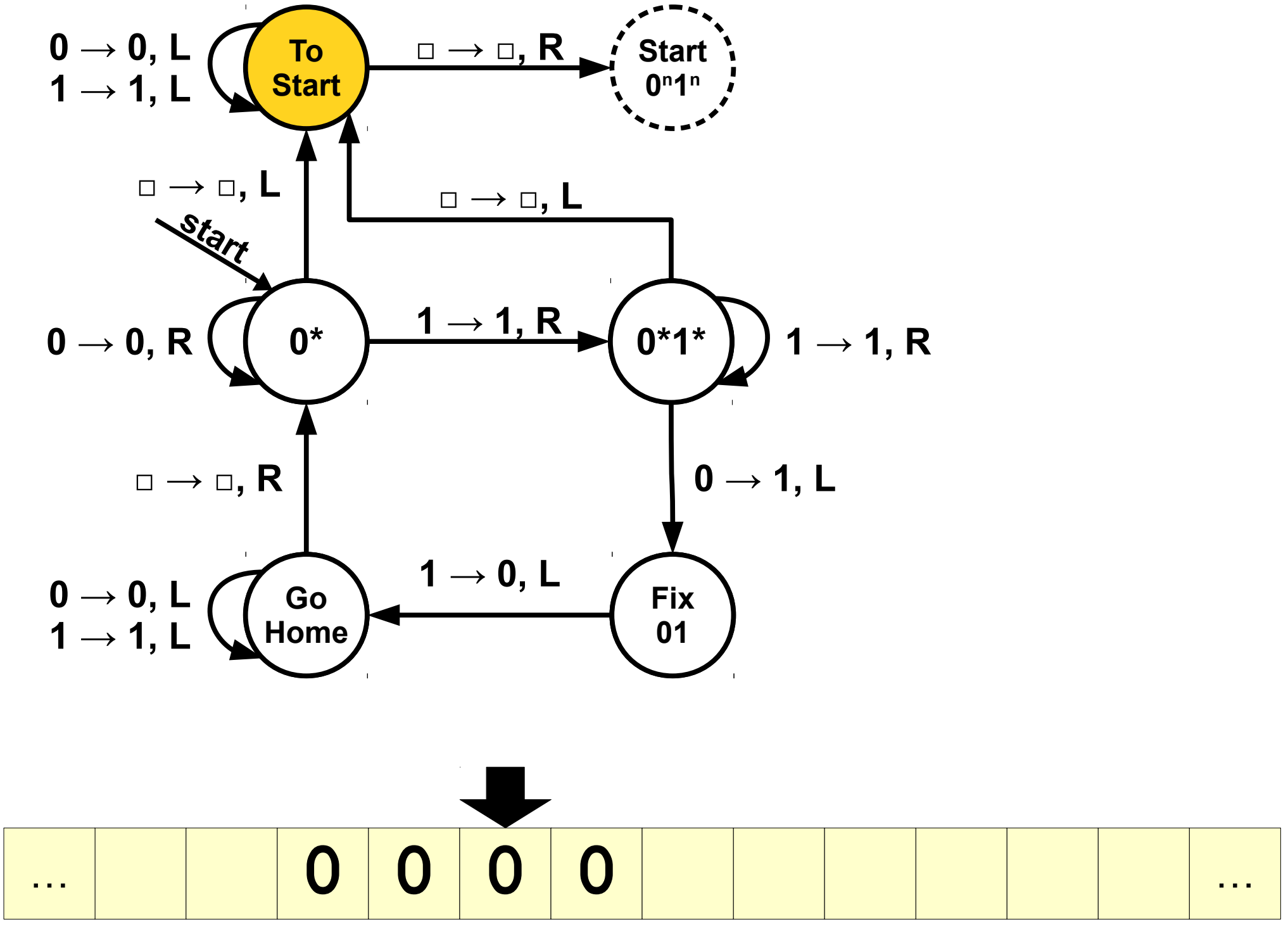

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0^n1^n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

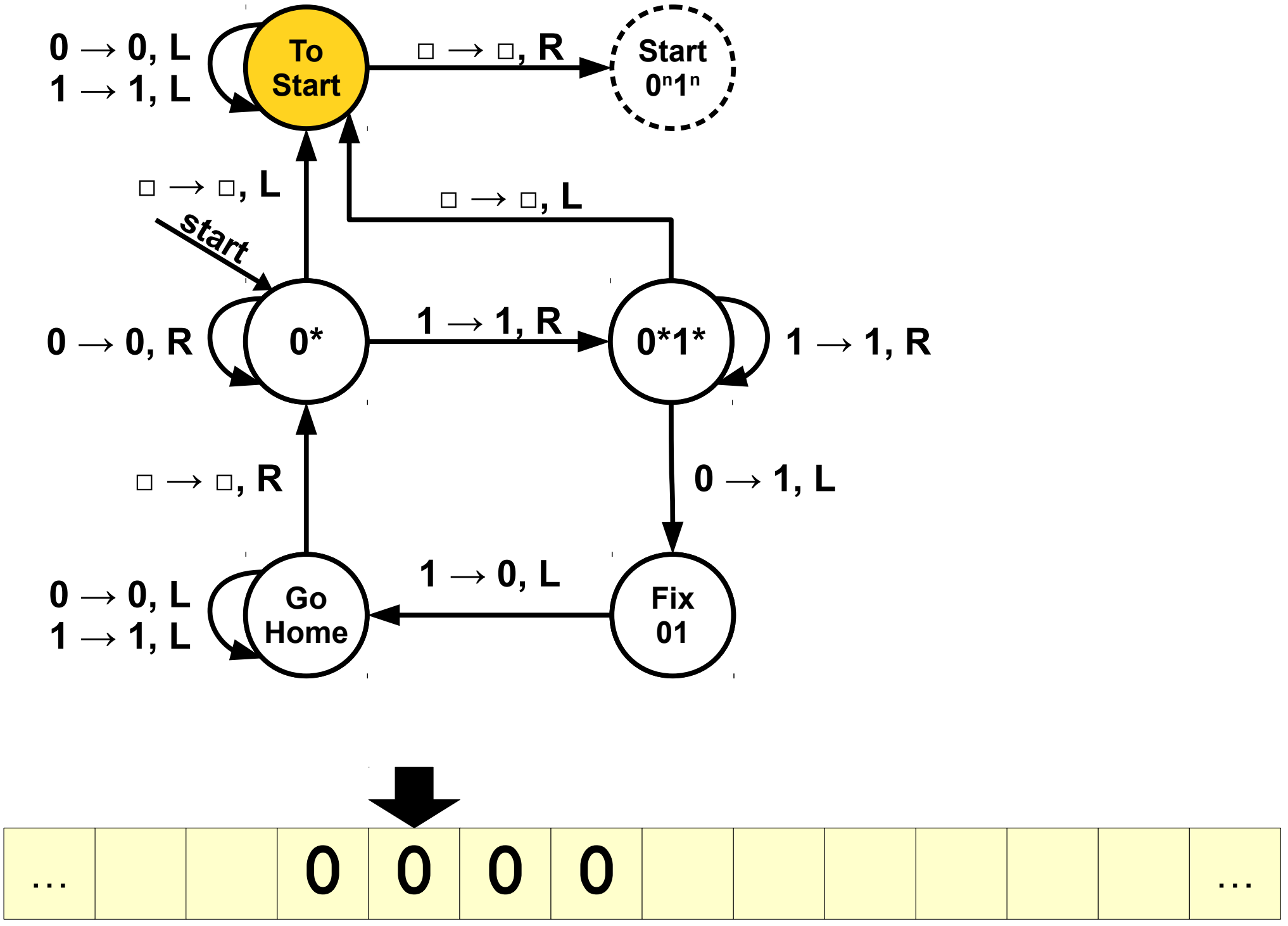

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

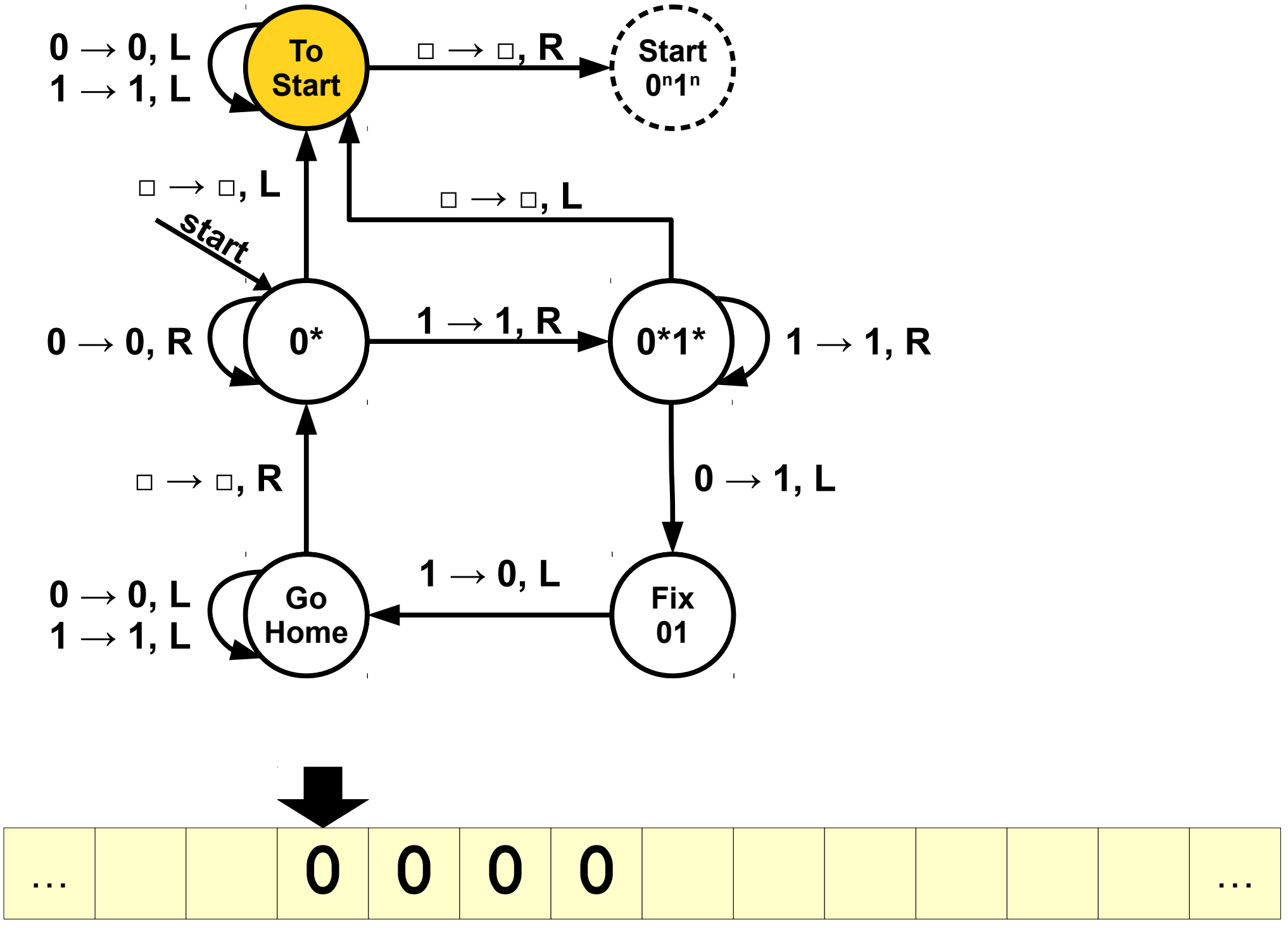

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

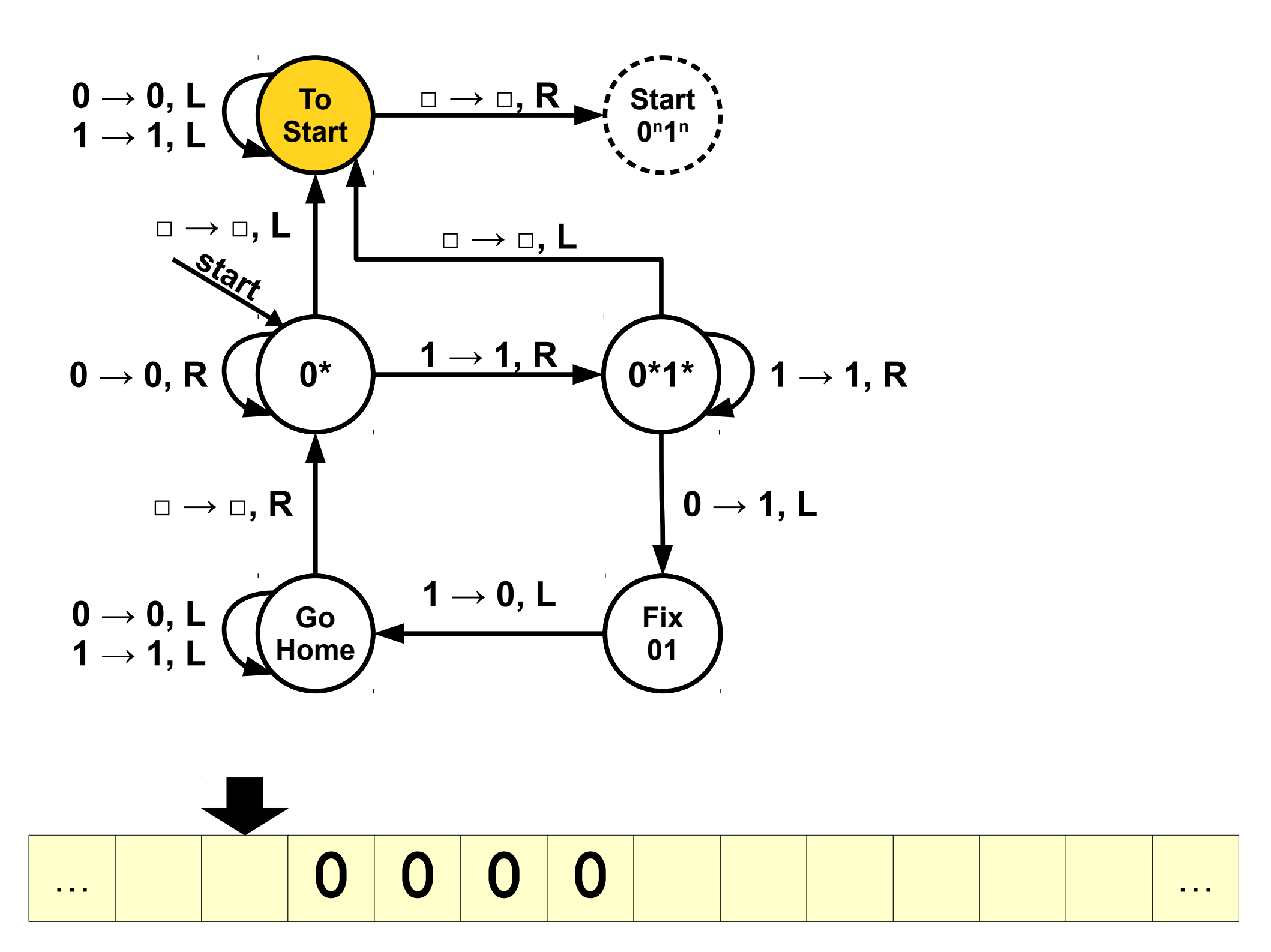

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

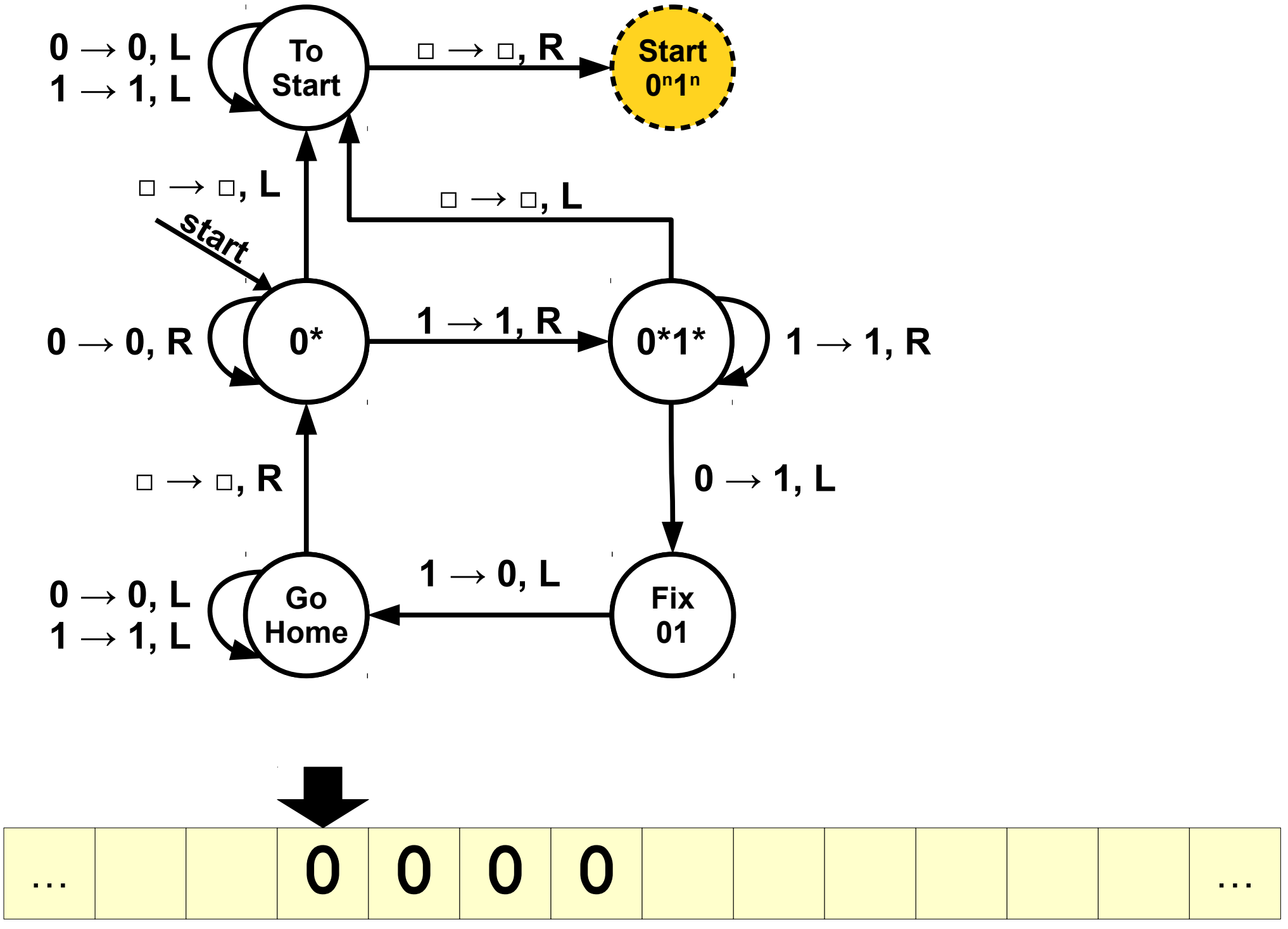

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

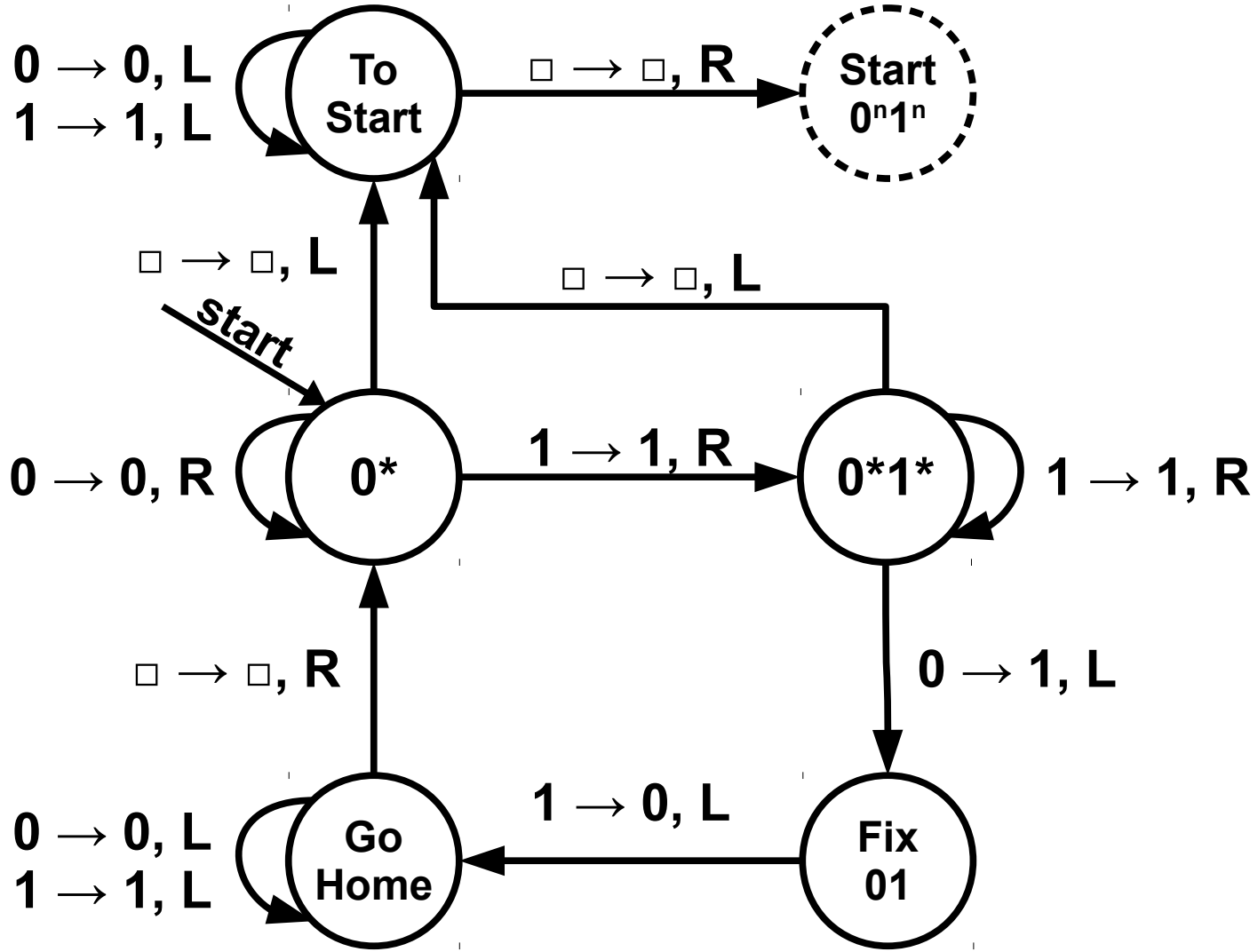

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…

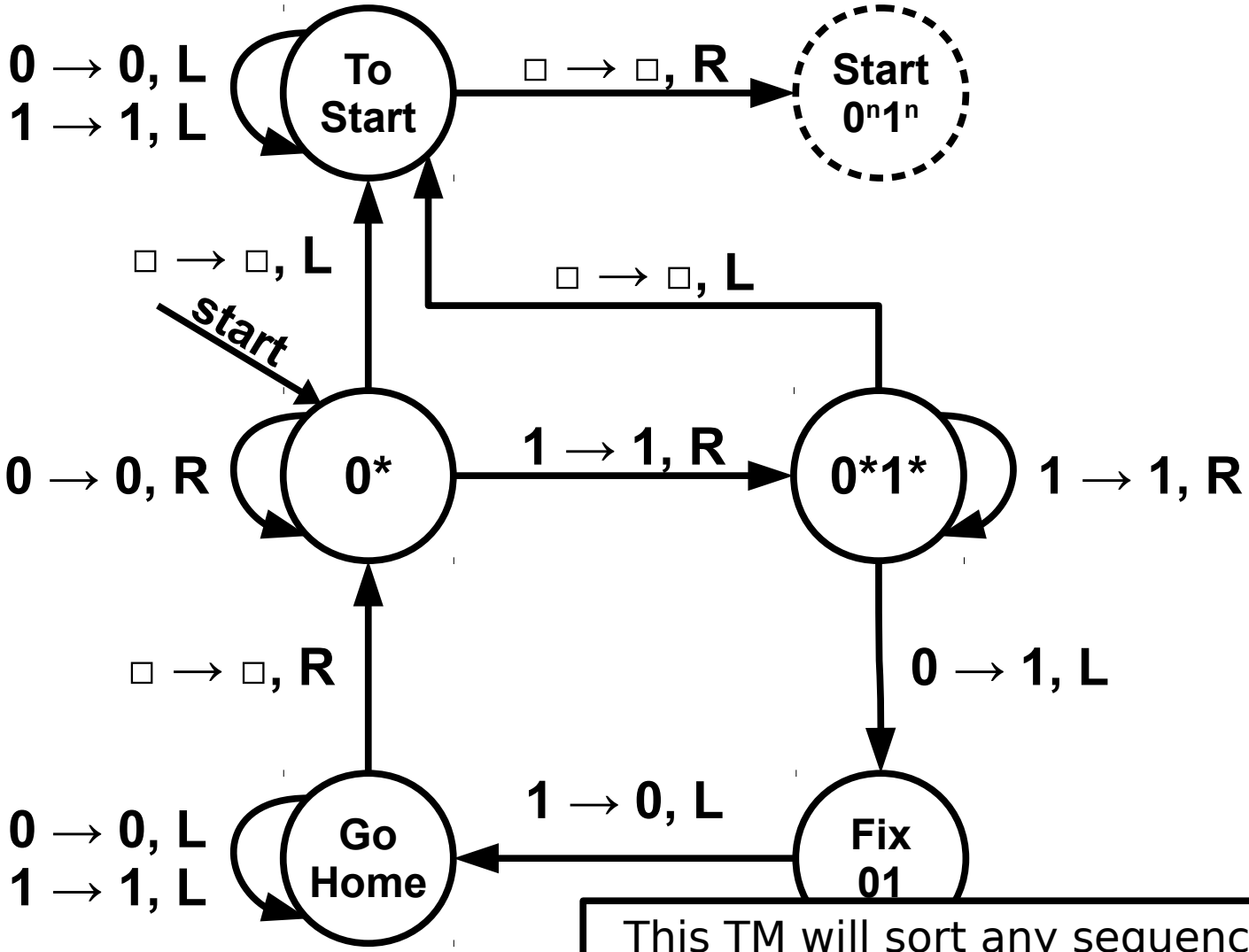

0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…


0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0^n1^n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…


0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…


0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01
…
0
0
0
0
…


0 → 0, L
1 → 1, L
To Start
□ → □, R
Start
0n1n
□ → □, L
□ → □, L
start
0 → 0, R
0*
1 → 1, R
0*1*
1 → 1, R
□ → □, R
0 → 1, L
0 → 0, L
1 → 1, L
Go Home
1 → 0, L
Fix 01



This TM will sort any sequence of 0s and 1s, but it might take a while.

Fun problem: design a TM that sorts a string of 0s and 1s, but does so while taking way fewer steps than this machine.

TM Subroutines

- A ***TM subroutine*** is a Turing machine that, instead of accepting or rejecting an input, does some sort of processing job.
- TM subroutines let us compose larger TMs out of smaller TMs, just as you'd write a larger program using lots of smaller helper functions.
- Here, we saw a TM subroutine that sorts a sequence of 0s and 1s into ascending order.

TM Subroutines

- Typically, when a subroutine is done running, you have it enter a state marked “done” with a dashed line around it.
- When we're composing multiple subroutines together – which we'll do in a bit – the idea is that we'll snap in some real state for the “done” state.

What other subroutines can we make?