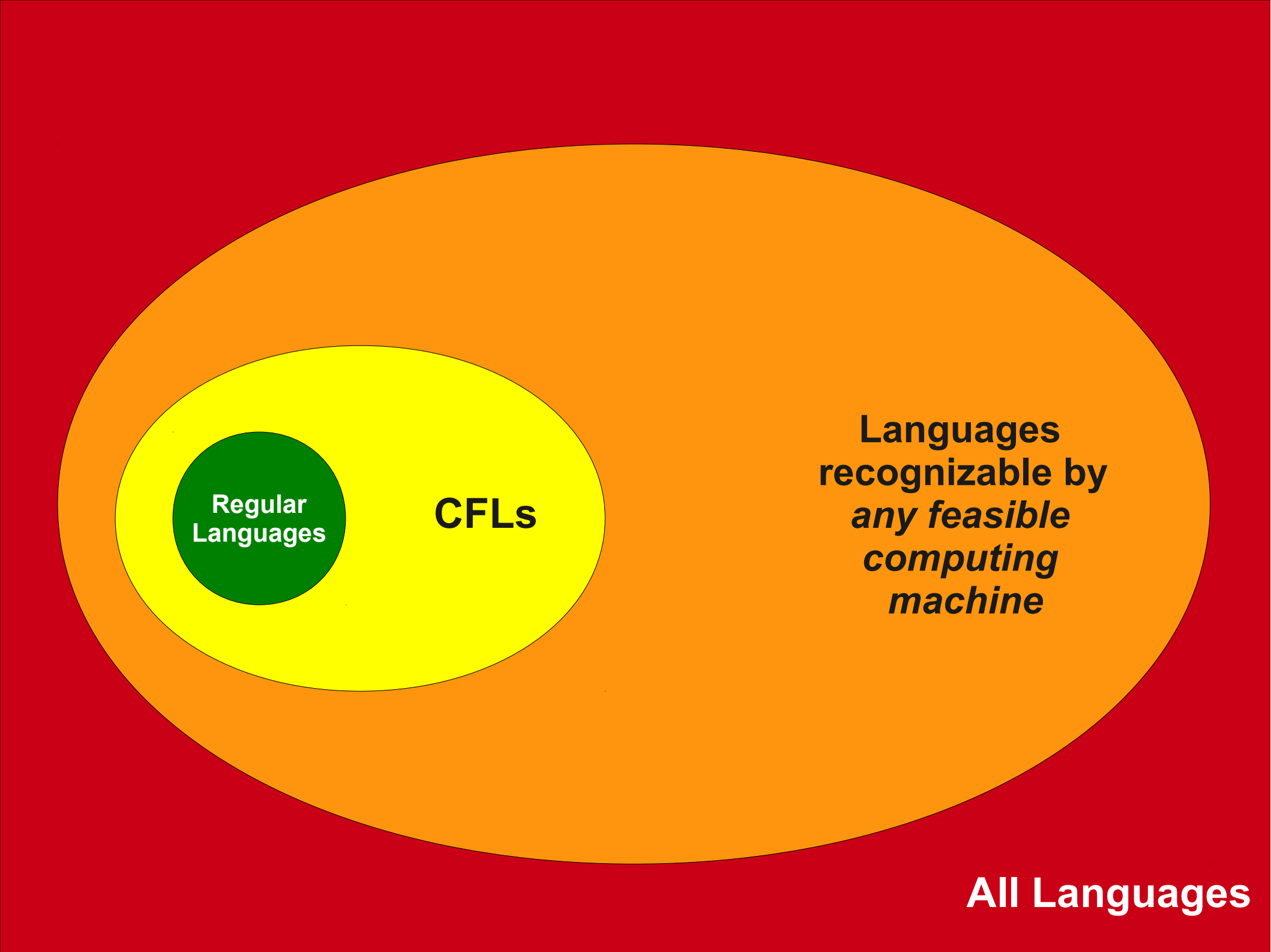
Turing Machines

Part One

Problem Set Five due in the box up front.

Are some problems inherently harder than others?


Regular Languages
CFLs
Languages recognizable by *any feasible computing machine*
All Languages

That same drawing, to scale.

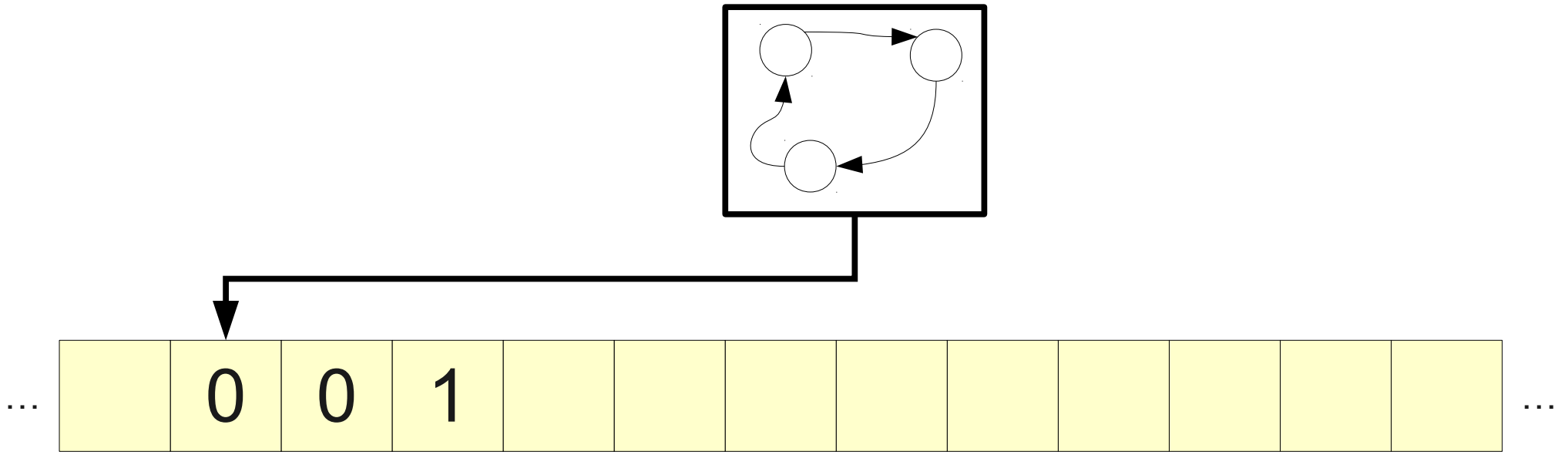
All Languages

The Problem

- Finite automata accept precisely the regular languages.
- We may need unbounded memory to recognize context-free languages.
  - e.g. $\{ \texttt{0}^n\texttt{1}^n \mid n \in \mathbb{N} \}$ requires unbounded counting.
- How do we build an automaton with finitely many states but unbounded memory?

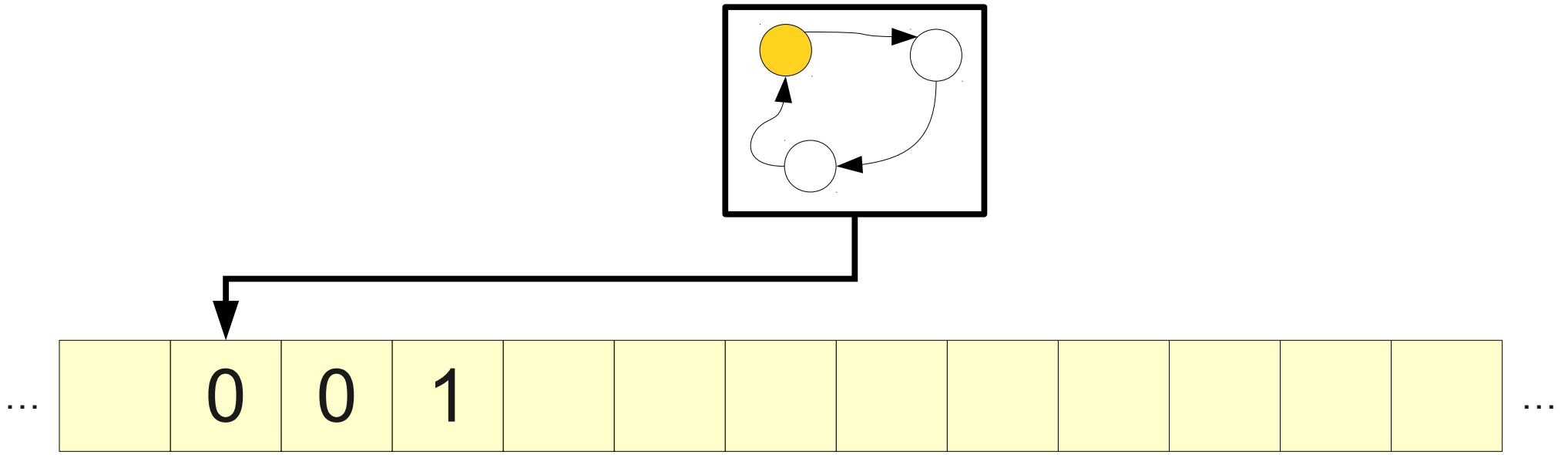
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



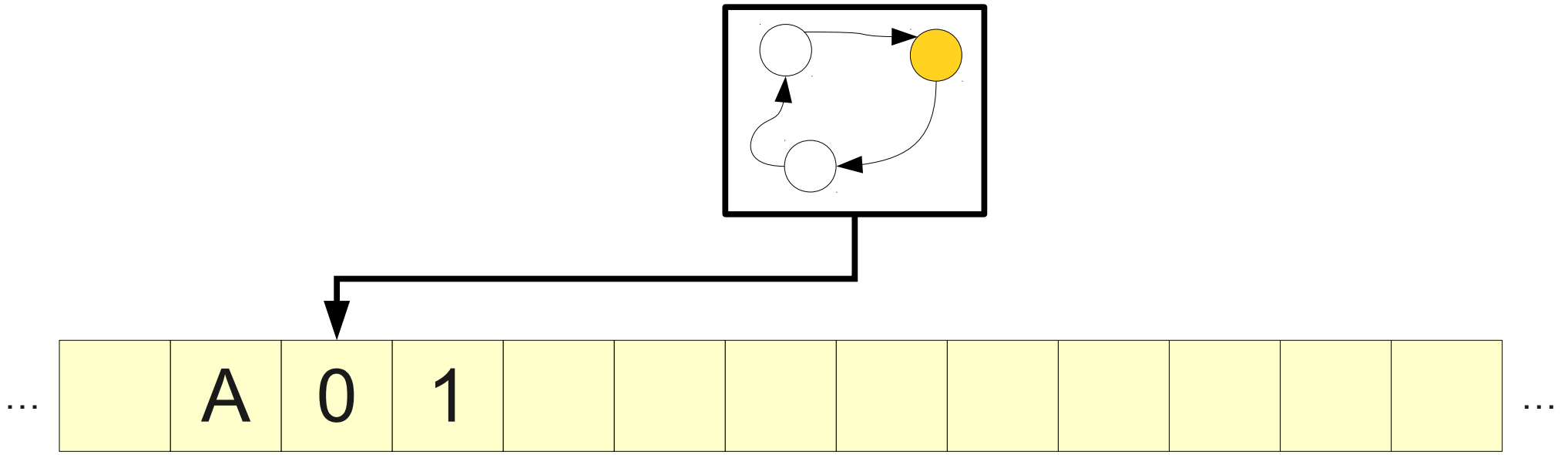
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



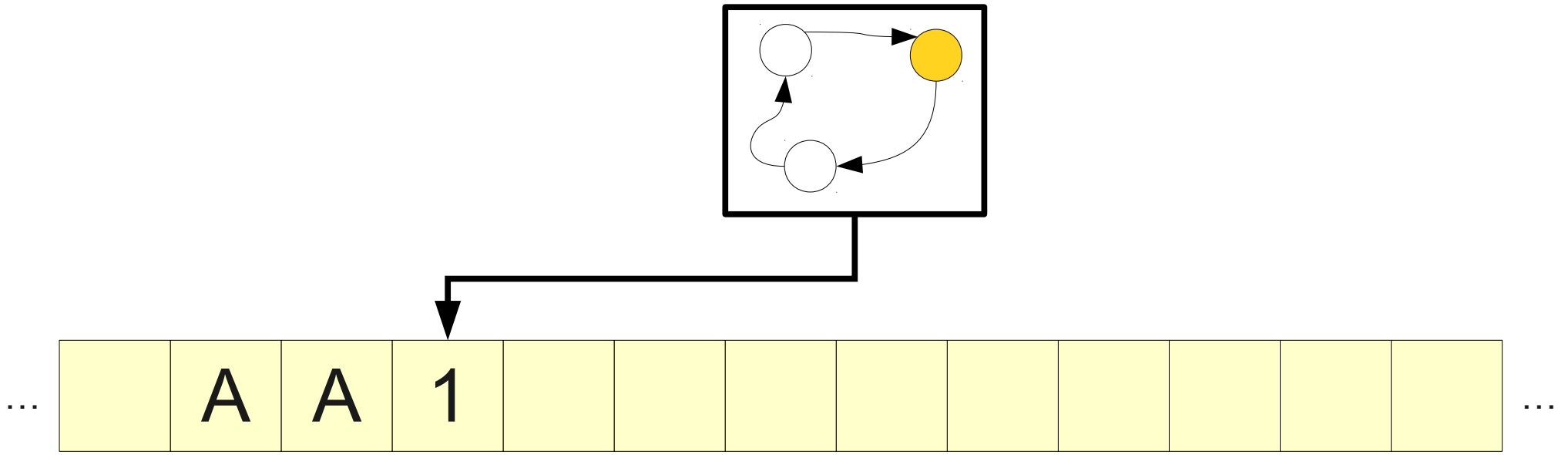
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



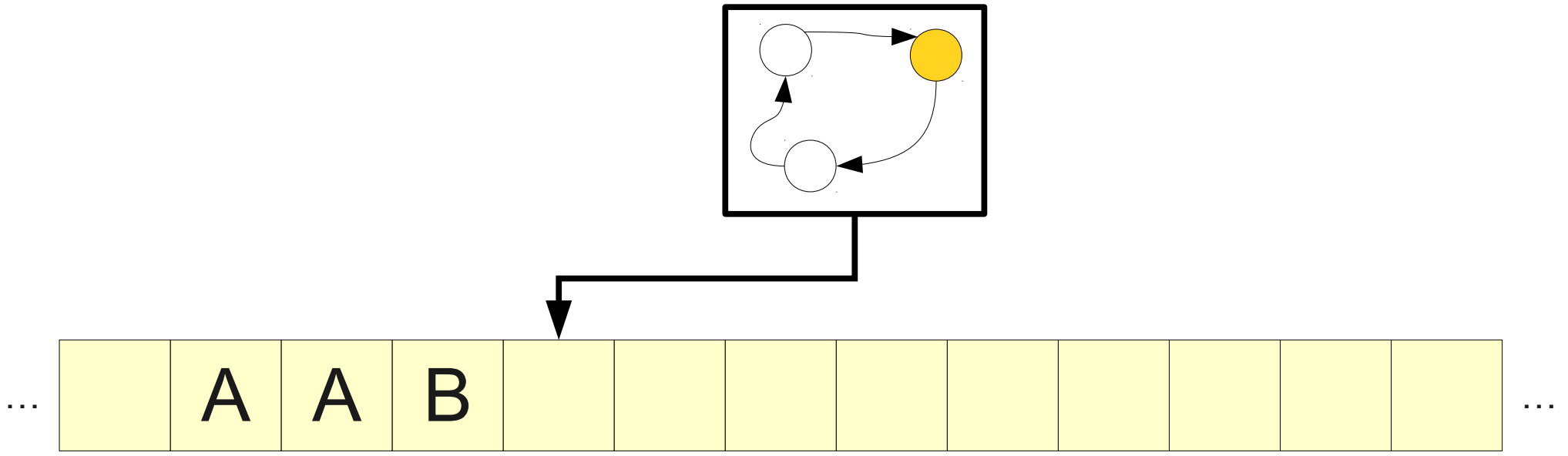
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



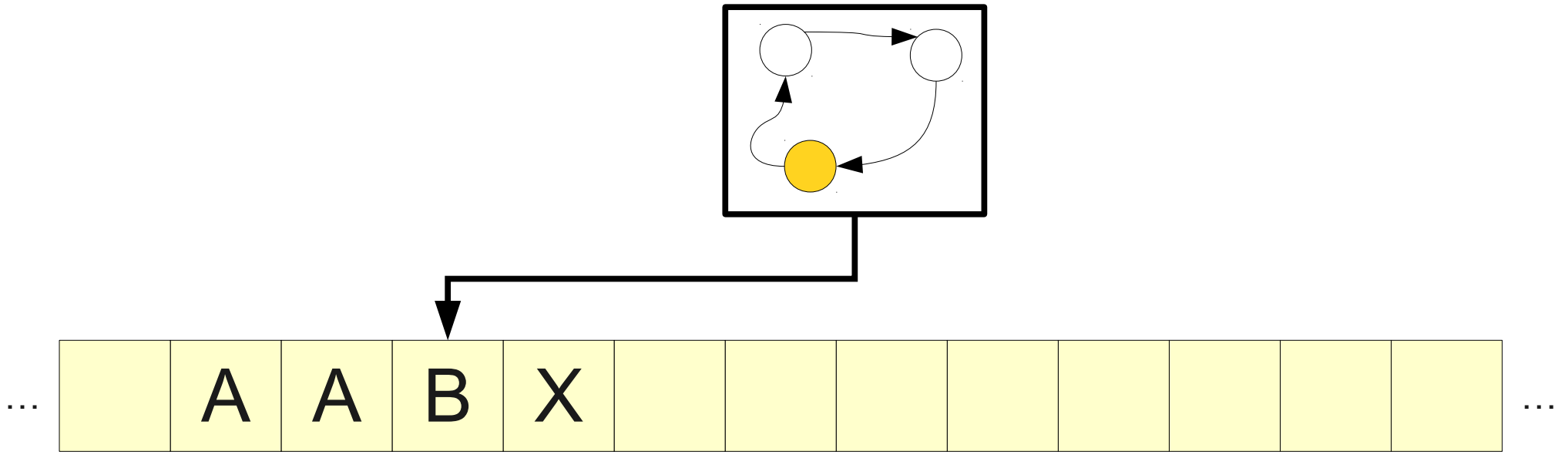
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



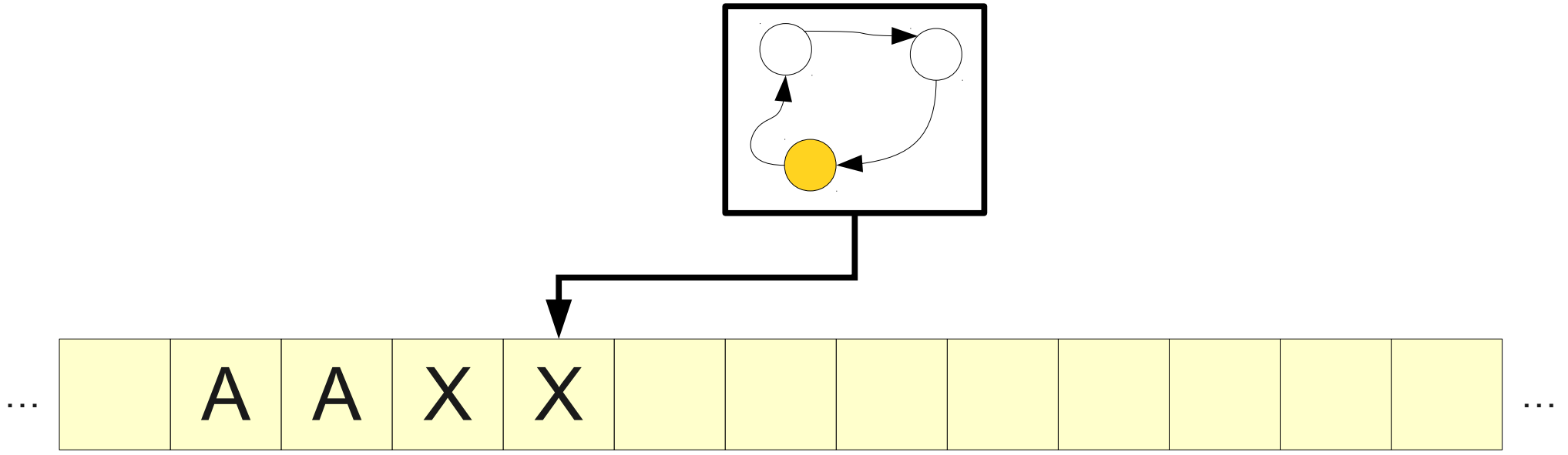
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



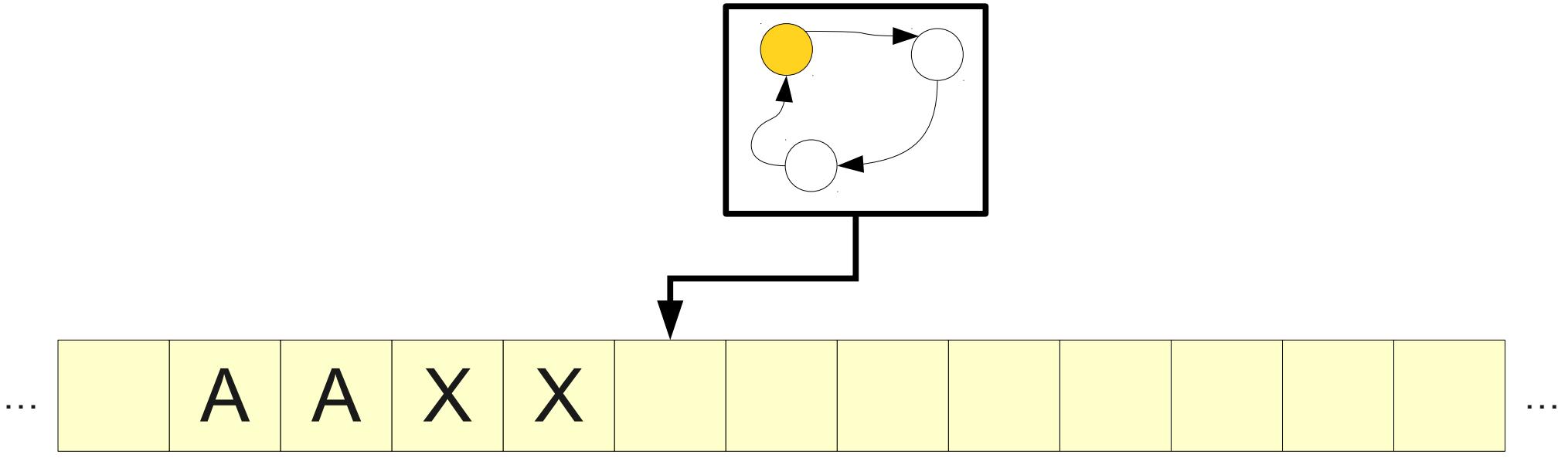
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



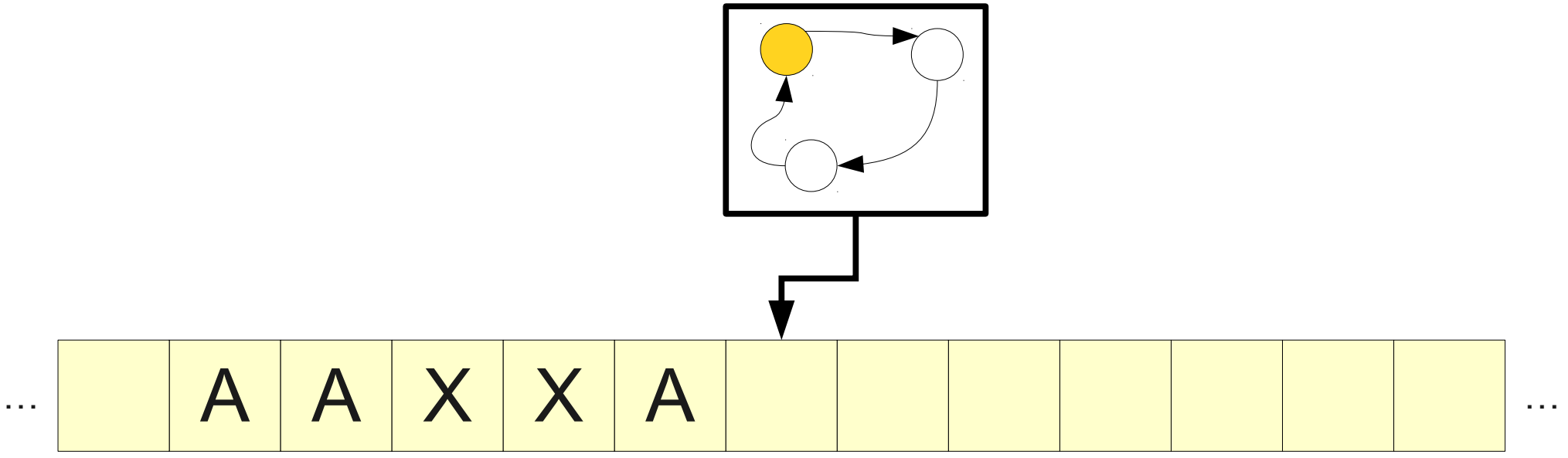
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



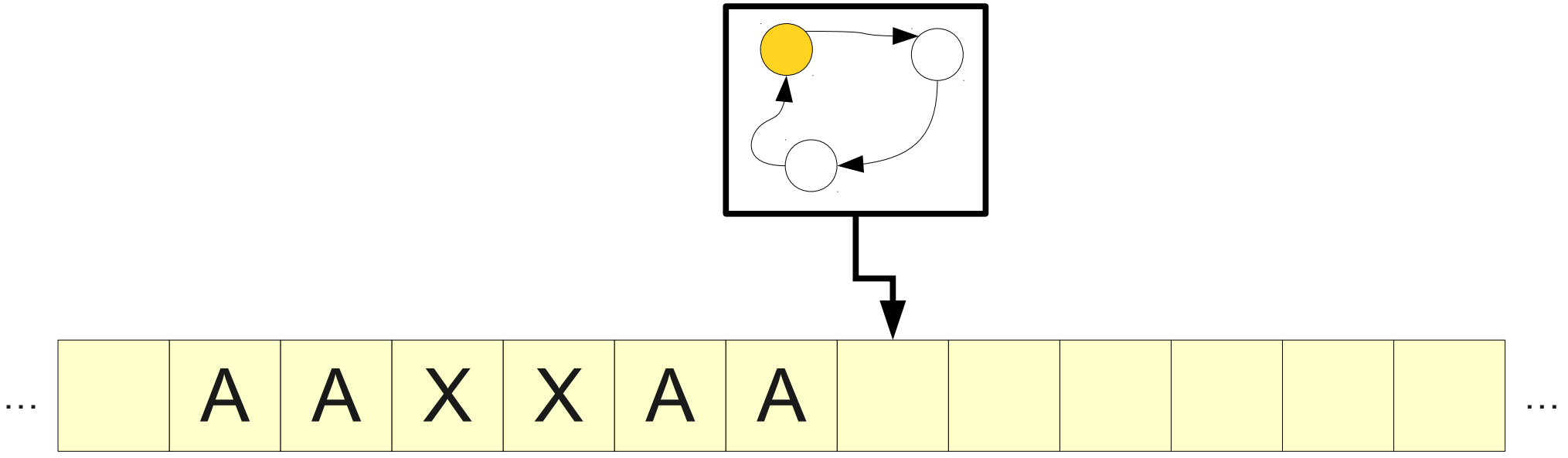
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



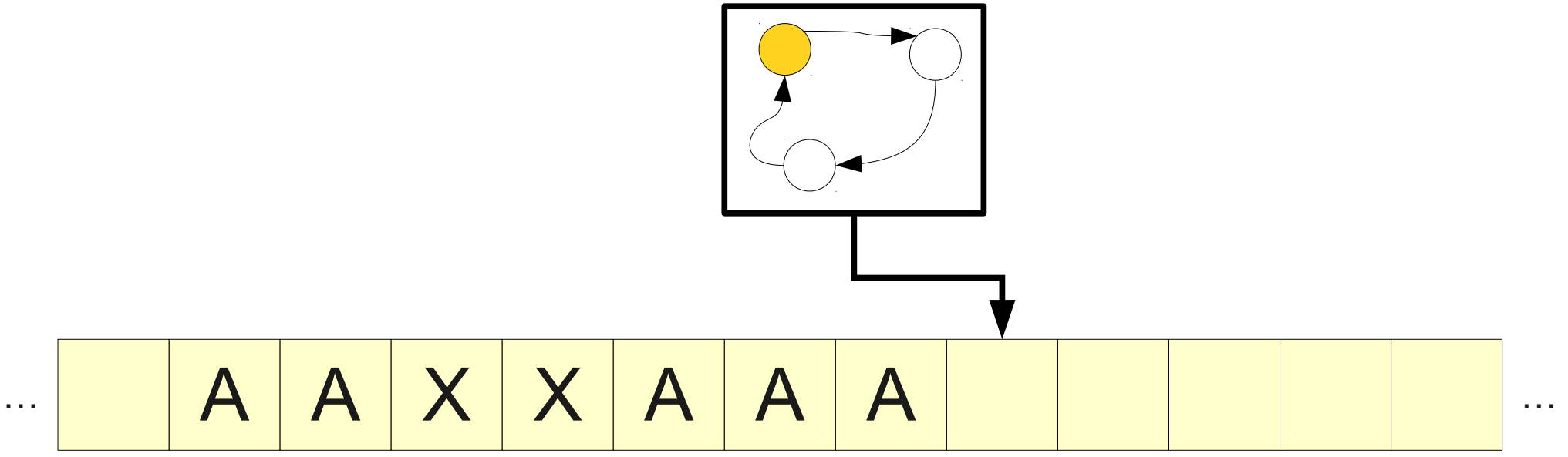
A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



A Better Memory Device

- A **Turing machine** is a finite automaton equipped with an **infinite tape** as its memory.
- The tape begins with the input to the machine written on it, surrounded by infinitely many blank cells.
- The machine has a **tape head** that can read and write a single memory cell at a time.



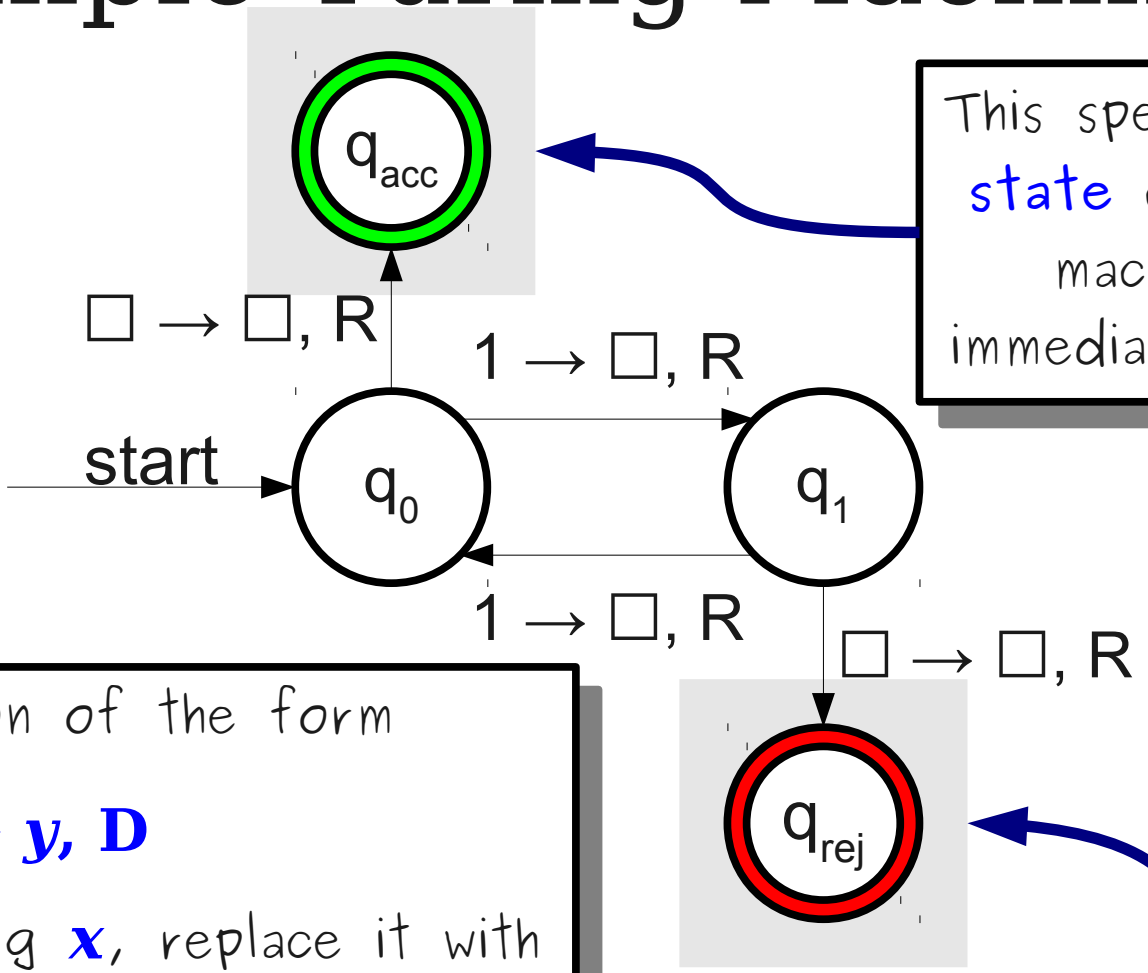
The Turing Machine

- A Turing machine consists of three parts:
  - A **finite-state control** that issues commands,
  - an **infinite tape** for input and scratch space, and
  - a **tape head** that can read and write a single tape cell.
- At each step, the Turing machine
  - writes a symbol to the tape cell under the tape head,
  - changes state, and
  - moves the tape head to the left or to the right.

Input and Tape Alphabets

- A Turing machine has two alphabets:
  - An **input alphabet** Σ. All input strings are written in the input alphabet.
  - A **tape alphabet** Γ, where $\Sigma \subseteq \Gamma$. The tape alphabet contains all symbols that can be written onto the tape.
- The tape alphabet Γ can contain any number of symbols, but always contains at least one **blank symbol**, denoted $\square$. You are guaranteed $\square \notin \Sigma$.
- At startup, the Turing machine begins with an infinite tape of $\square$ symbols with the input written at some location. The tape head is positioned at the start of the input.

A Simple Turing Machine



This special **accept state** causes the machine to immediately accept.

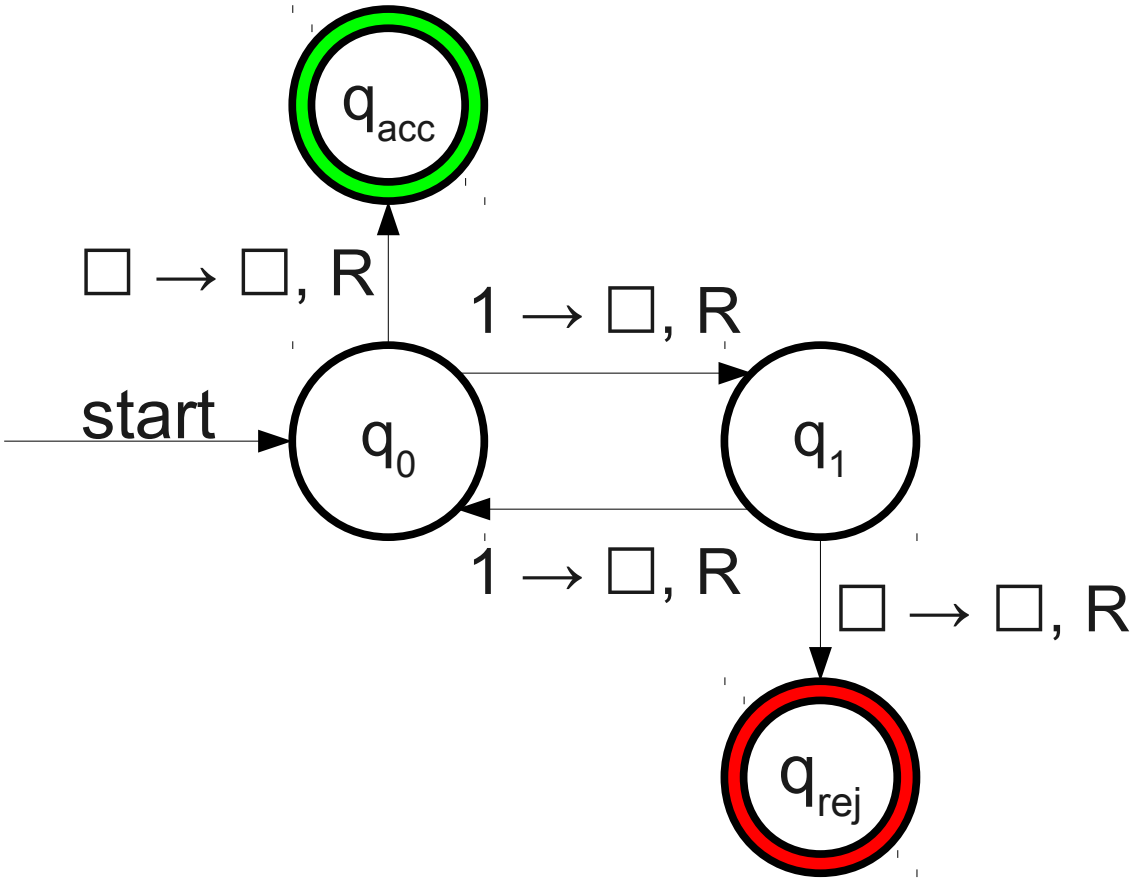
Each transition of the form

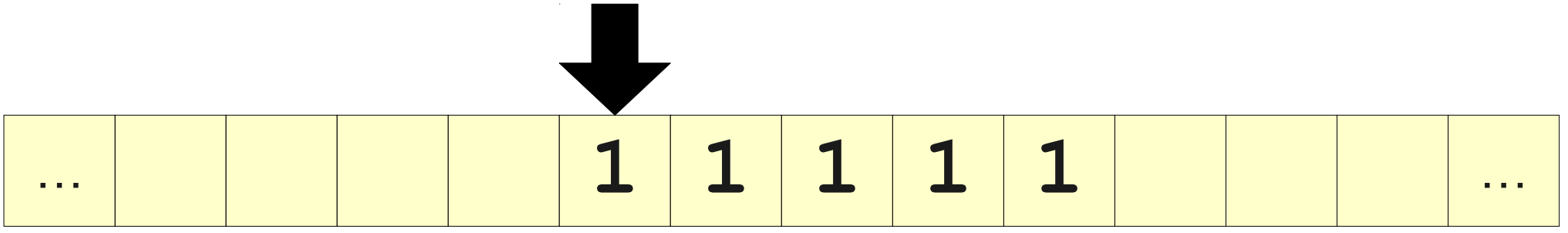
$$x \to y, D$$

means "upon reading x, replace it with symbol y and move the tape head in direction D (which is either **L** or **R**). The symbol □ represents the **blank symbol**.

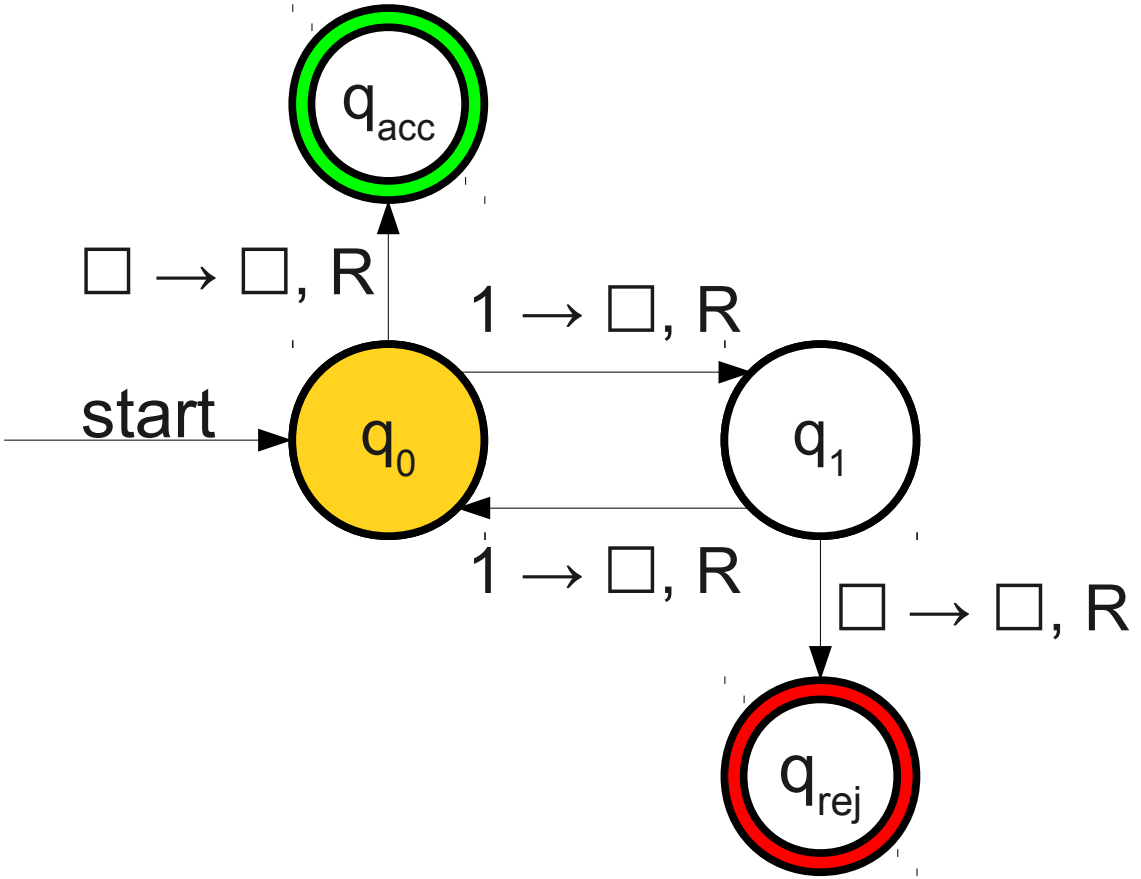
This special **reject state** causes the machine to immediately reject.

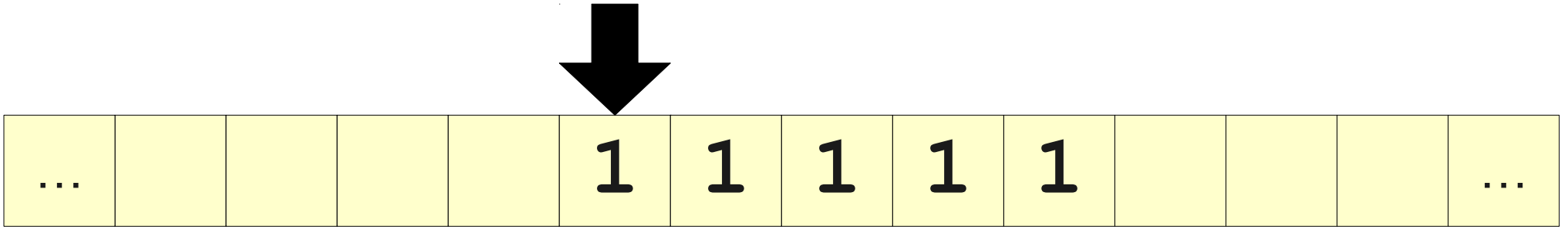
A Simple Turing Machine



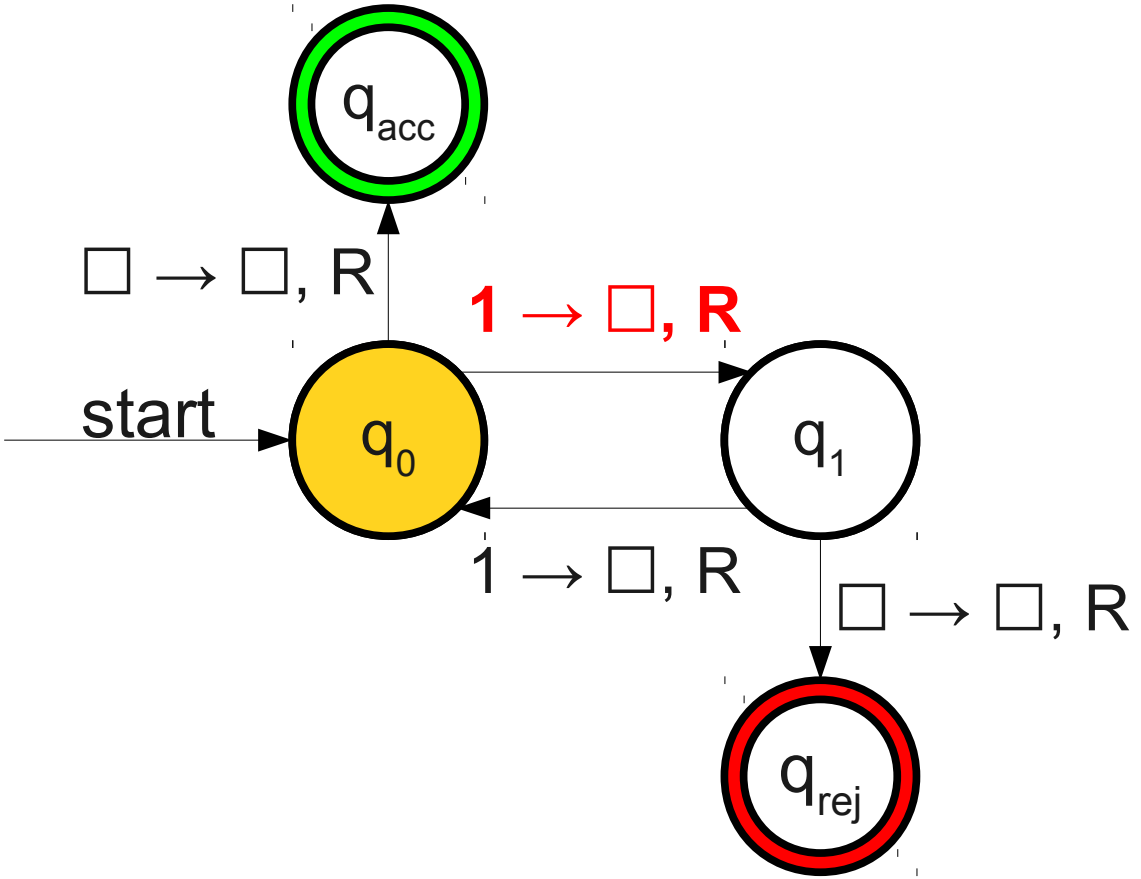


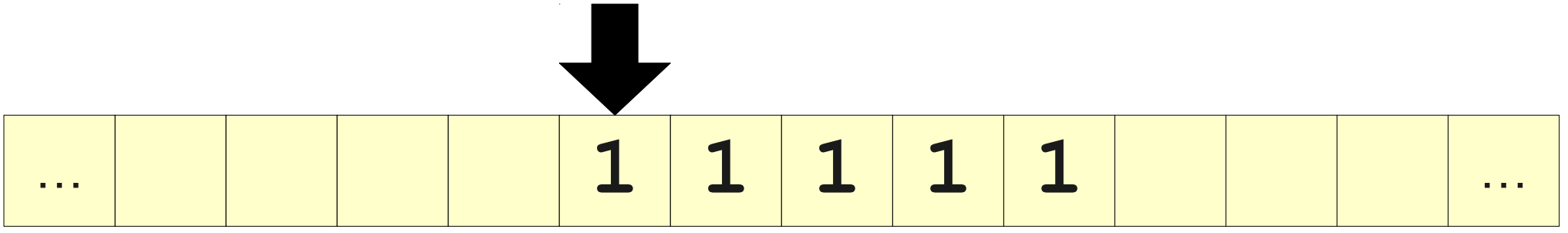
A Simple Turing Machine



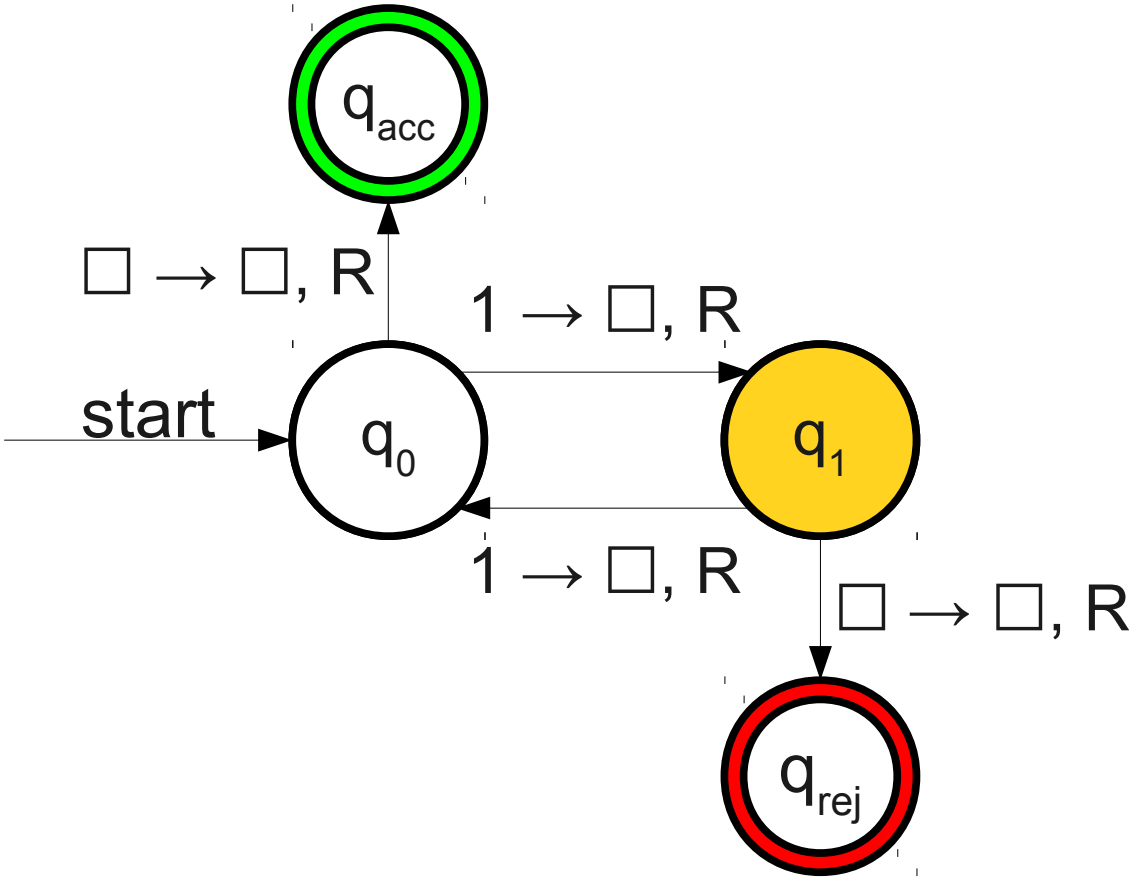


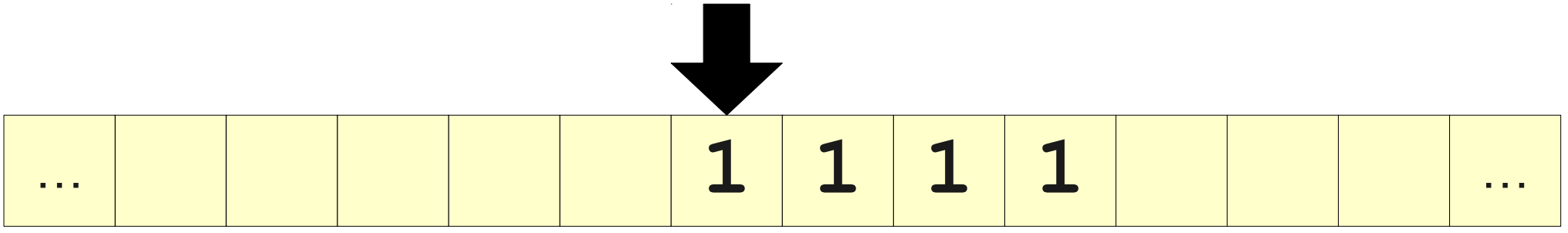
A Simple Turing Machine



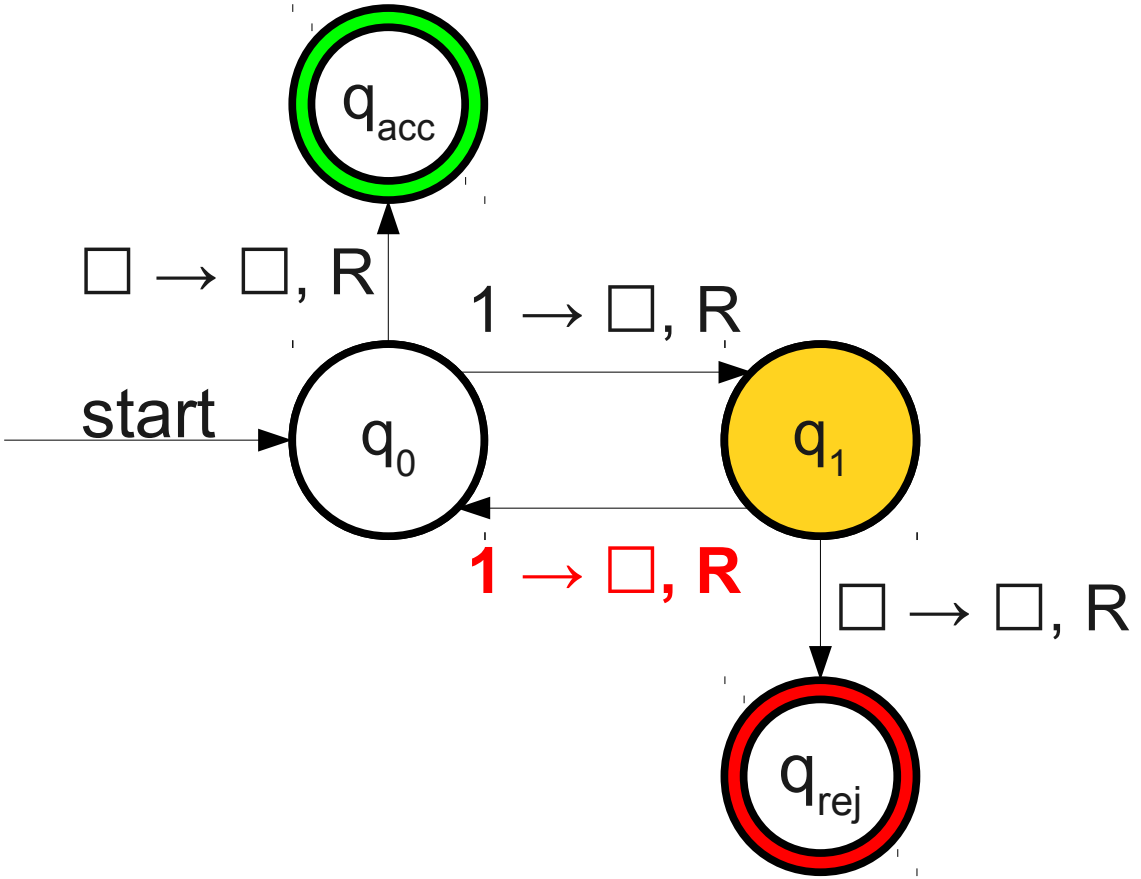


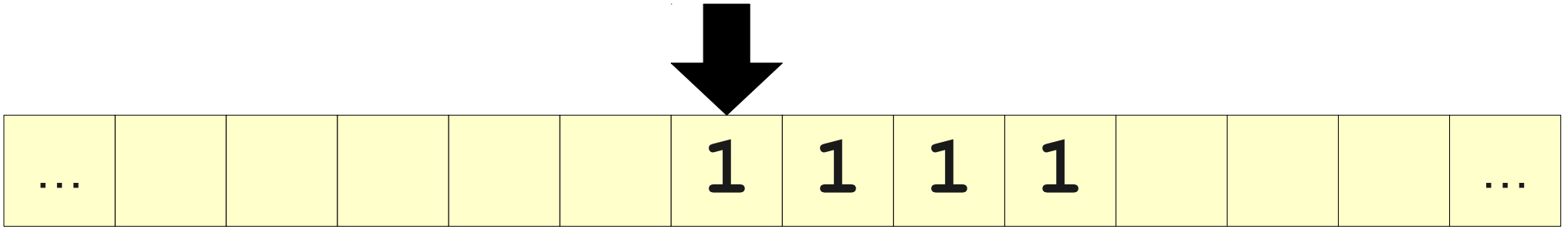
A Simple Turing Machine



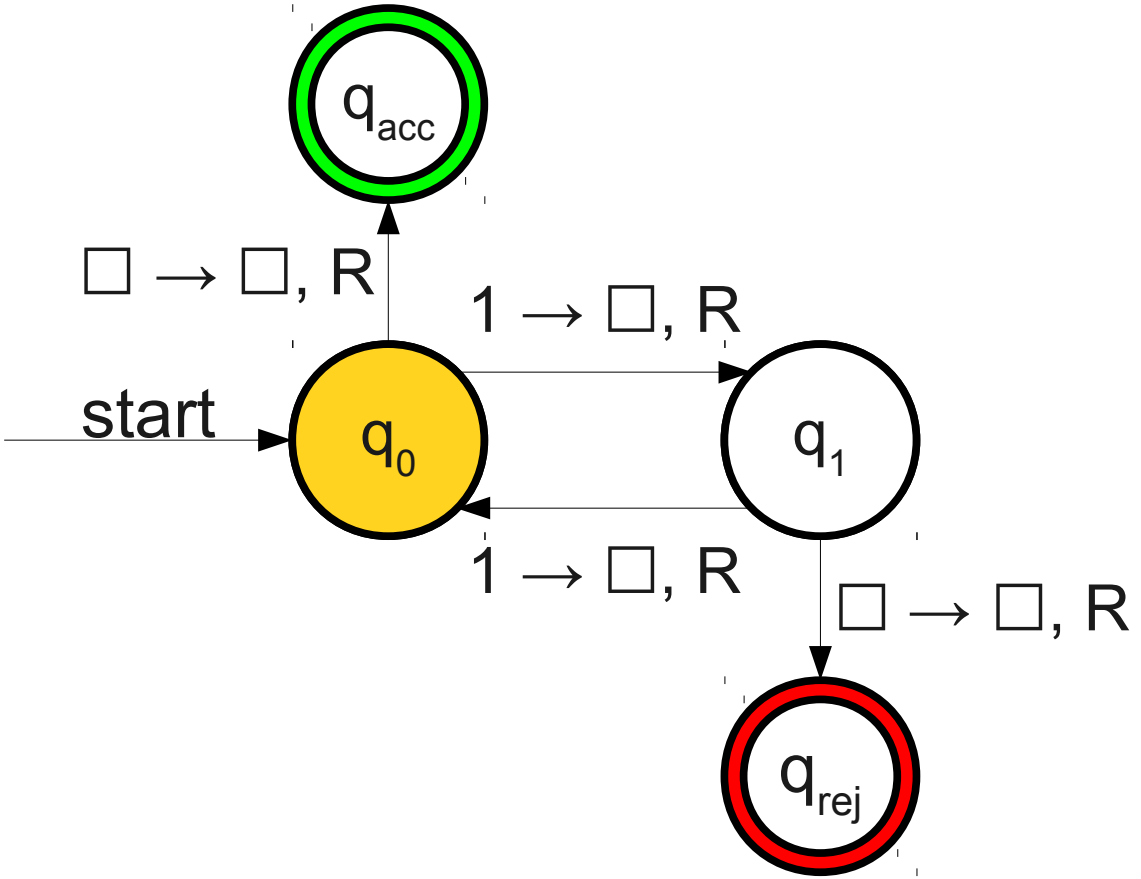


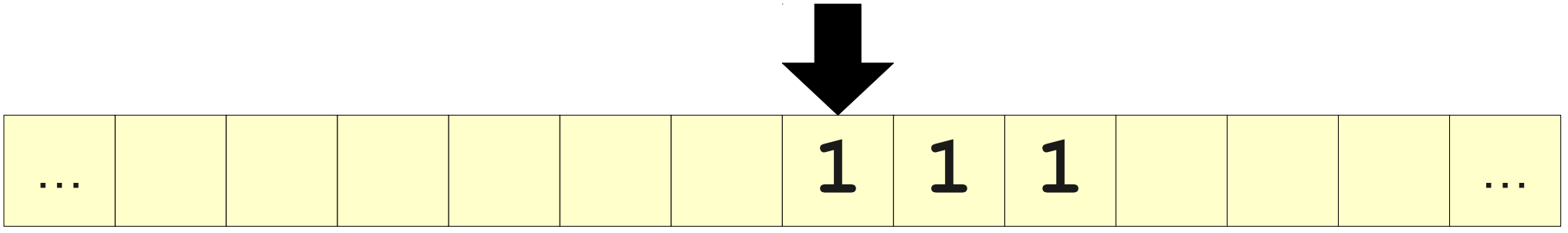
A Simple Turing Machine



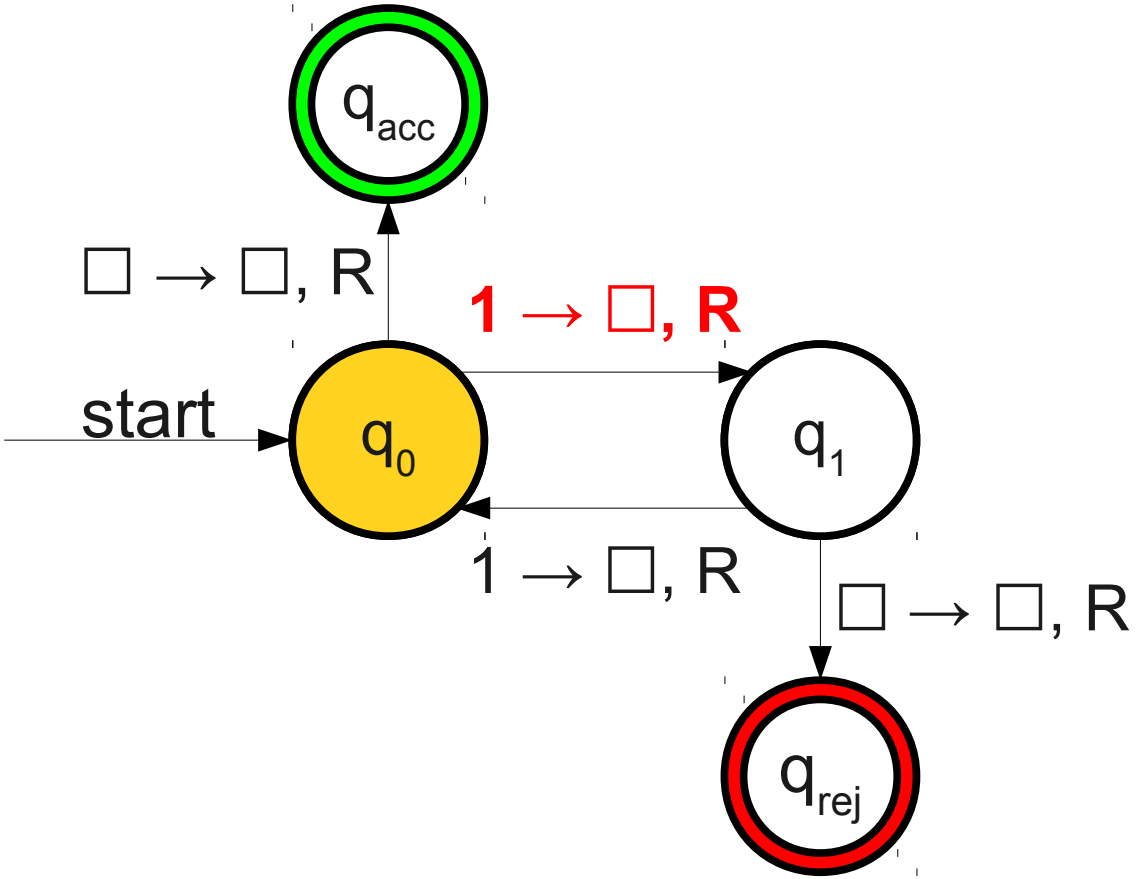


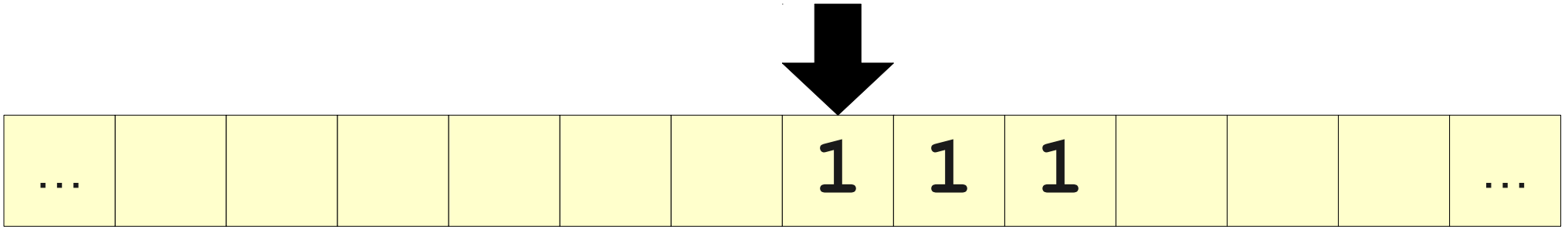
A Simple Turing Machine



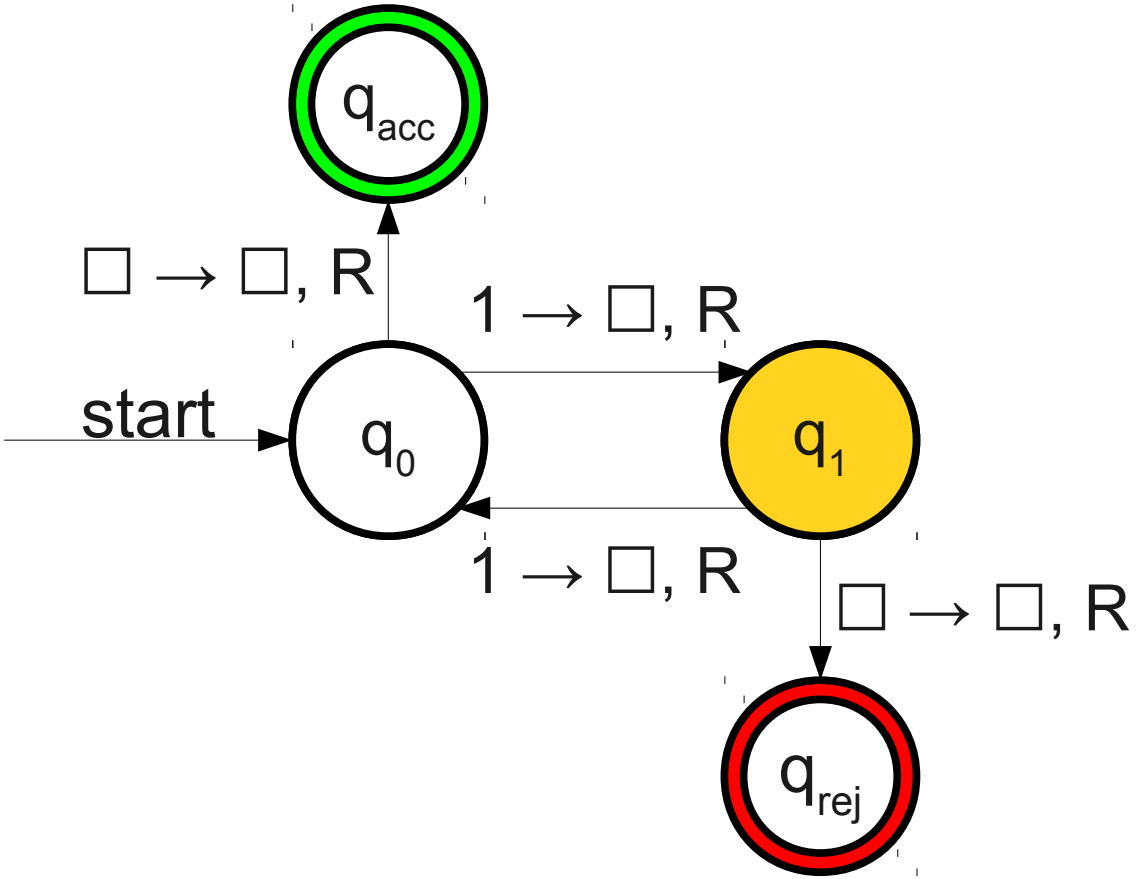


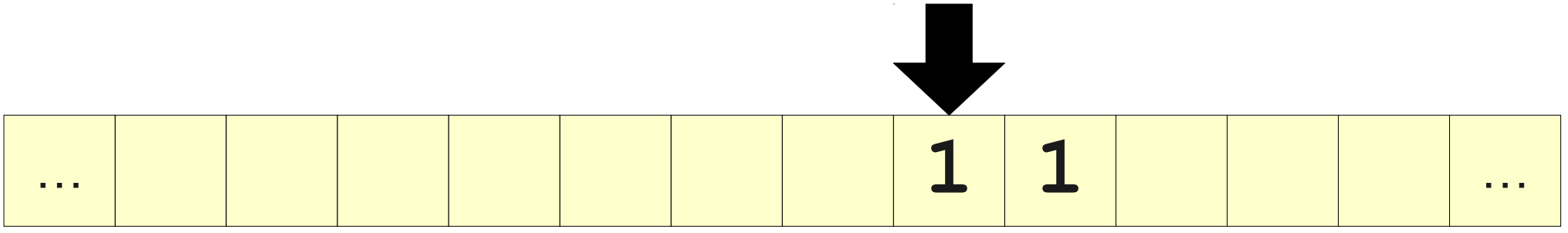
A Simple Turing Machine



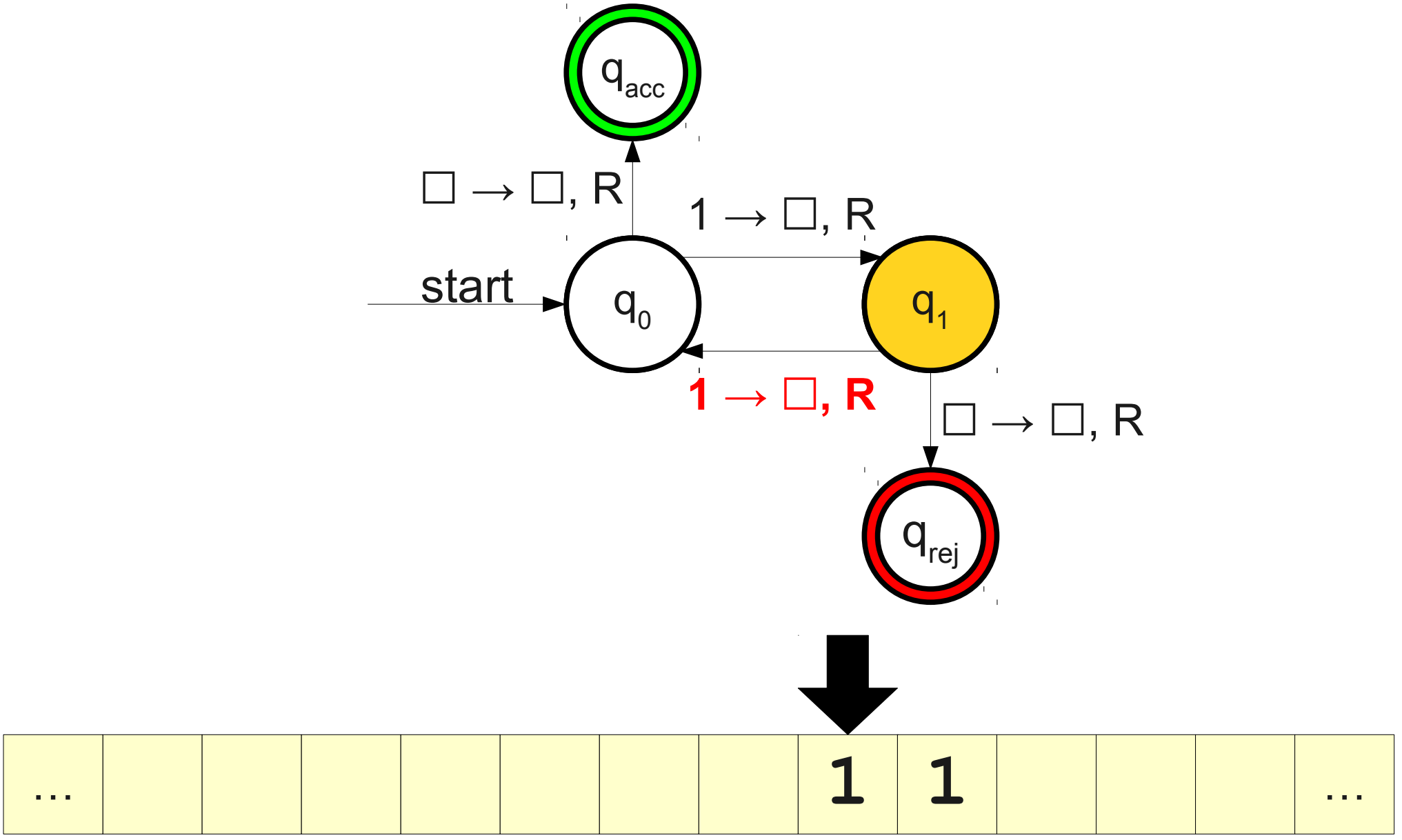


A Simple Turing Machine

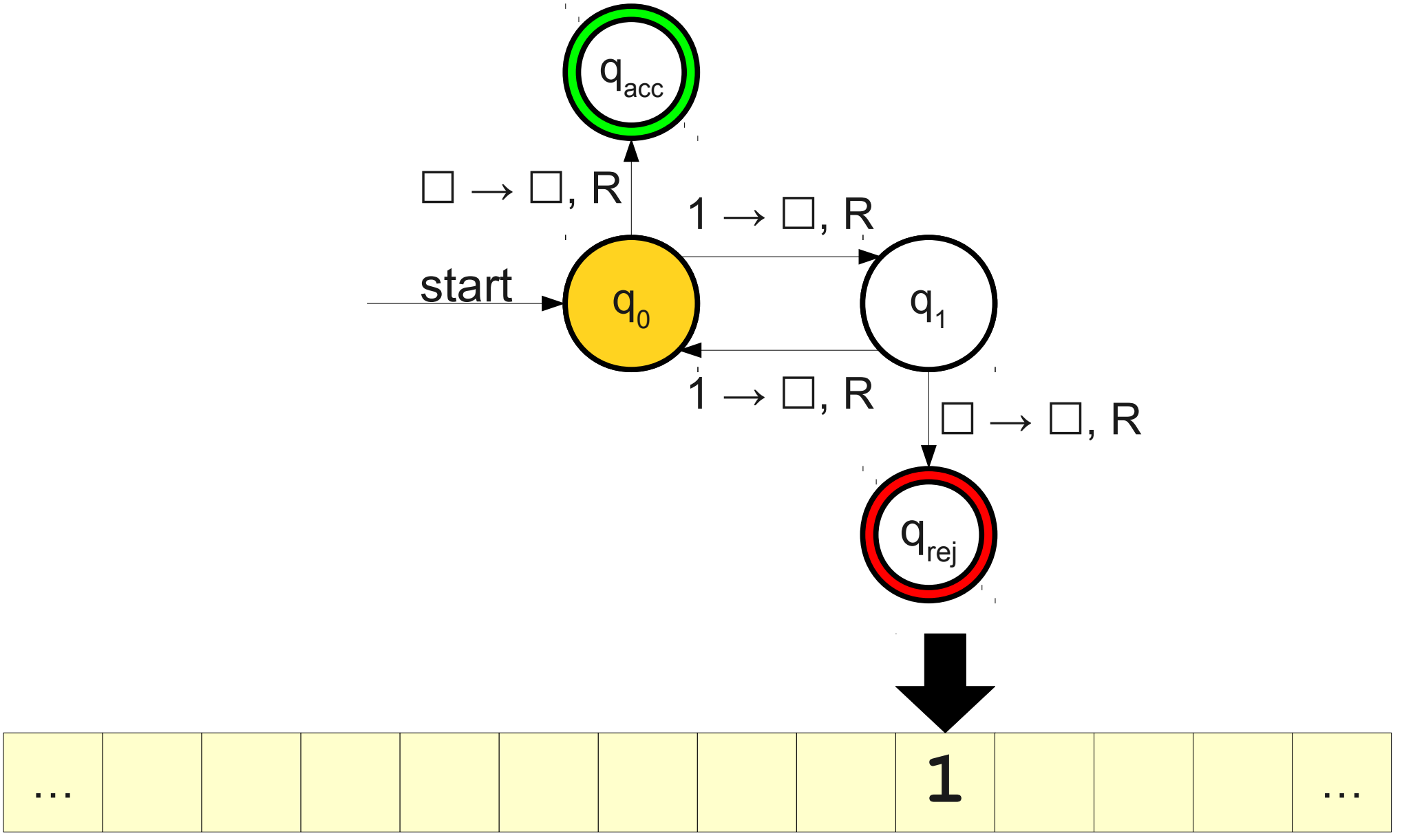




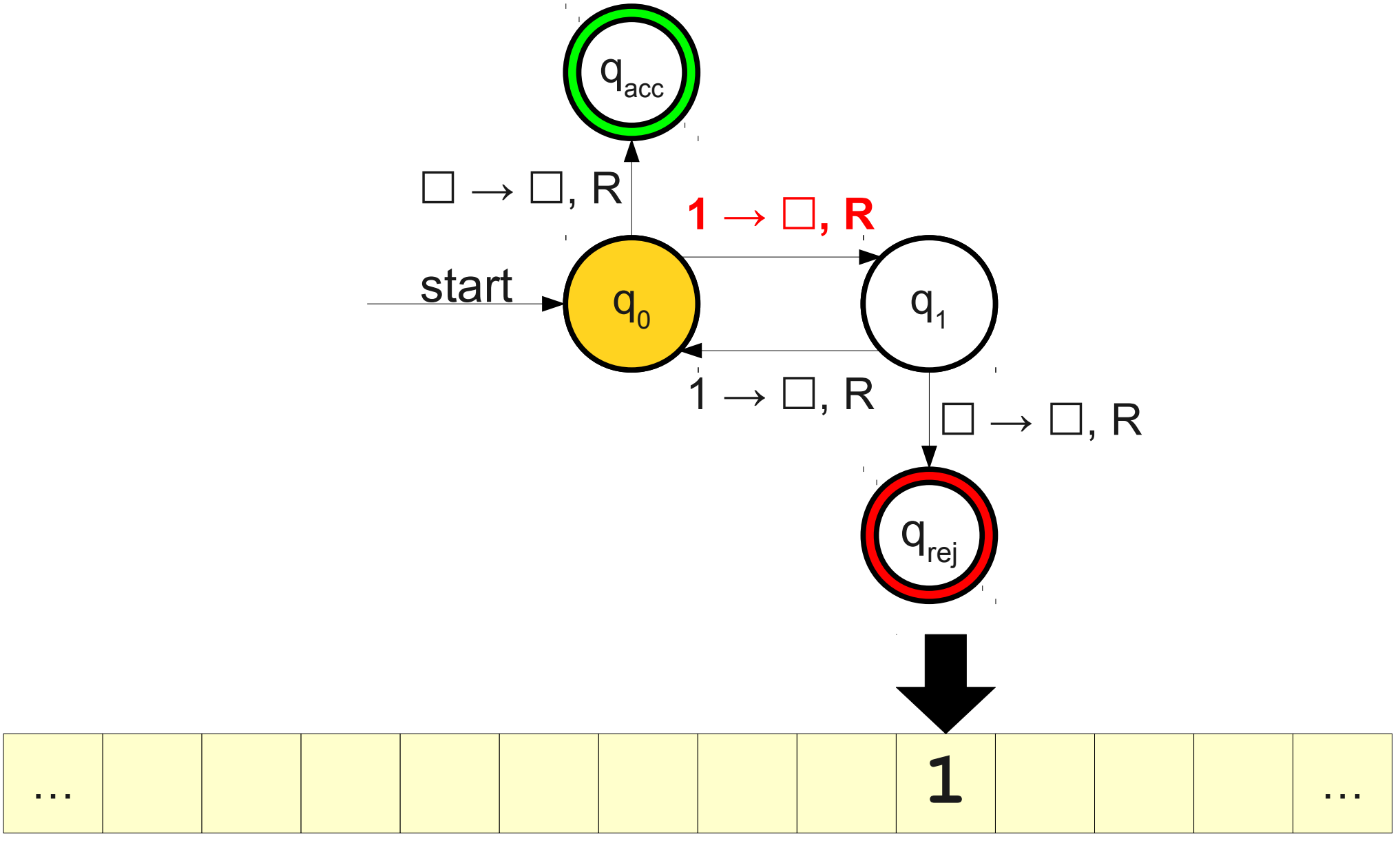
A Simple Turing Machine



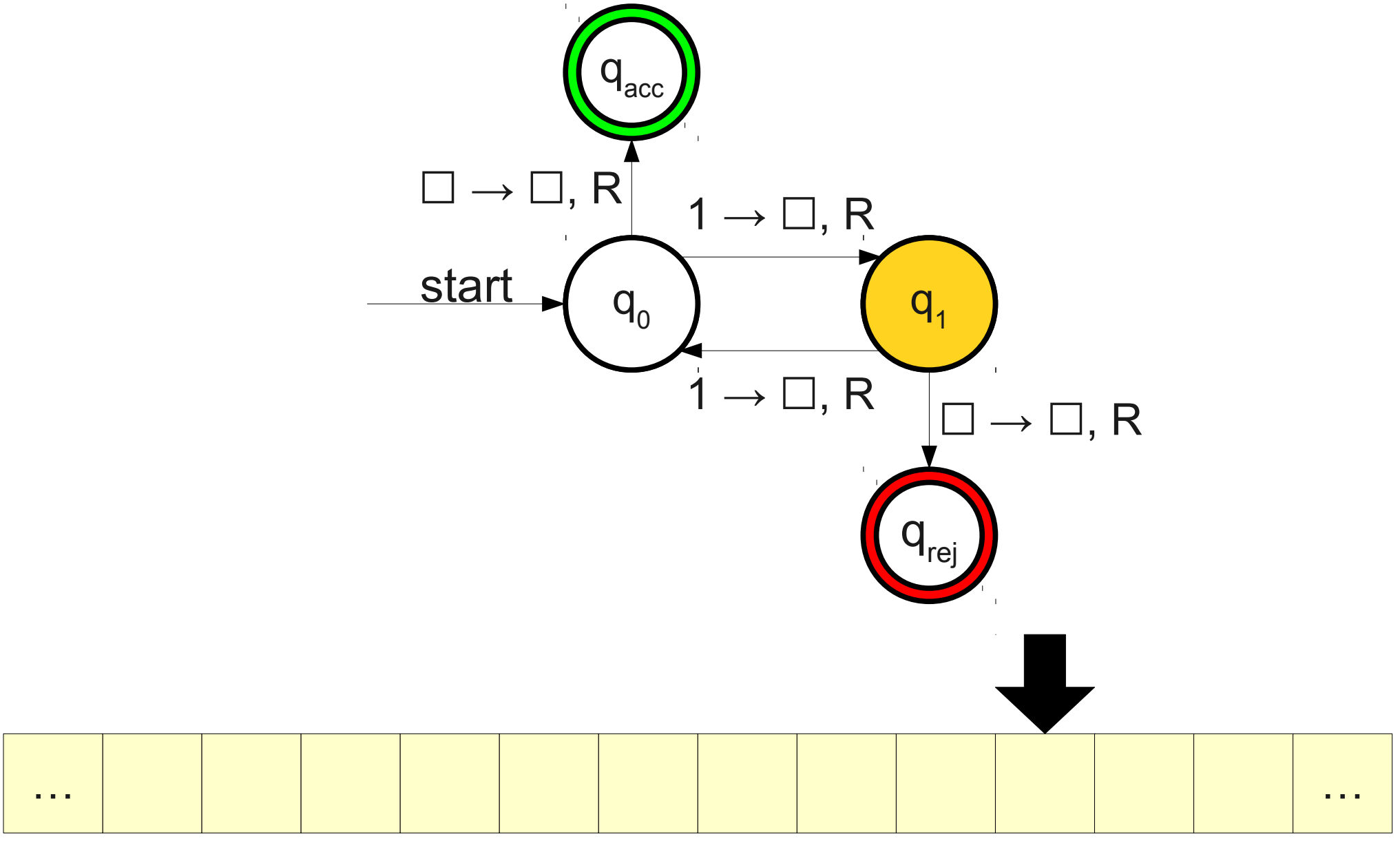
A Simple Turing Machine



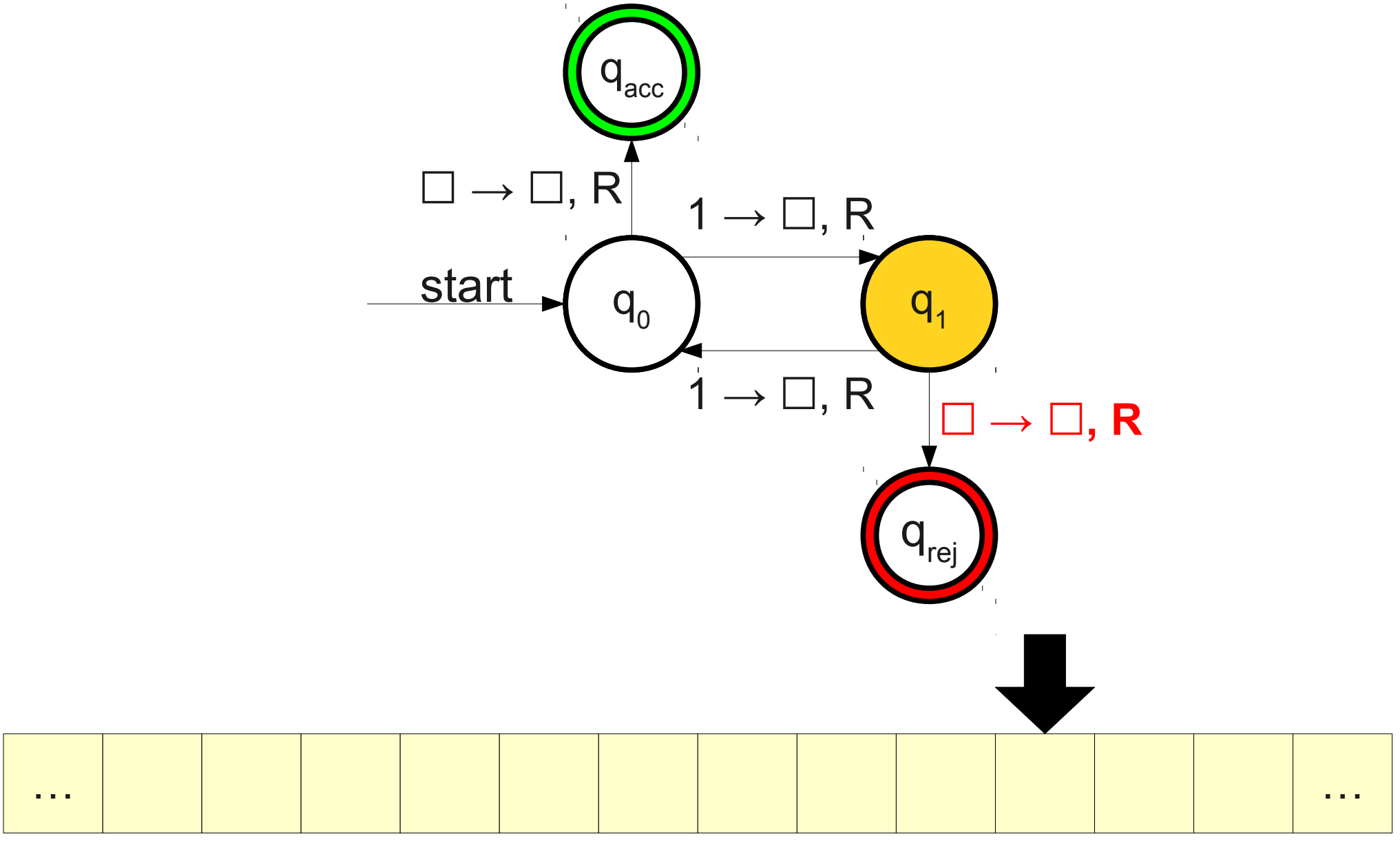
A Simple Turing Machine



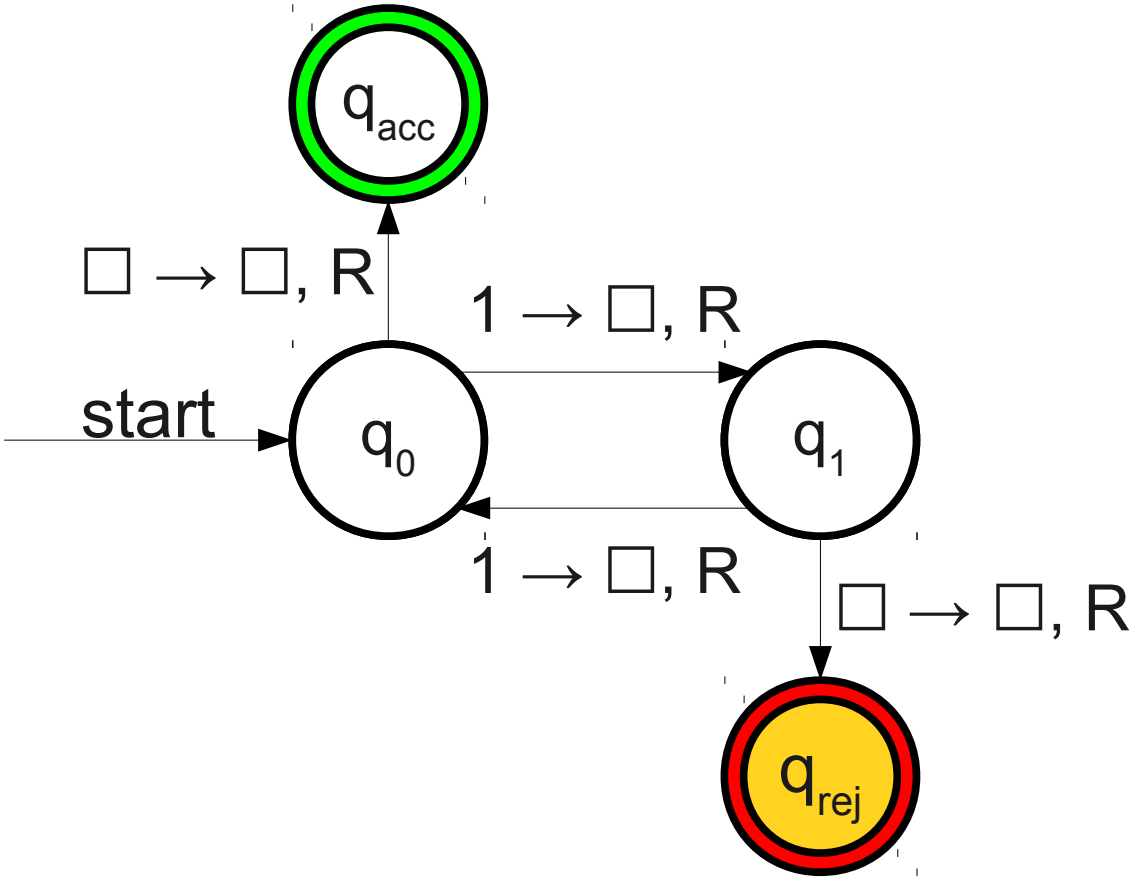
A Simple Turing Machine

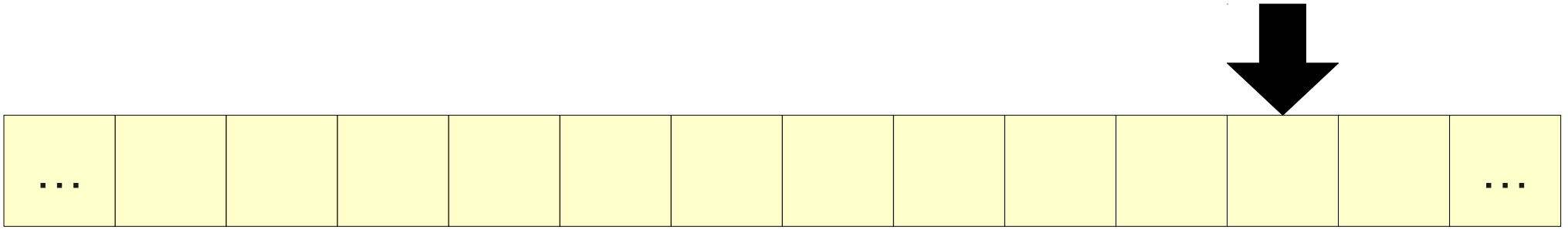


A Simple Turing Machine

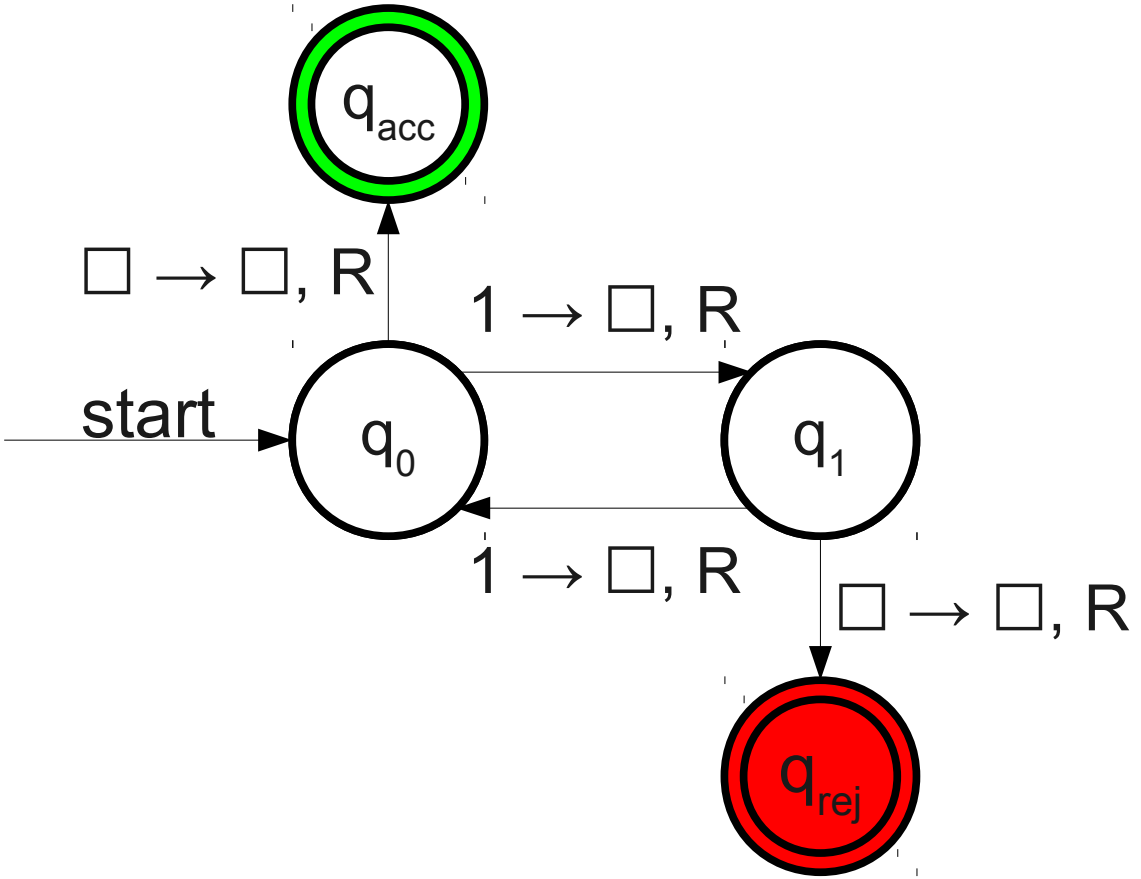


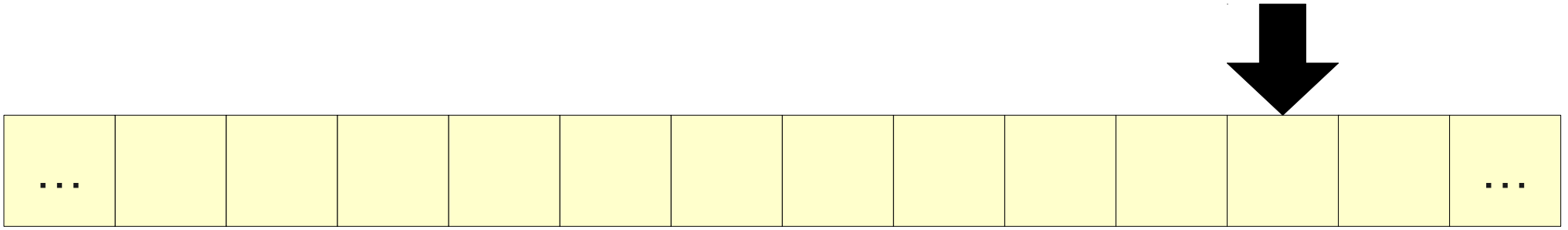
A Simple Turing Machine



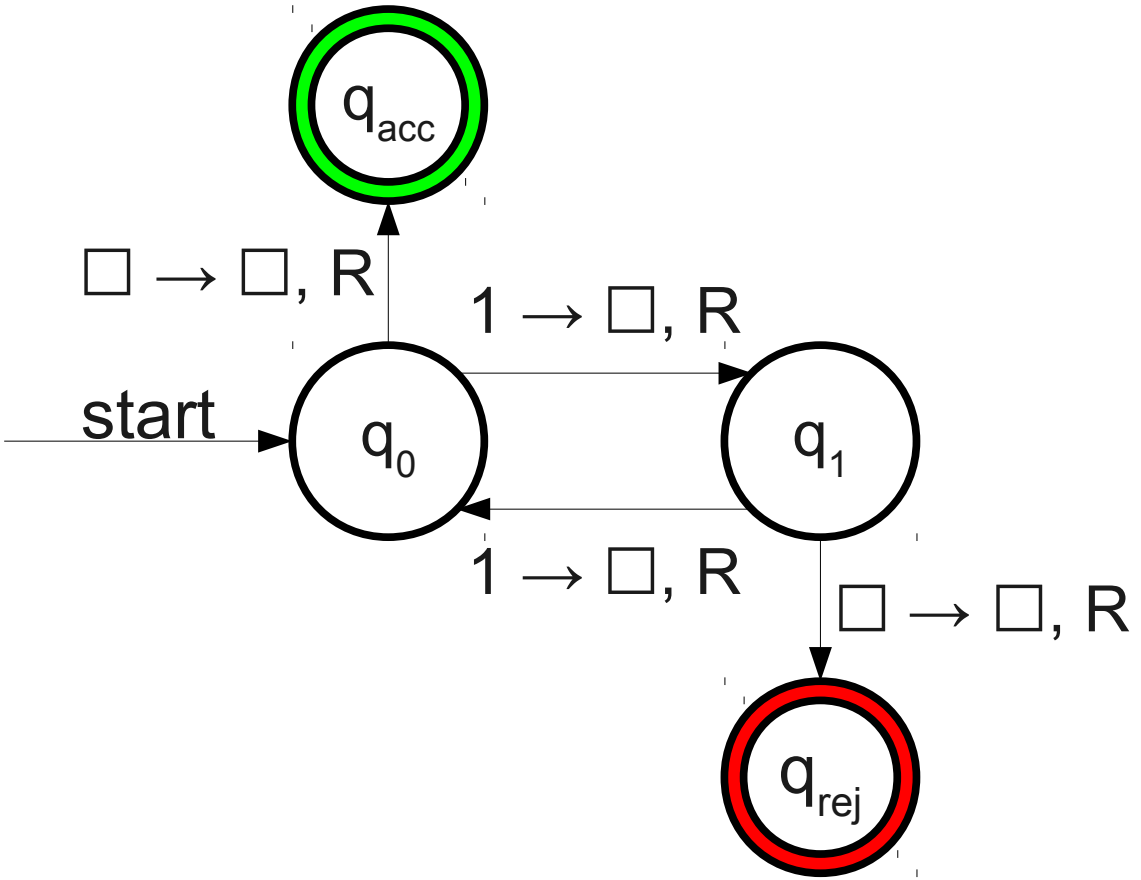


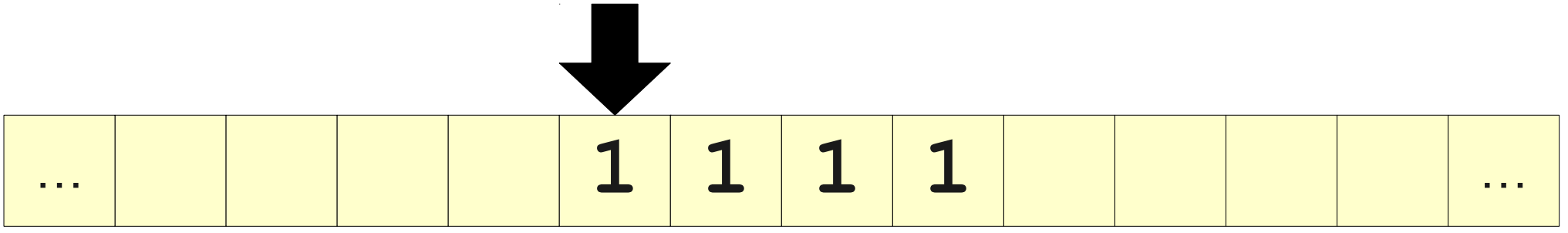
A Simple Turing Machine



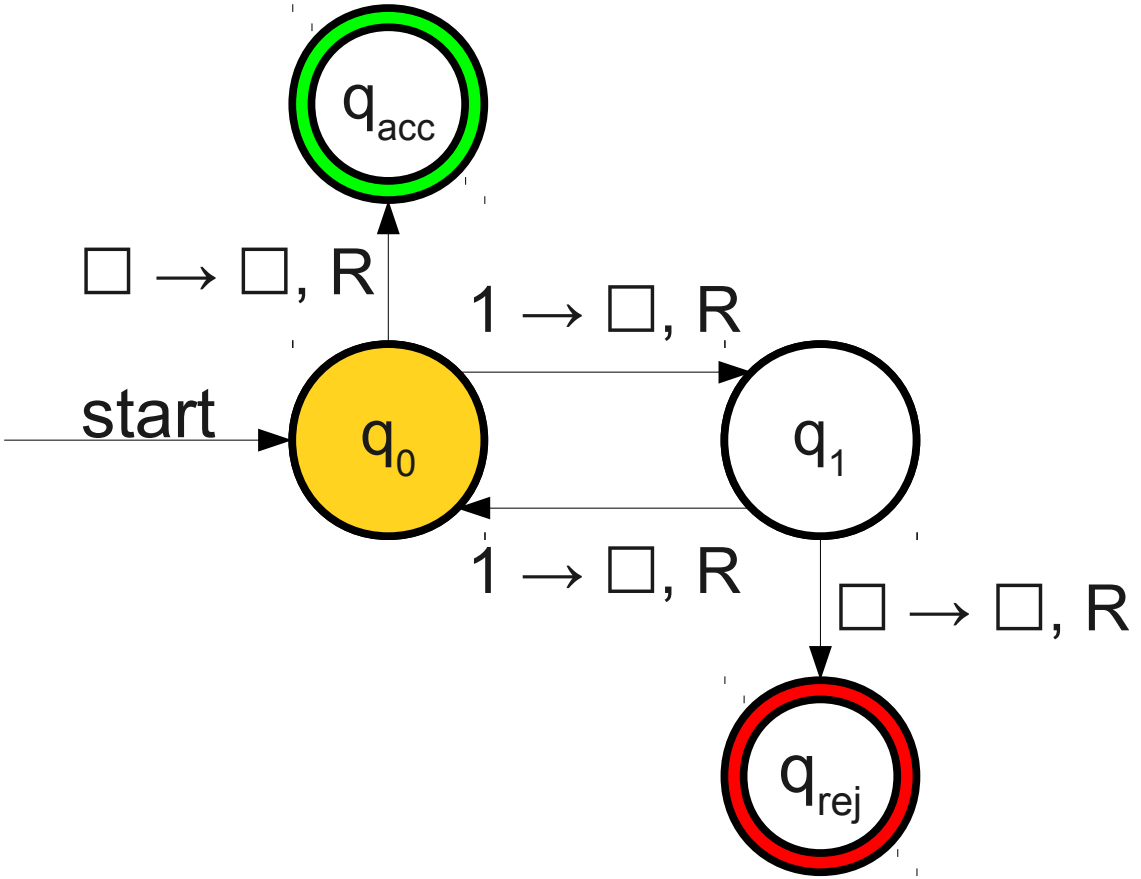


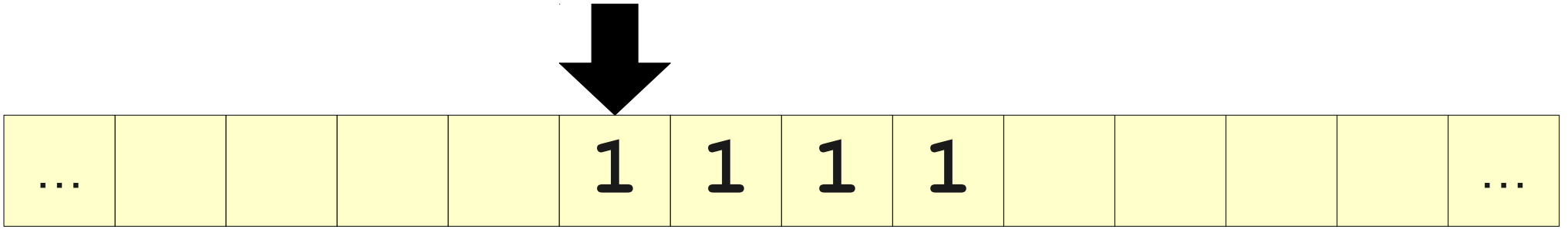
A Simple Turing Machine



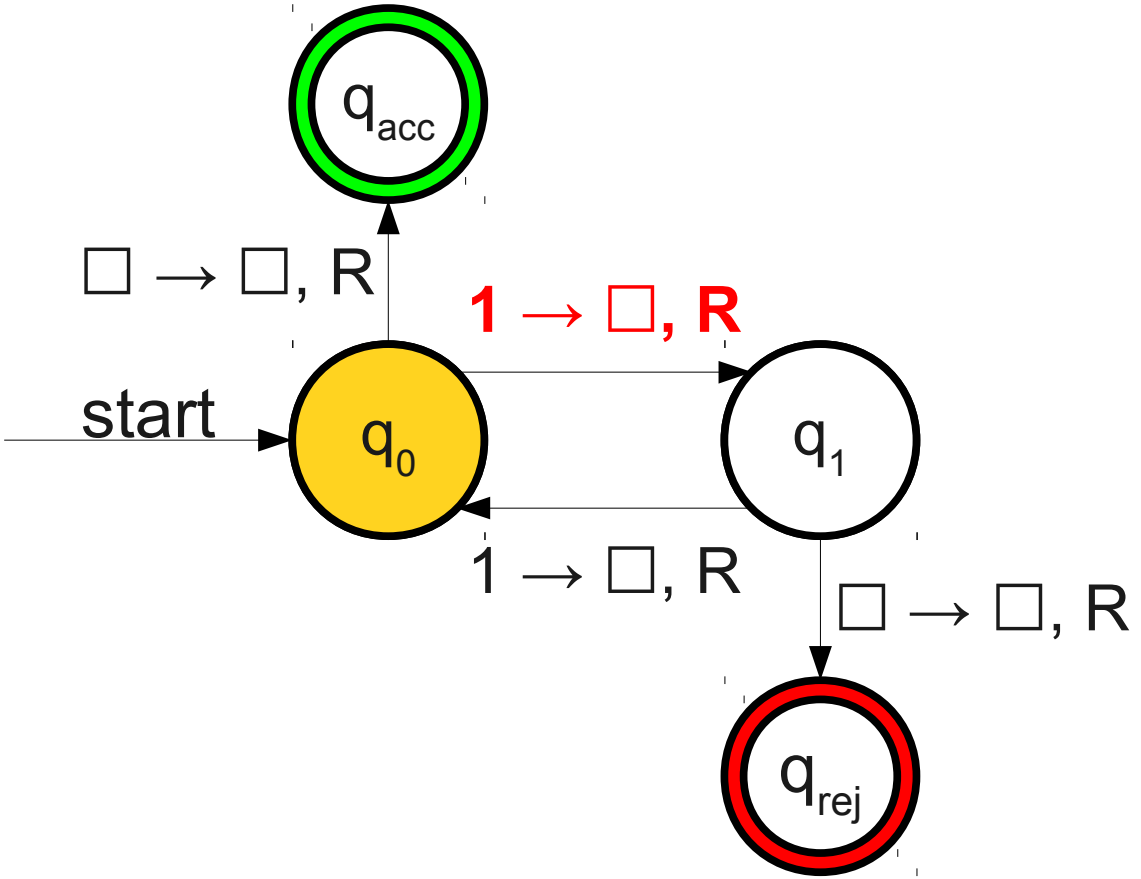


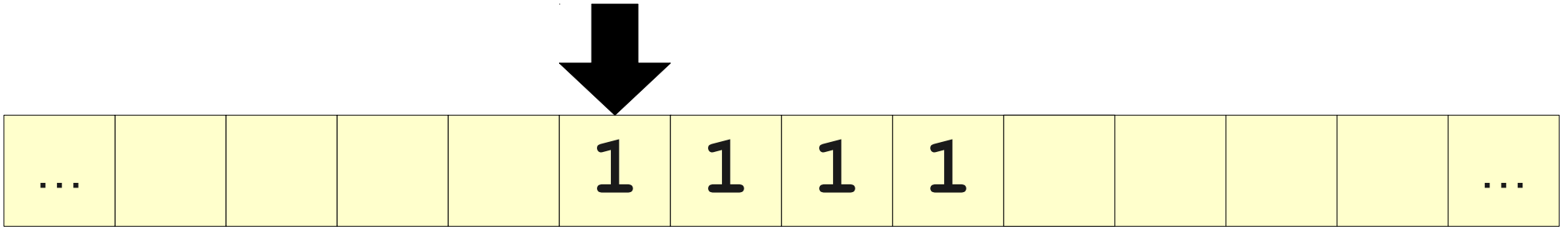
A Simple Turing Machine



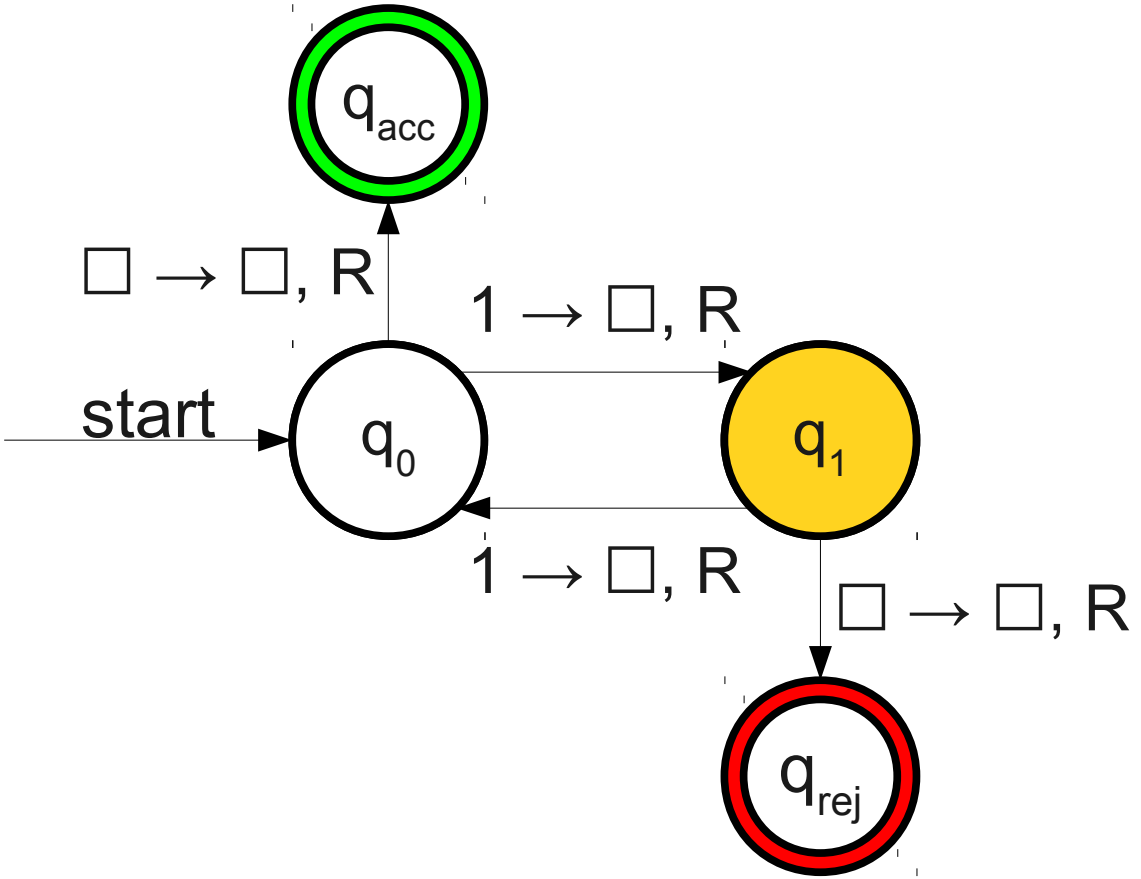


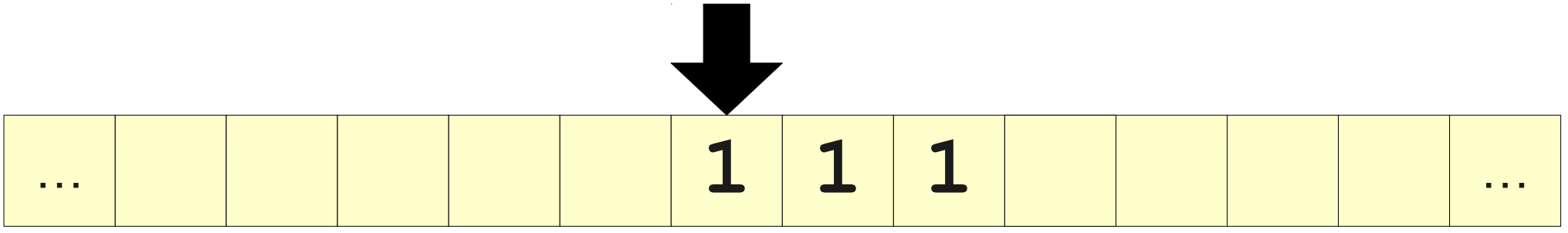
A Simple Turing Machine



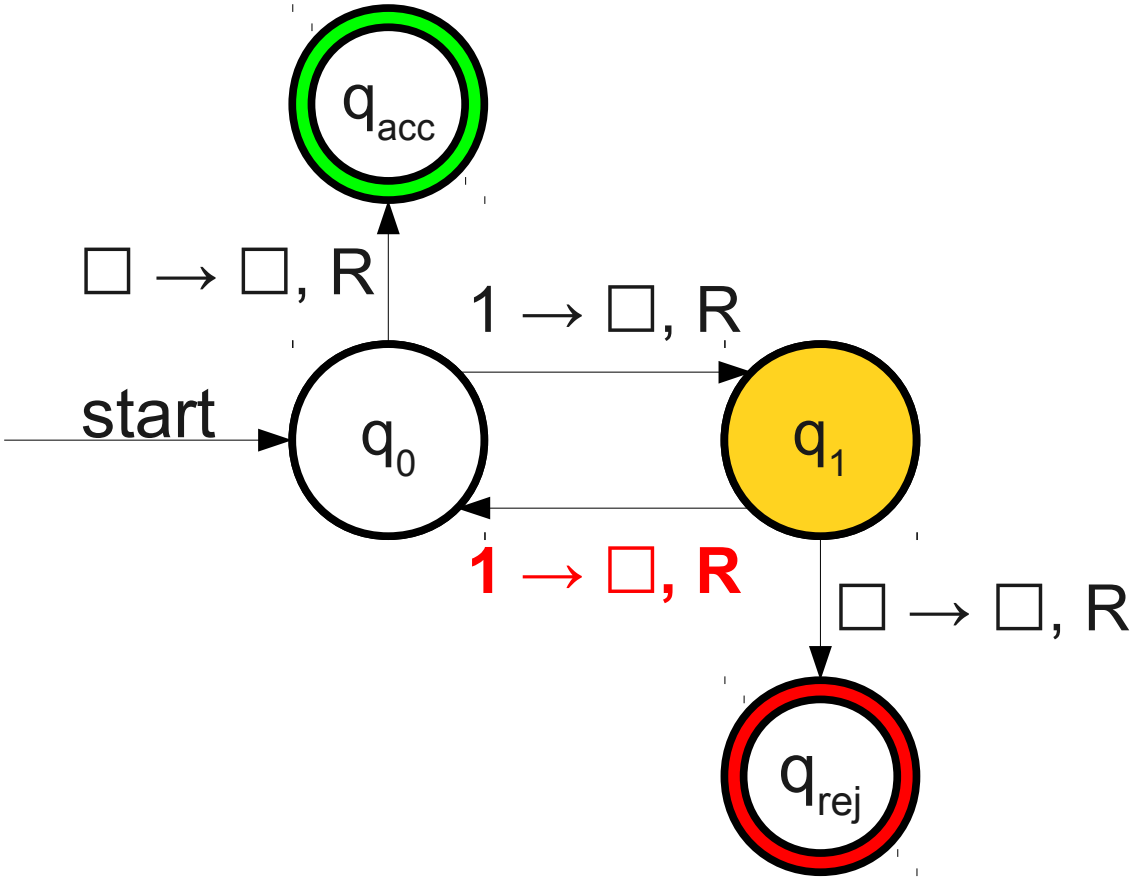


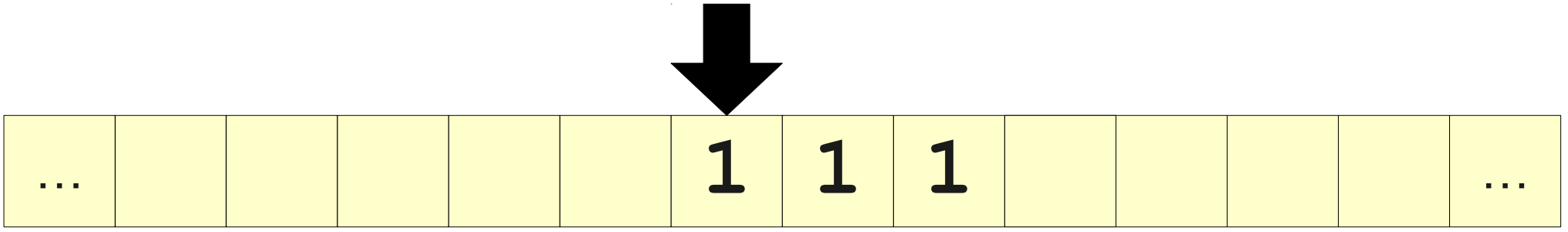
A Simple Turing Machine



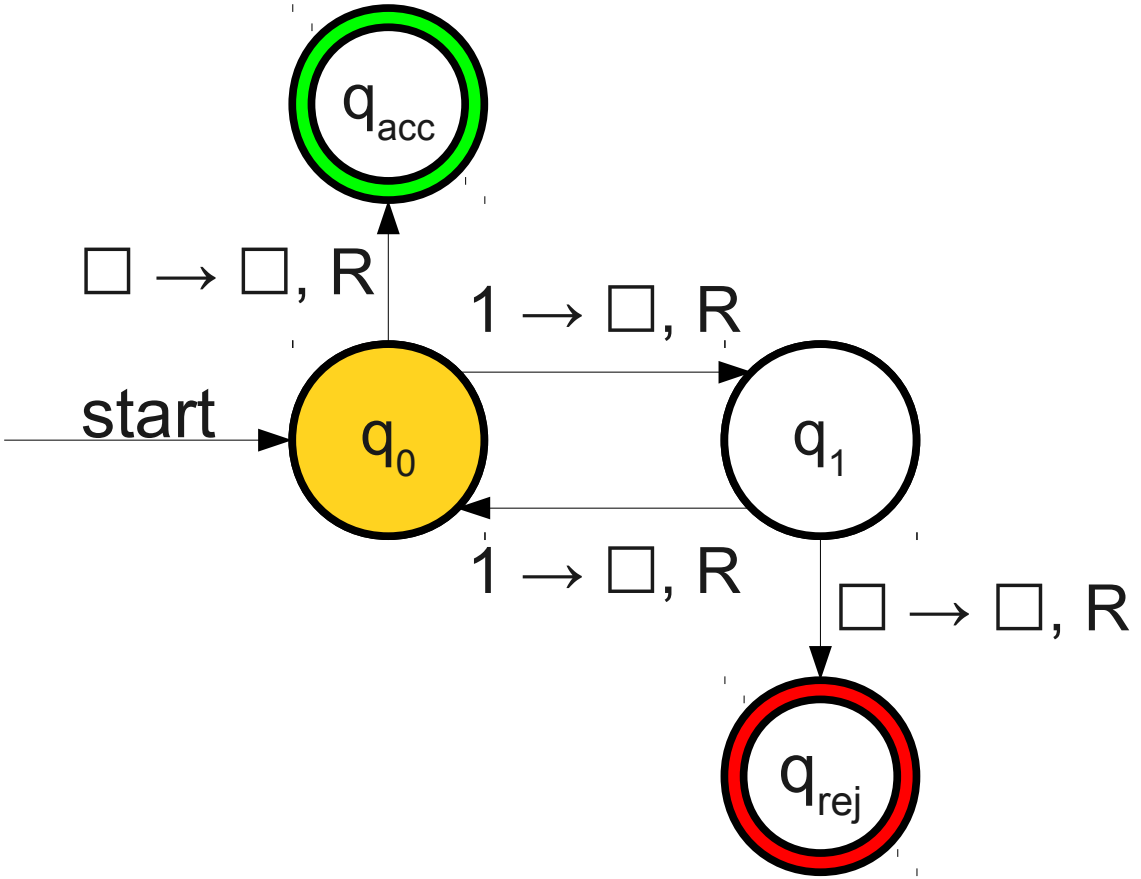


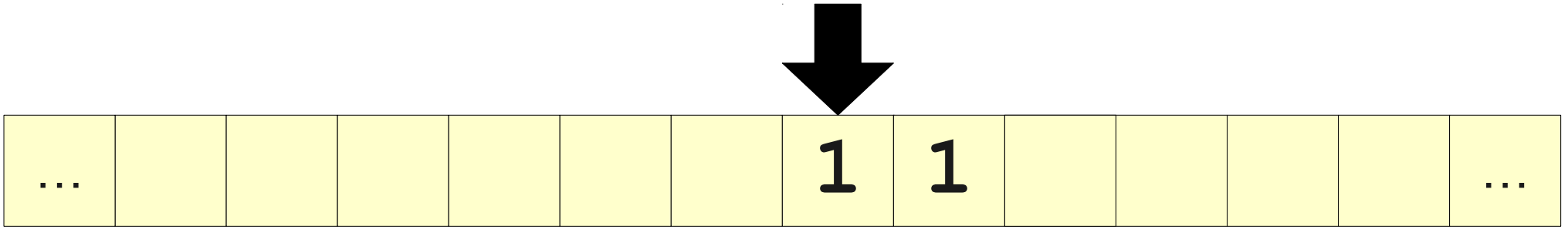
A Simple Turing Machine



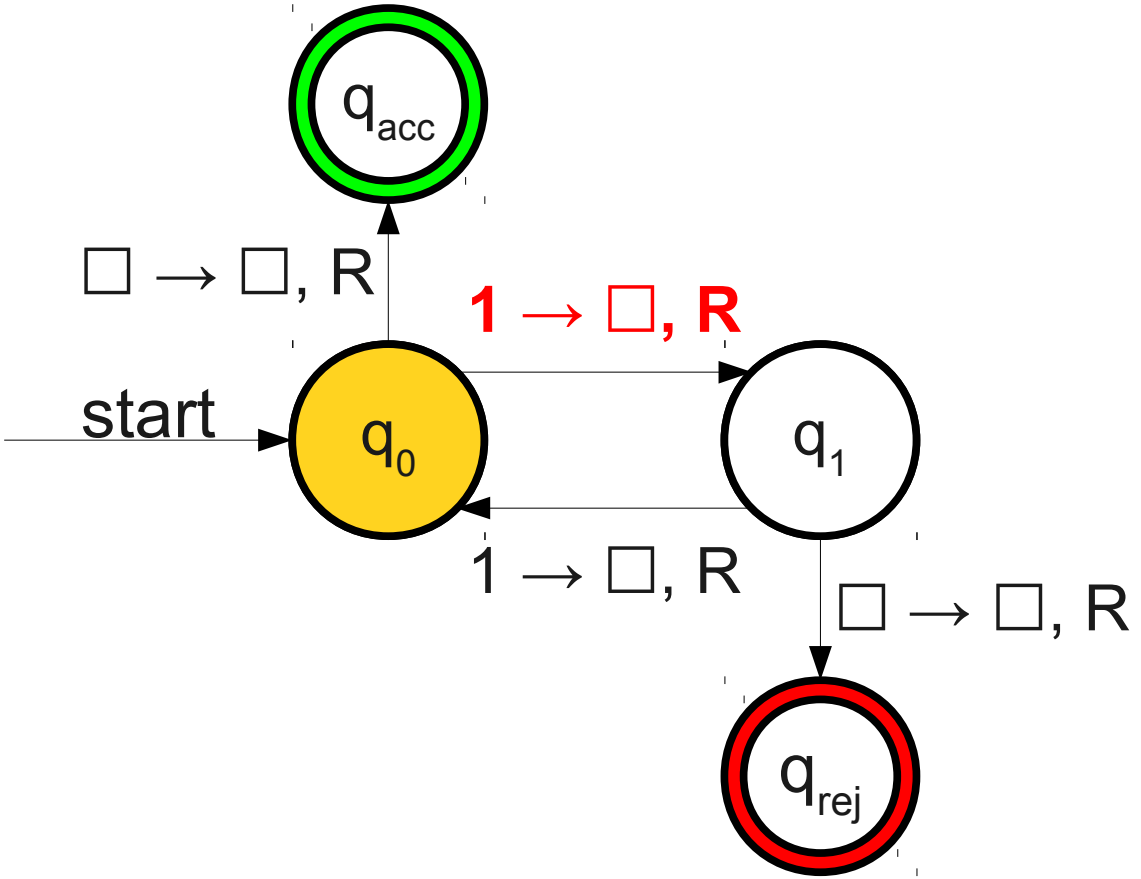


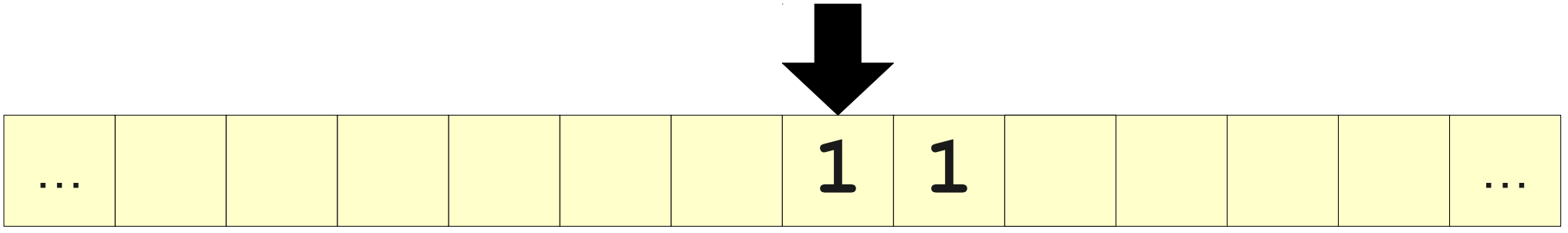
A Simple Turing Machine



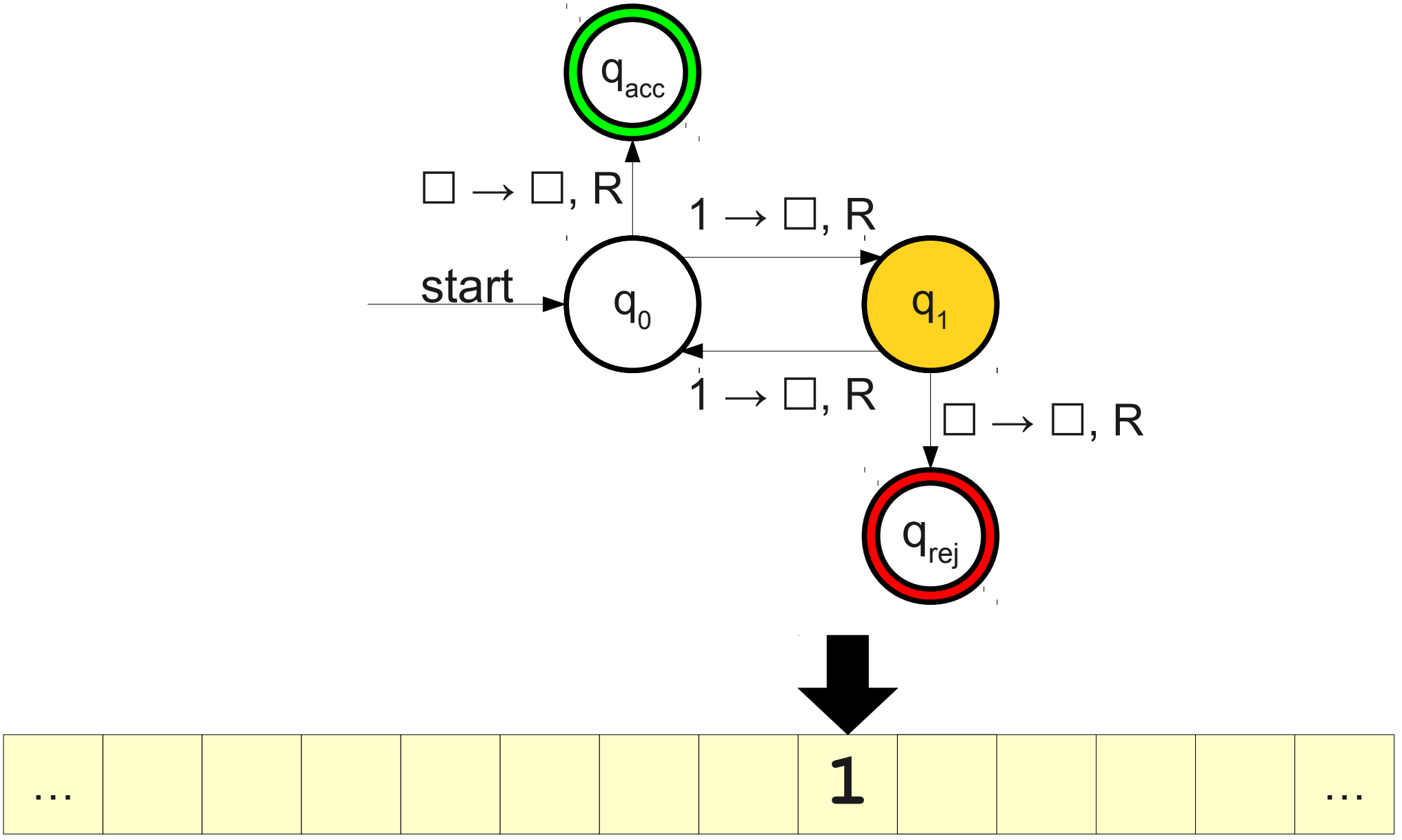


A Simple Turing Machine

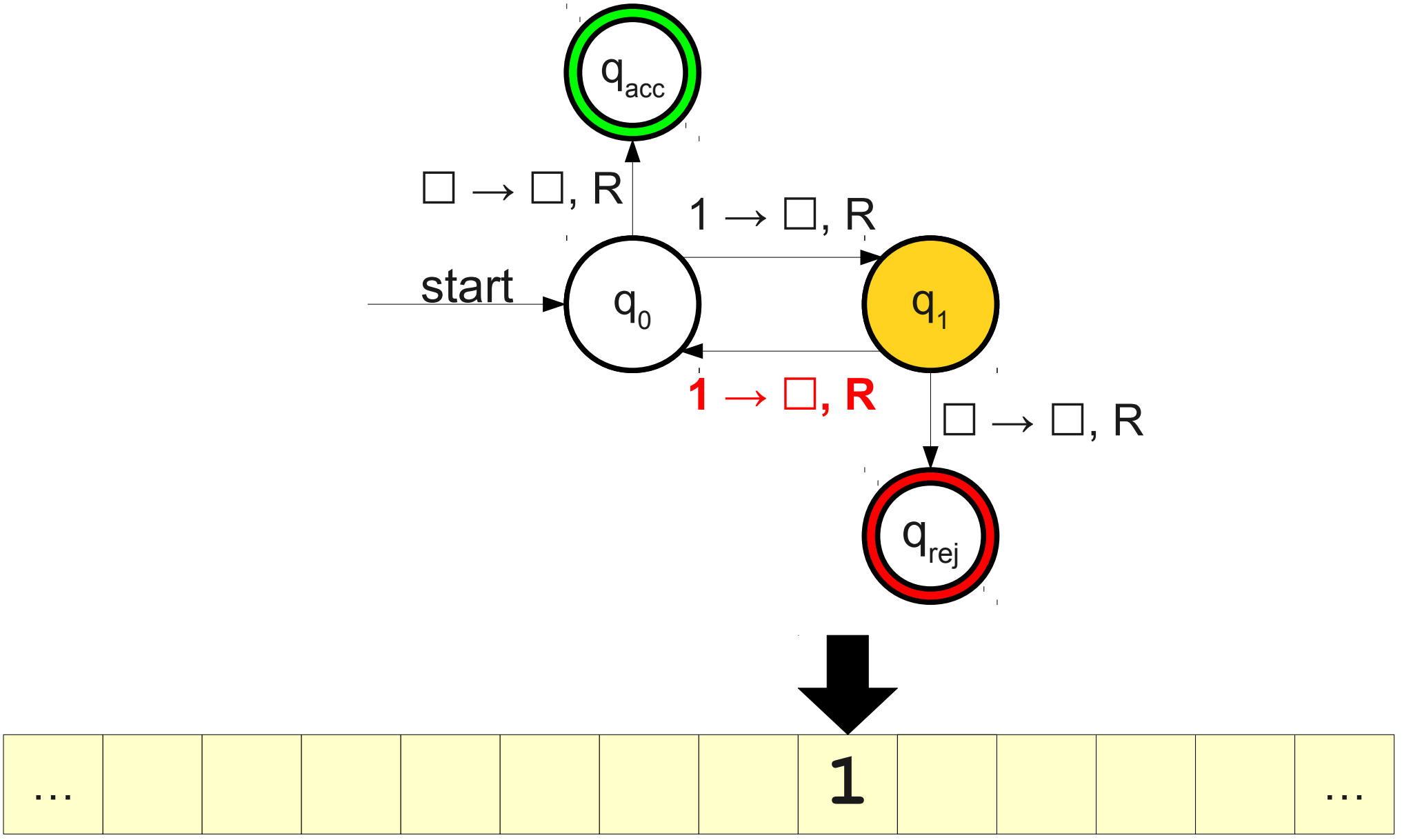




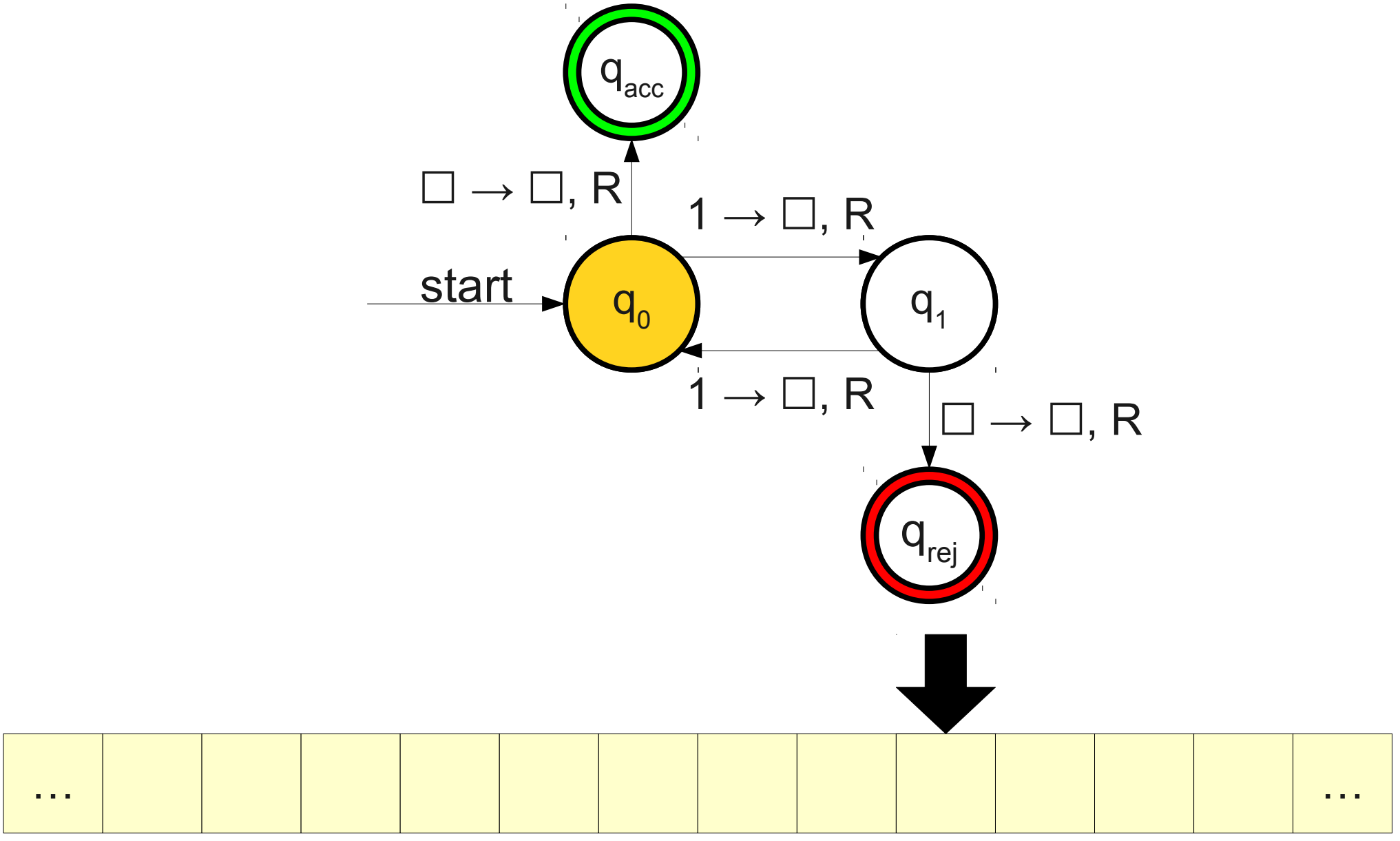
A Simple Turing Machine



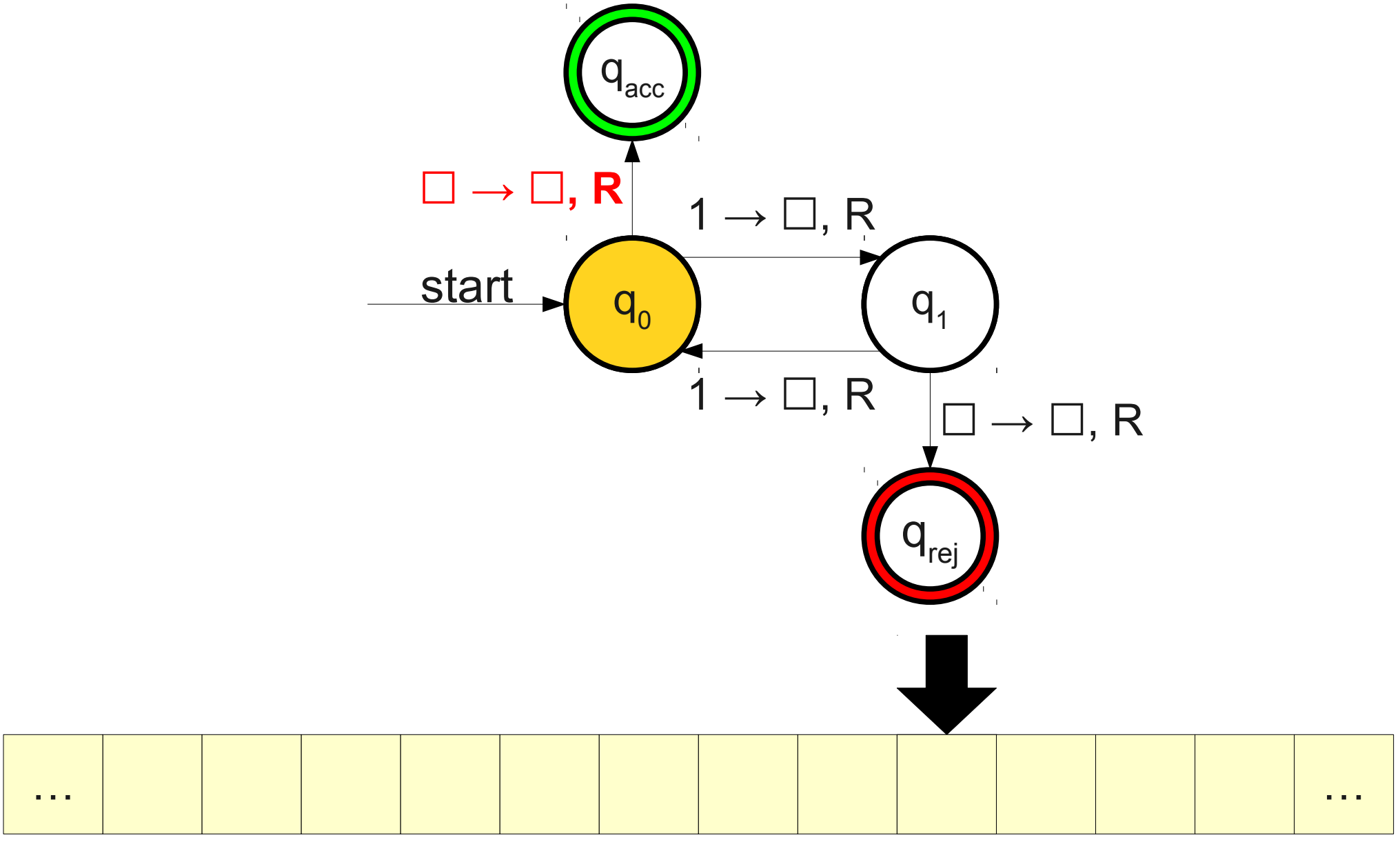
A Simple Turing Machine



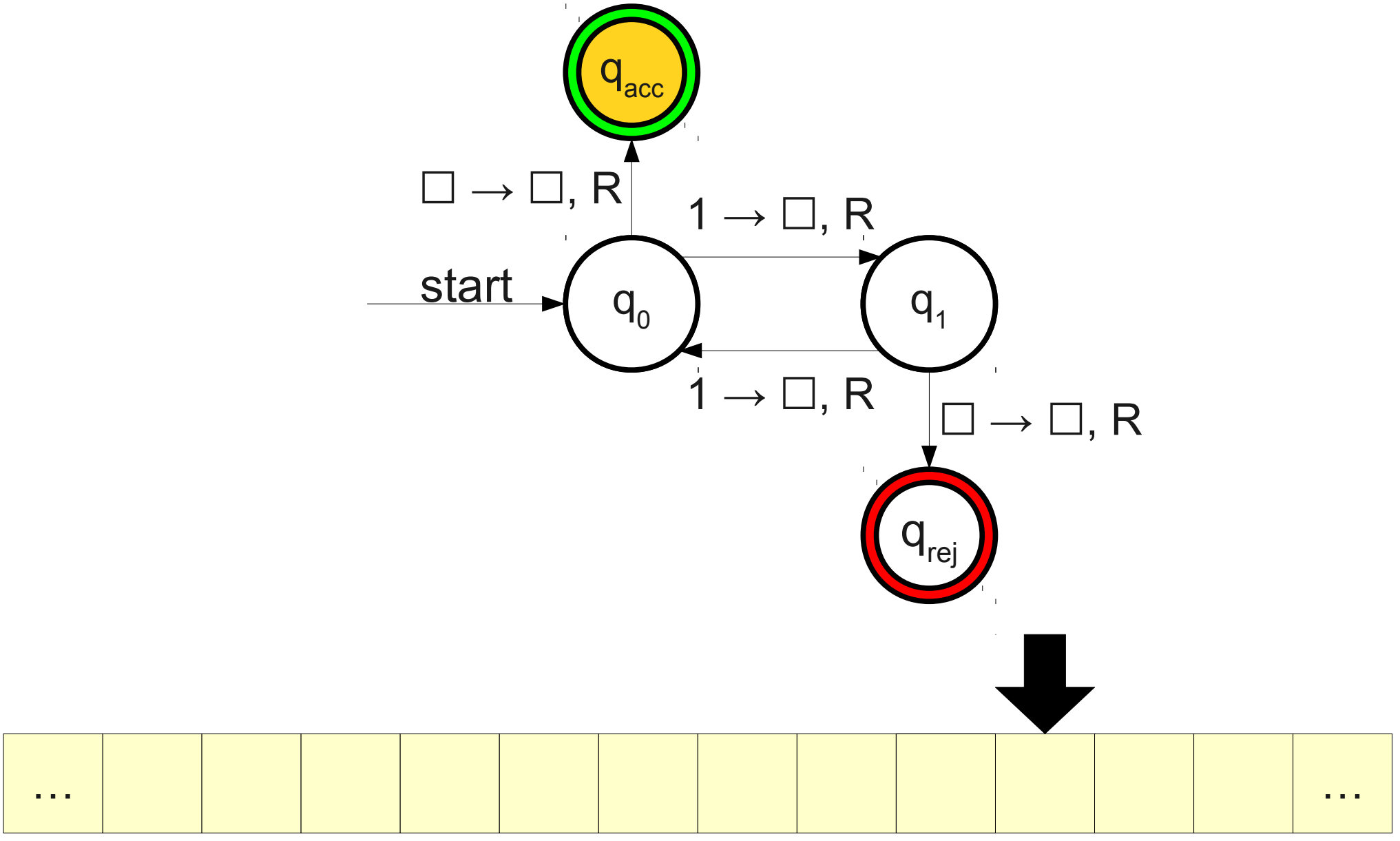
A Simple Turing Machine



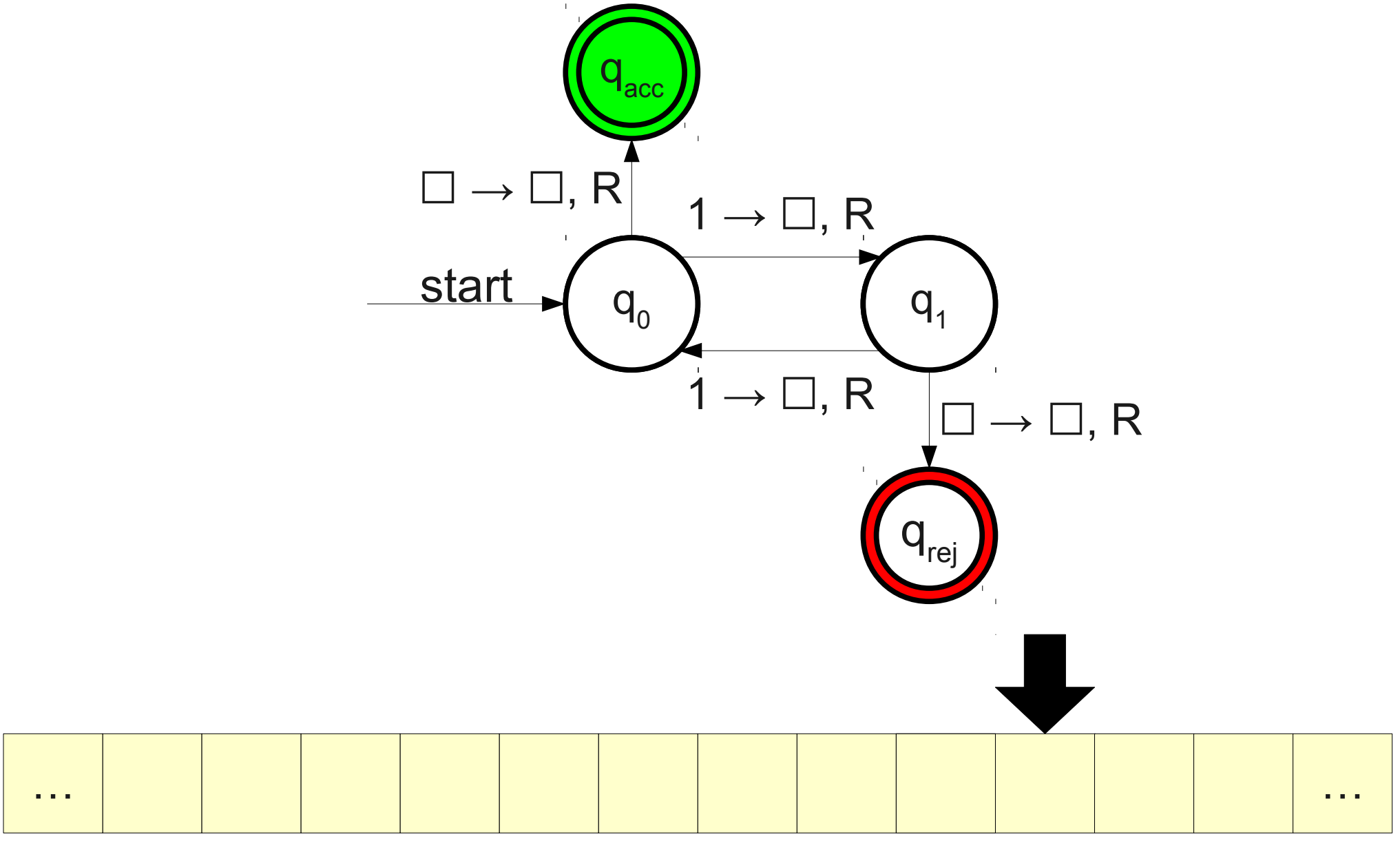
A Simple Turing Machine



A Simple Turing Machine



A Simple Turing Machine



Accepting and Rejecting States

- Unlike DFAs, Turing machines do not stop processing the input when they finish reading it.
- Turing machines decide when (and if!) they will accept or reject their input.
- Turing machines can enter infinite loops and never accept or reject; more on that later...

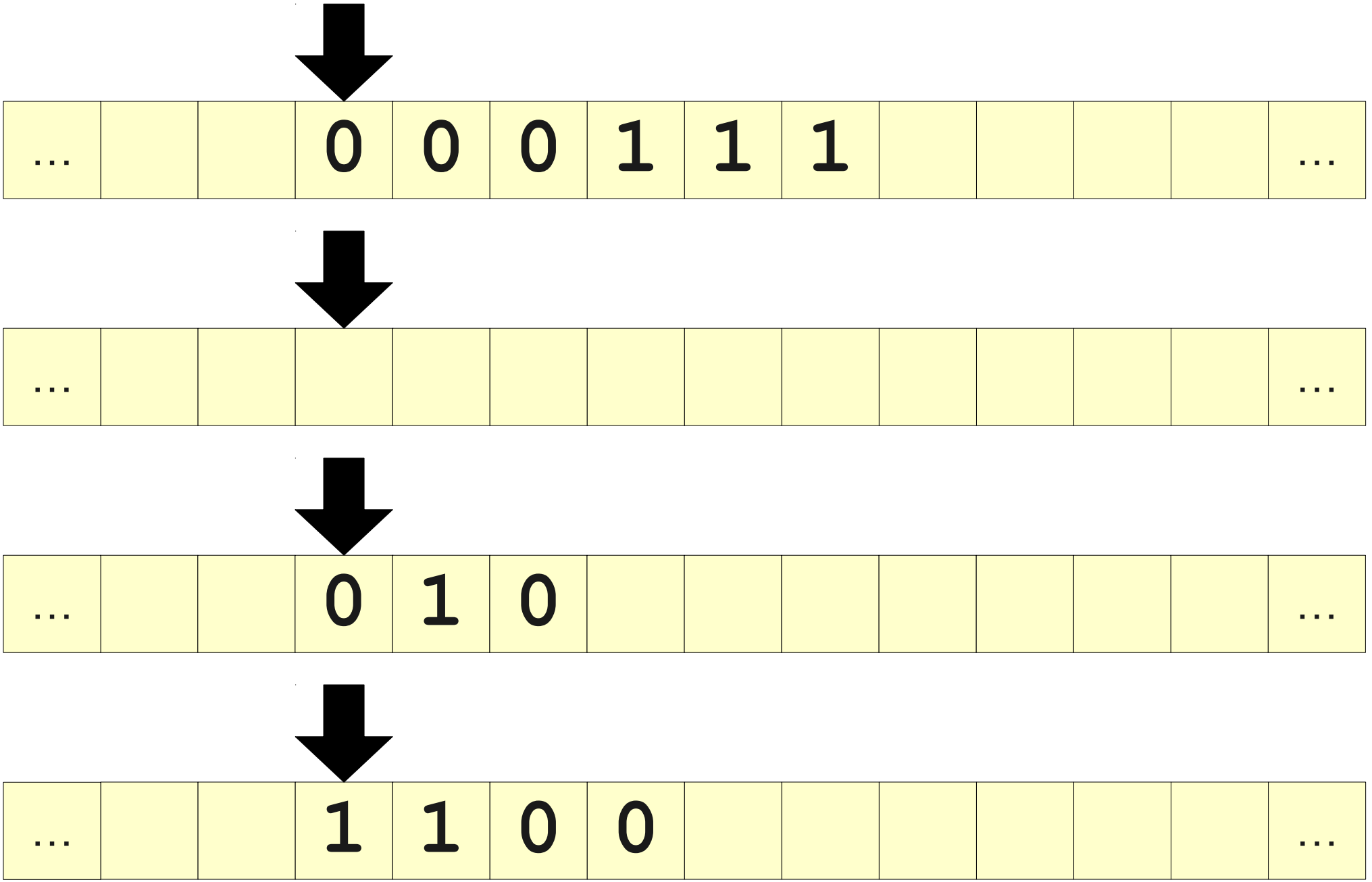
Designing Turing Machines

- Despite their simplicity, Turing machines are very powerful computing devices.
- Today's lecture explores how to design Turing machines for various languages.

Designing Turing Machines

- Let $\Sigma = \{\texttt{0}, \texttt{1}\}$ and consider the language $L = \{\texttt{0}^n\texttt{1}^n \mid n \in \mathbb{N}\}$.
- We know that L is context-free.
- How might we build a Turing machine for it?

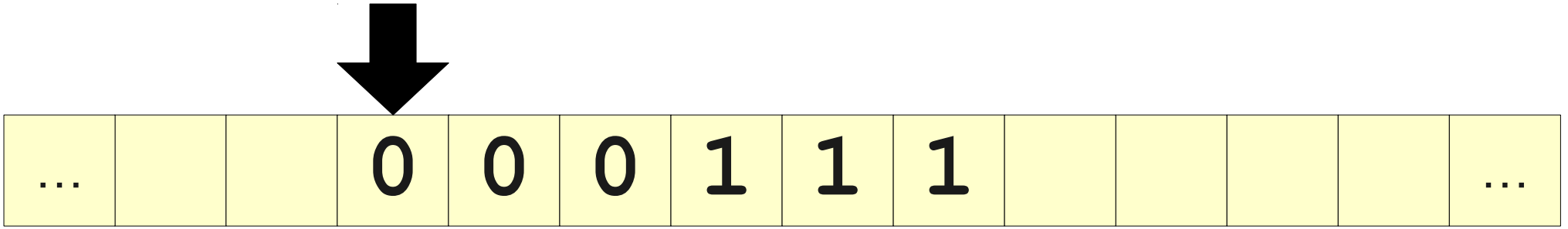
$$L = \{ \mathbf{0}^n \mathbf{1}^n \mid n \in \mathbb{N} \}$$



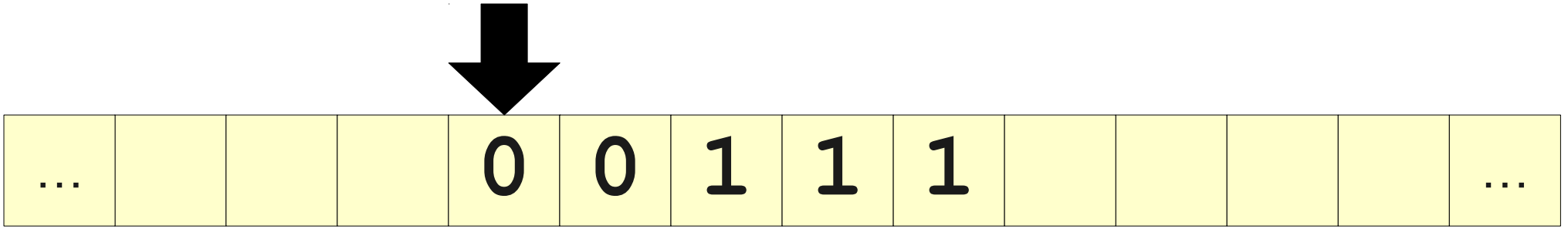
A Recursive Approach

- The string ε is in L.
- The string `0`w`1` is in L iff w is in L.
- Any string starting with `1` is not in L.
- Any string ending with `0` is not in L.

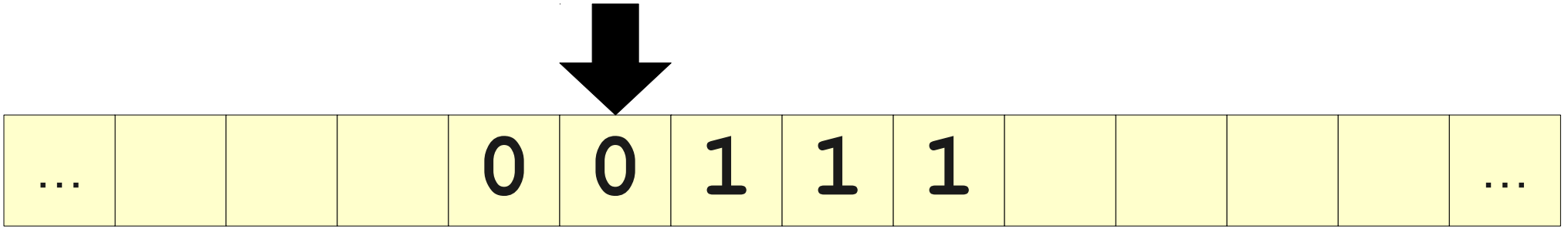
A Sketch of the TM



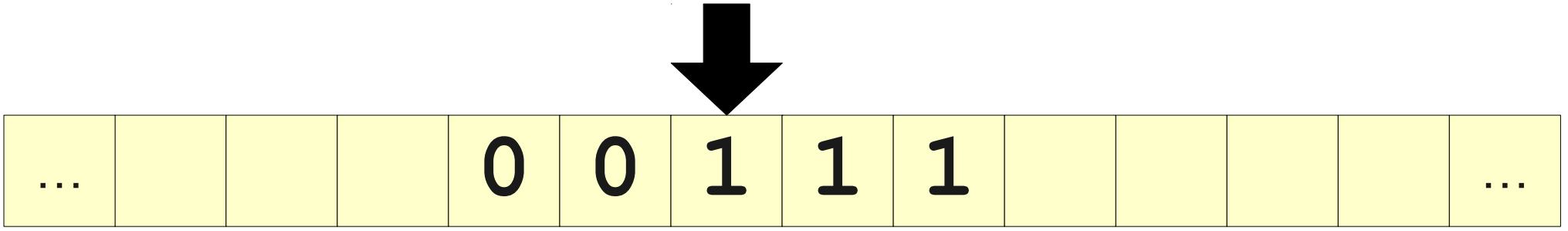
A Sketch of the TM



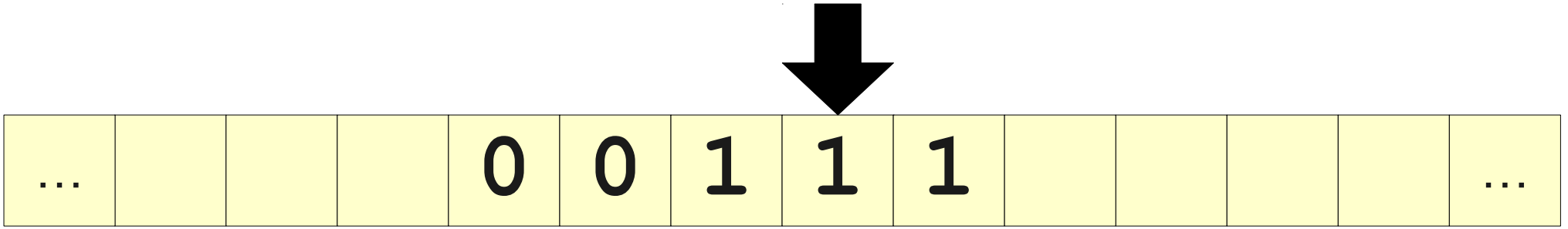
A Sketch of the TM



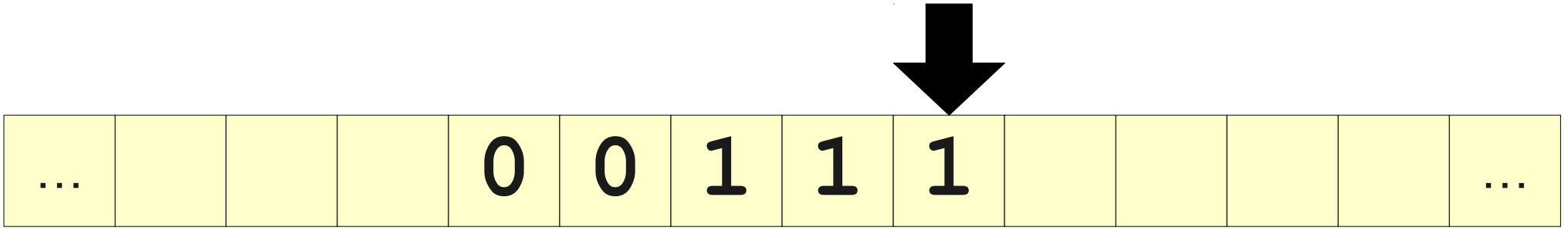
A Sketch of the TM



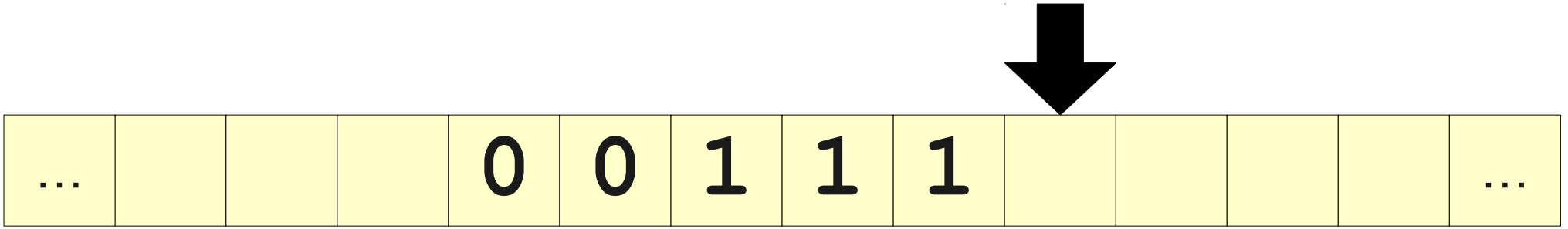
A Sketch of the TM



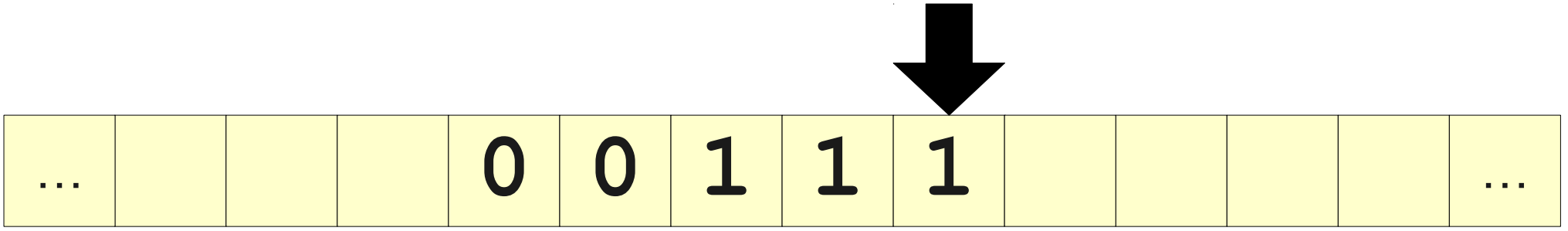
A Sketch of the TM



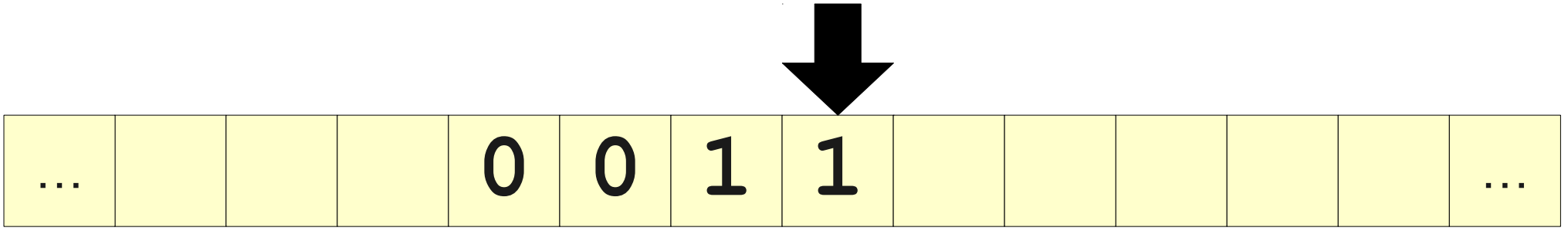
A Sketch of the TM



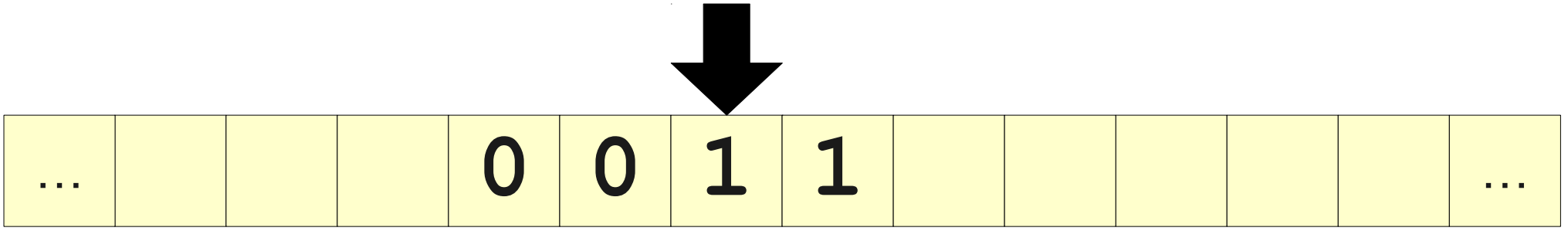
A Sketch of the TM



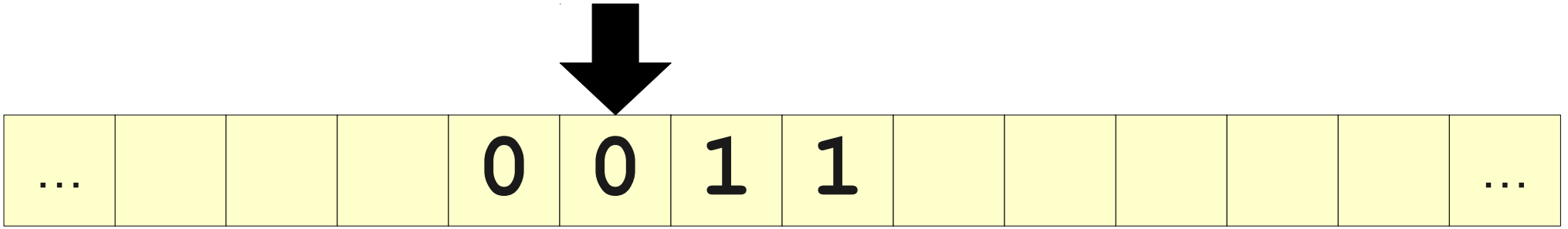
A Sketch of the TM



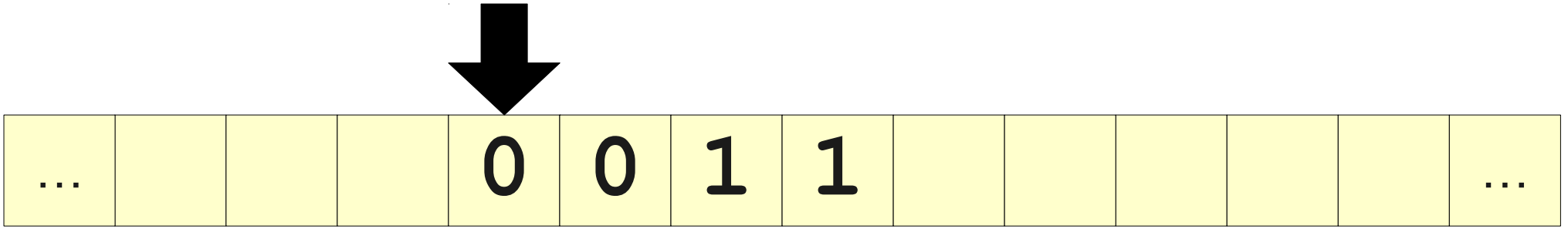
A Sketch of the TM



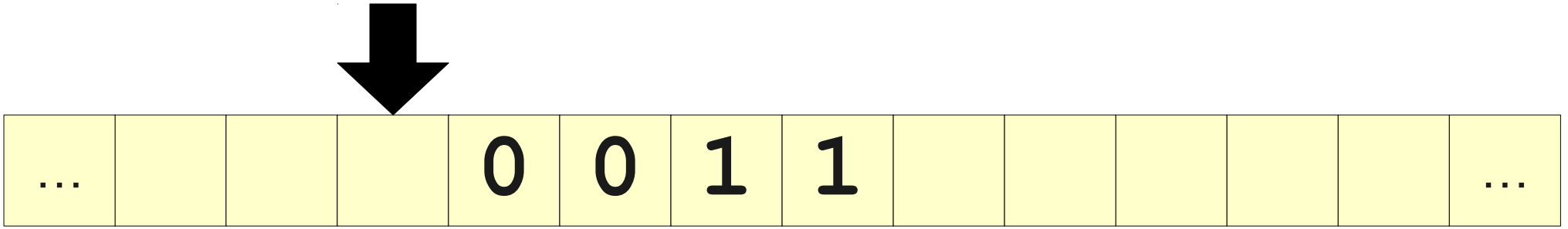
A Sketch of the TM



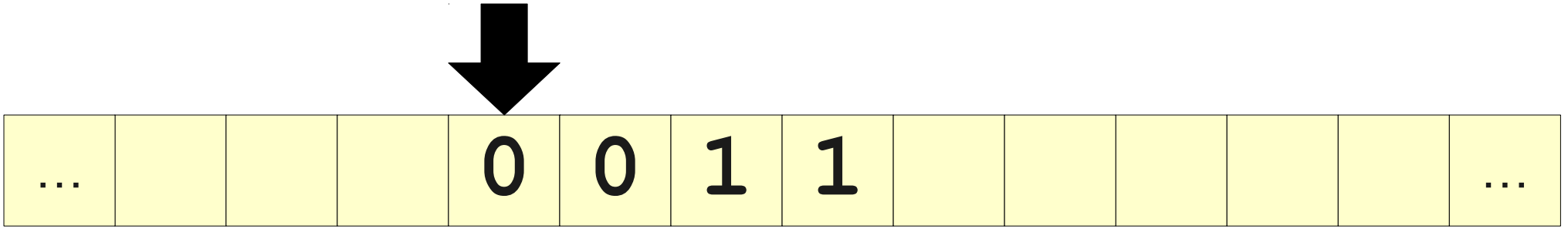
A Sketch of the TM



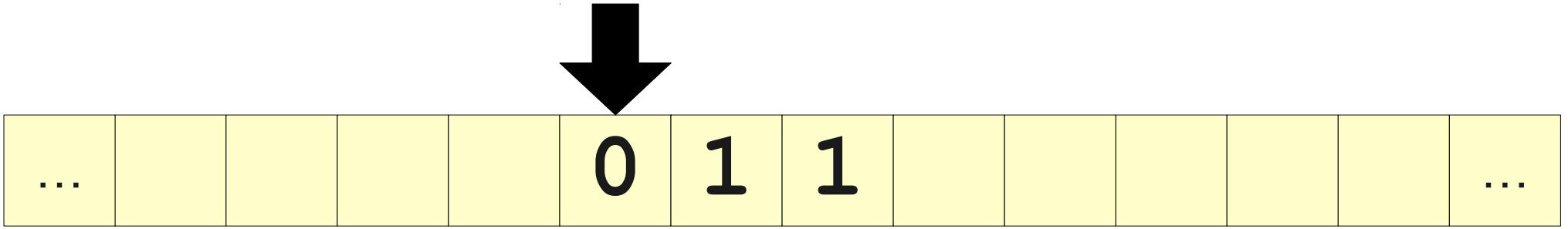
A Sketch of the TM



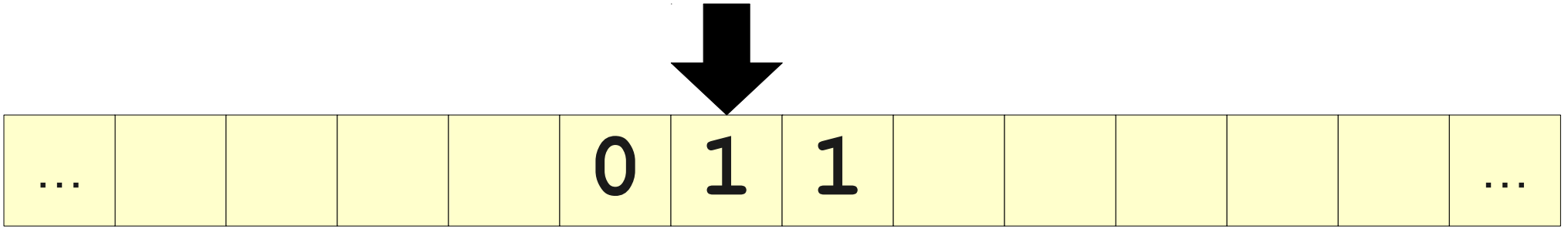
A Sketch of the TM



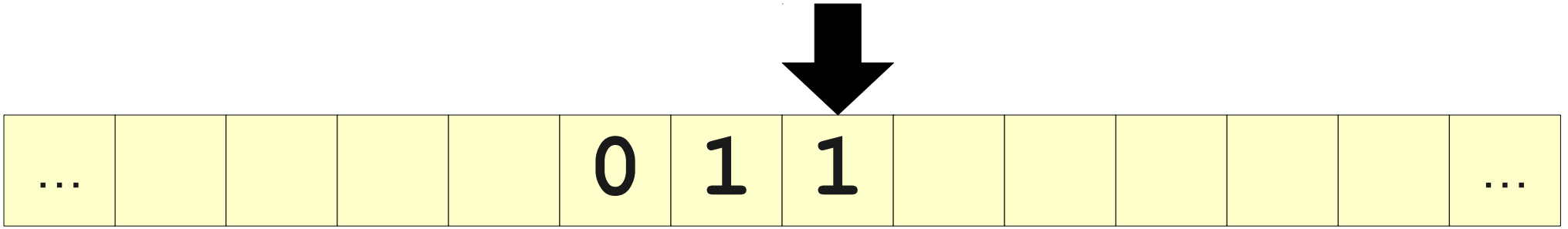
A Sketch of the TM



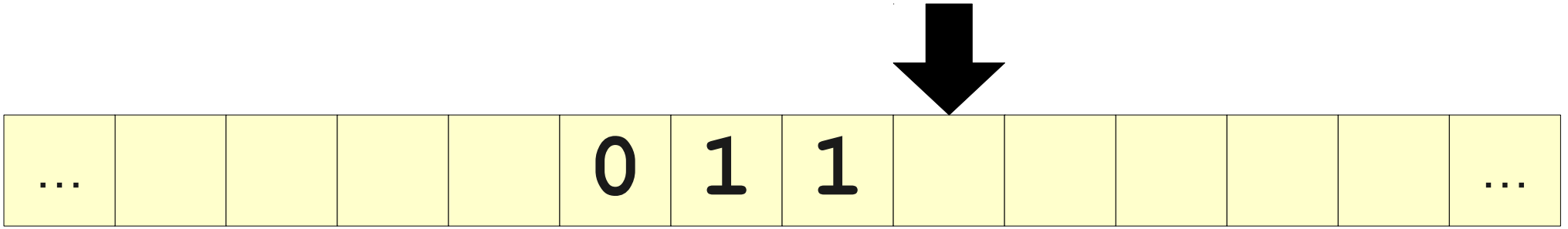
A Sketch of the TM



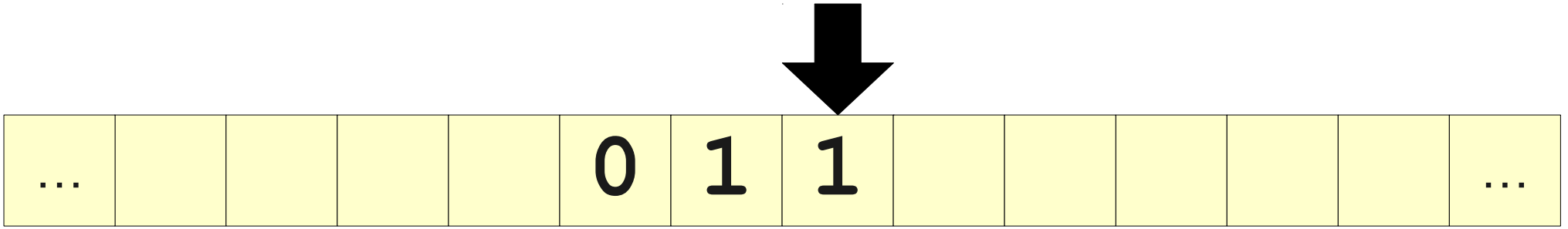
A Sketch of the TM



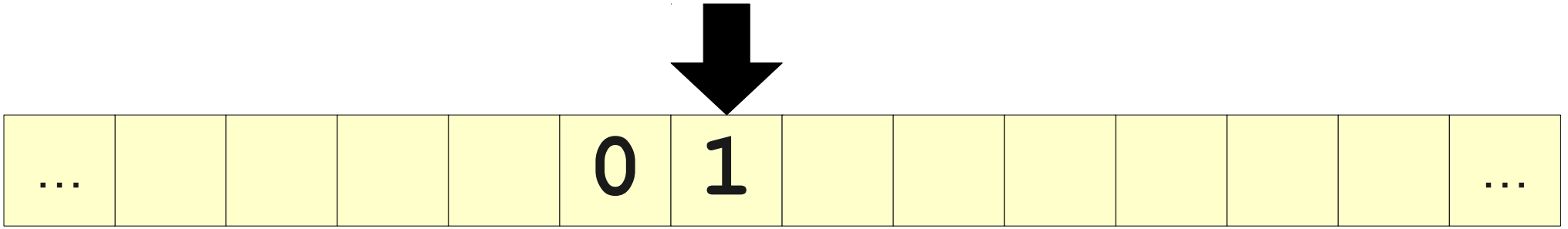
A Sketch of the TM



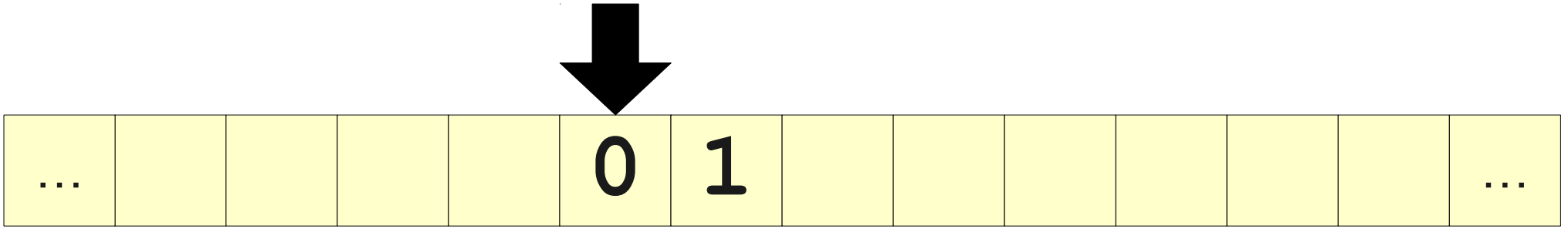
A Sketch of the TM



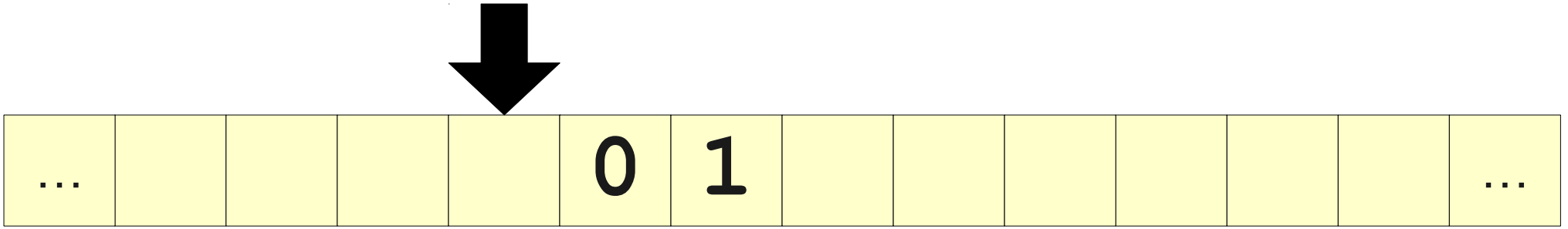
A Sketch of the TM



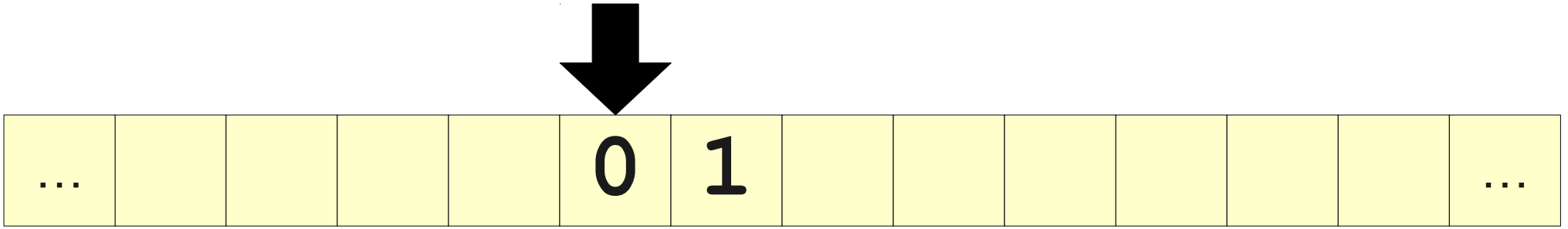
A Sketch of the TM



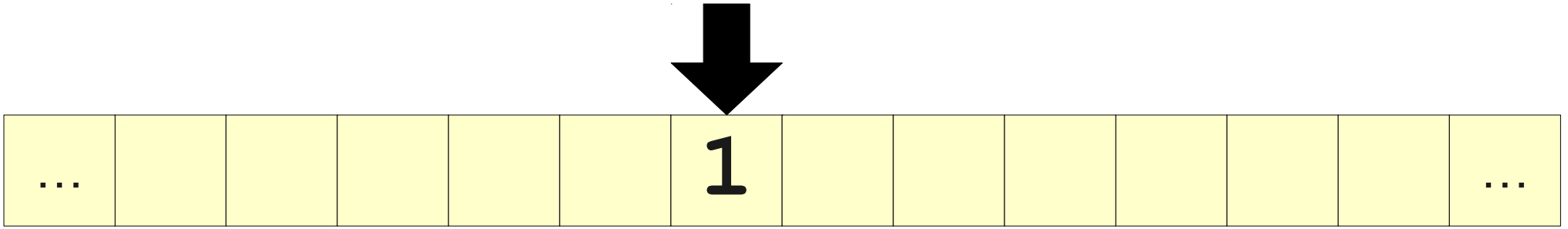
A Sketch of the TM



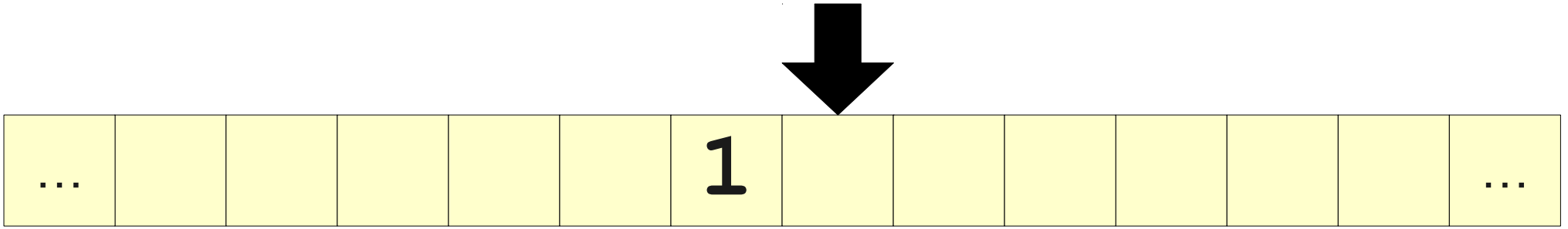
A Sketch of the TM



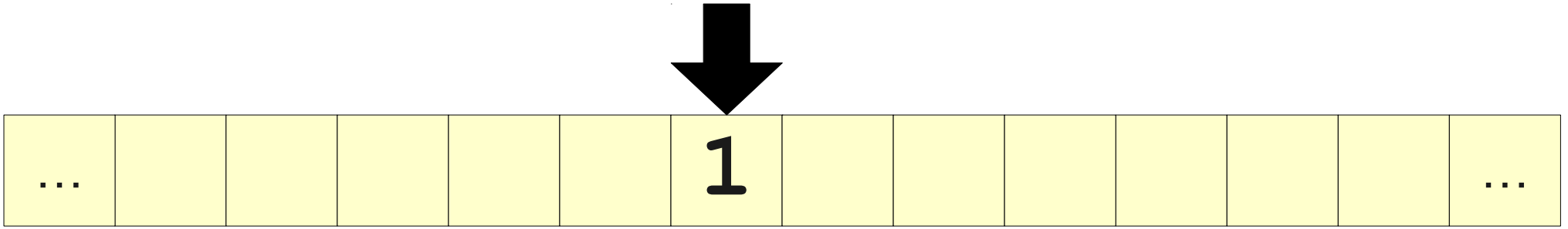
A Sketch of the TM



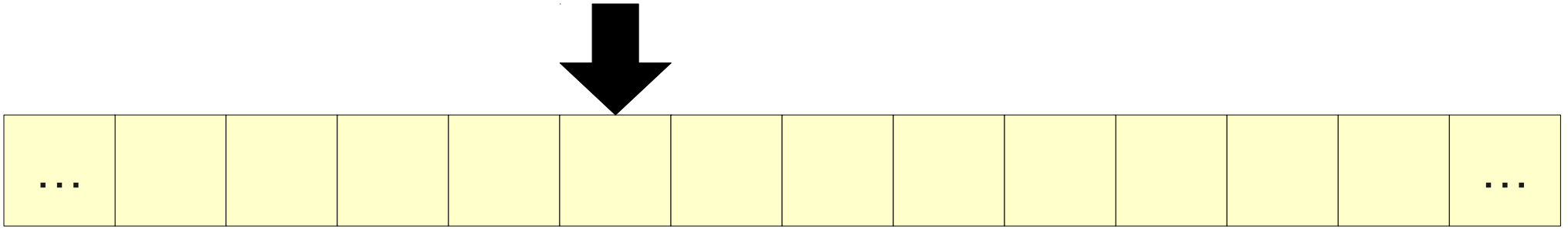
A Sketch of the TM



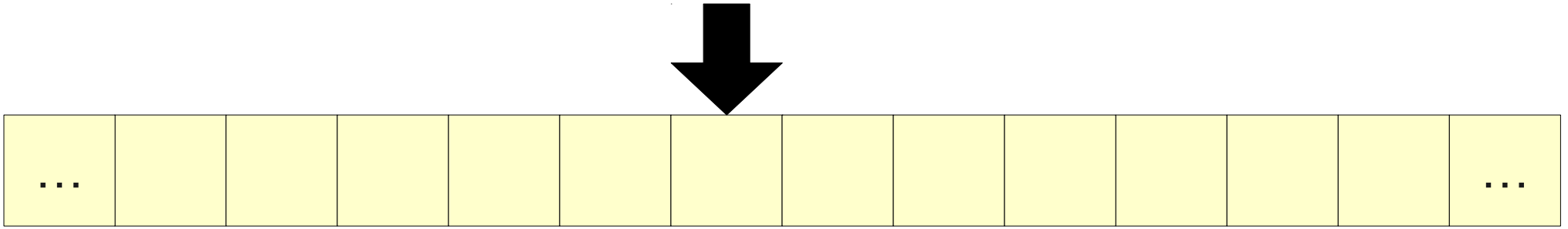
A Sketch of the TM

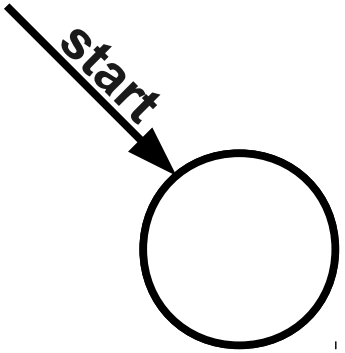


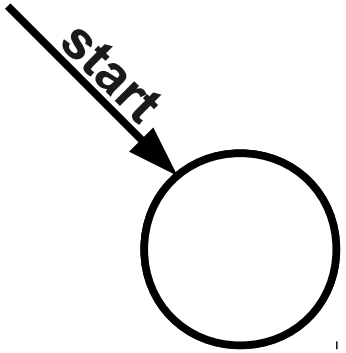
A Sketch of the TM

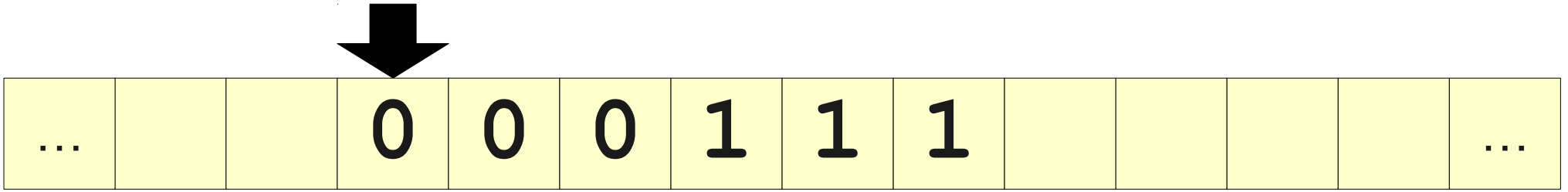


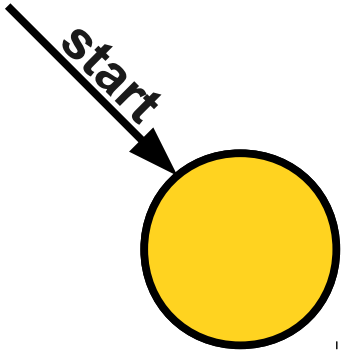
A Sketch of the TM

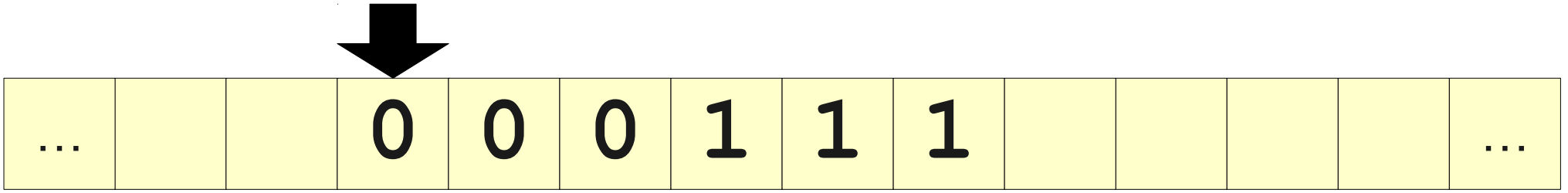


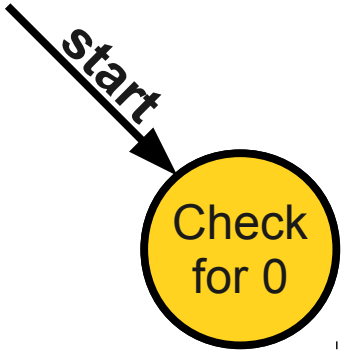

start

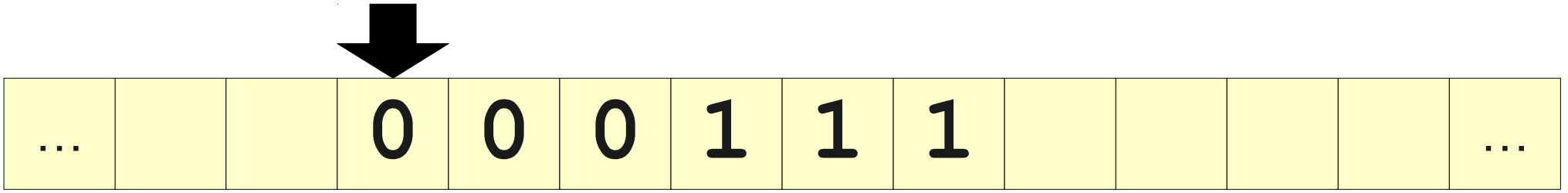

start

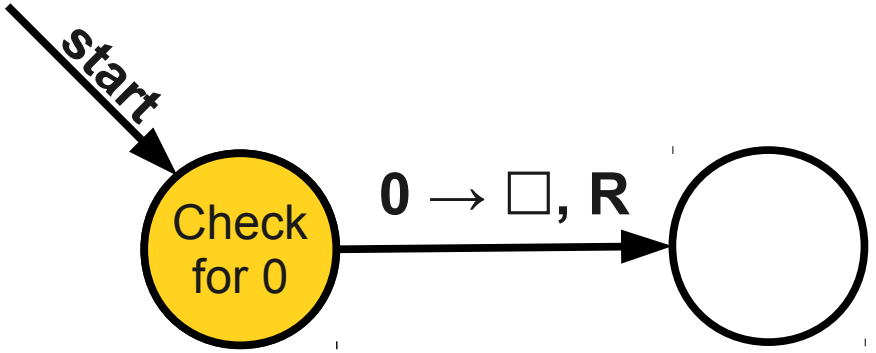

...
0 0 0 1 1 1
...

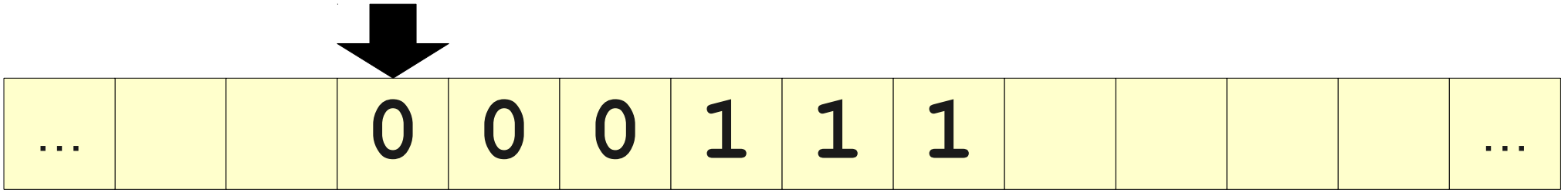

start

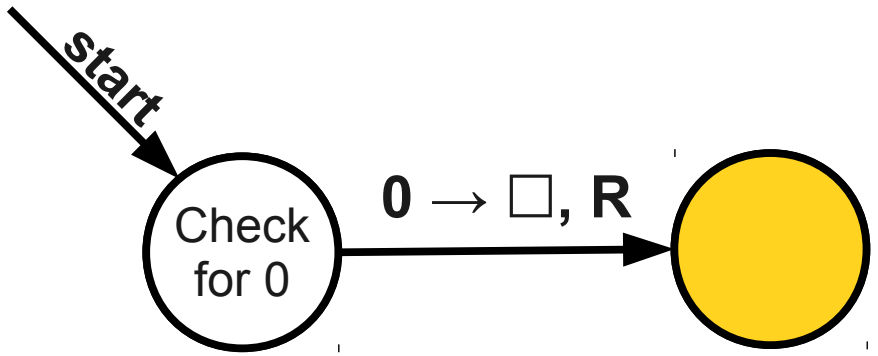

...
0 0 0 1 1 1
...

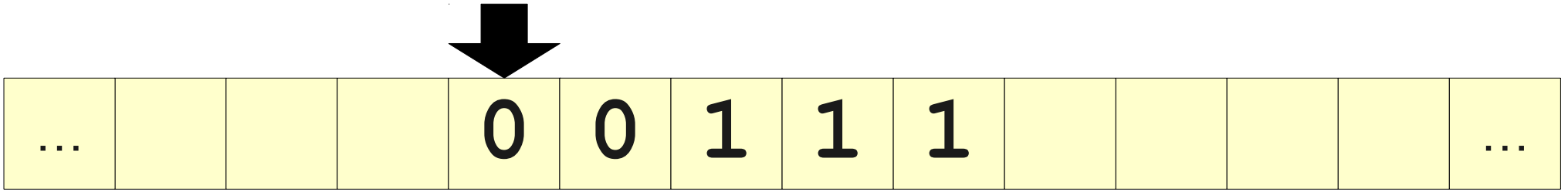

start
Check
for 0

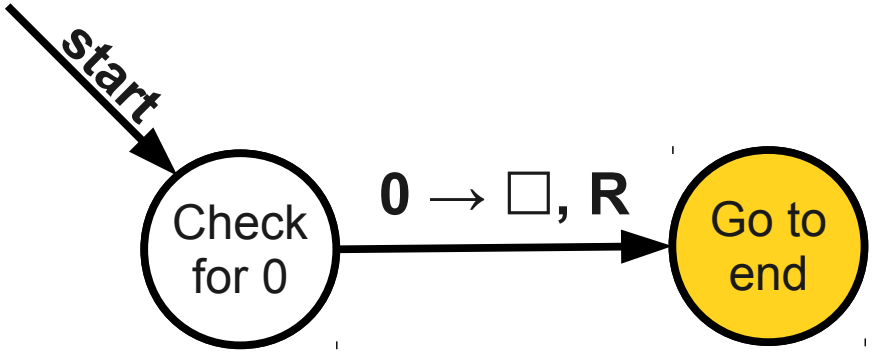

...
0 0 0 1 1 1
...

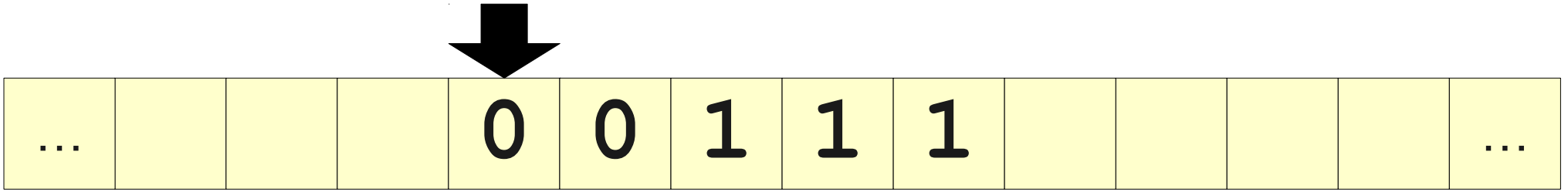

start
Check for 0
0 → □, R

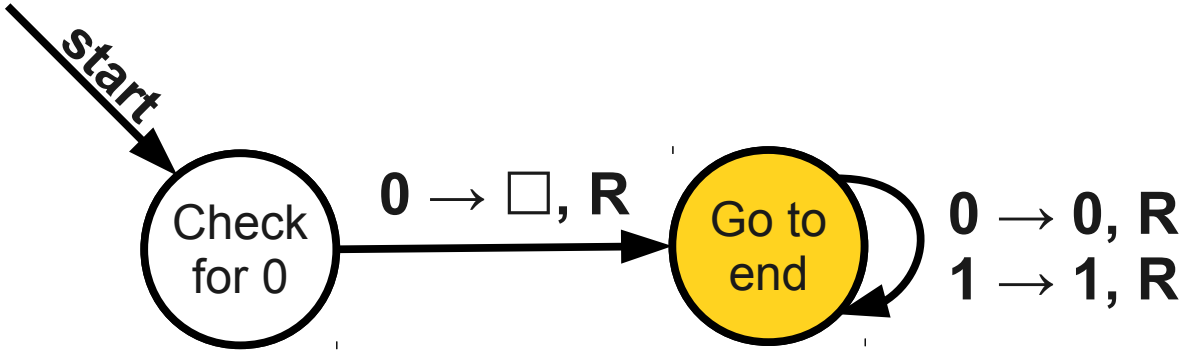

...
0 0 0 1 1 1
...

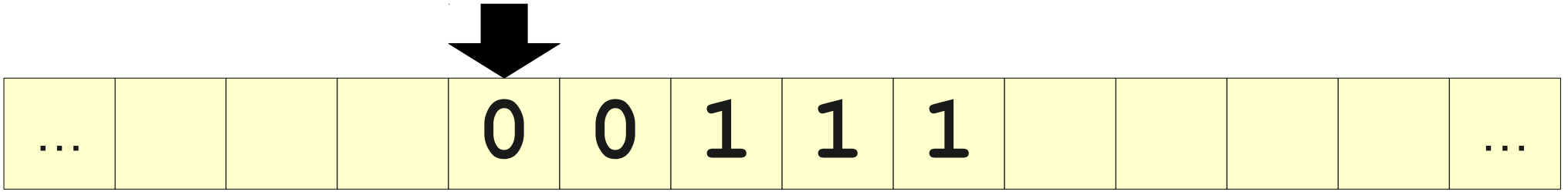

start
Check
for 0
0 → □, R

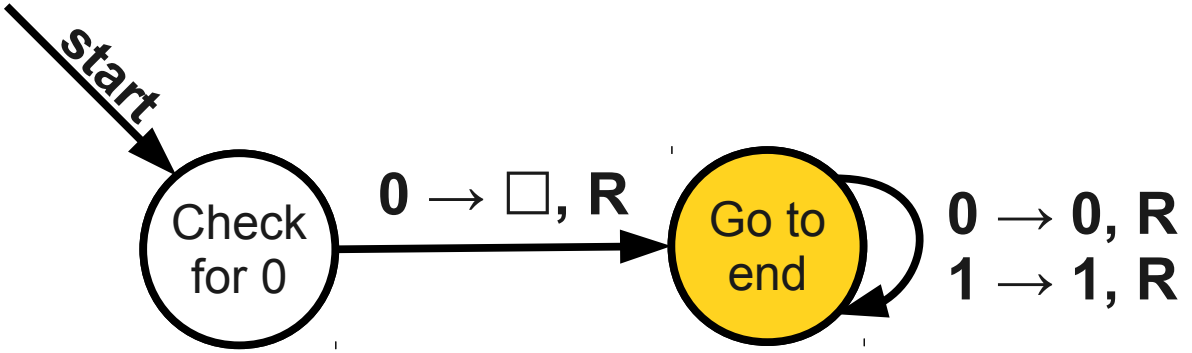

...
0 0 1 1 1
...

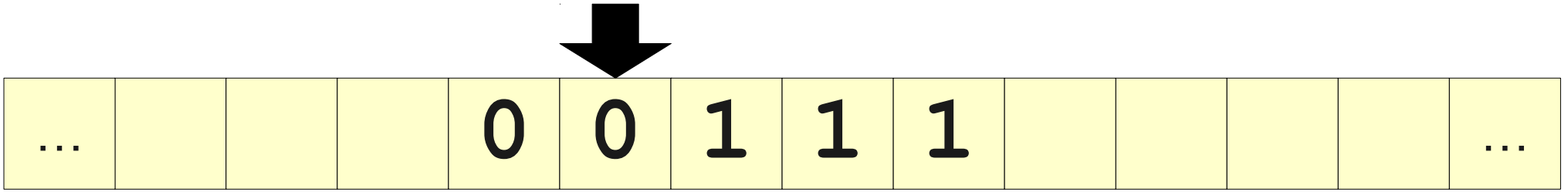

start
Check for 0
0 → □, R
Go to end

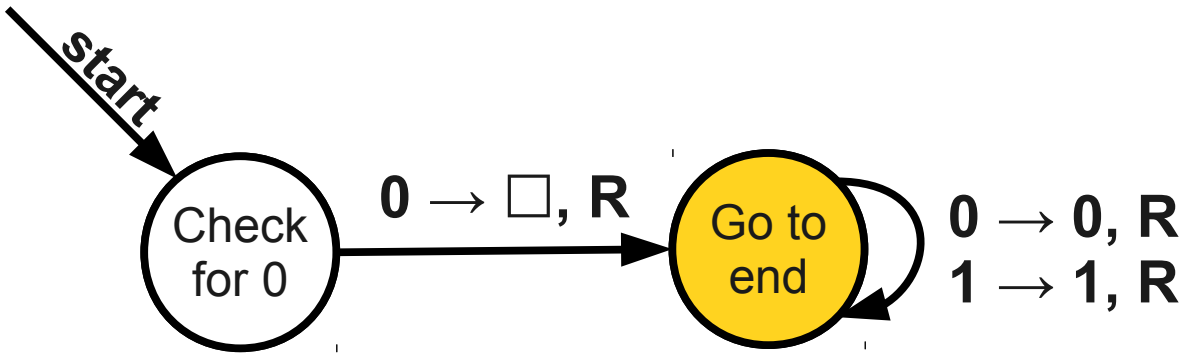

...
0 0 1 1 1
...

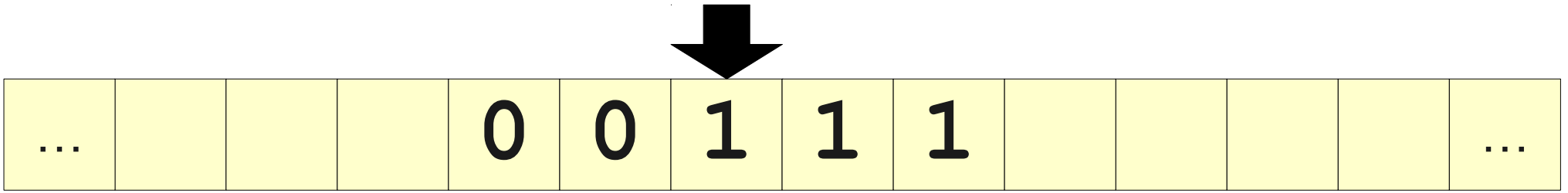

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

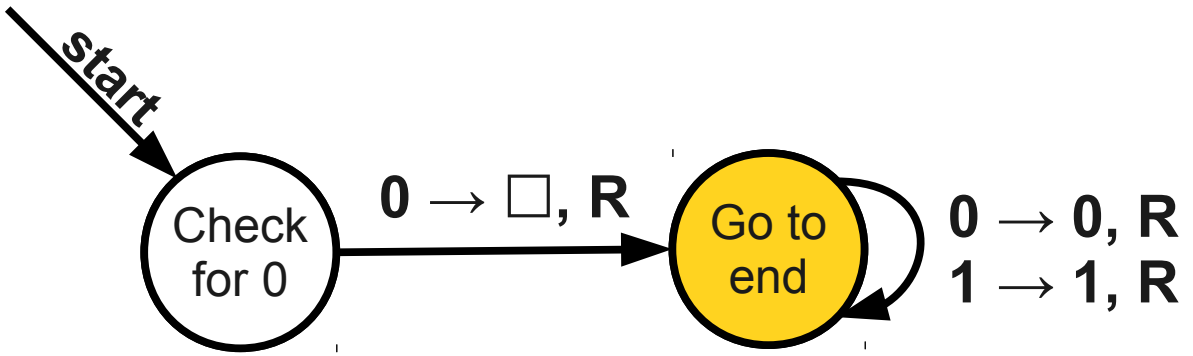

...
0 0 1 1 1
...

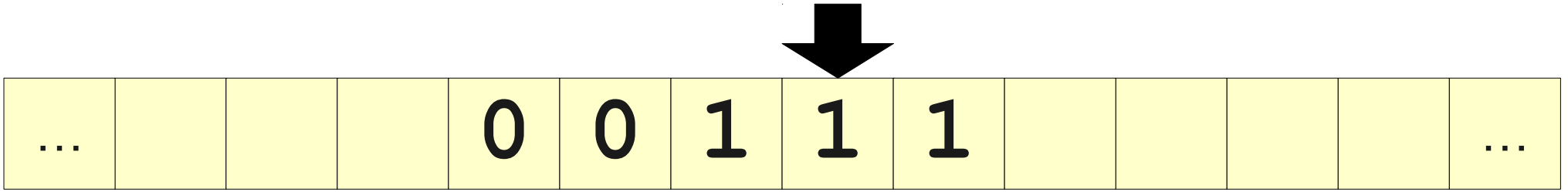

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

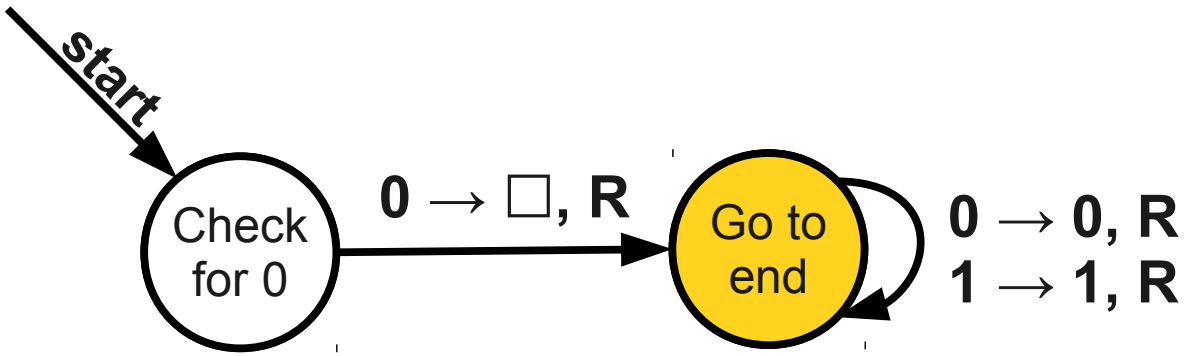

...
0 0 1 1 1
...

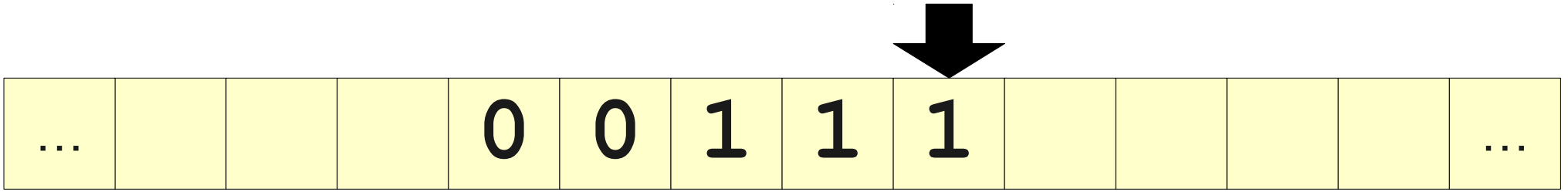

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

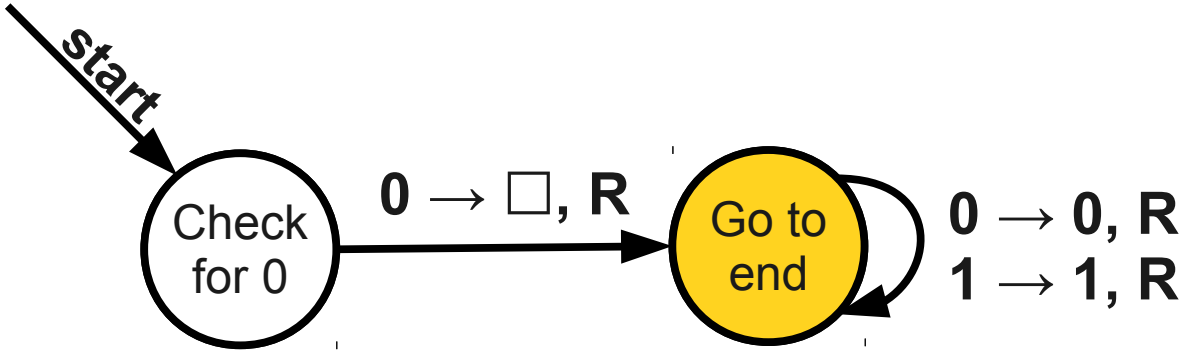

...
0 0 1 1 1
...

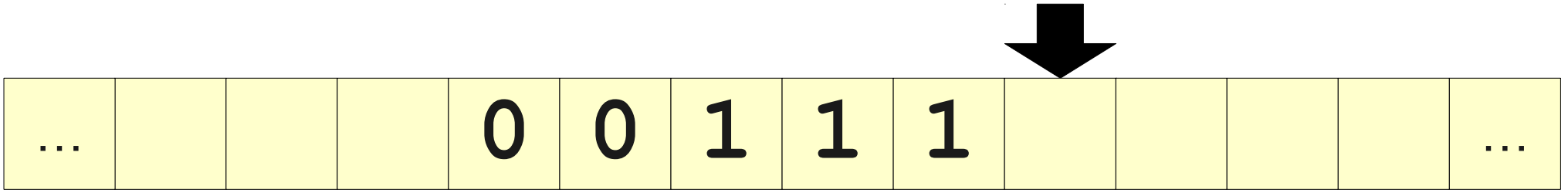

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

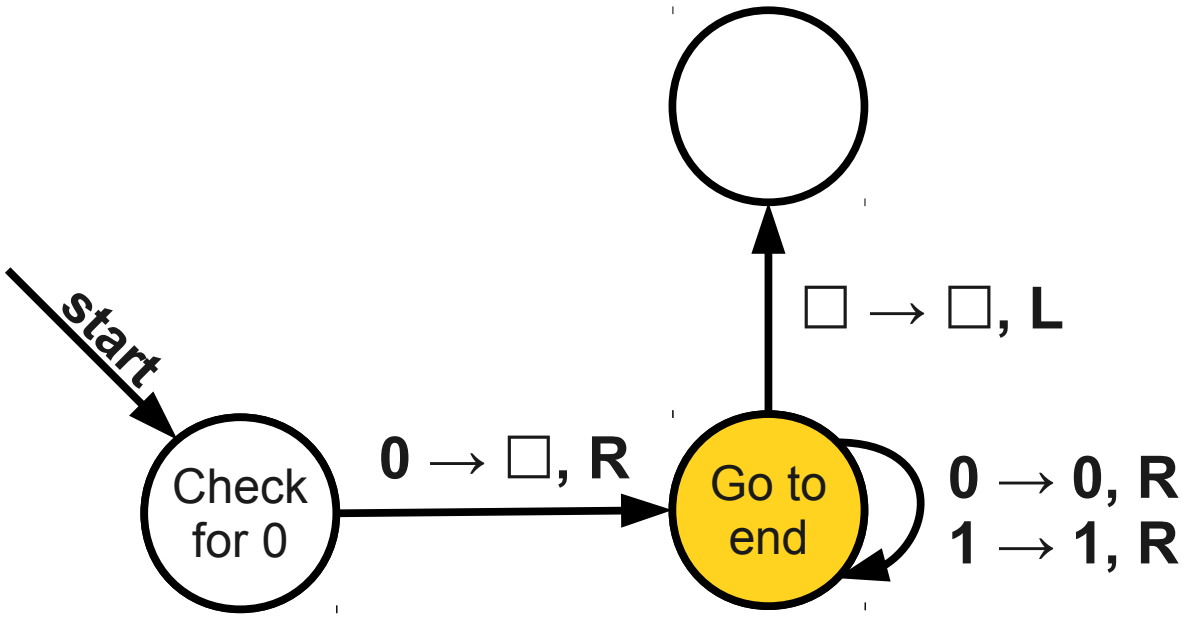

...
0 0 1 1 1
...

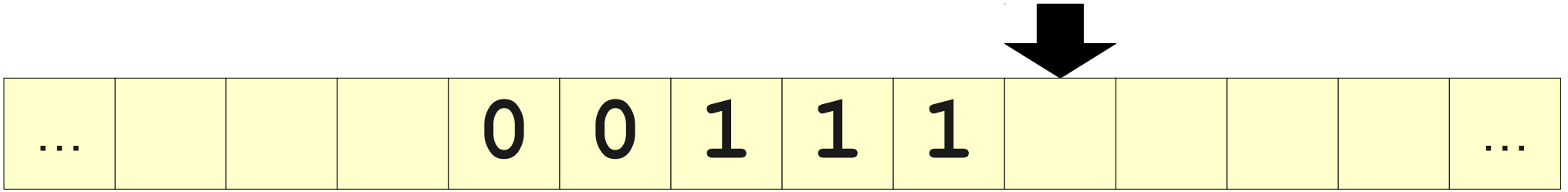

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

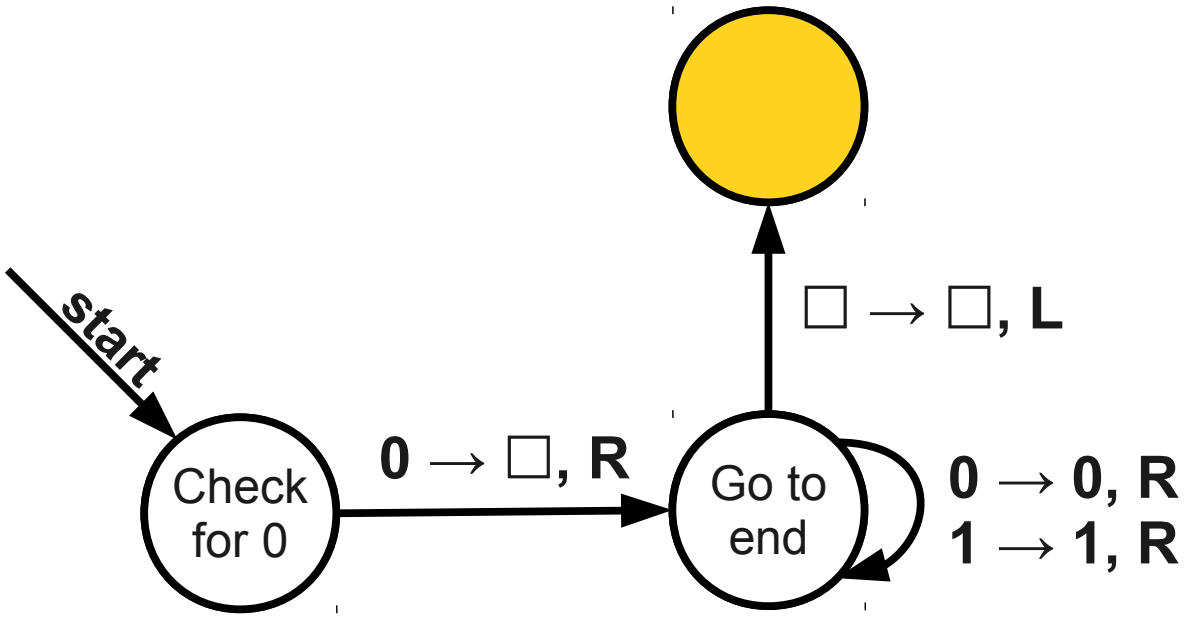

...
0 0 1 1 1
...

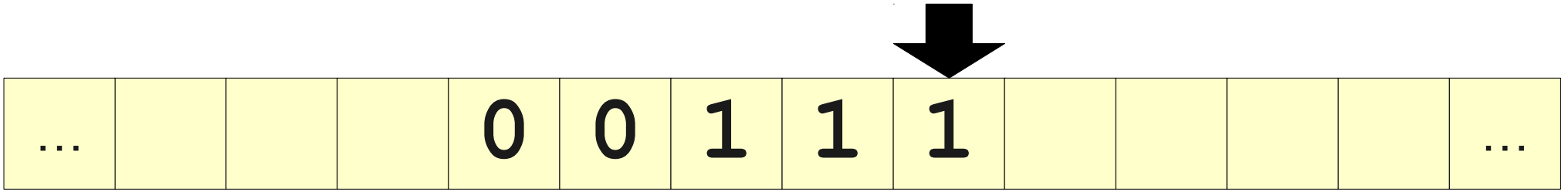

start
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

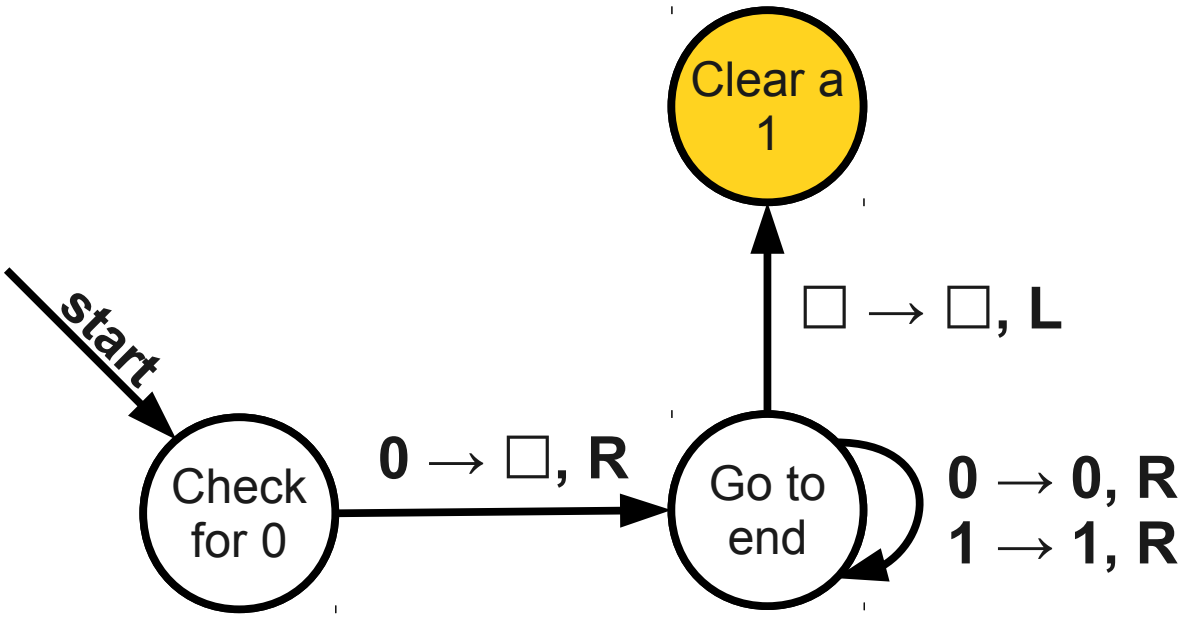

...
0 0 1 1 1
...

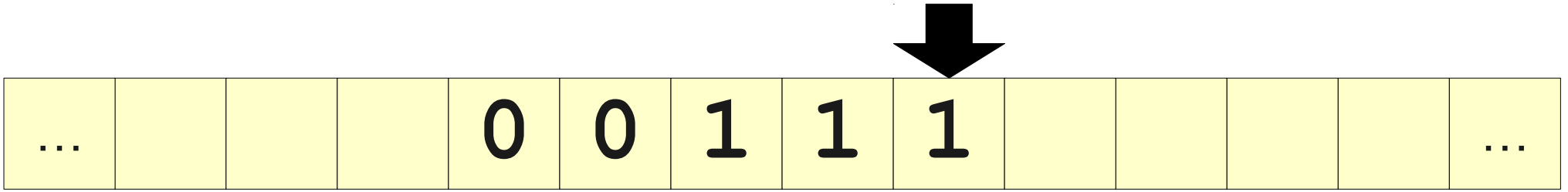

start
Check for 0
0 → □, R
Go to end
□ → □, L
0 → 0, R
1 → 1, R

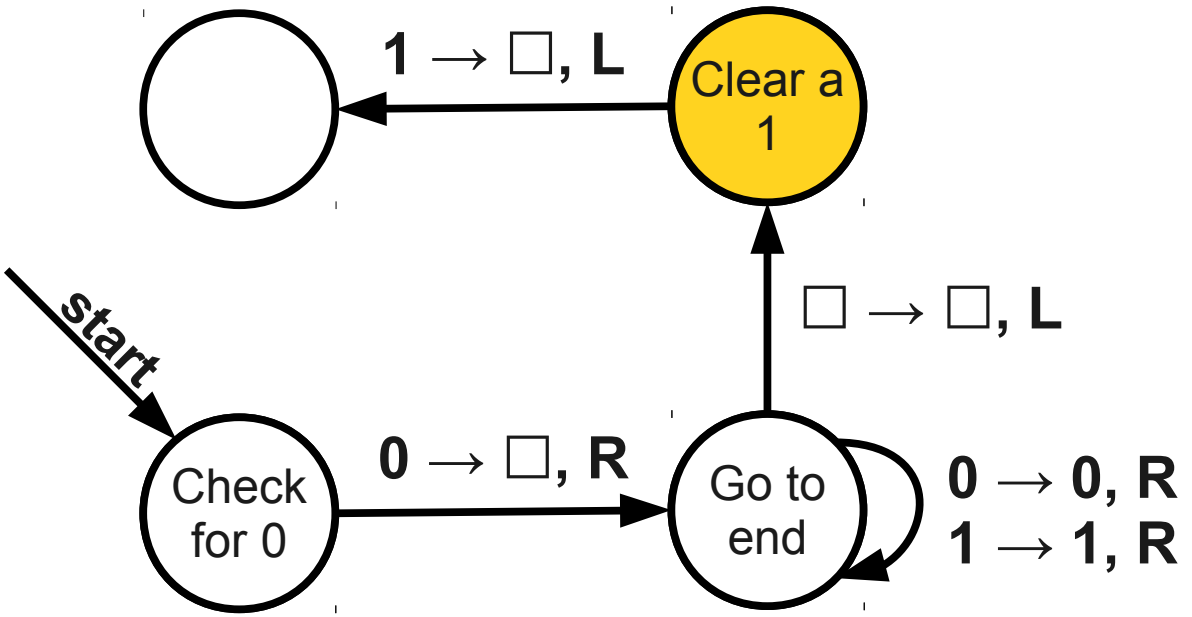

...
0 0 1 1 1
...

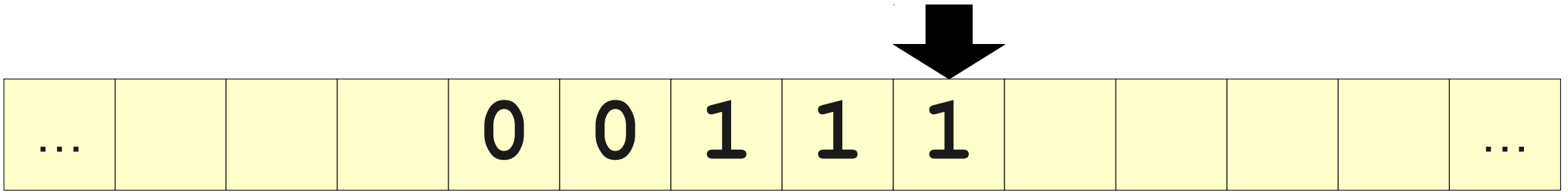

start
Check for 0
0 → □, R
Go to end
□ → □, L
0 → 0, R
1 → 1, R

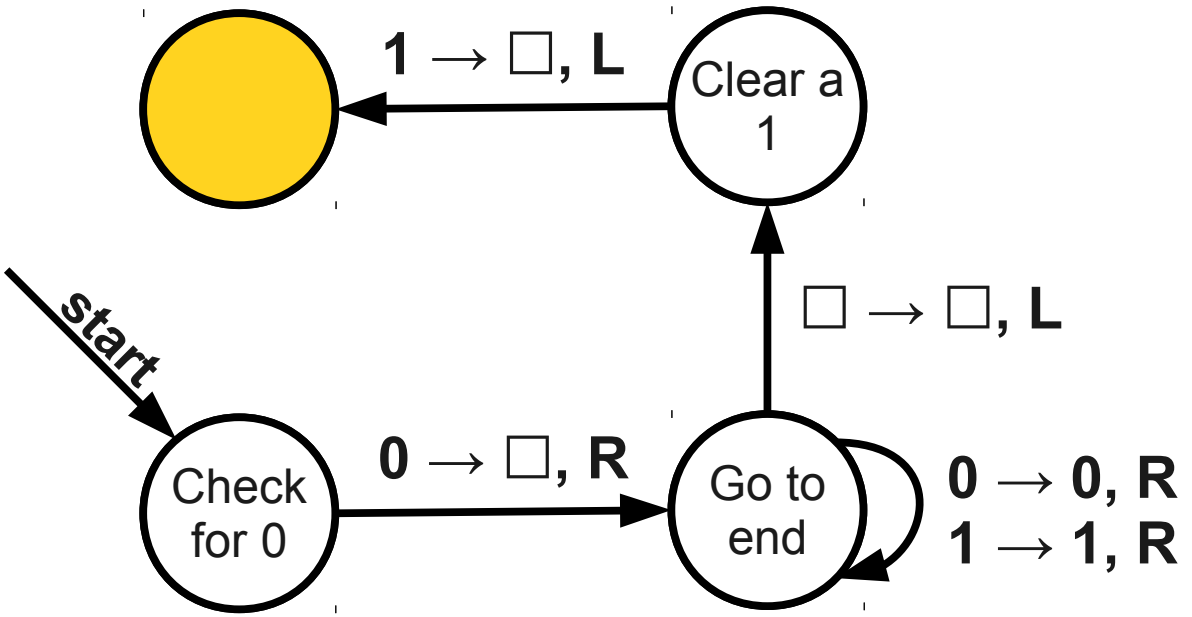

...
0 0 1 1 1
...

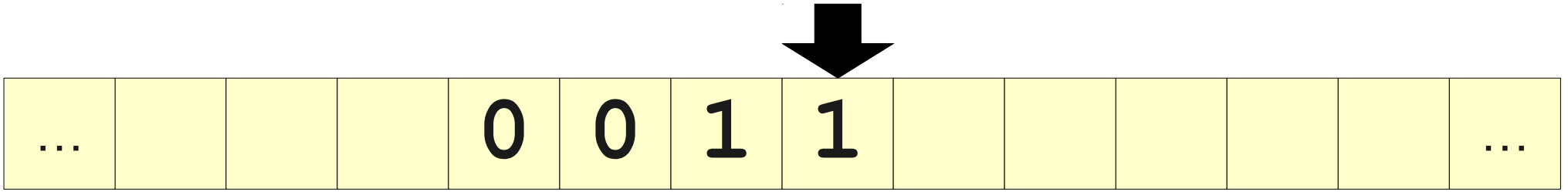

Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

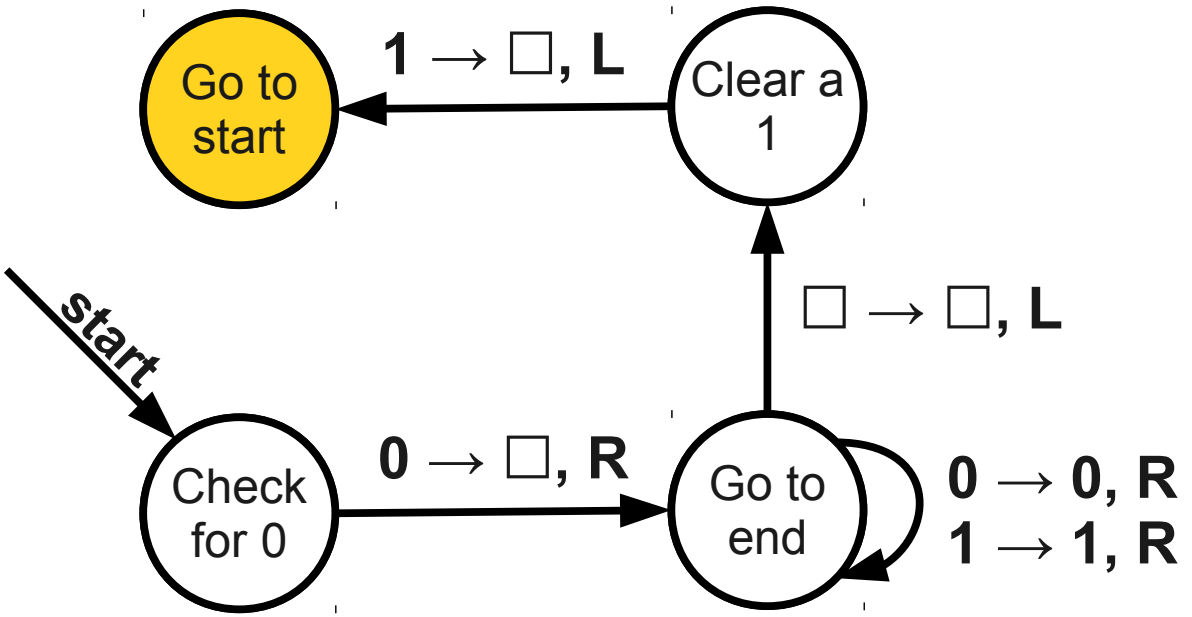

...
0 0 1 1 1
...

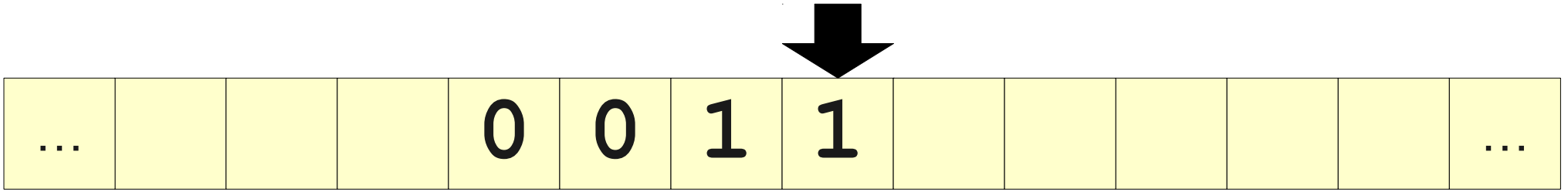

1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

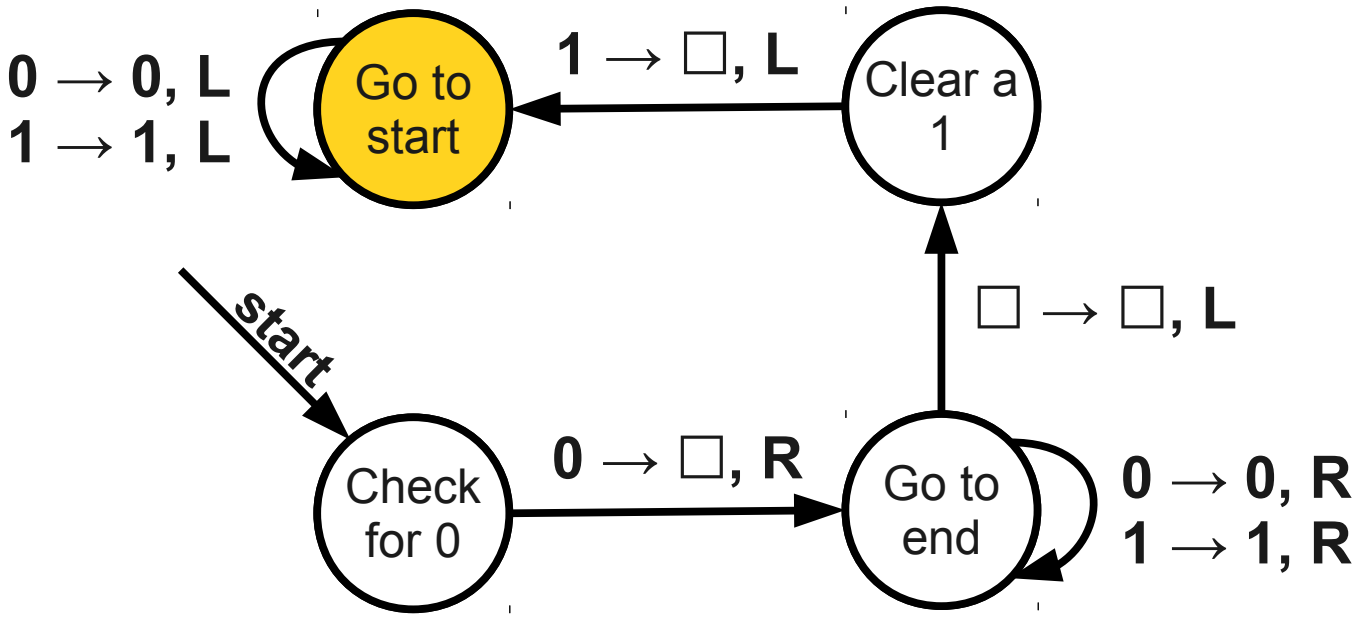

...
0 0 1 1 1
...

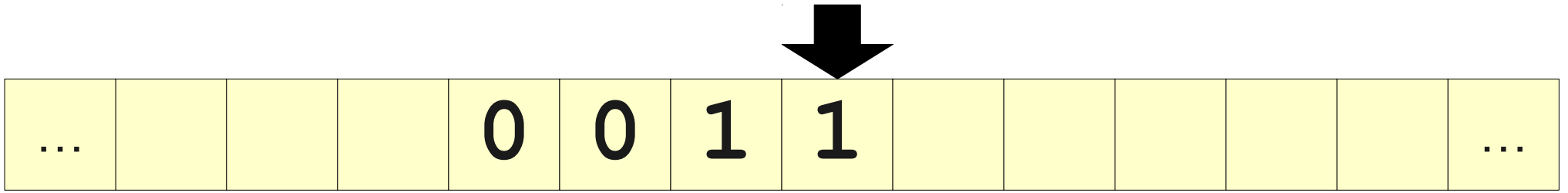

1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

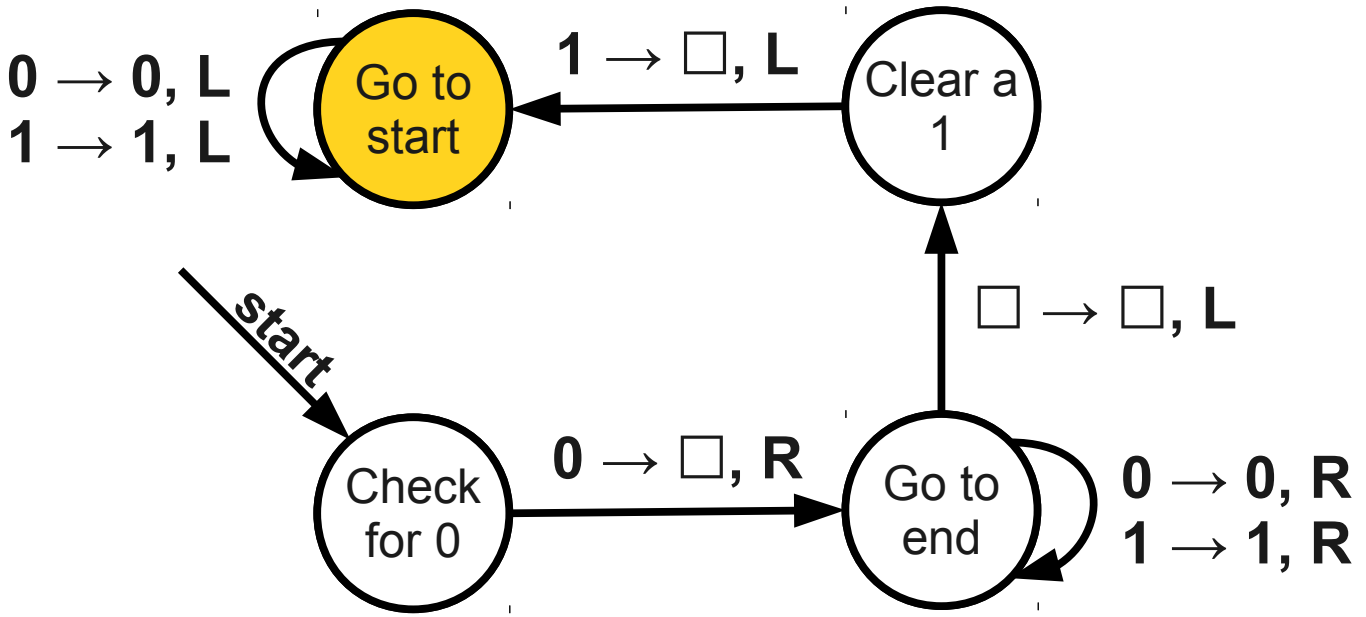

...
0 0 1 1
...

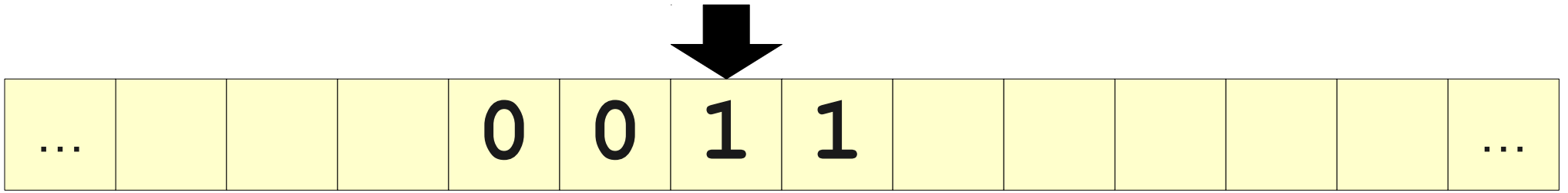

Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

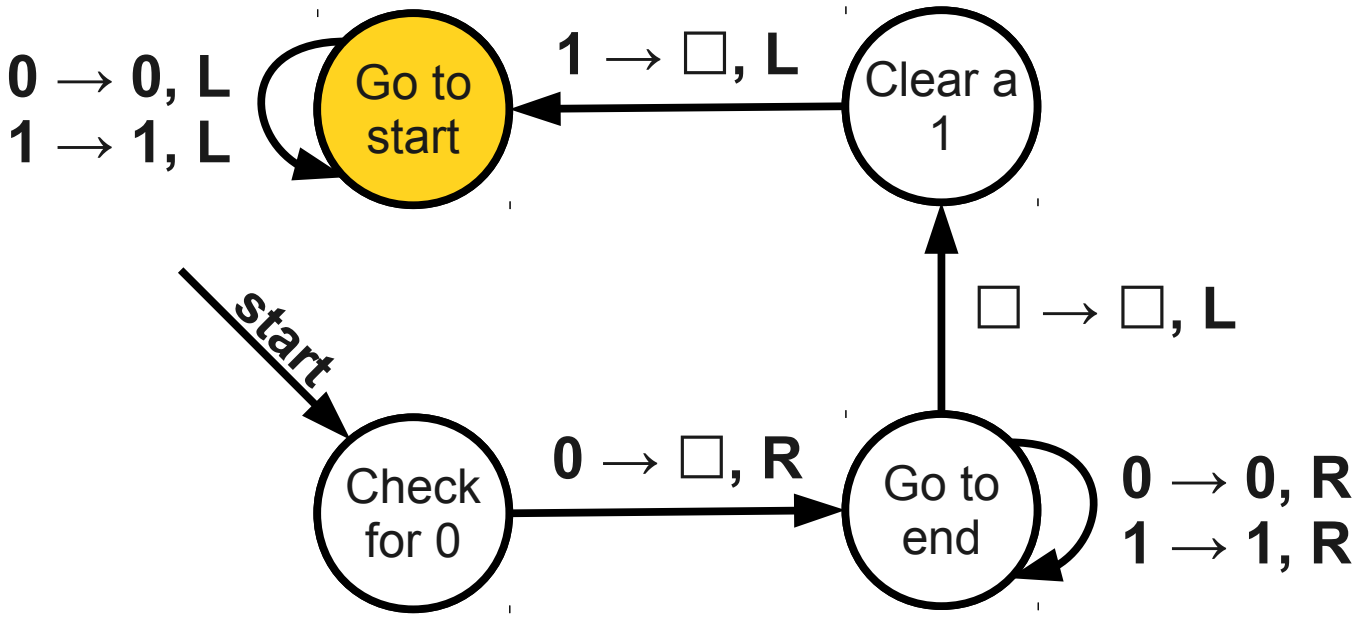

...
0 0 1 1
...

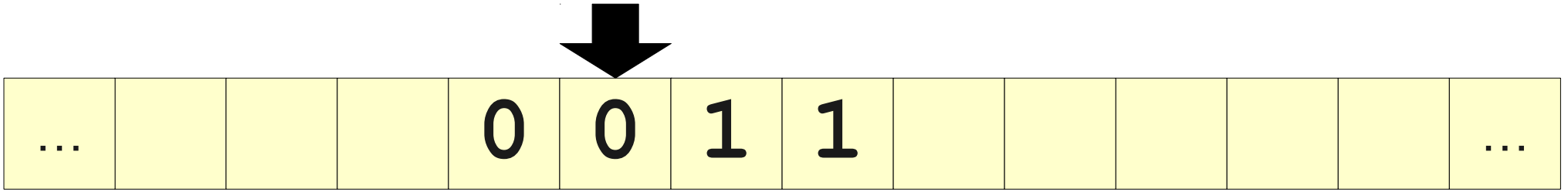

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

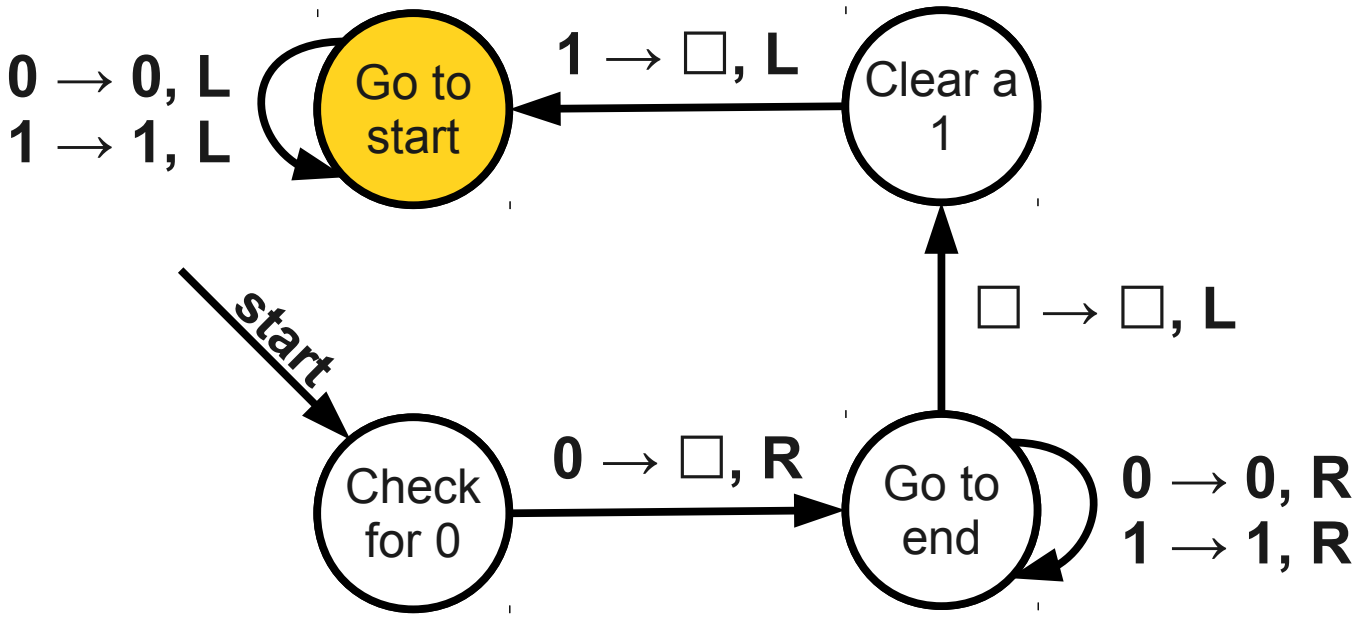

…
0
0
1
1
…

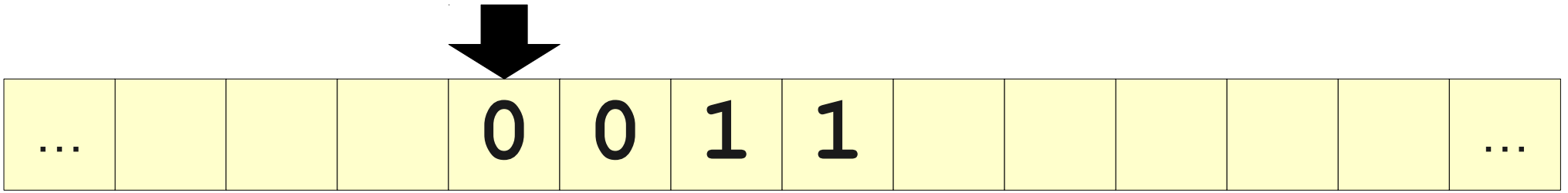

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

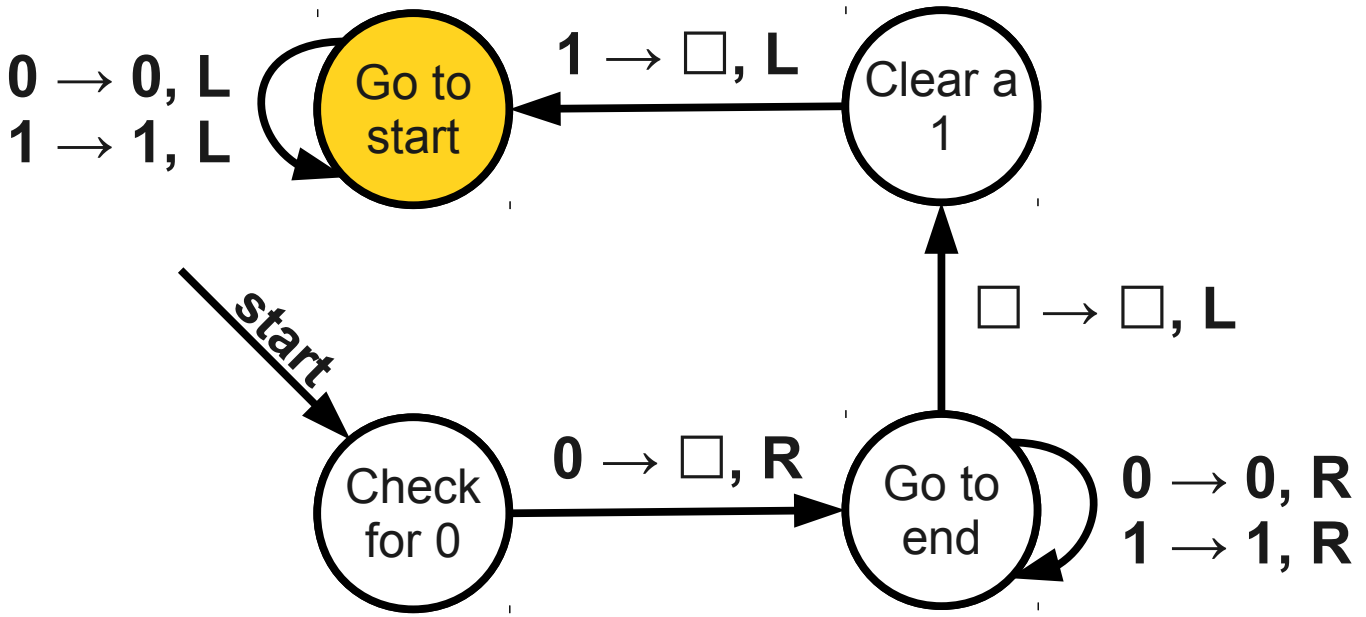

...
0
0
1
1
...

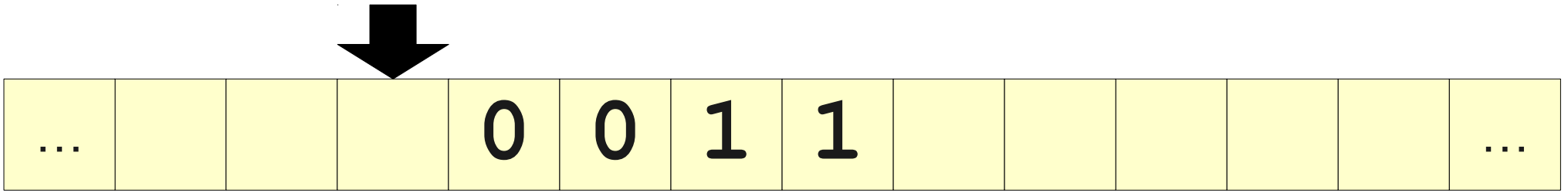

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

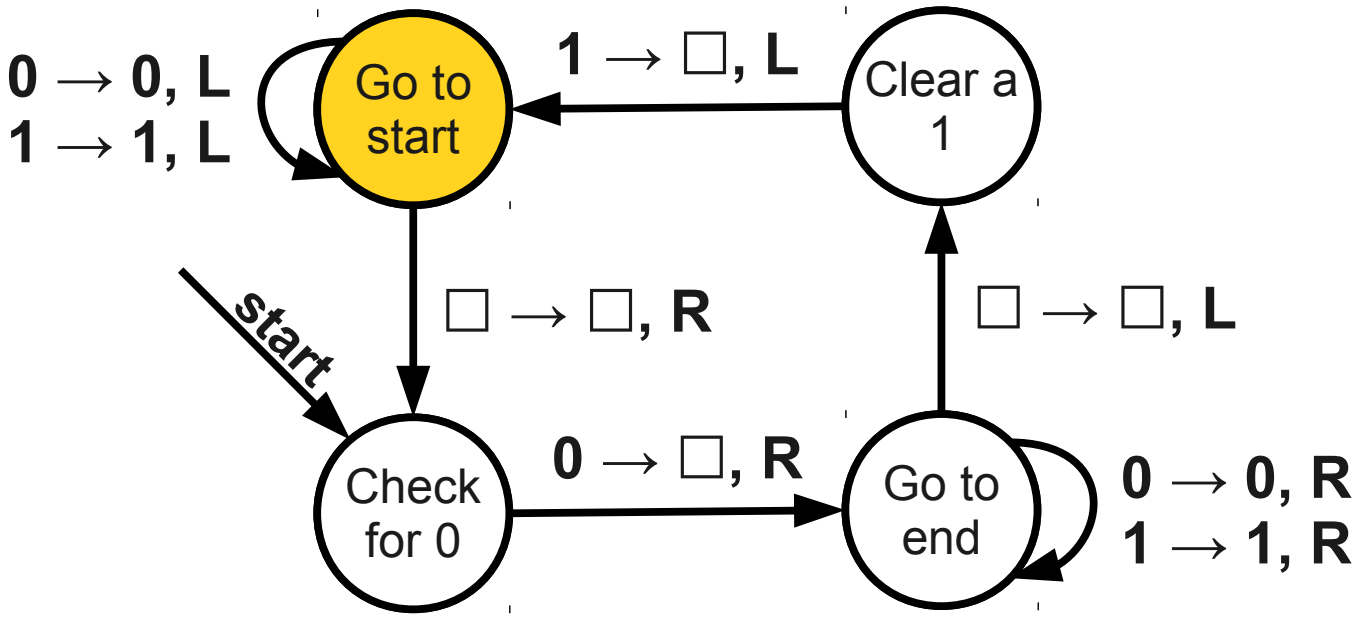

…
0
0
1
1
…

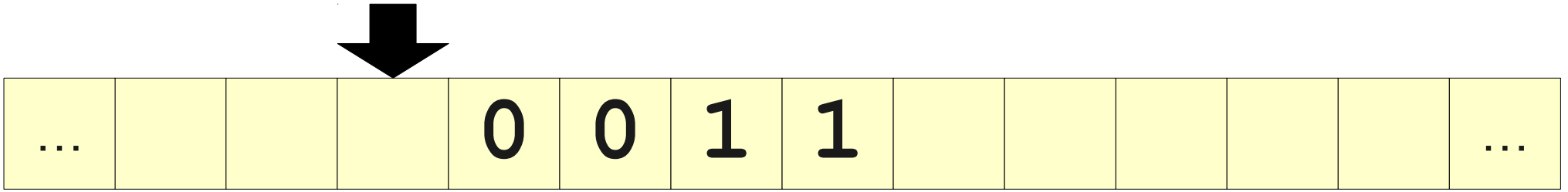

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

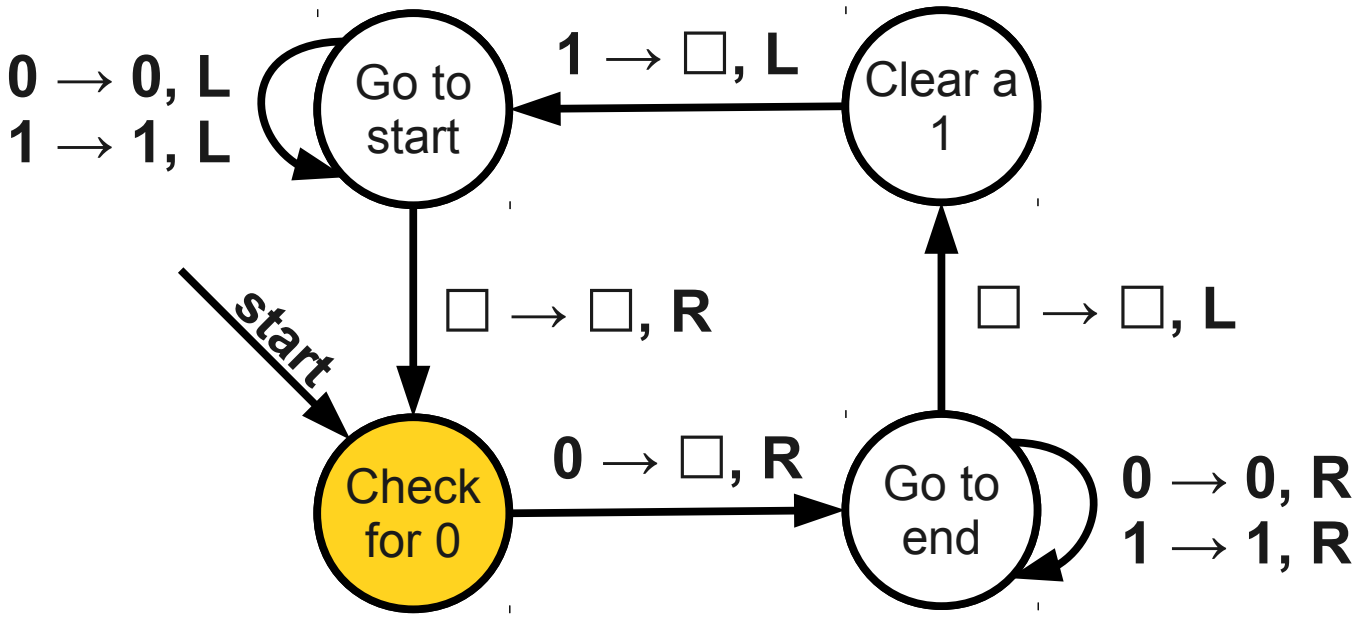

…
0
0
1
1
…

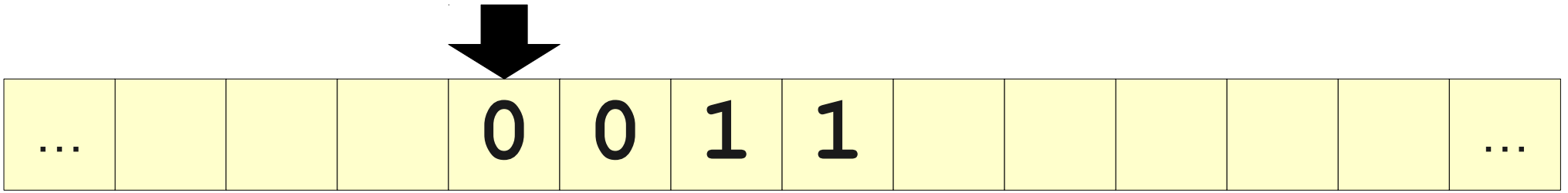

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

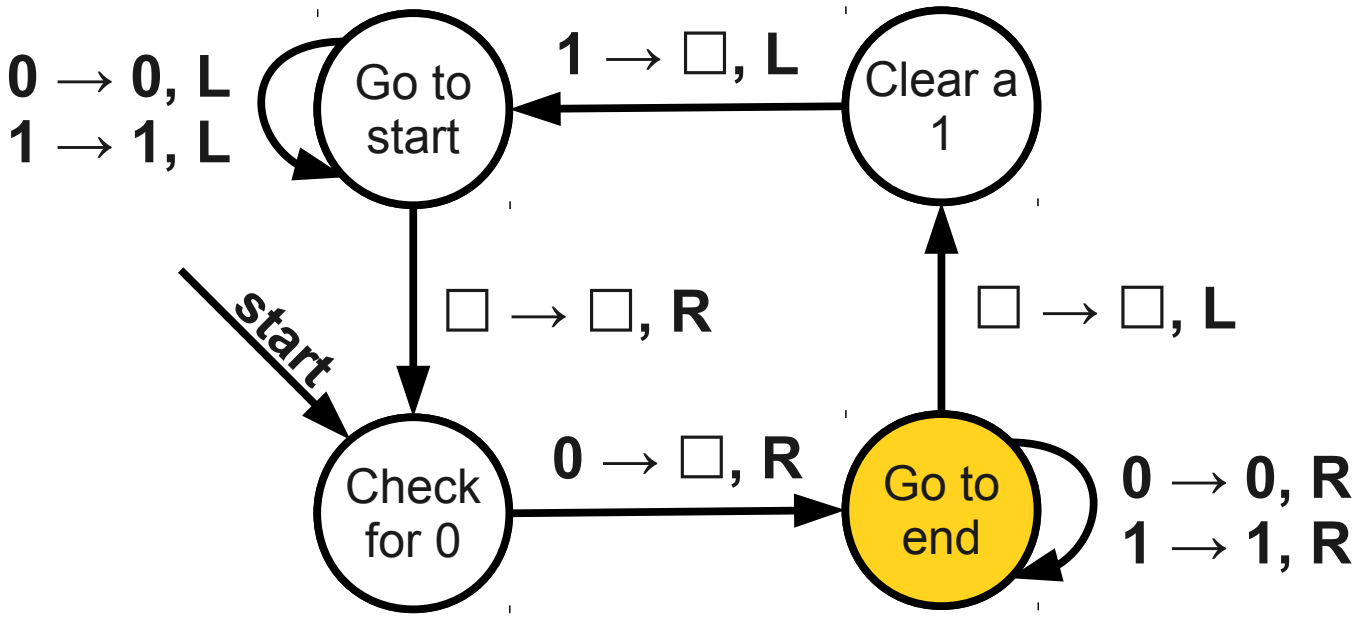

…
0
0
1
1
…

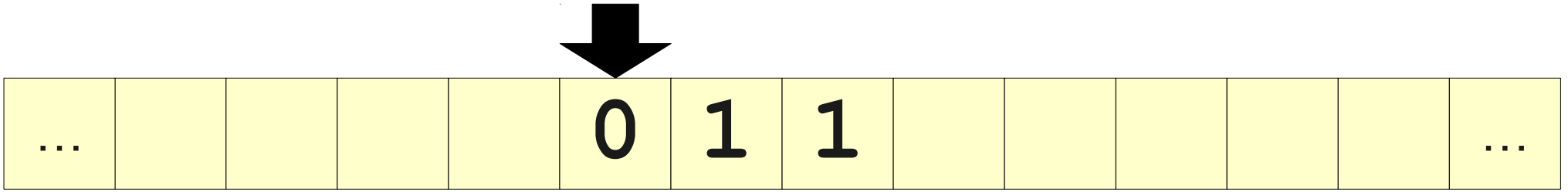

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

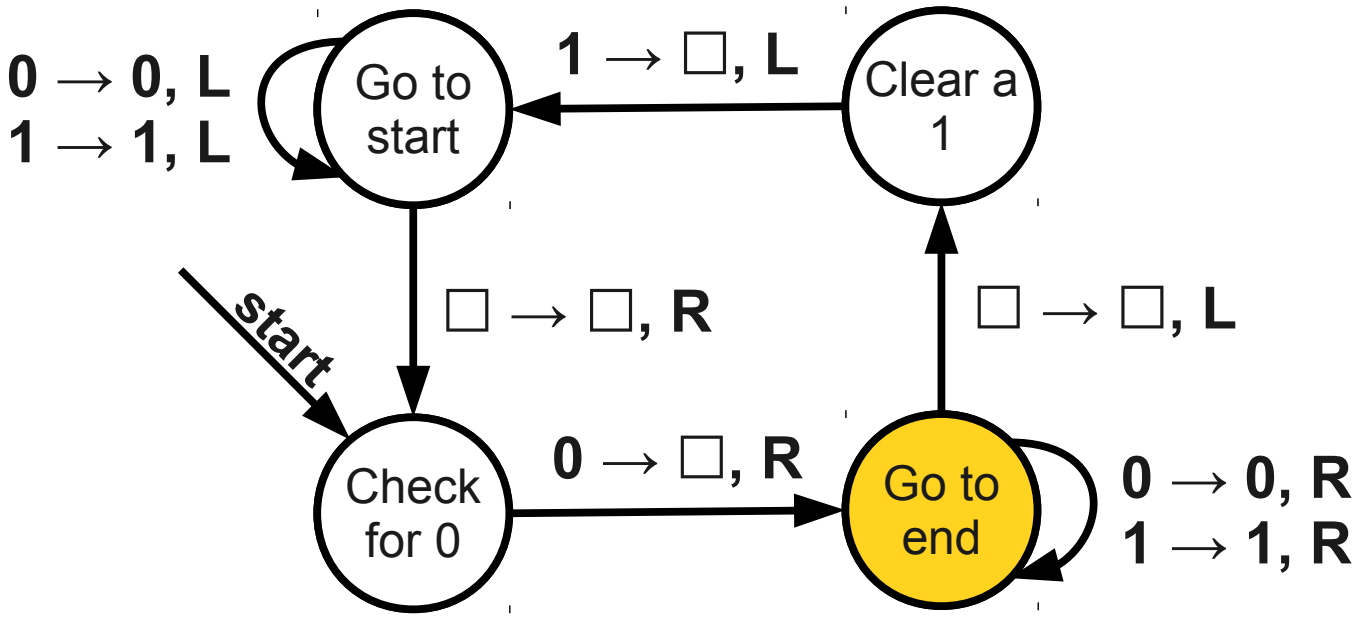

...
0
0
1
1
...

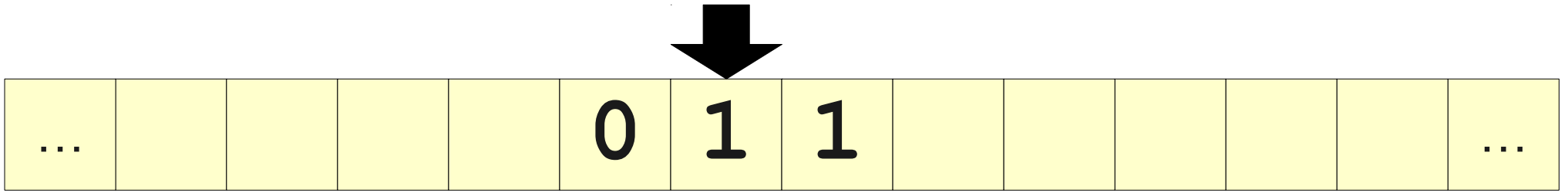

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

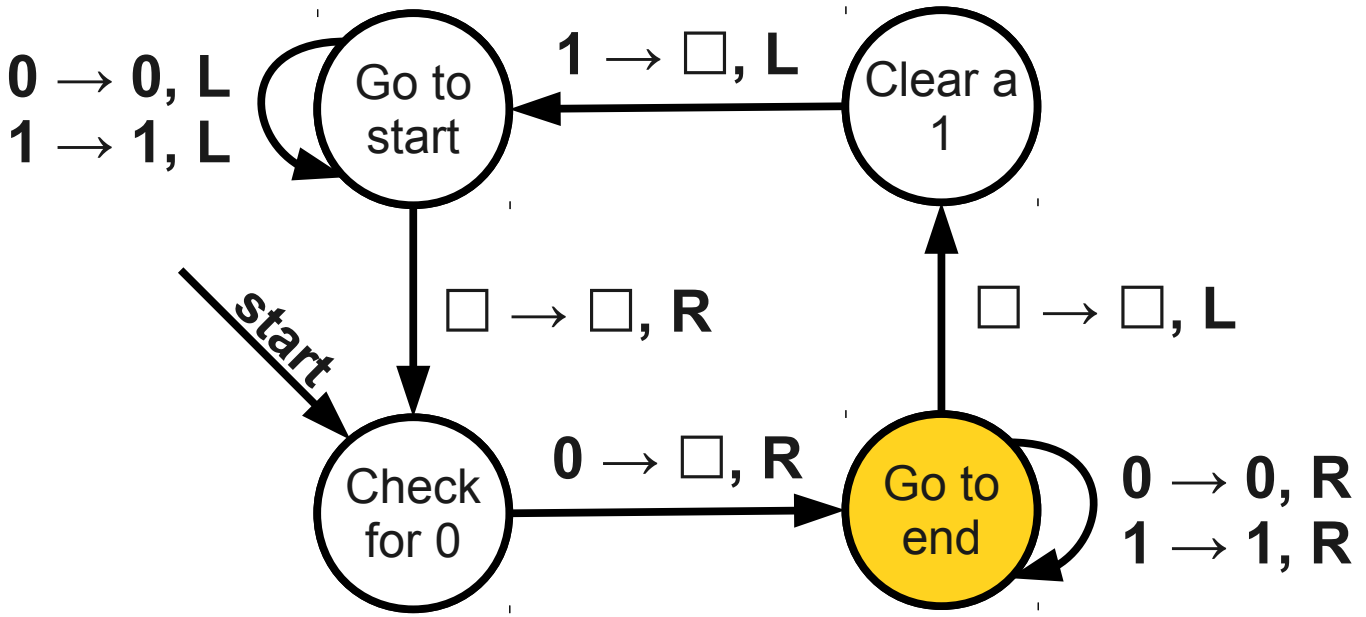

...
0
0
1
1
...

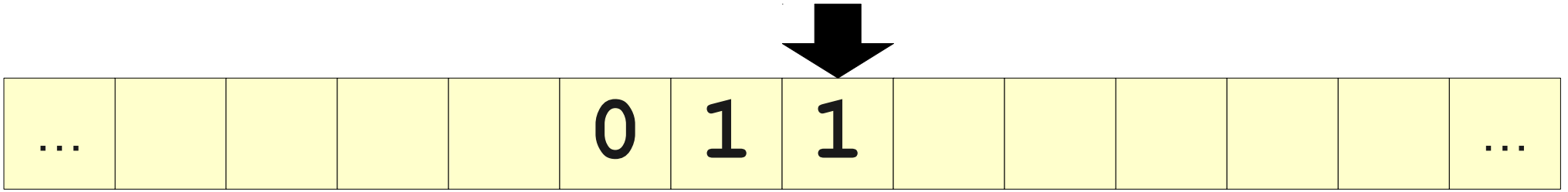

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

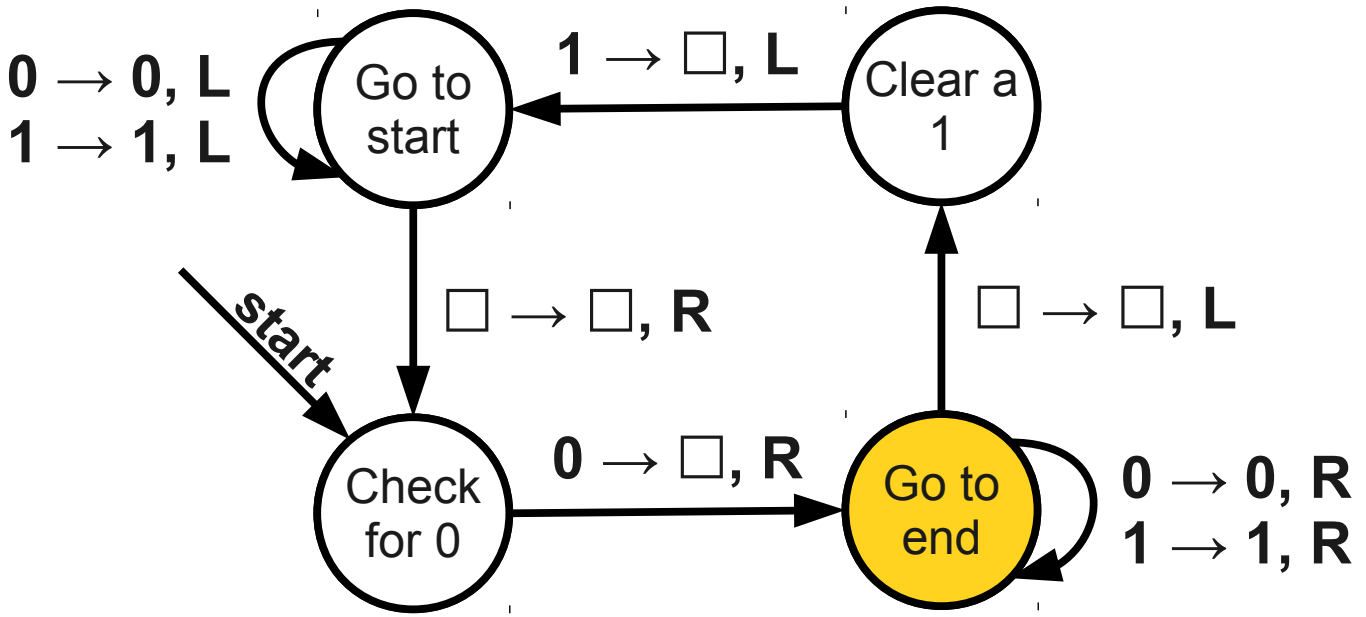

...
0
1
1
...

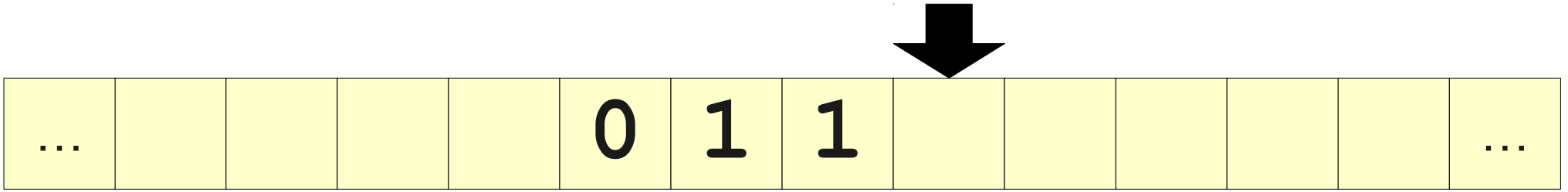

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

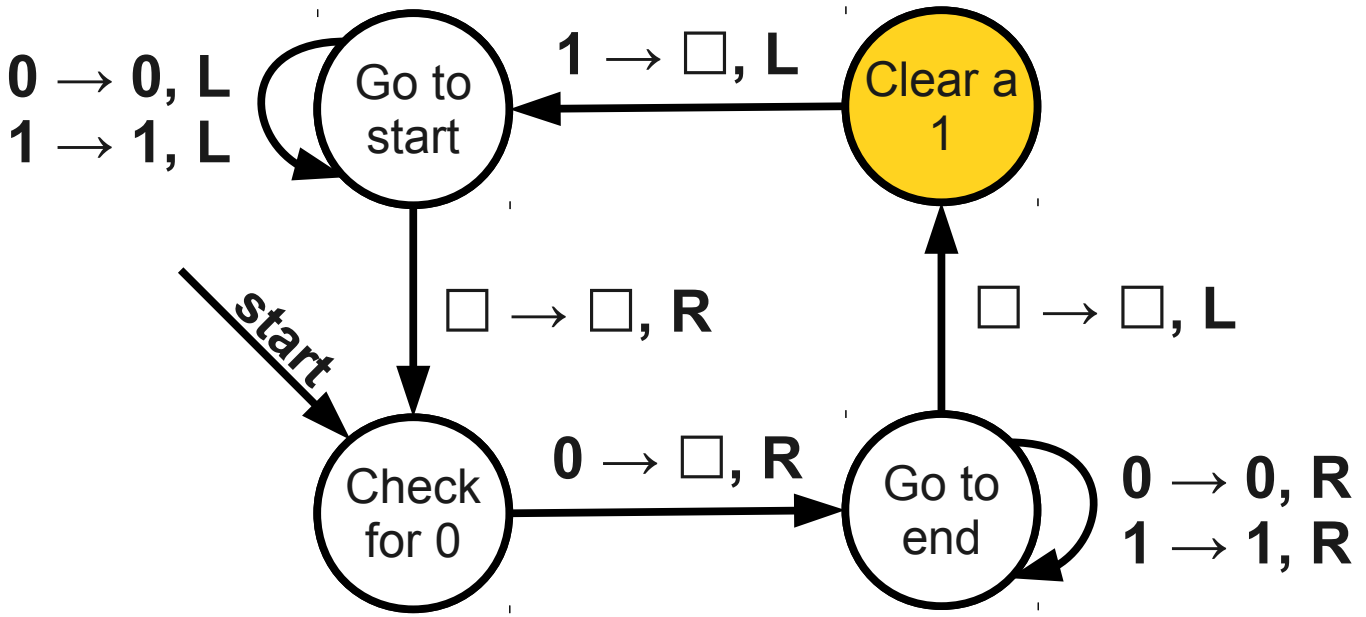

...
0
1
1
...

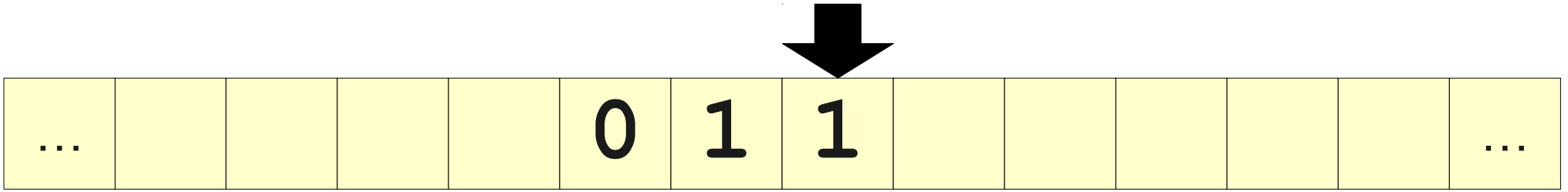

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

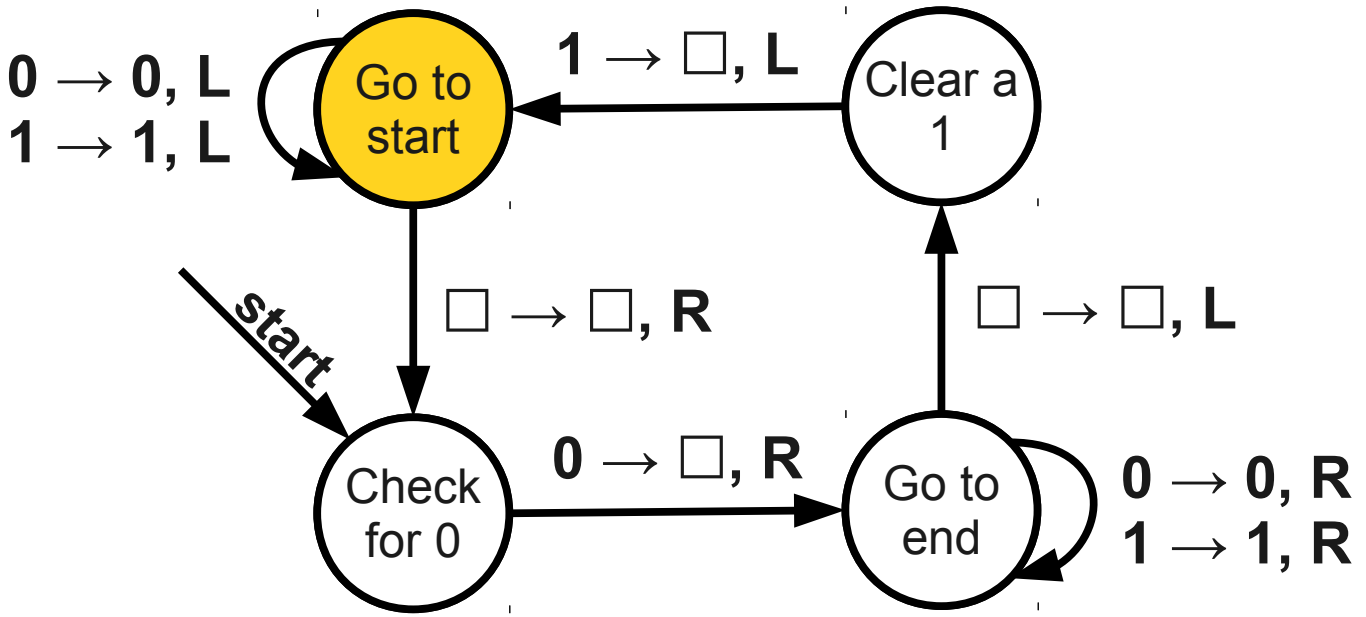

...
0
1
1
...

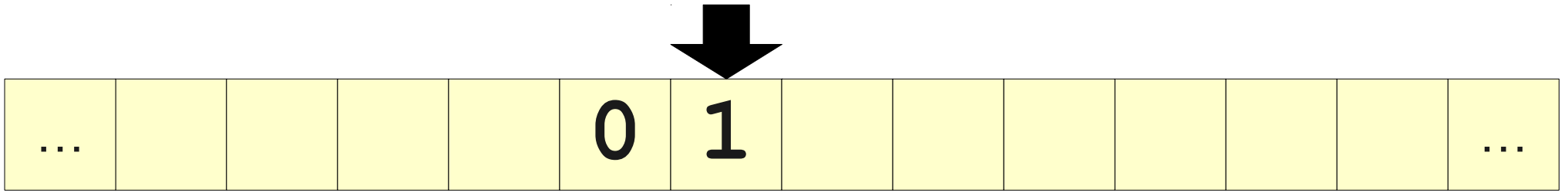

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

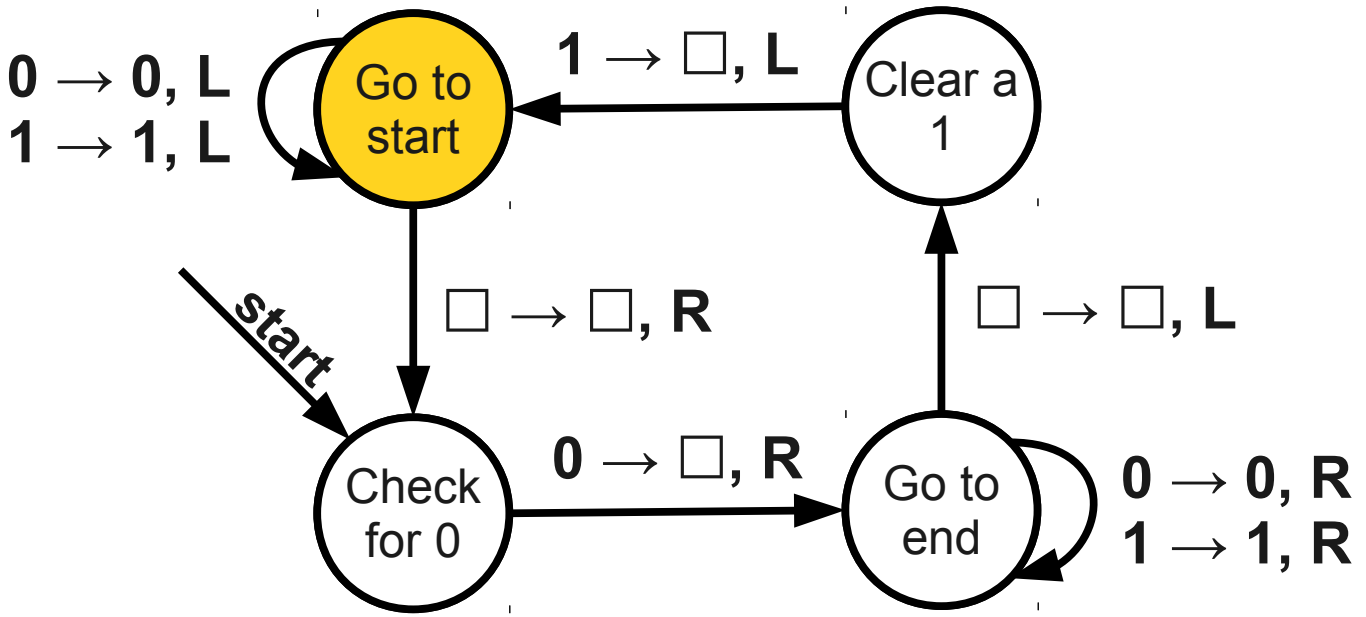

...
0
1
1
...

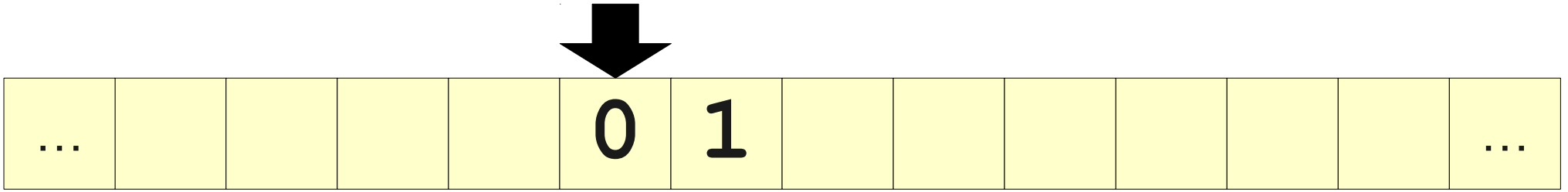

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

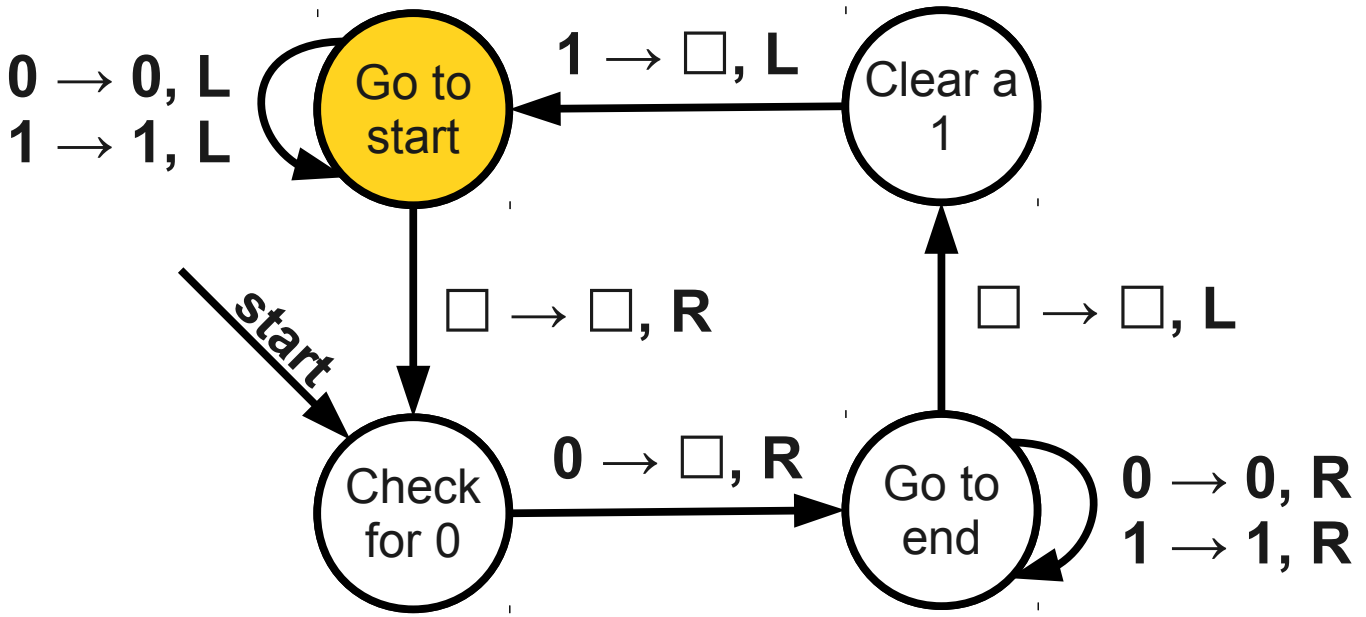

...
0
1
1
...

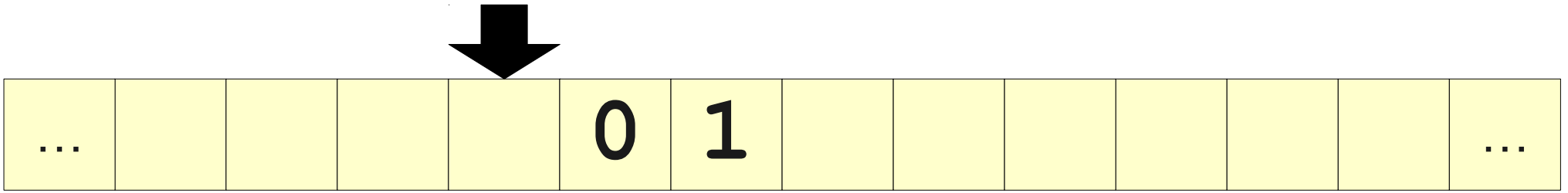

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

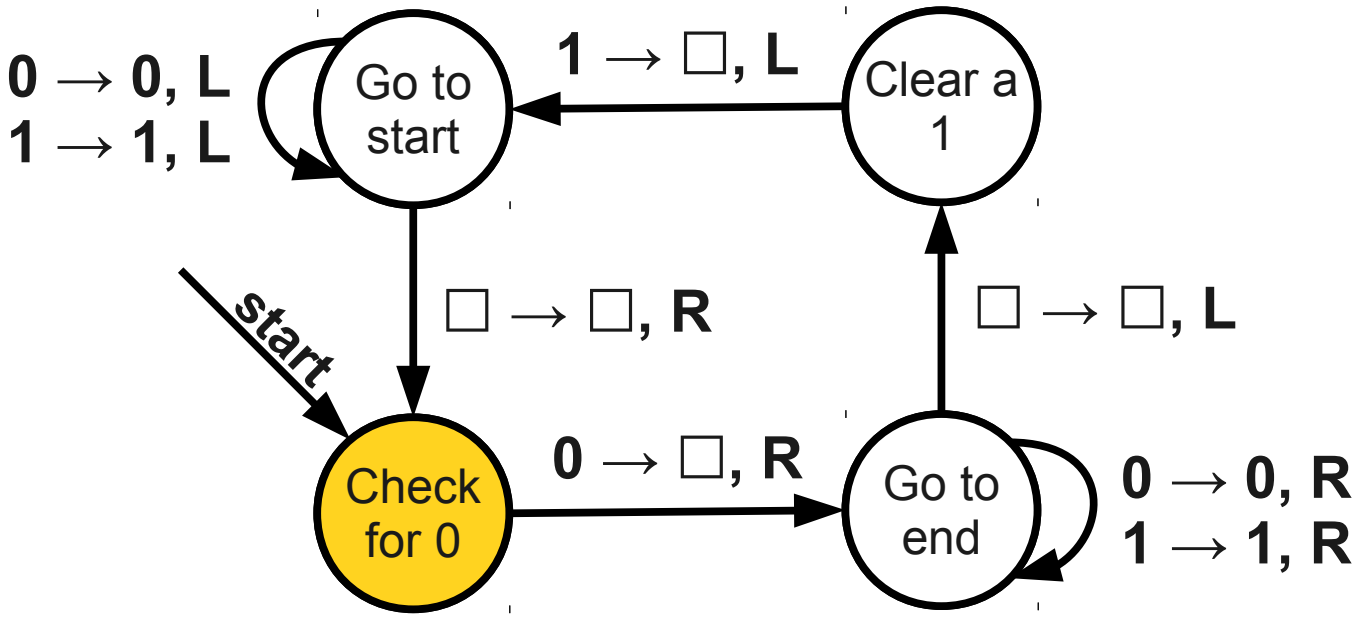

...
0
1
...

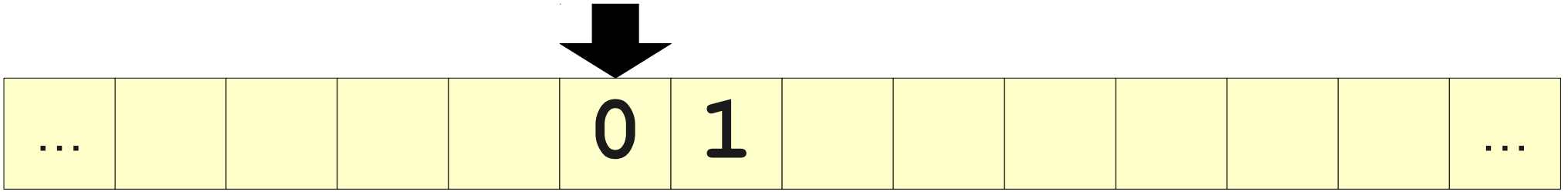

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

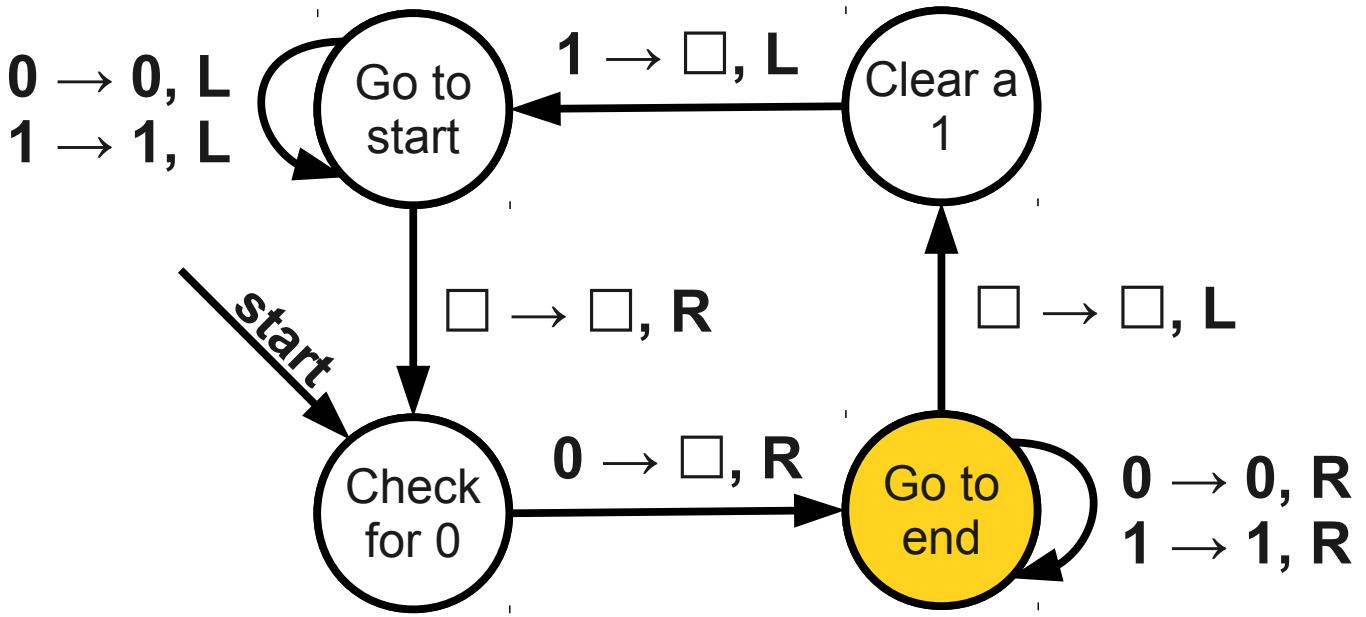

...
0
1
...

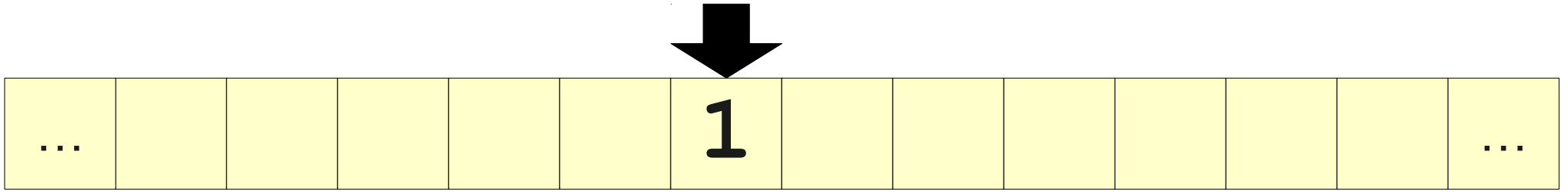

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

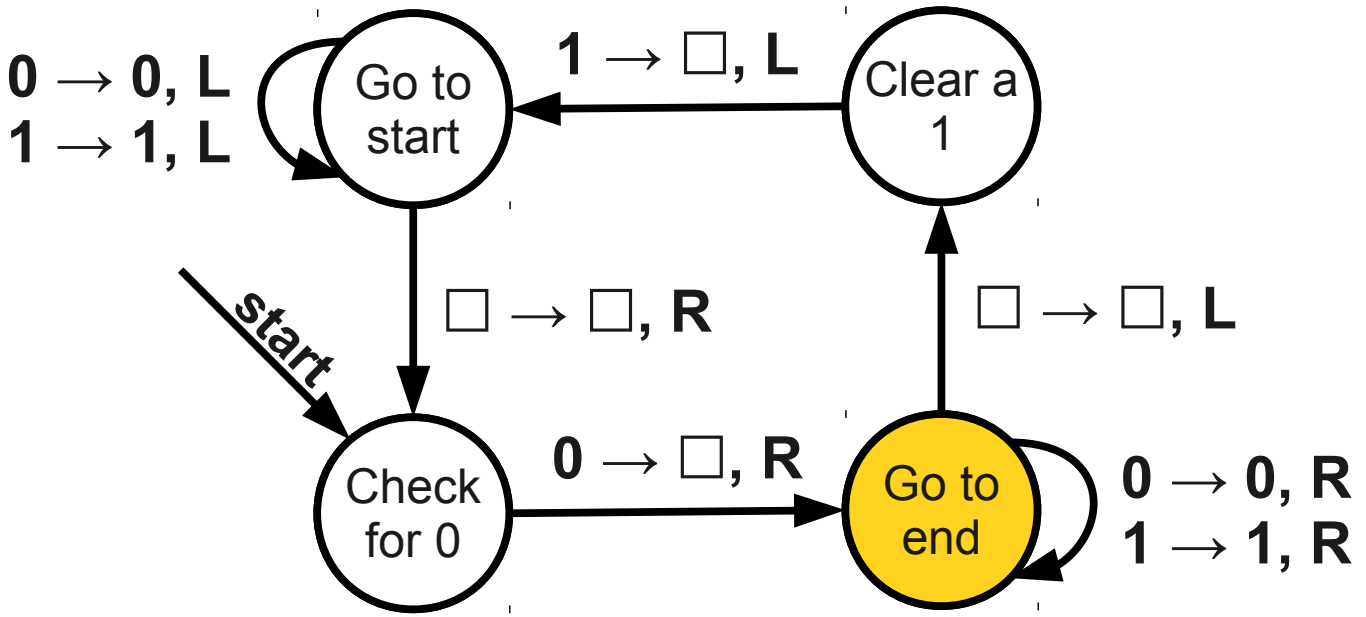

...
0
1
...

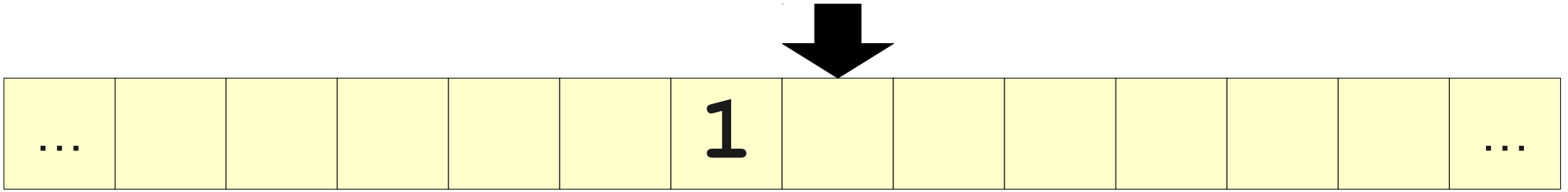

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

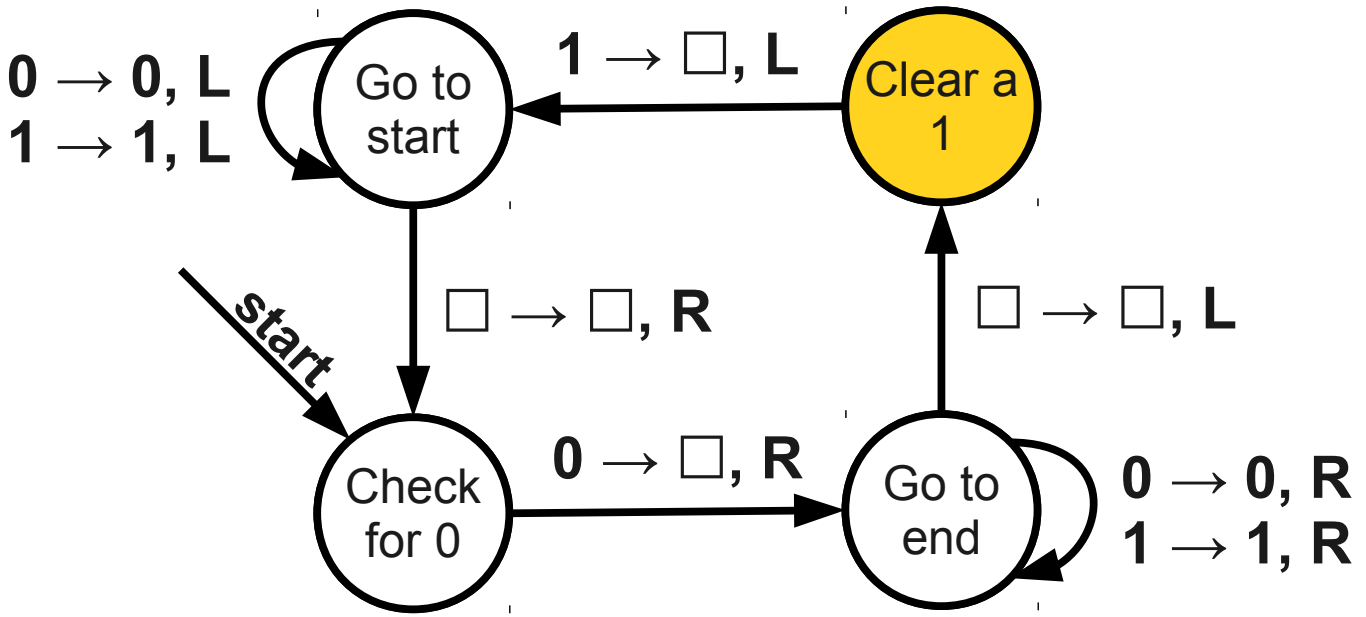

...
0
1
...

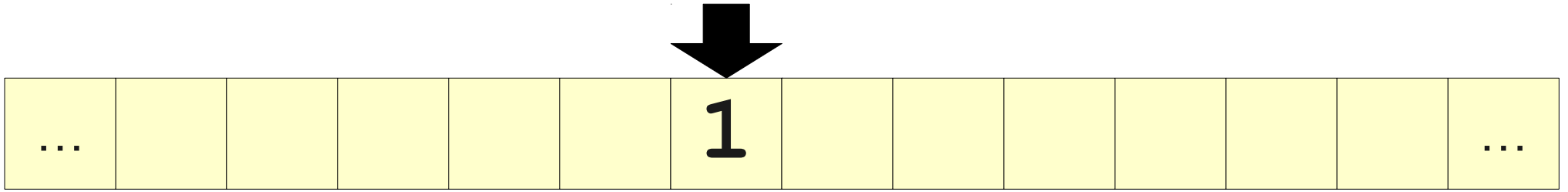

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

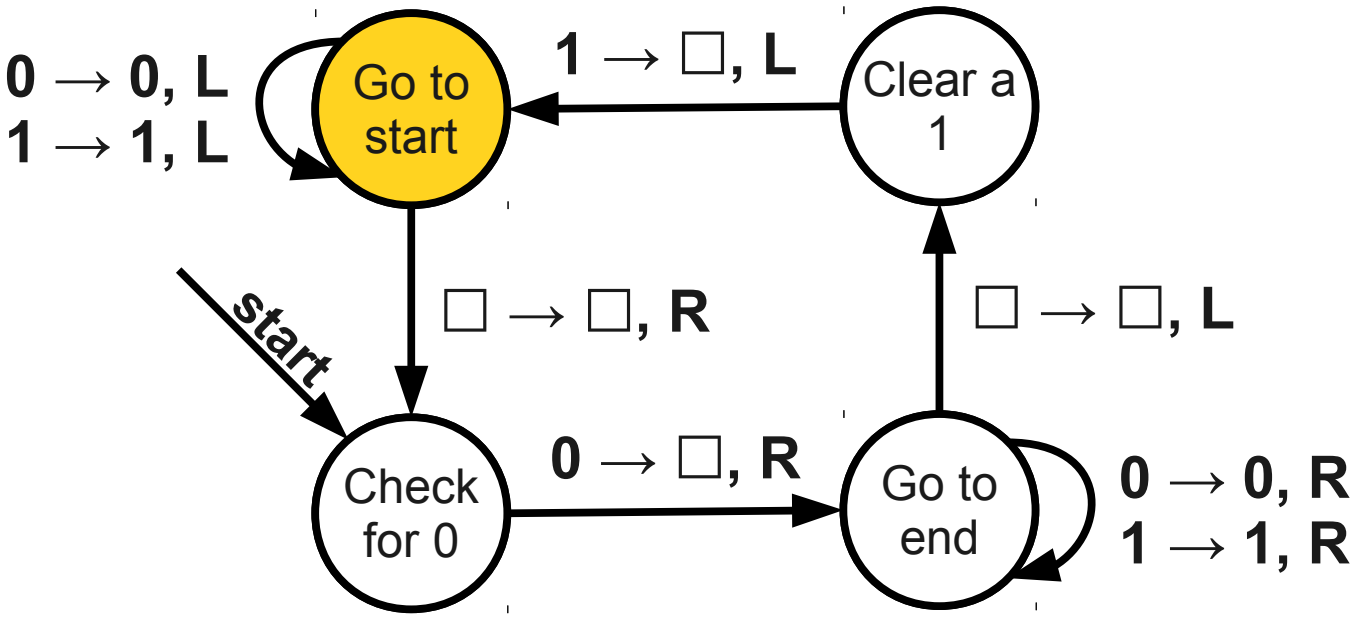

...
1
...

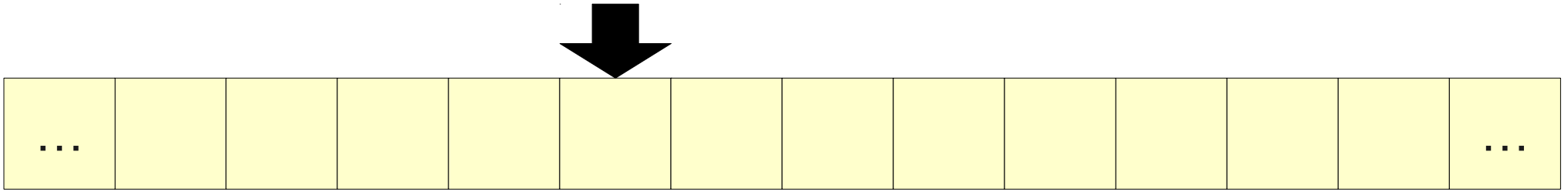

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

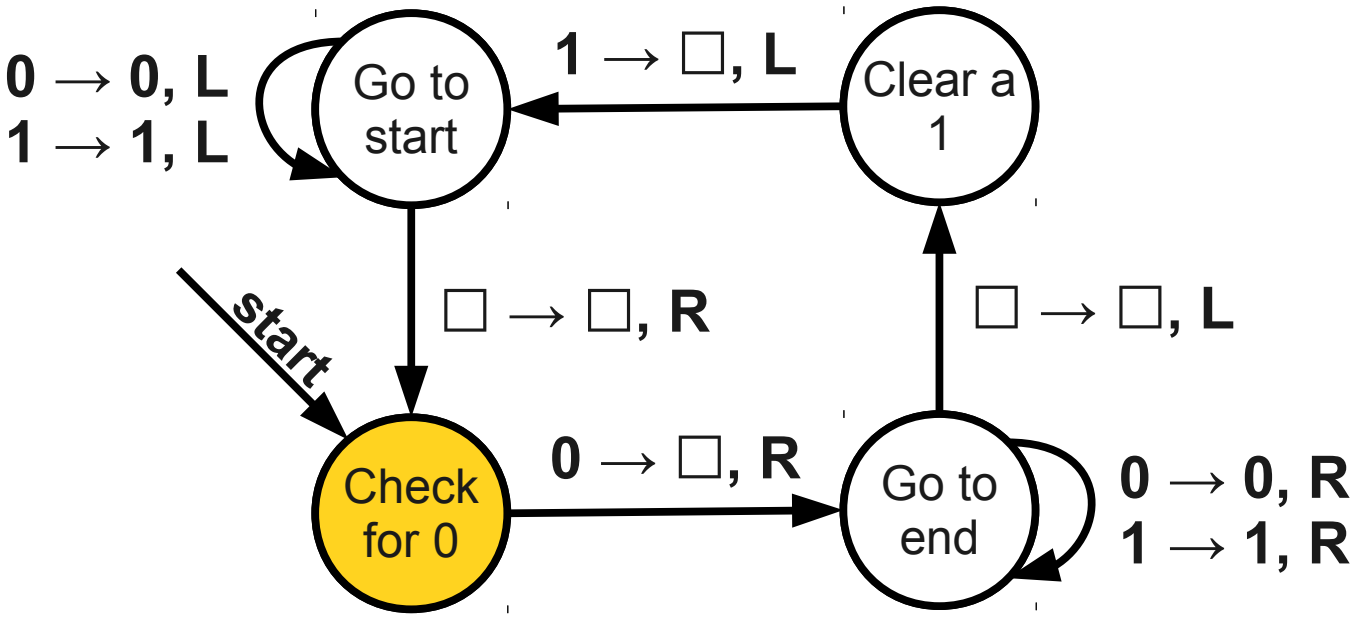

...
1
...

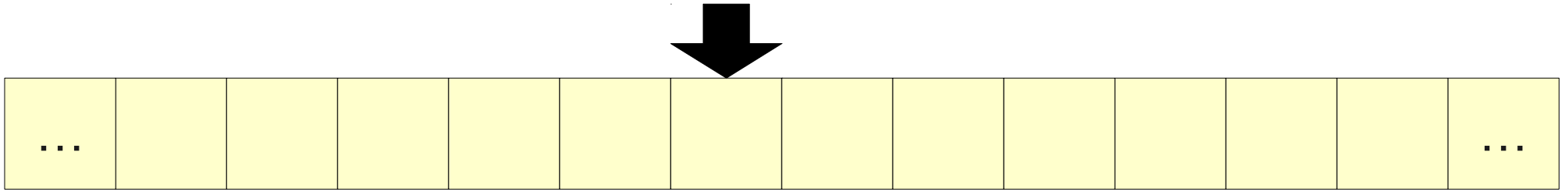

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

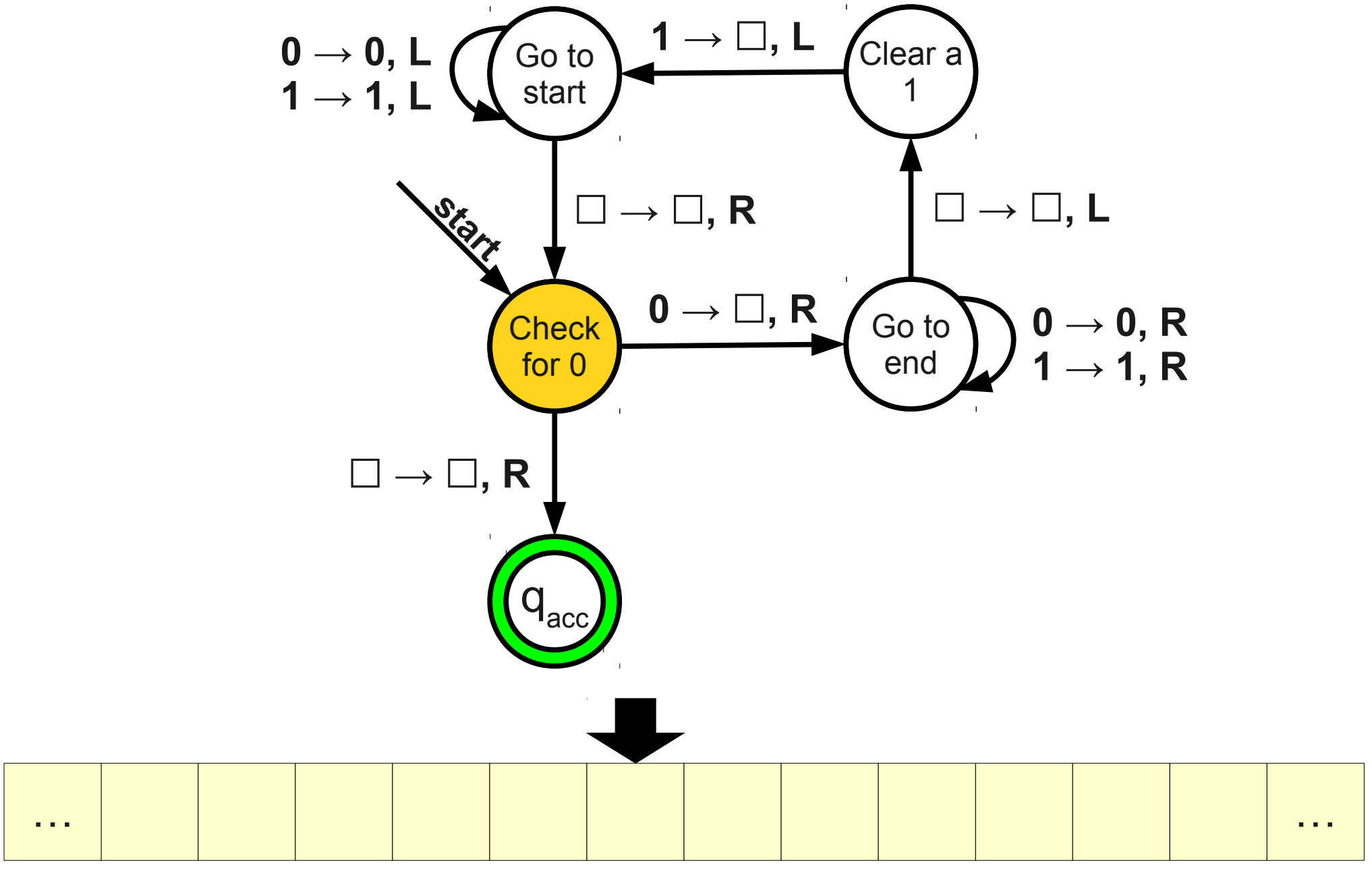

...
1
...

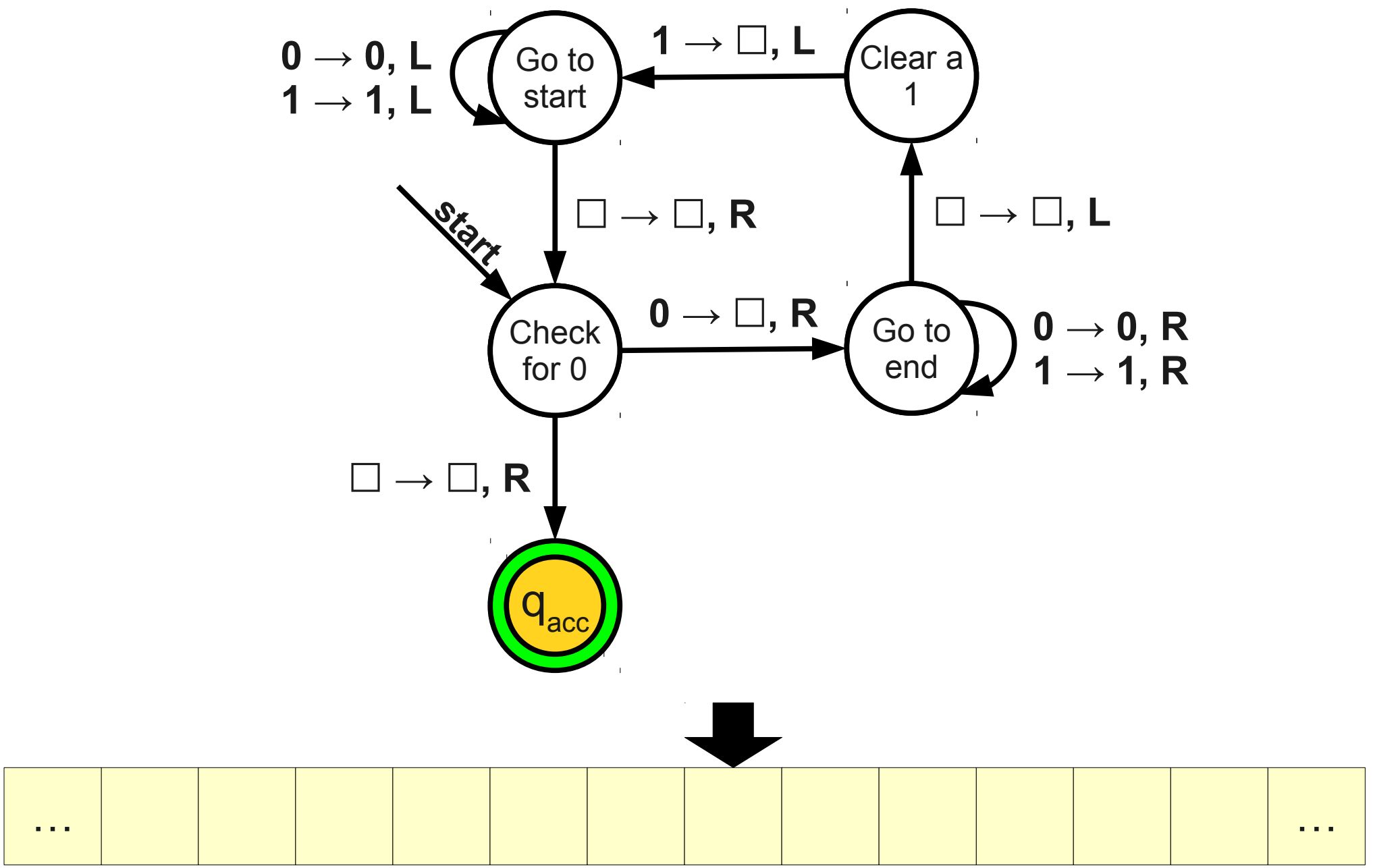

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

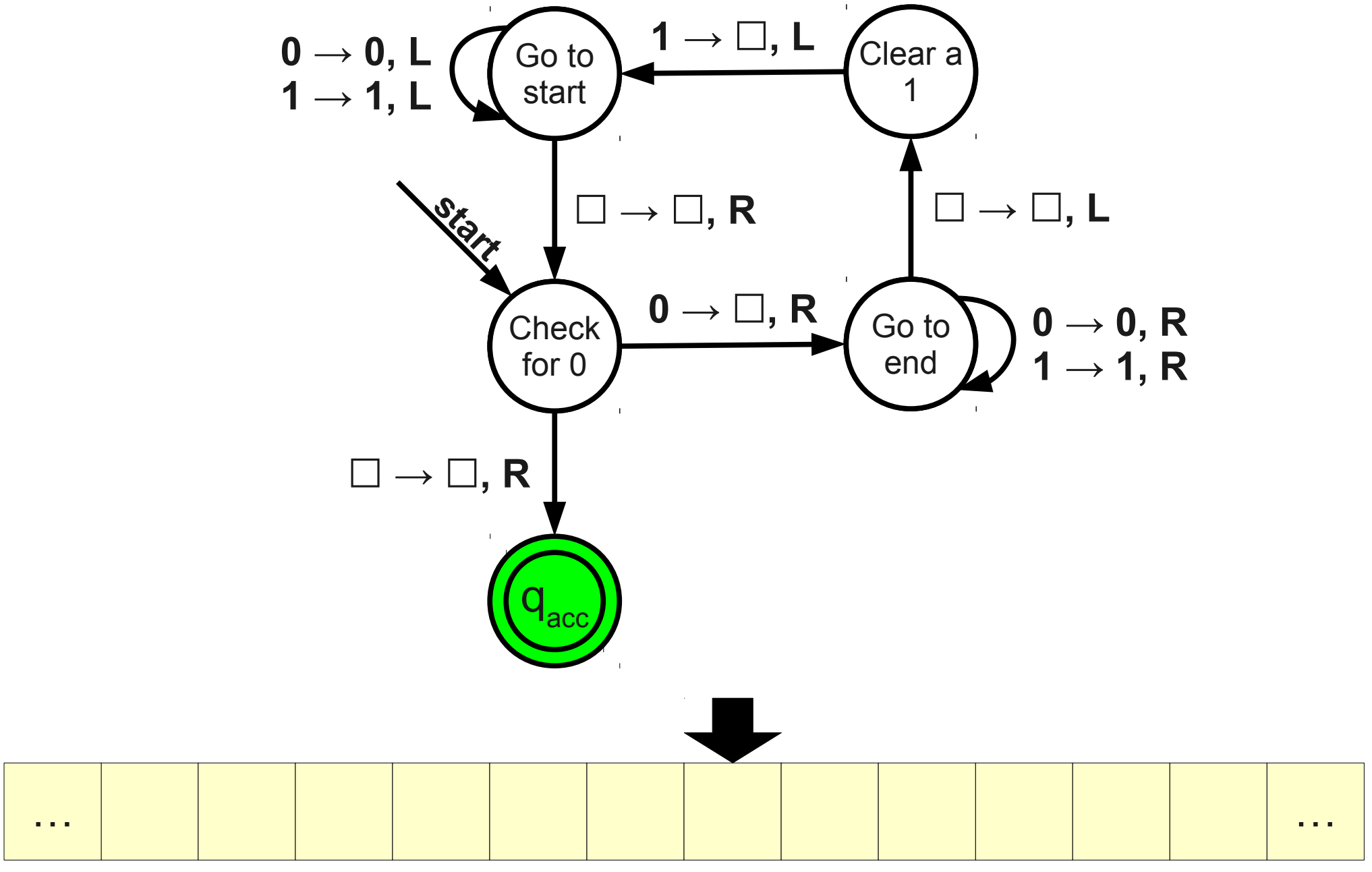

...
...

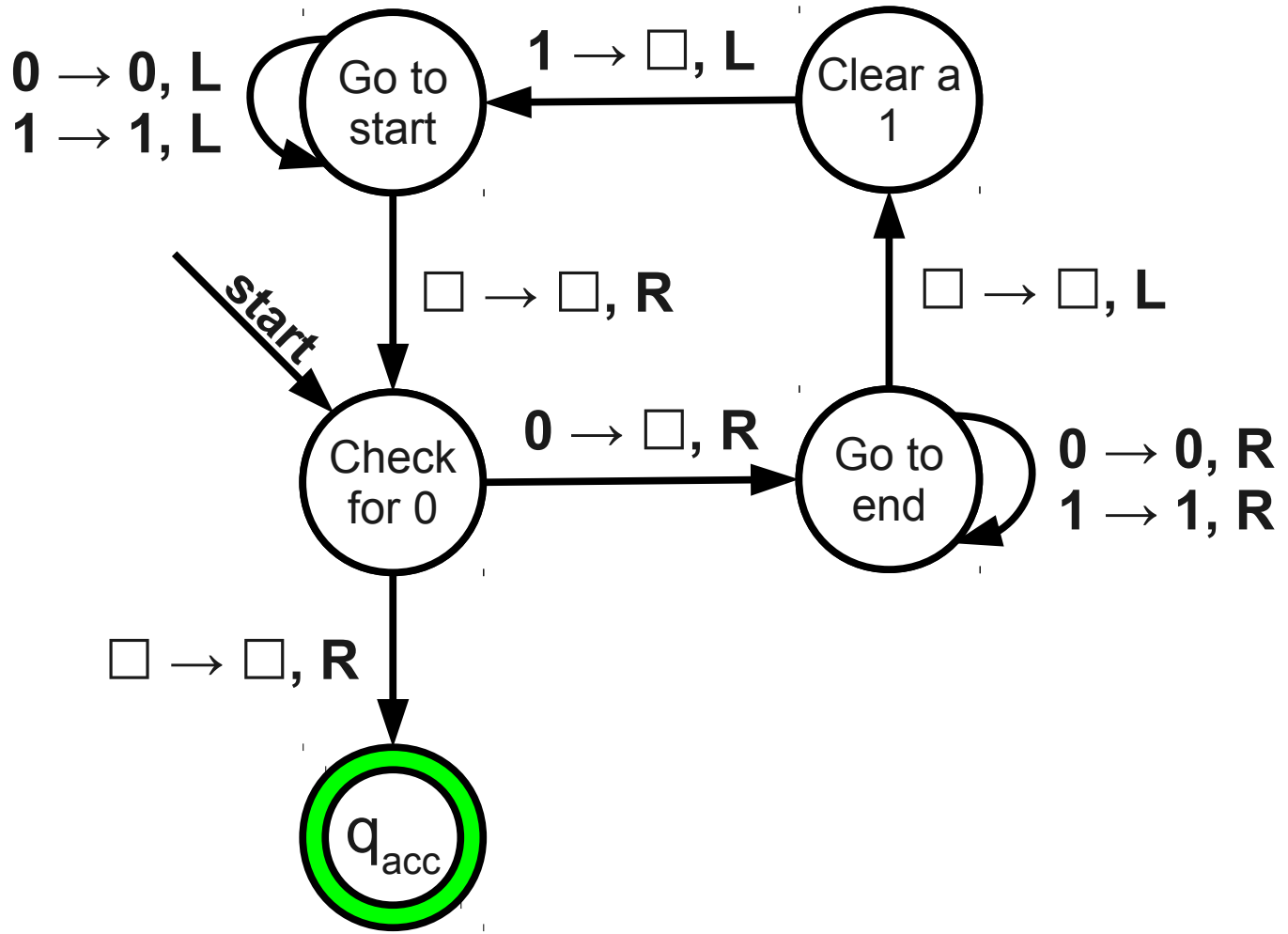

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R

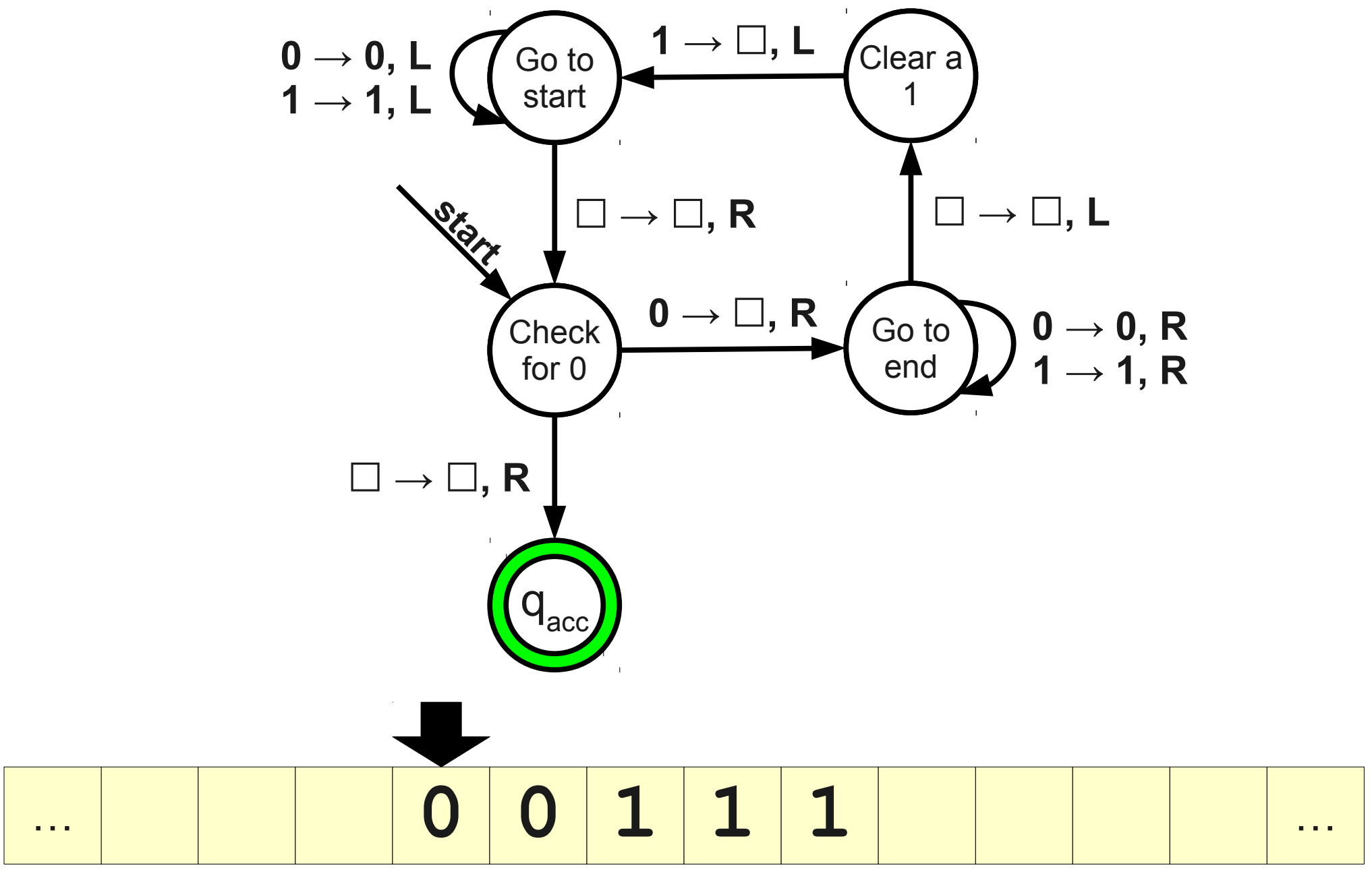

...
...

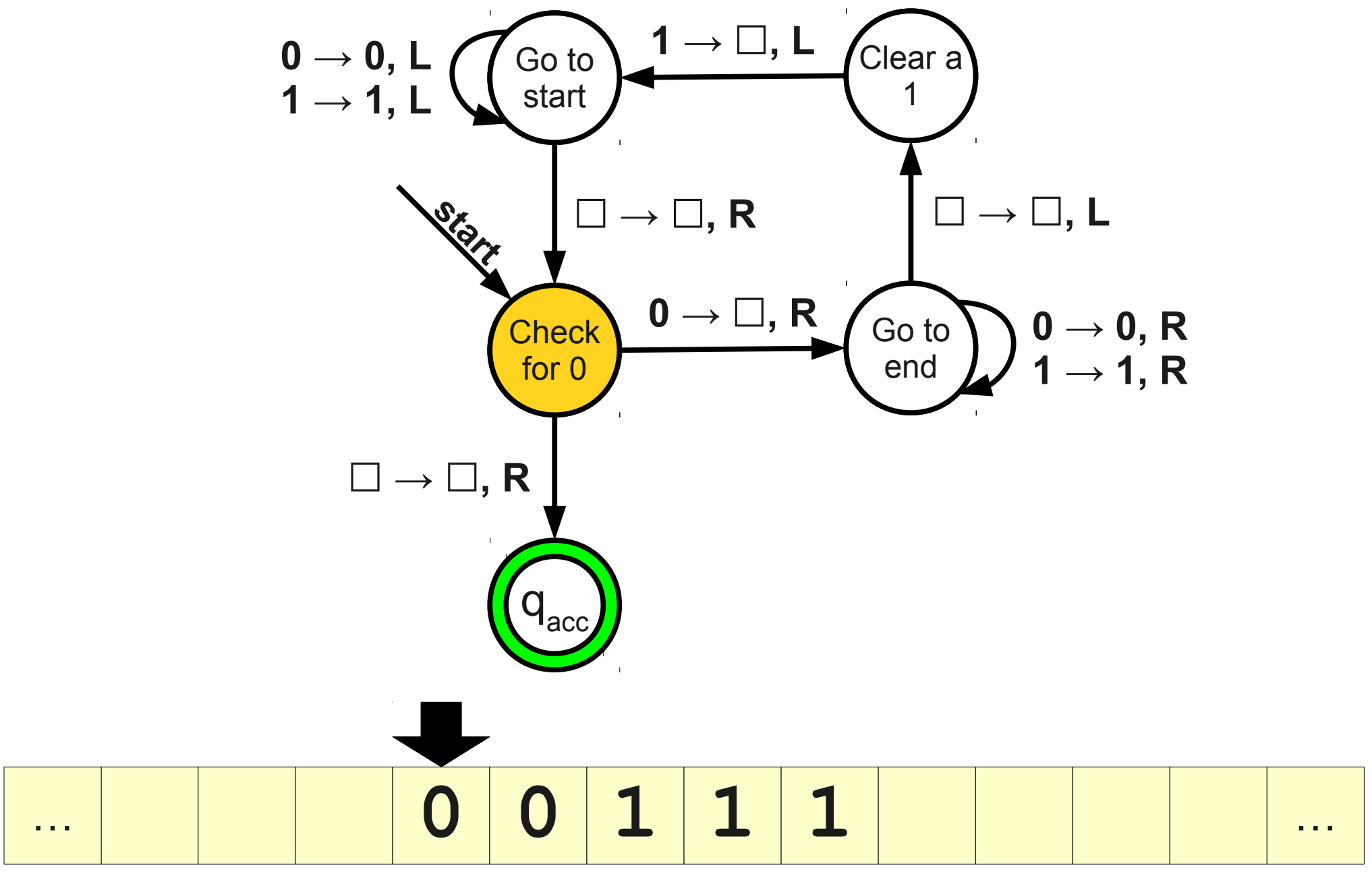

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
...

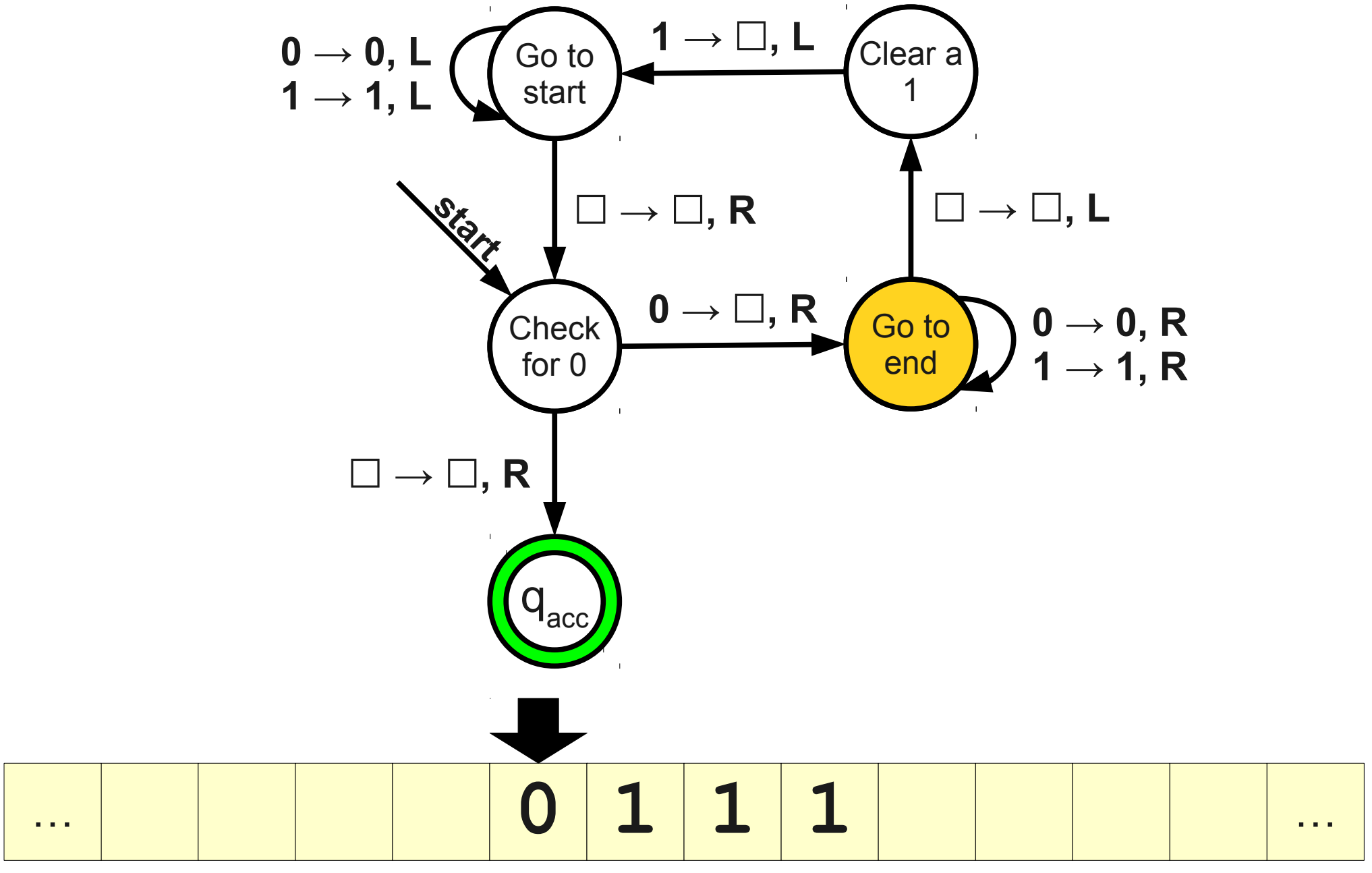

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_acc
...
...

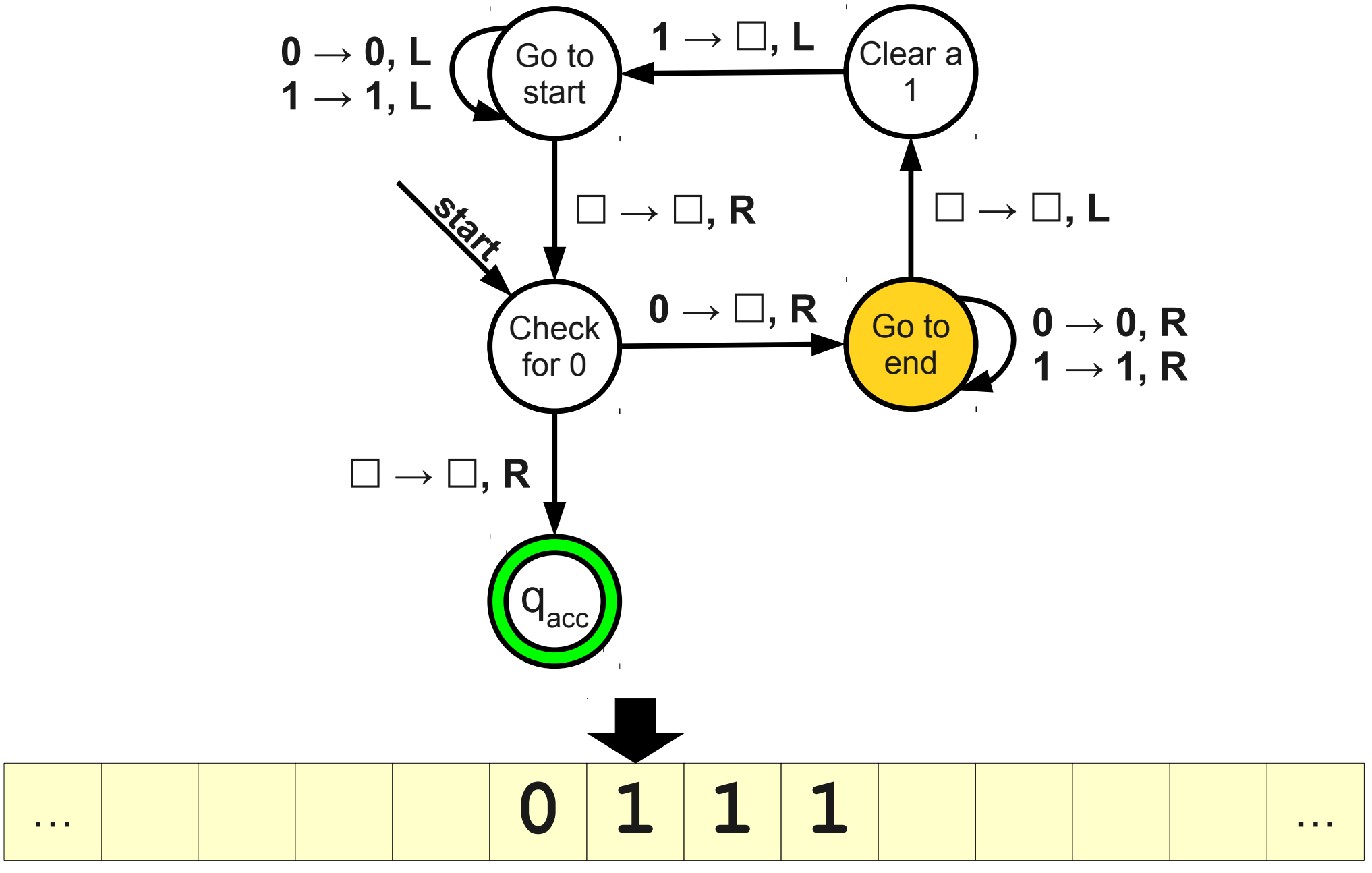

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
...

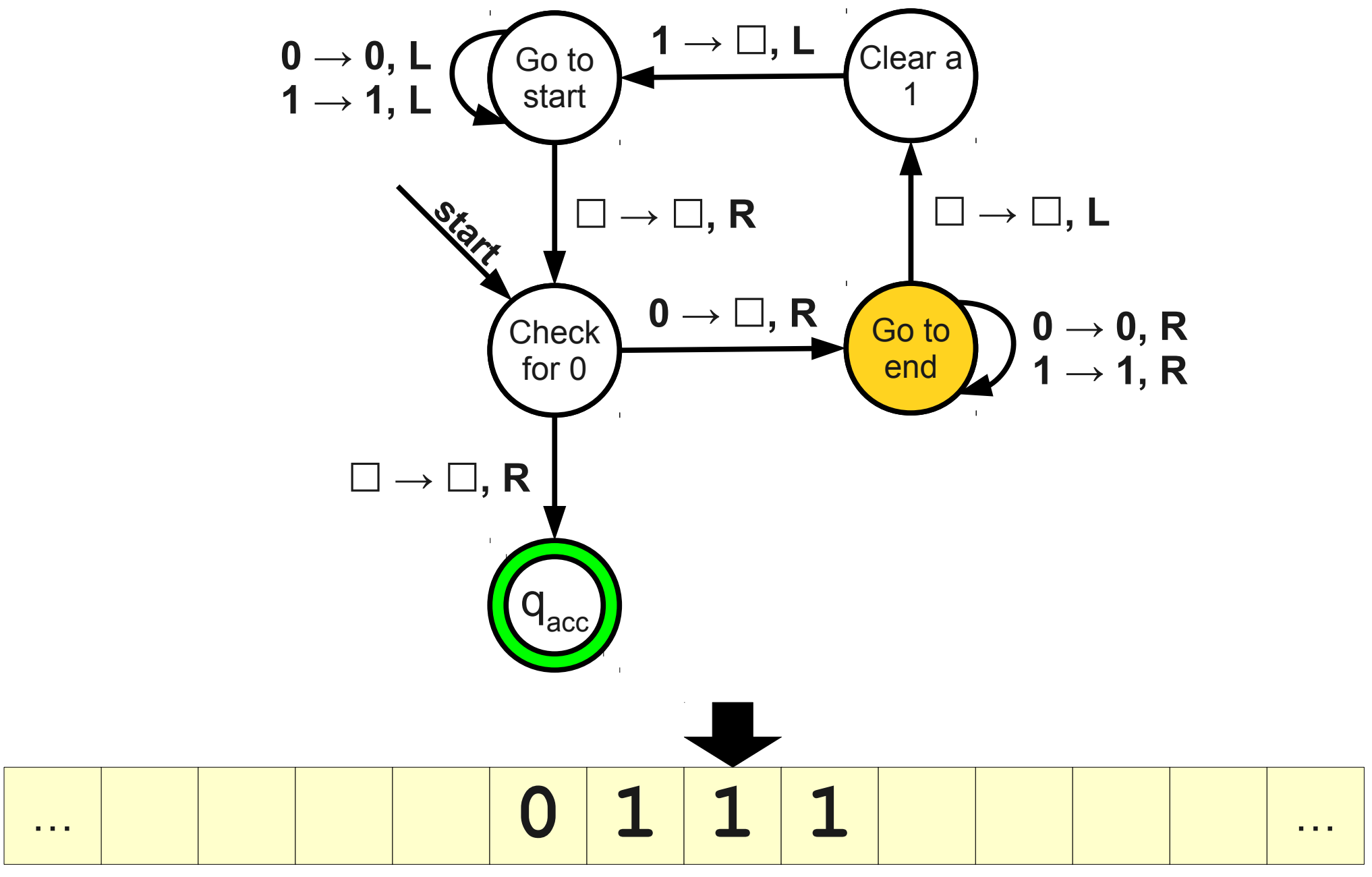

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}

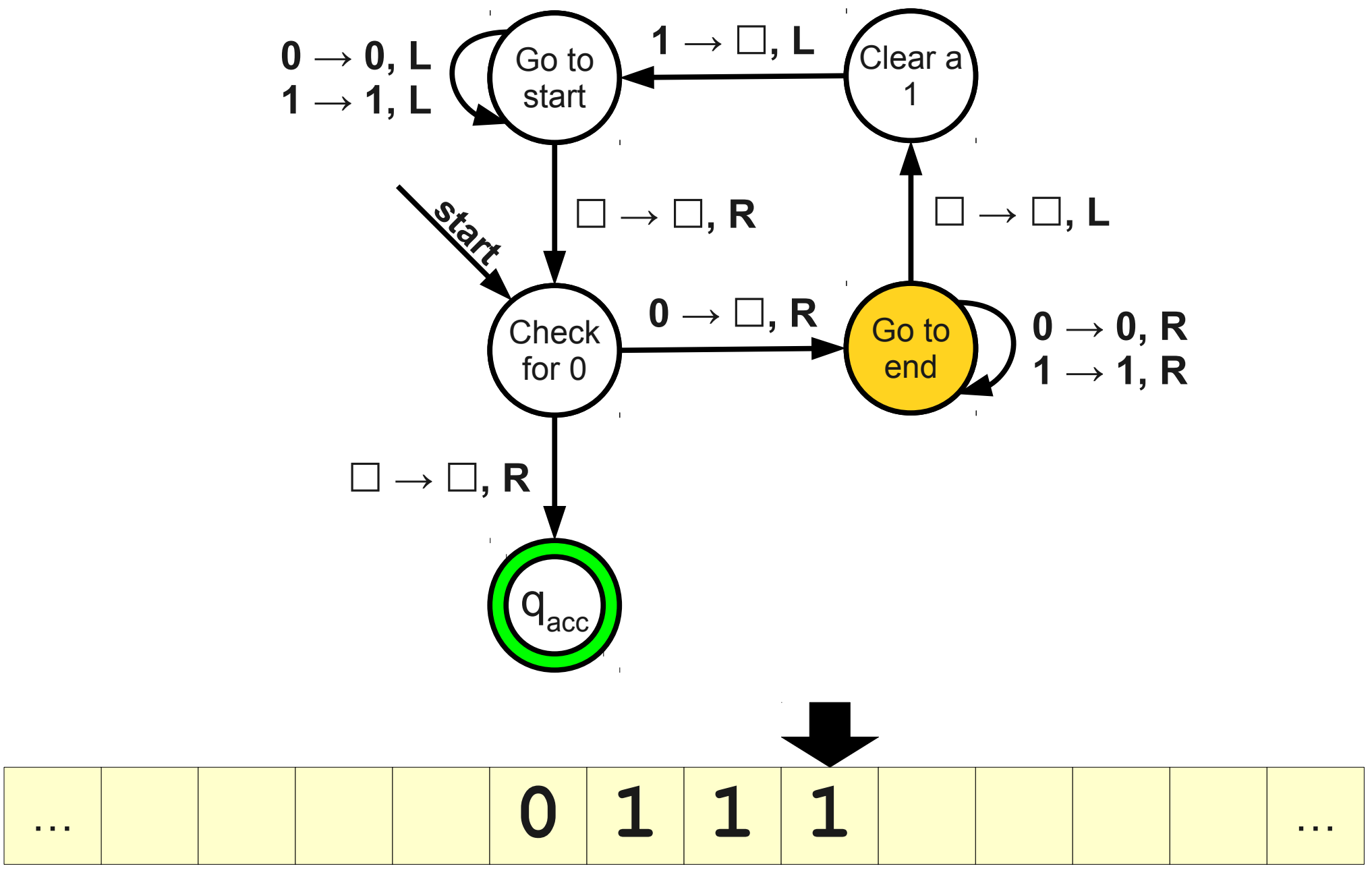

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
0
1
1
1
...

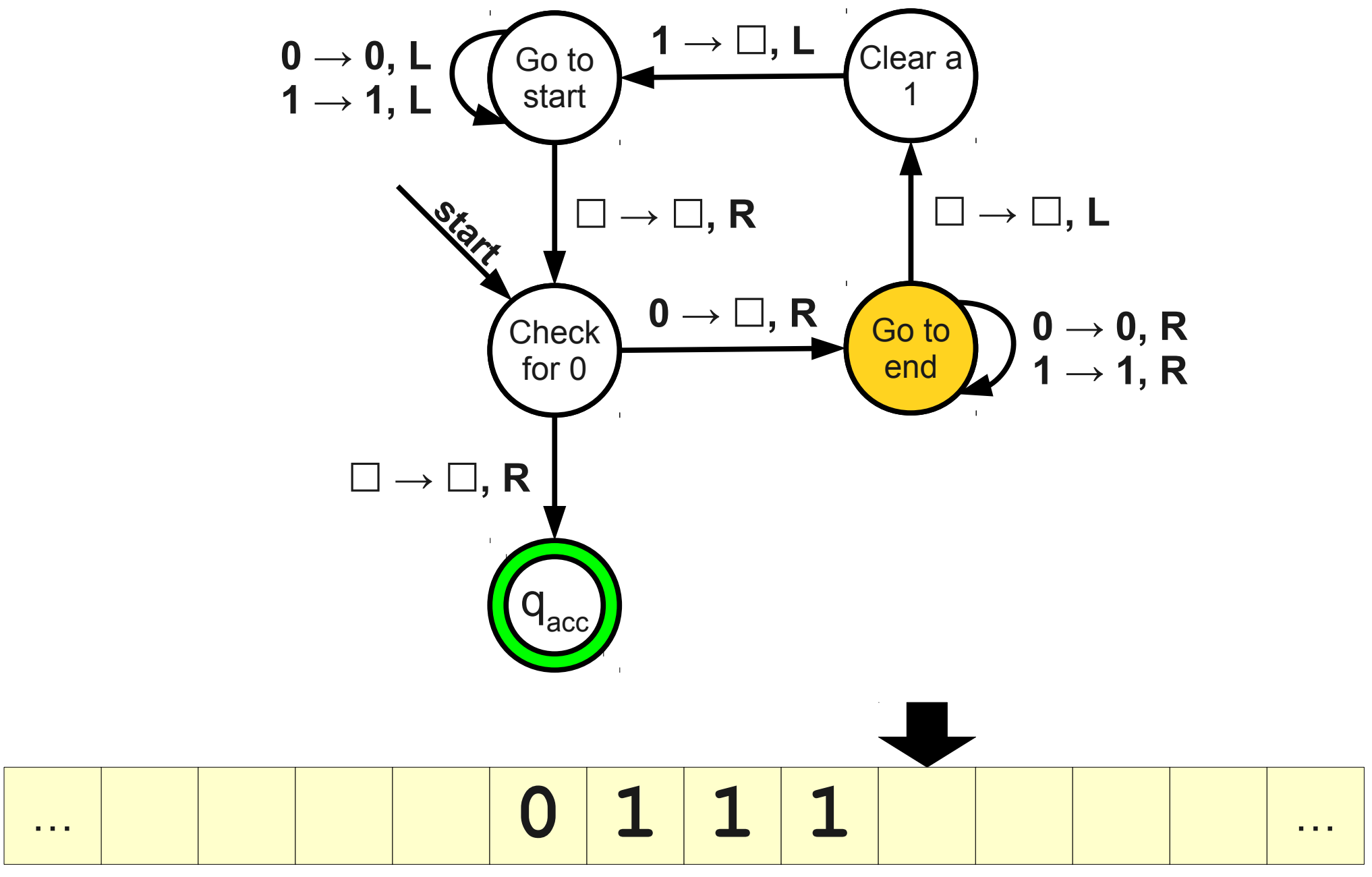

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0 0 1 1 1
...

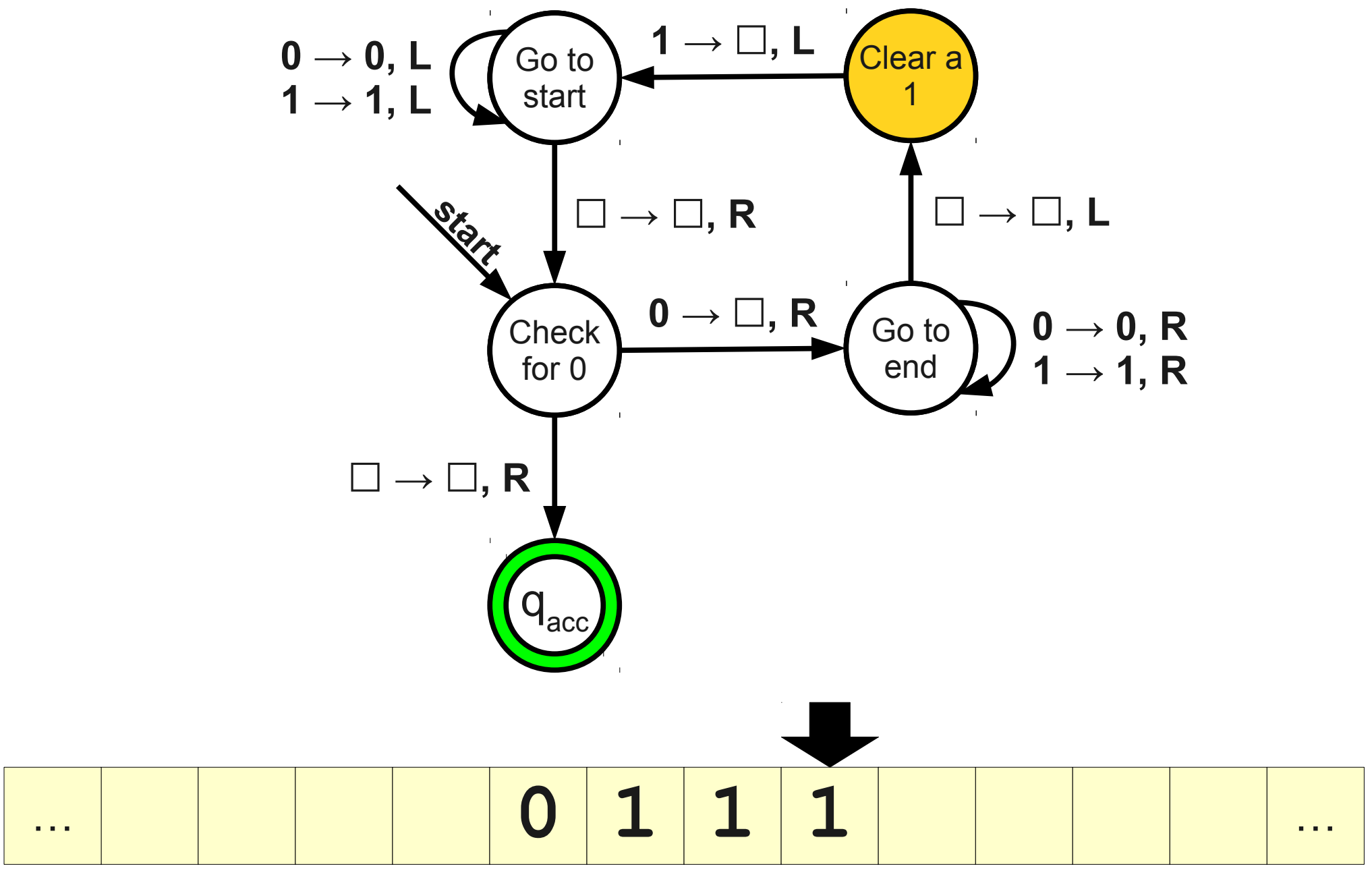

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
1
...

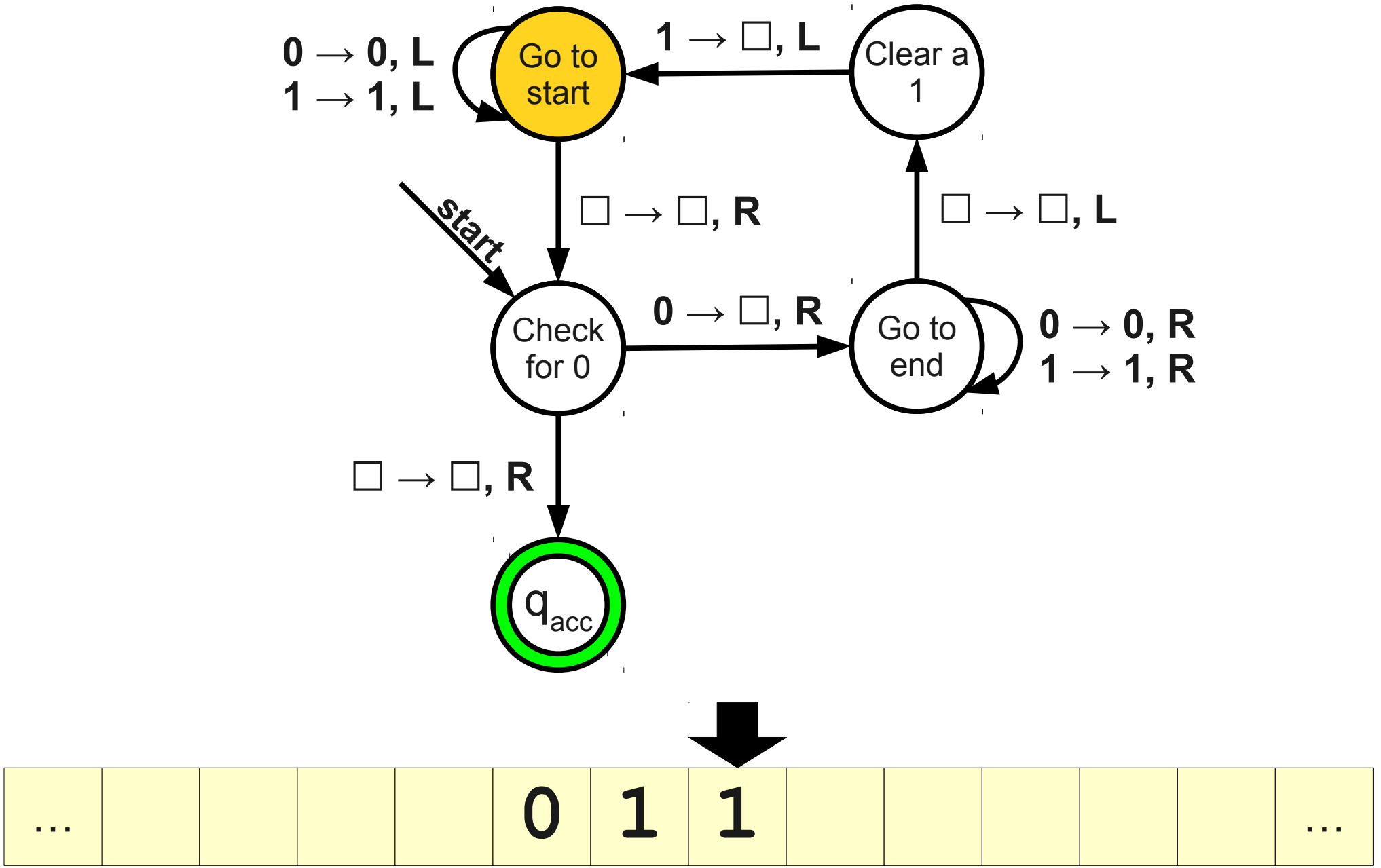

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
1
...

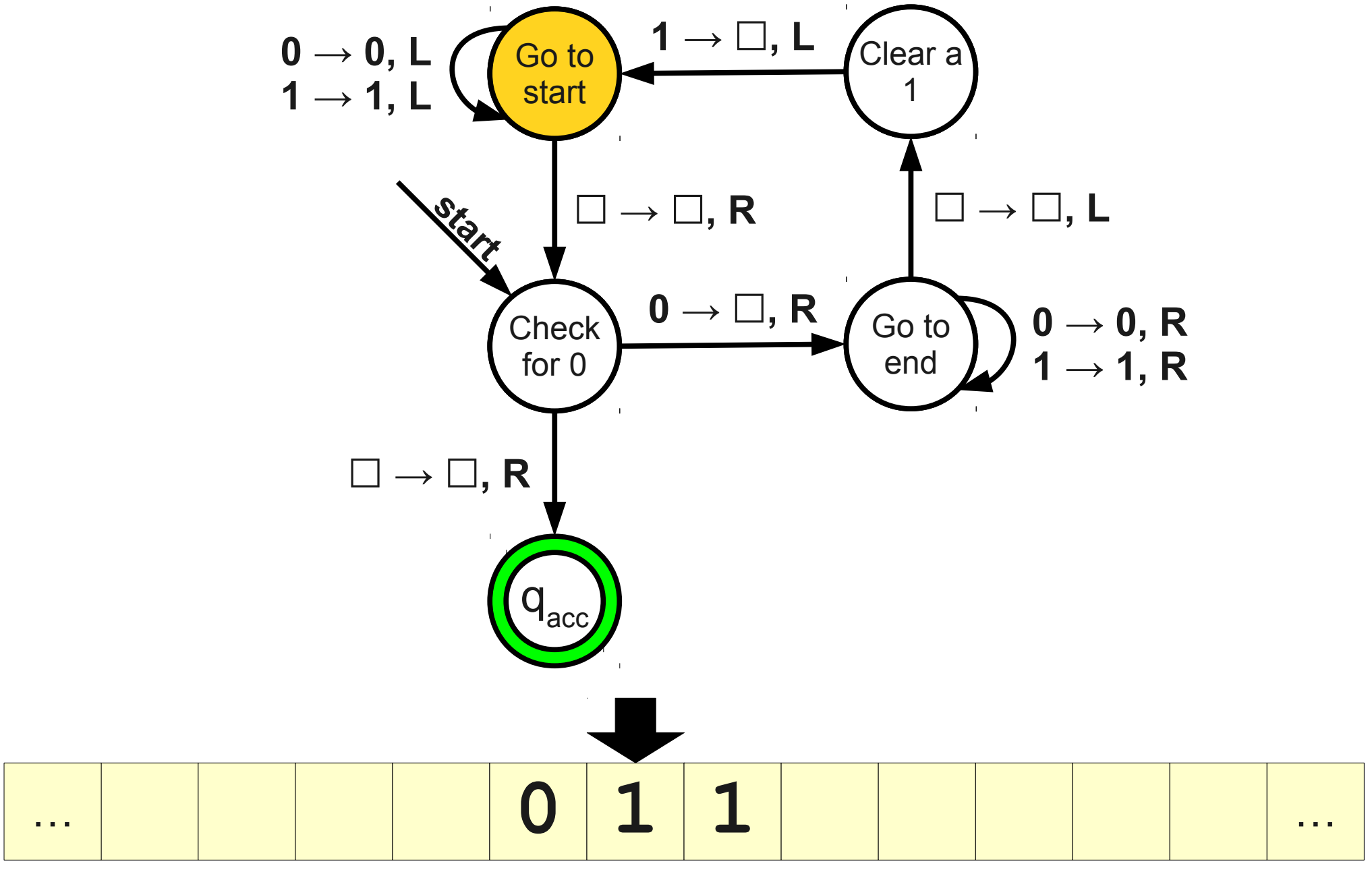

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0 1 1 1
...

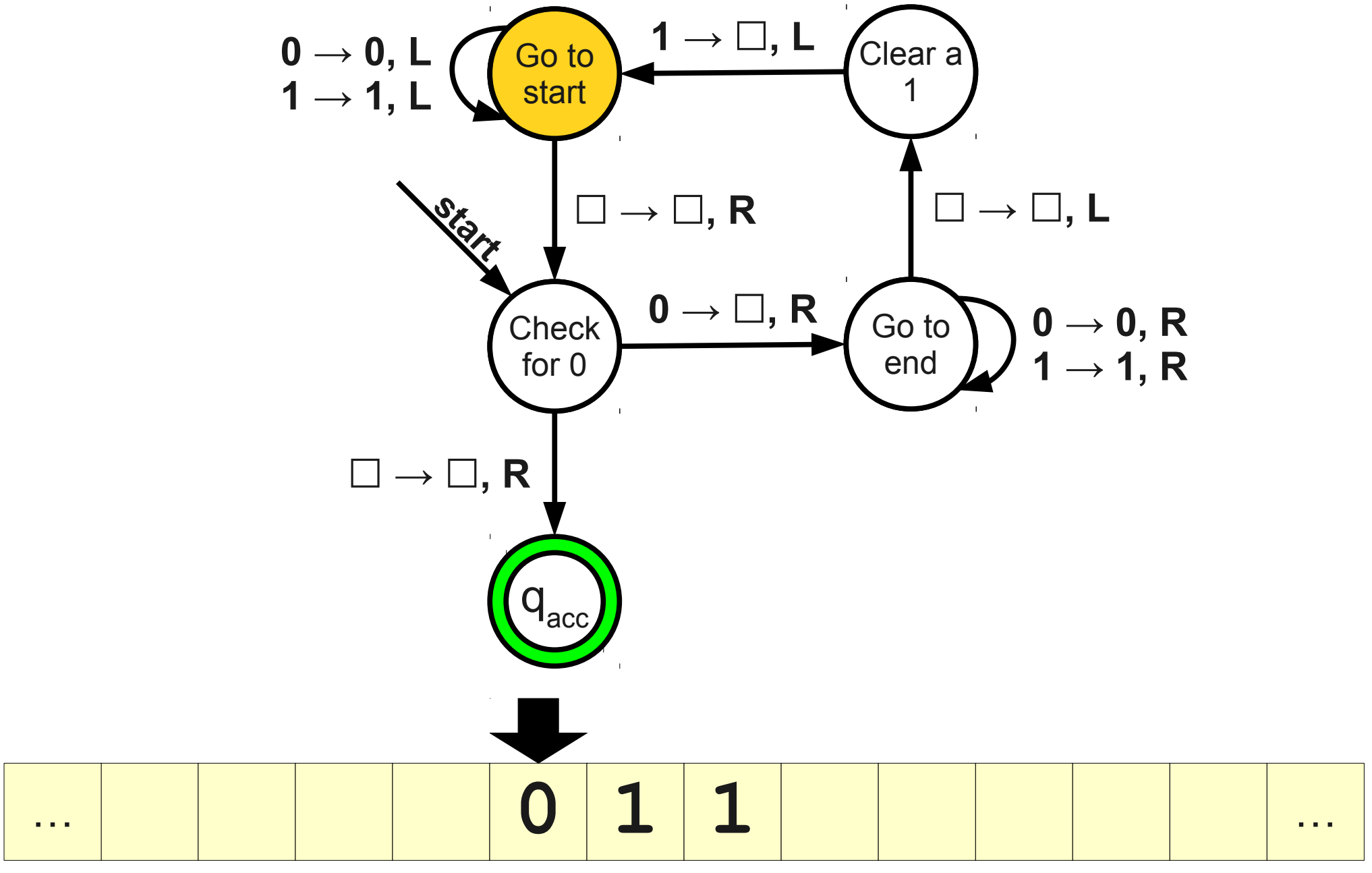

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
1
...

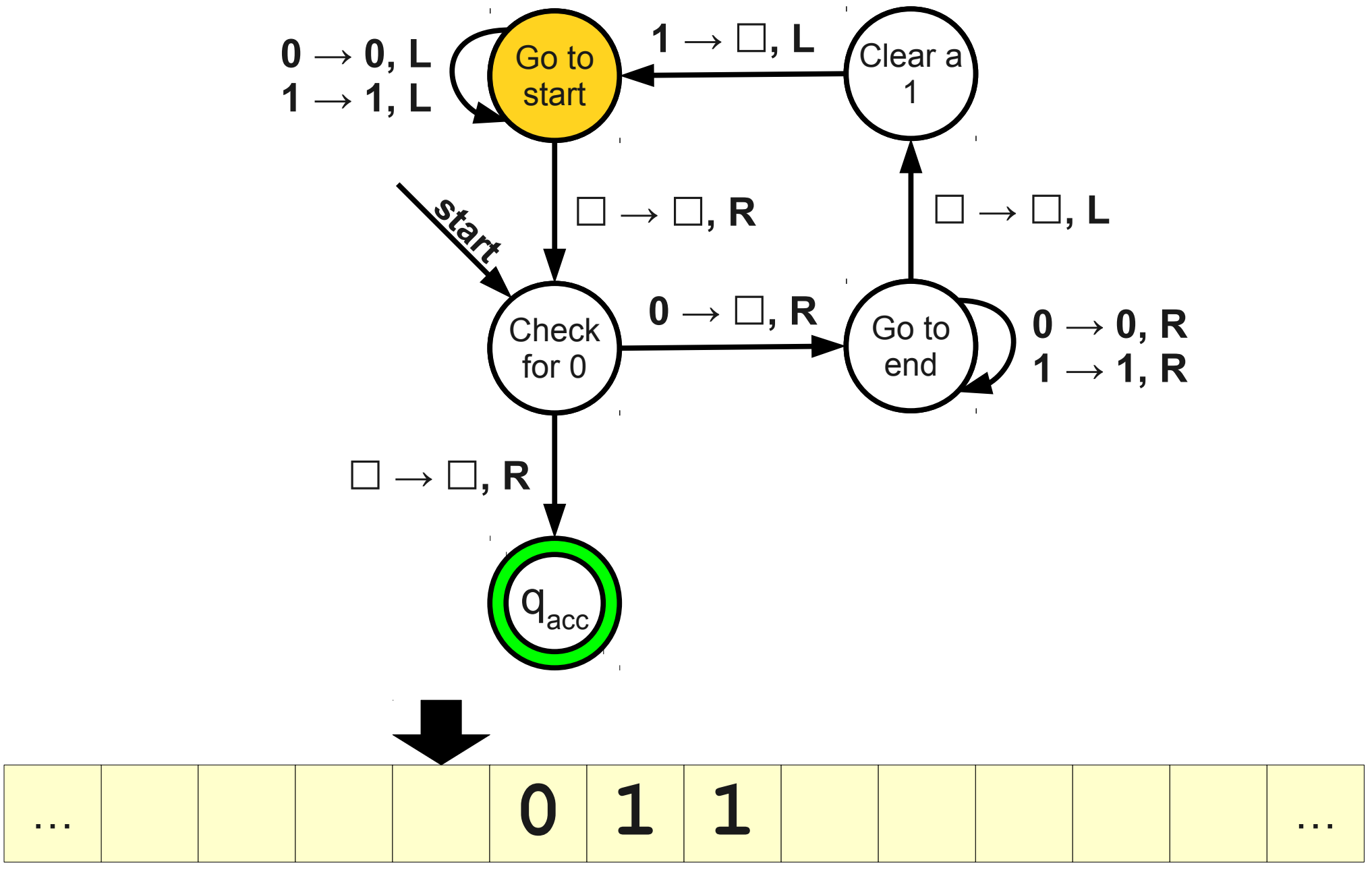

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0 1 1 1
...

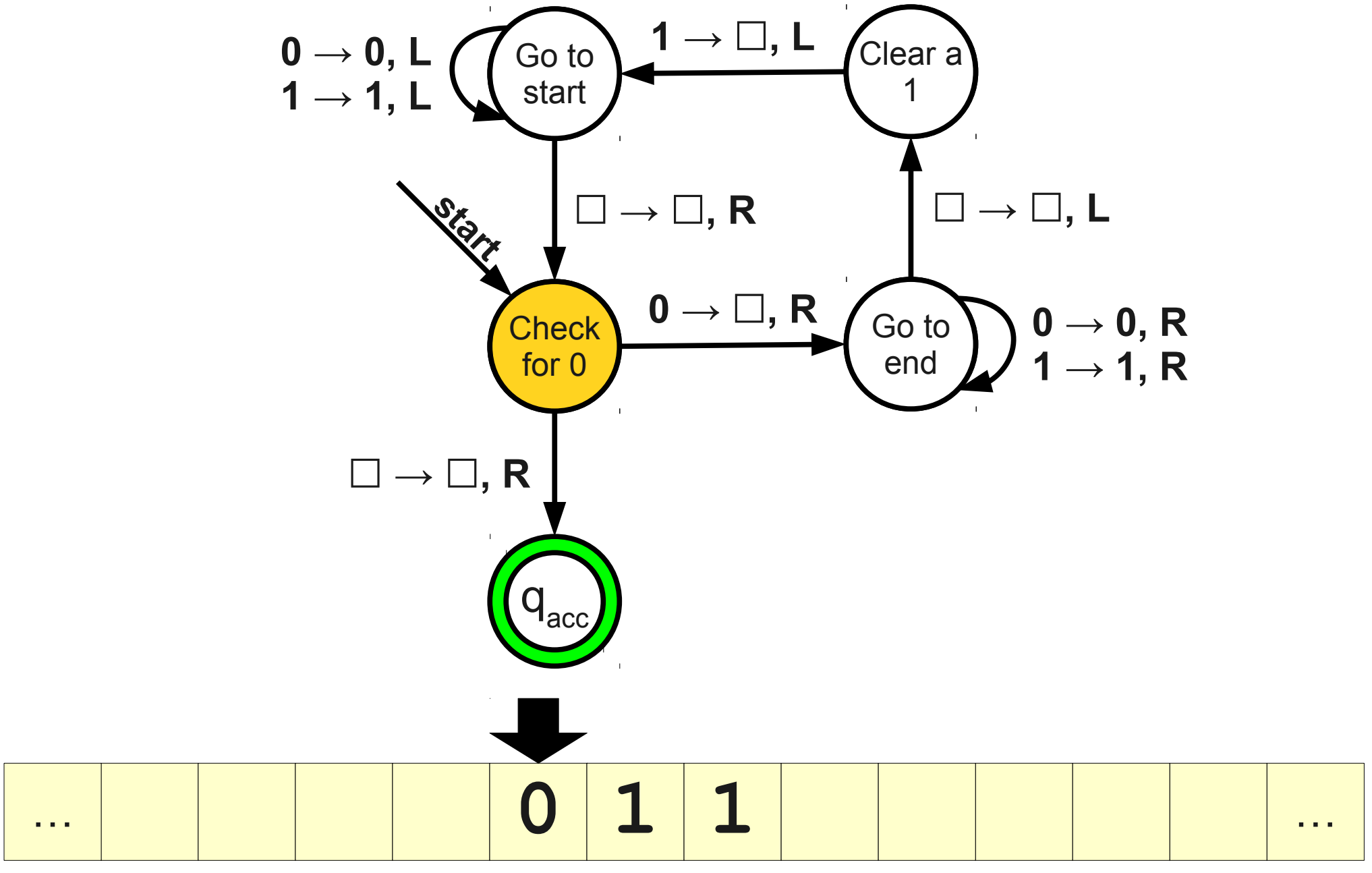

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0 1 1 1
...

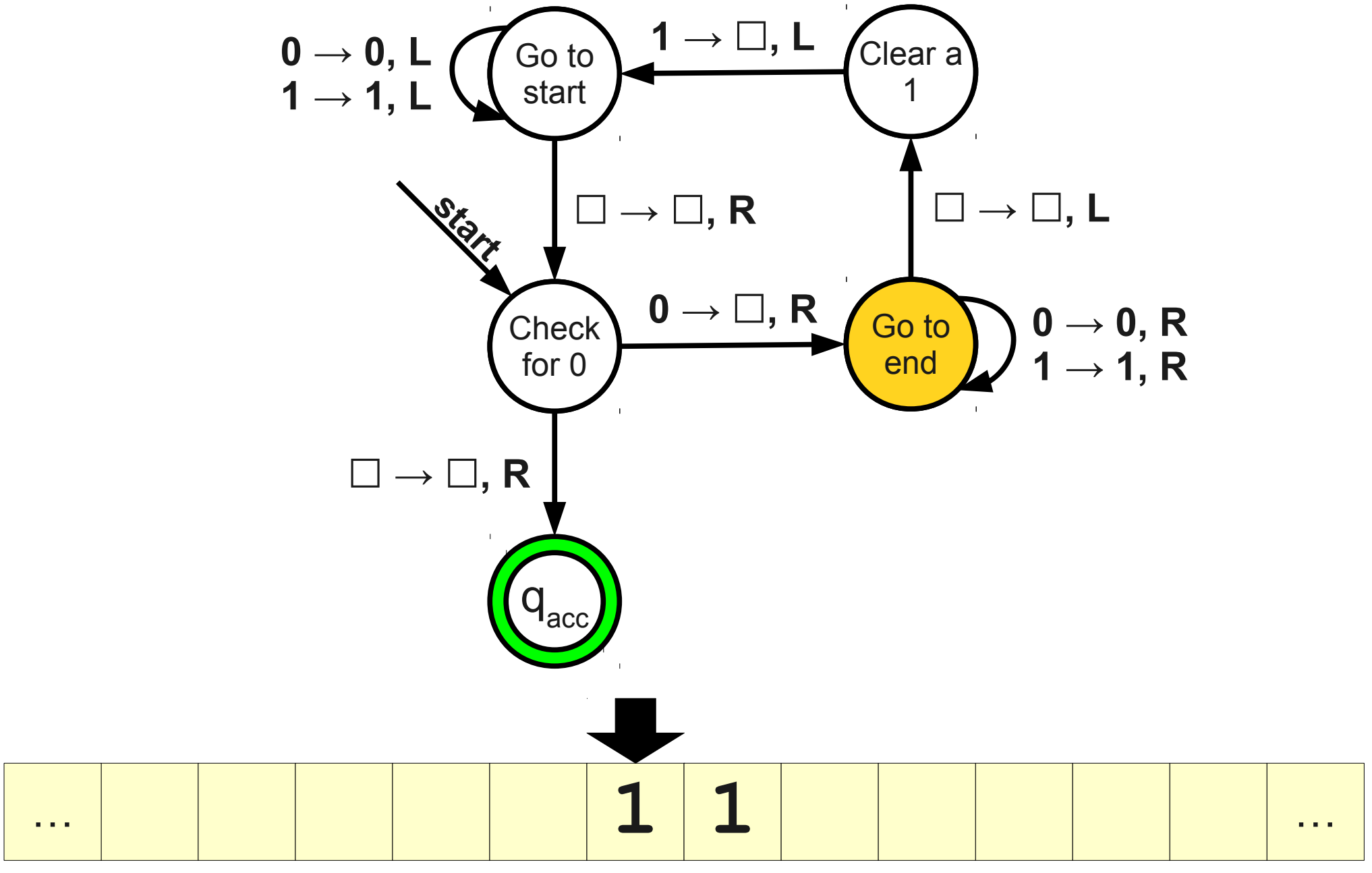

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
...

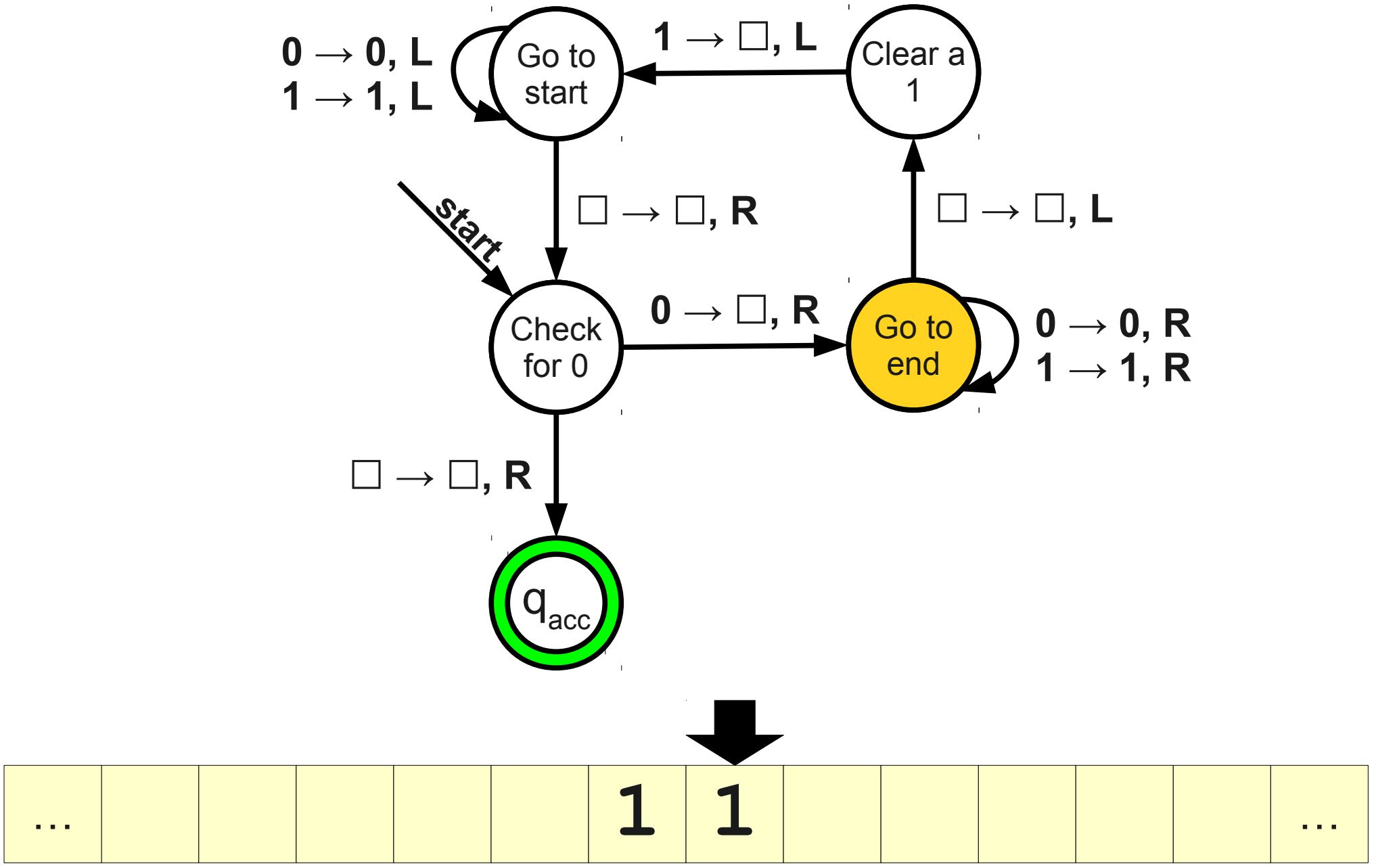

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
...

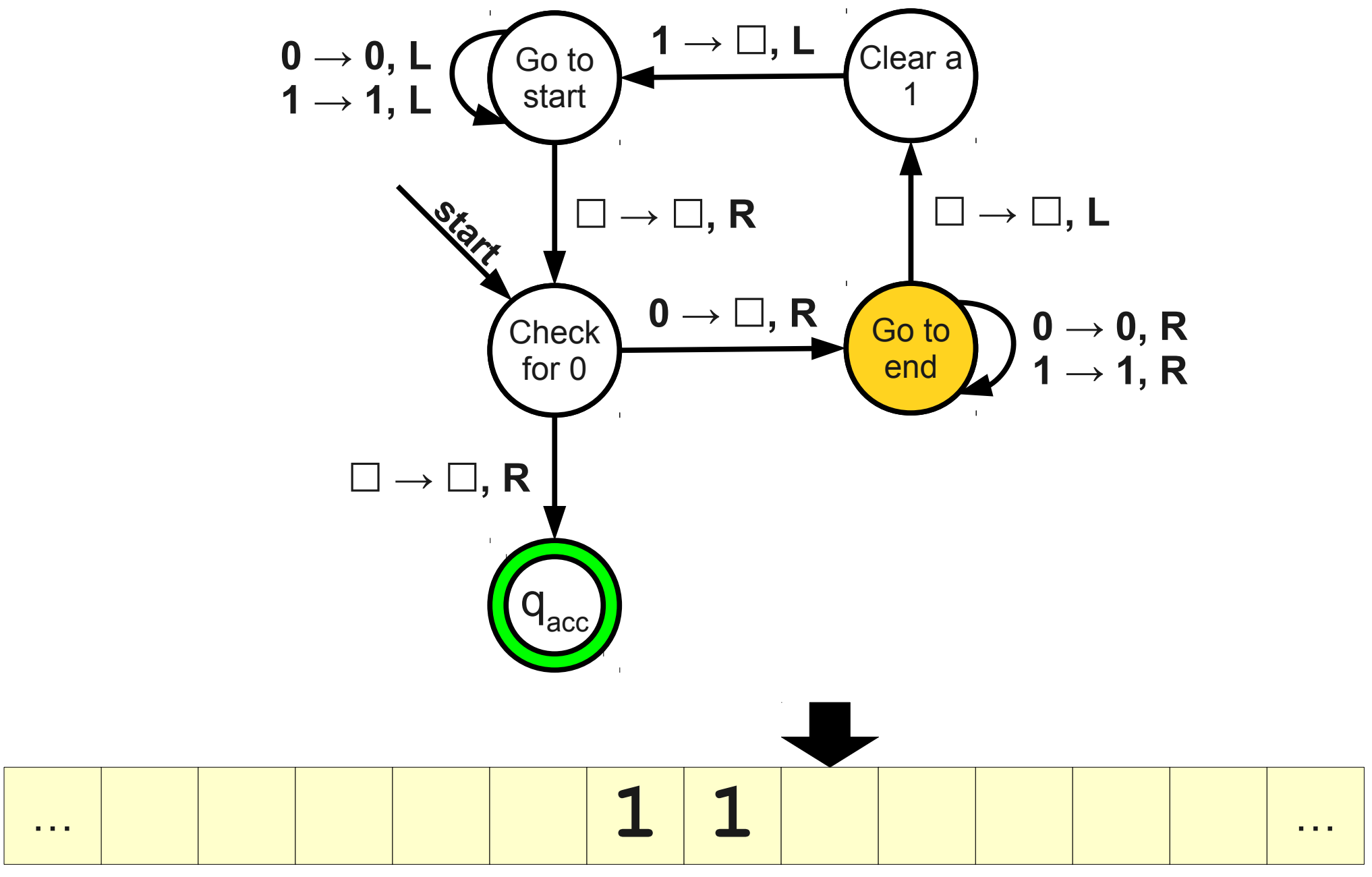

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
...

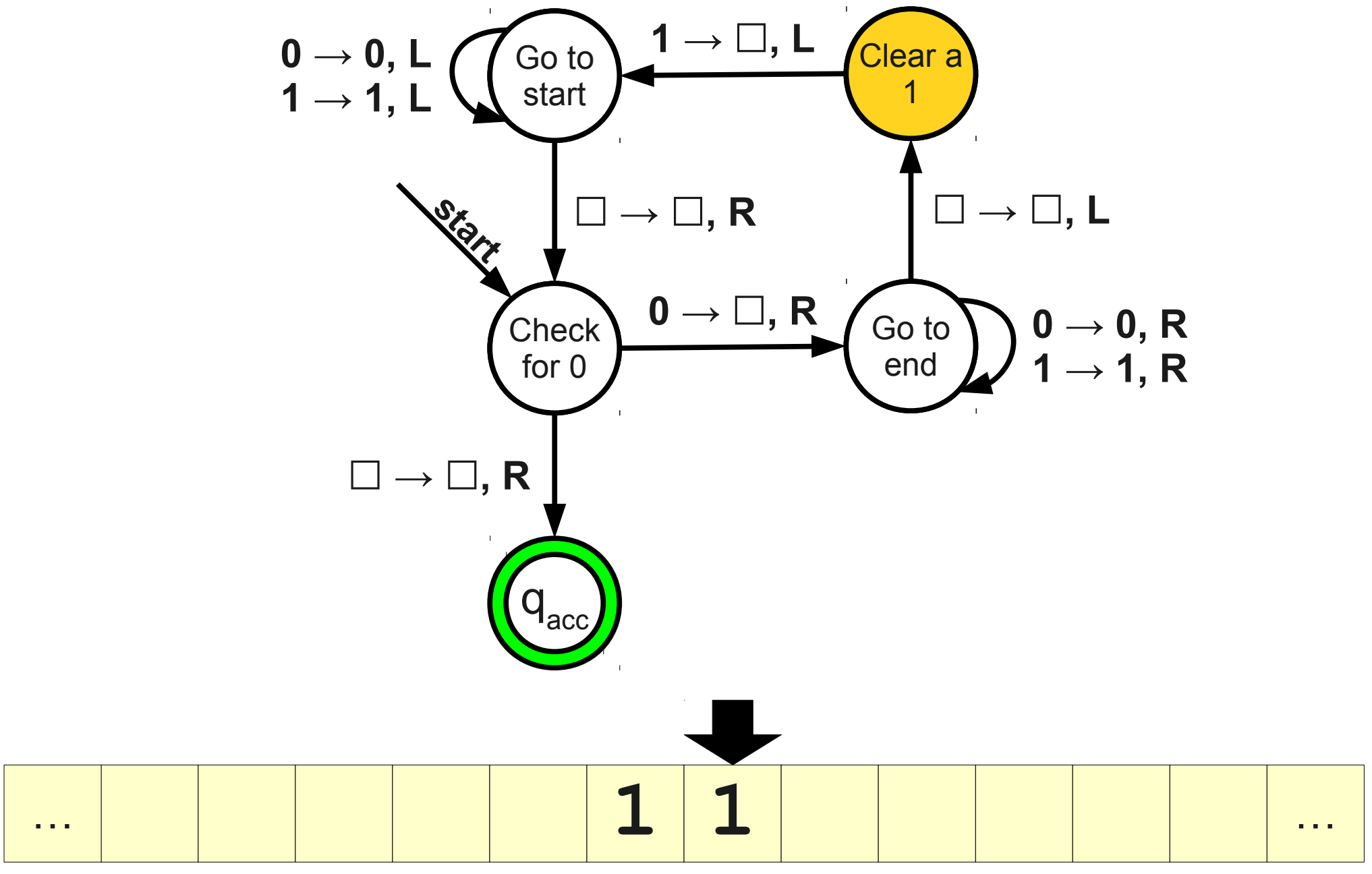

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
...

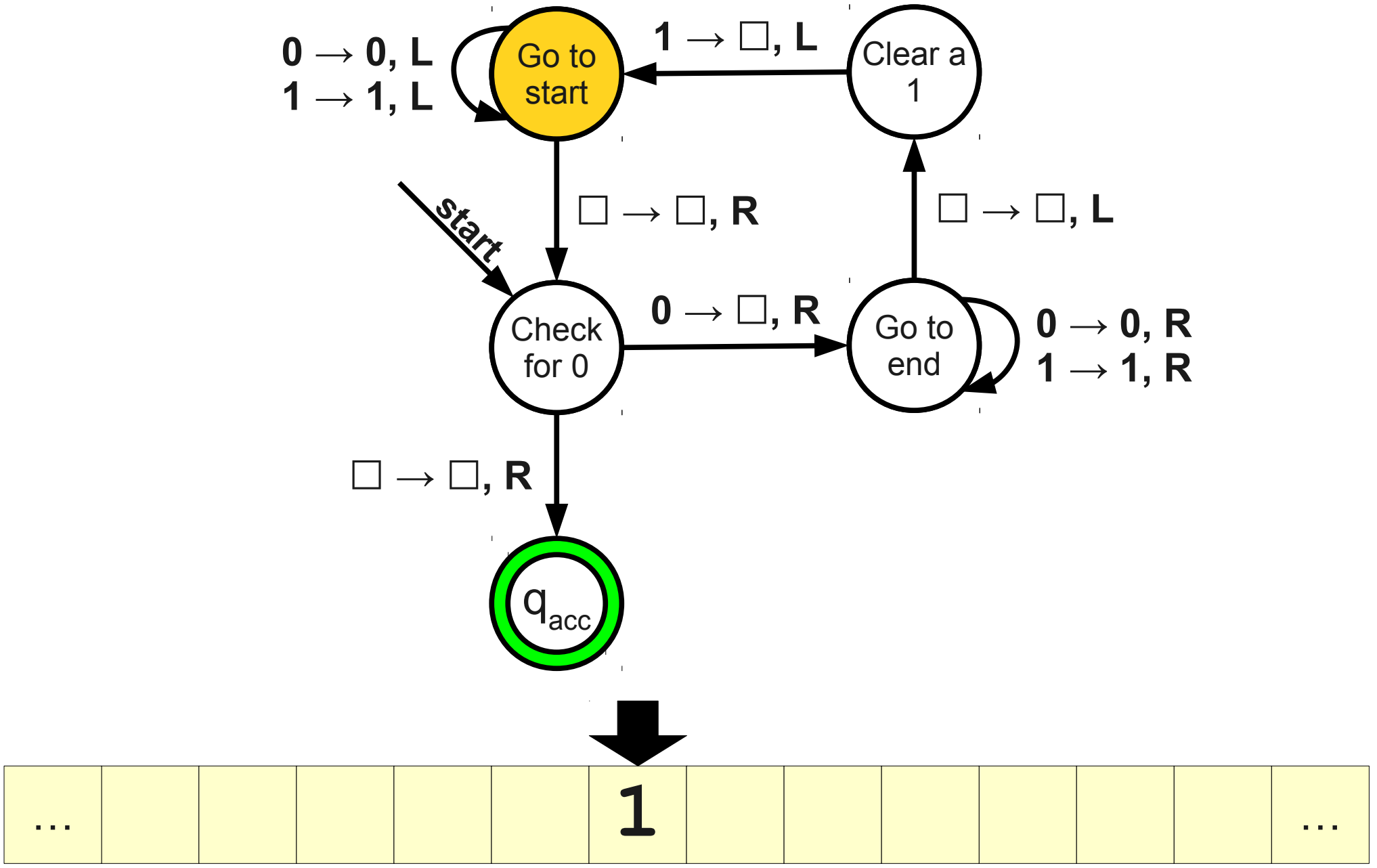

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
0
1
1
...

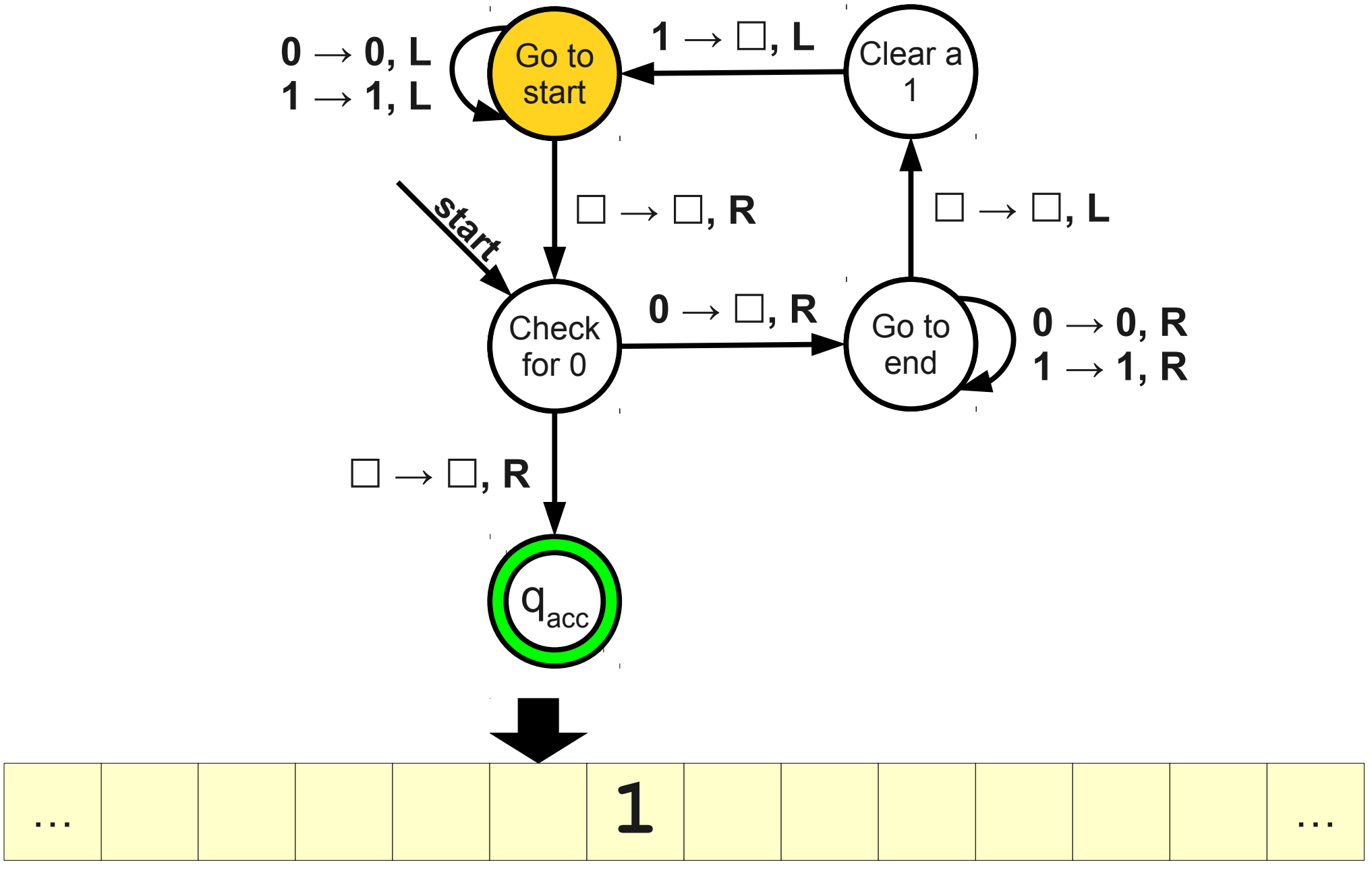

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
1
1
...

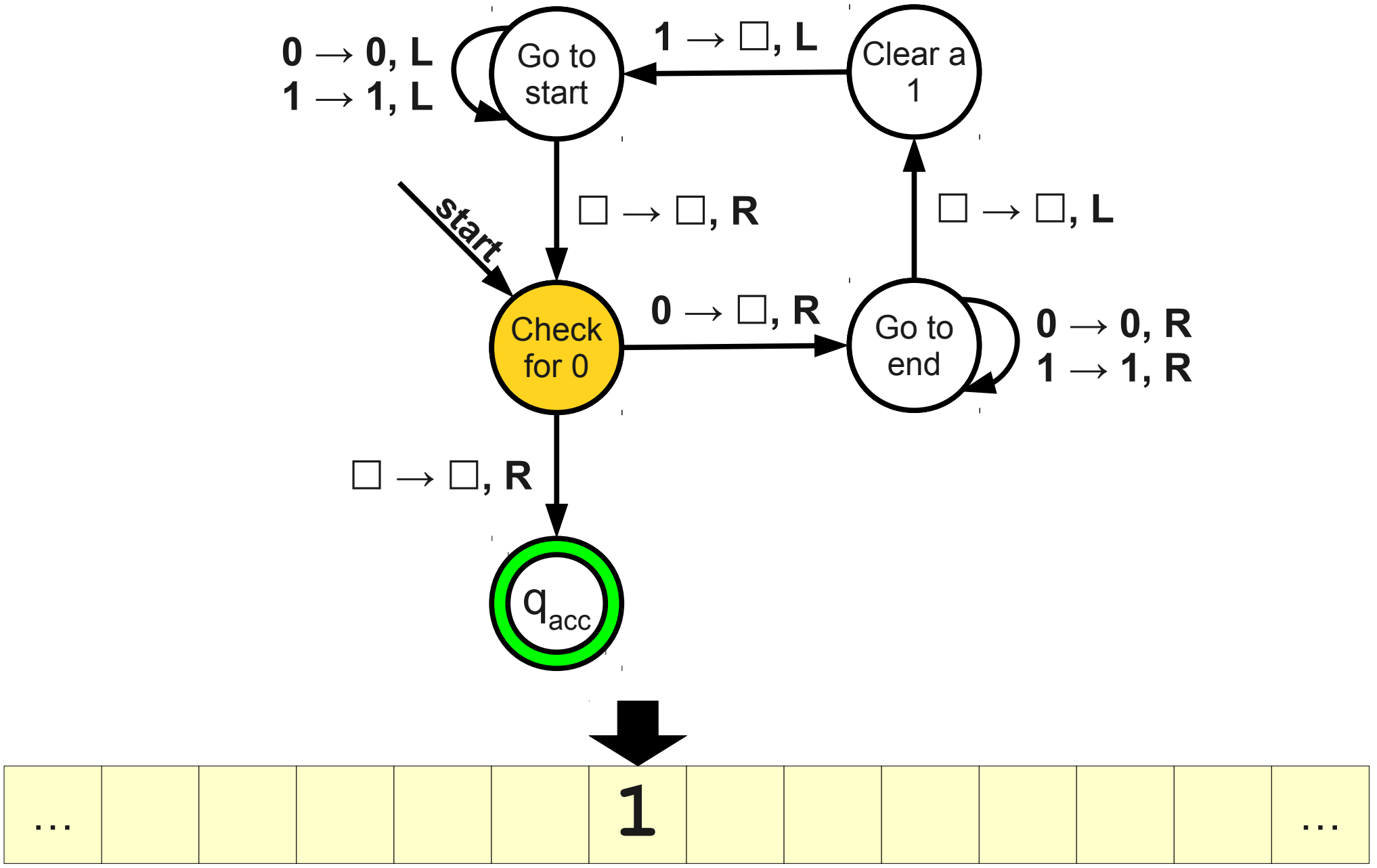

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
1
1
...

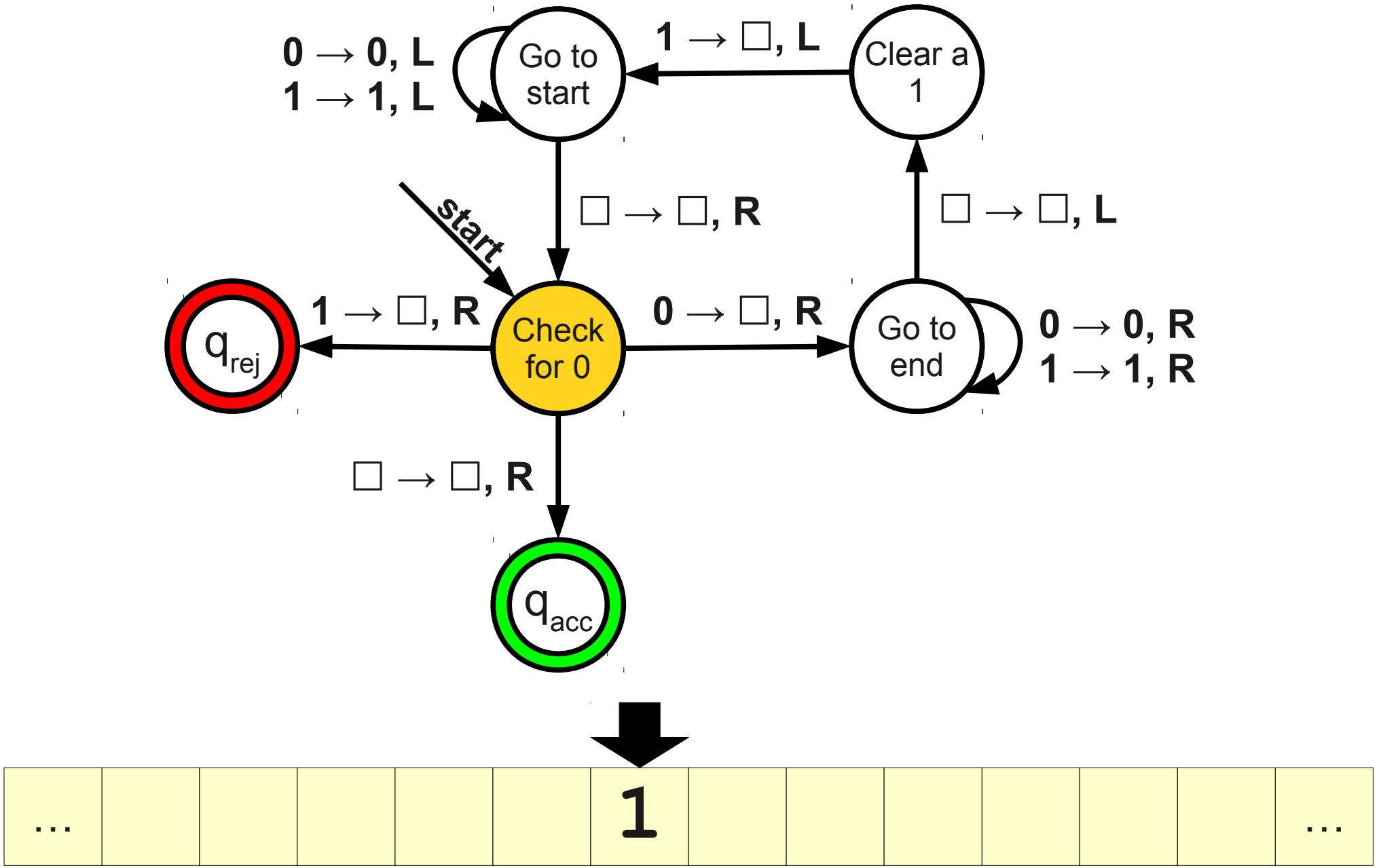

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
1
1
...

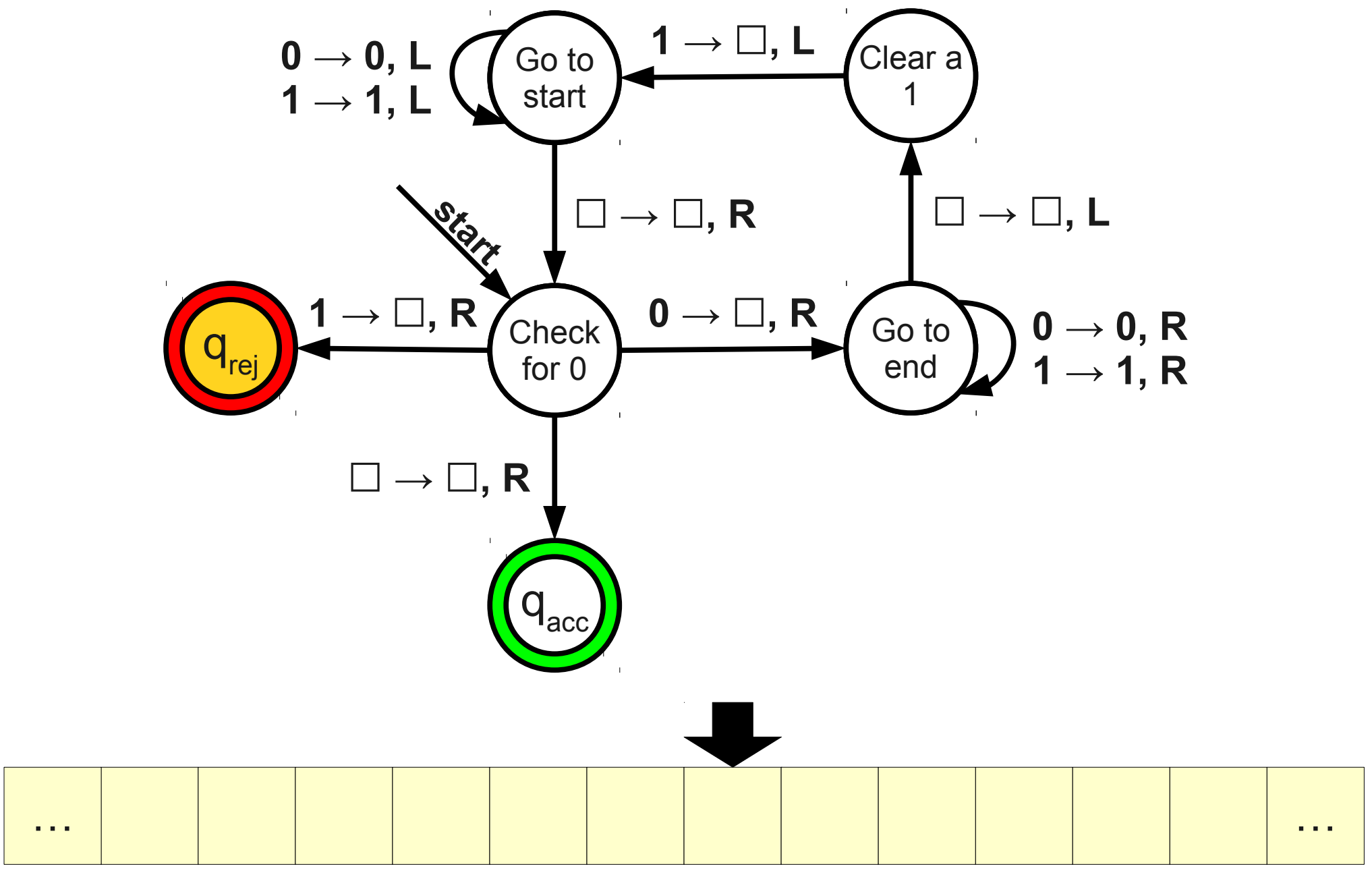

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
1
1
...

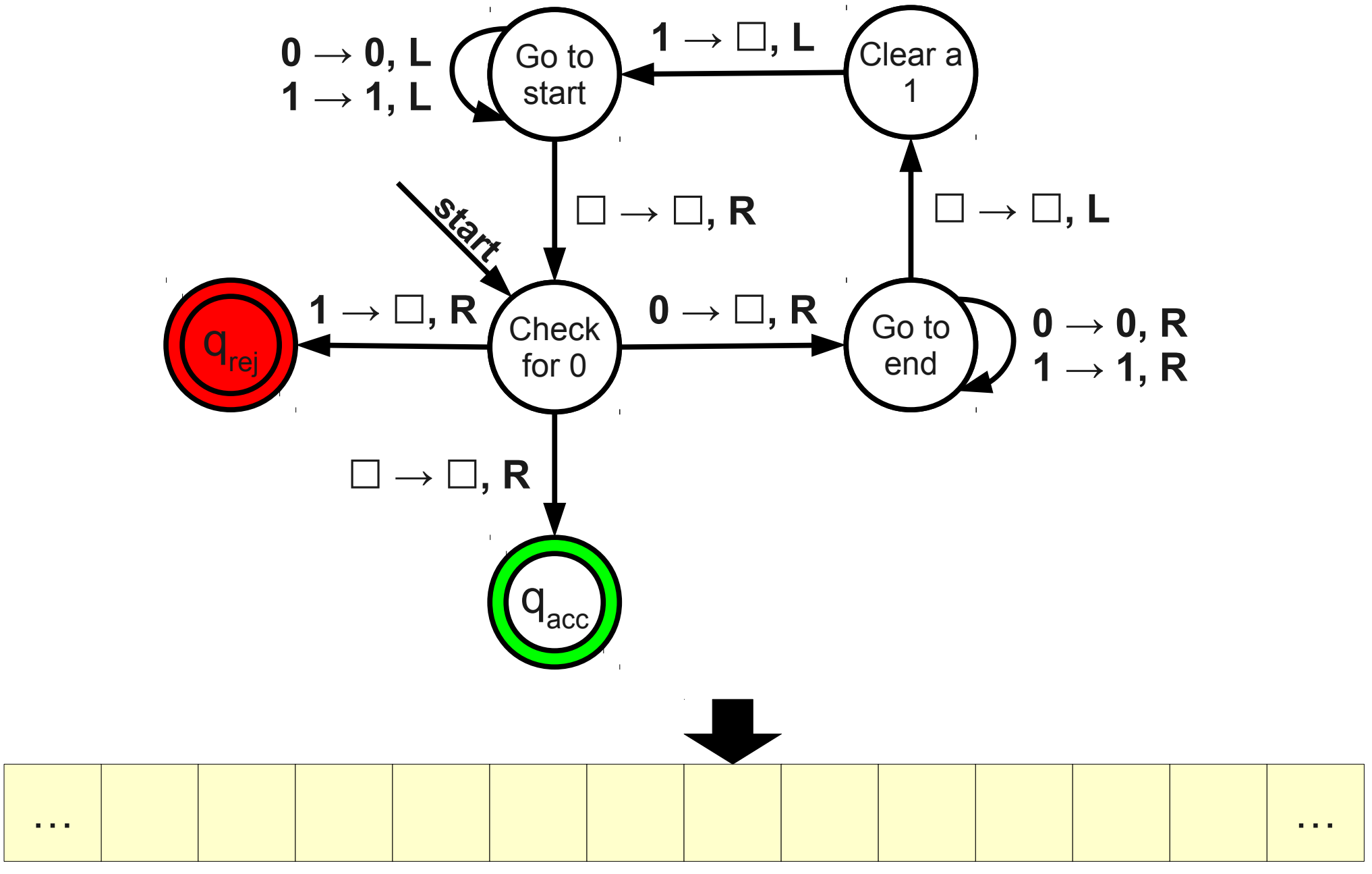

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_acc
...
1
...

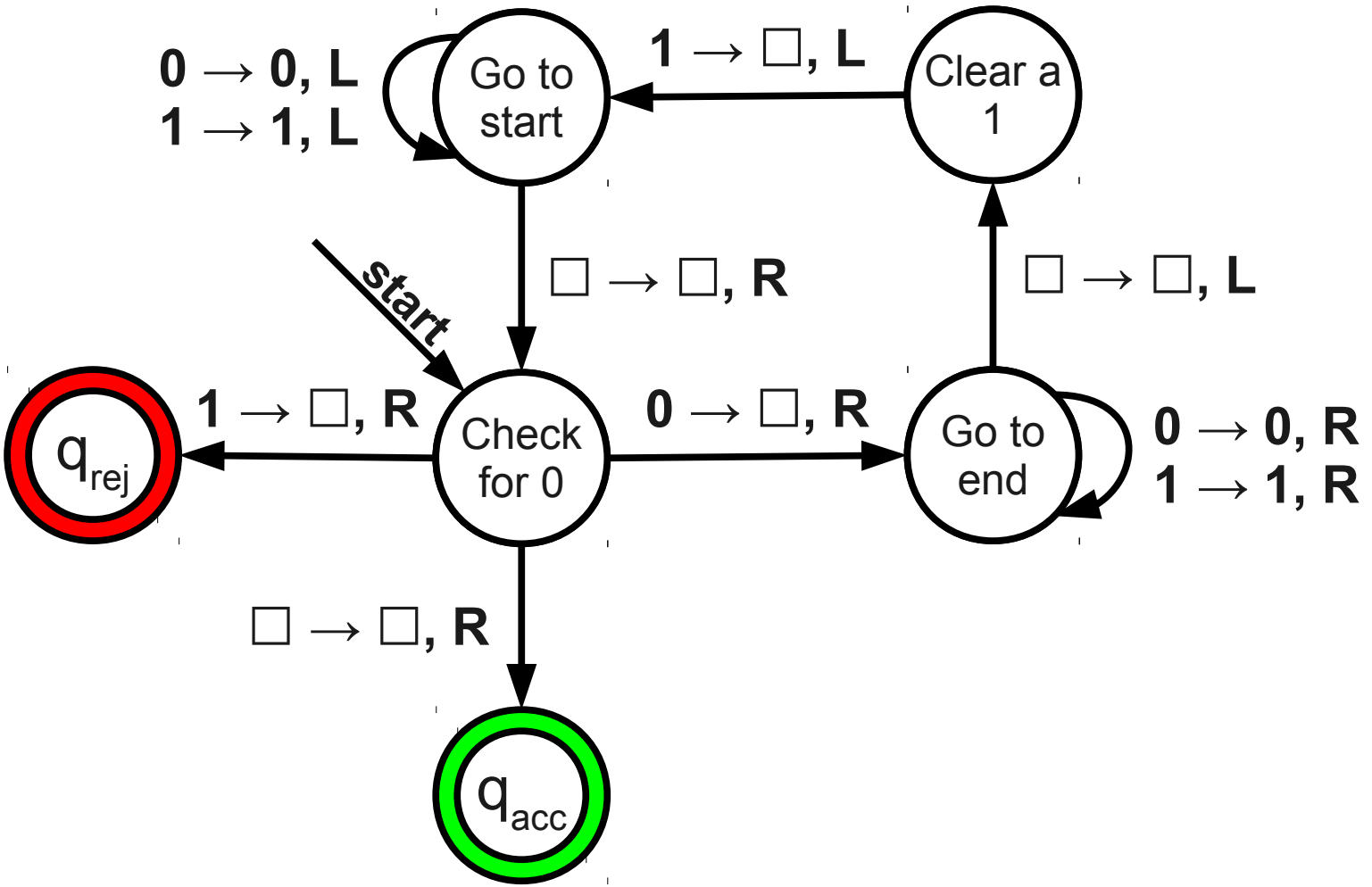

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_{acc}
...
1
...

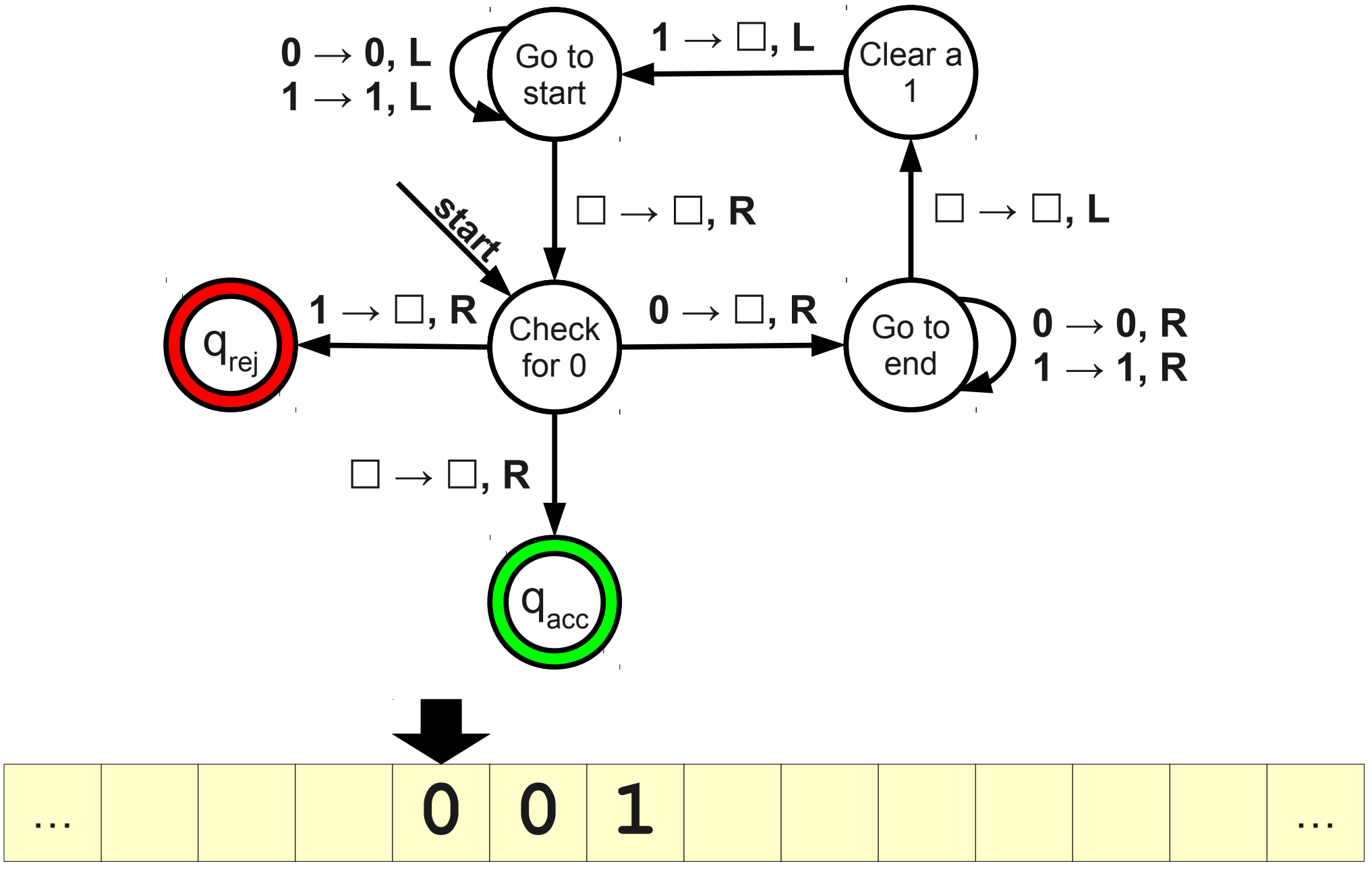

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_acc
...
1
...

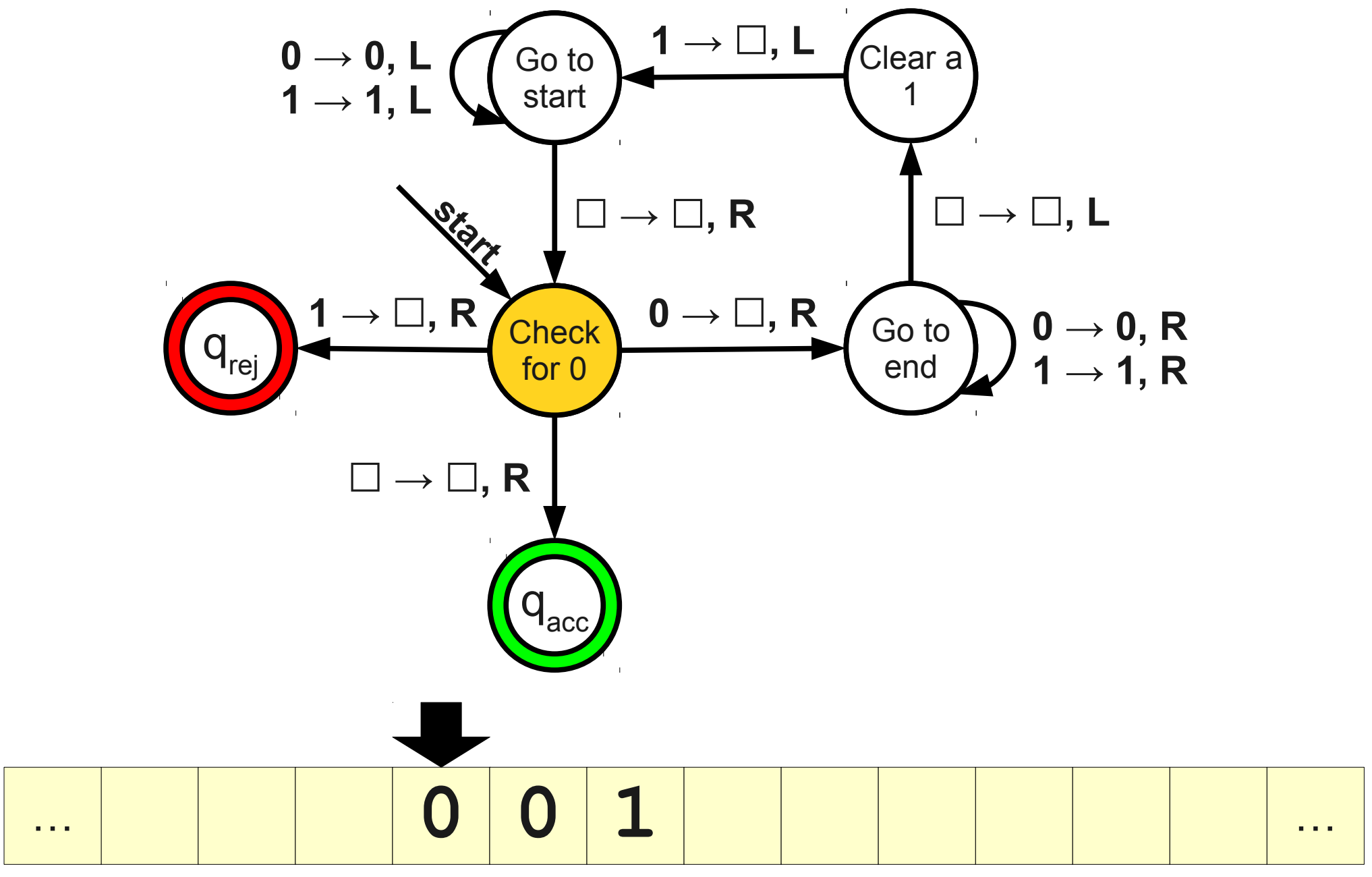

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
...

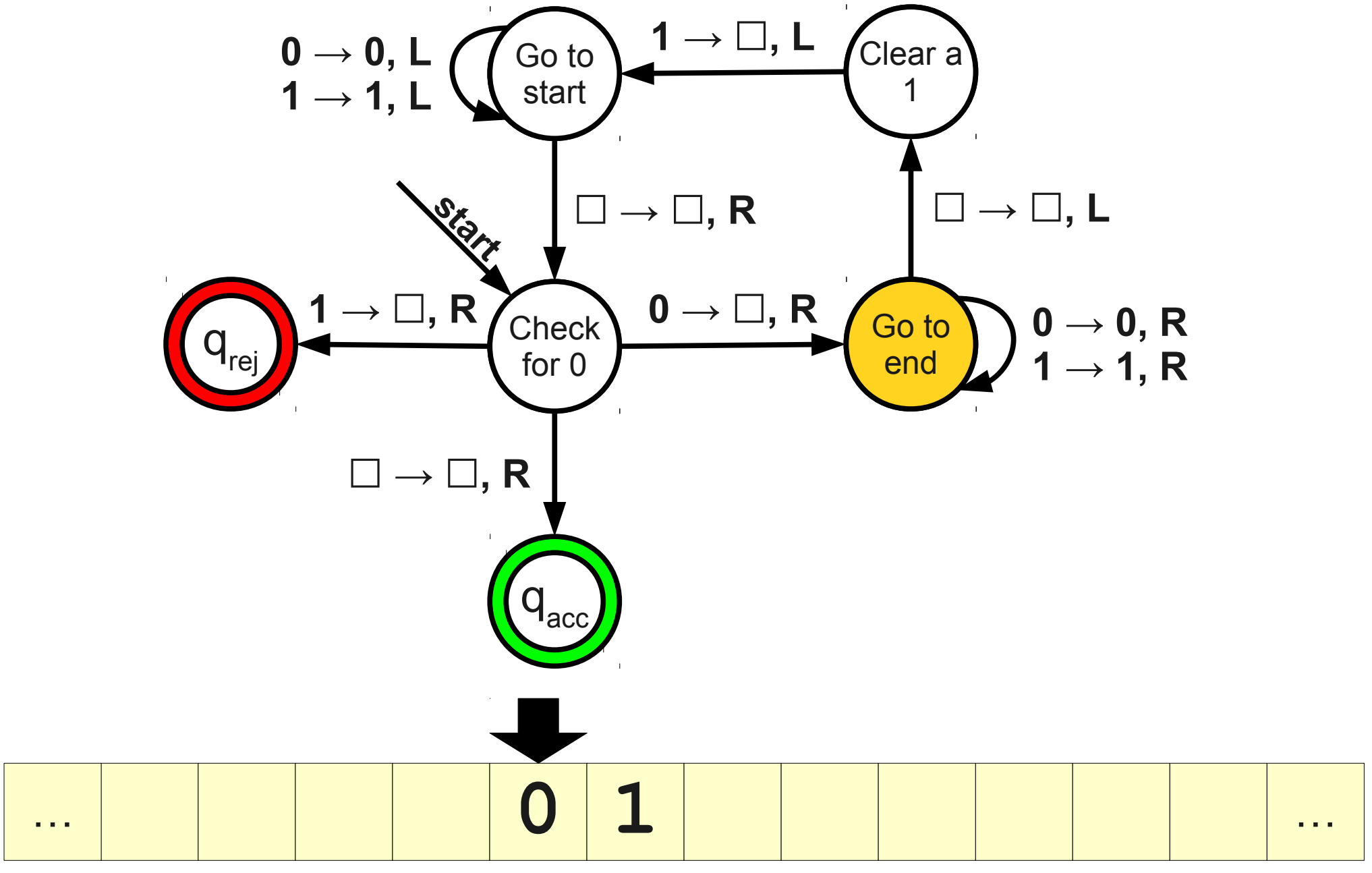

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

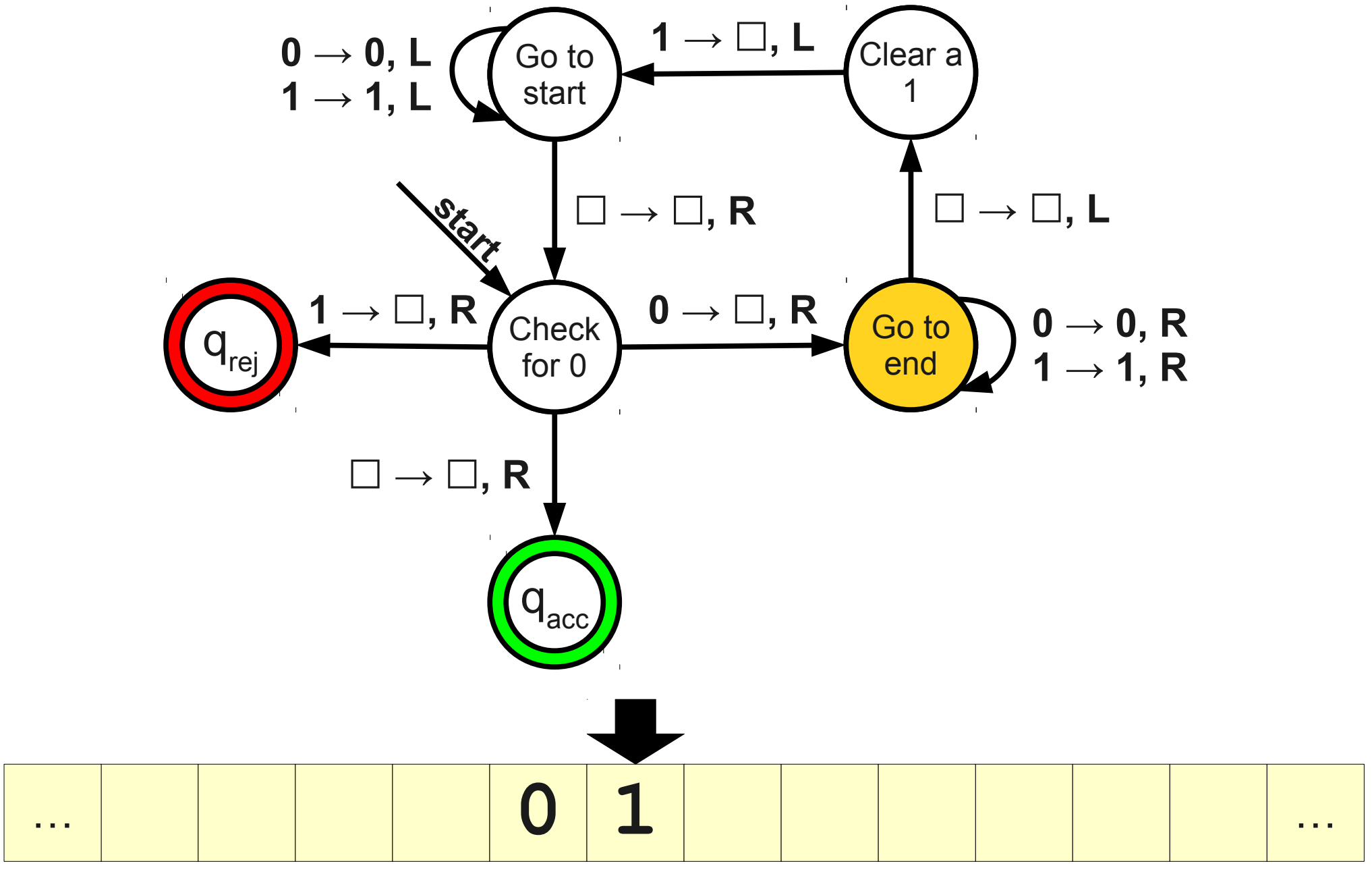

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

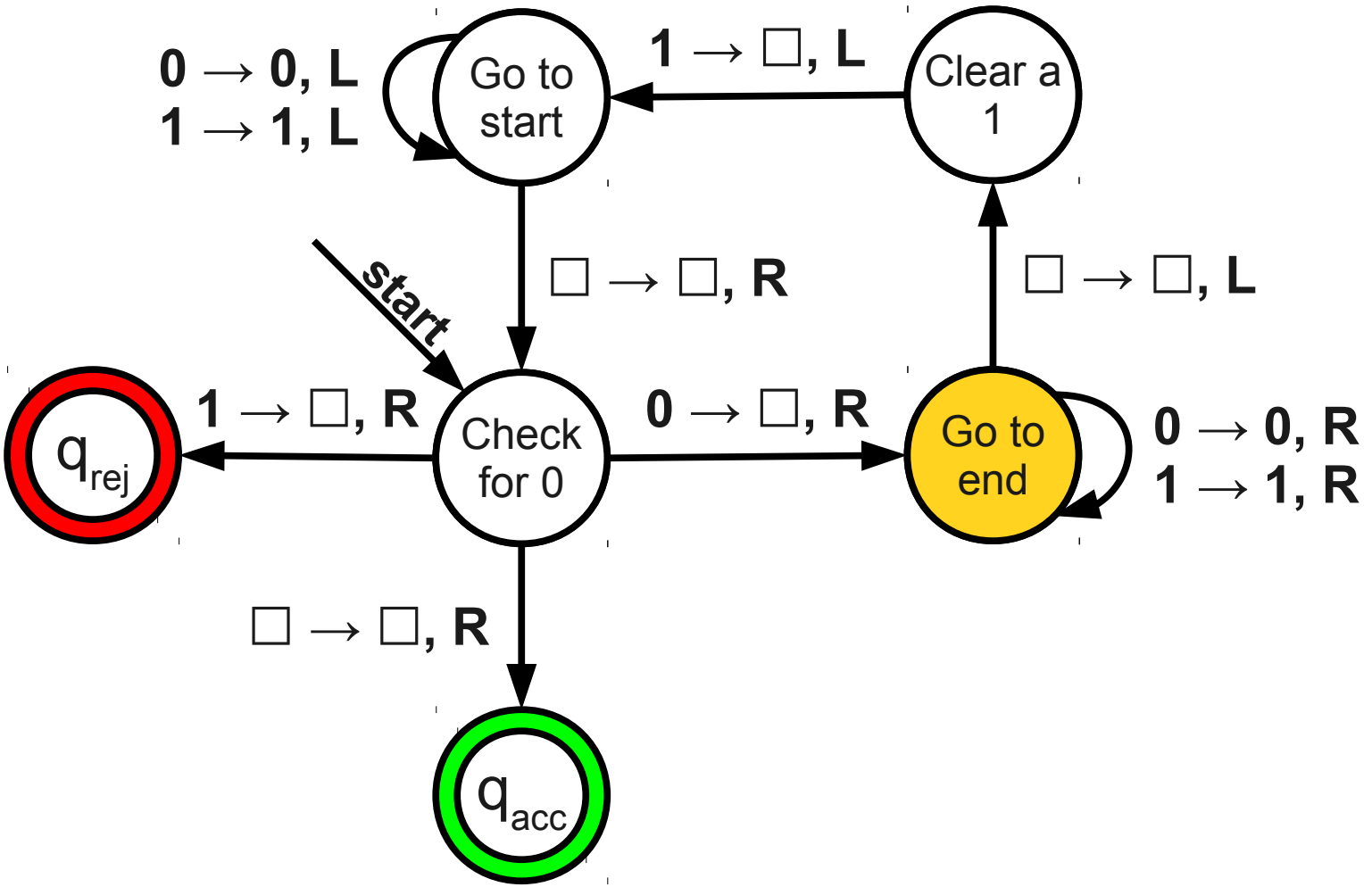

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc

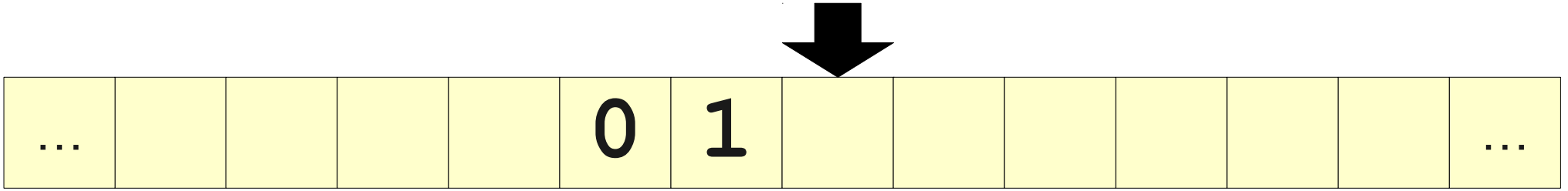

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
0
1
...

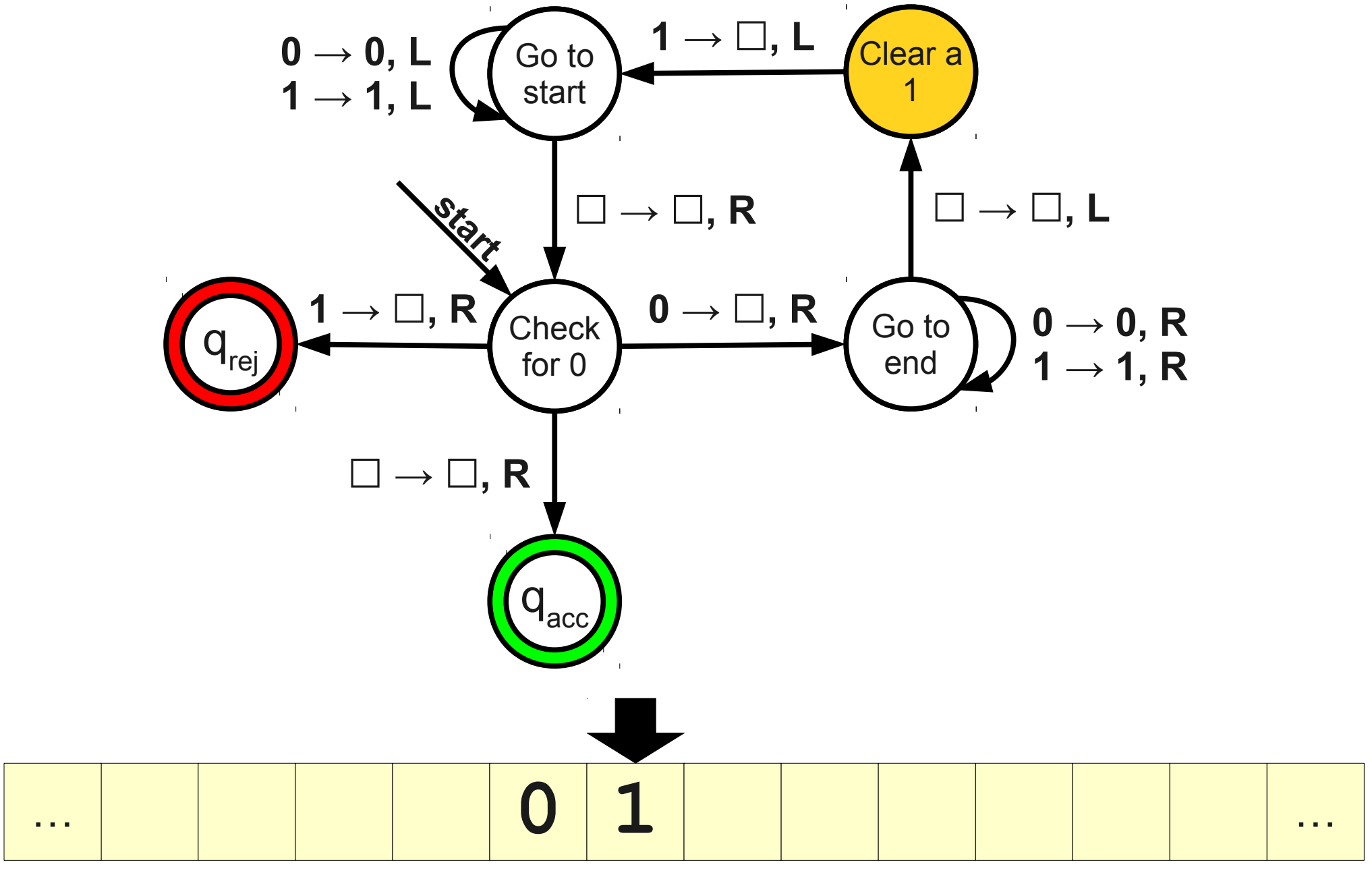

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
0
1
...

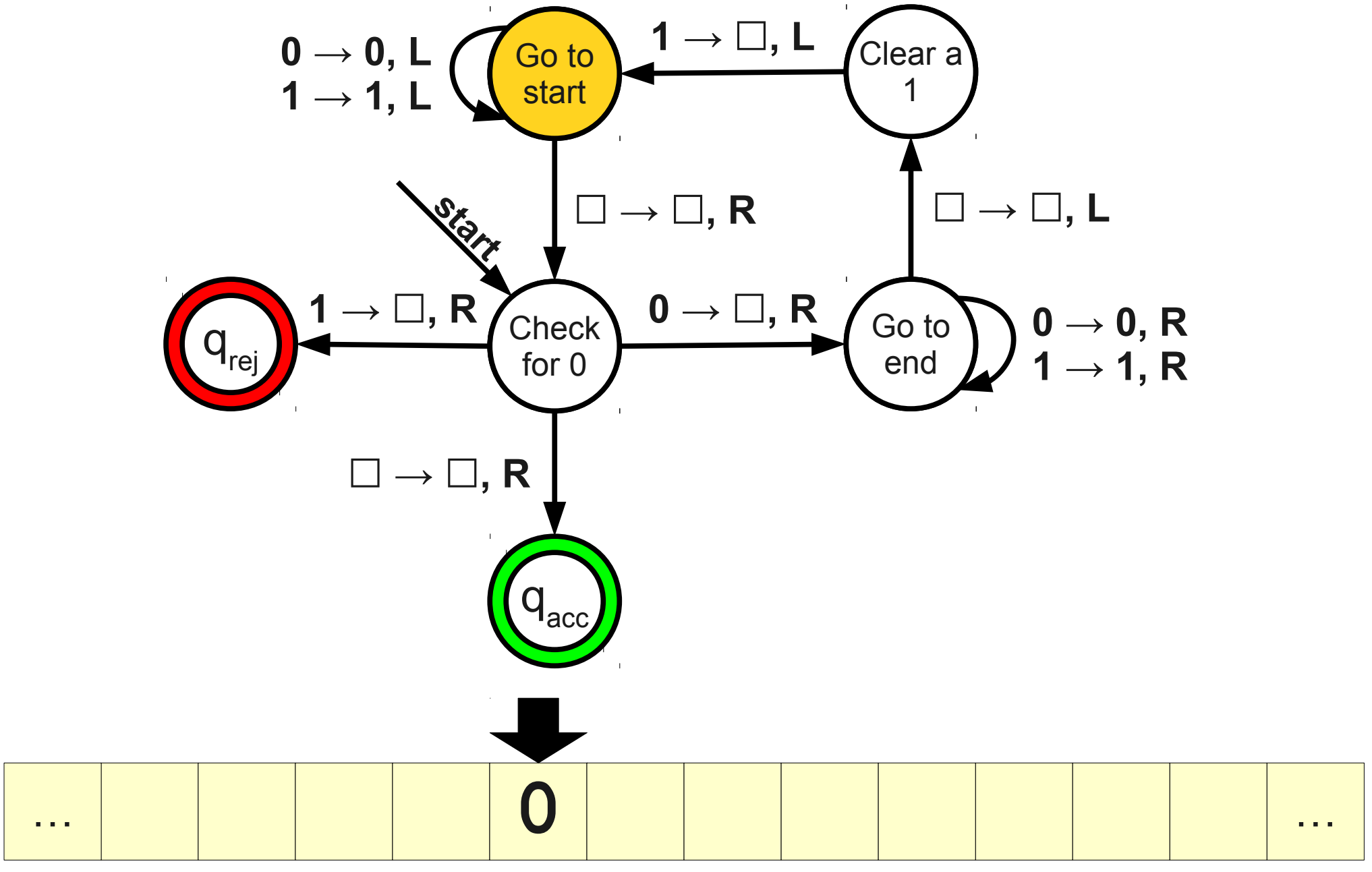

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
1
...

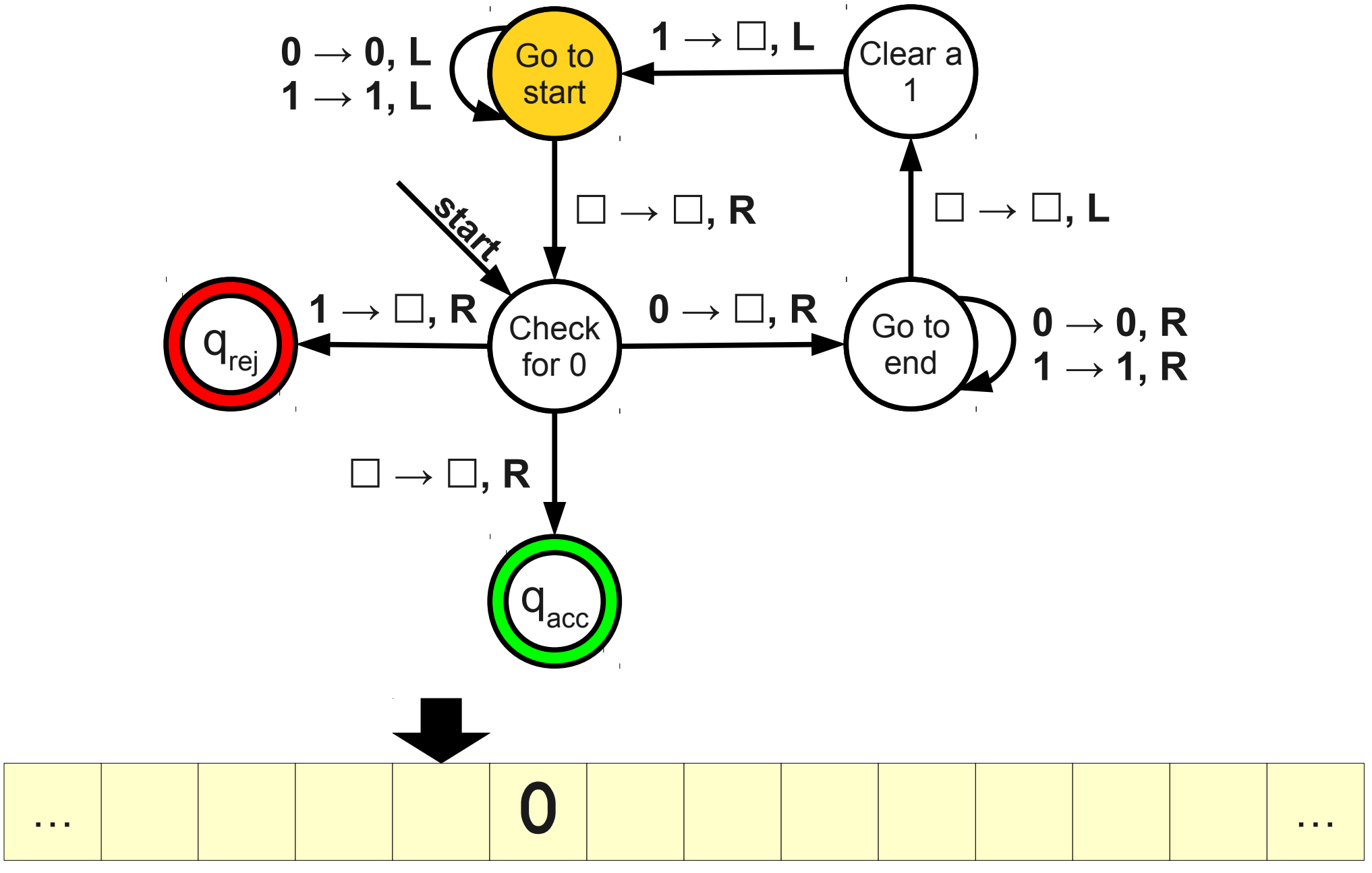

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
1
...

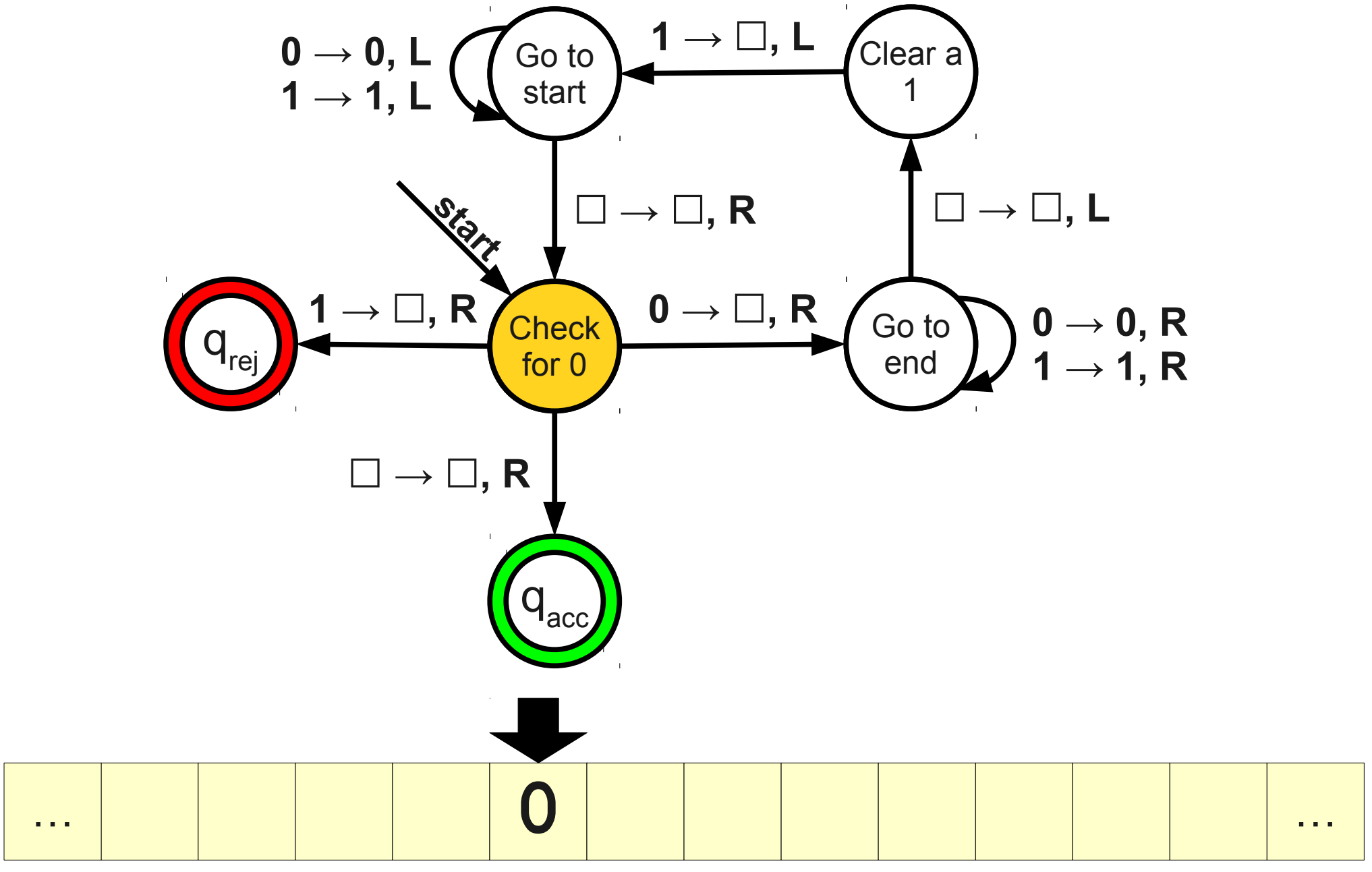

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc

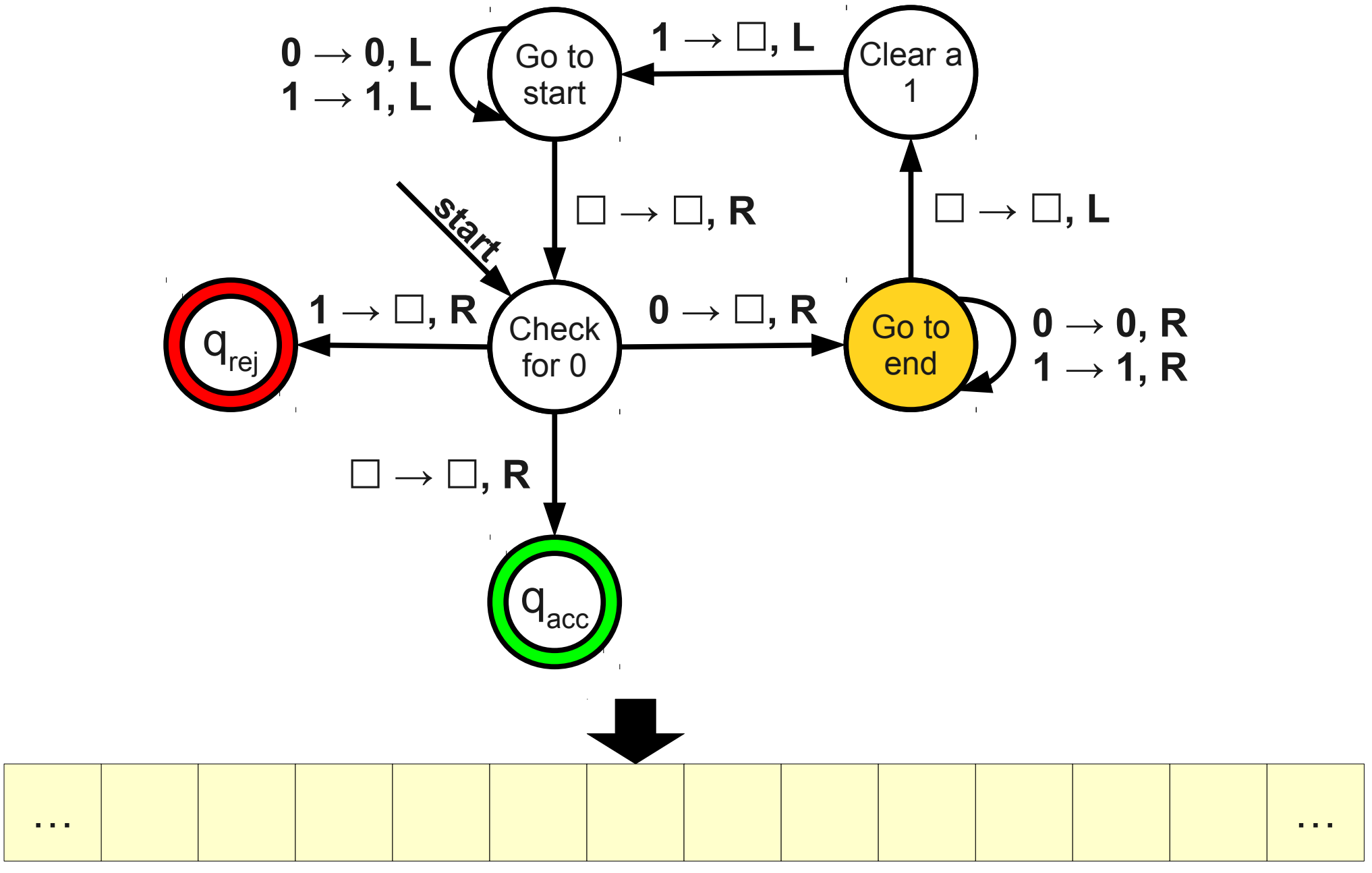

...
0
1
...

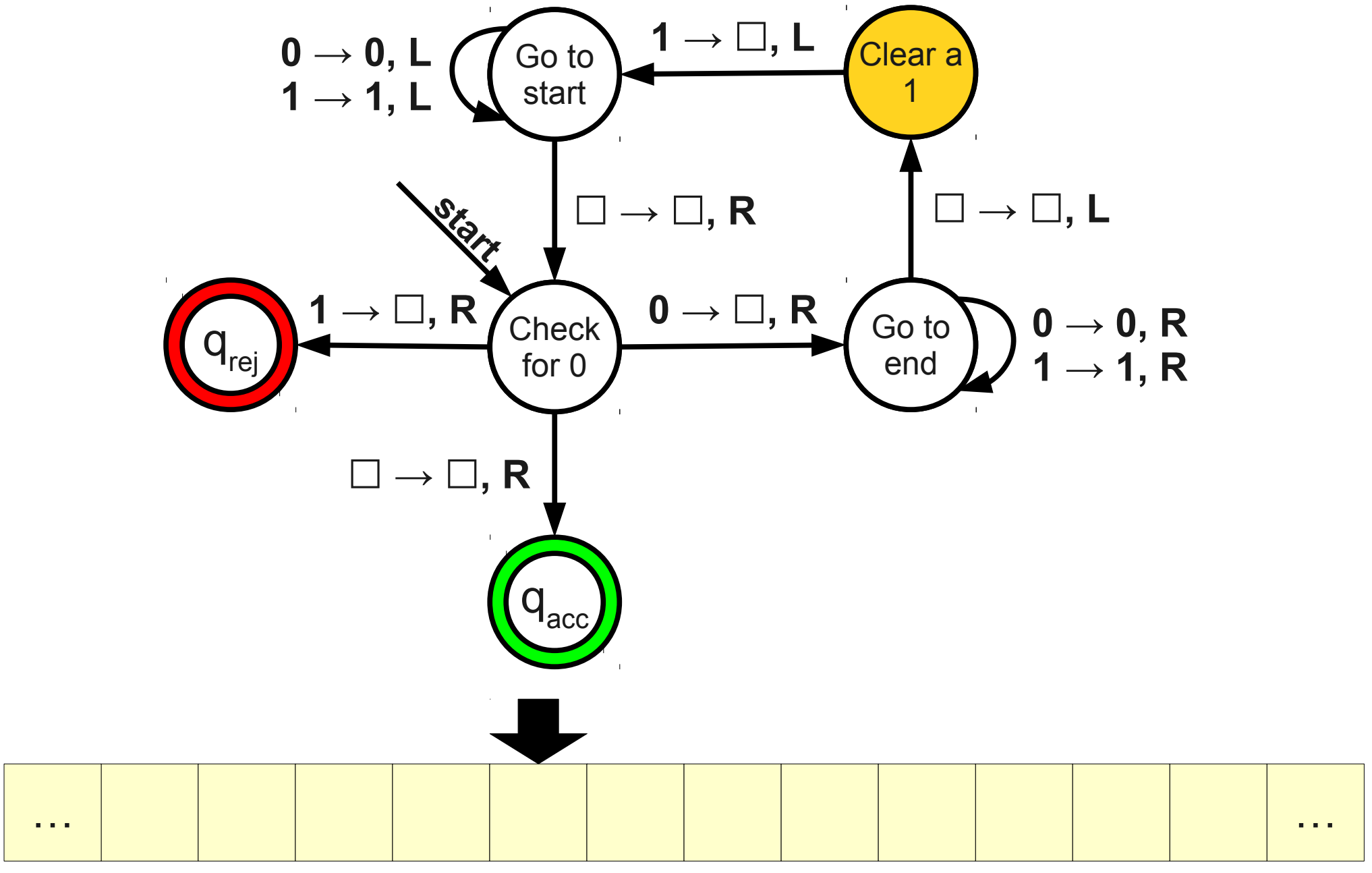

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
1
...

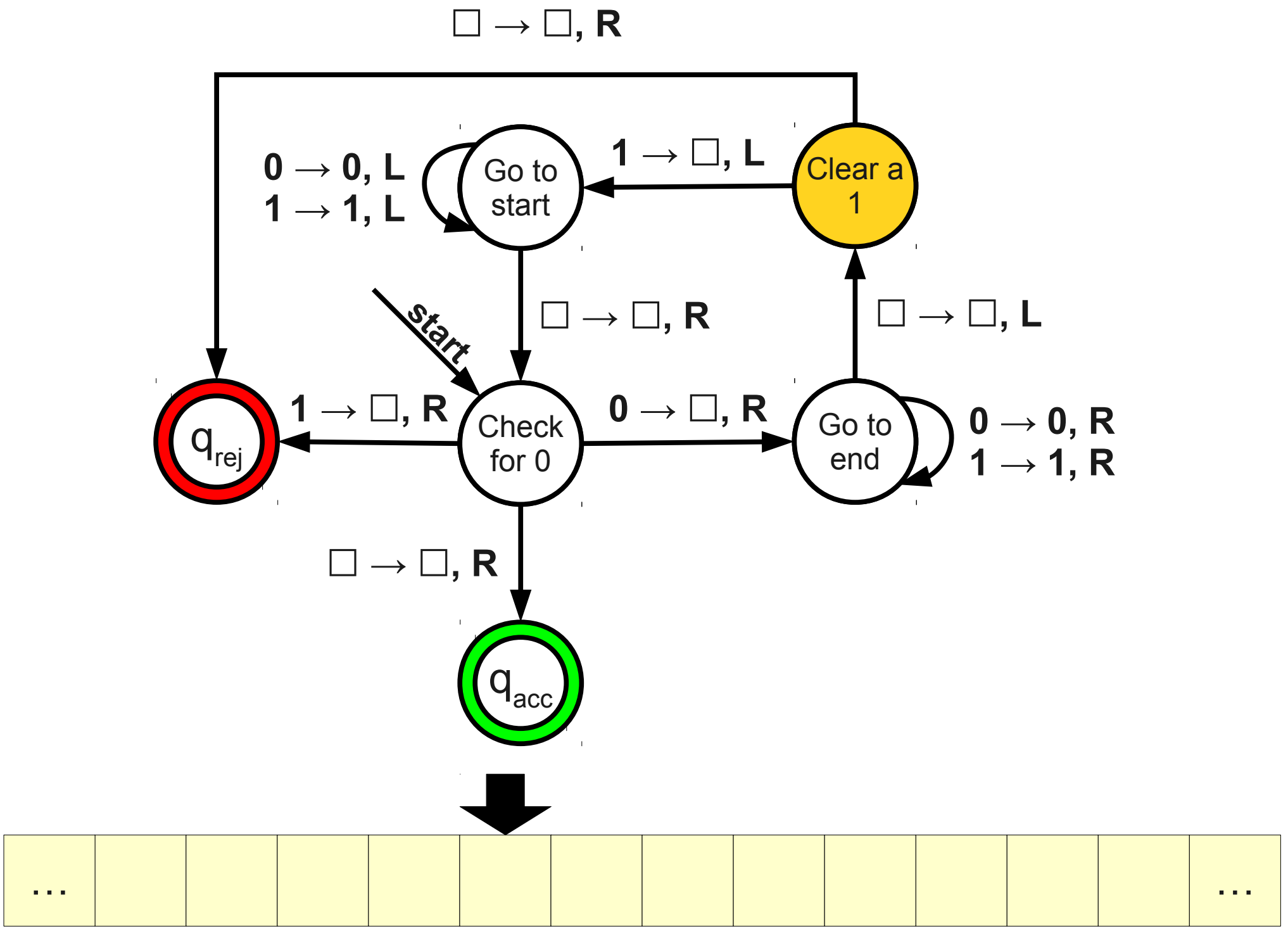

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
...

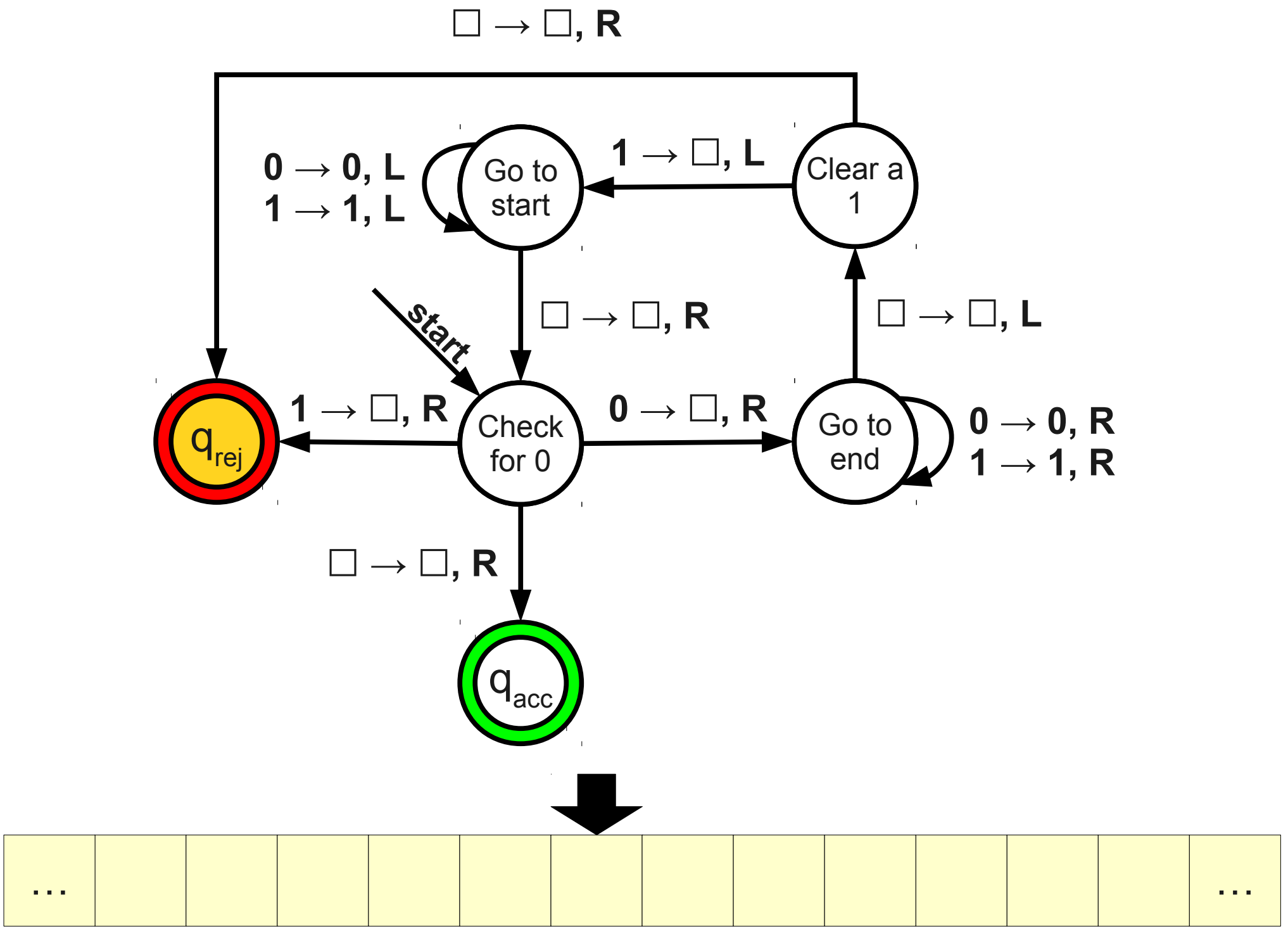

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
...

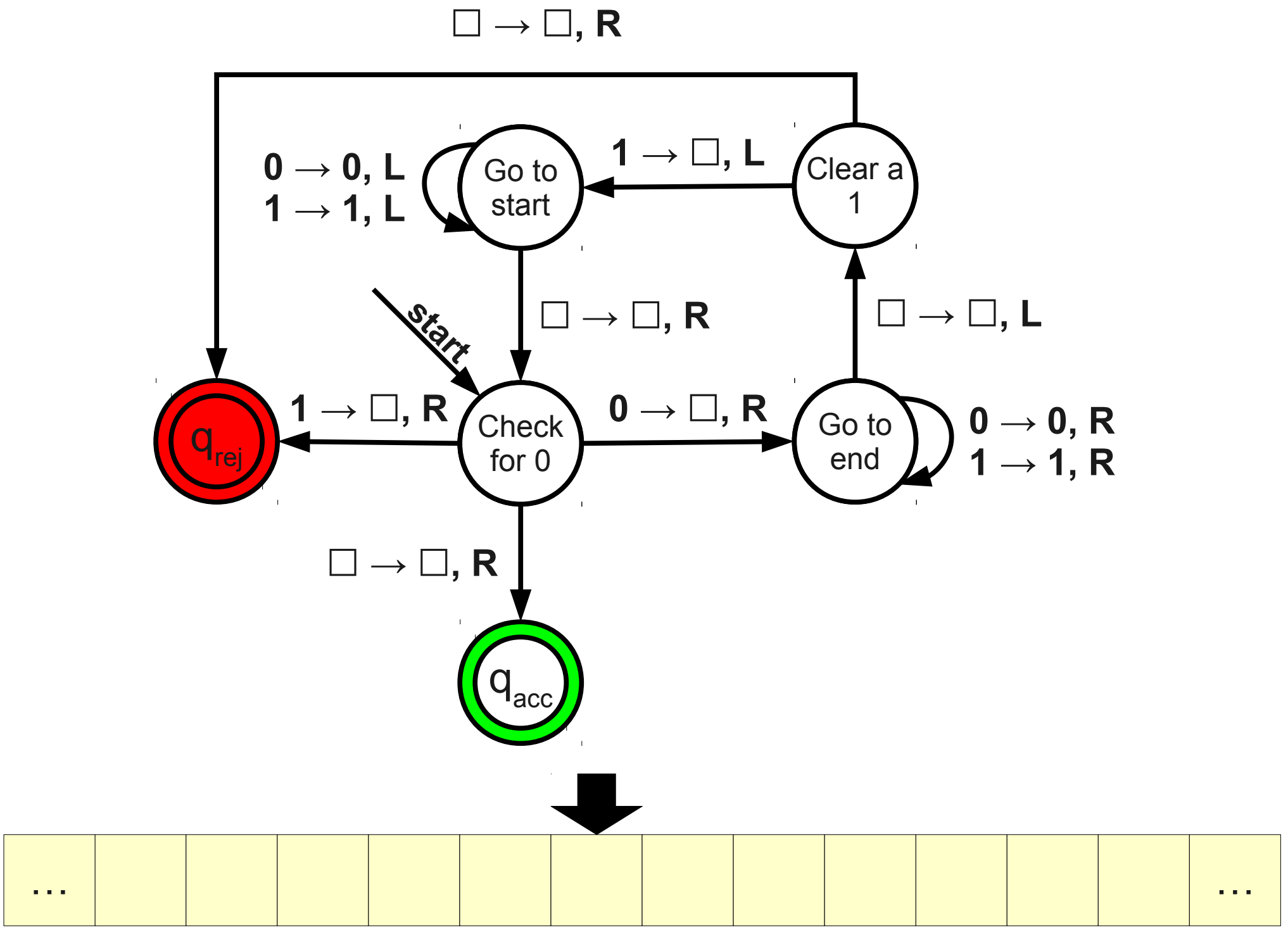

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0
...

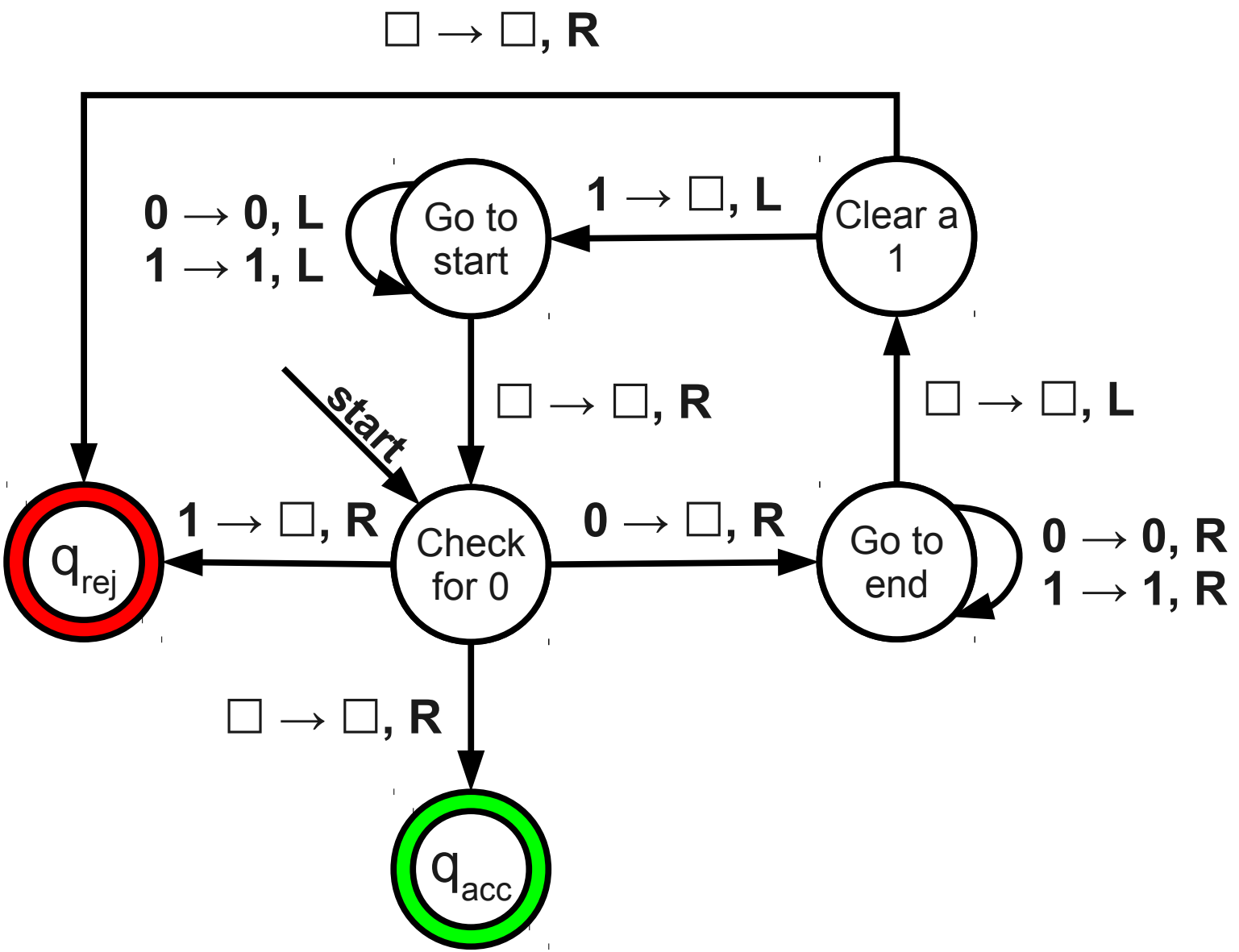

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

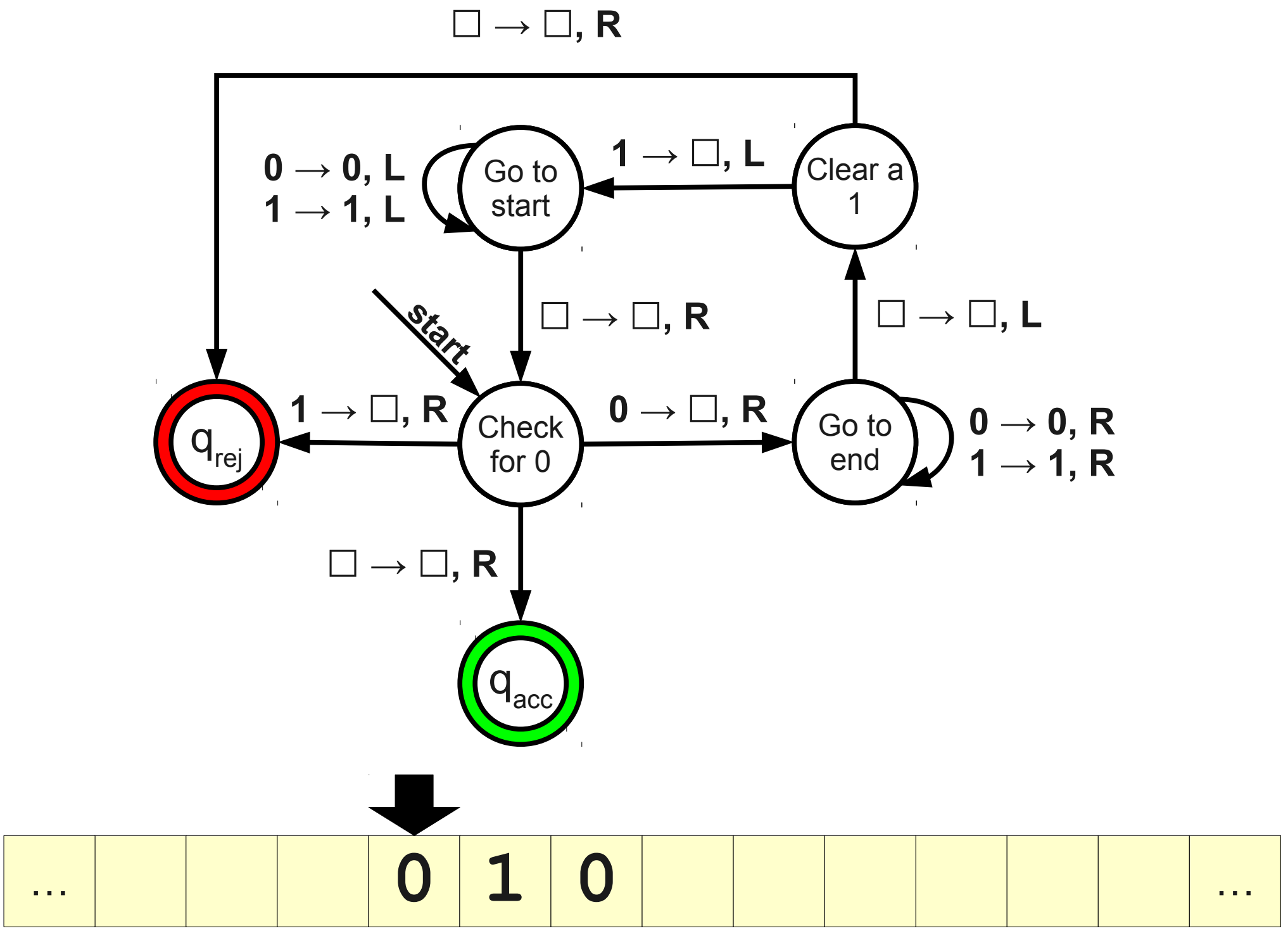

0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

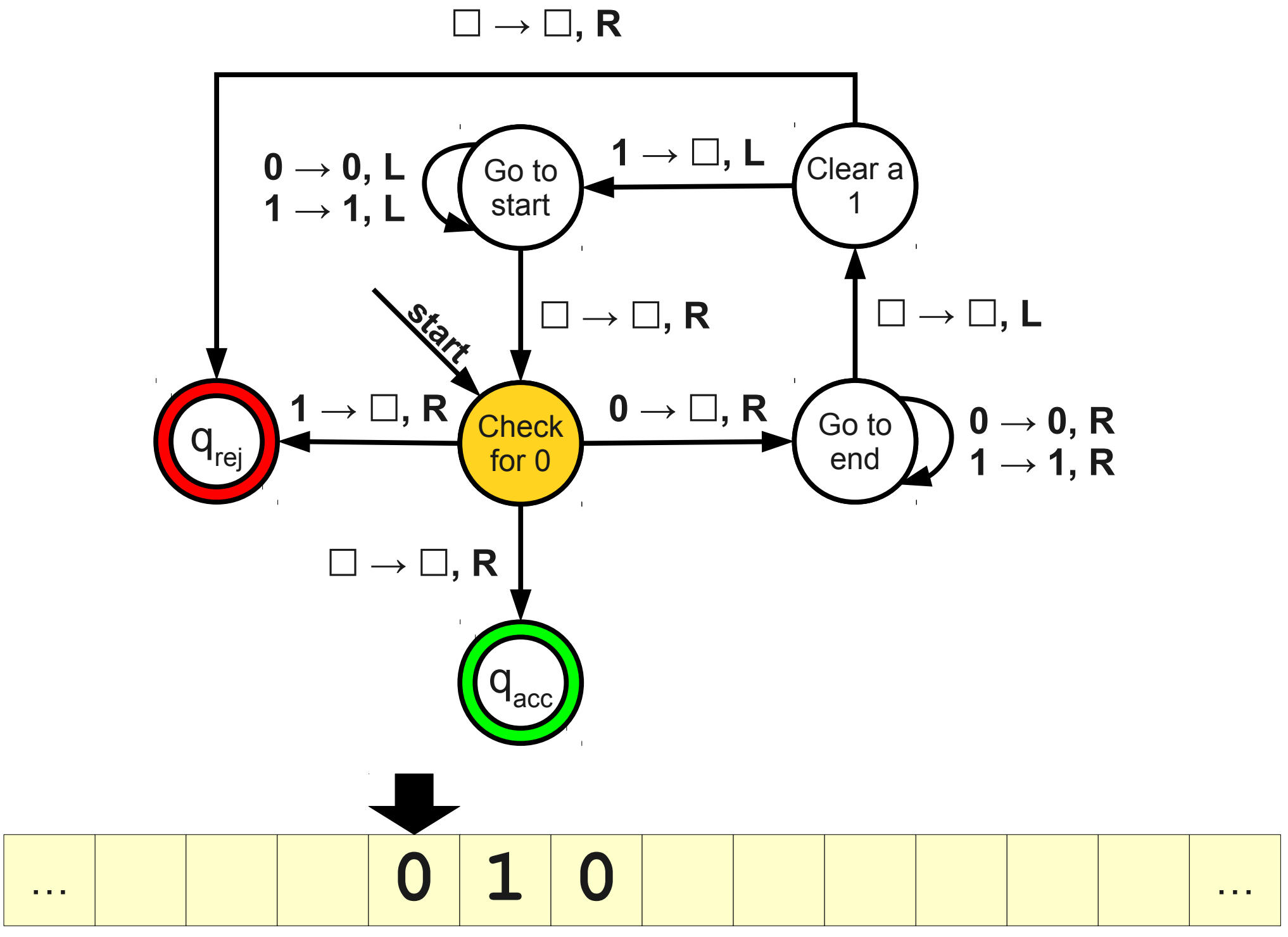

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_{rej}
□ → □, R
q_{acc}
...
...

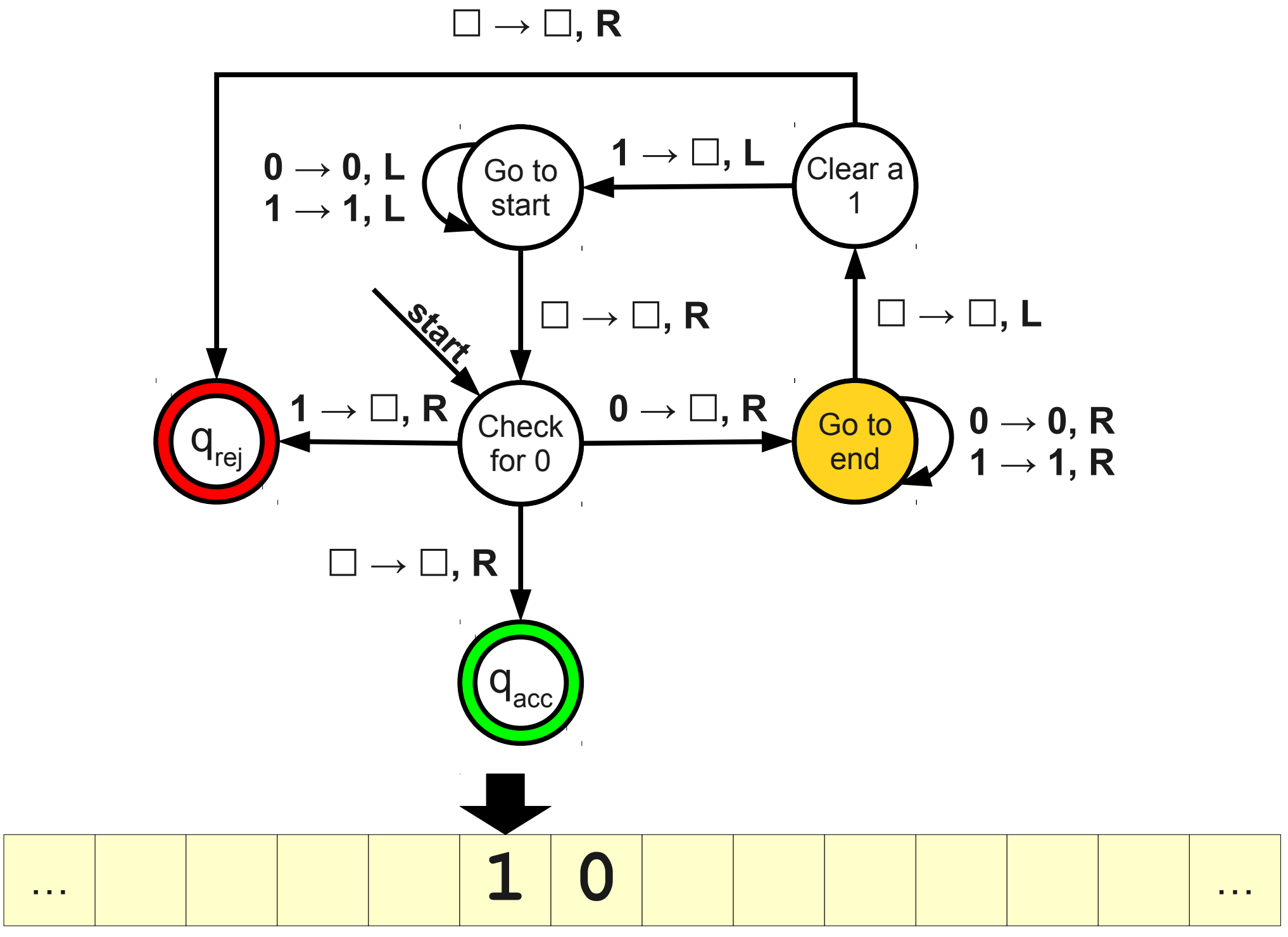

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

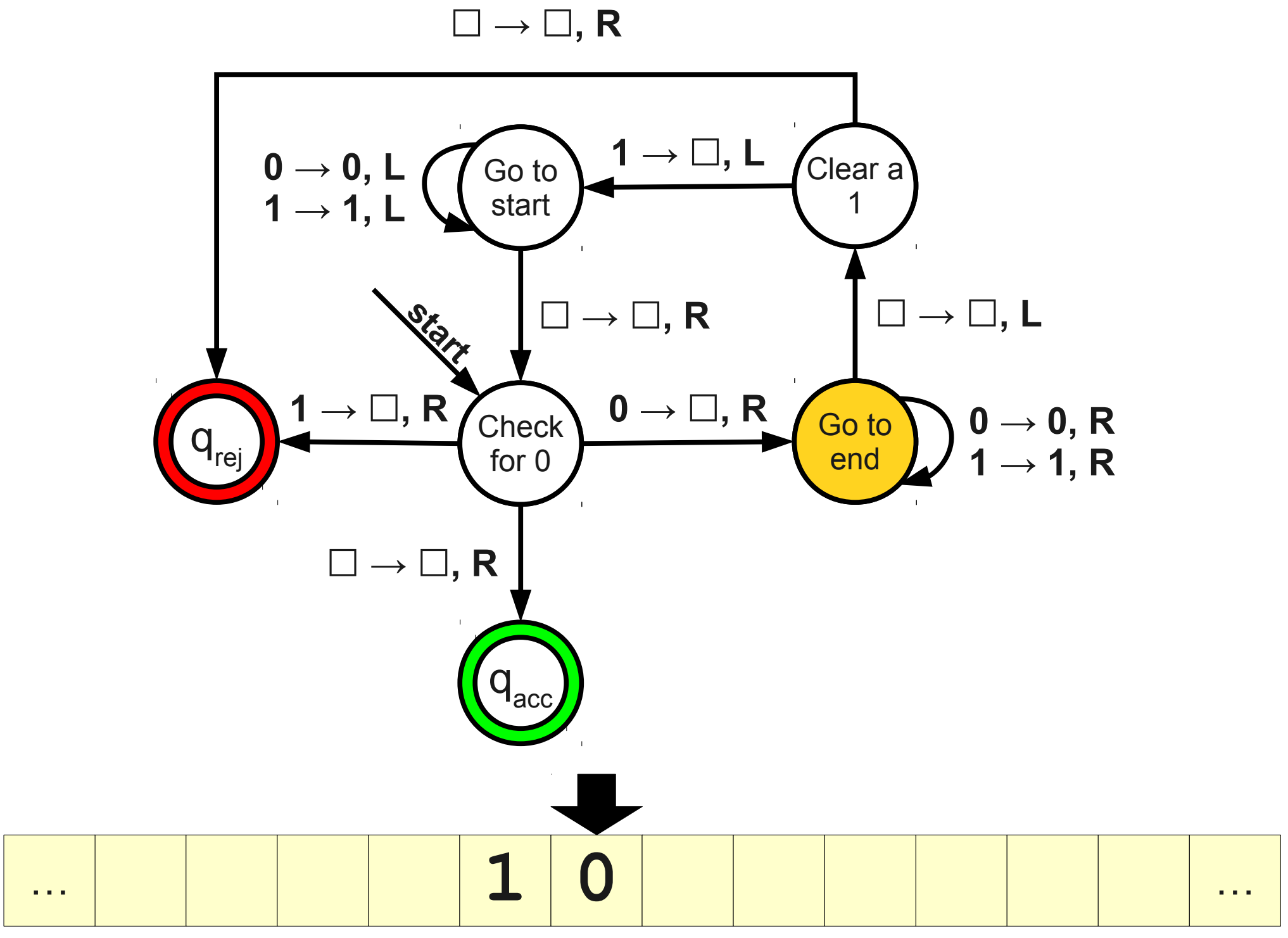

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
...

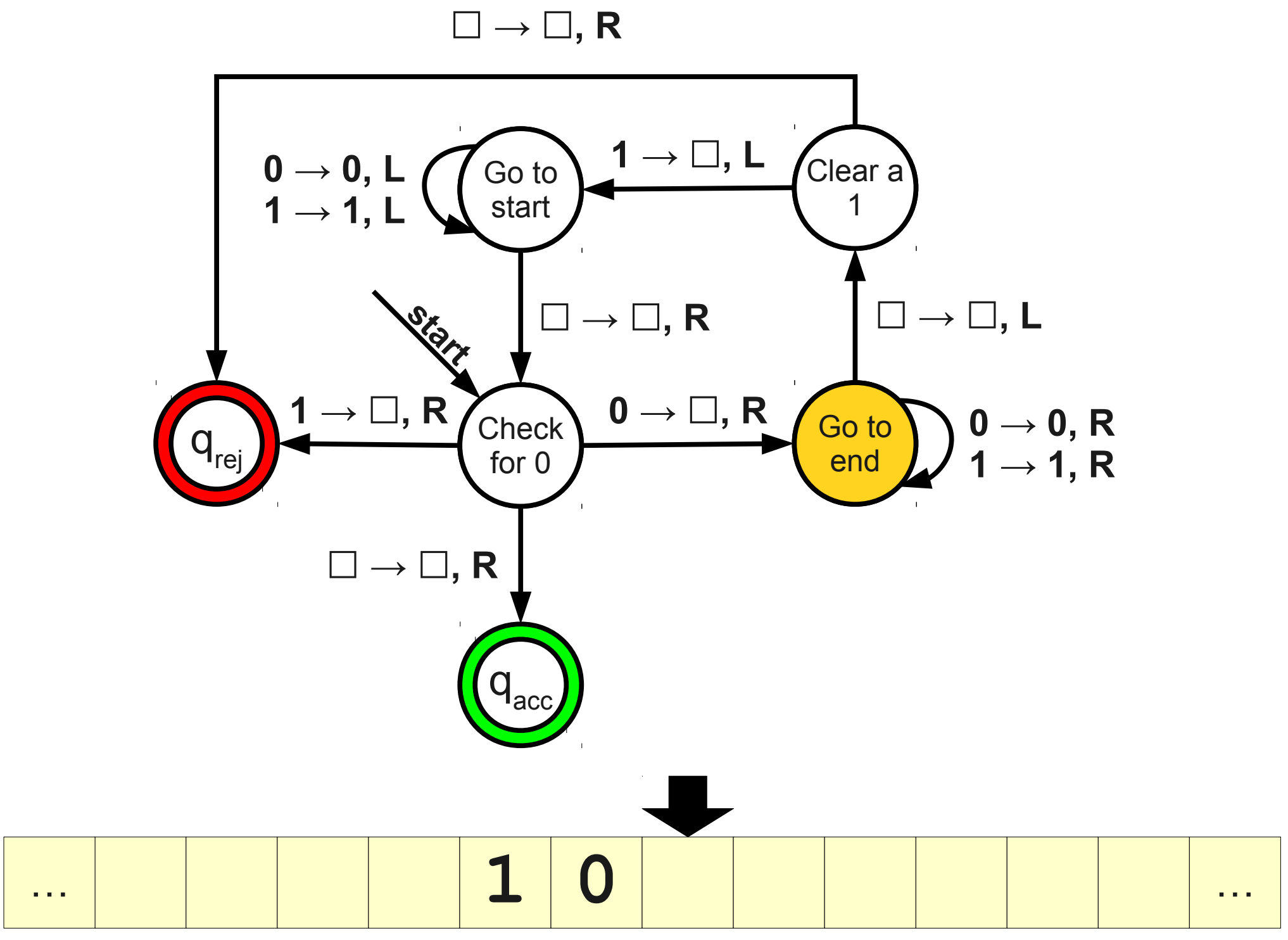

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc

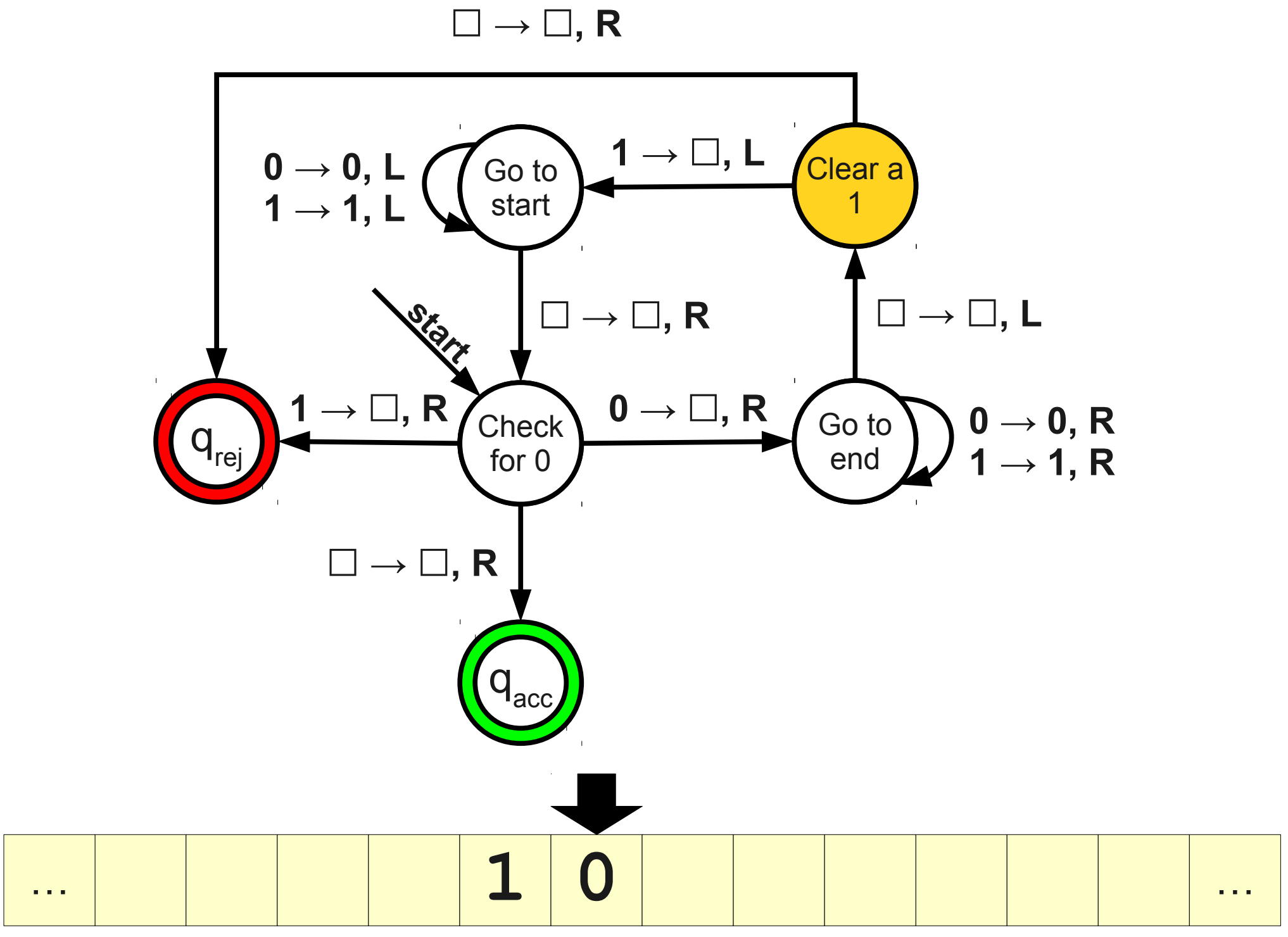

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0 1 0
...

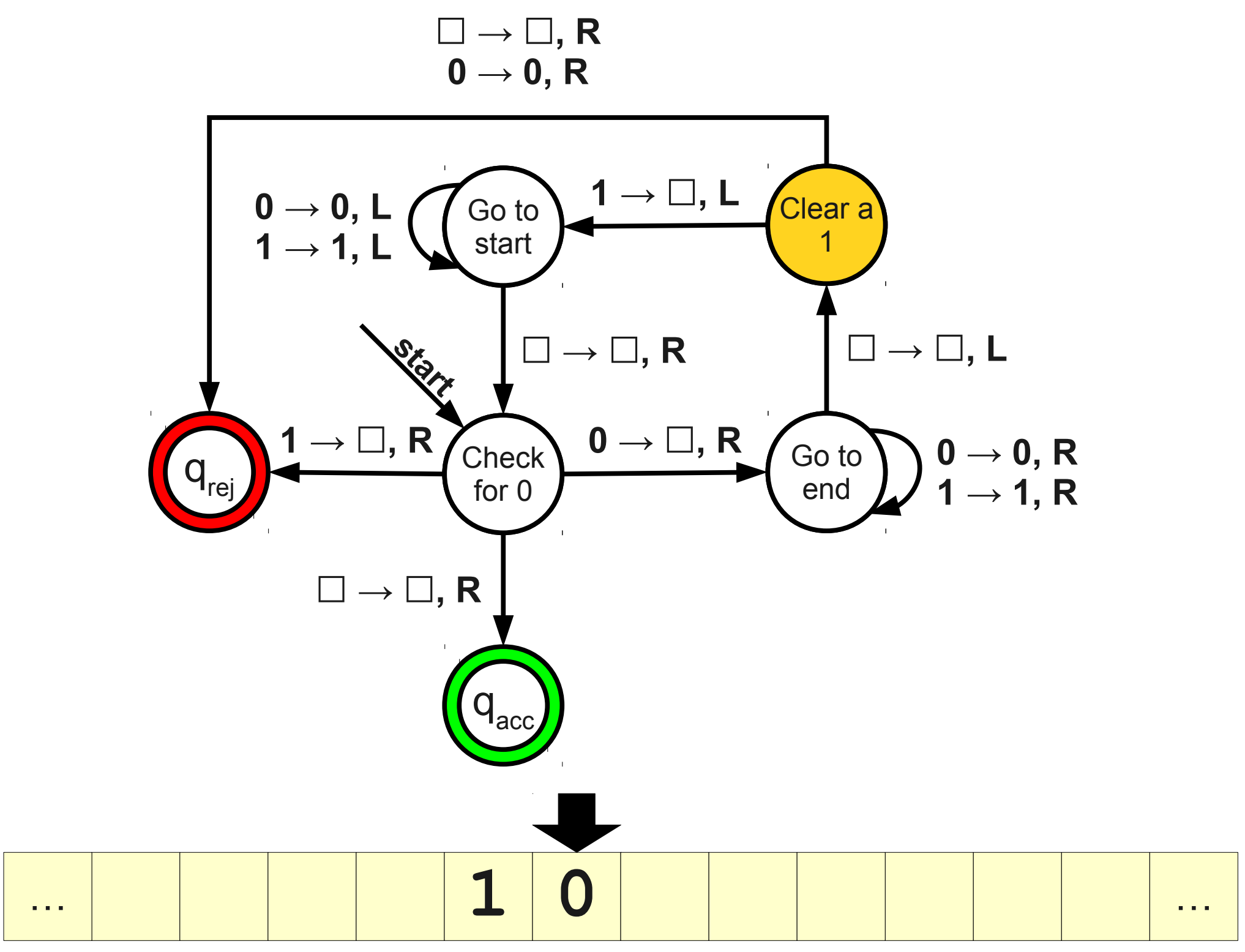

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
0 1 0
...

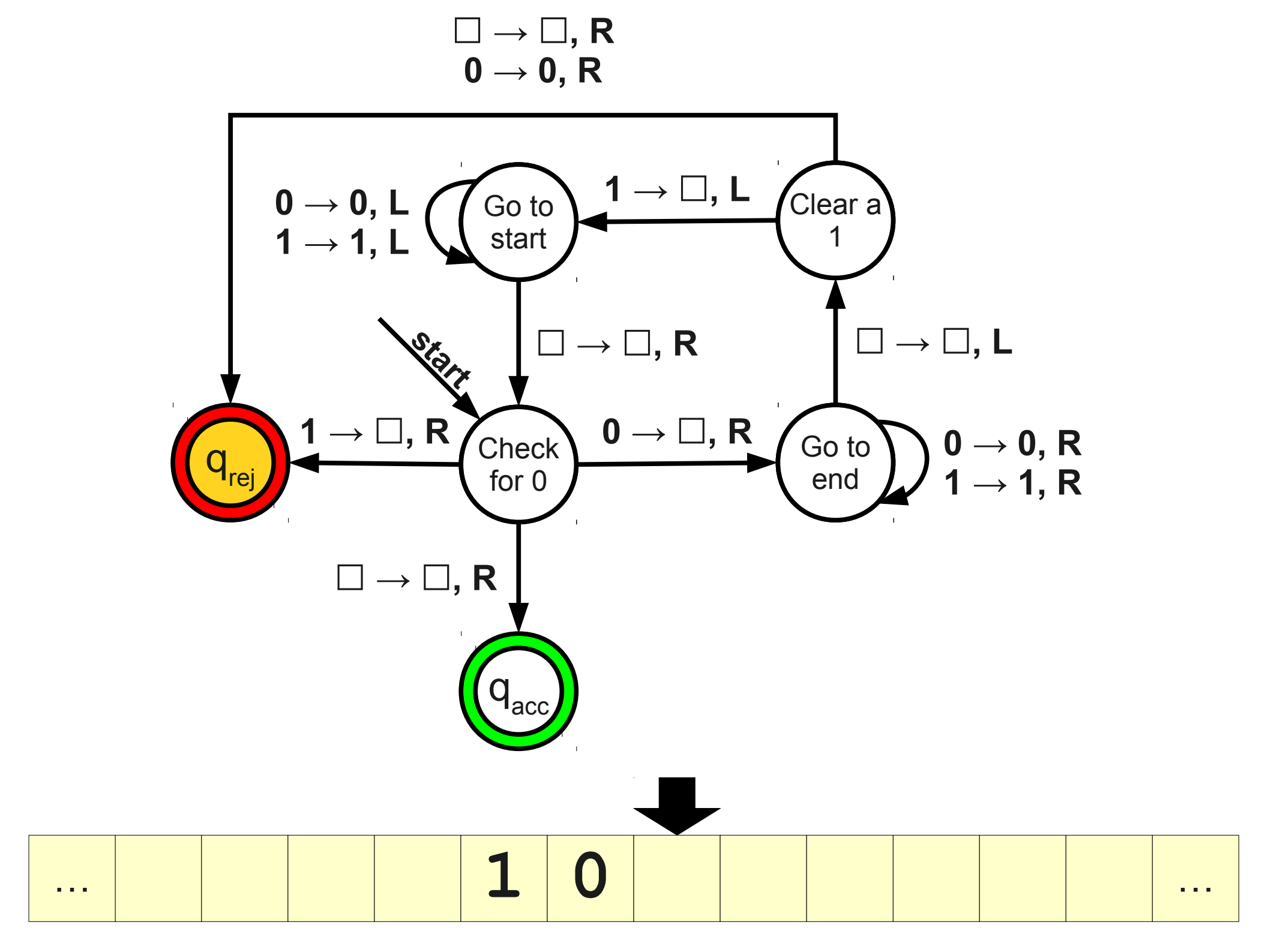

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
0
...

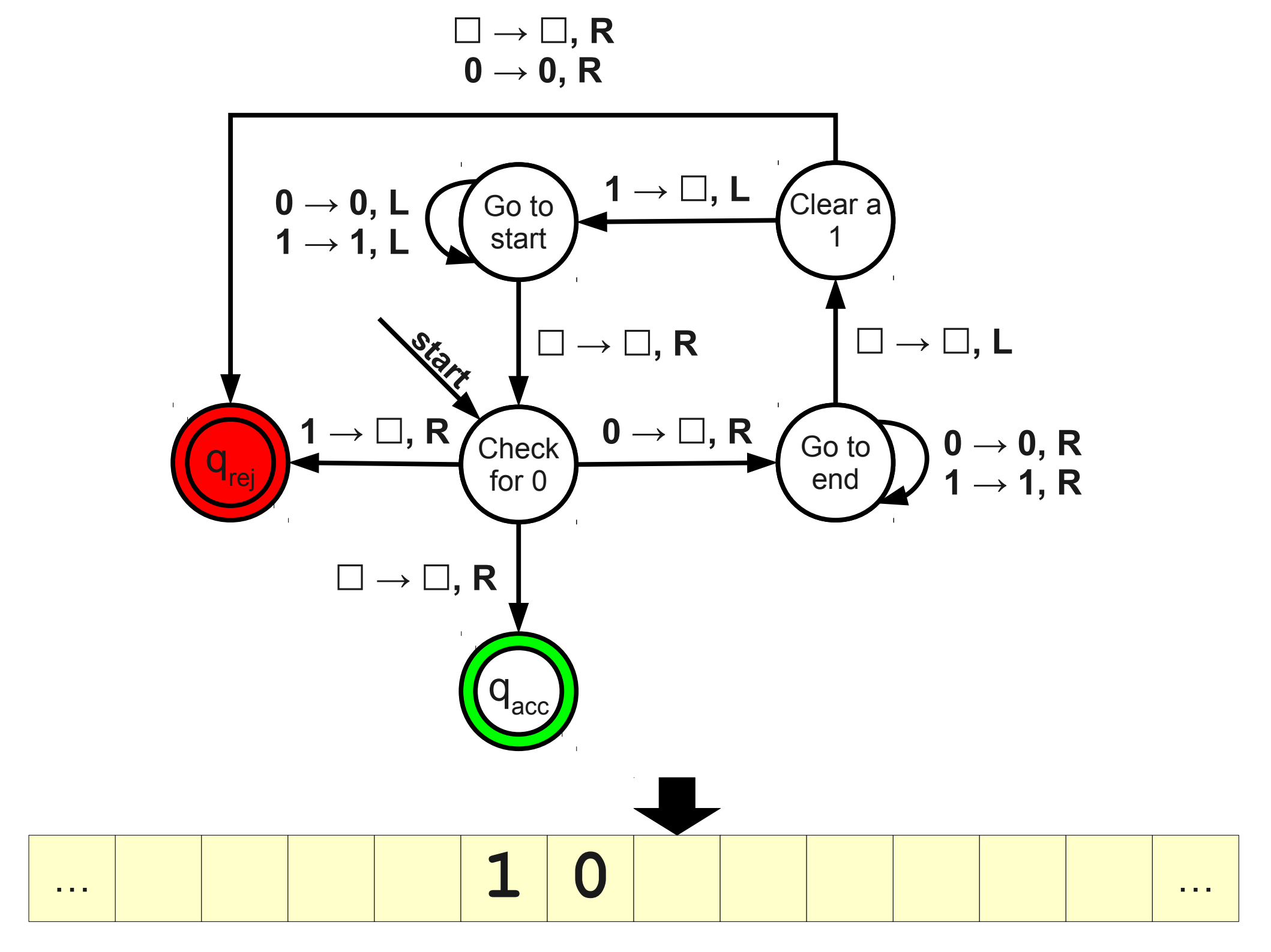

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
0
...

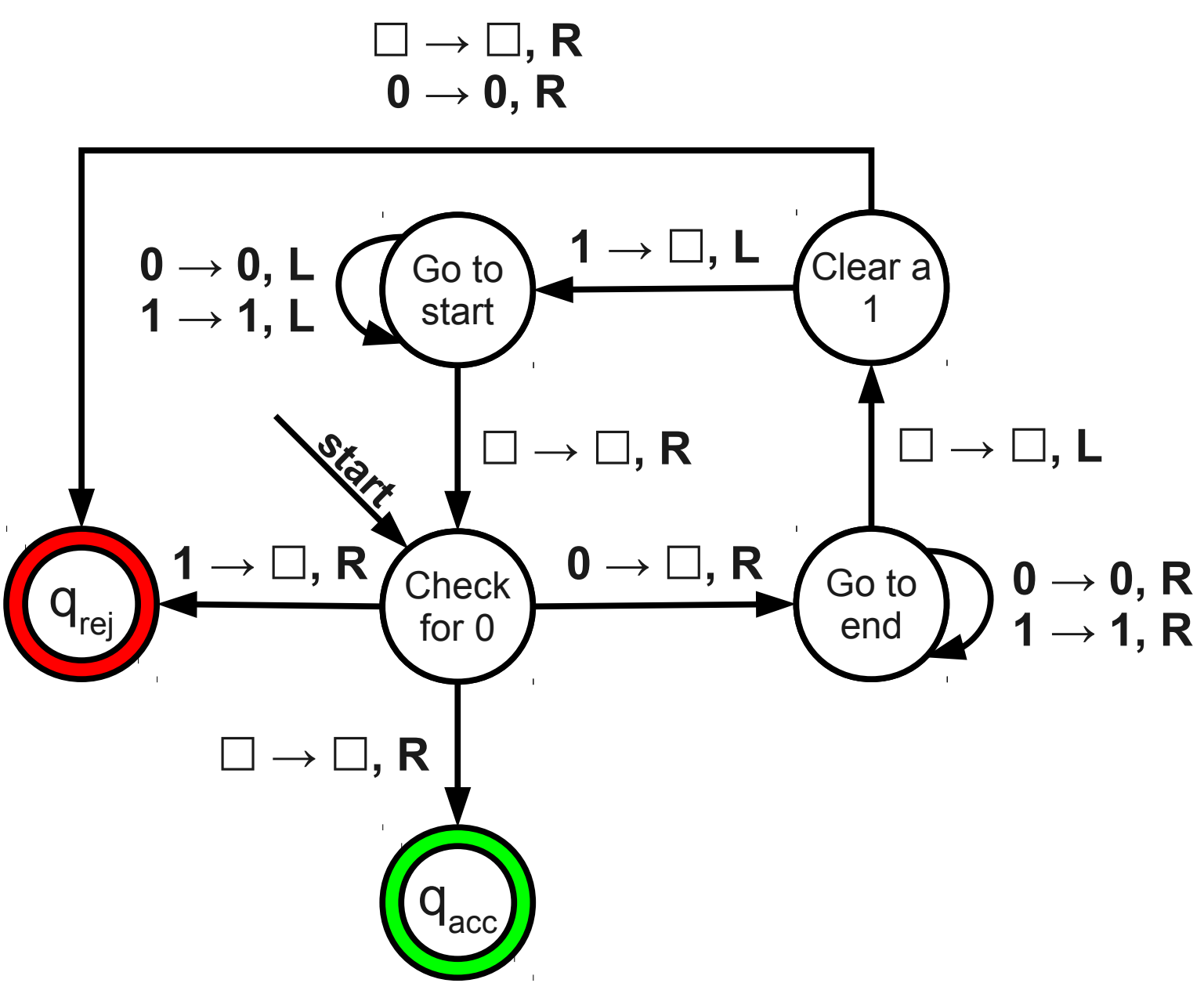

□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1 0
...


□ → □, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
0
...


□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_{rej}
□ → □, R
q_{acc}
...
1
0
...


□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
0
...


□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
q_rej
□ → □, R
q_acc
...
1
0
...


□ → □, R
0 → 0, R
0 → 0, L
1 → 1, L
Go to start
1 → □, L
Clear a 1
start
□ → □, R
□ → □, L
1 → □, R
Check for 0
0 → □, R
Go to end
0 → 0, R
1 → 1, R
□ → □, R
q_rej
q_acc

Time-Out For Announcements!

Problem Set Six

- Problem Set Six out, due next Monday at 2:15PM.
  - Explore the limits of regular languages!
  - Play around with context-free languages!
- New office hours schedule to be released soon; stay tuned!

Midterms Graded

- Midterms have been graded and will be returned at end of lecture.
- **We curve generously in this course**. Look at your *relative* score rather than your *raw* score.
- Solutions and graded exams will be released at end of lecture. You can pick them up in the return filing cabinet in Gates if you don't pick it up today.

Talk this Thursday

- Charlie Hale of Google[x] will be talking about policy implications of new technology.
- This Thursday, November 7 from 6:30PM – 7:30PM in Gates 463.
- Dinner will be served; please RSVP so we can estimate headcount!
- Totally optional, but should be a lot of fun!

Your Questions

“Can you give us any information about how scores on problem sets/exams translate into letter grades for this course? What are some ballpark numbers that would translate into an A, B, etc?”

More Turing Machines

Multiplication

- Let $\Sigma = \{\texttt{1}, \texttt{×}, \texttt{=}\}$ and consider the language $L = \{ \texttt{1}^m\texttt{×}\texttt{1}^n\texttt{=}\texttt{1}^{mn} \mid m, n \in \mathbb{N} \}$
- This language is ***not*** regular (can prove using the Myhill-Nerode theorem)
- This language is ***not*** context-free (can prove this with the **pumping lemma for context-free languages**, though we didn't cover it this quarter).
- Can we build a TM for it?

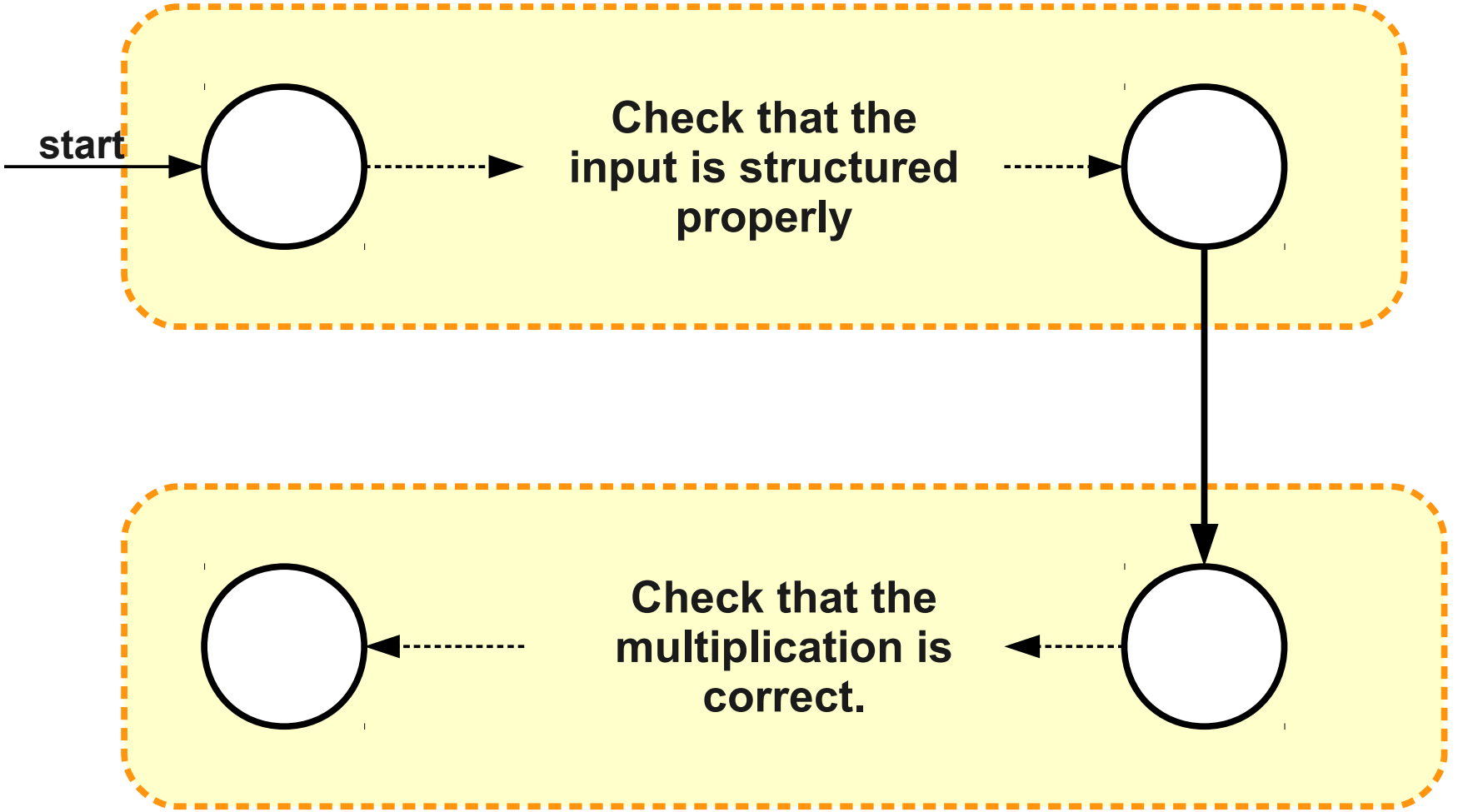
Things To Watch For

- The input has to have the right format.
  - Don't allow `11==×11×`, etc.
- The input must do the multiplication correctly.
  - Don't allow `11×11=11111`, for example.
- How do we handle this?

Key Idea: Subroutines

- A **subroutine** of a Turing machine is a small set of states in the TM such that performs a small computation.
- Design states where
  - There is a designated **entry** state where the subroutine begins, and
  - There is a designated **exit** state where the subroutine ends.
- Design complex TMs by building smaller parts to handle each task.

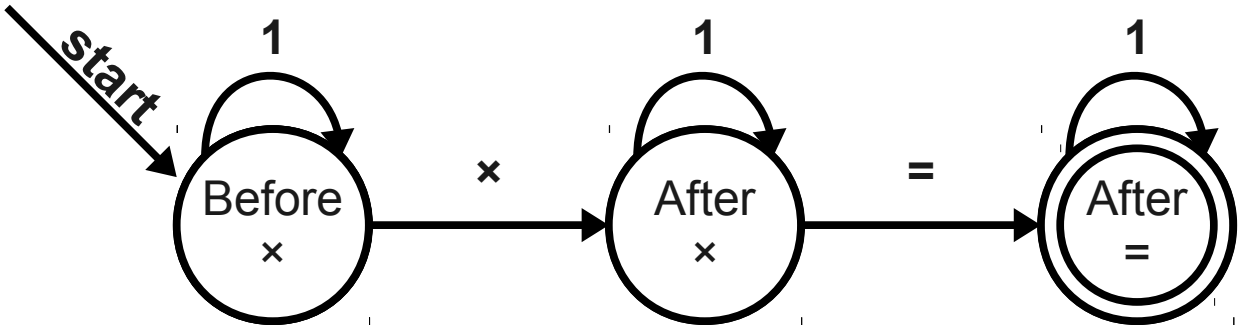
$$L = \{ 1^m \times 1^n = 1^{mn} \mid m, n \in \mathbb{N} \}$$



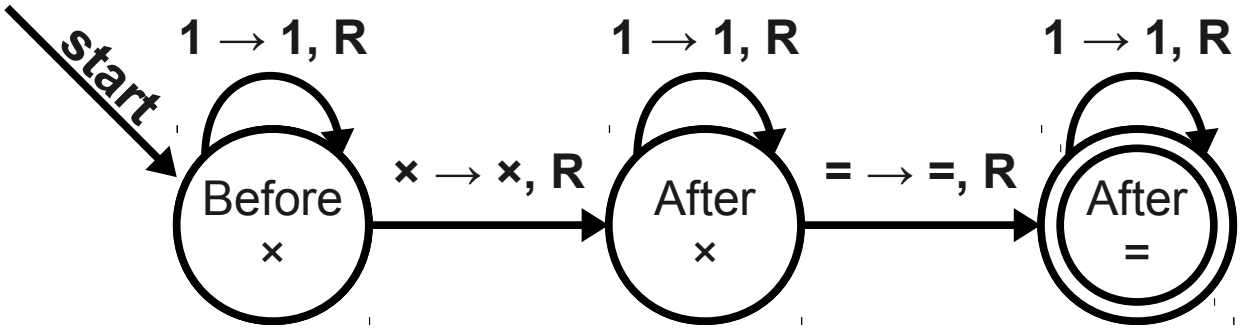
Validating the Input

- First, we need to check that the structure of the input is correct by rejecting strings that aren't of the form $\texttt{1}^m\texttt{×1}^n\texttt{=1}^p$.
- Just need to check the *relative ordering* of the symbols, not the quantities.
- Useful fact: strings are in this relative order iff the string is in the language given by regex $\texttt{1}^*\texttt{×1}^*\texttt{=1}^*$.
- Start with a DFA for this language and convert it to a TM!

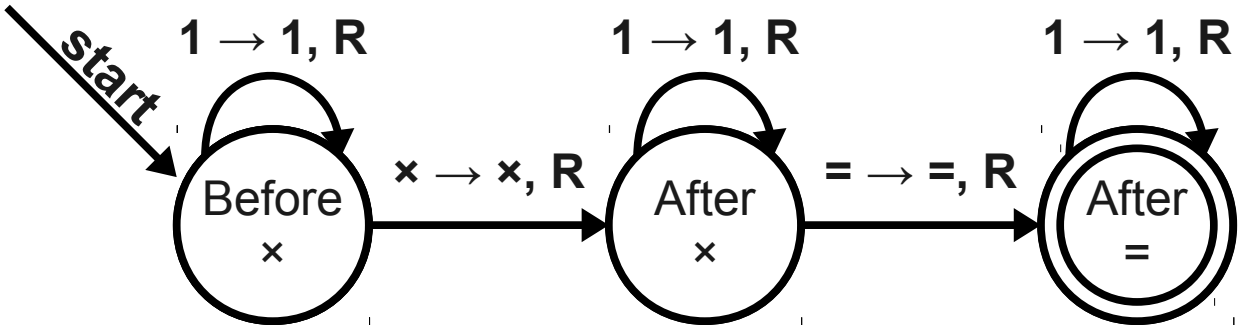
Checking for $1^*\times1^*=1^*$

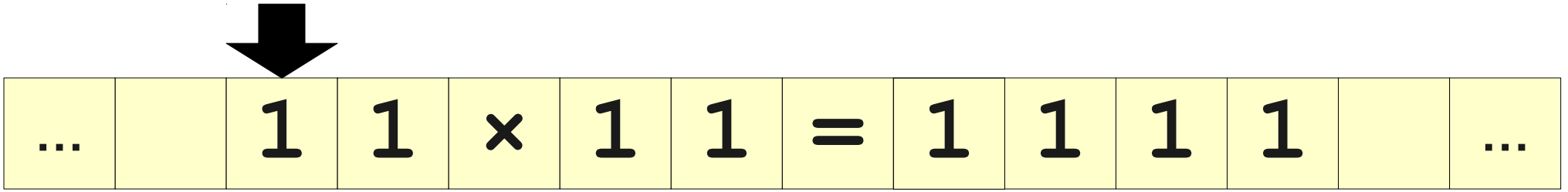


Checking for $1^*\times1^*=1^*$

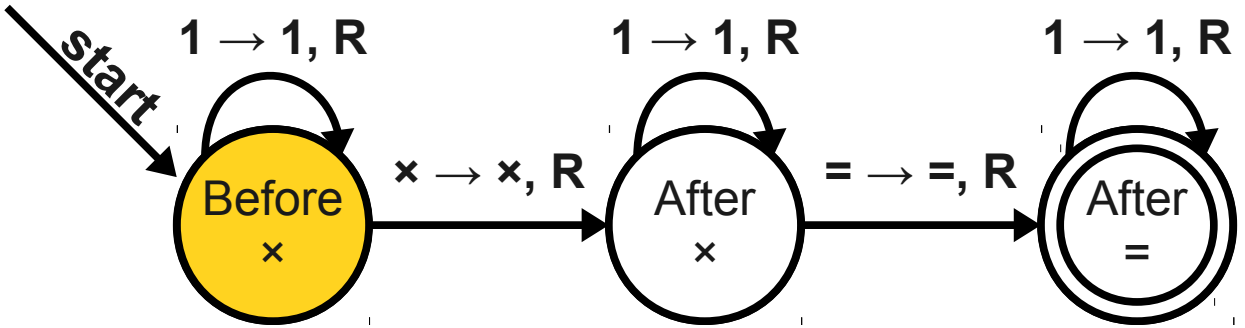


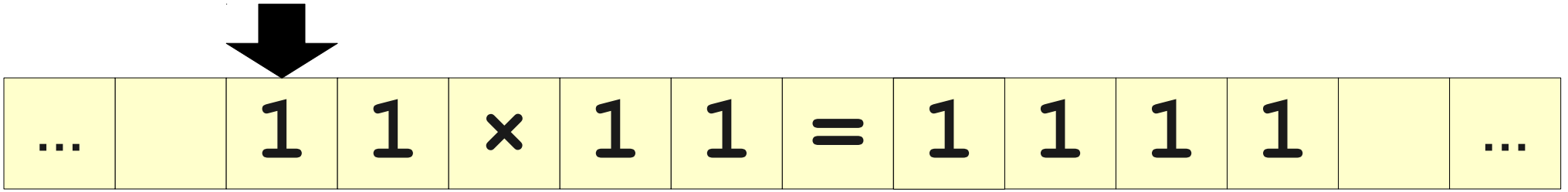
Checking for $1^*\times1^*=1^*$



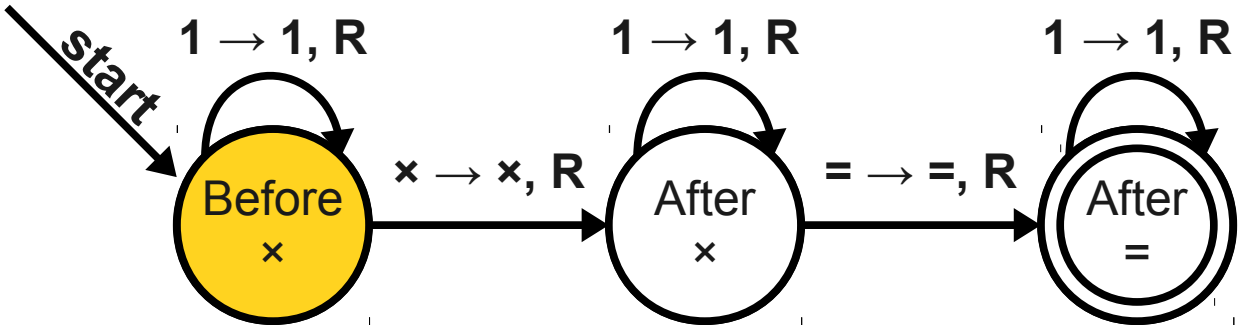


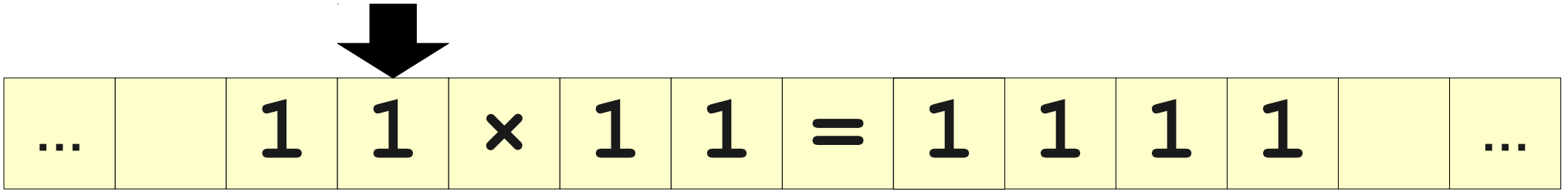
Checking for $1^*\times1^*=1^*$



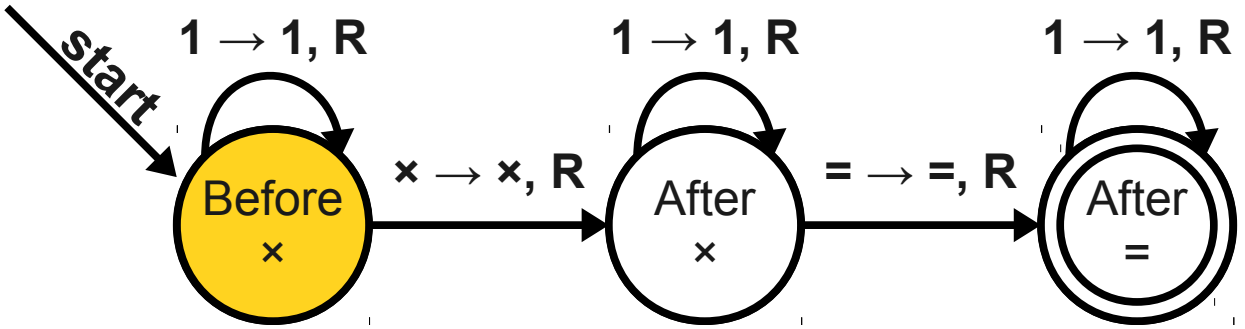


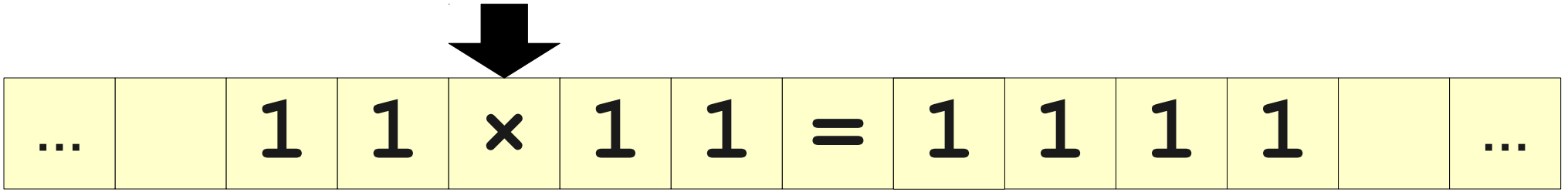
Checking for $1^*\times1^*=1^*$



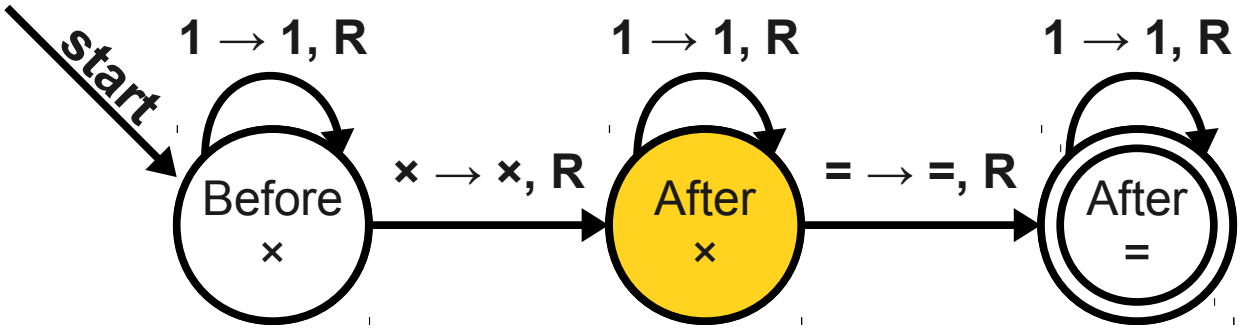


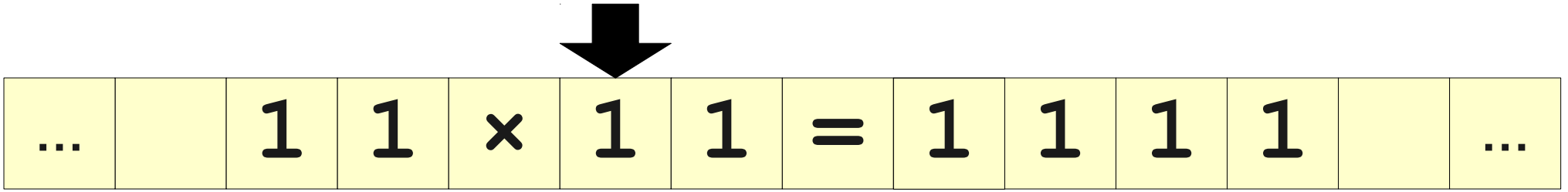
Checking for $1^*\times1^*=1^*$



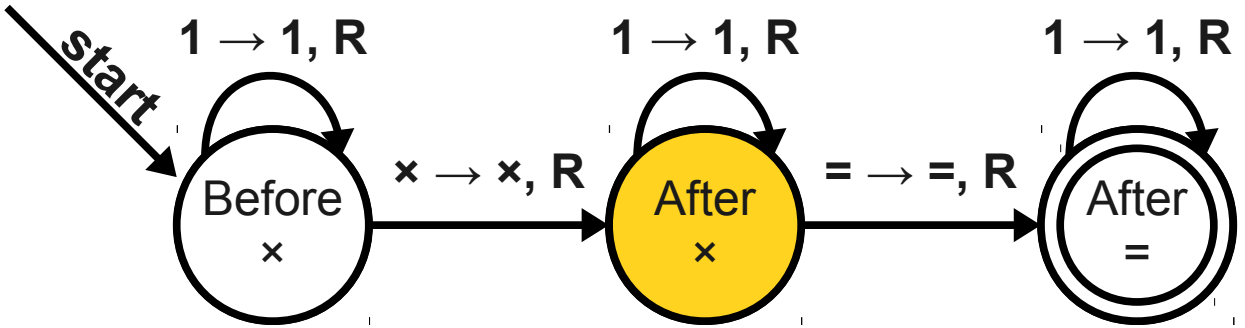


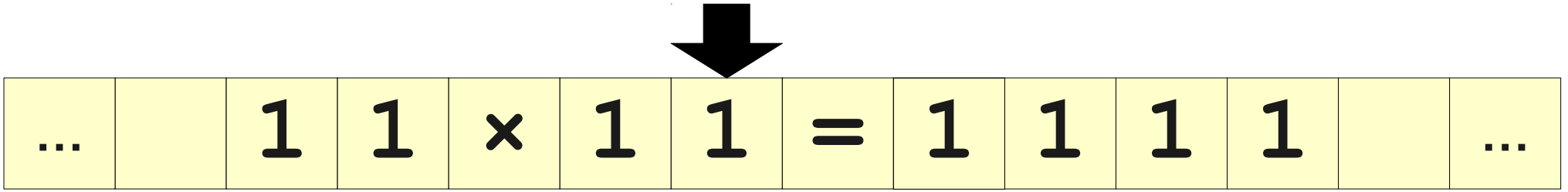
Checking for $1^*\times1^*=1^*$



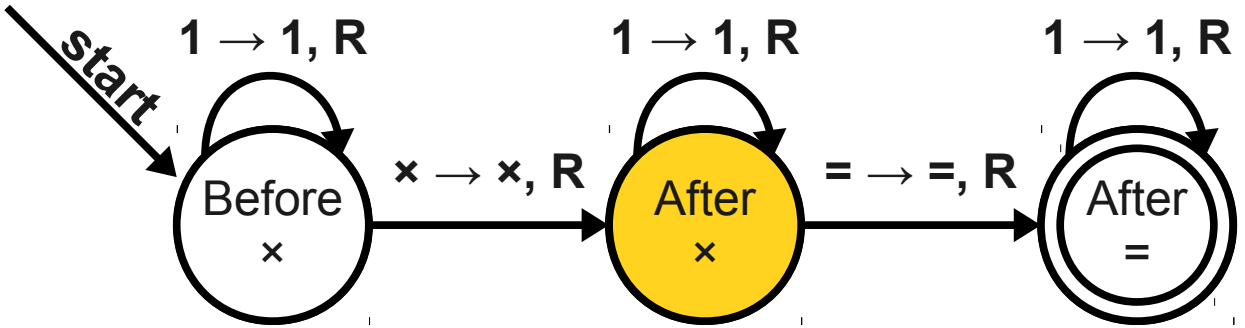


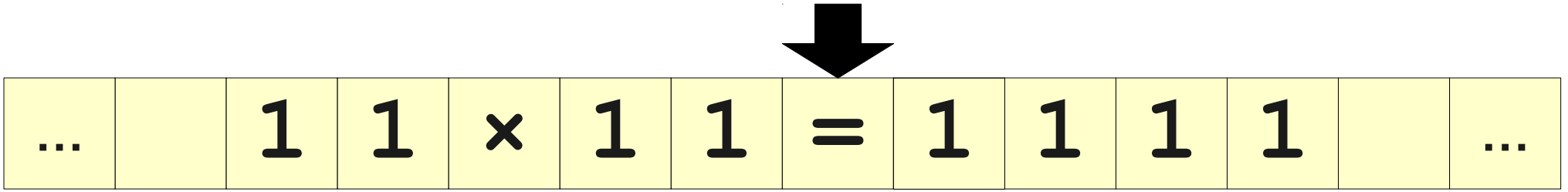
Checking for $1^*\times1^*=1^*$



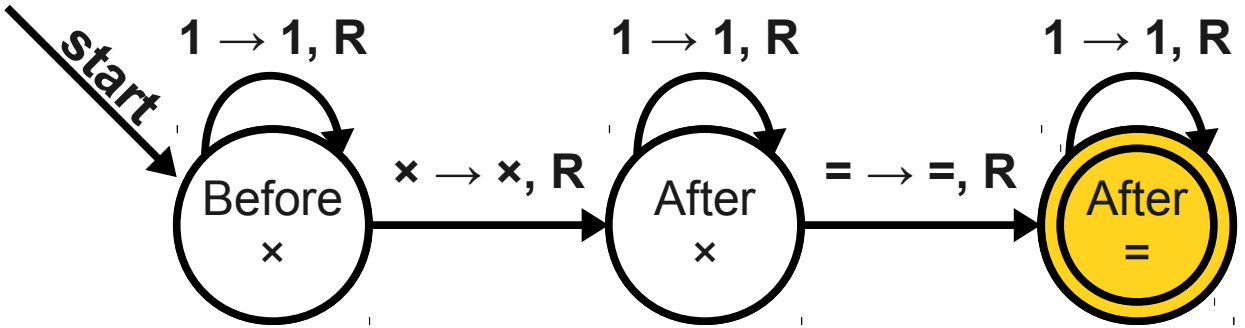


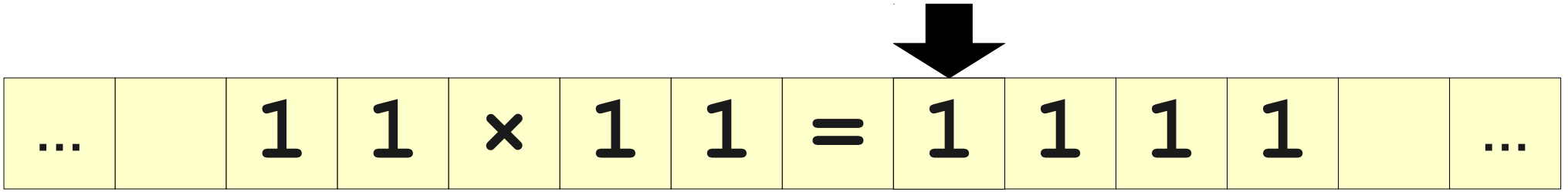
Checking for $1^*\times1^*=1^*$



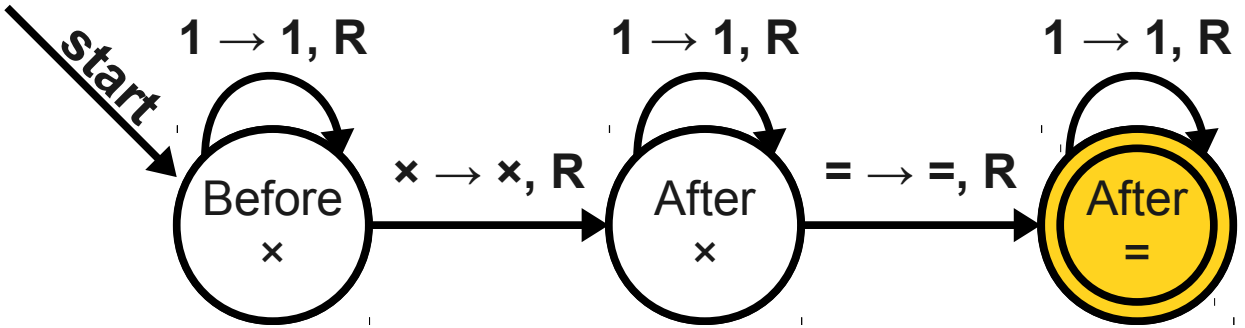


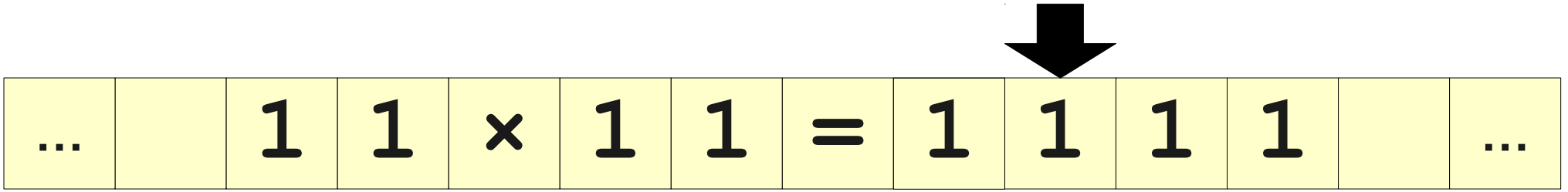
Checking for $1^*\times1^*=1^*$



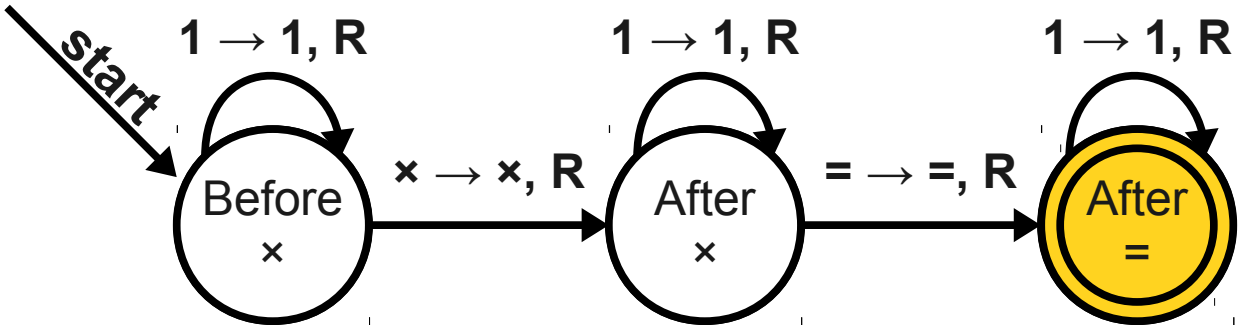


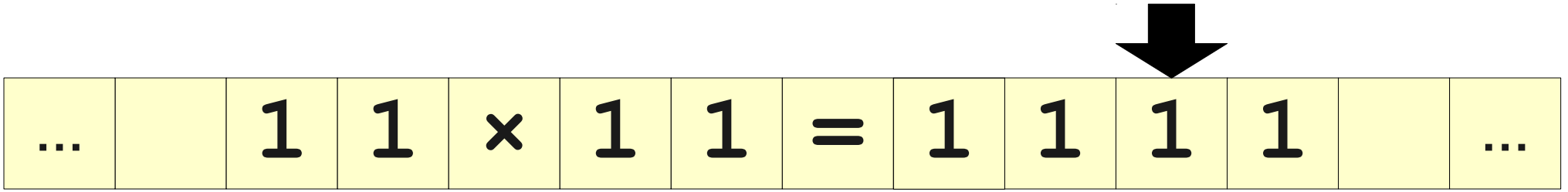
Checking for $1^*\times1^*=1^*$



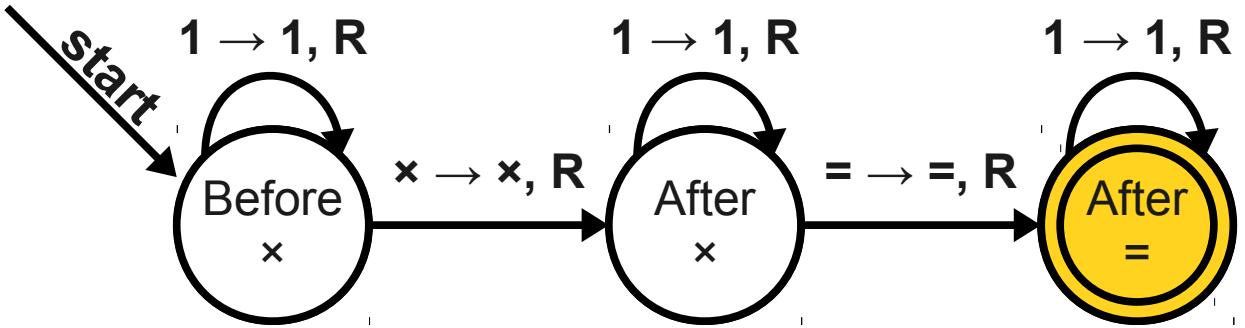


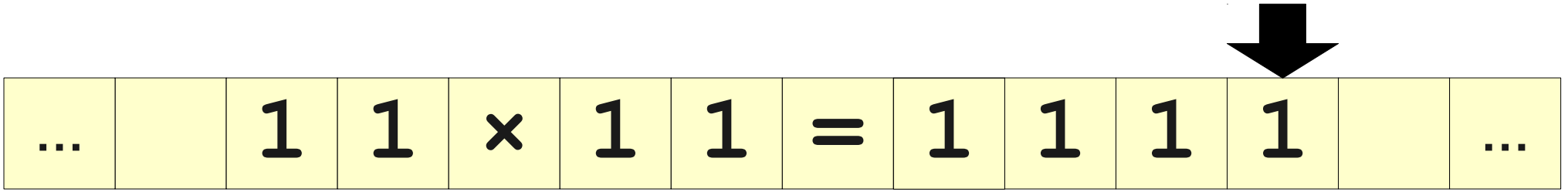
Checking for 1*×1*=1*



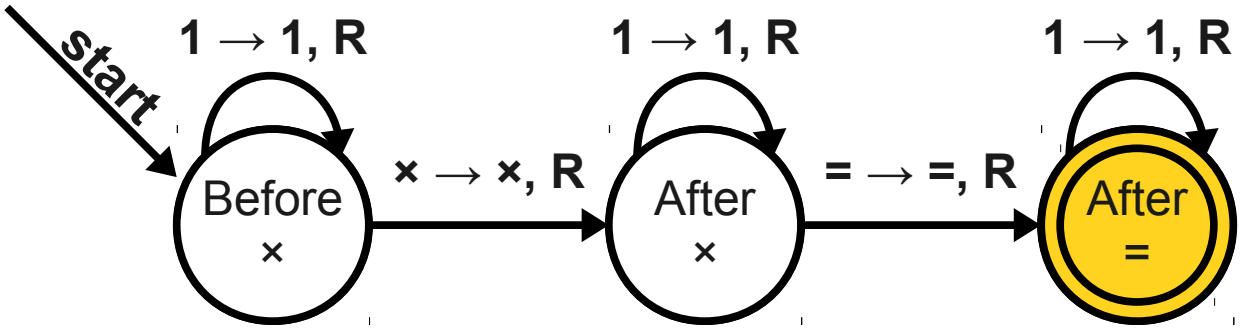


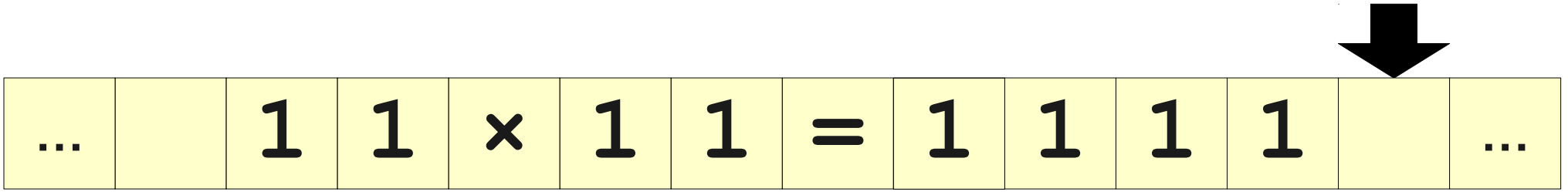
Checking for $1^*\times1^*=1^*$



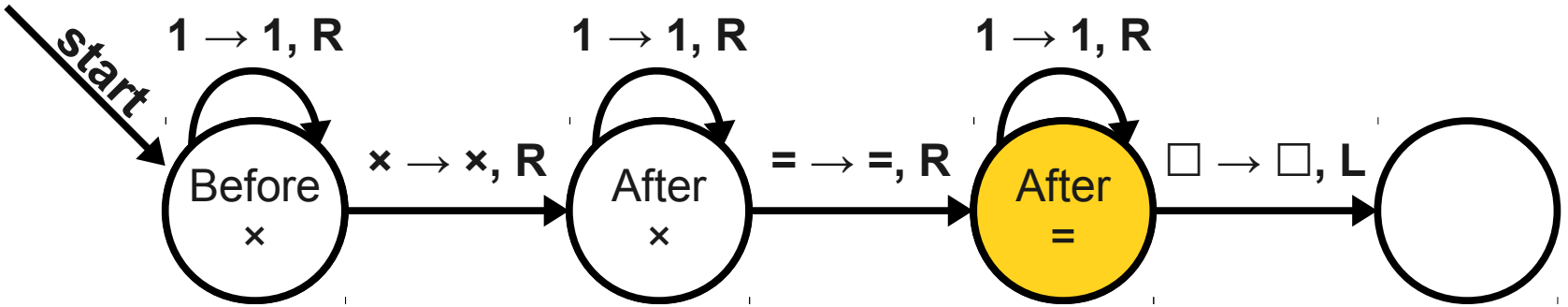


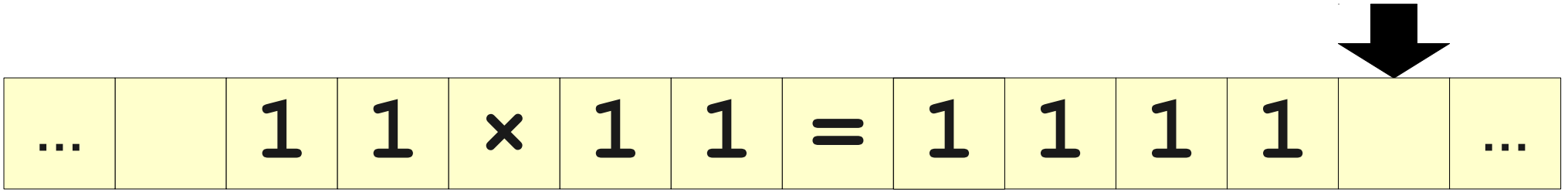
Checking for $1^*\times1^*=1^*$



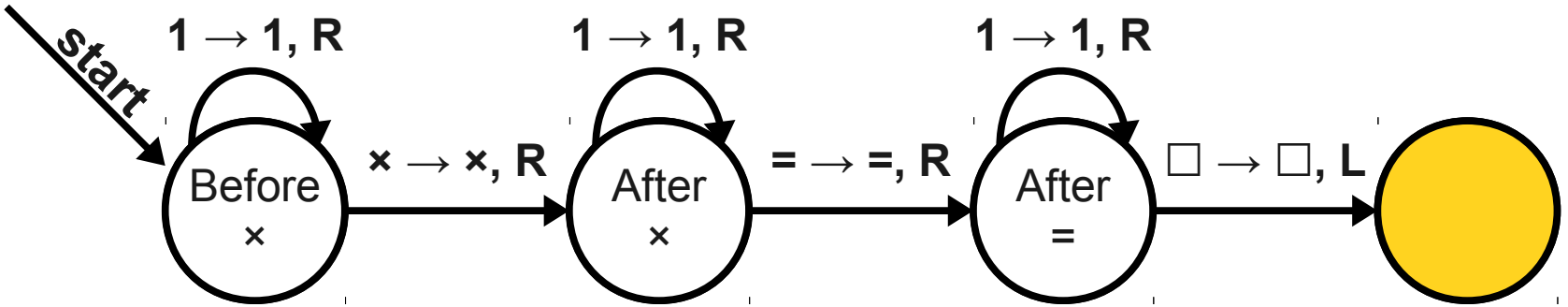


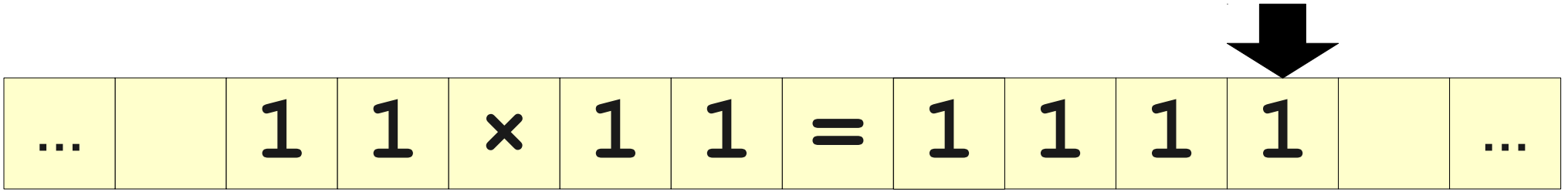
Checking for $1^*\times1^*=1^*$



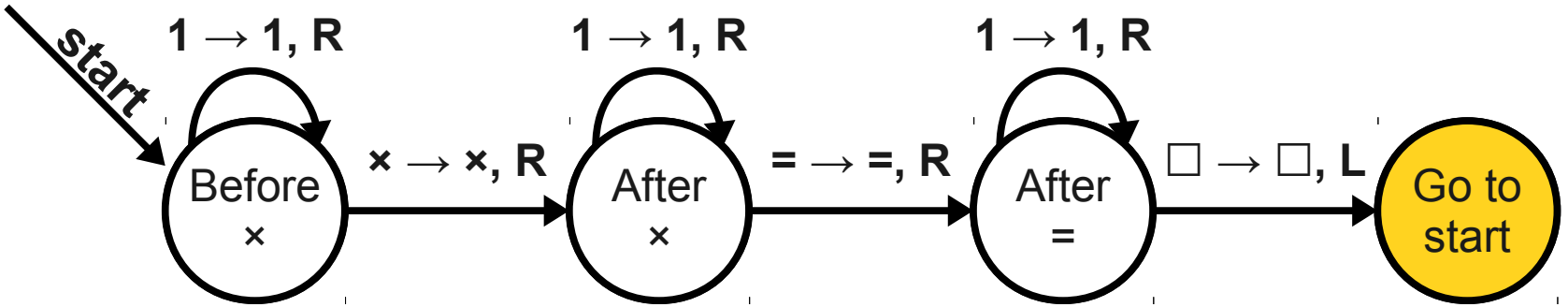


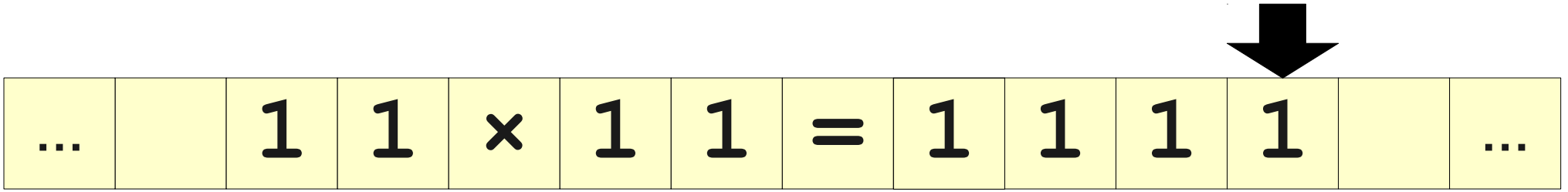
Checking for $1^*\times1^*=1^*$



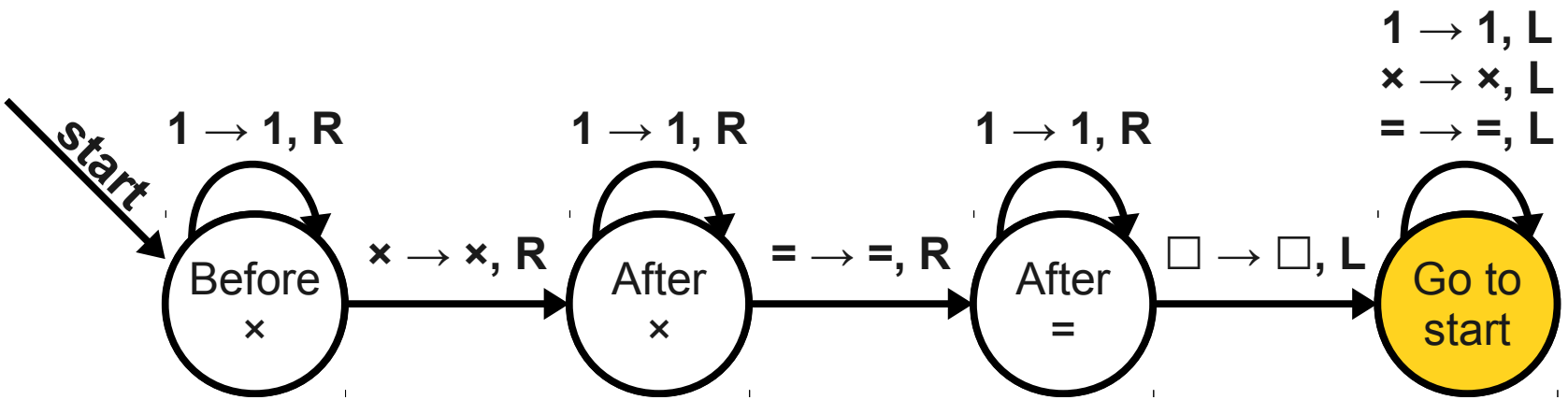


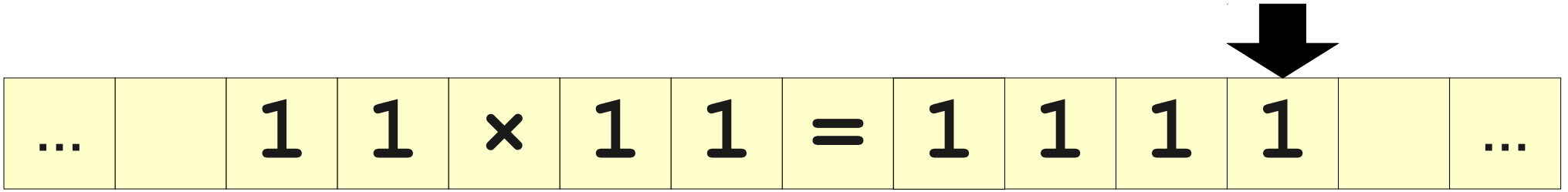
Checking for $1^*\times1^*=1^*$



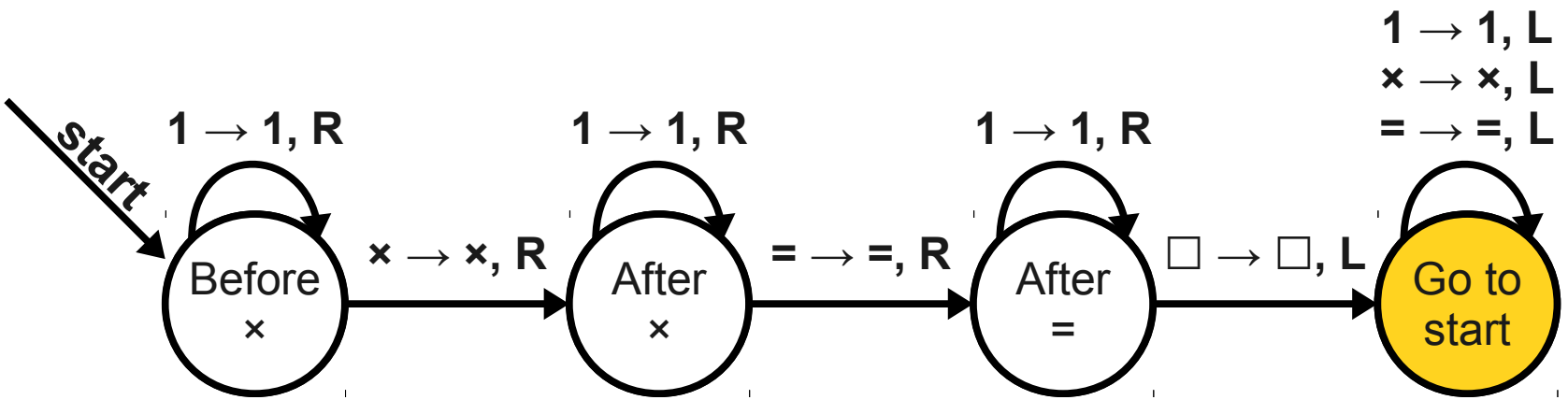


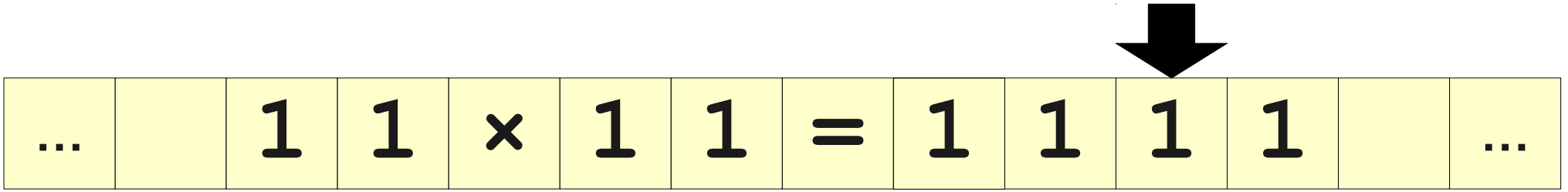
Checking for $1^*\times1^*=1^*$



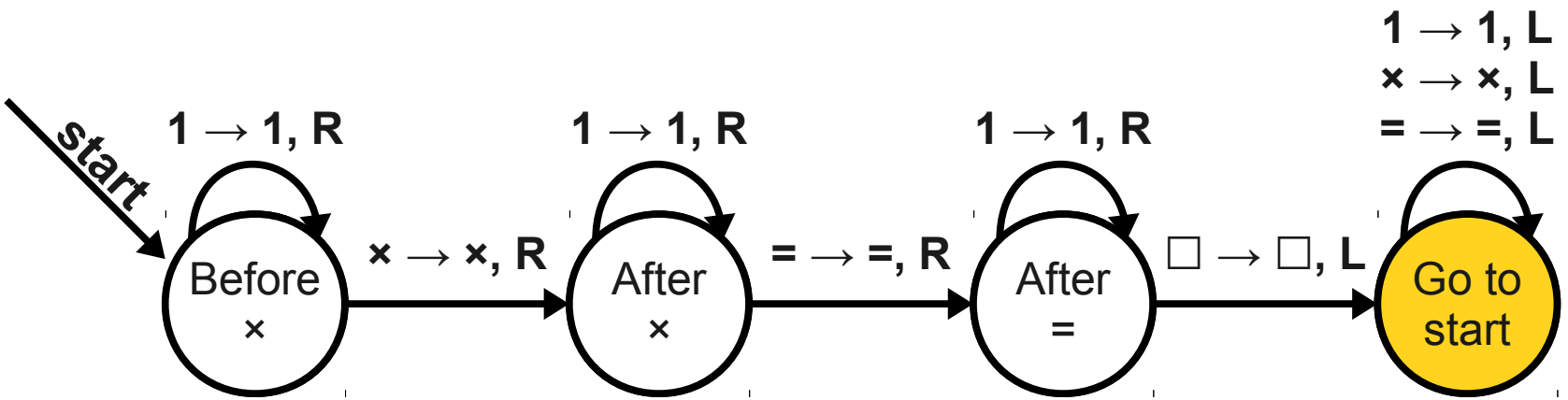


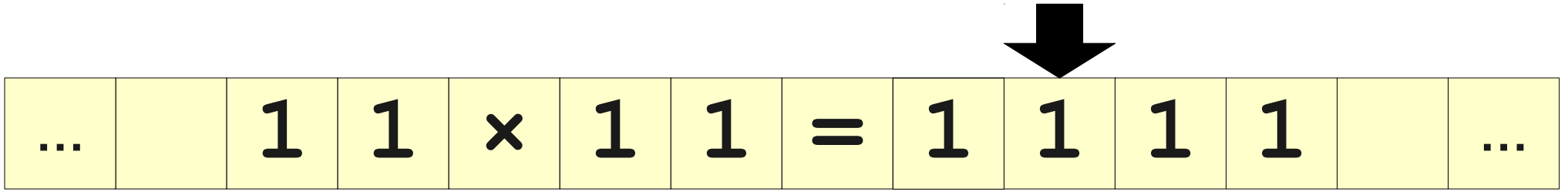
Checking for $1^*\times1^*=1^*$



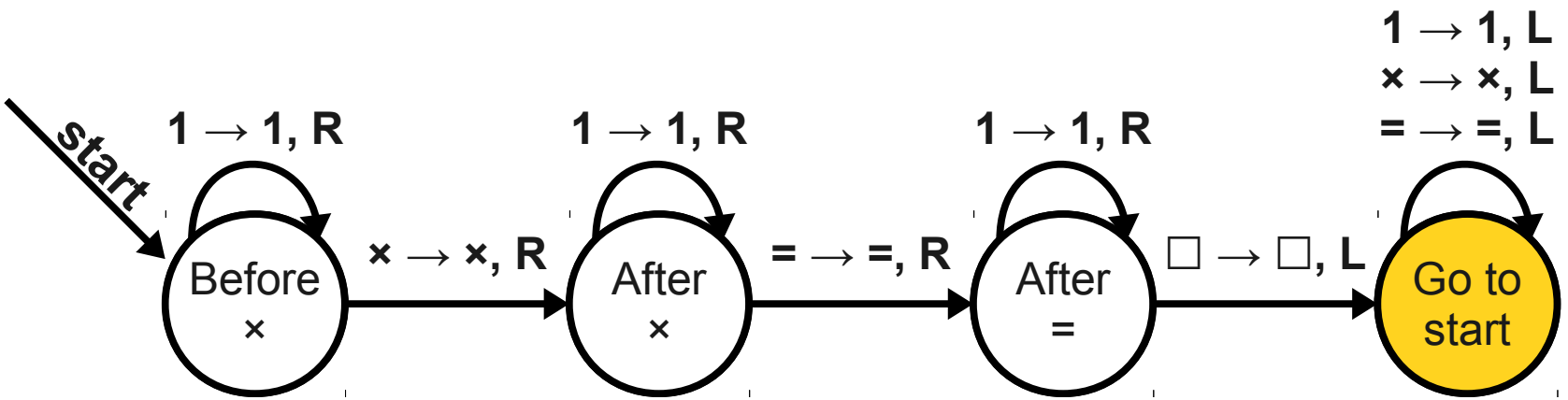


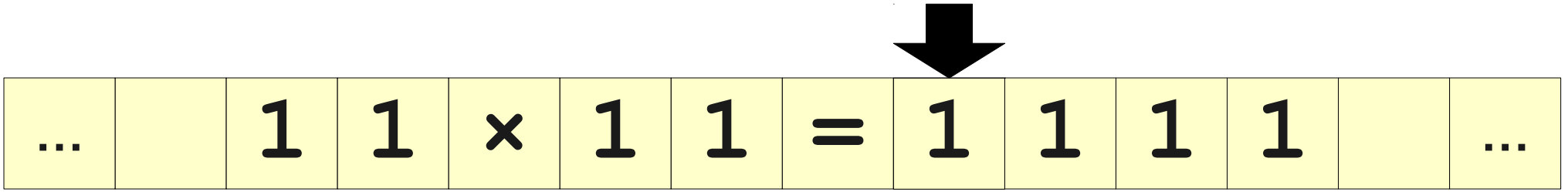
Checking for $1^*\times1^*=1^*$



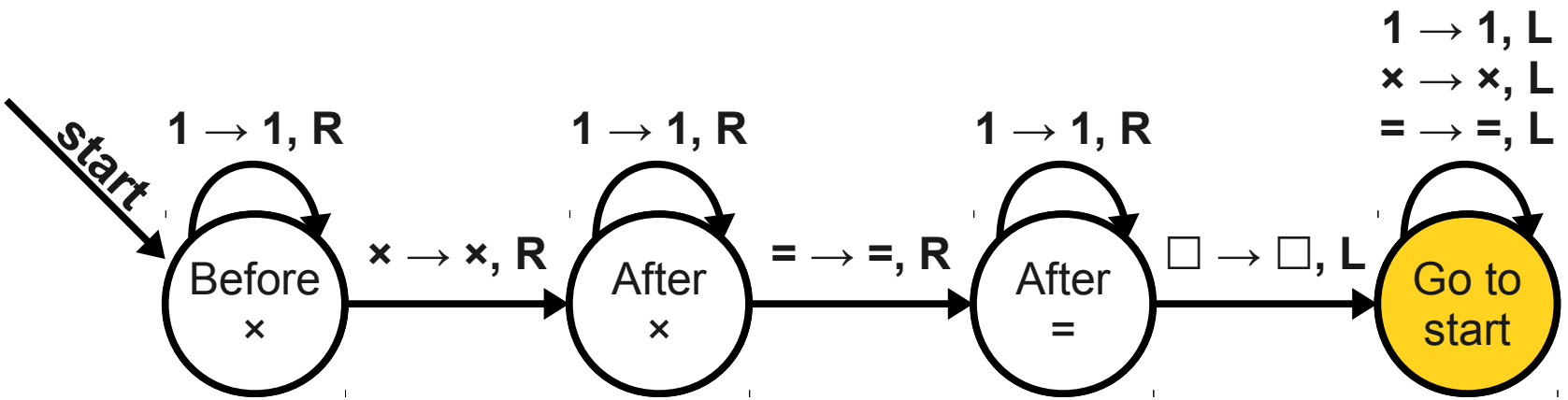


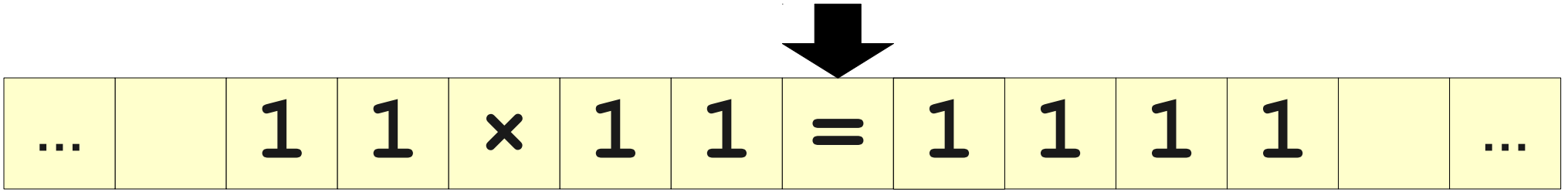
Checking for $1^*\times 1^*=1^*$



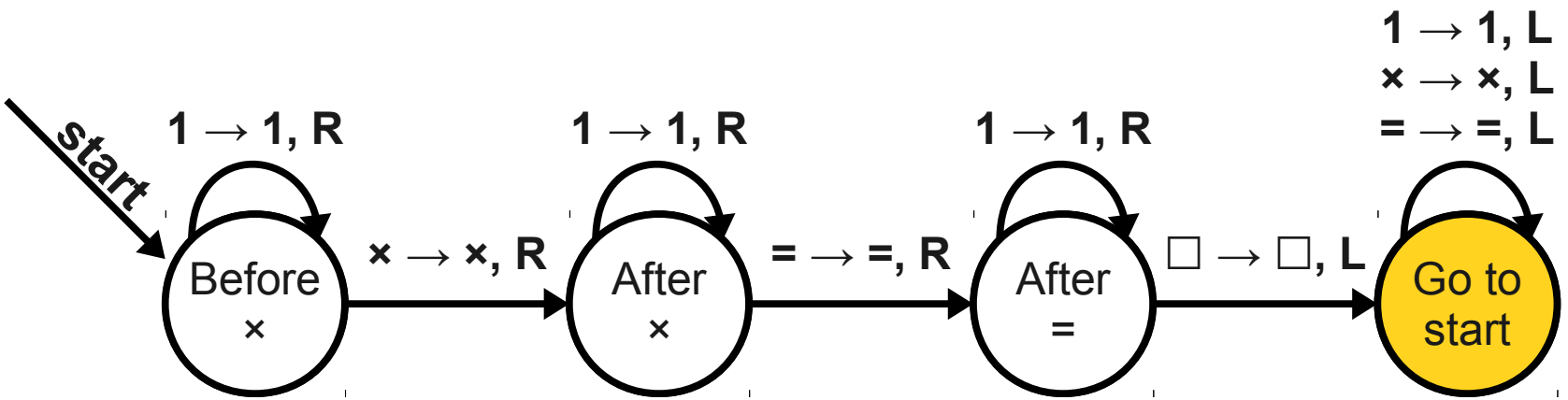


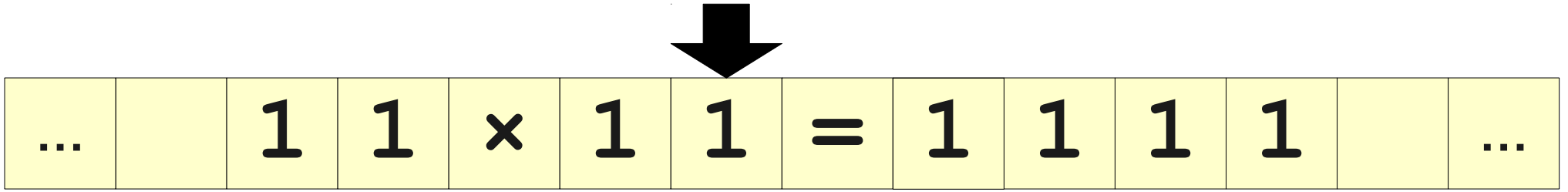
Checking for $1^*\times 1^*=1^*$



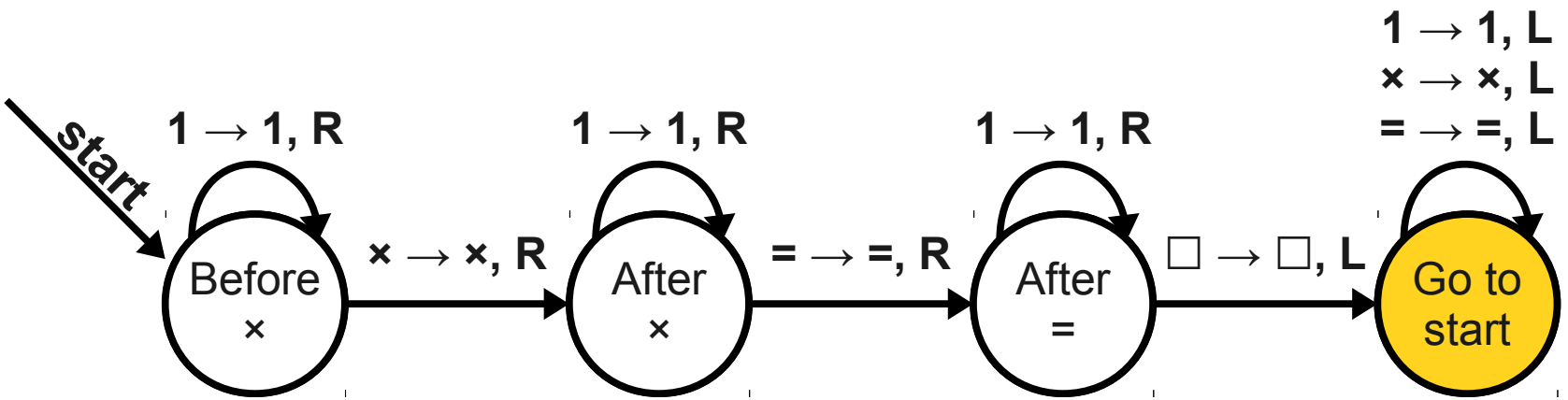


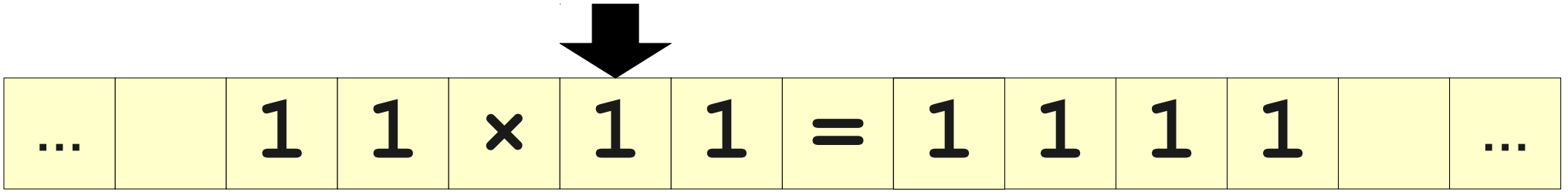
Checking for $1^*\times1^*=1^*$



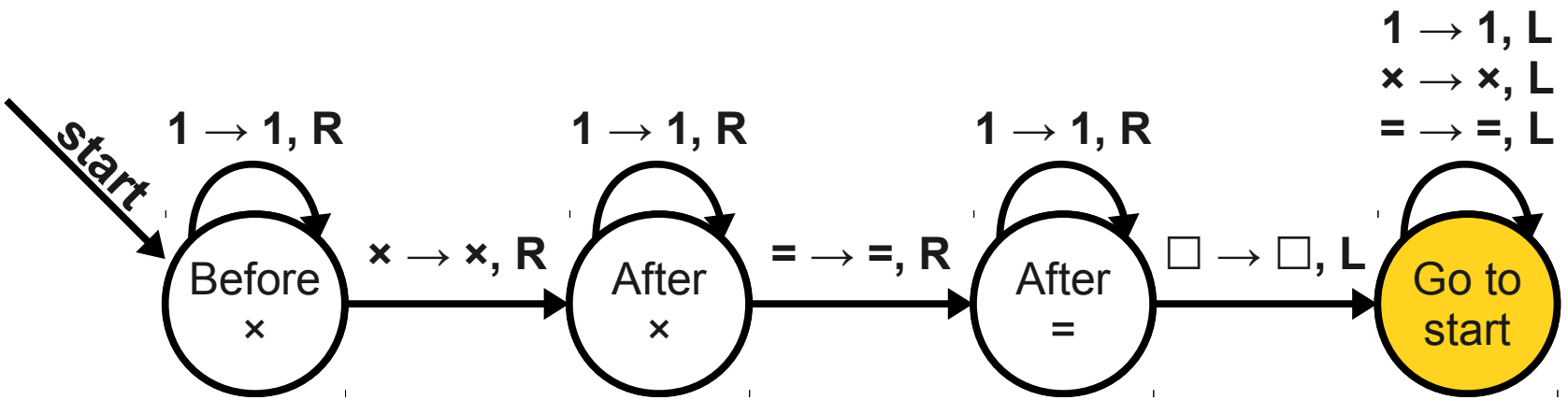


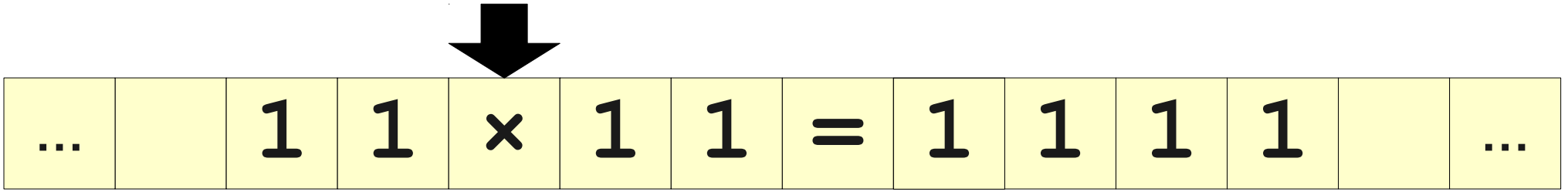
Checking for $1^*\times1^*=1^*$



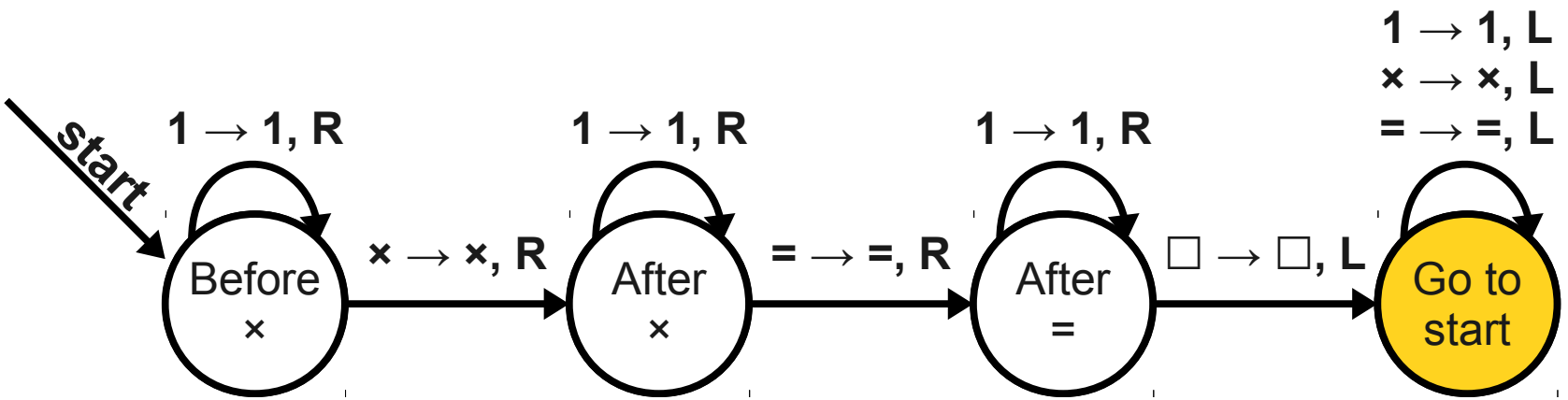


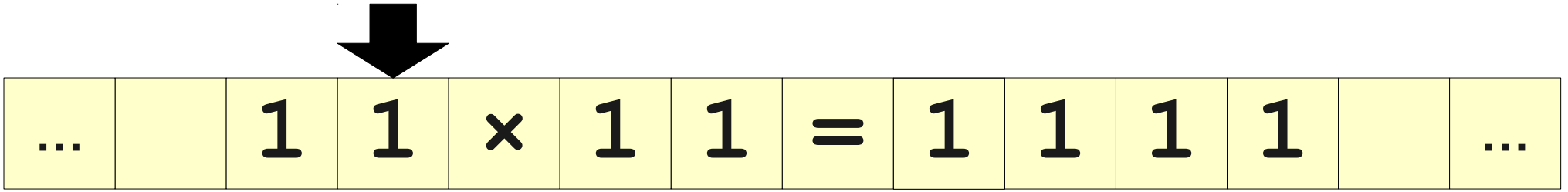
Checking for $1^*\times1^*=1^*$



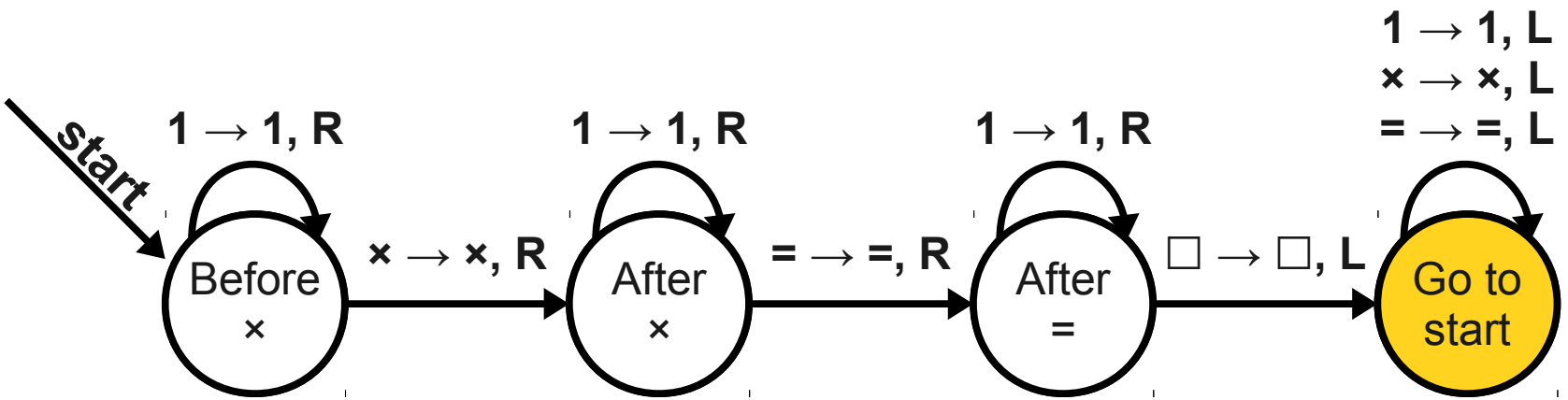


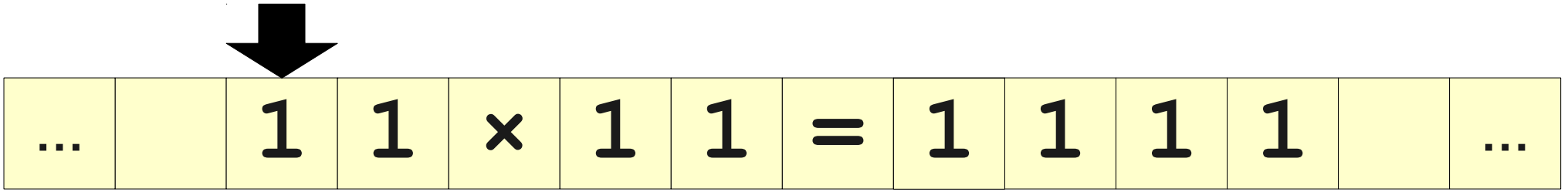
Checking for $1^*\times 1^*=1^*$



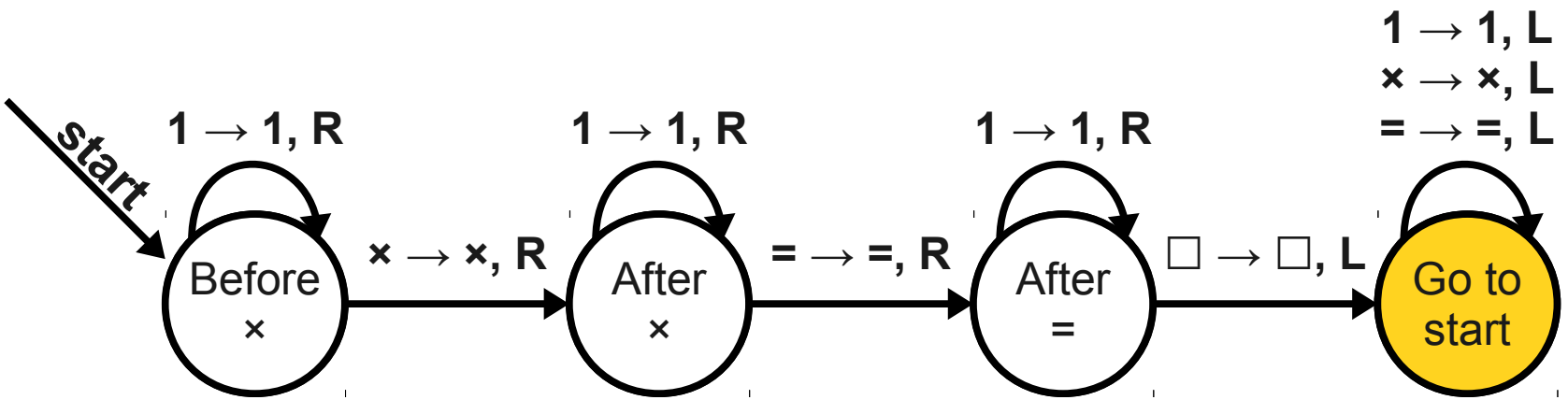


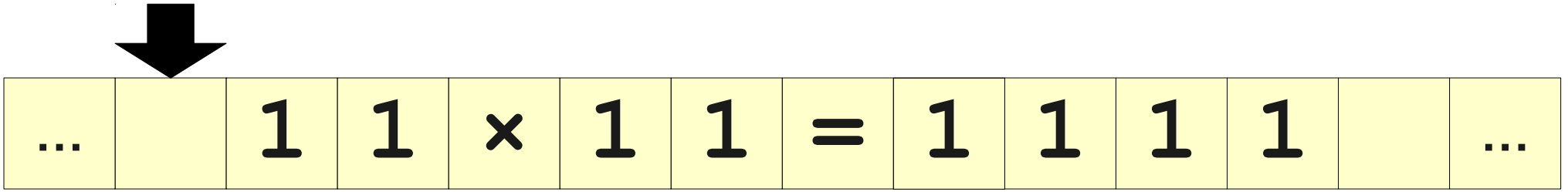
Checking for $1^*\times1^*=1^*$



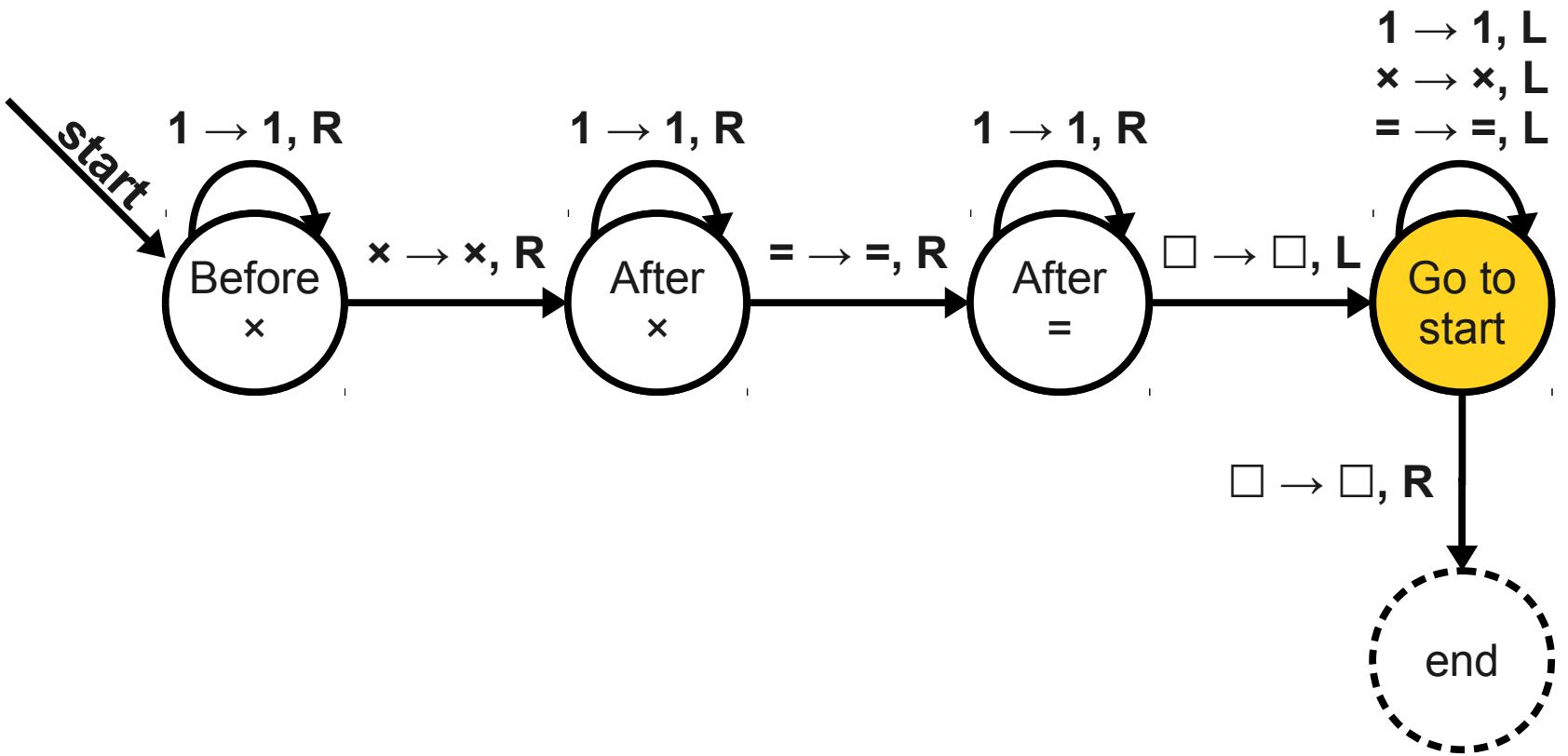


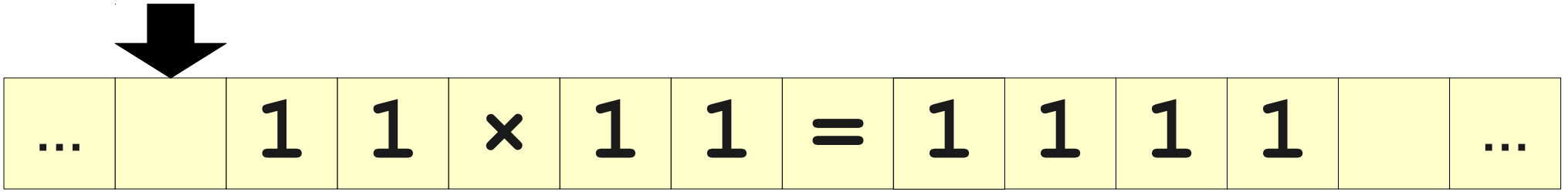
Checking for $1^*\times1^*=1^*$



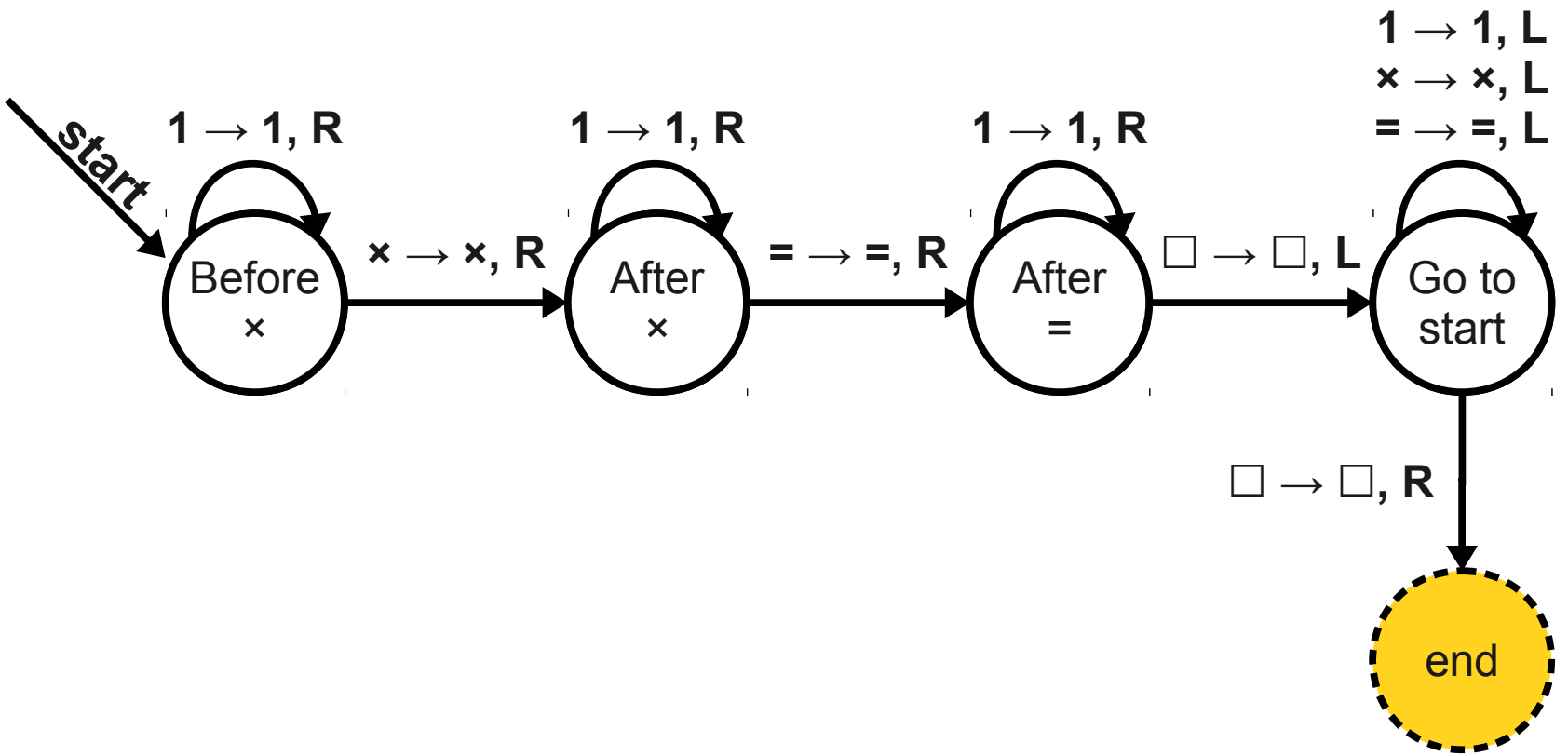


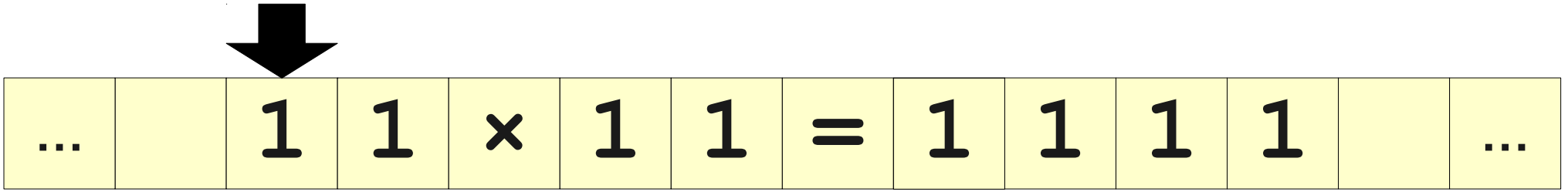
Checking for $1^*\times 1^*=1^*$



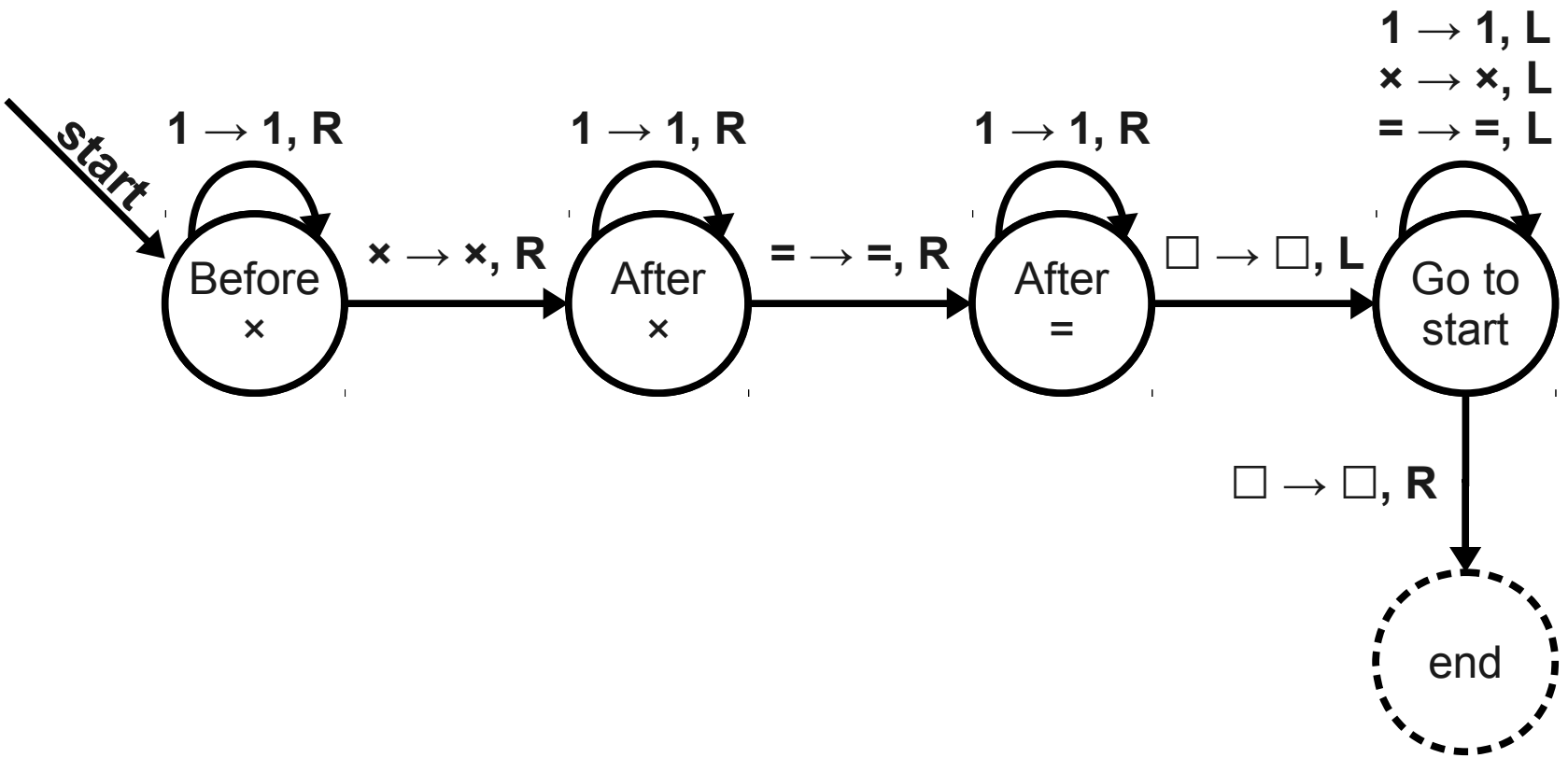


Checking for $1^*\times1^*=1^*$

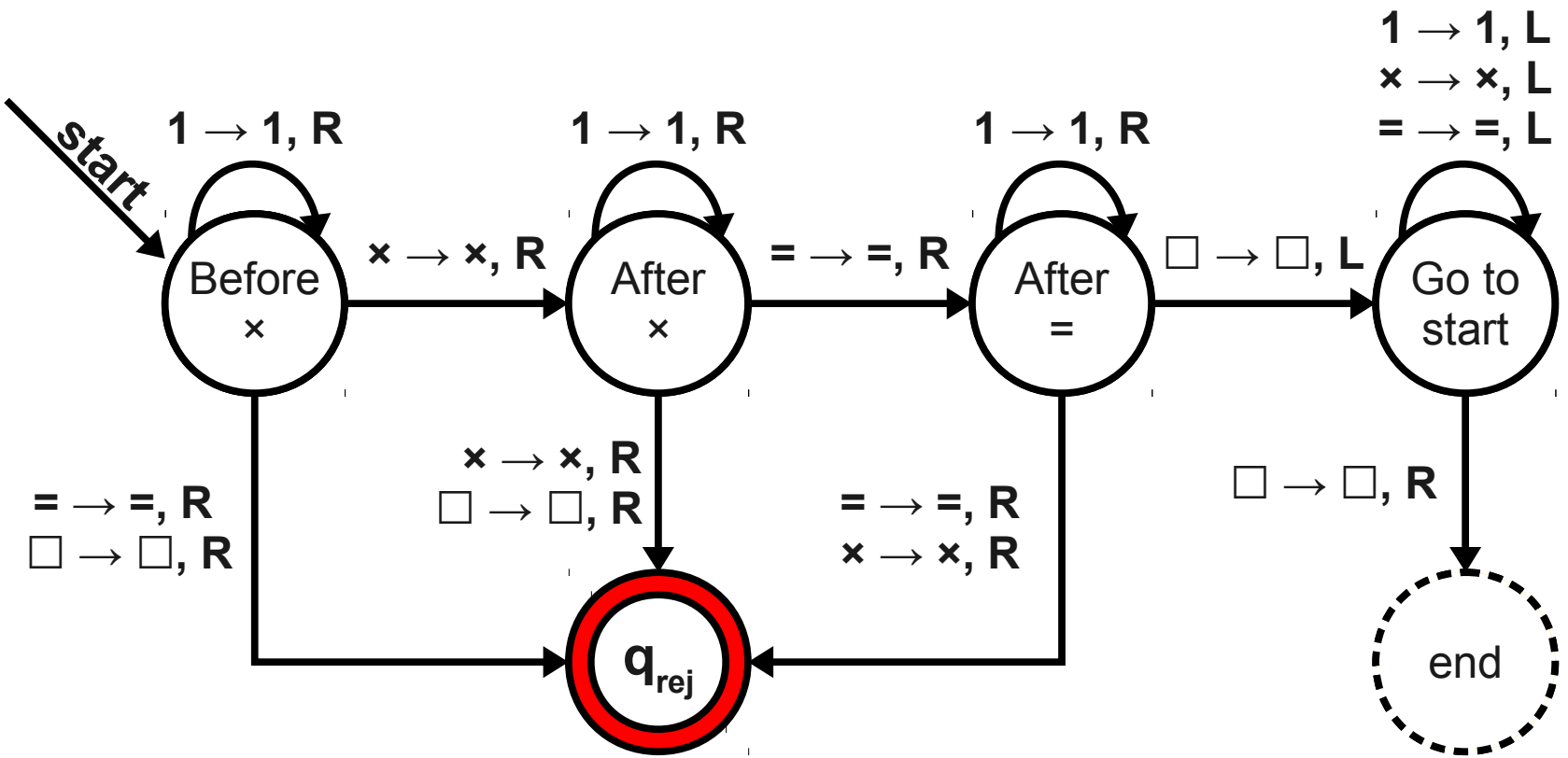




Checking for $1^*\times1^*=1^*$



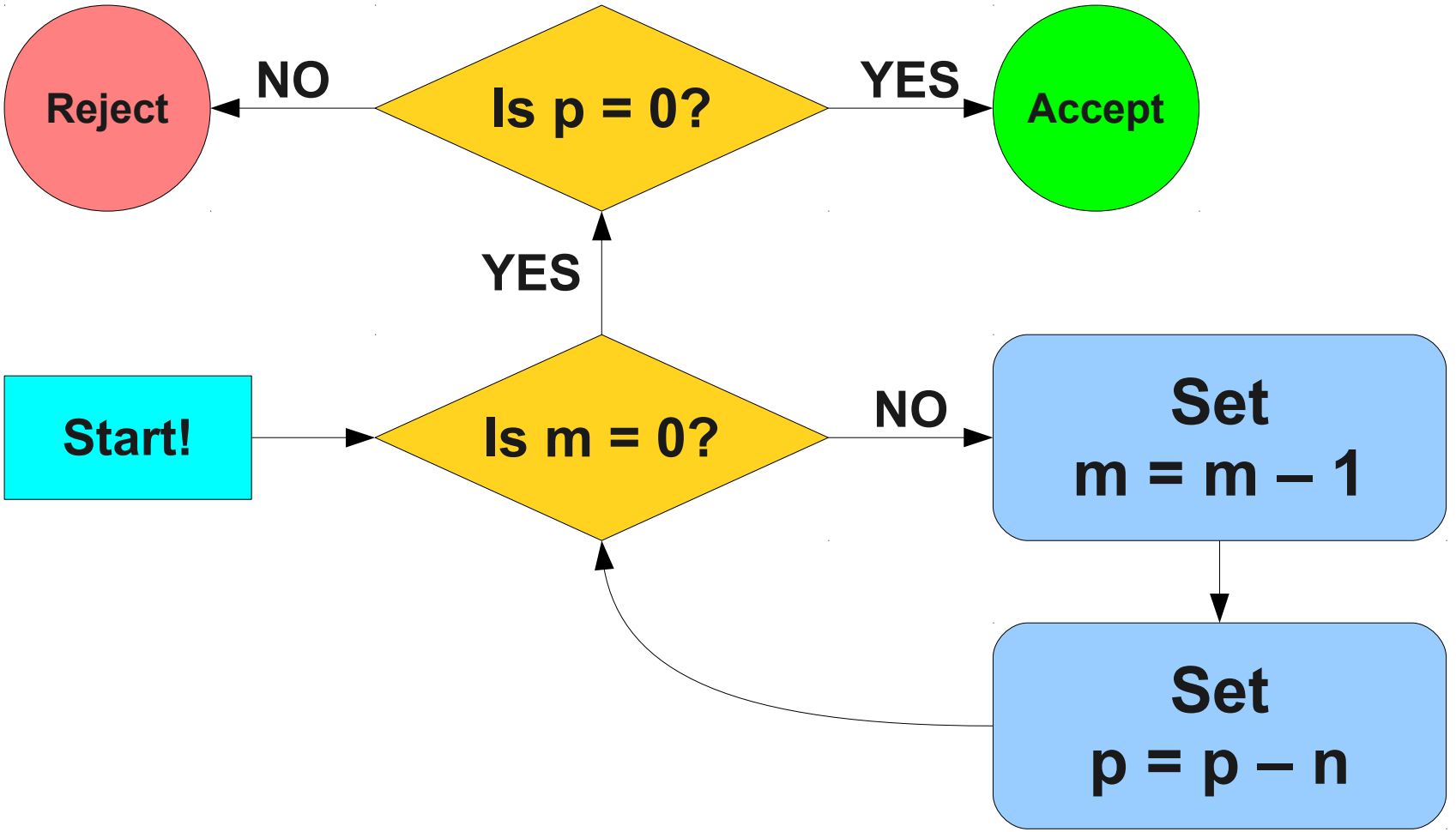
Checking for $1^*\times1^*=1^*$



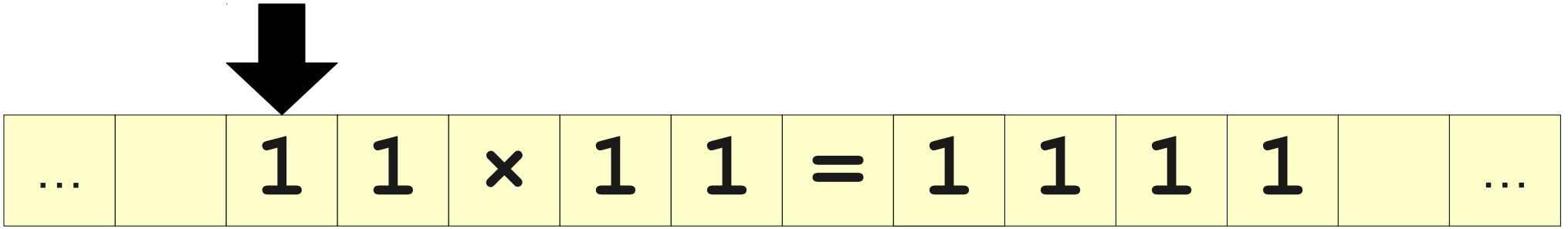
Performing Multiplication

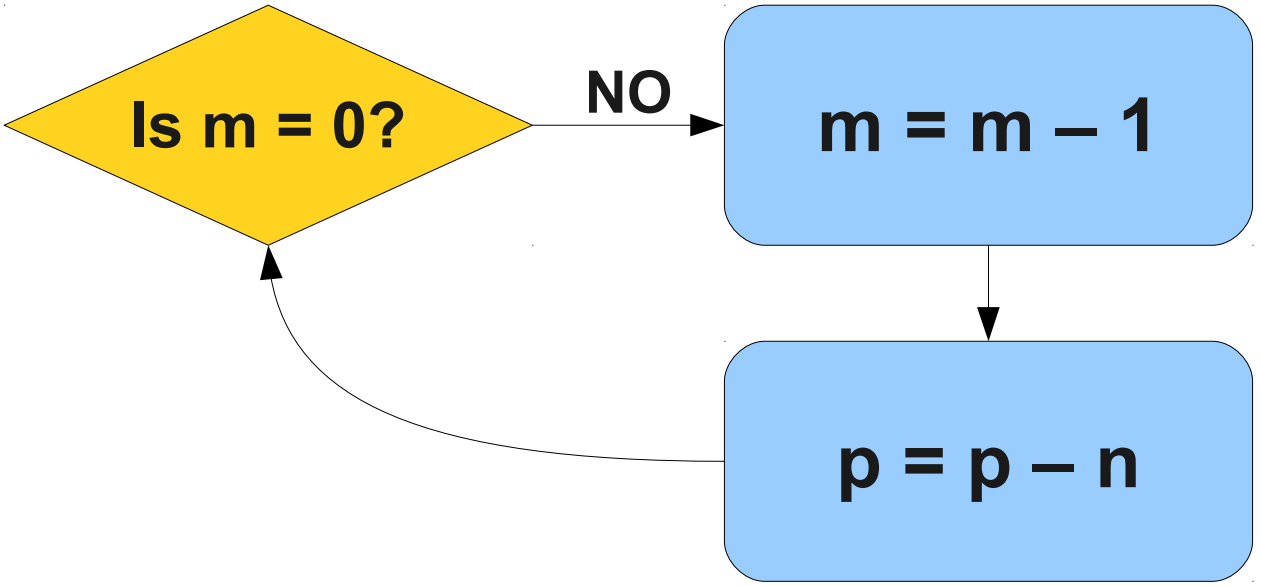
- How would you check that $m \times n = p$?
- **Idea:** Use a recursive/inductive approach to multiplication:
  - $0 \times n = p$ iff $p = 0$
  - $(m + 1) \times n = p$ iff $m \times n = p - n$
- To check the multiplication, we can keep subtracting one from m and subtracting n from p until m is zero. We can then check at that time if p is zero.

Checking if $m \times n = p$

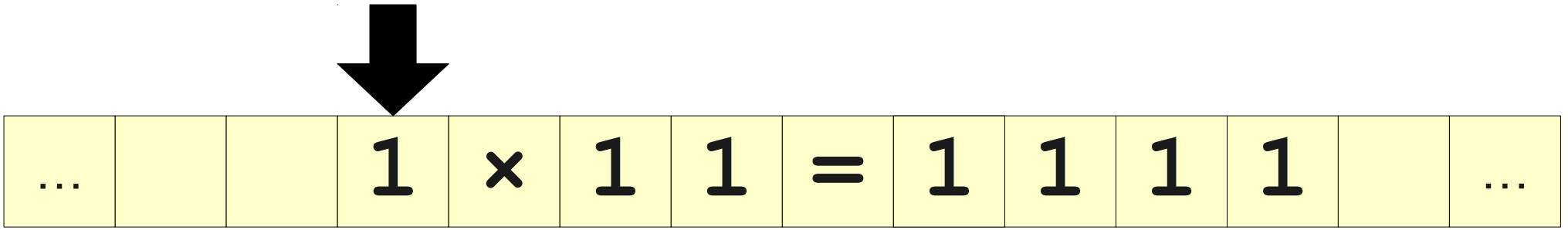


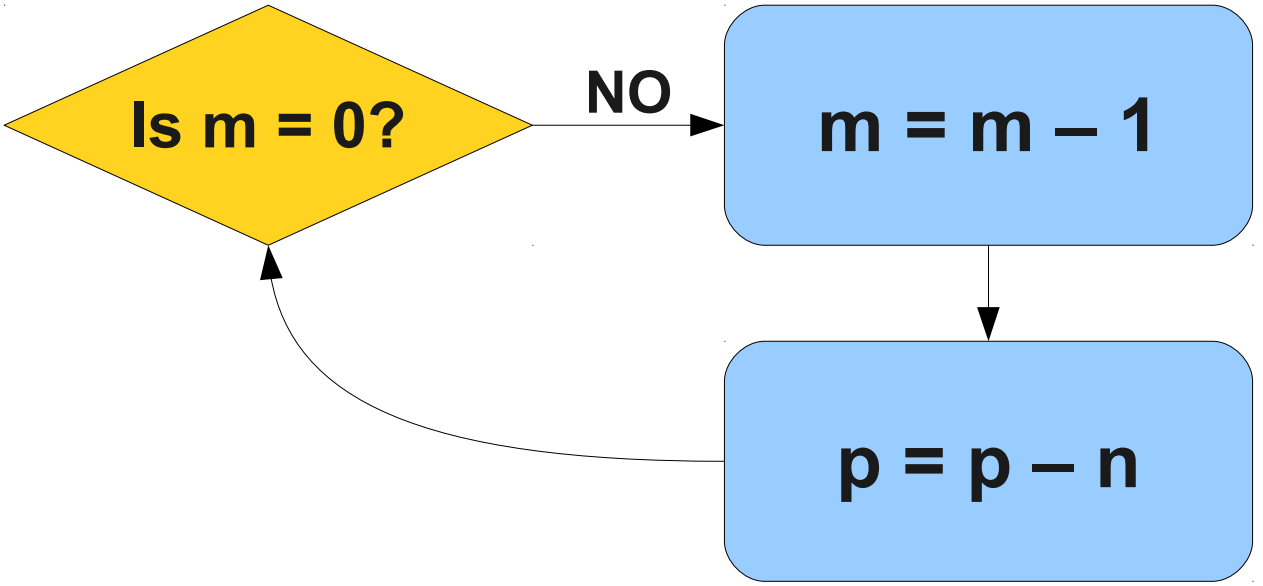
A Sketch of the Algorithm



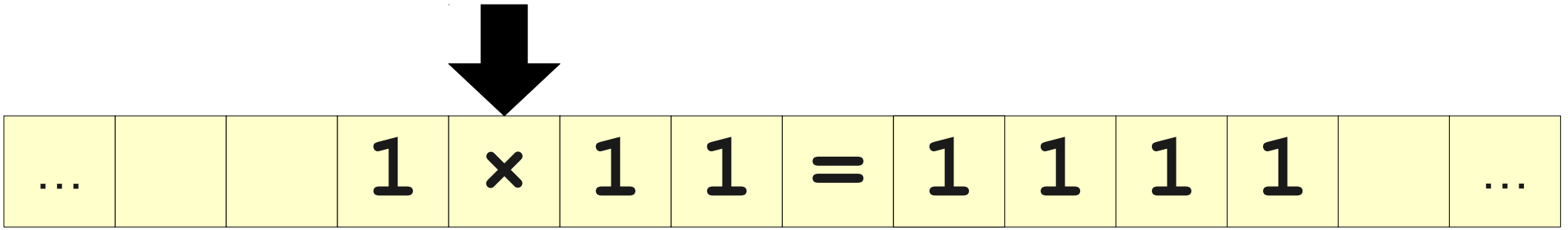


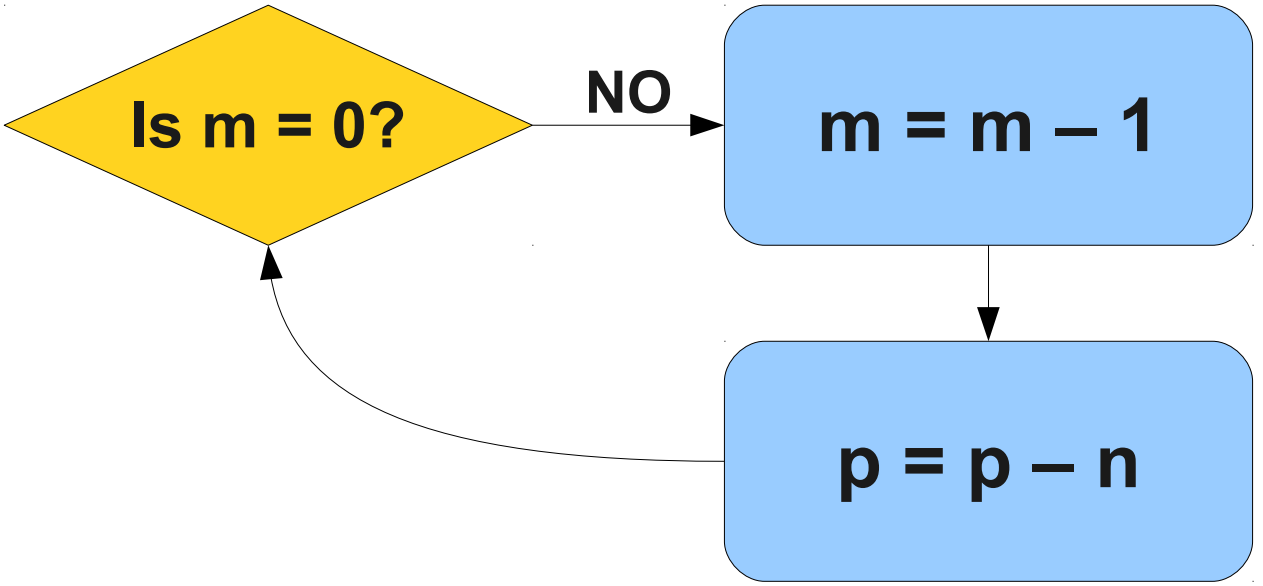
A Sketch of the Algorithm



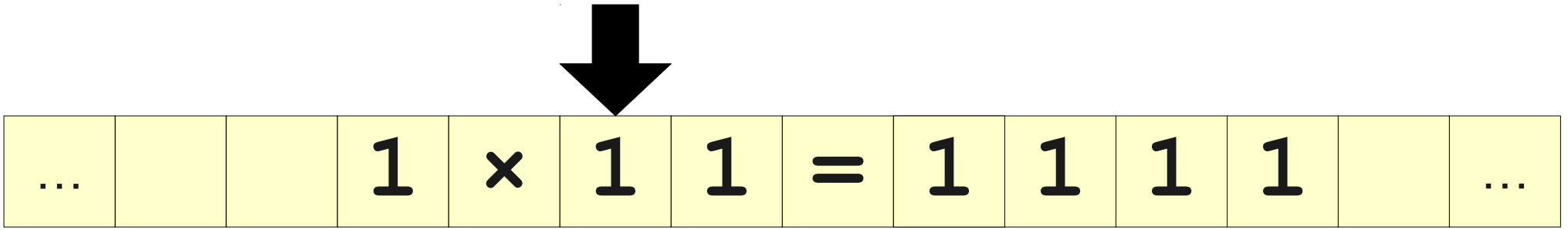


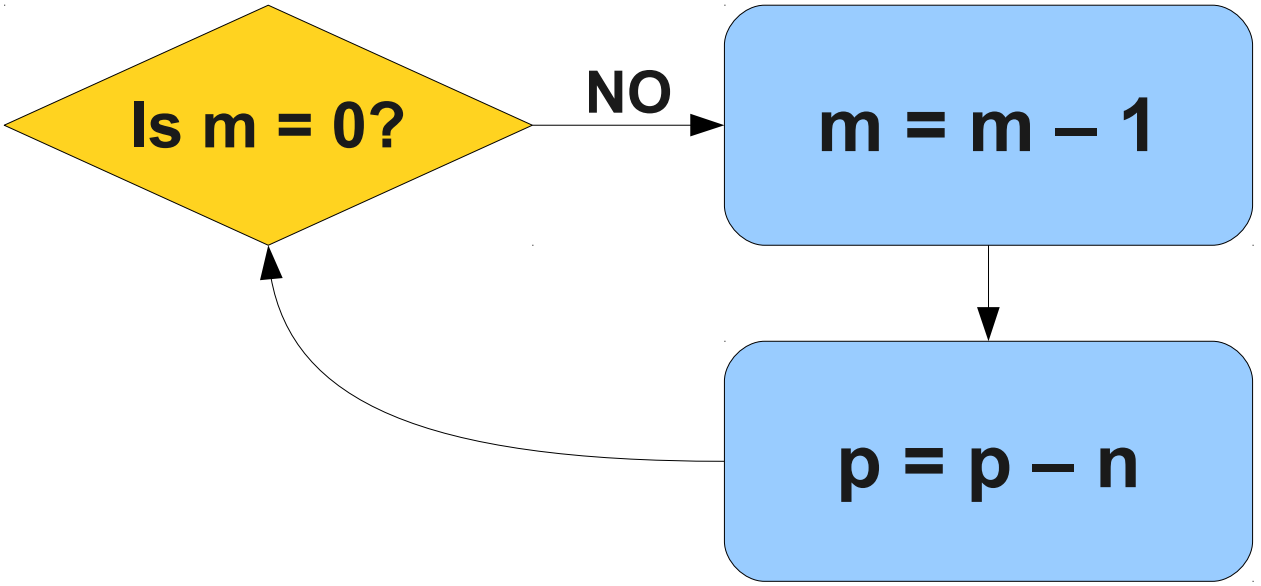
A Sketch of the Algorithm



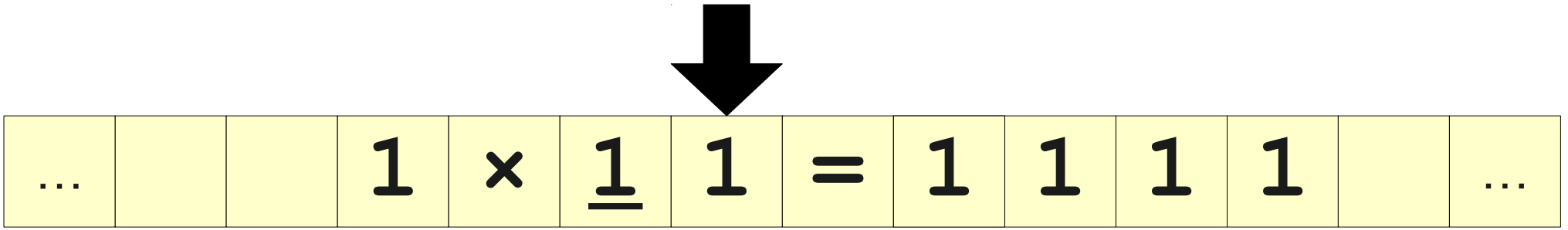


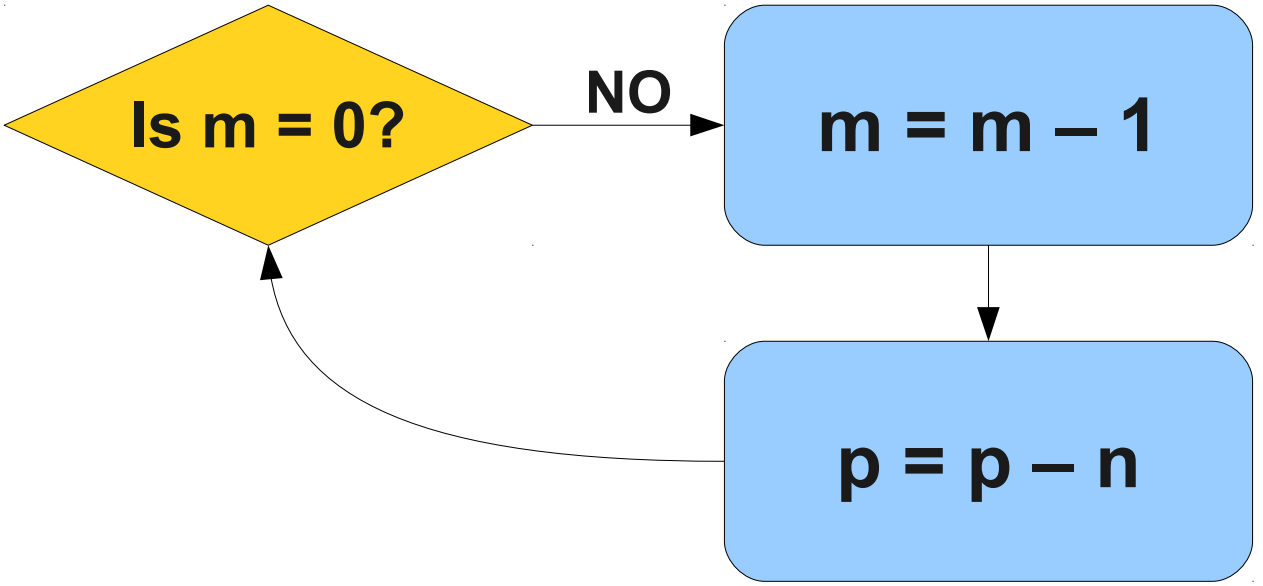
A Sketch of the Algorithm



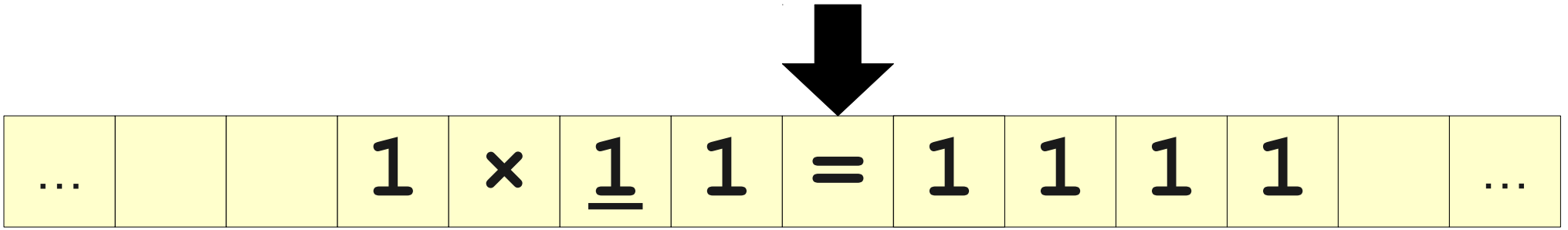


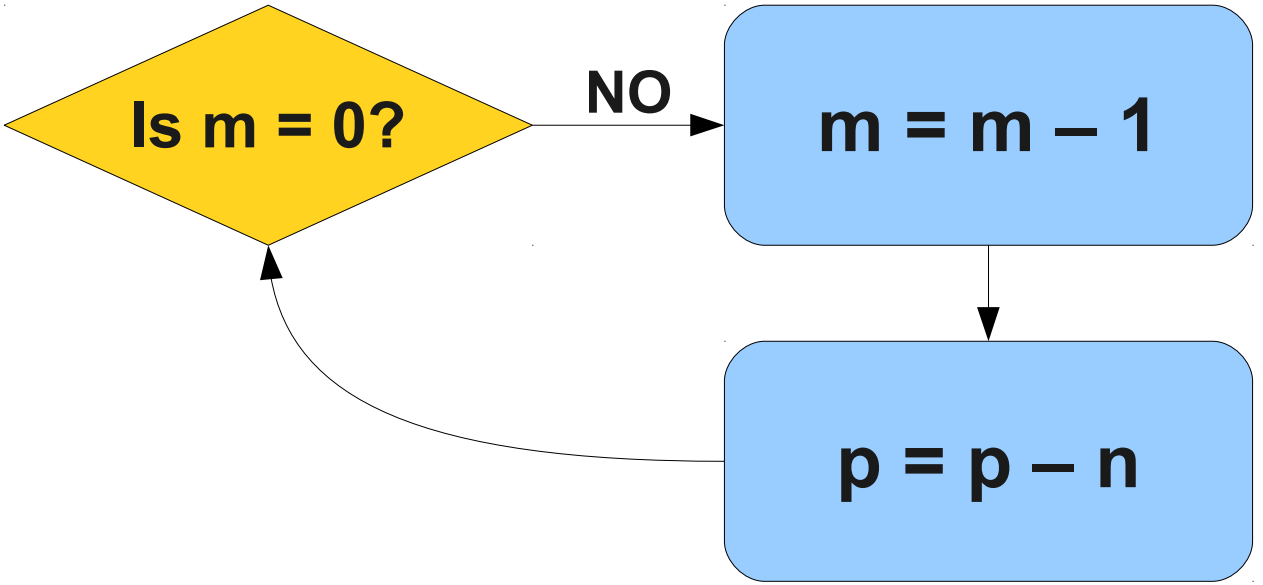
A Sketch of the Algorithm



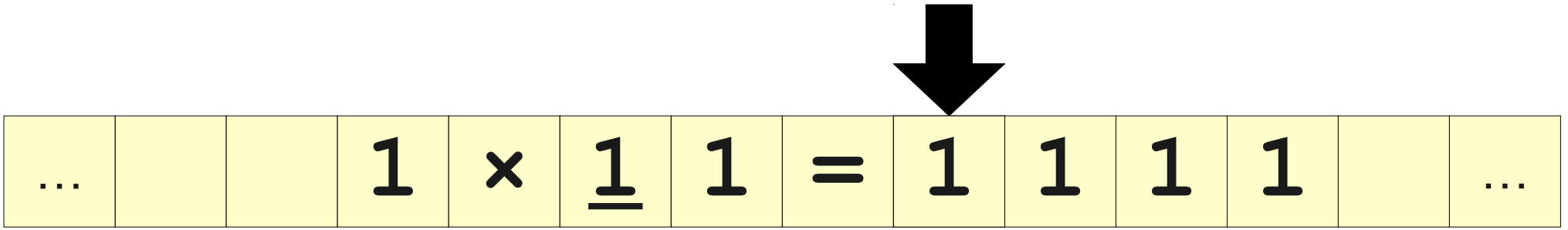


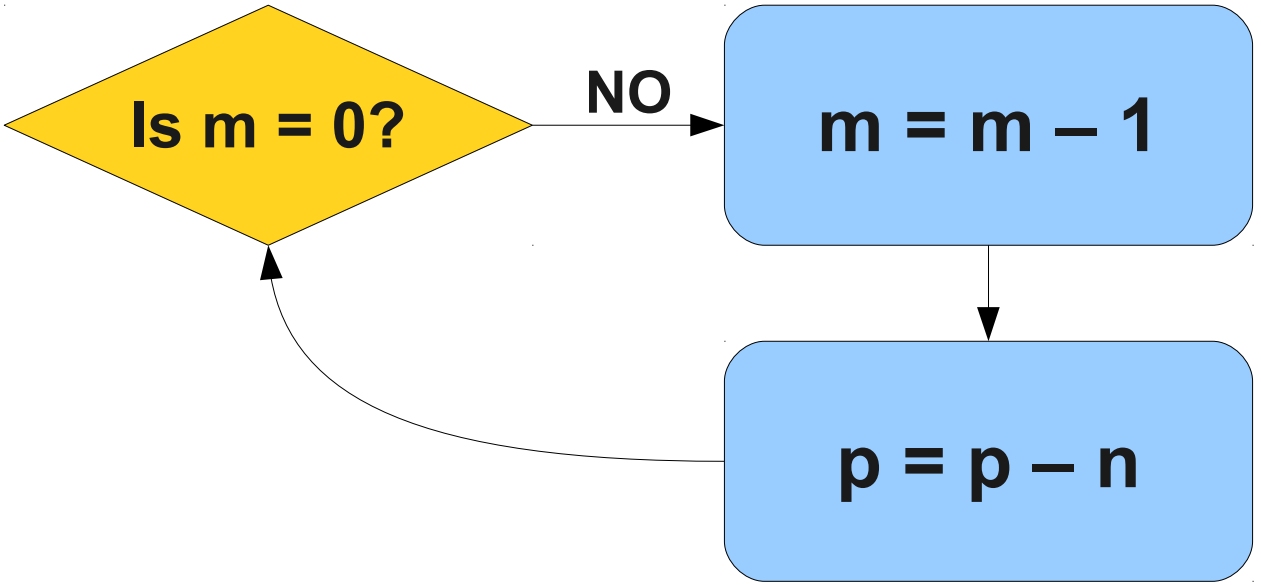
A Sketch of the Algorithm



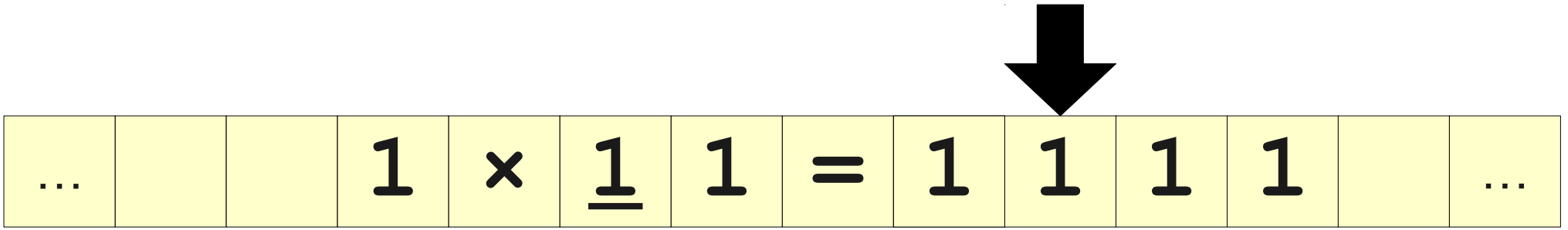


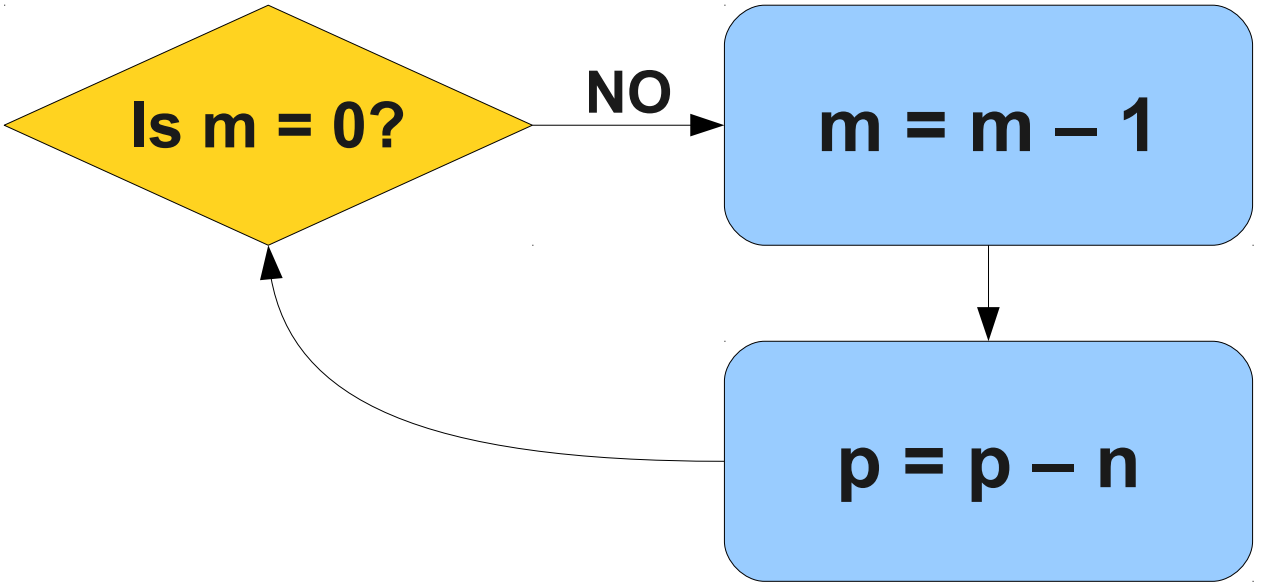
A Sketch of the Algorithm



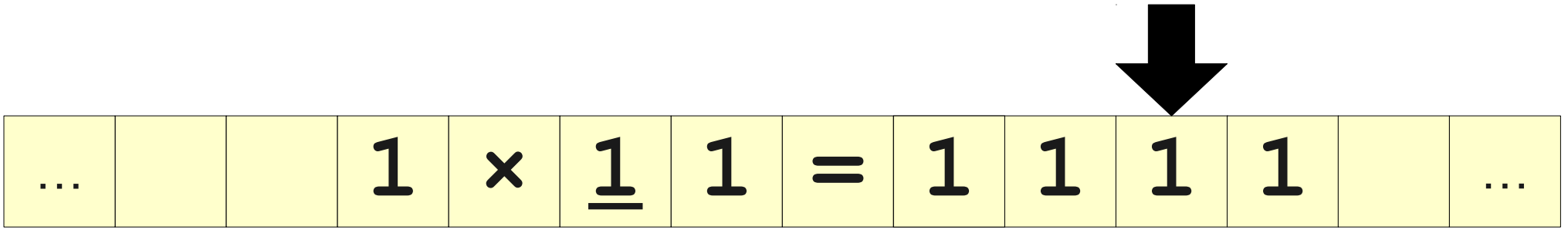


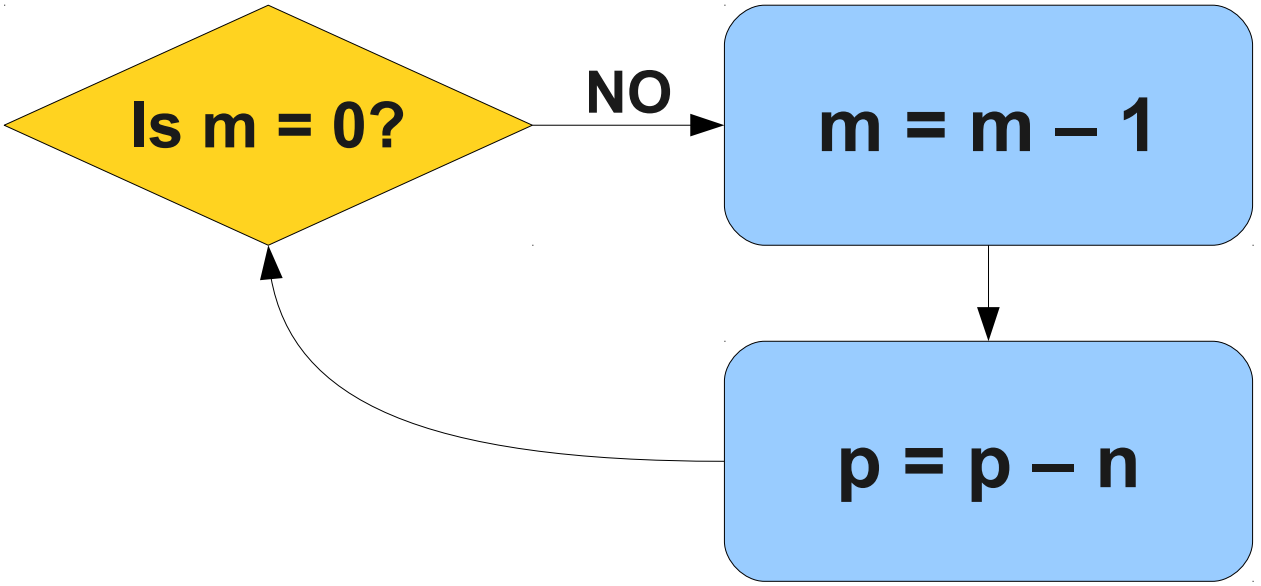
A Sketch of the Algorithm



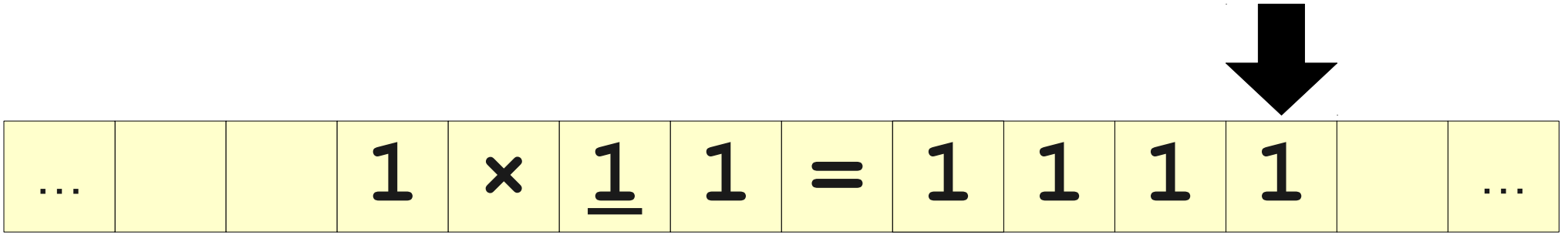


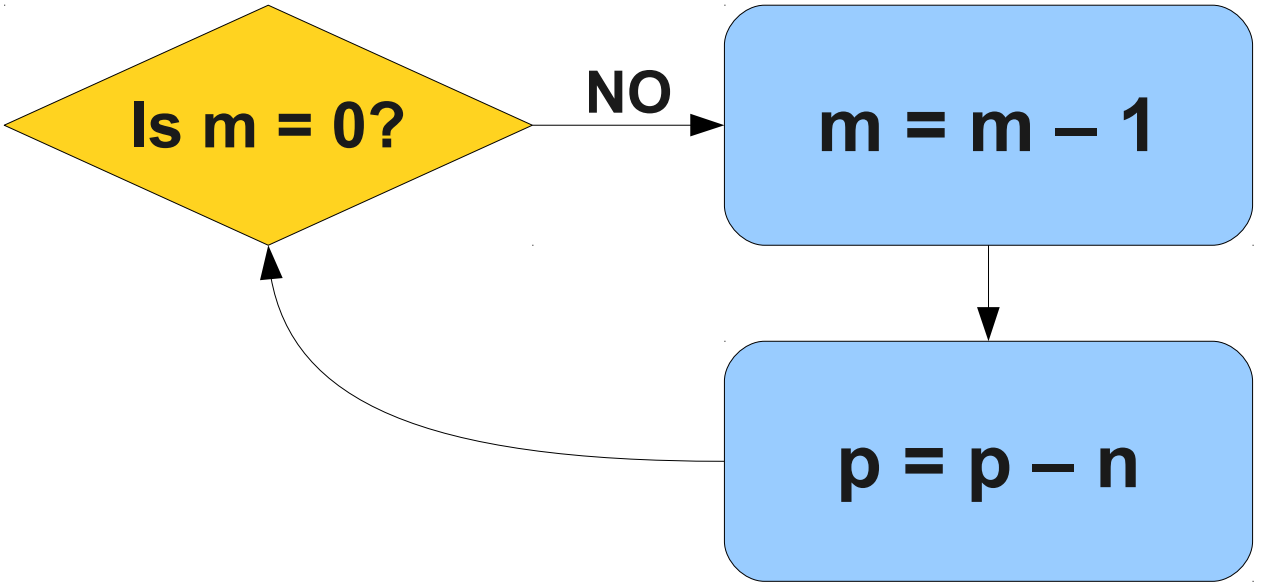
A Sketch of the Algorithm



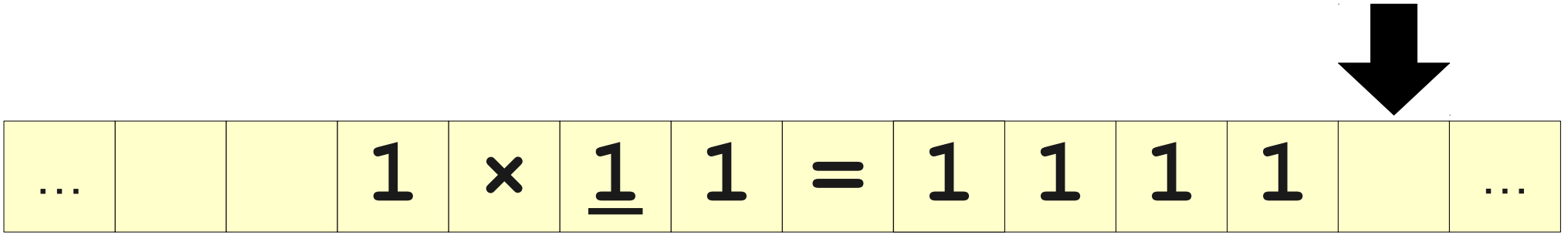


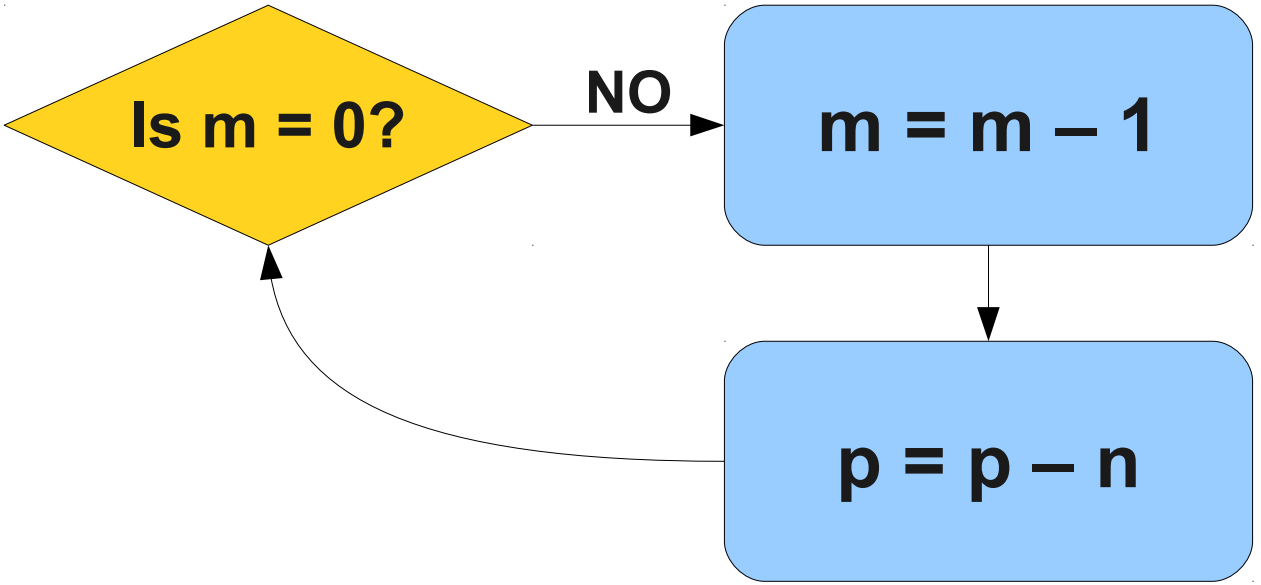
A Sketch of the Algorithm



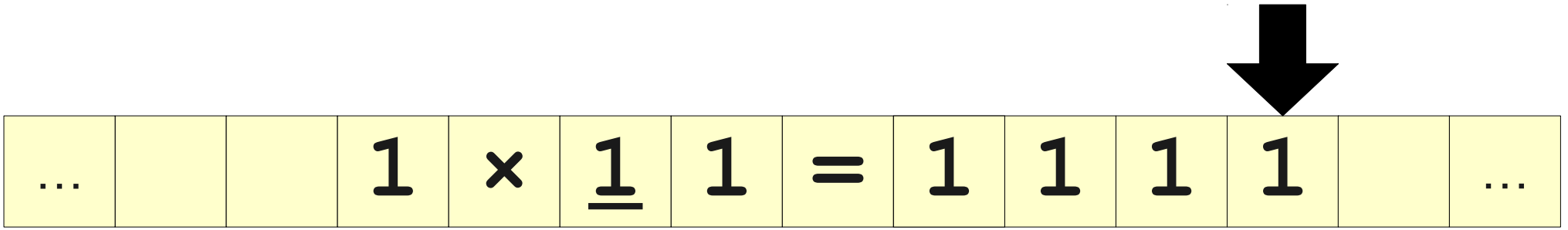


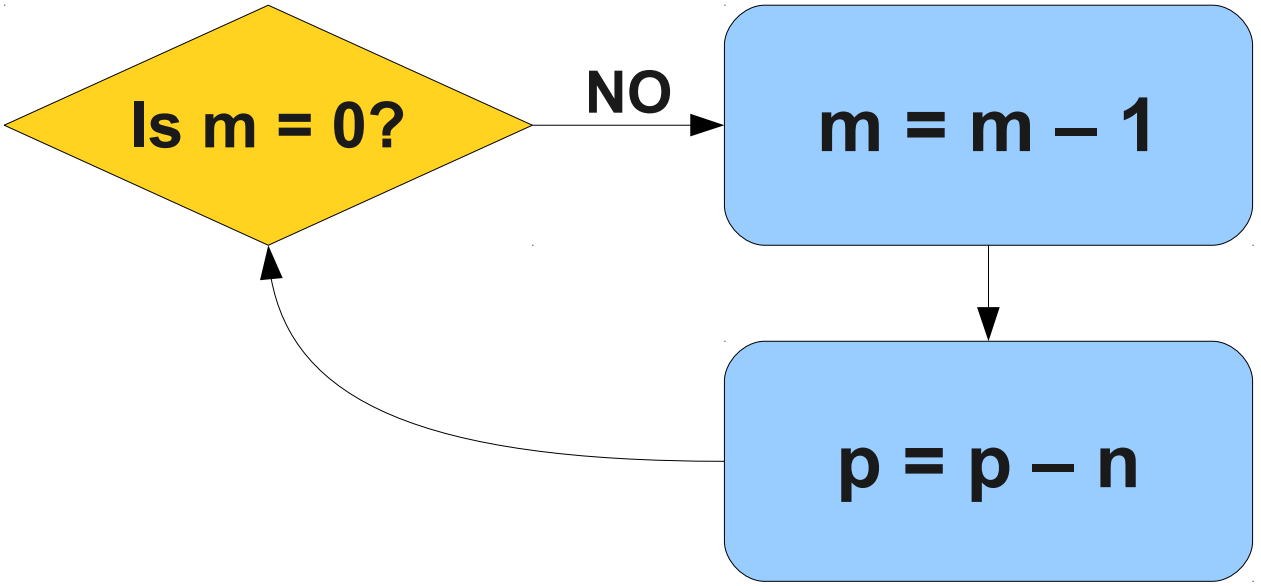
A Sketch of the Algorithm



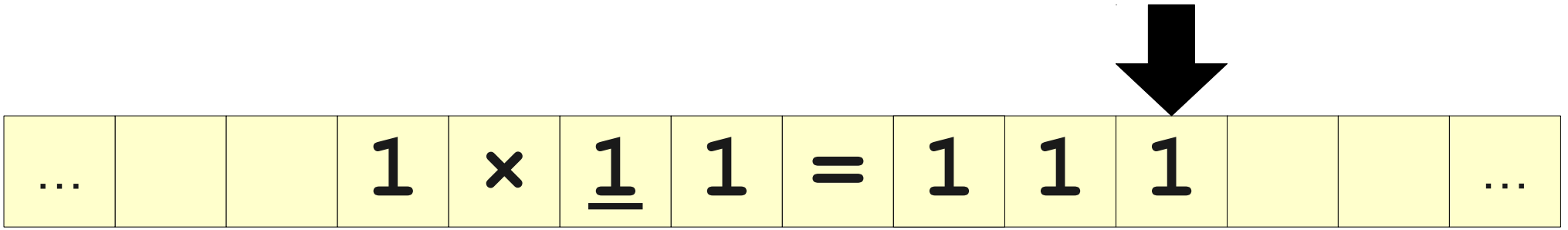


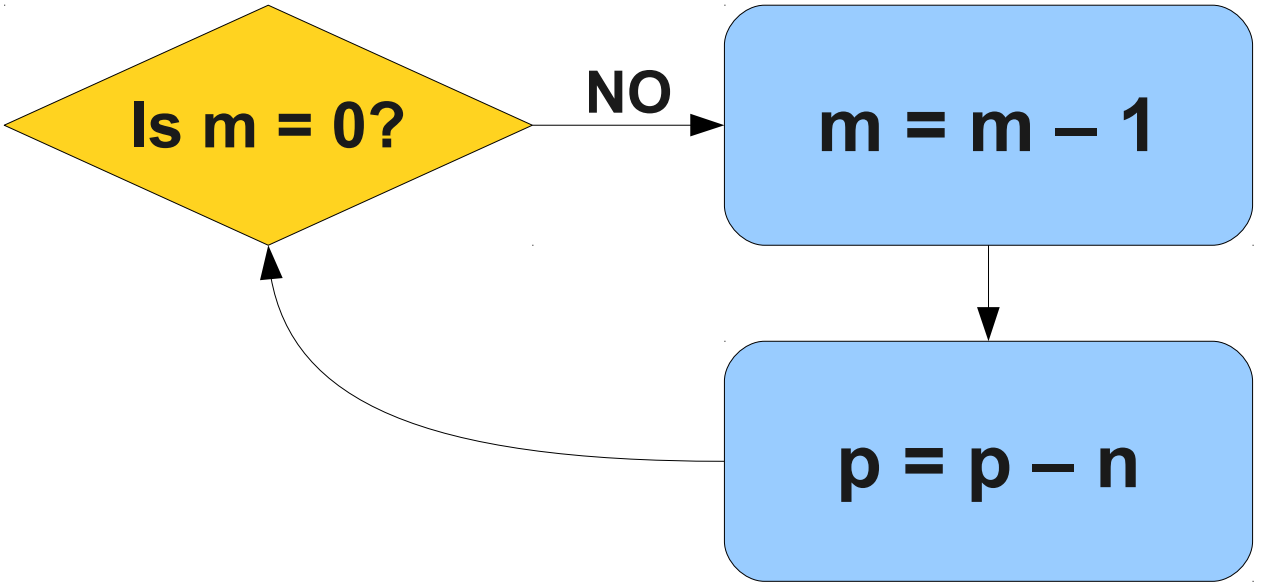
A Sketch of the Algorithm



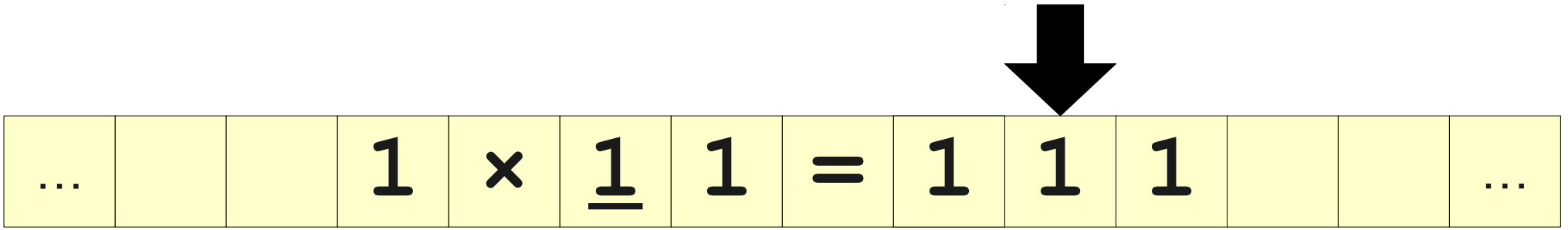


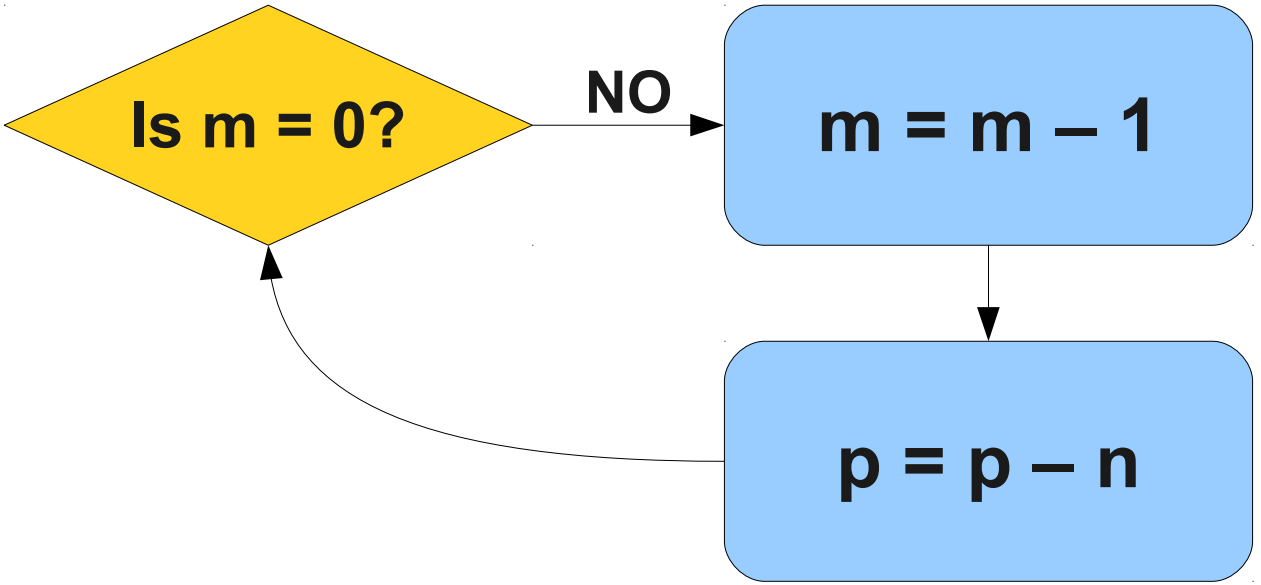
A Sketch of the Algorithm



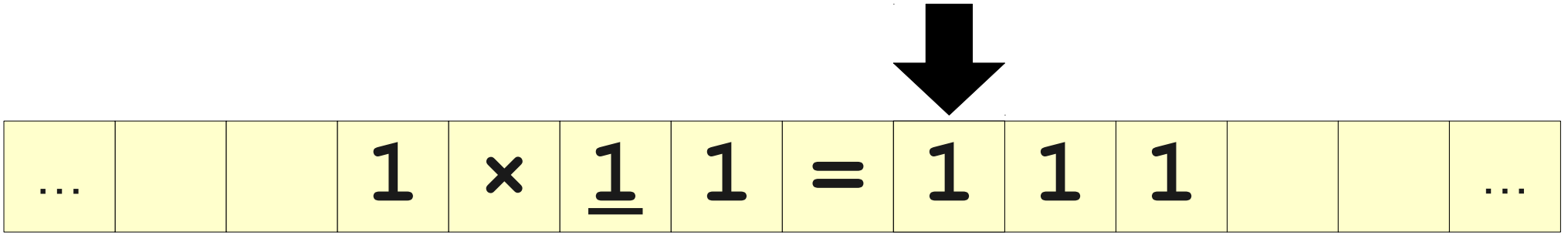


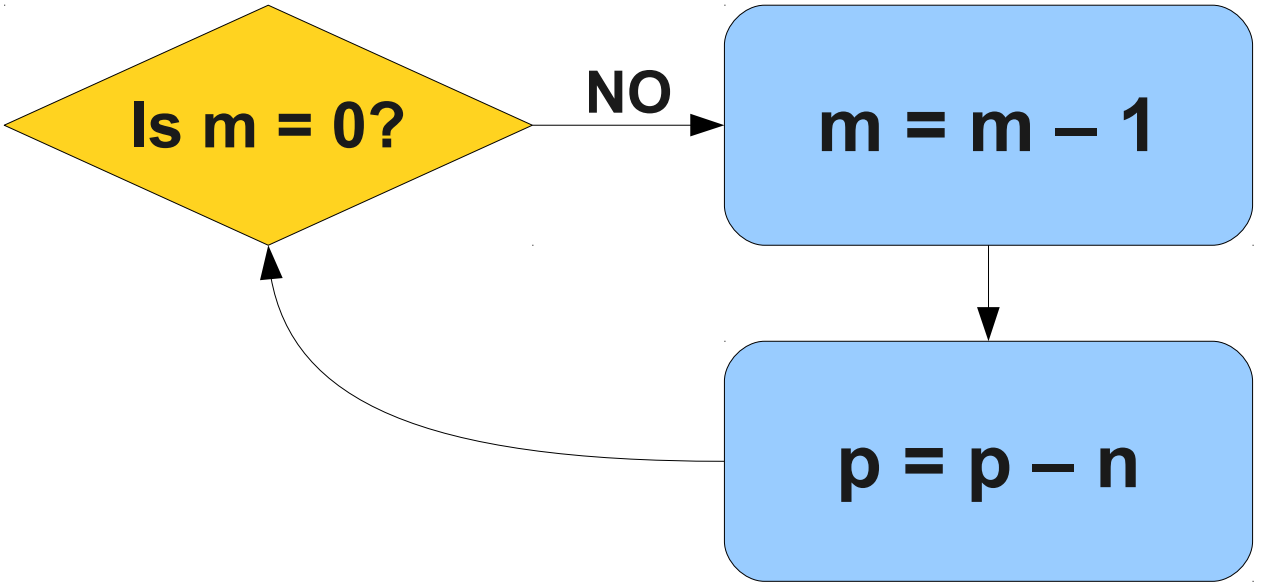
A Sketch of the Algorithm



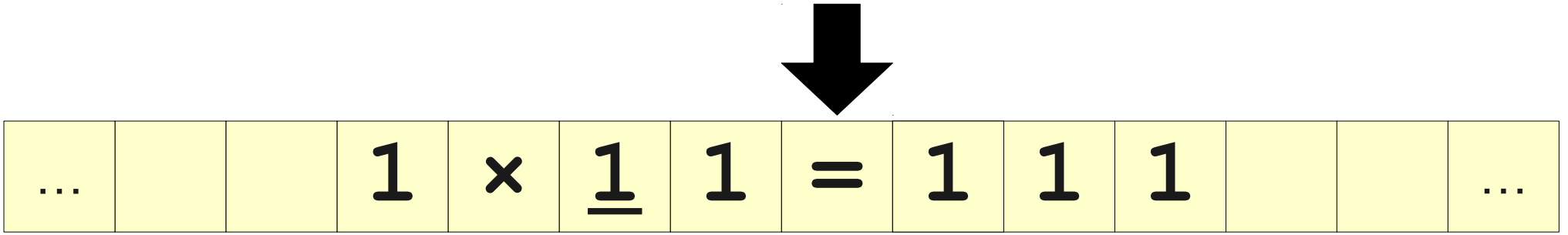


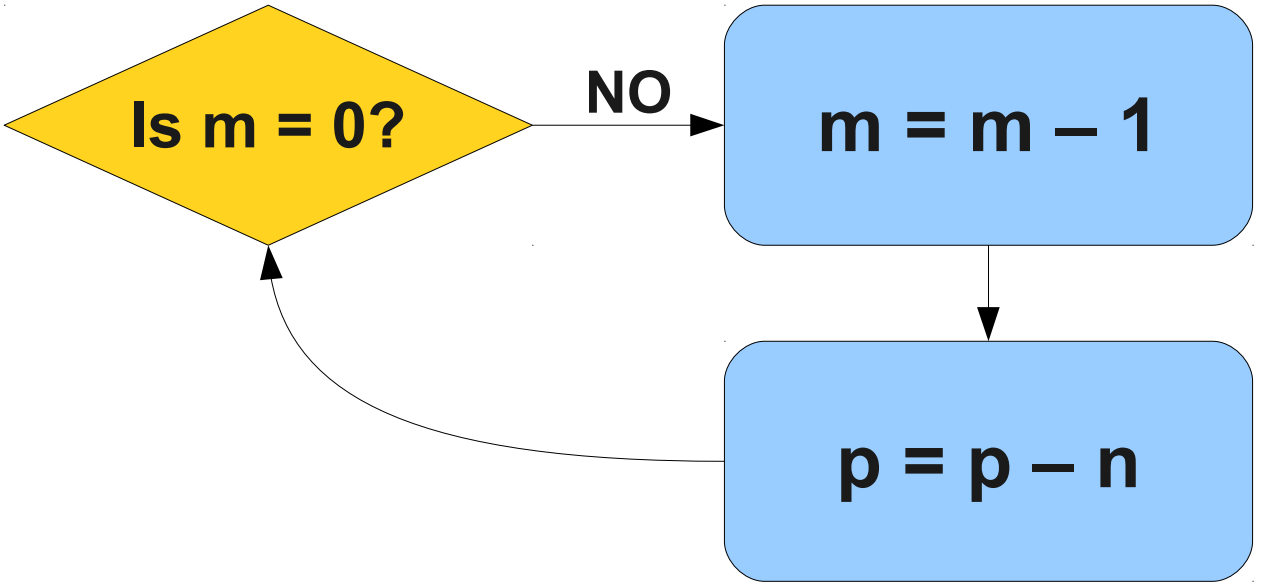
A Sketch of the Algorithm



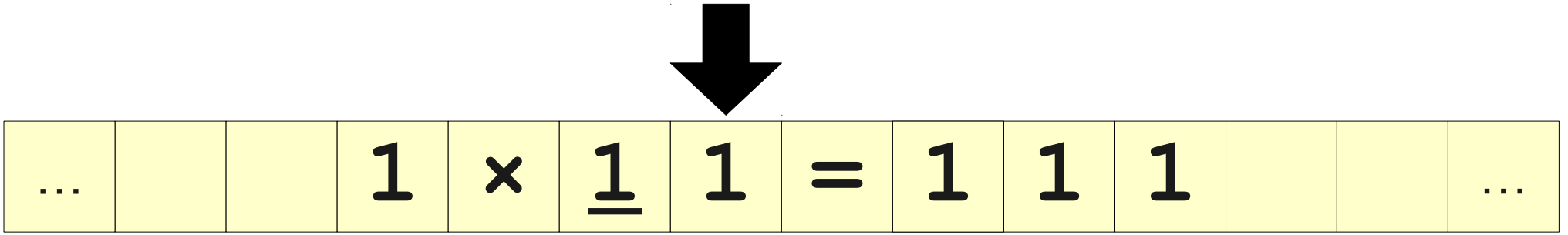


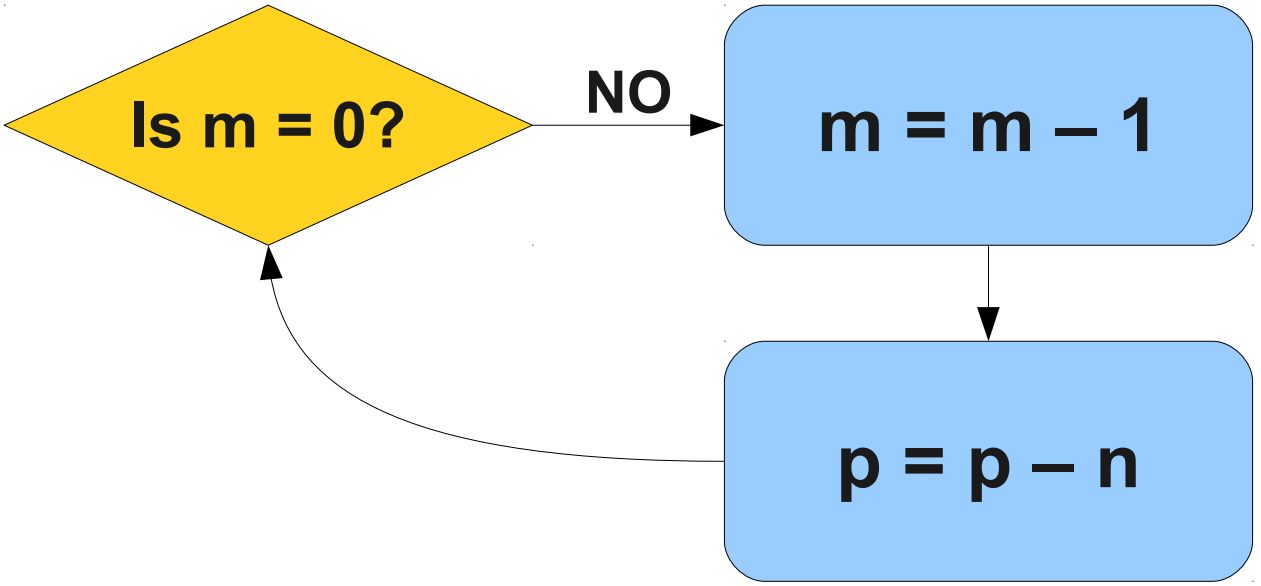
A Sketch of the Algorithm



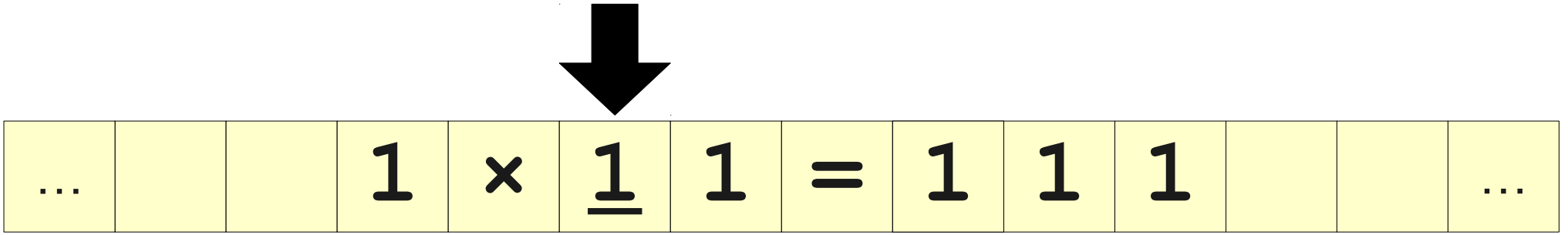


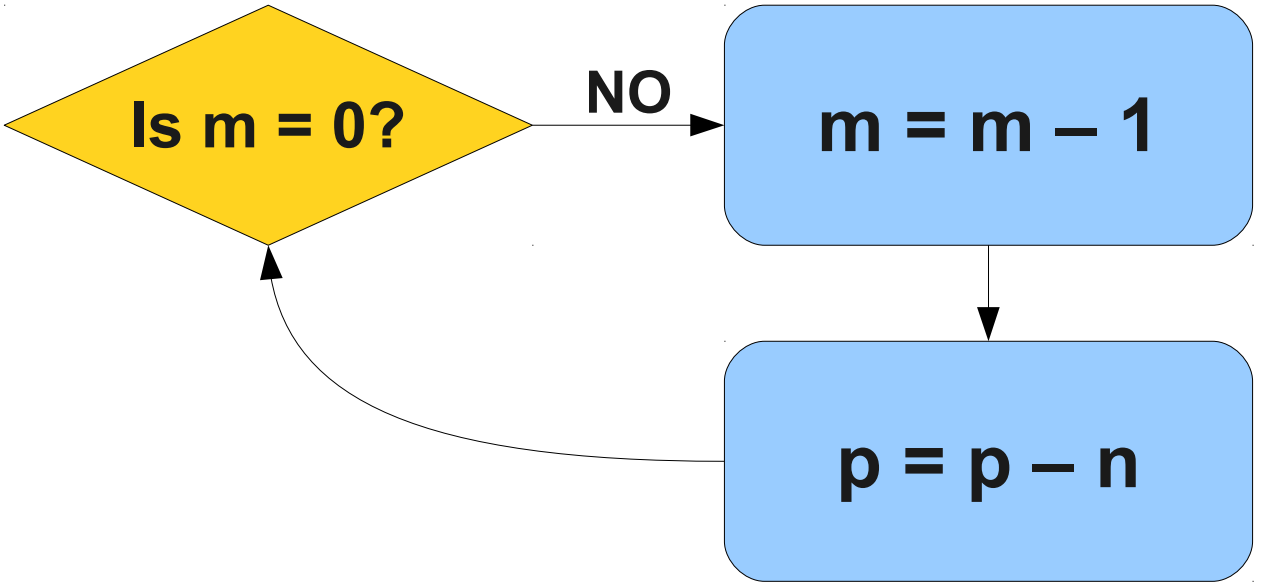
A Sketch of the Algorithm



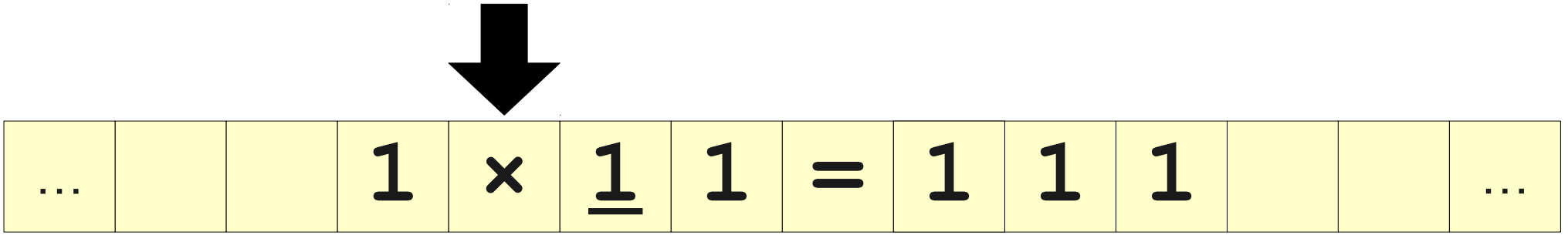


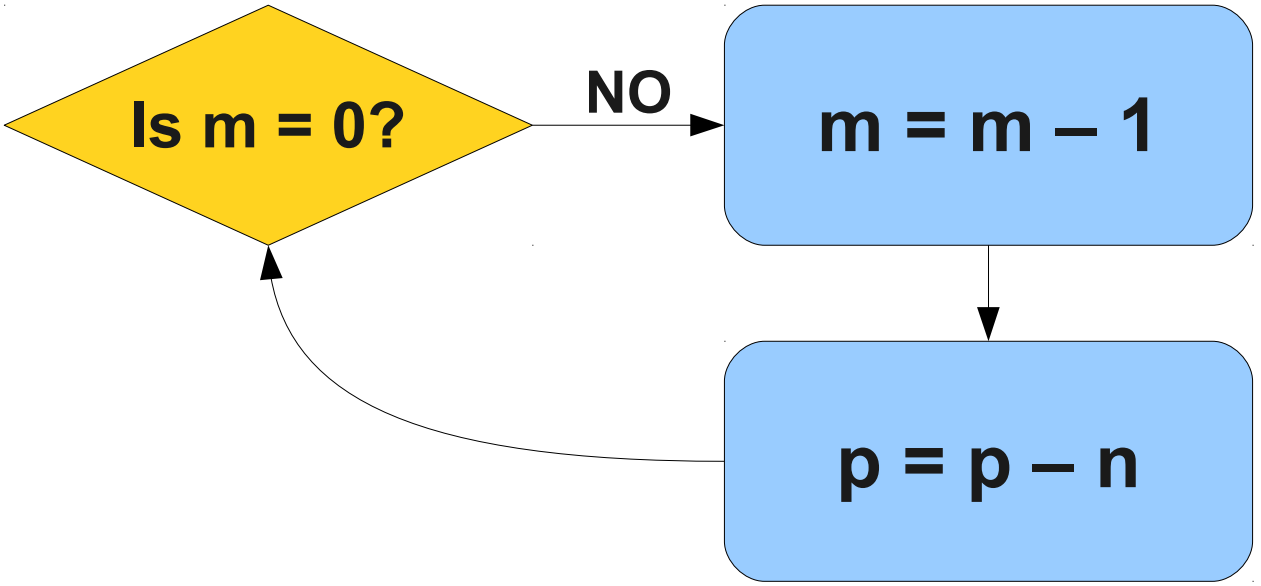
A Sketch of the Algorithm



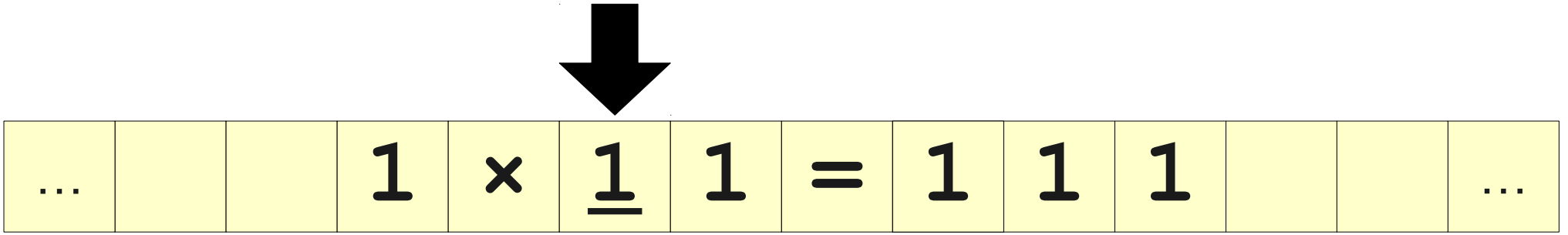


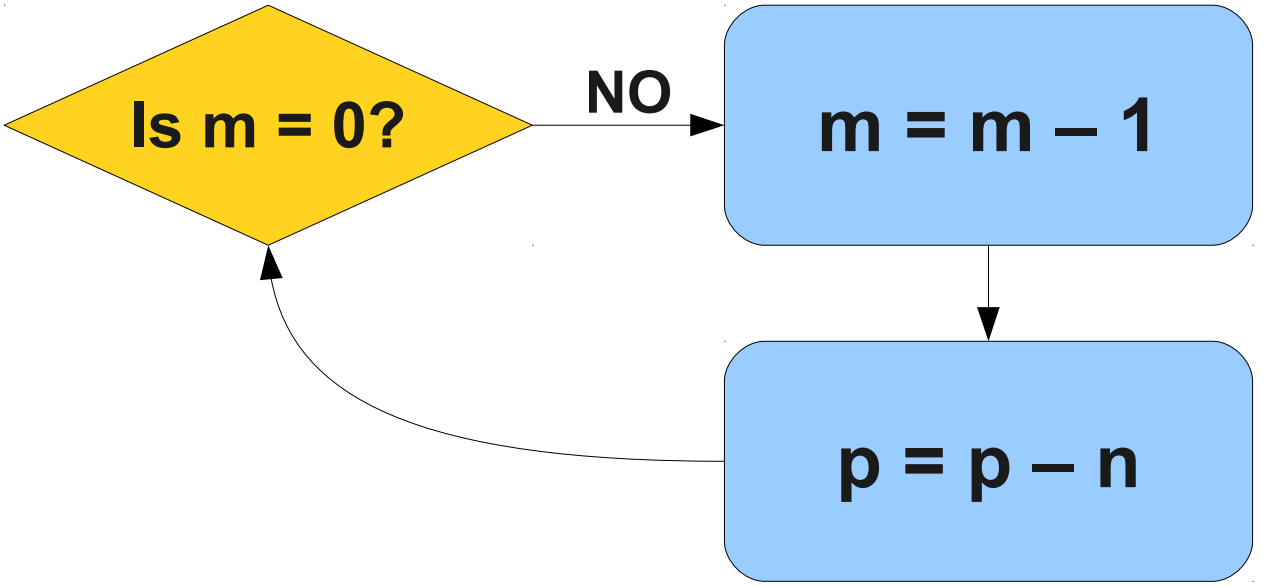
A Sketch of the Algorithm



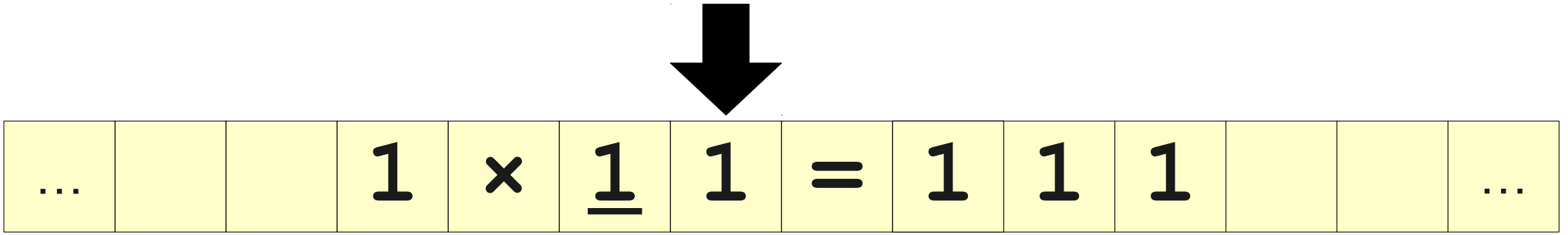


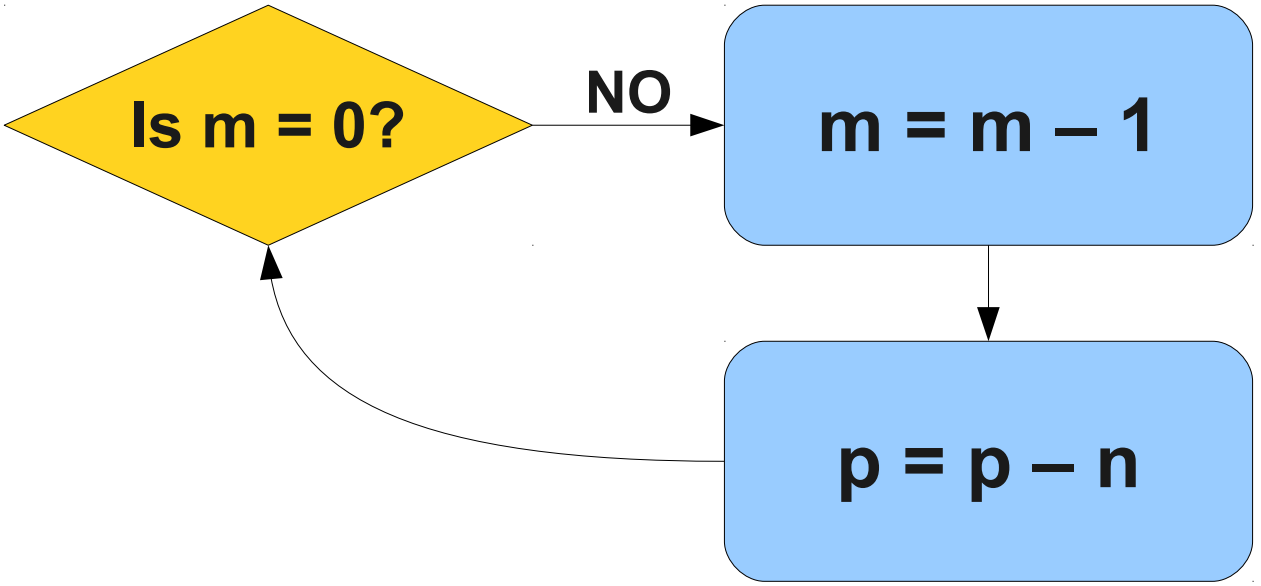
A Sketch of the Algorithm



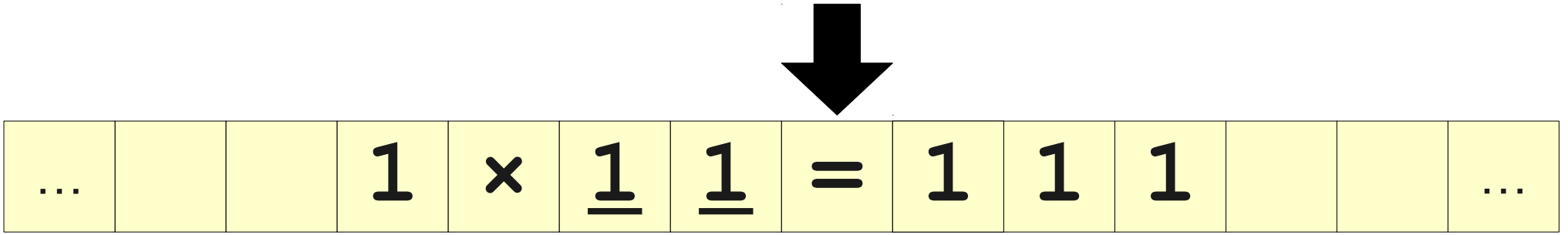


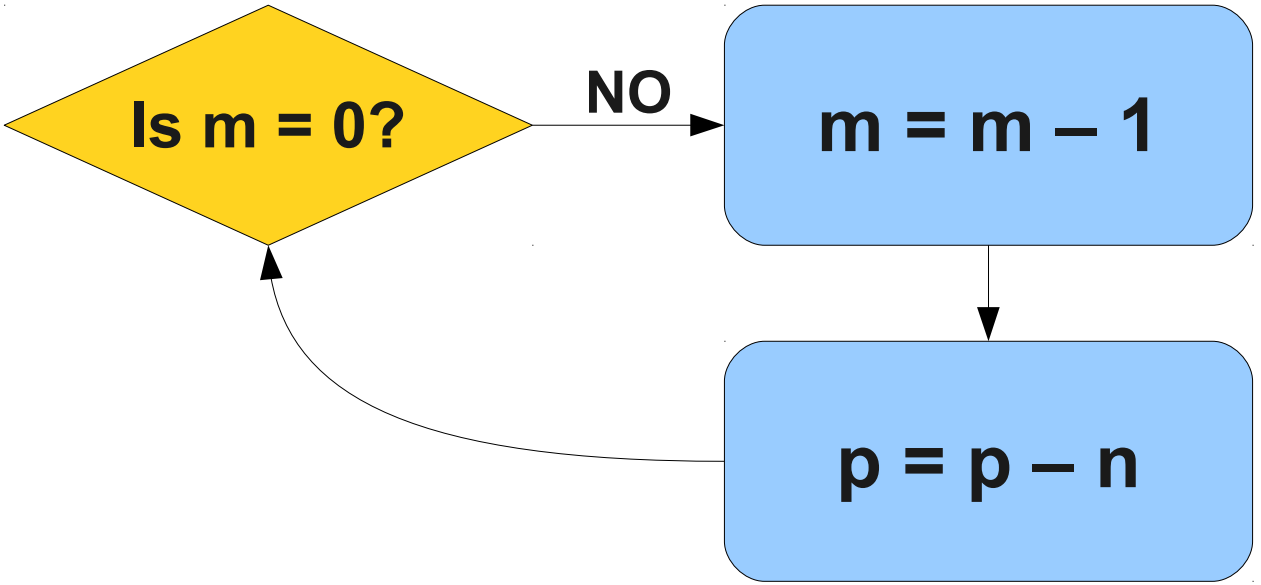
A Sketch of the Algorithm



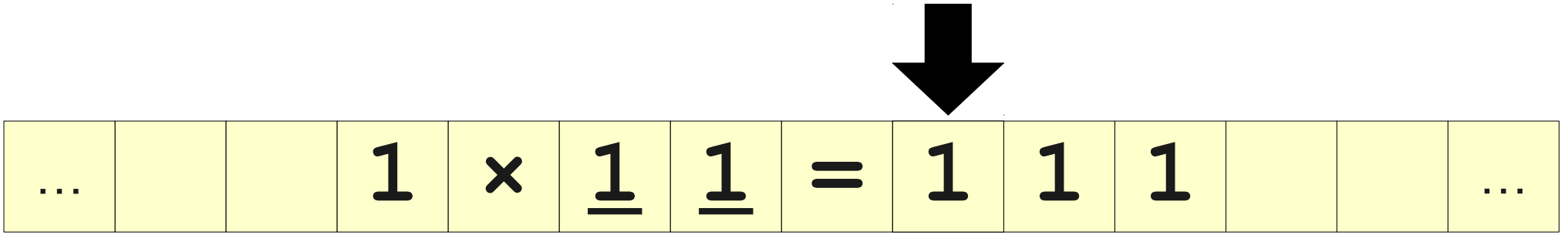


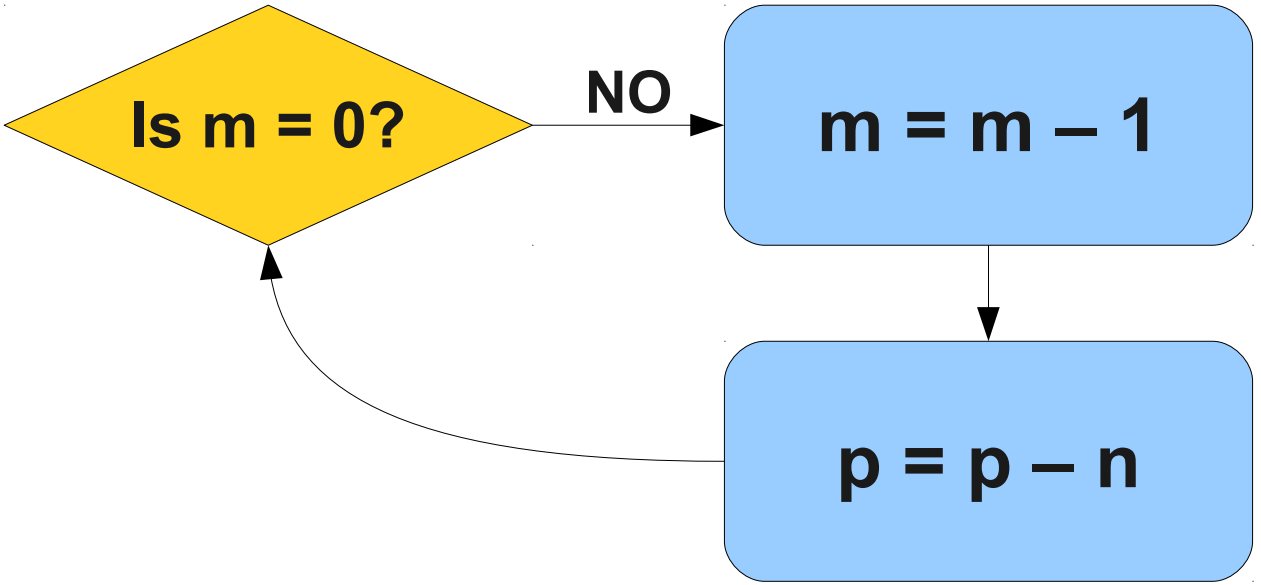
A Sketch of the Algorithm



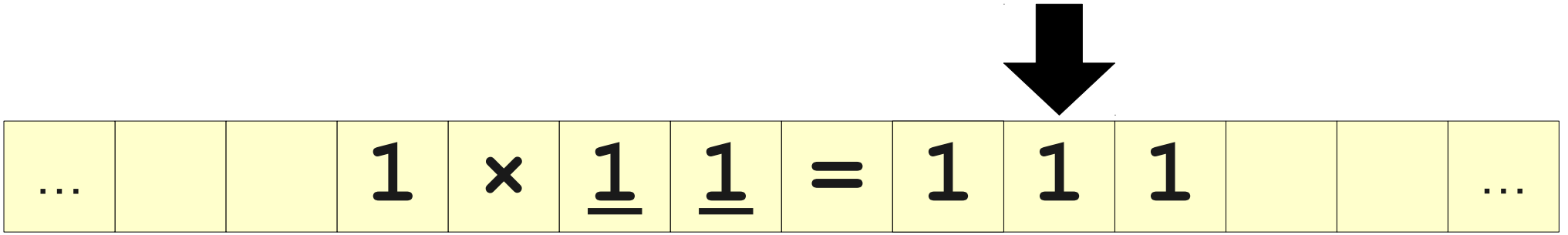


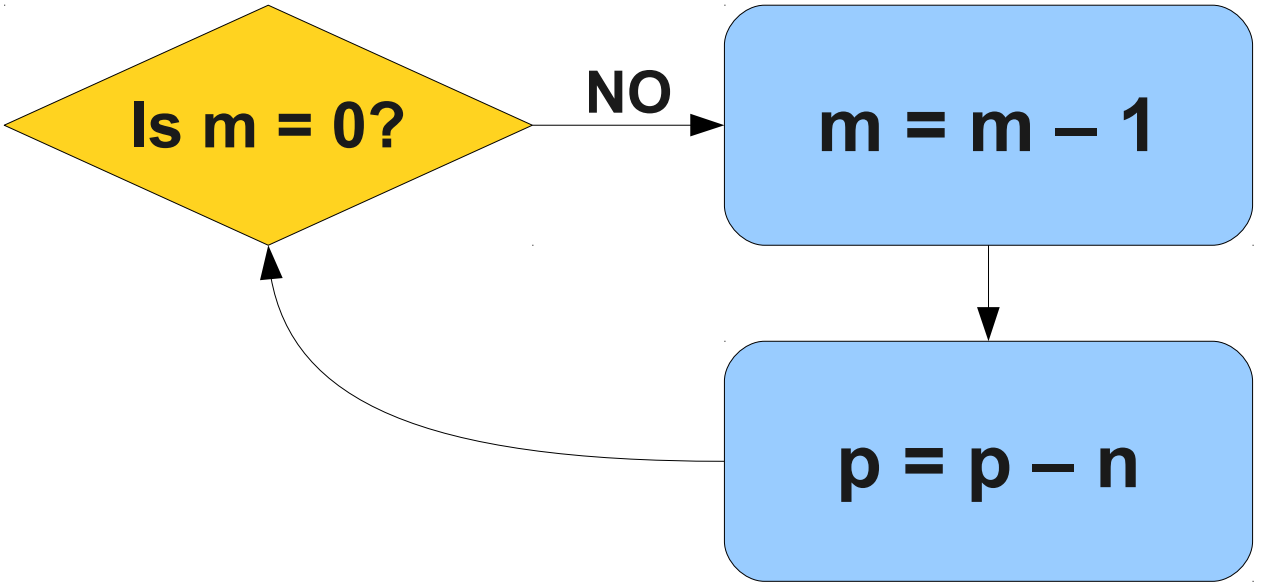
A Sketch of the Algorithm



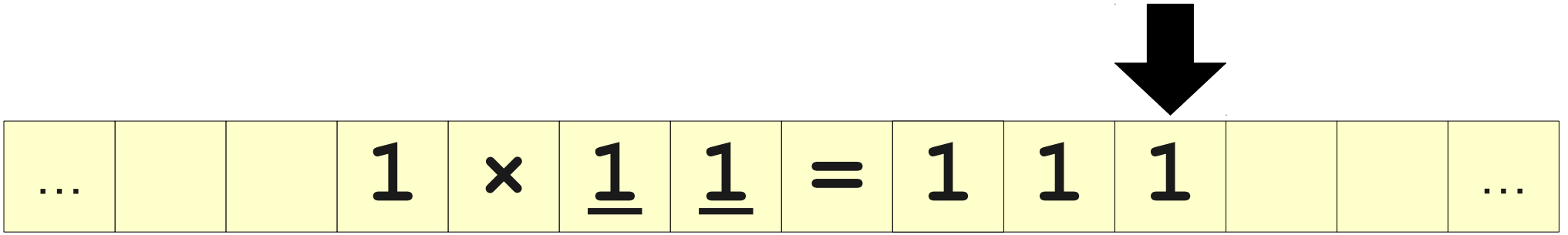


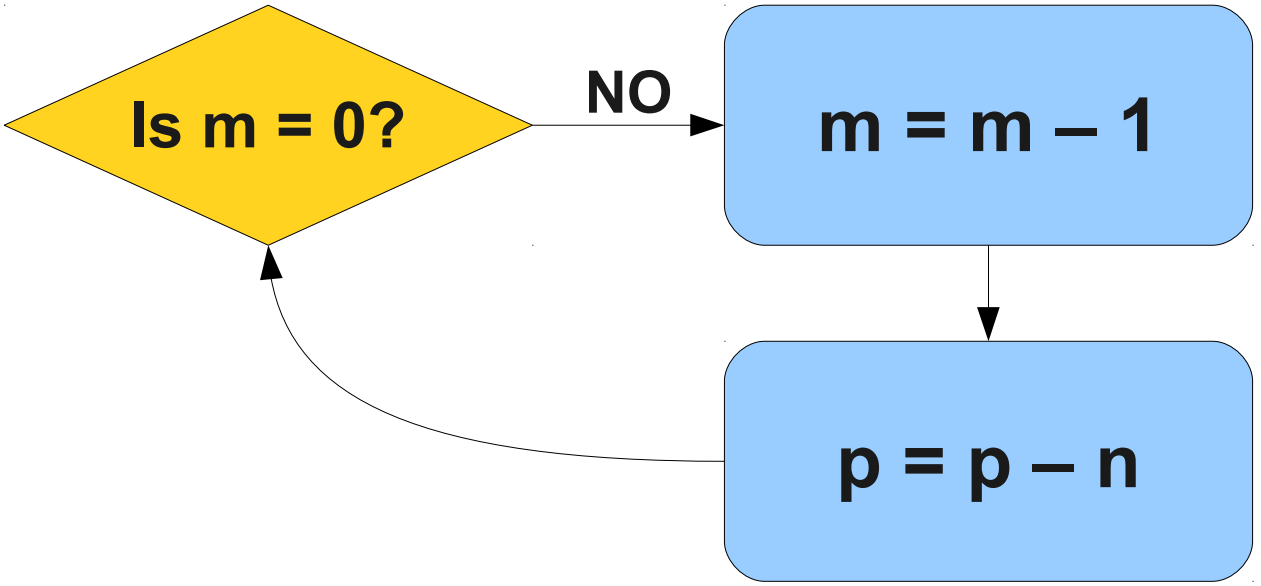
A Sketch of the Algorithm



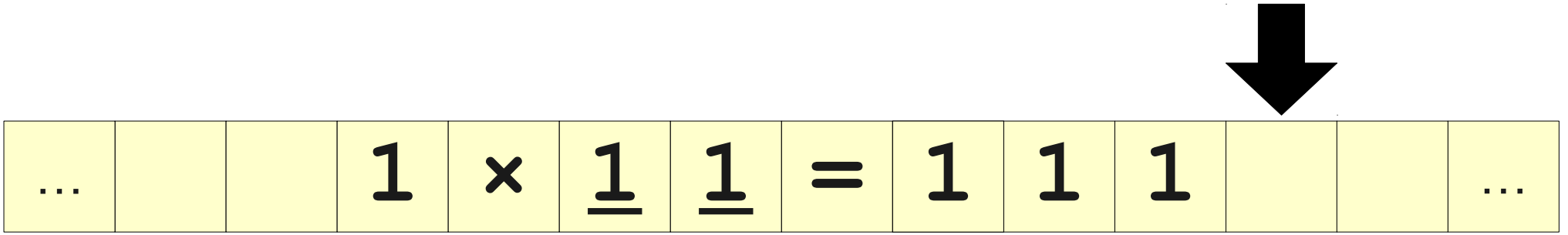


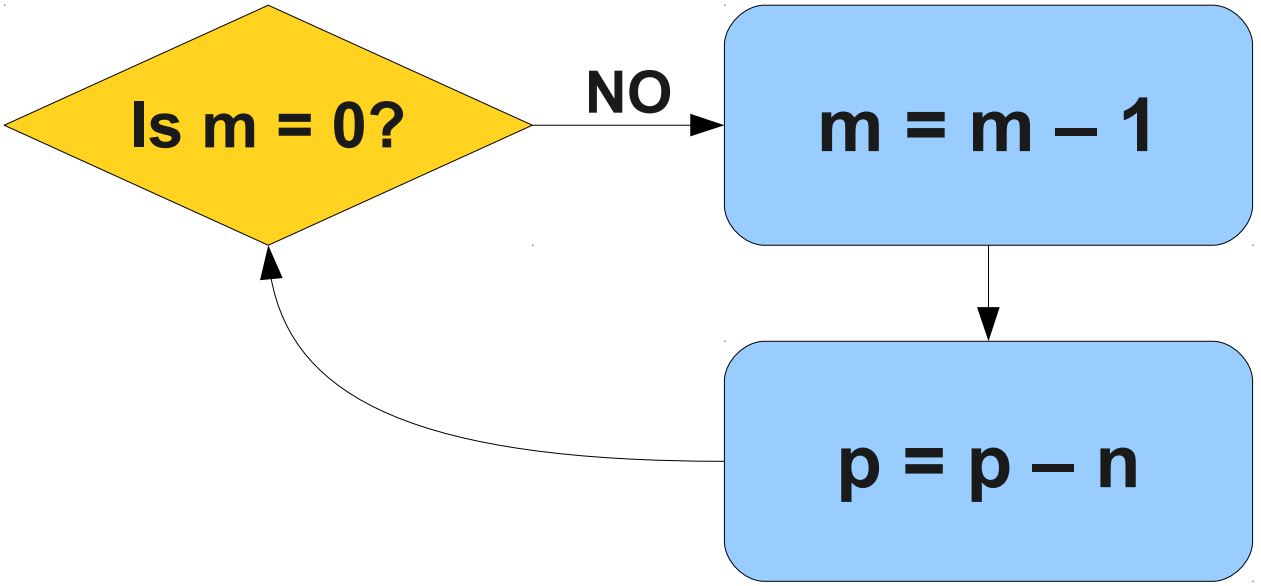
A Sketch of the Algorithm



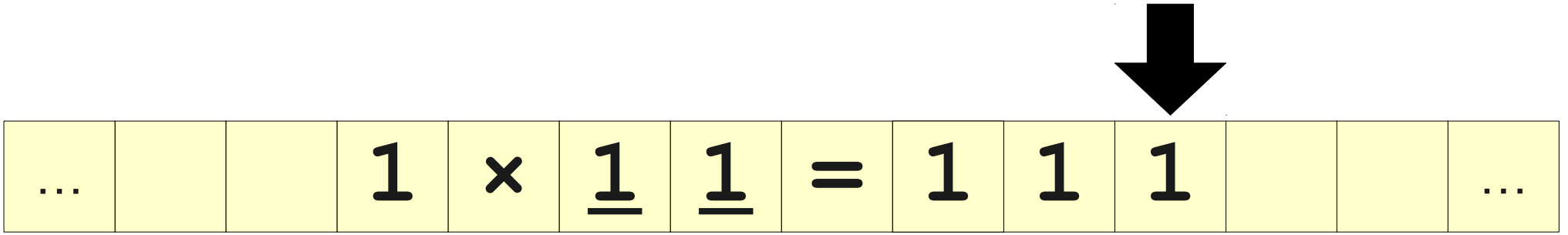


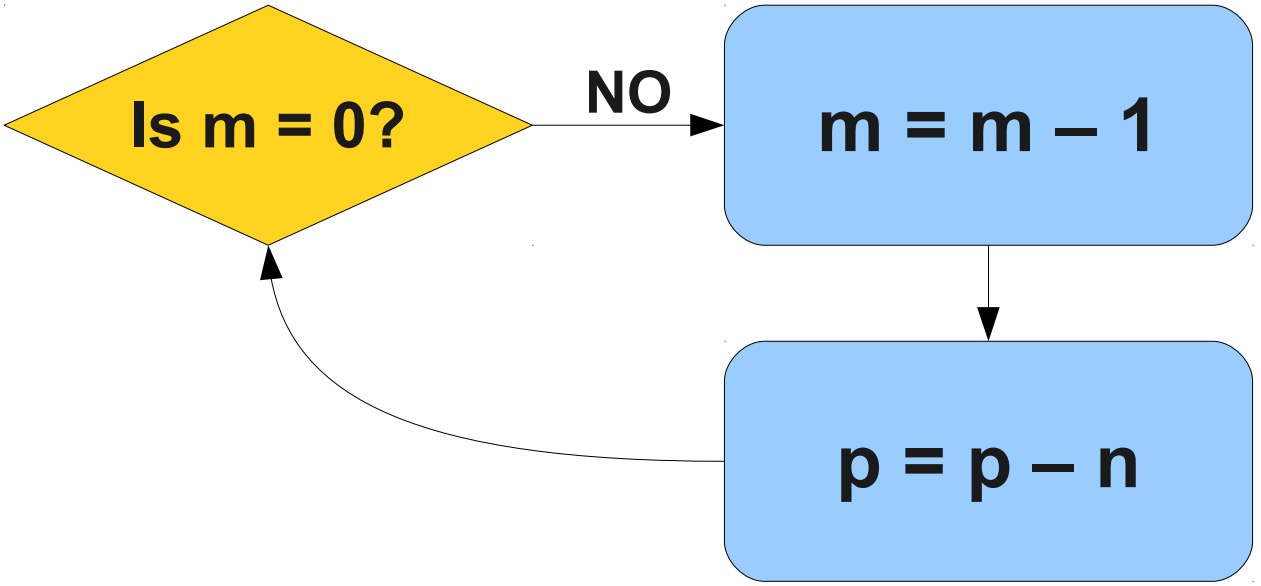
A Sketch of the Algorithm



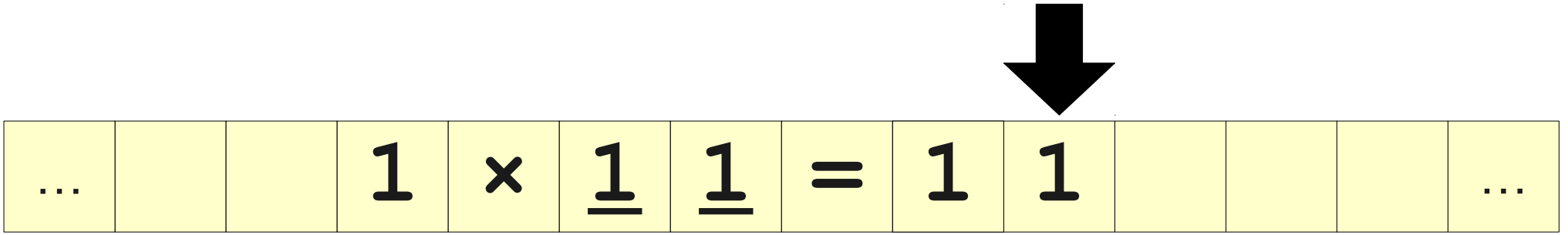


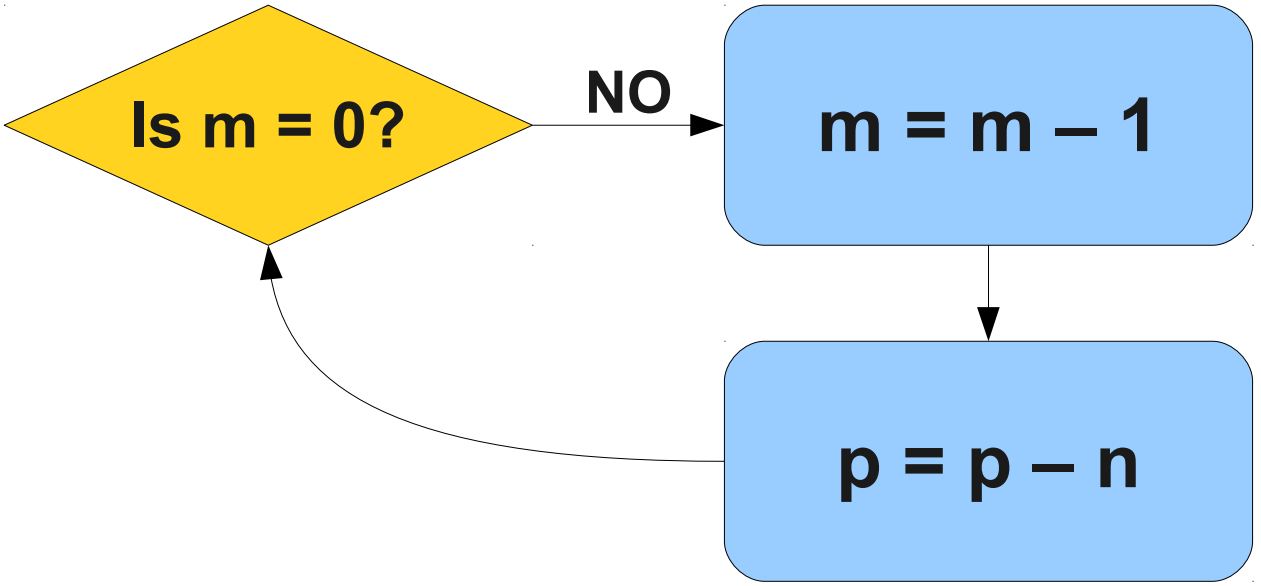
A Sketch of the Algorithm



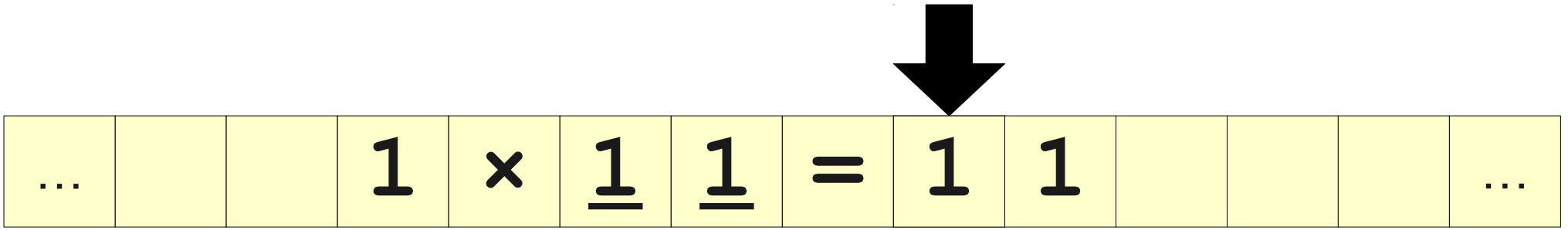


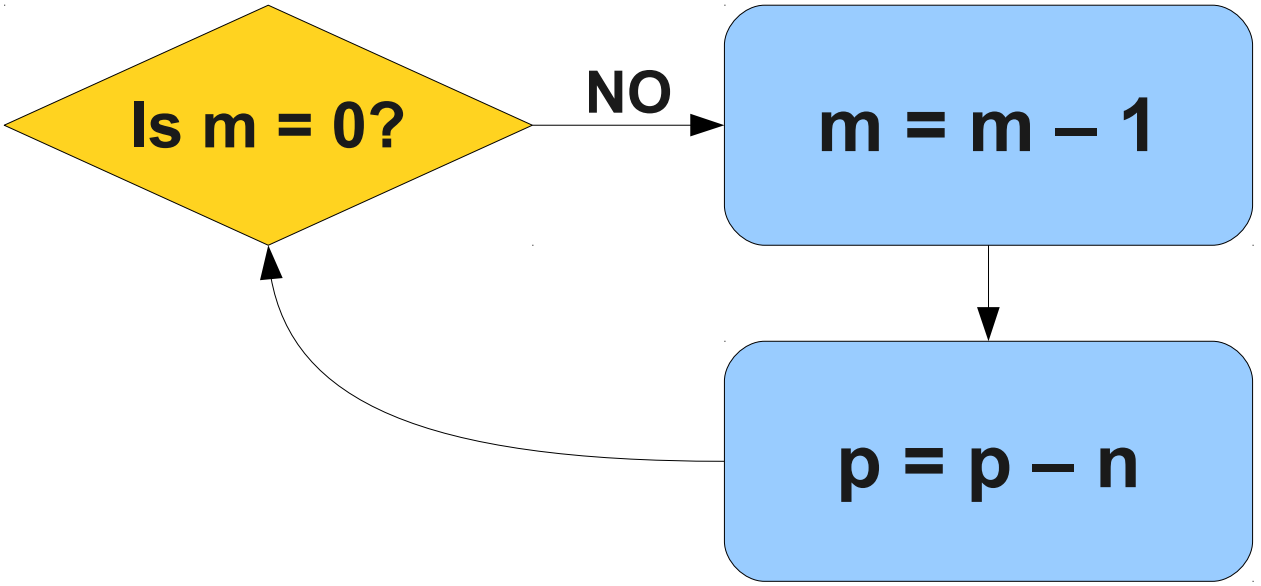
A Sketch of the Algorithm



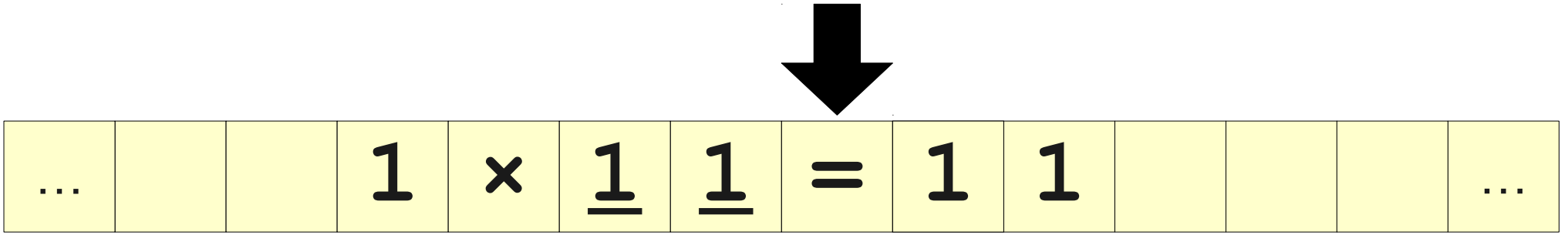


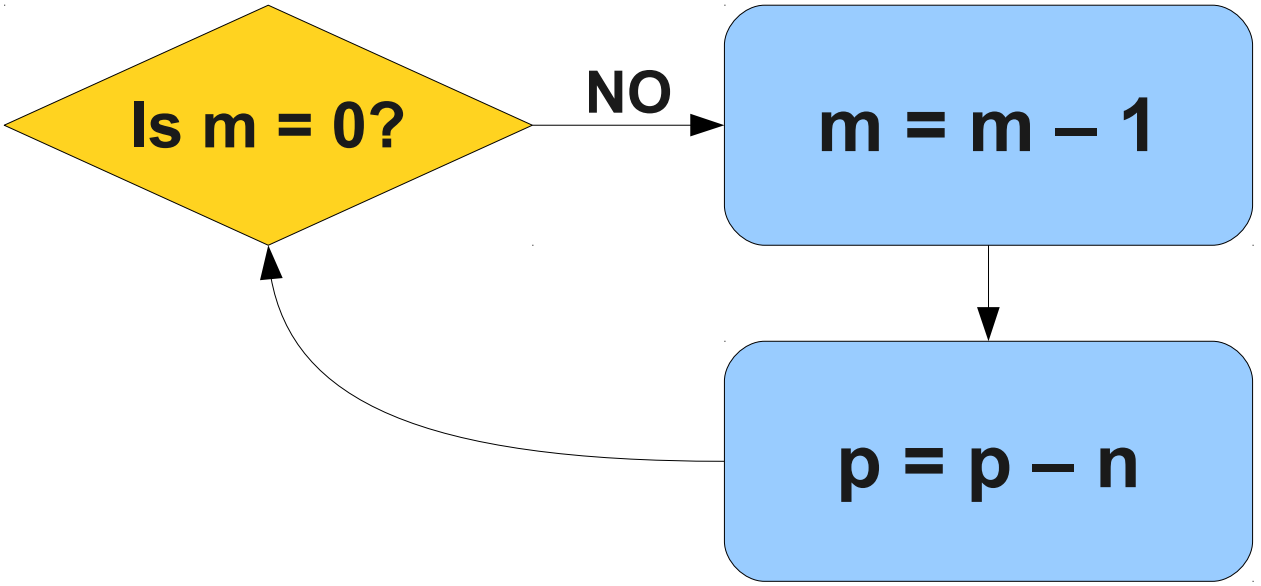
A Sketch of the Algorithm



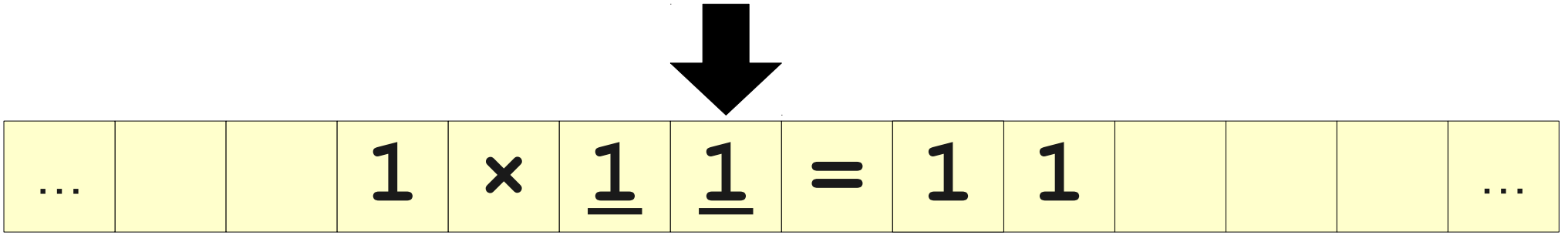


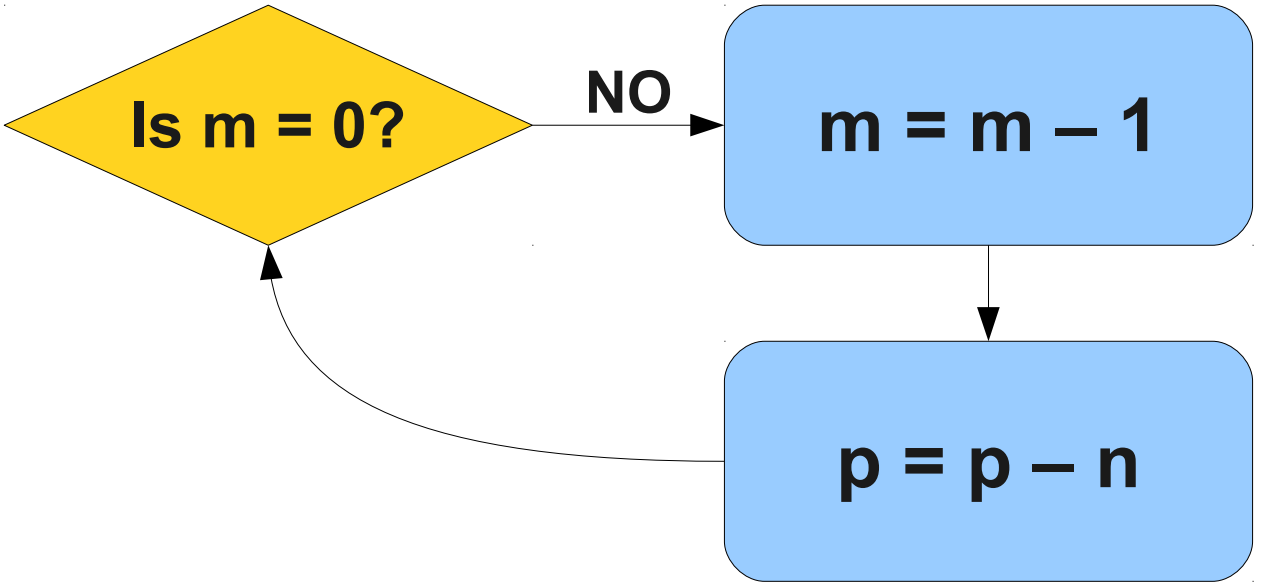
A Sketch of the Algorithm



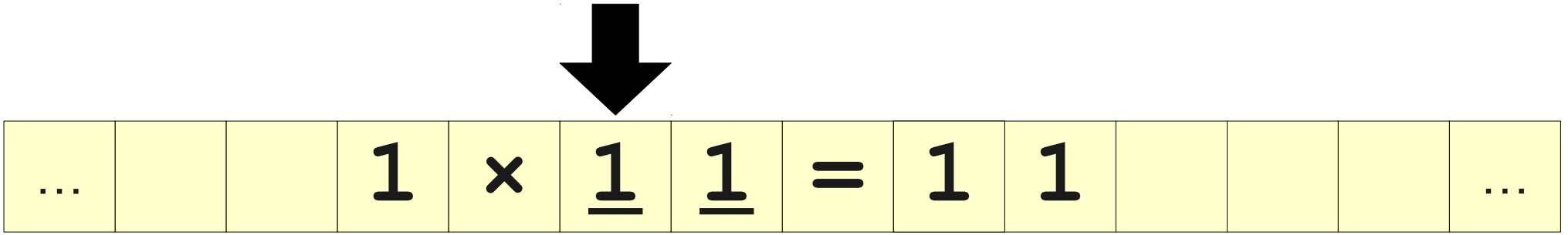


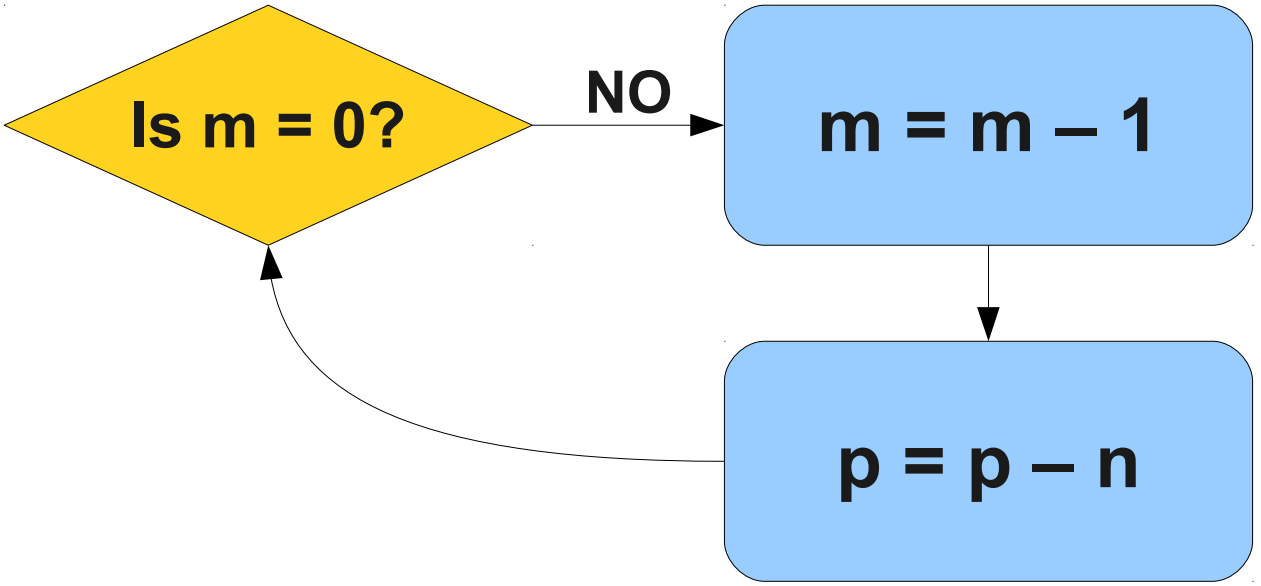
A Sketch of the Algorithm



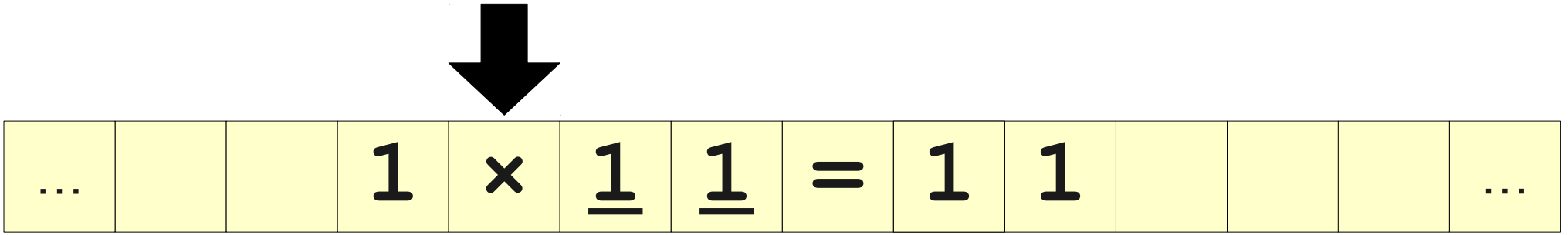


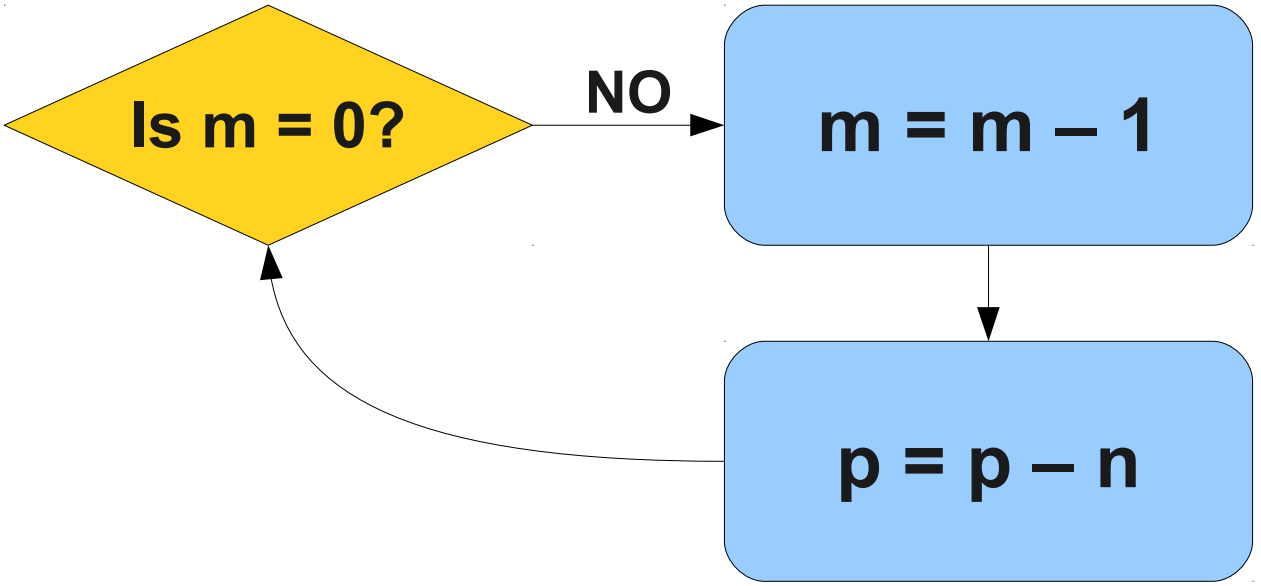
A Sketch of the Algorithm



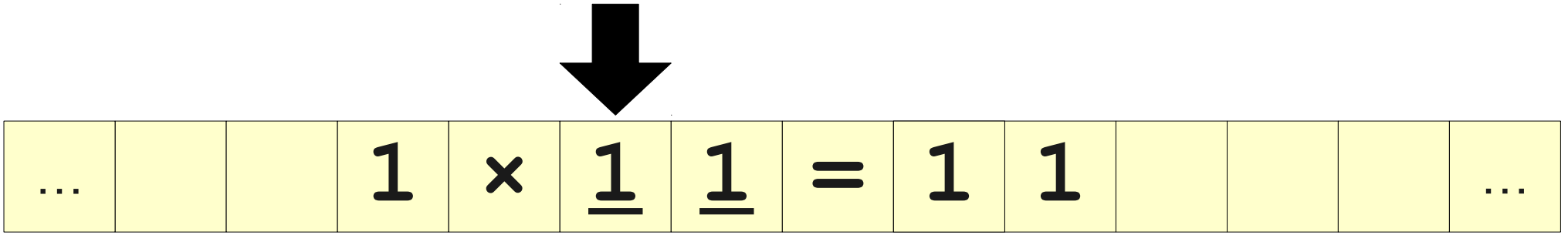


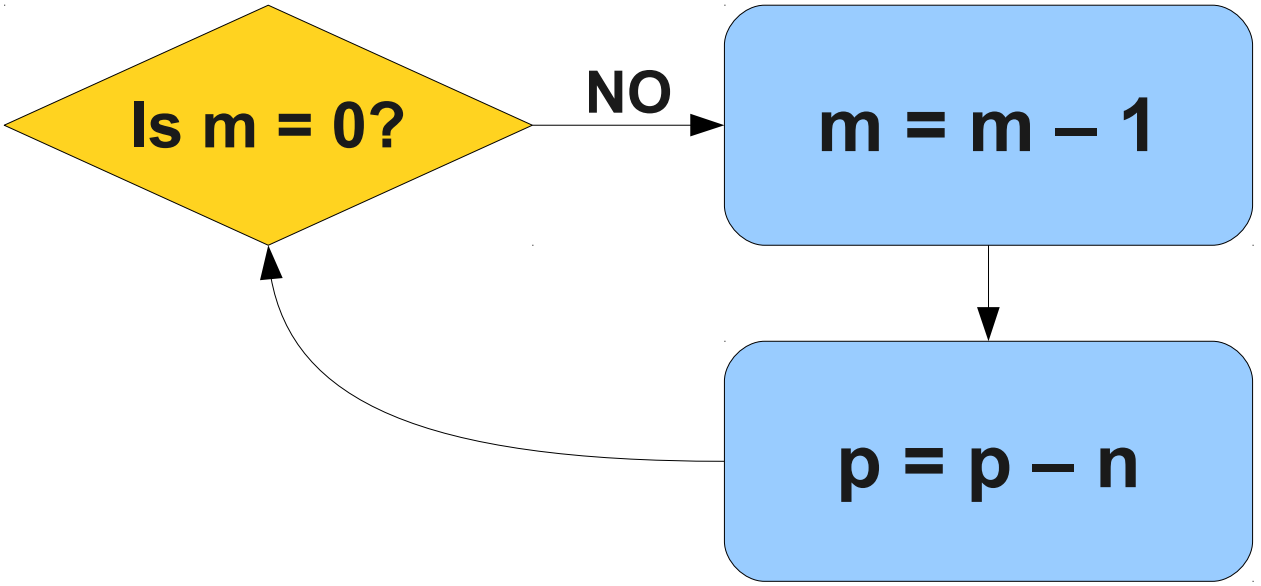
A Sketch of the Algorithm



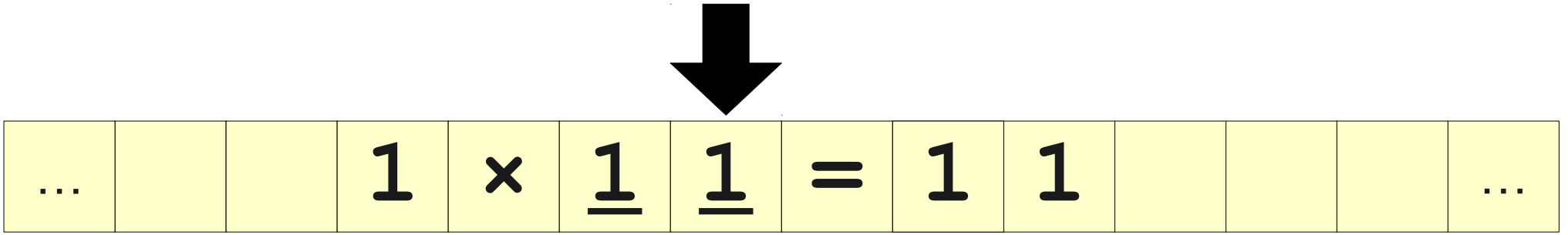


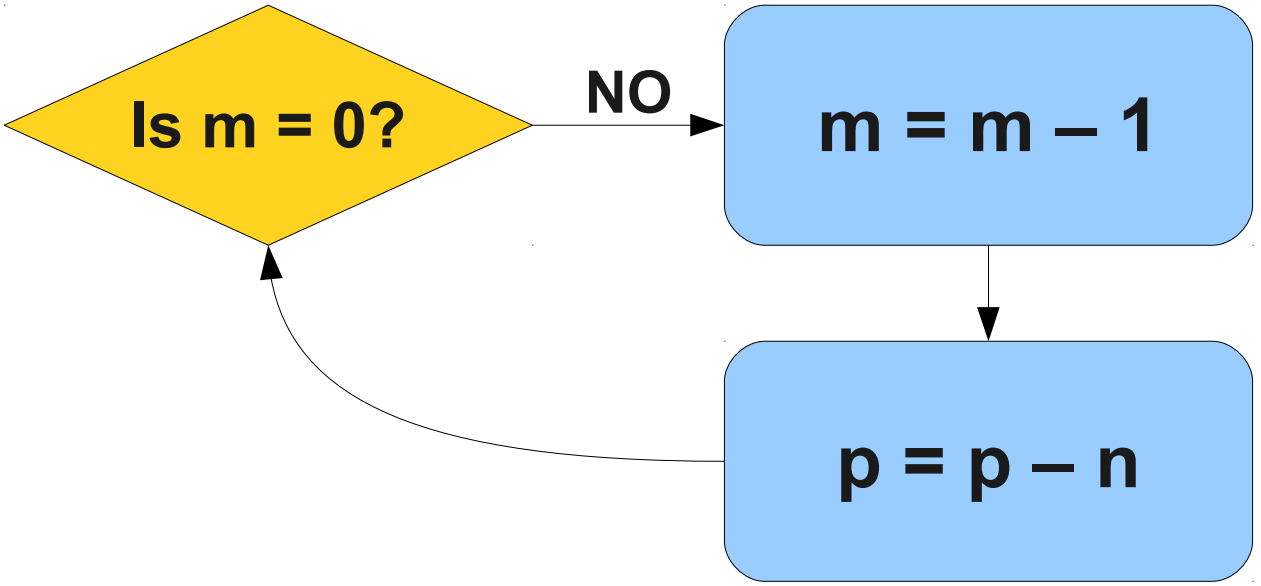
A Sketch of the Algorithm



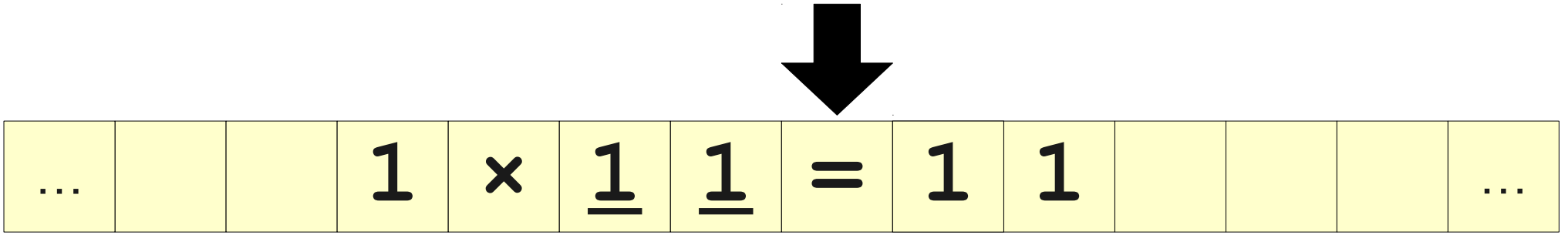


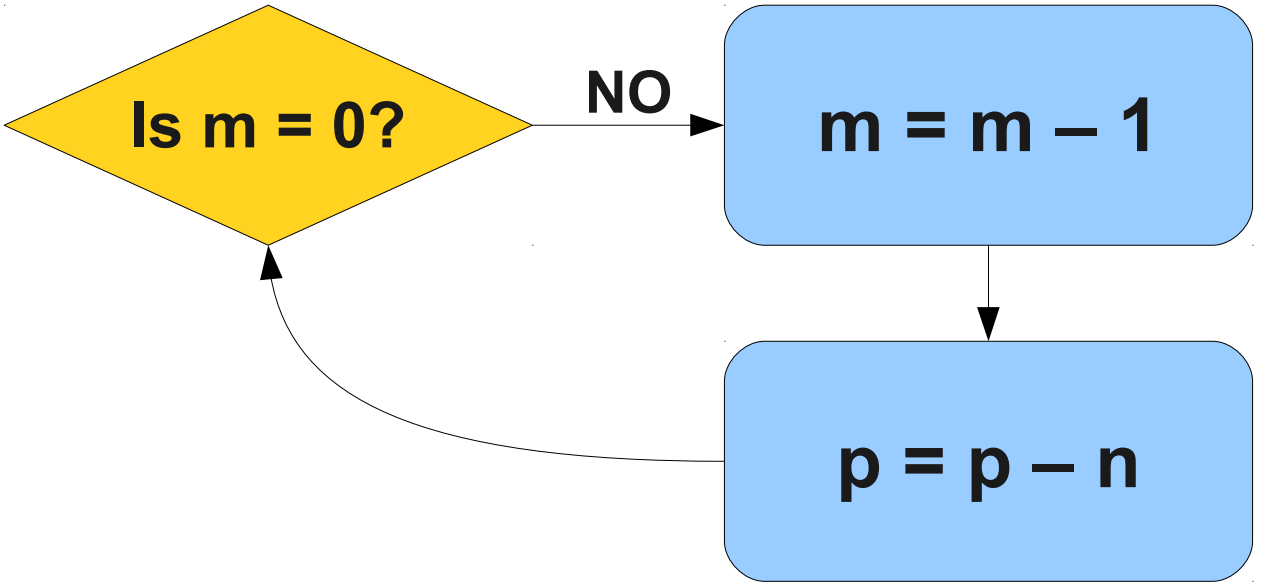
A Sketch of the Algorithm



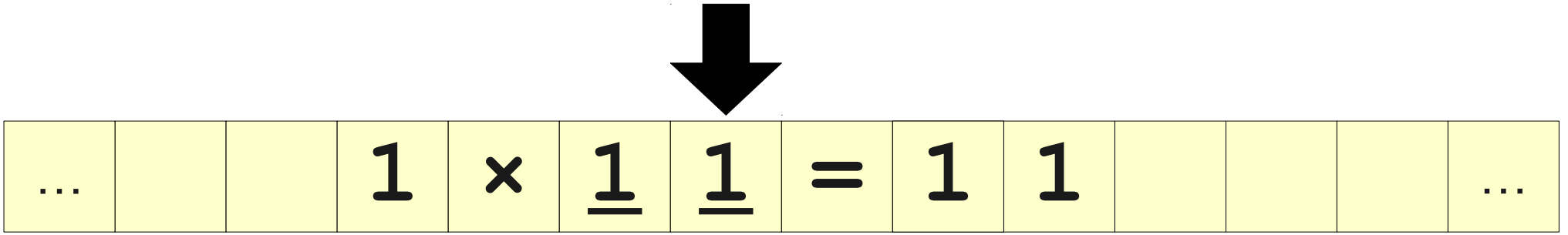


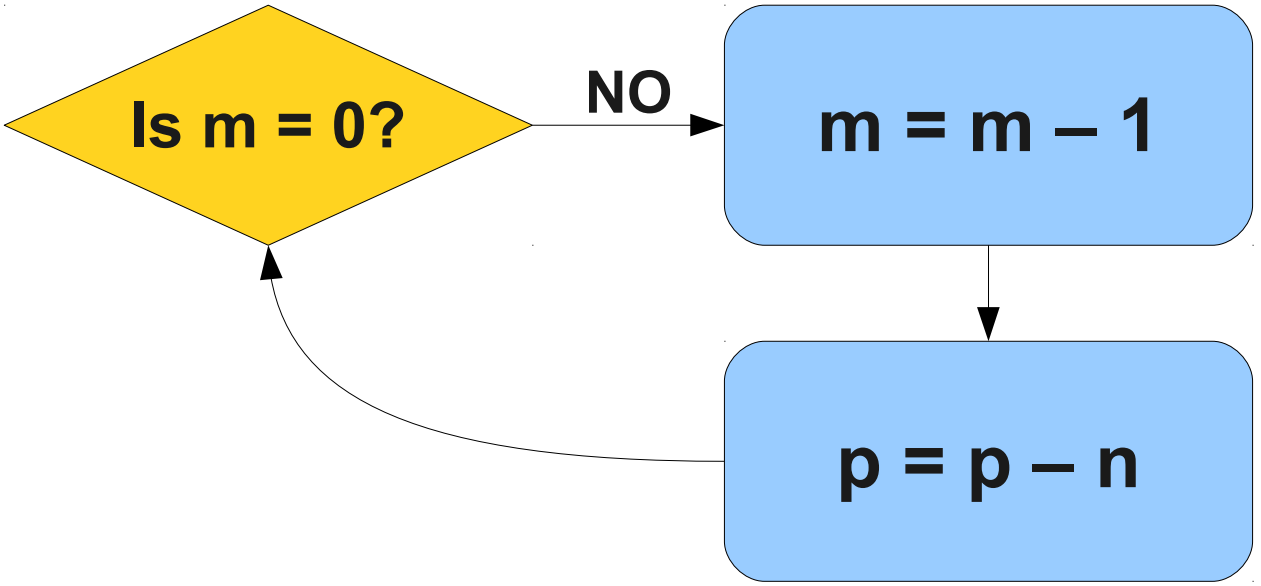
A Sketch of the Algorithm



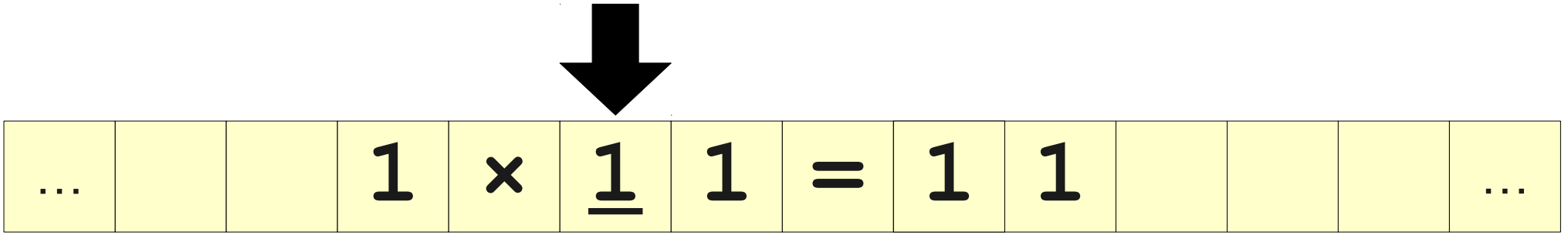


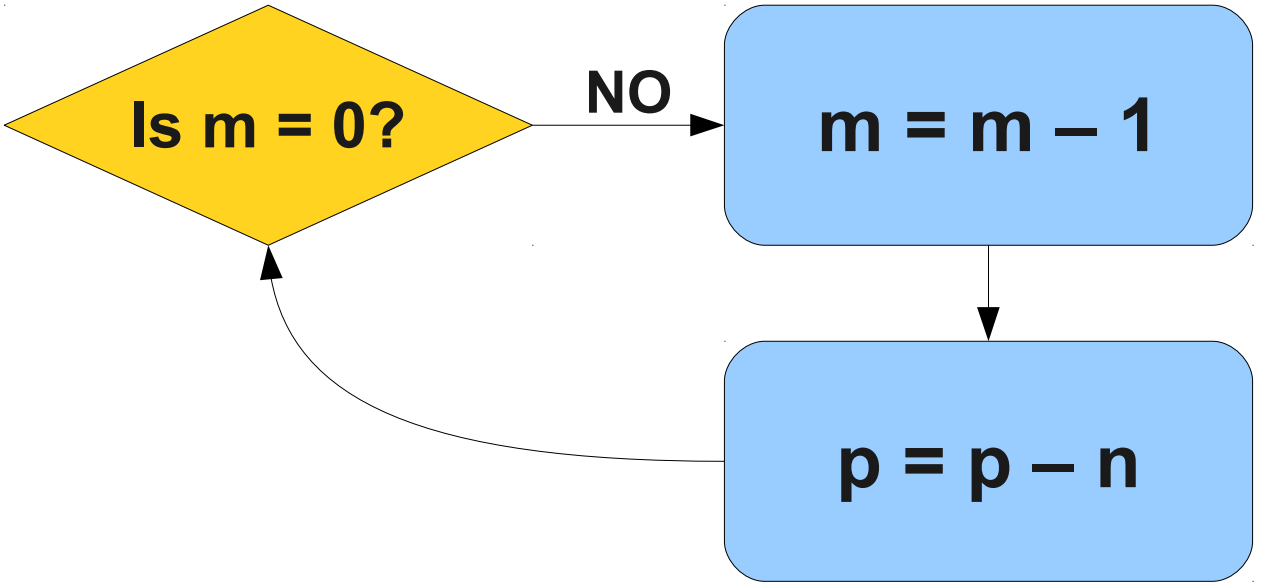
A Sketch of the Algorithm



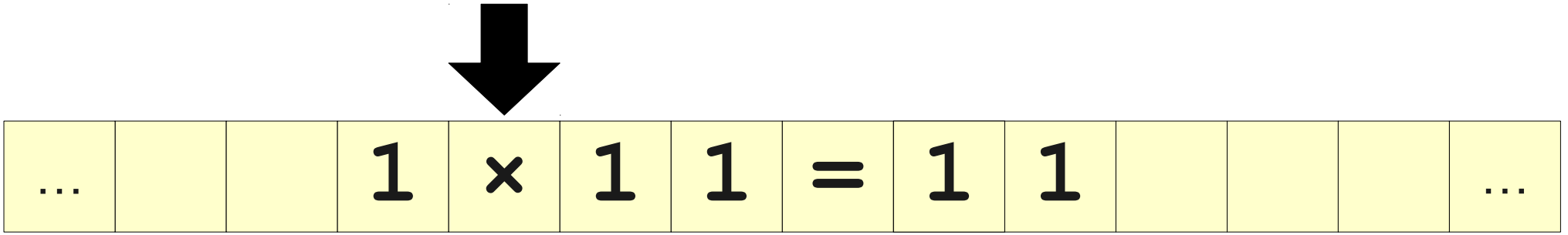


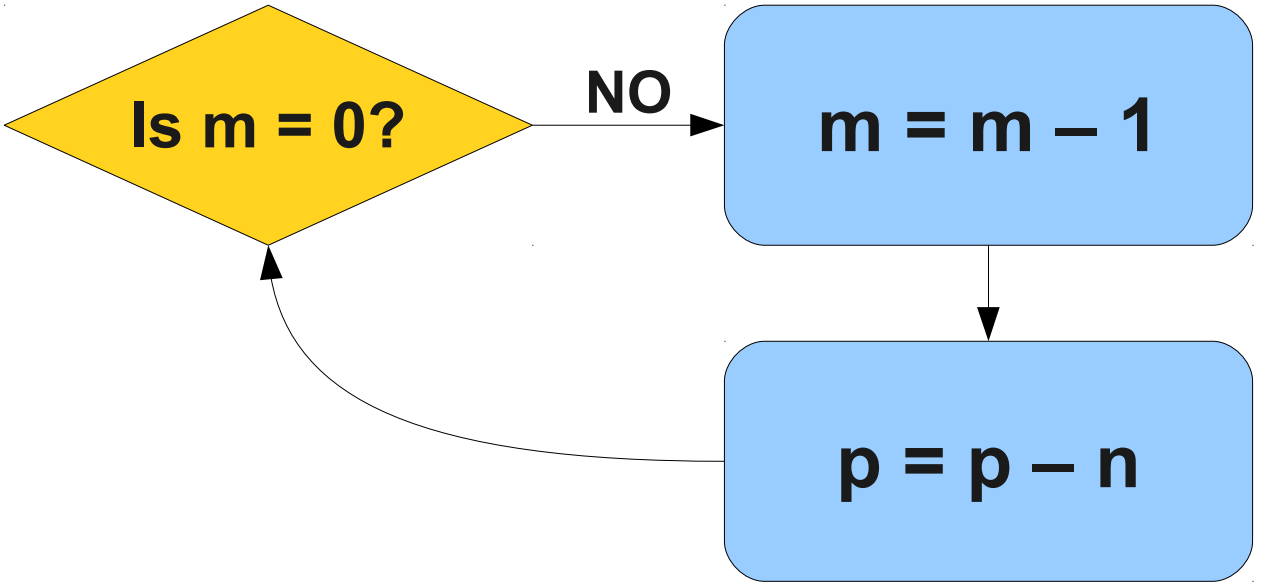
A Sketch of the Algorithm



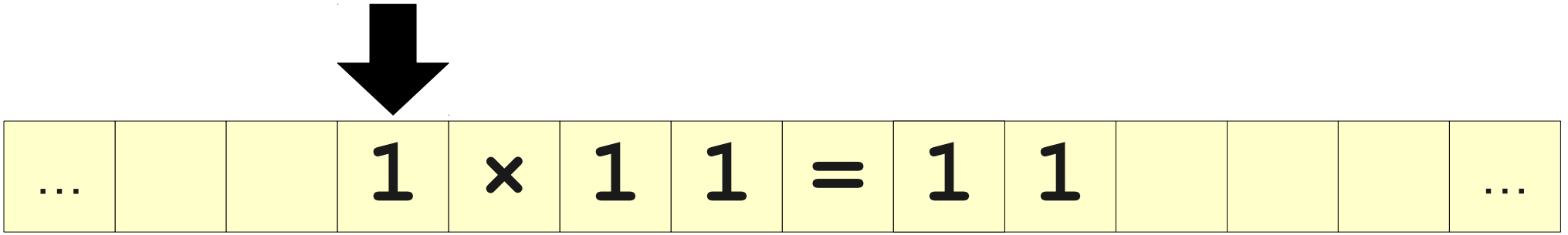


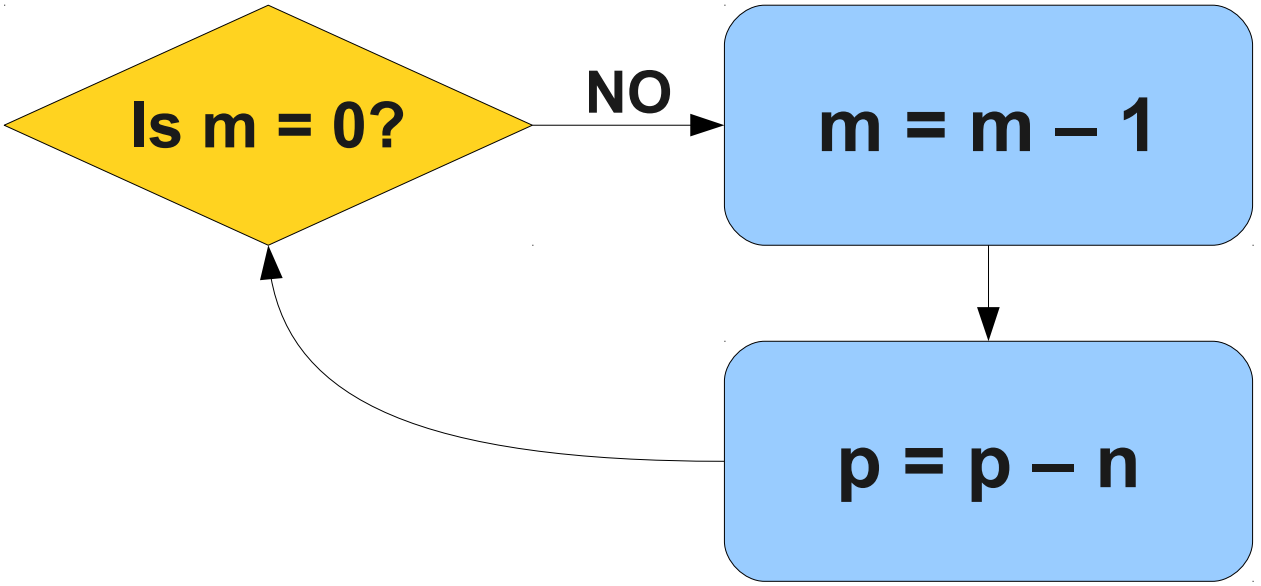
A Sketch of the Algorithm



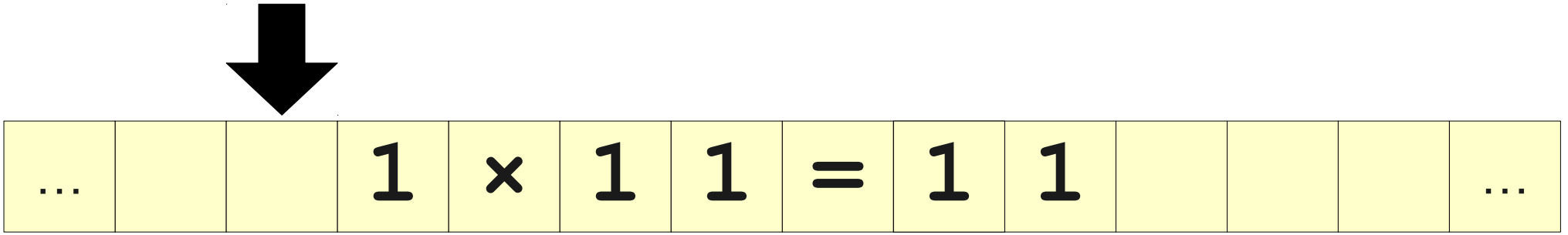


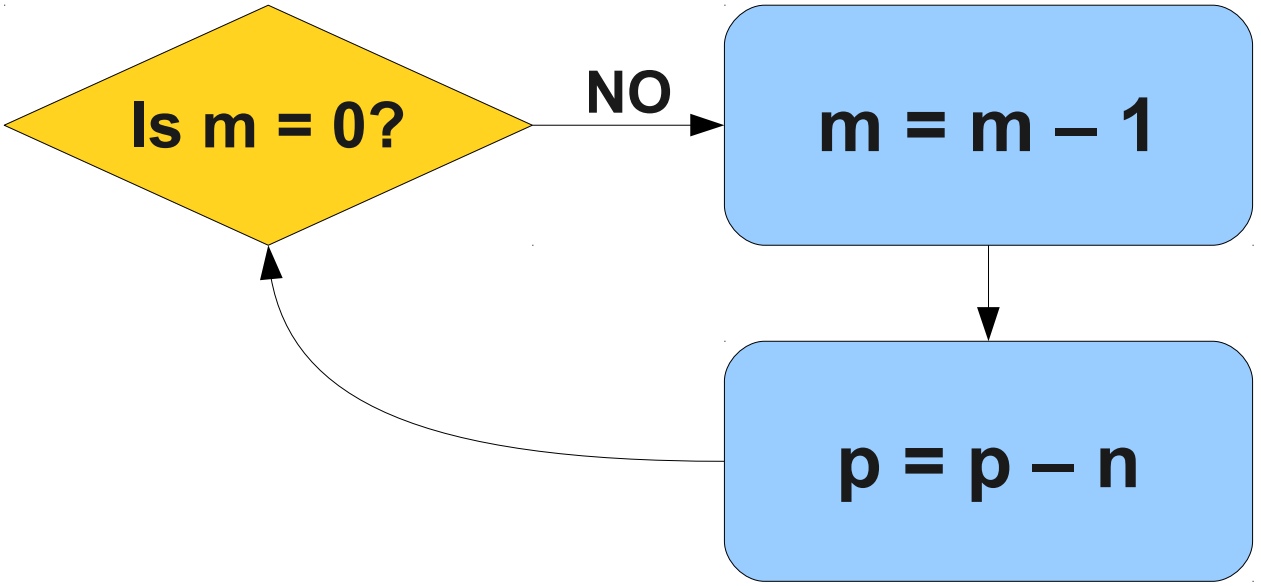
A Sketch of the Algorithm



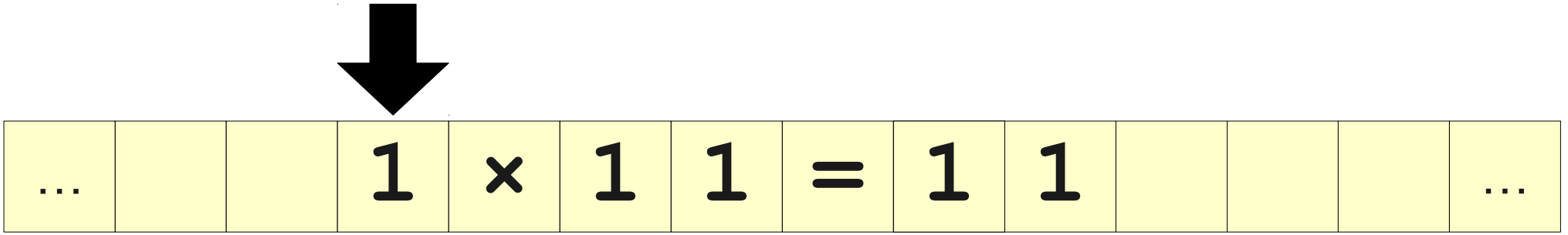


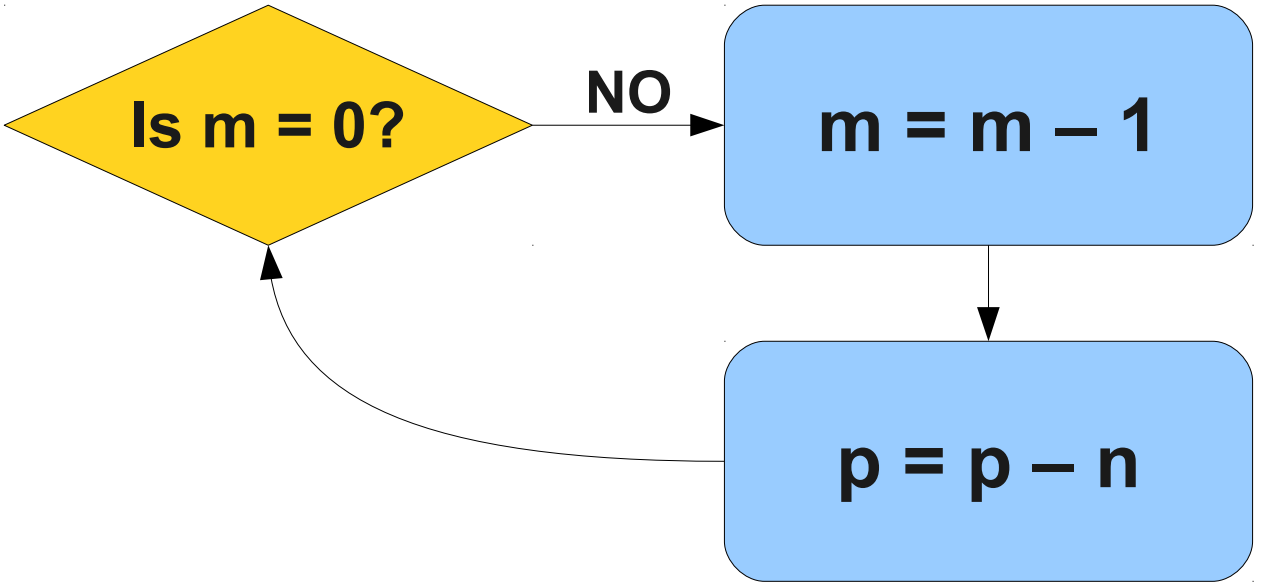
A Sketch of the Algorithm

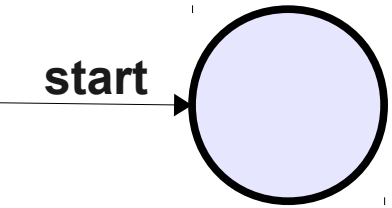


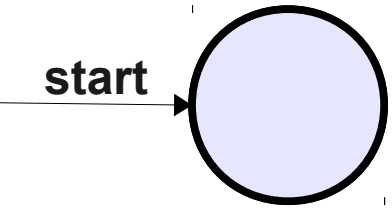


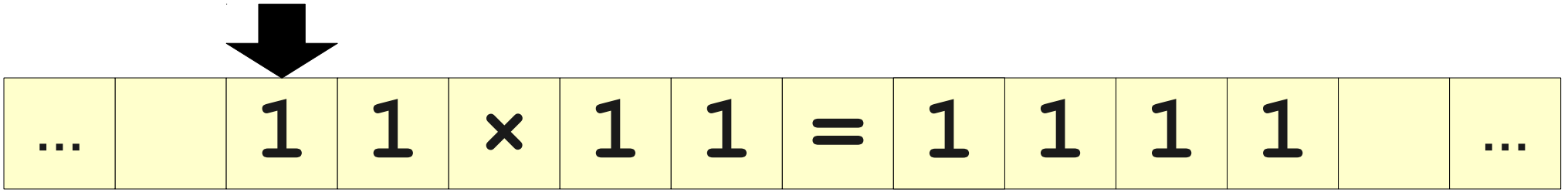
A Sketch of the Algorithm

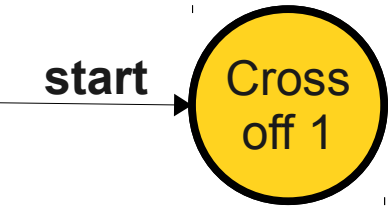


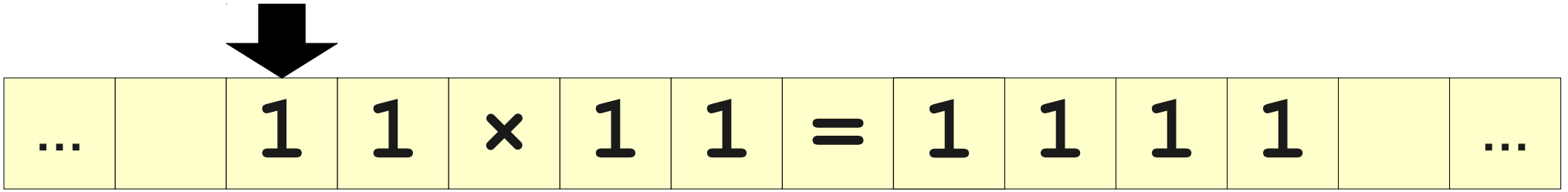


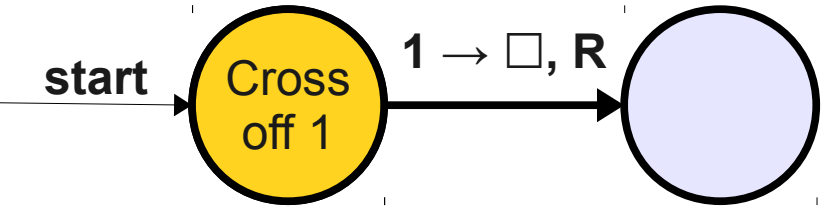

start

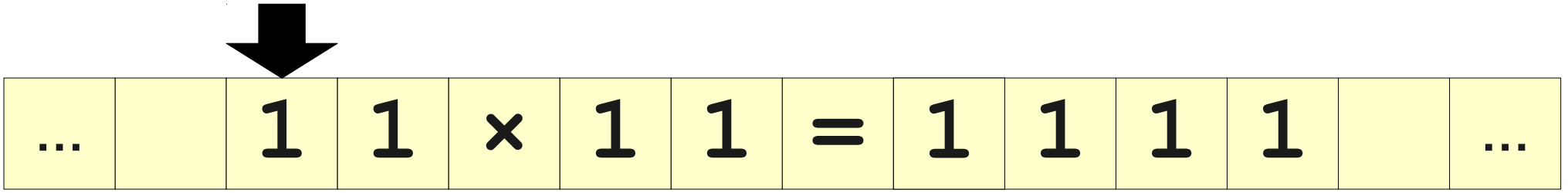

start

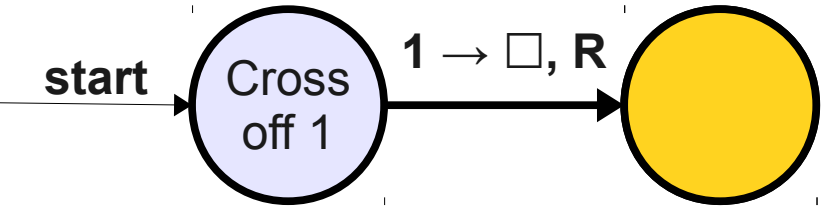

...
1 1 × 1 1 = 1 1 1 1
...

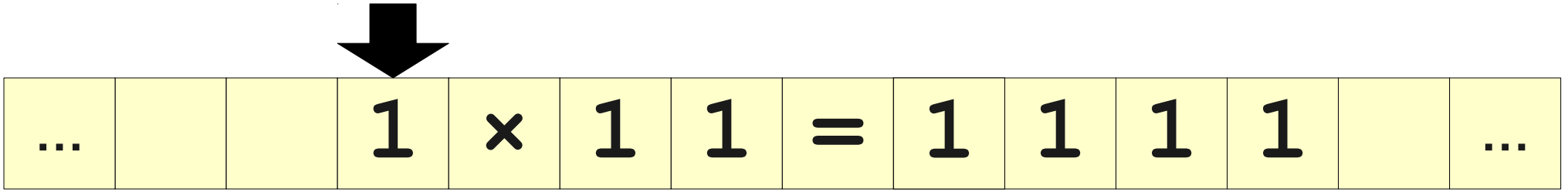

start
Cross
off 1

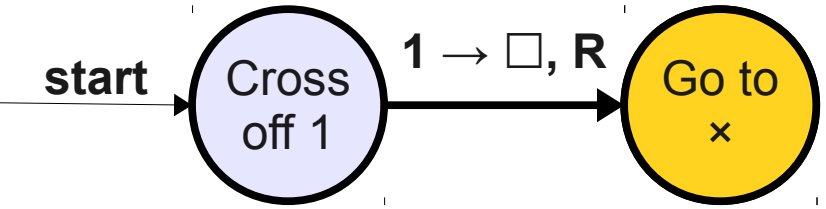

...
1
1
×
1
1
=
1
1
1
1
...

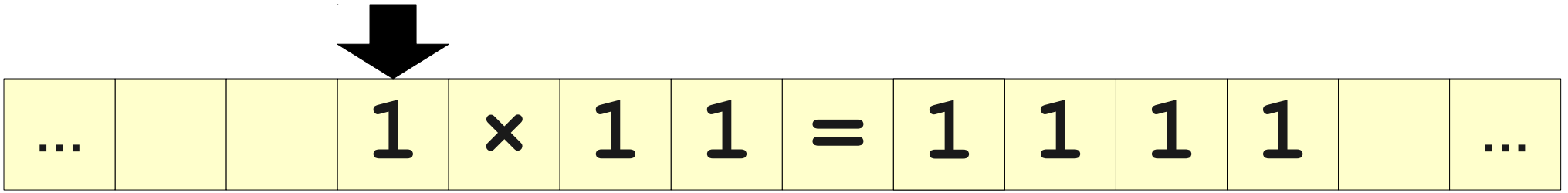

start
Cross off 1
1 → □, R

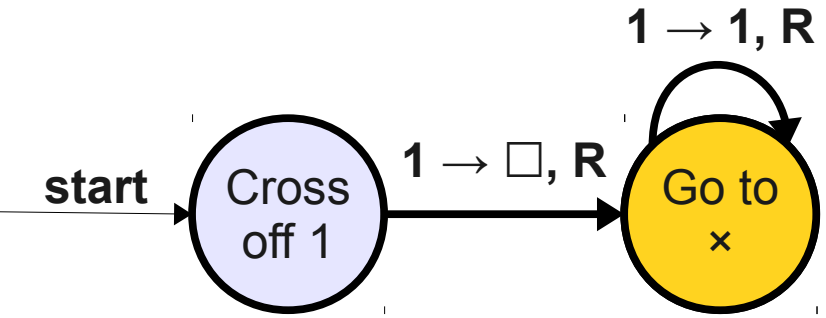

...
1 1 × 1 1 = 1 1 1 1
...

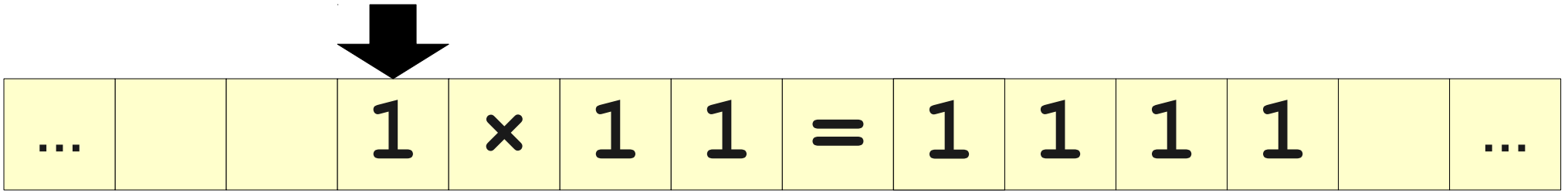

start
Cross off 1
1 → □, R

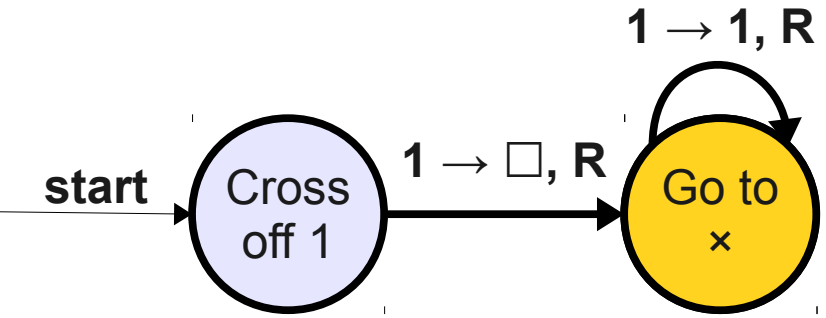

...
1 × 1 1 = 1 1 1 1
...

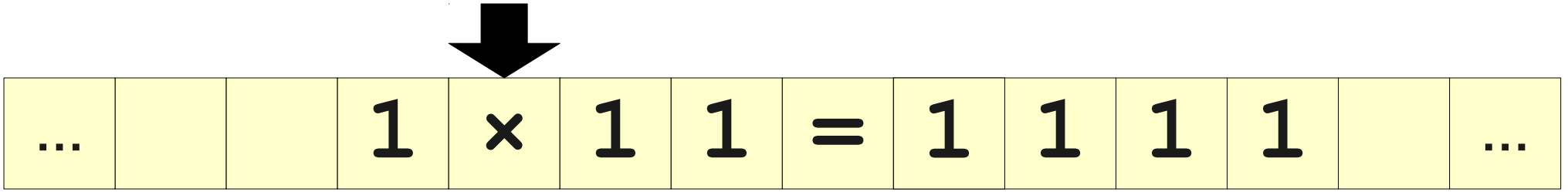

start
Cross off 1
1 → □, R
Go to ×

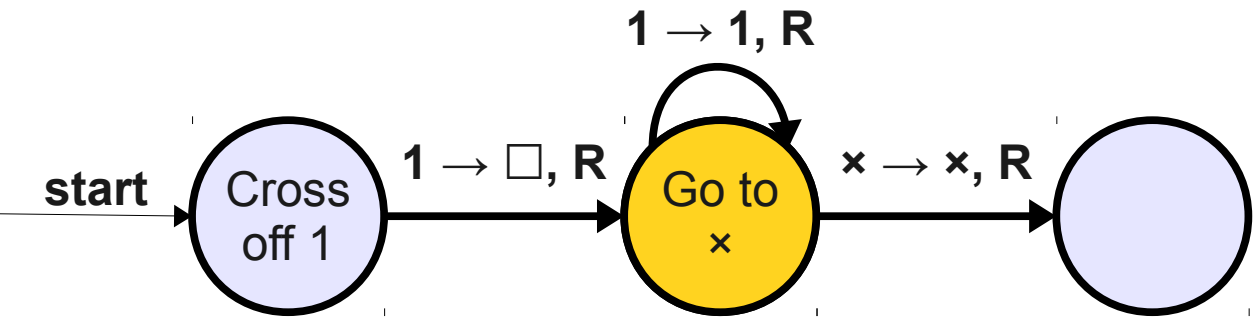

...
1 × 1 1 = 1 1 1 1
...

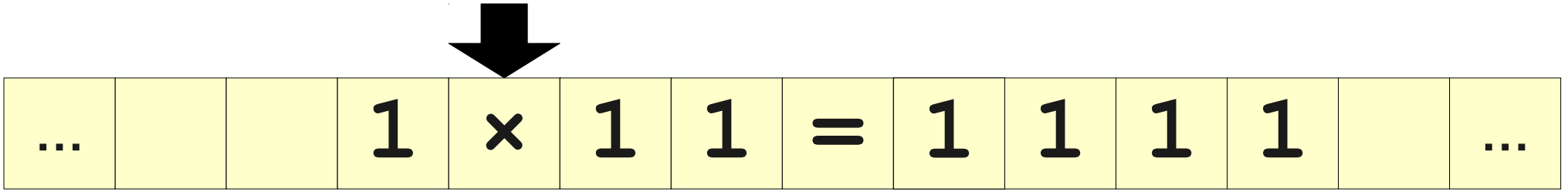

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×

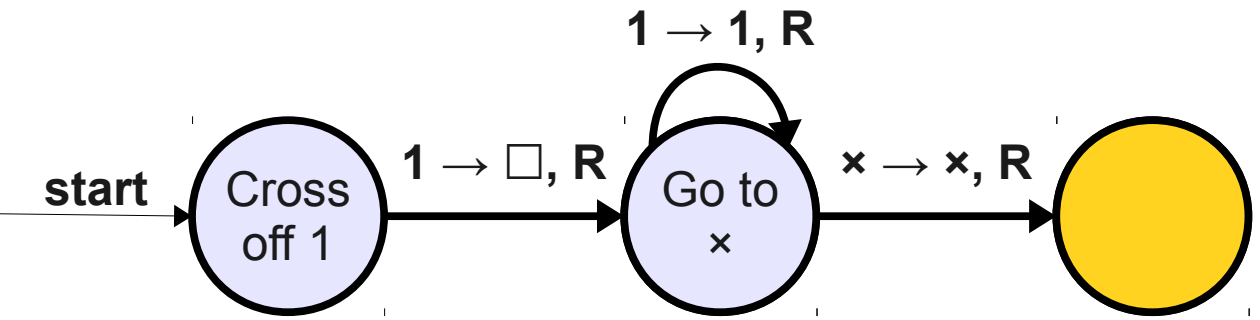

...
1 × 1 1 = 1 1 1 1
...

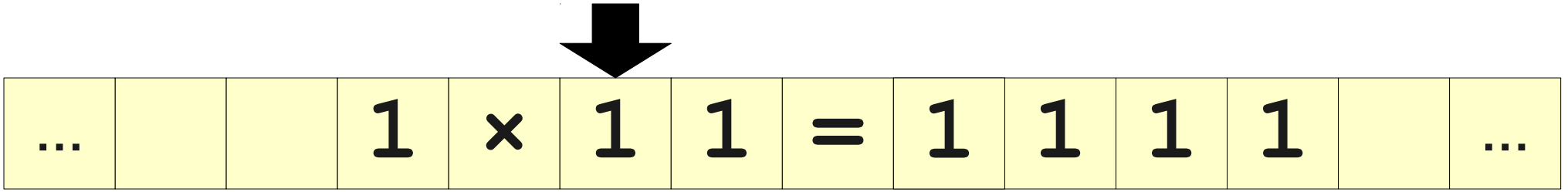

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×

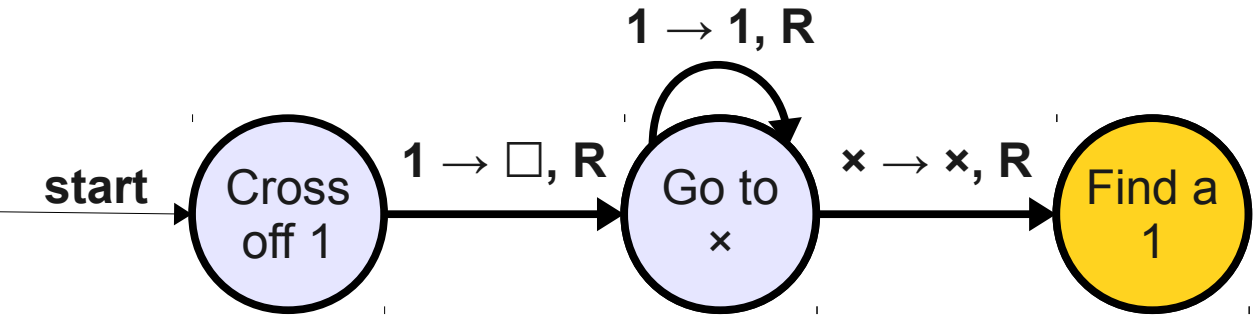

...
1 × 1 1 = 1 1 1 1
...

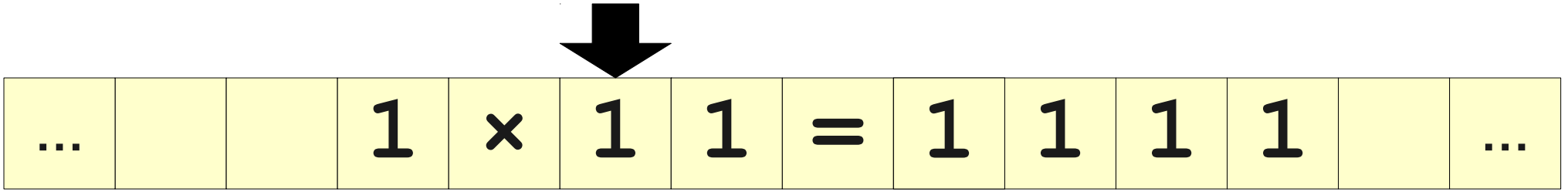

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R

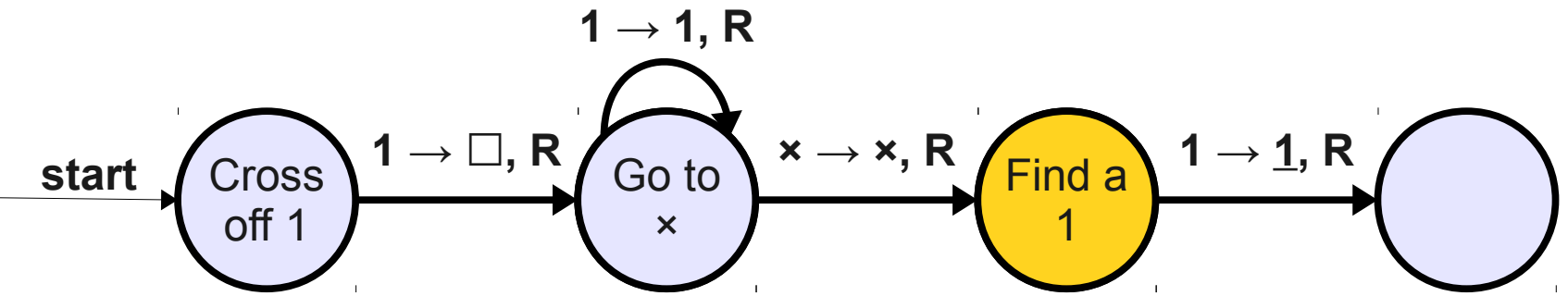

...
1
×
1
1
=
1
1
1
1
...

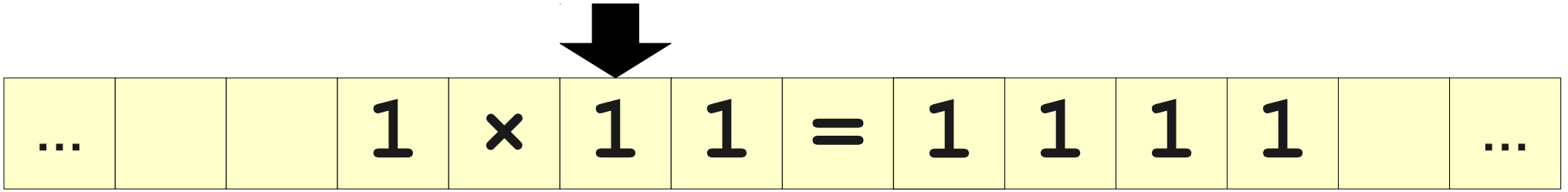

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R

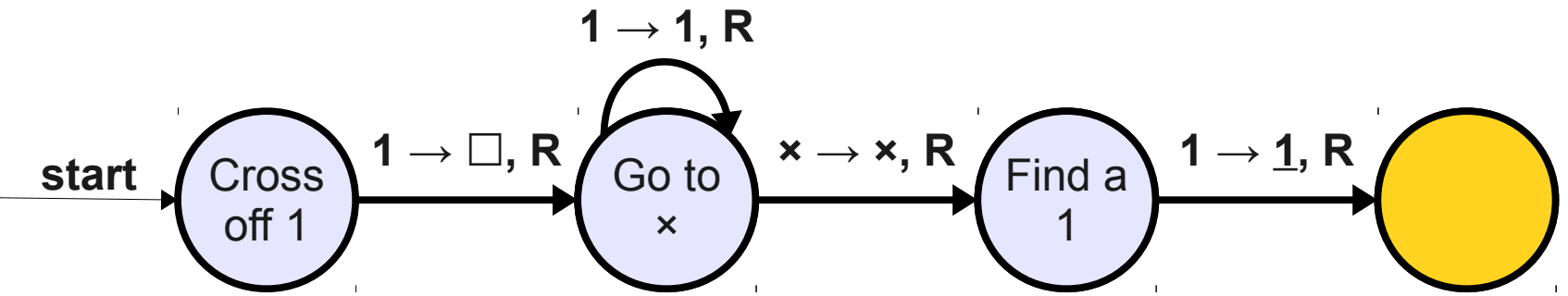

...
1
×
1
1
=
1
1
1
1
...

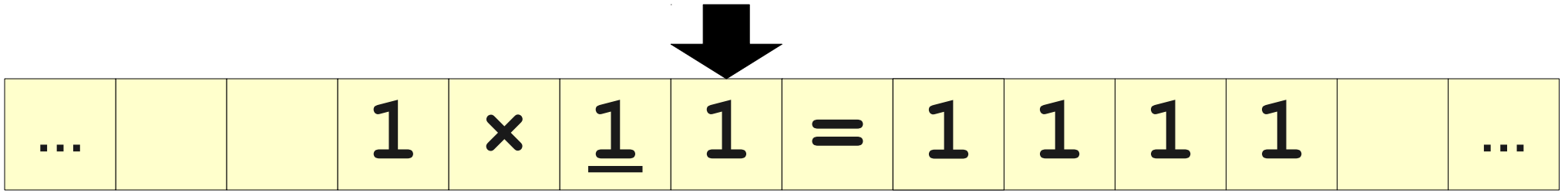

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1

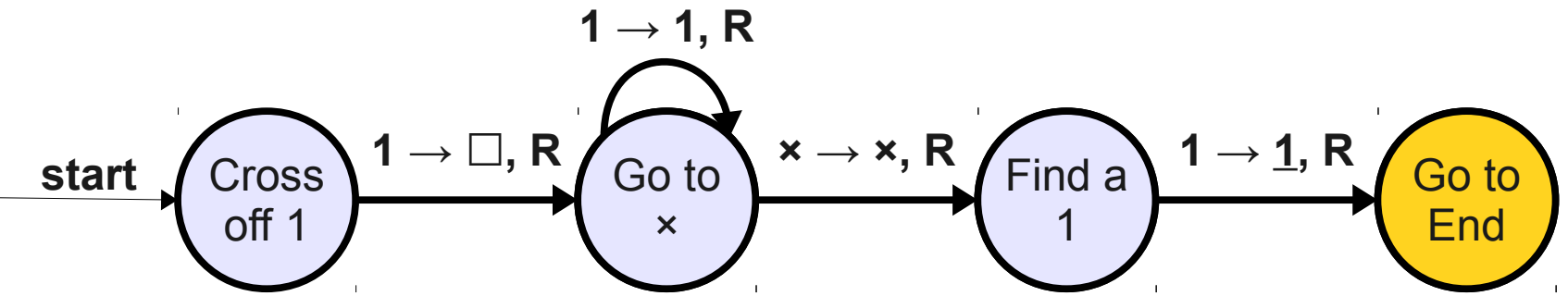

...
1 × 1 1 = 1 1 1 1
...

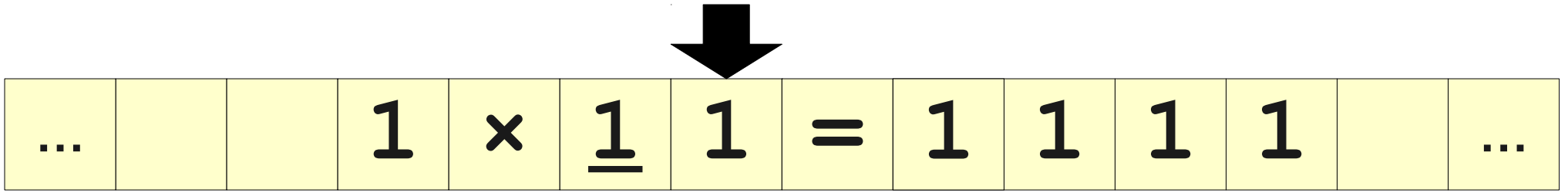

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R

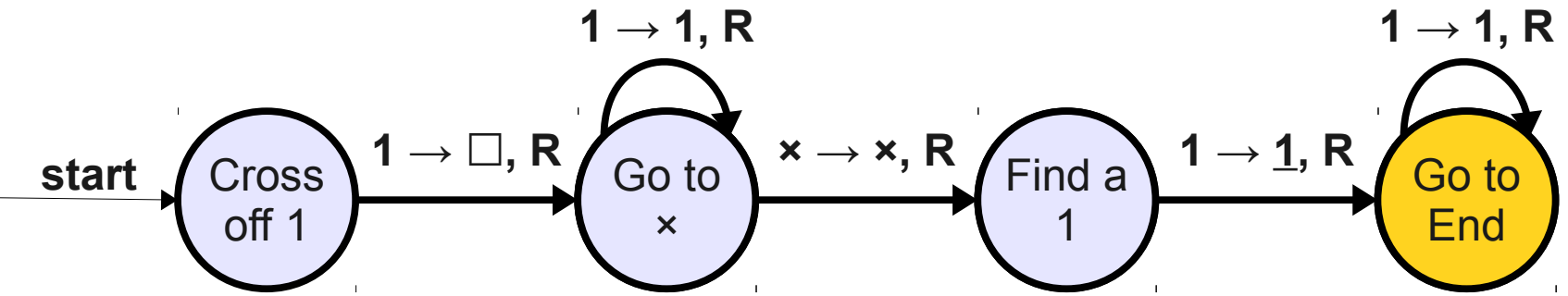

...
1 × 1 1 = 1 1 1 1
...

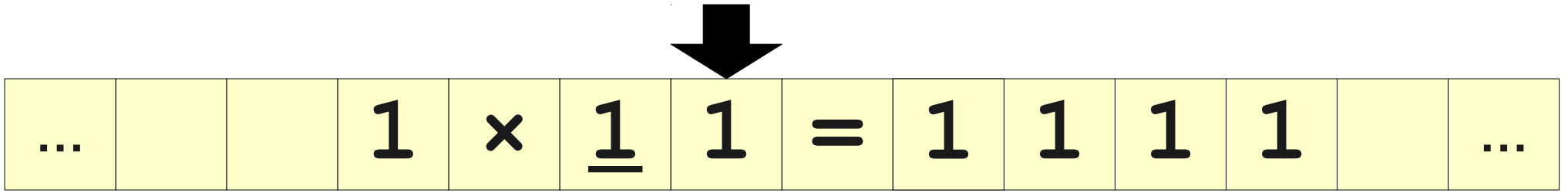

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R

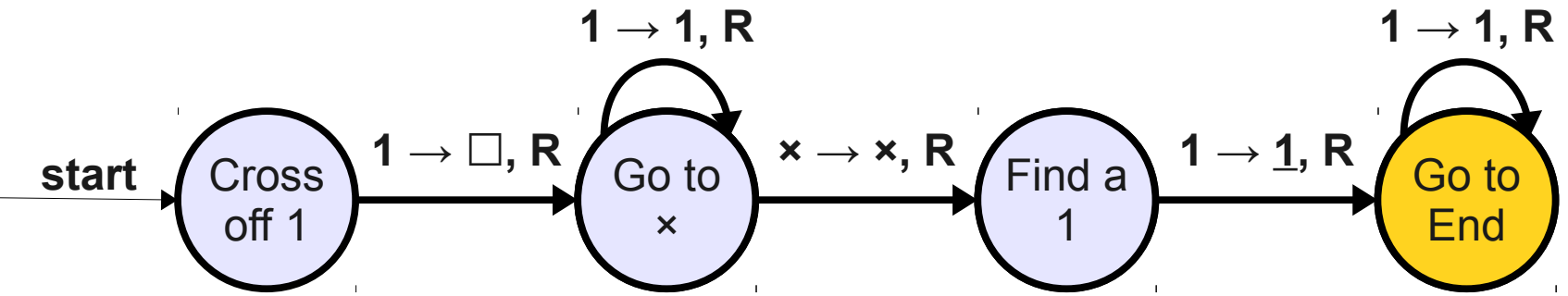

...
1 × 1 1 = 1 1 1 1
...

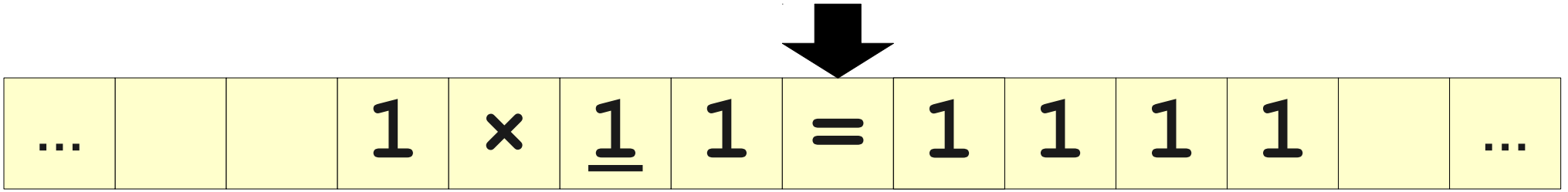

1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End

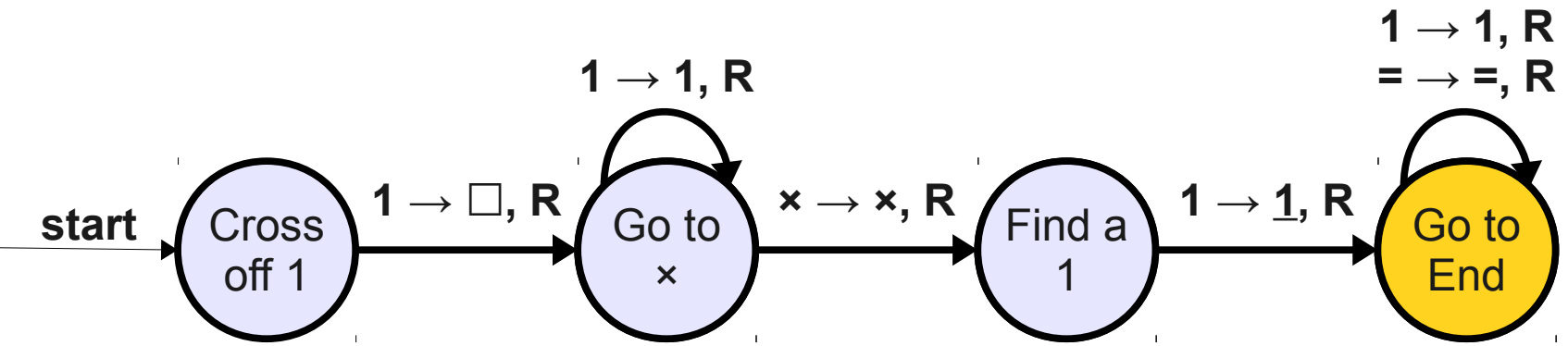

...
1 × 1 1 = 1 1 1 1
...

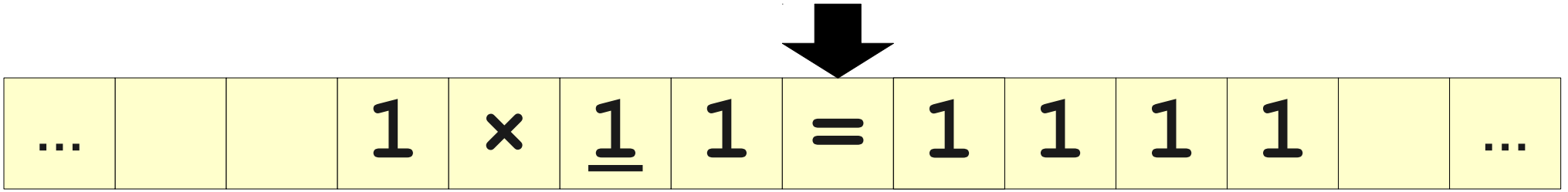

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R

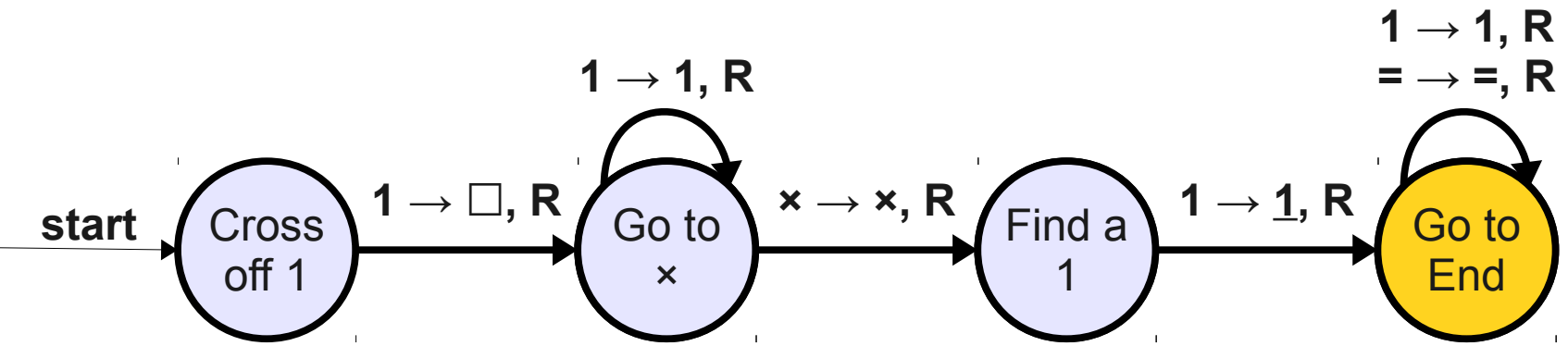

...
1 × 1 1 = 1 1 1 1
...

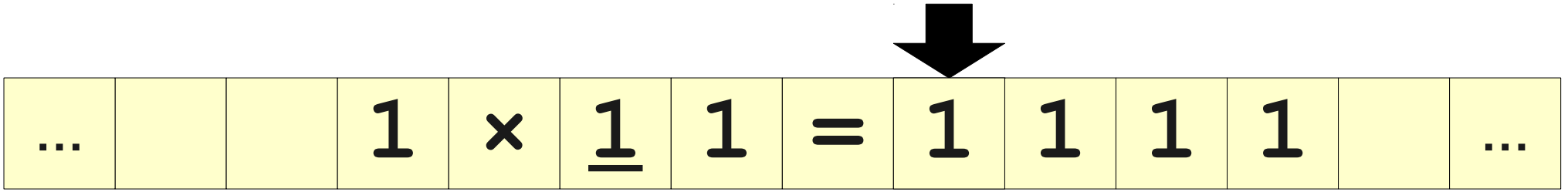

1 → 1, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End

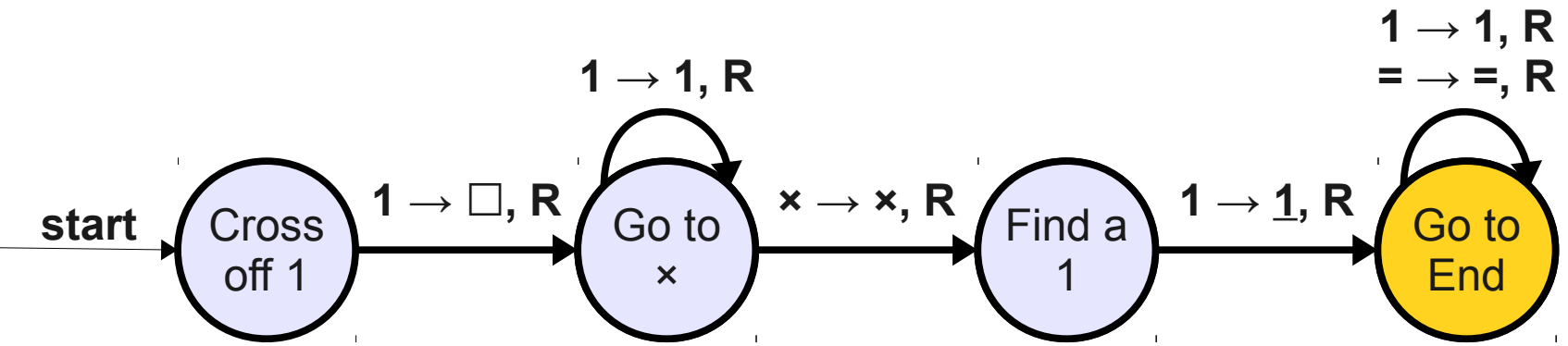

...
1 × 1 1 = 1 1 1 1
...

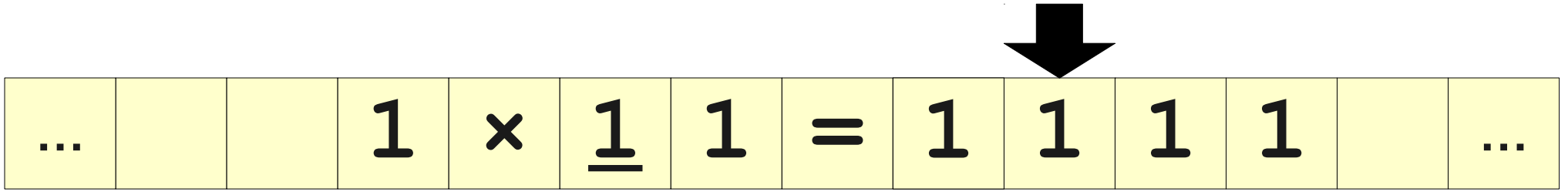

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

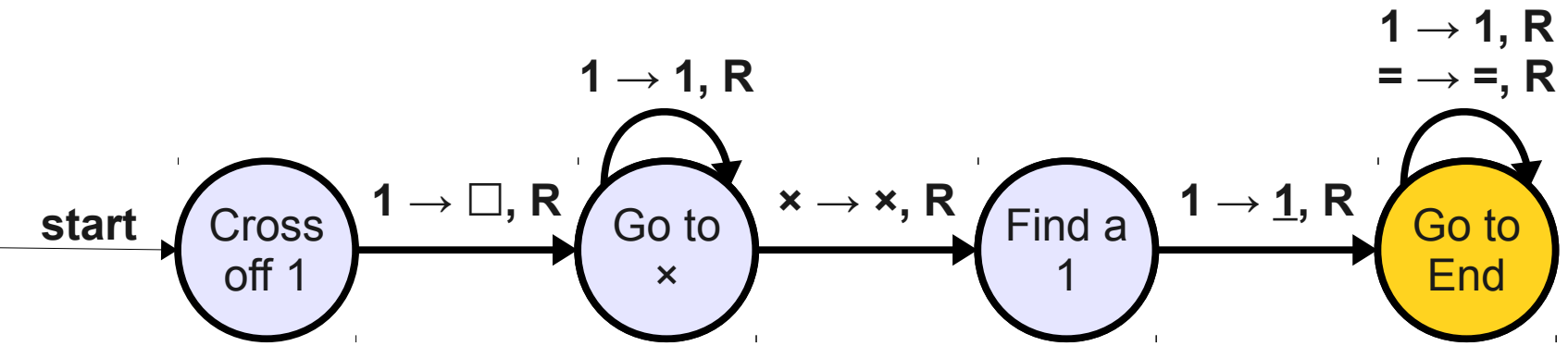

...
1
×
1
1
=
1
1
1
1
...

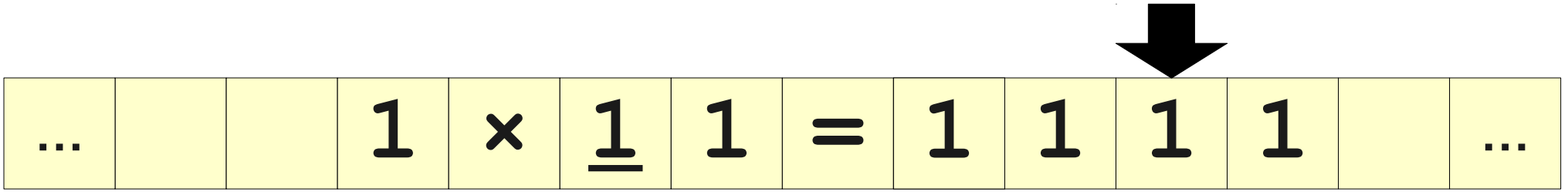

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

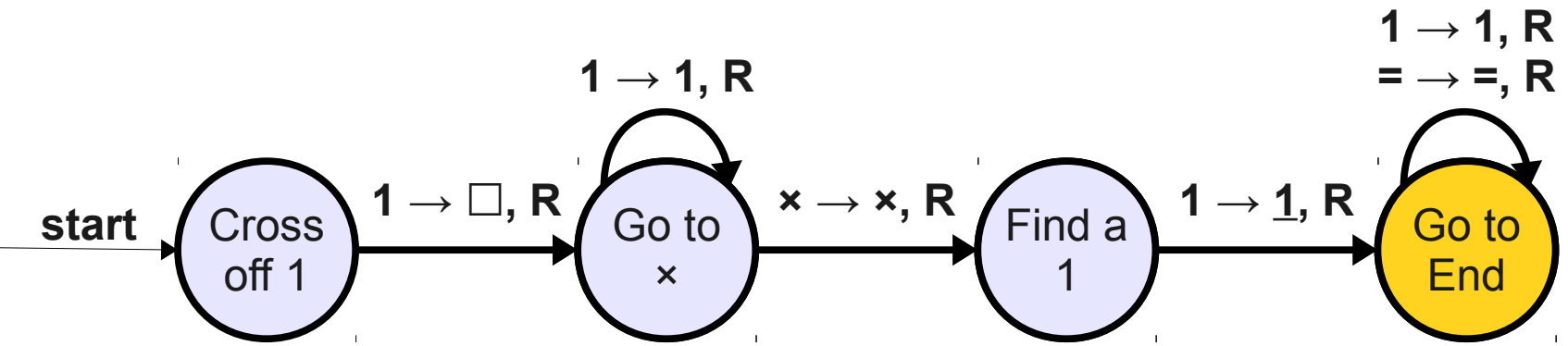

...
1 × 1 1 = 1 1 1 1
...

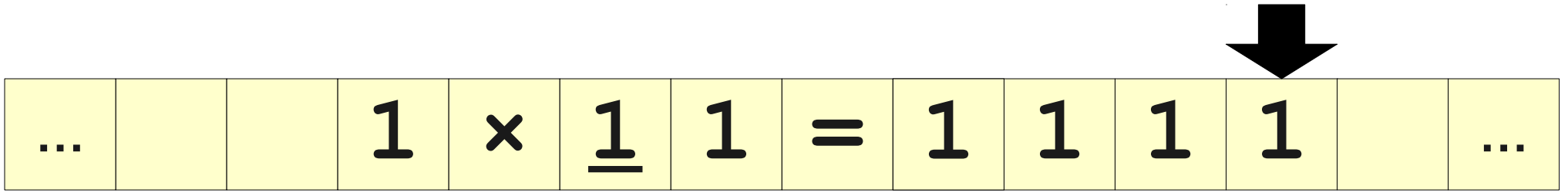

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

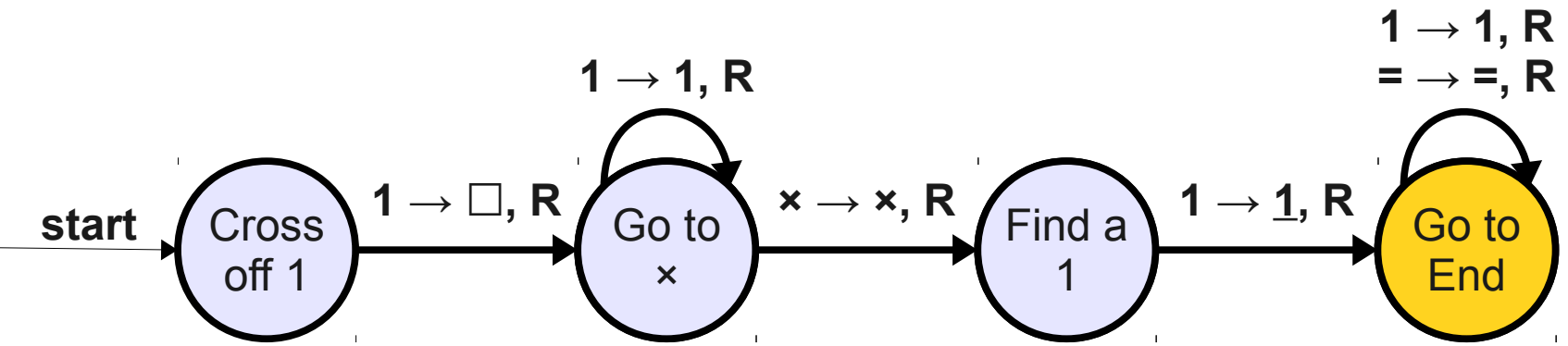

...
1 × 1 1 = 1 1 1 1
...

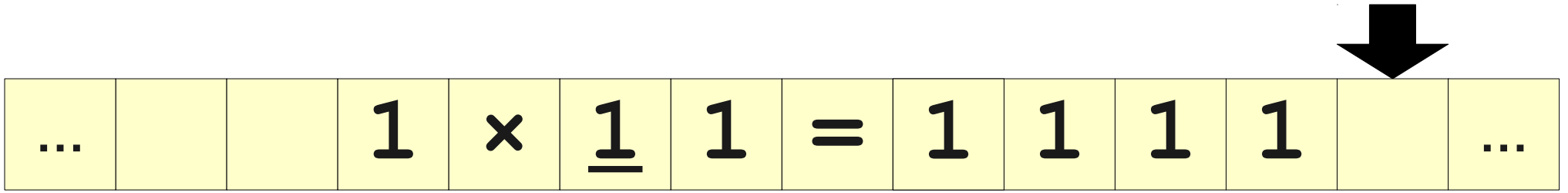

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

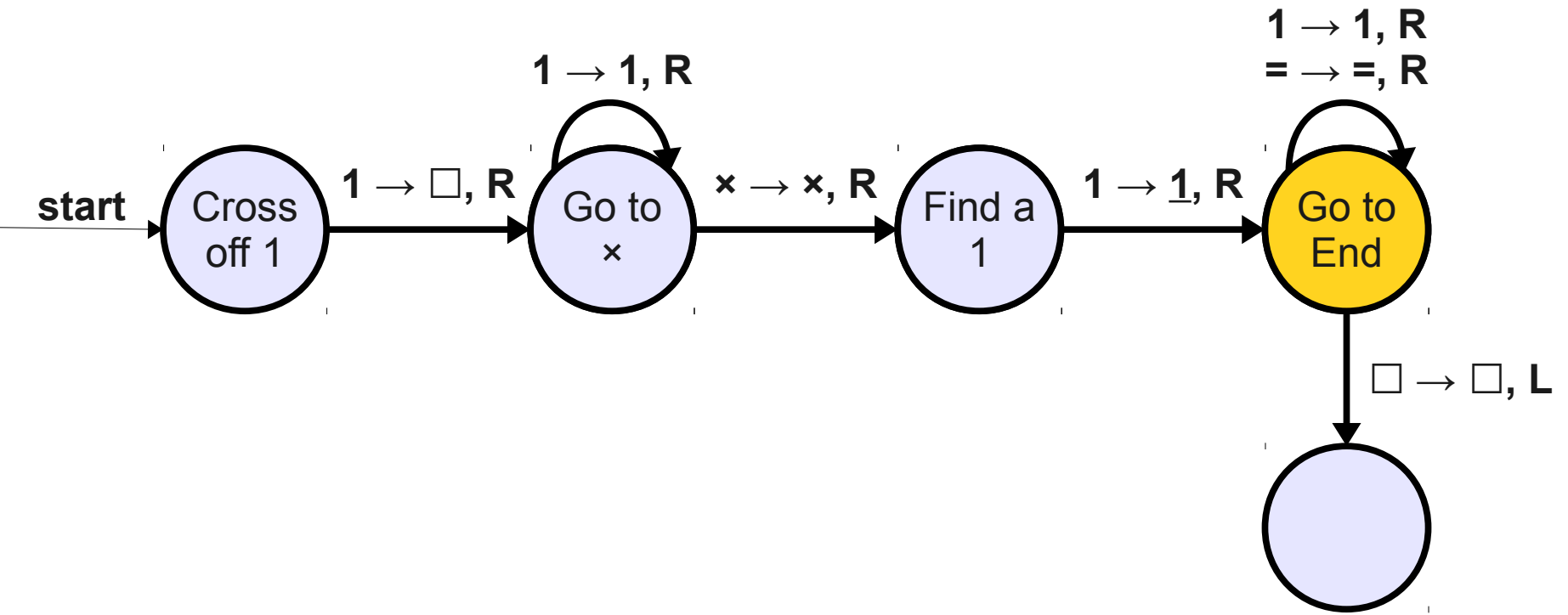

...
1
×
1
1
=
1
1
1
1
...

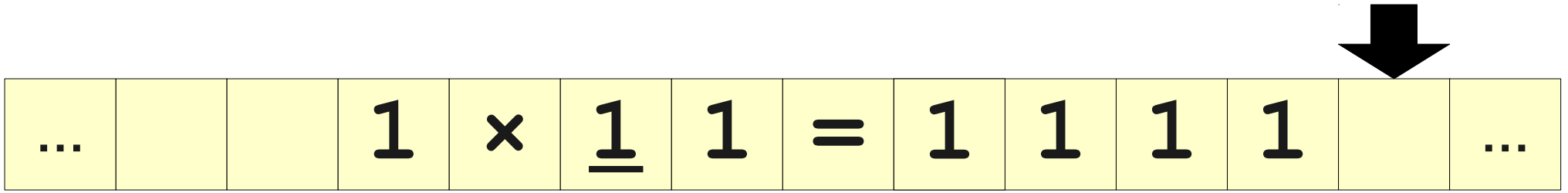

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

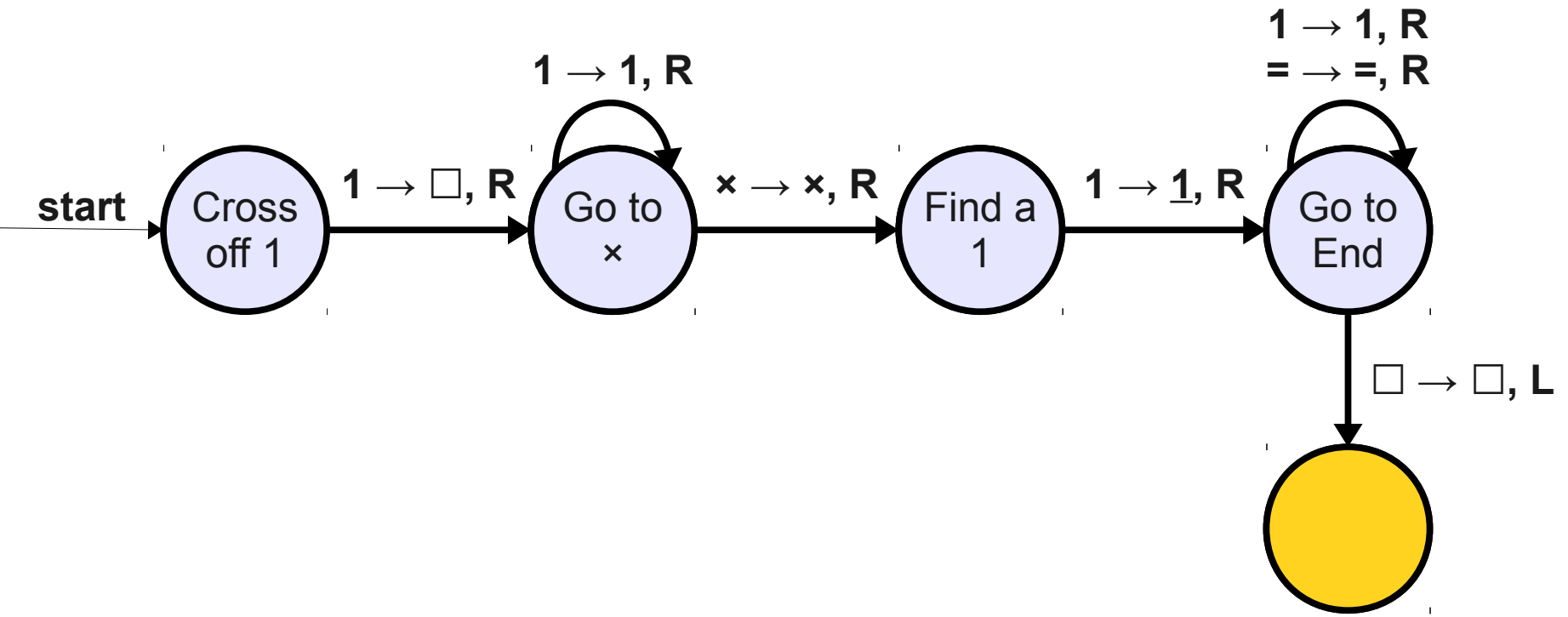

...
1 × 1 1 = 1 1 1 1
...

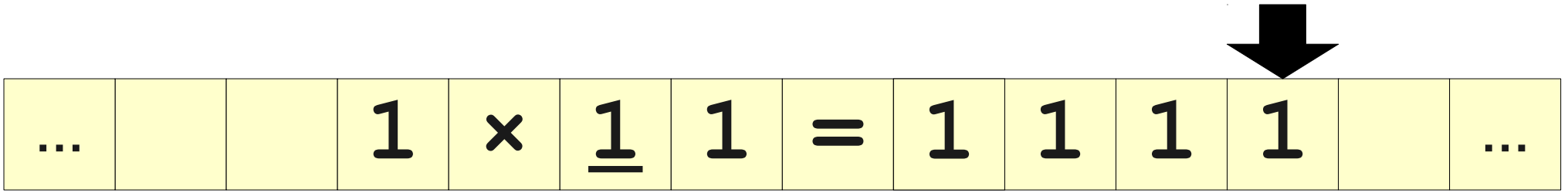

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R

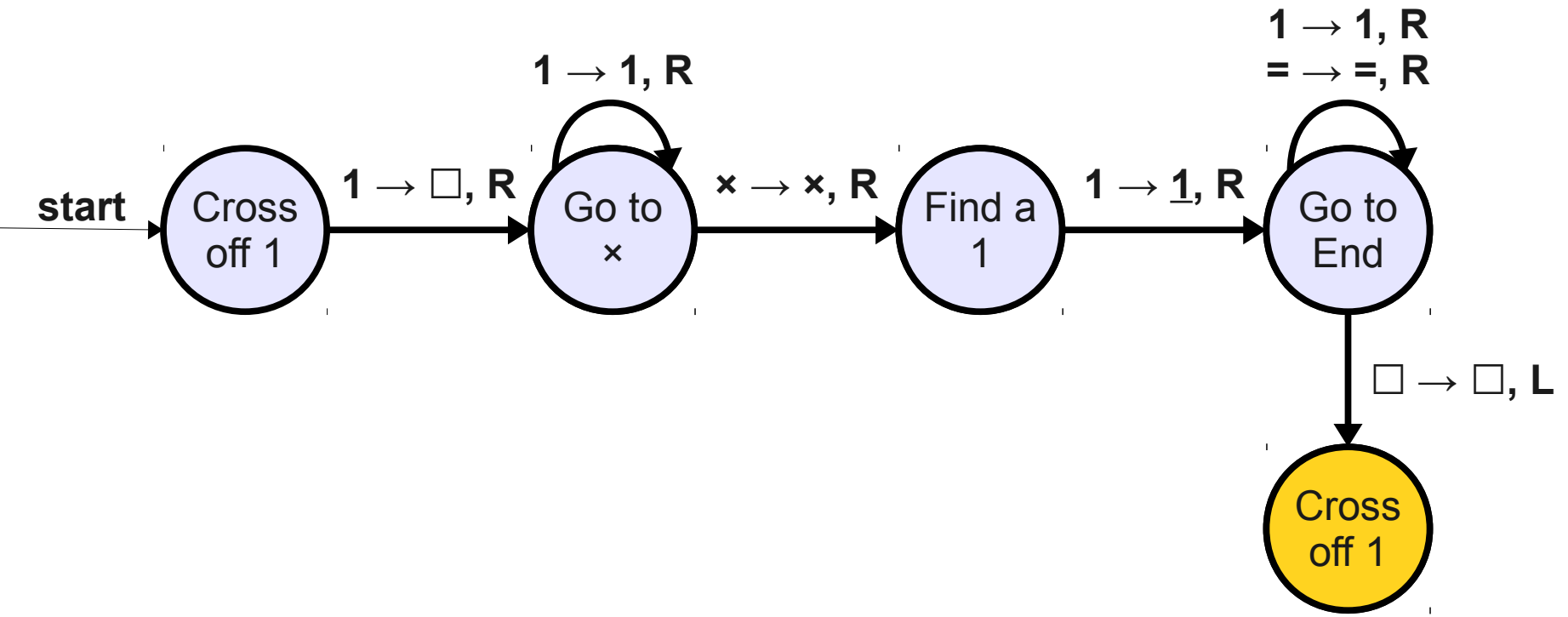

...
1 × 1 1 = 1 1 1 1
...

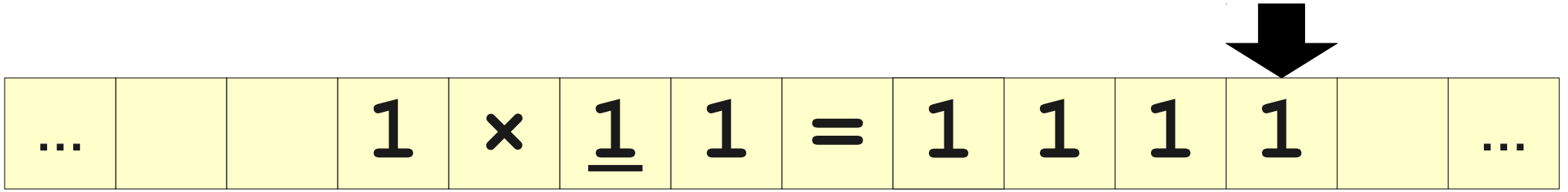

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L

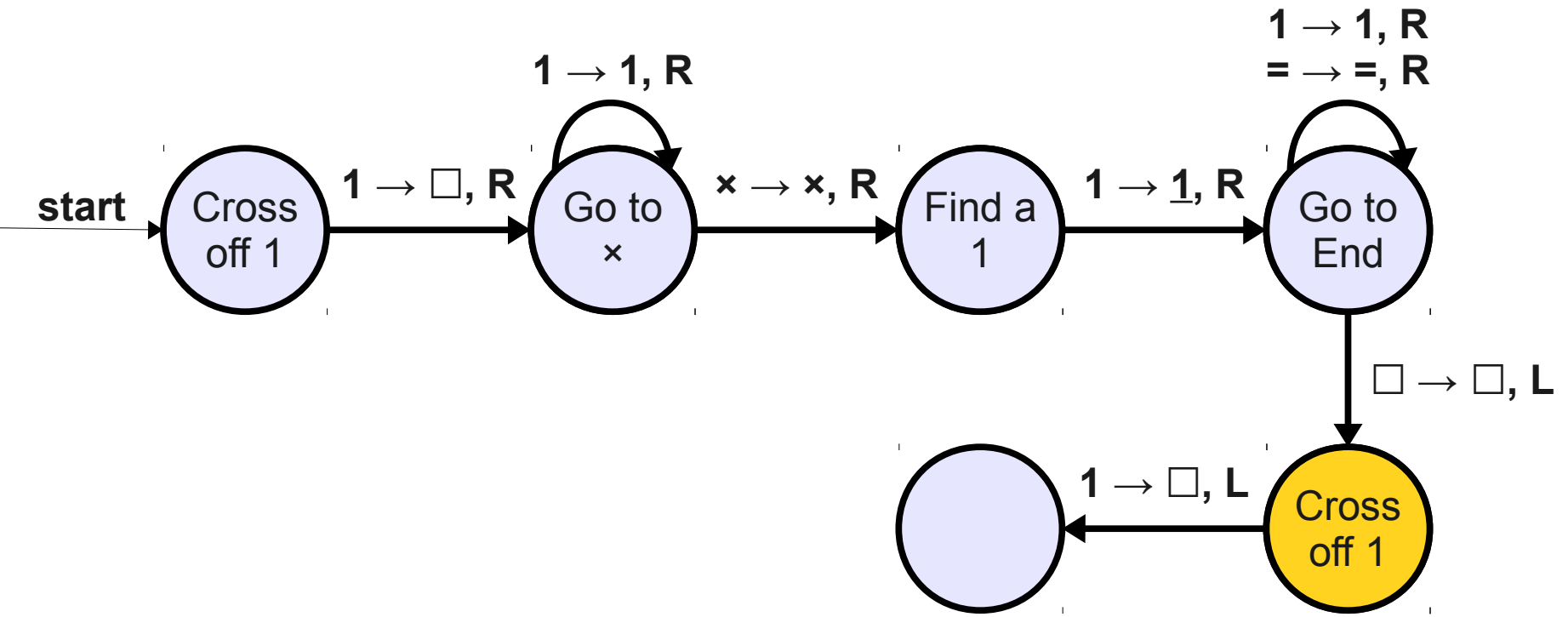

...
1 × 1 1 = 1 1 1 1
...

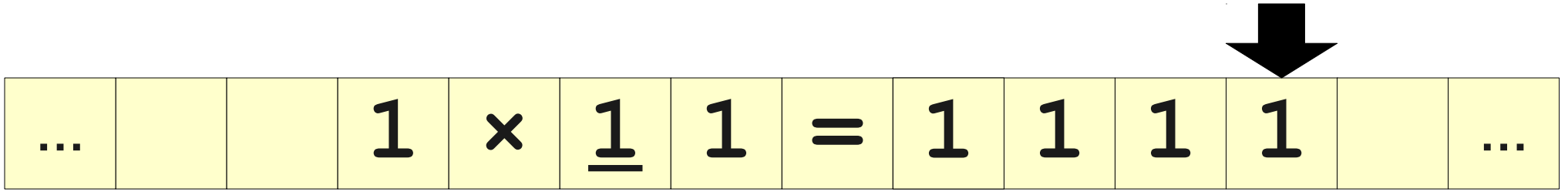

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L

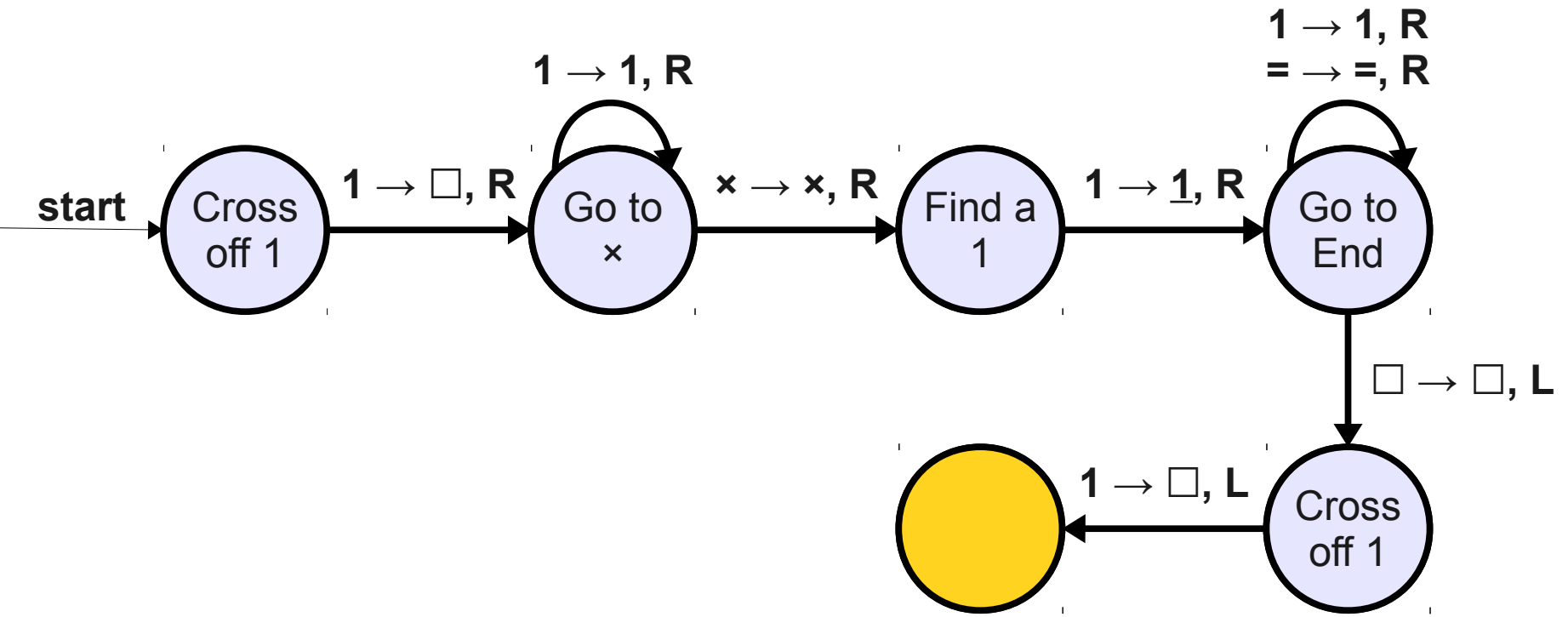

...
1 × 1 1 = 1 1 1 1
...

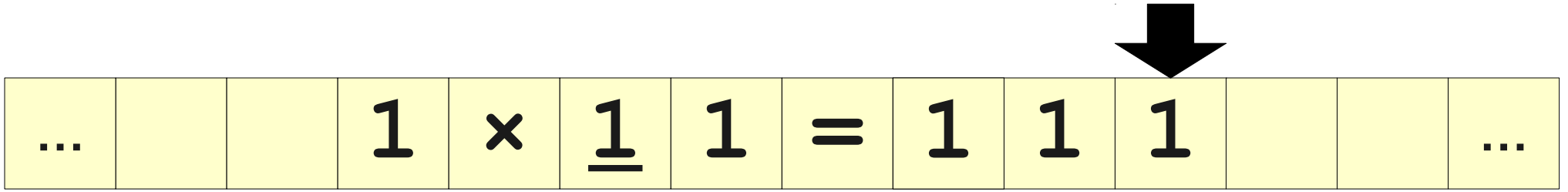

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1

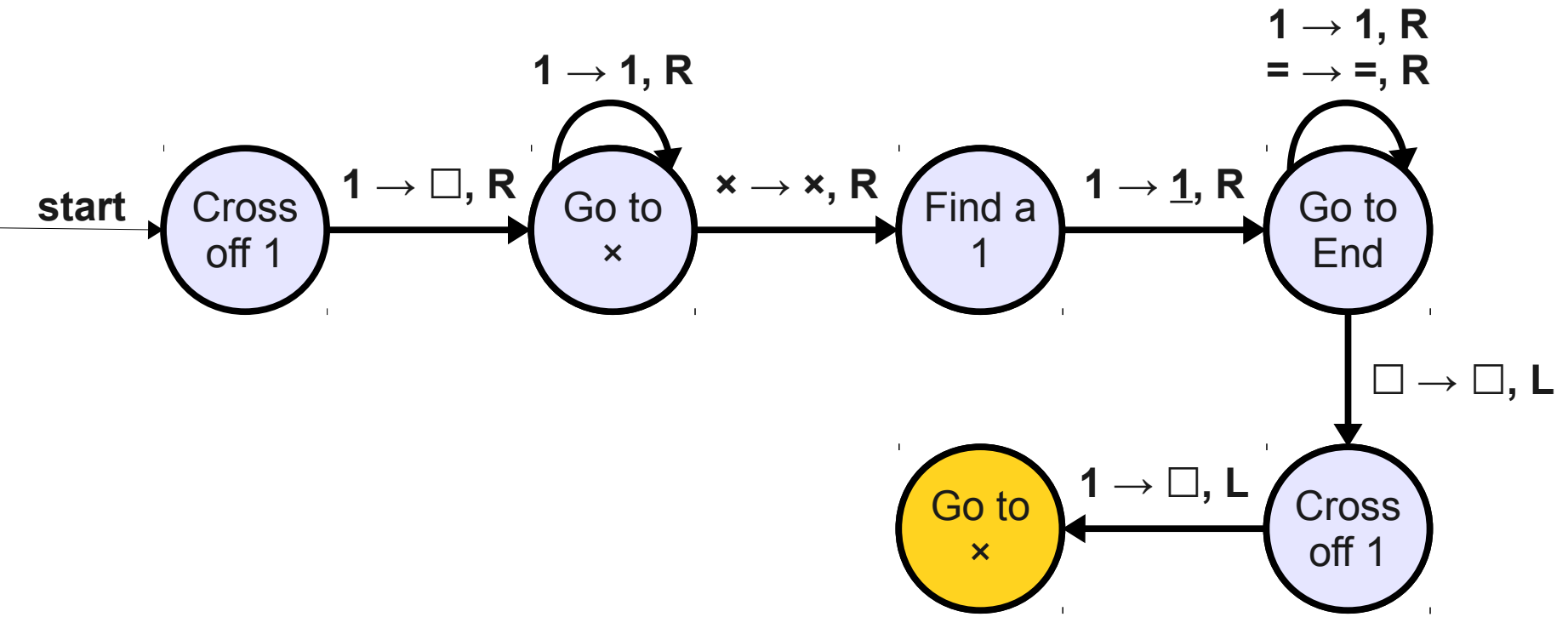

...
1 × 1 1 = 1 1 1 1
...

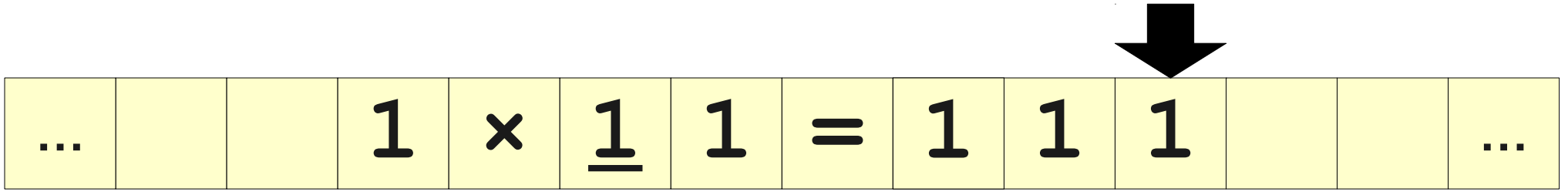

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L

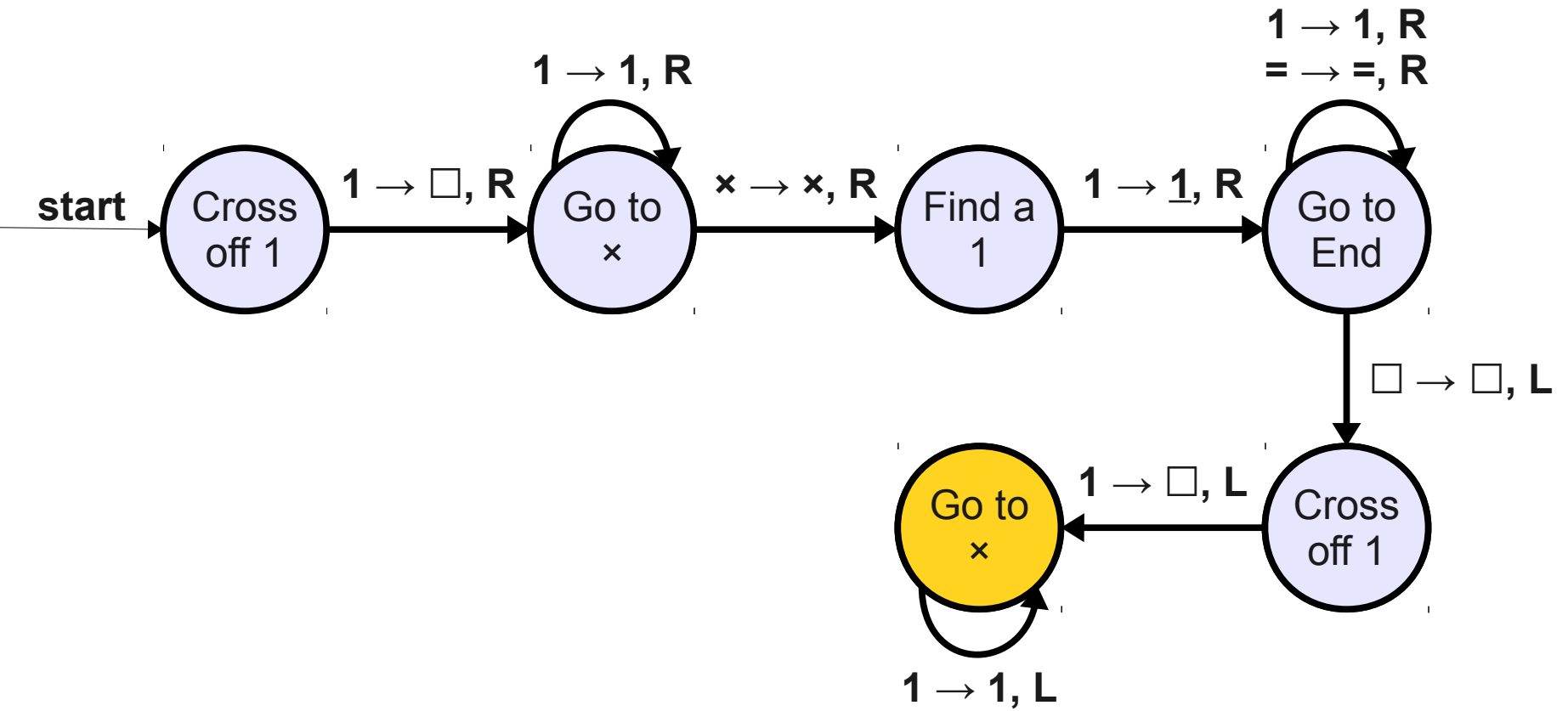

...
1 × 1 1 = 1 1 1 1
...

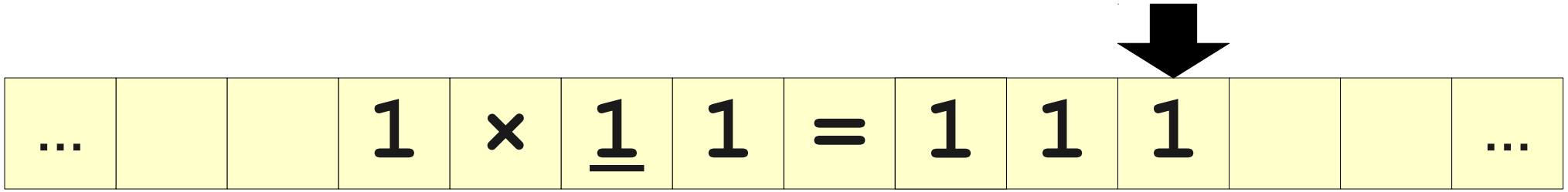

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L

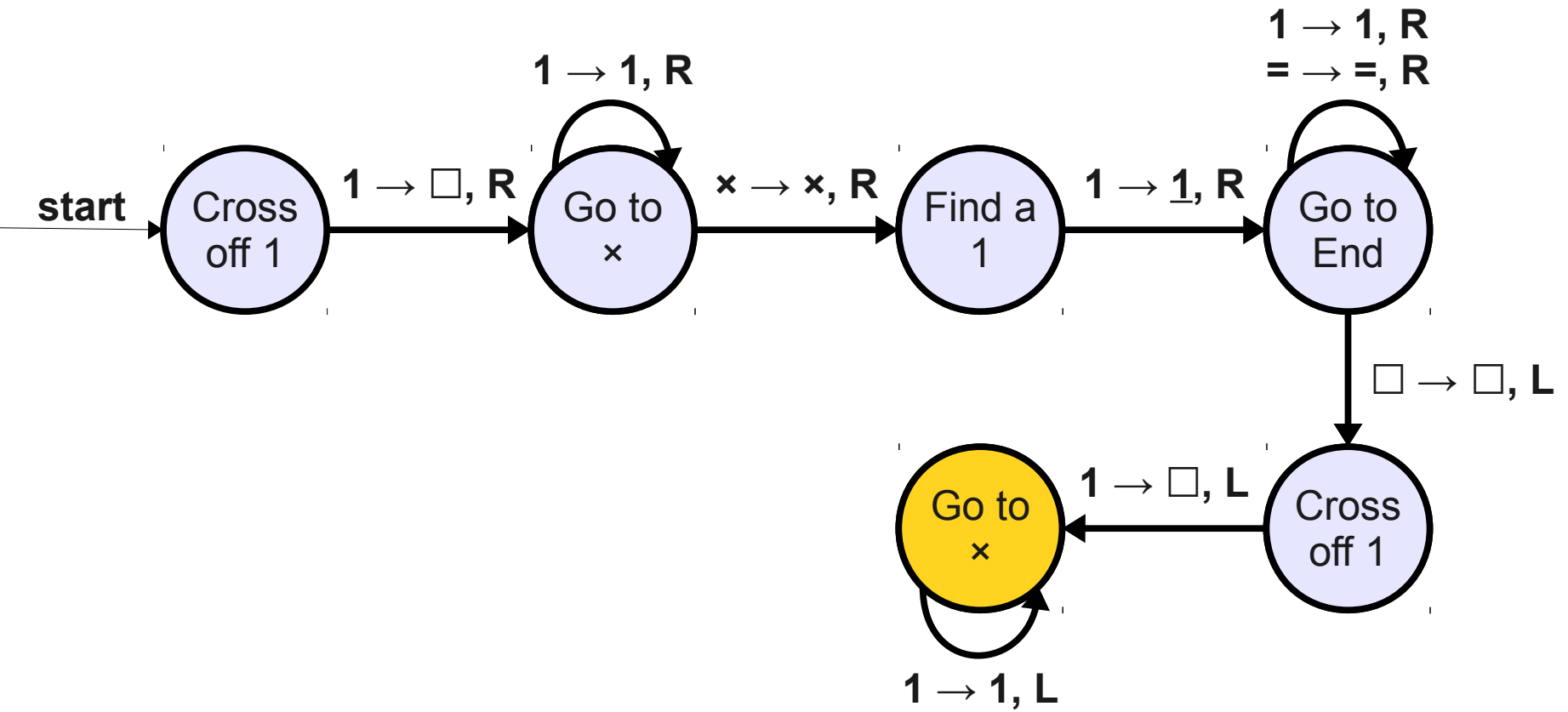

...
1 × 1 1 = 1 1 1
...

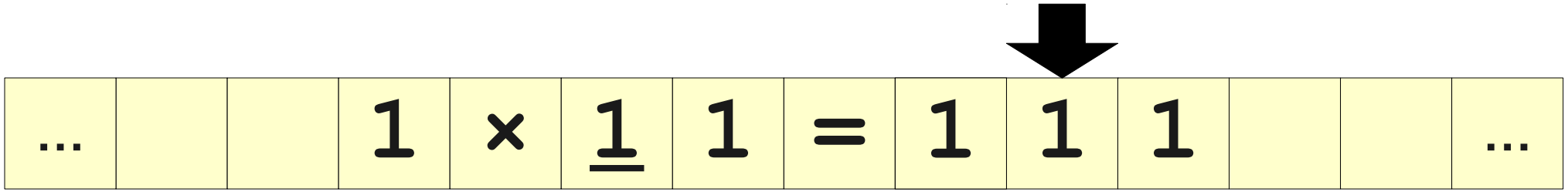

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×

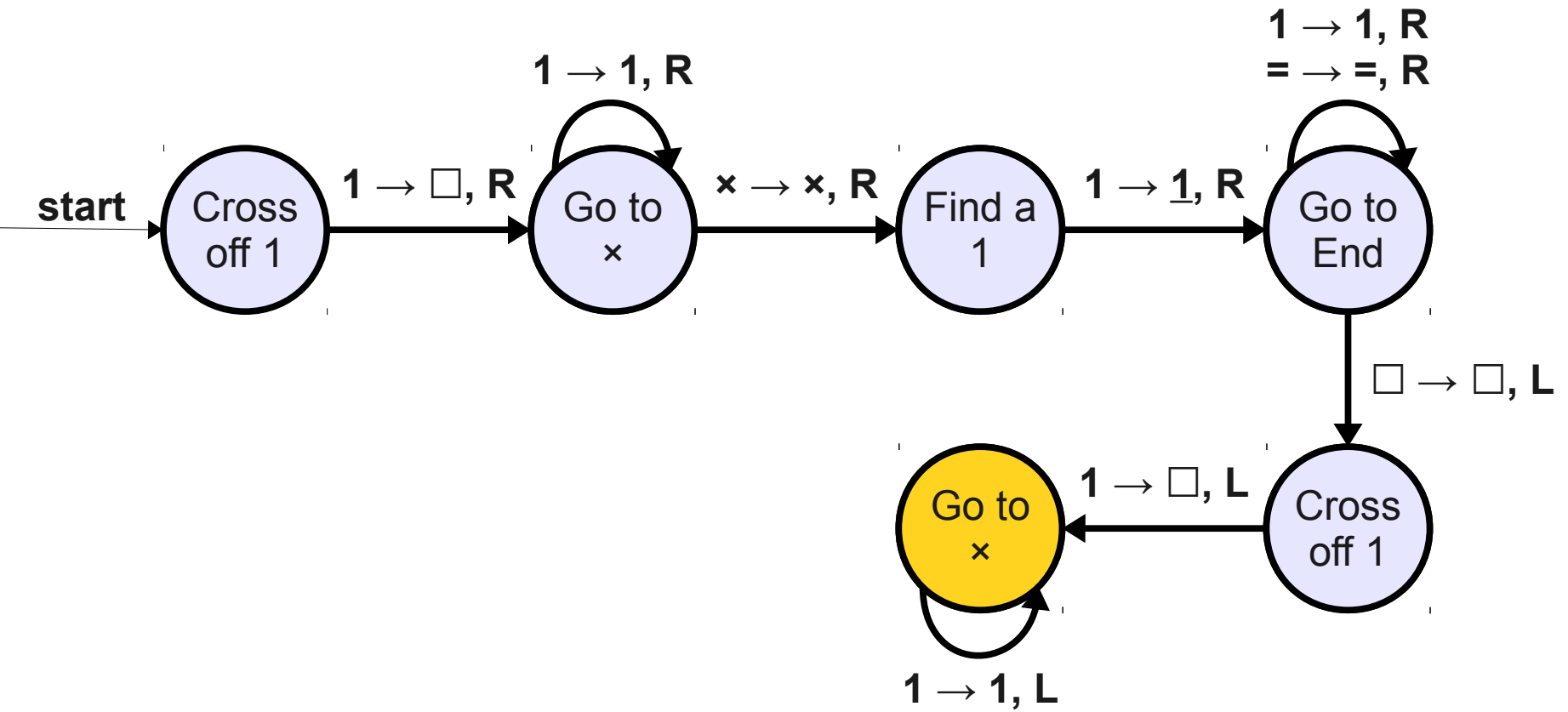

... 1 × 1 1 = 1 1 1 ...

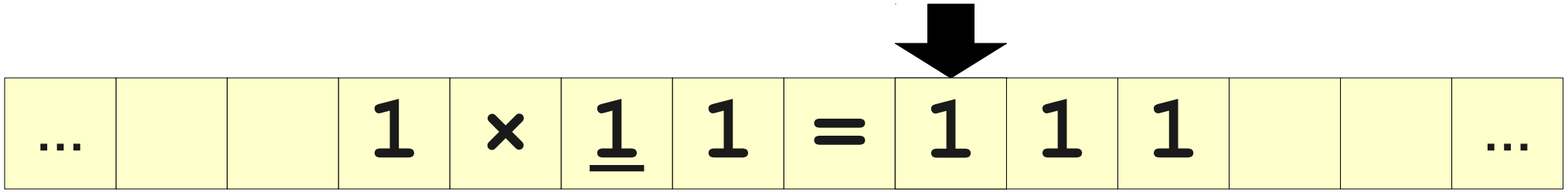

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L

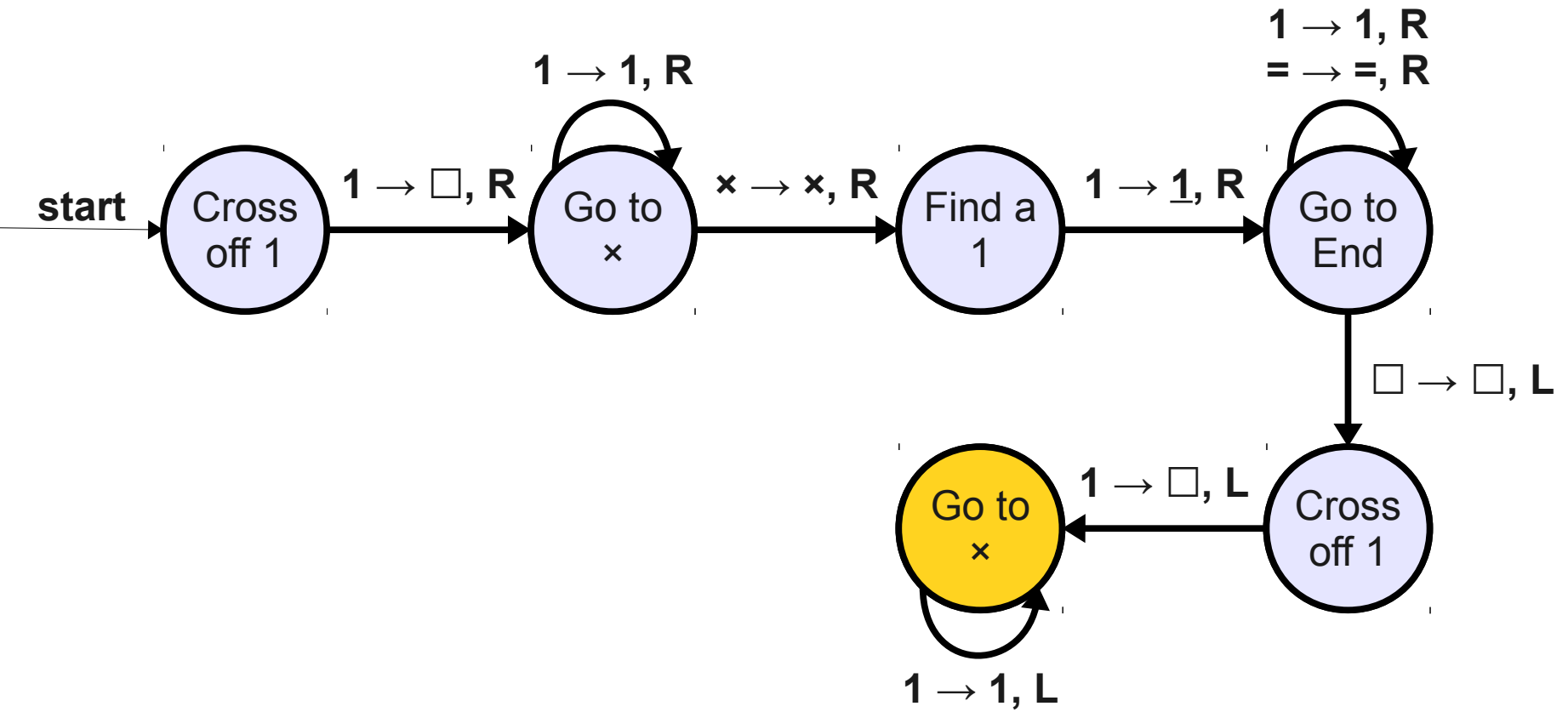

... 1 × 1 1 = 1 1 1 ...

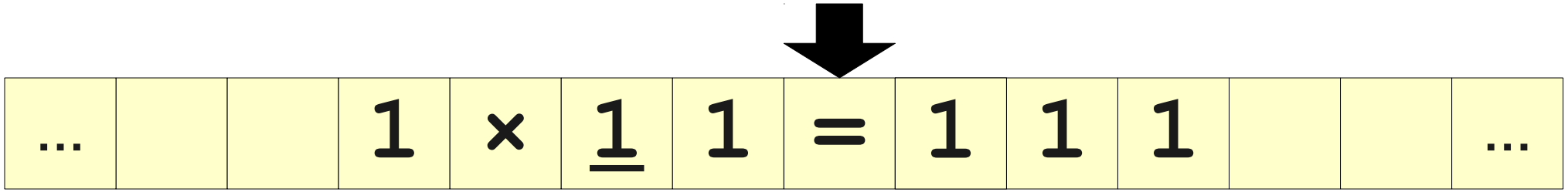

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L

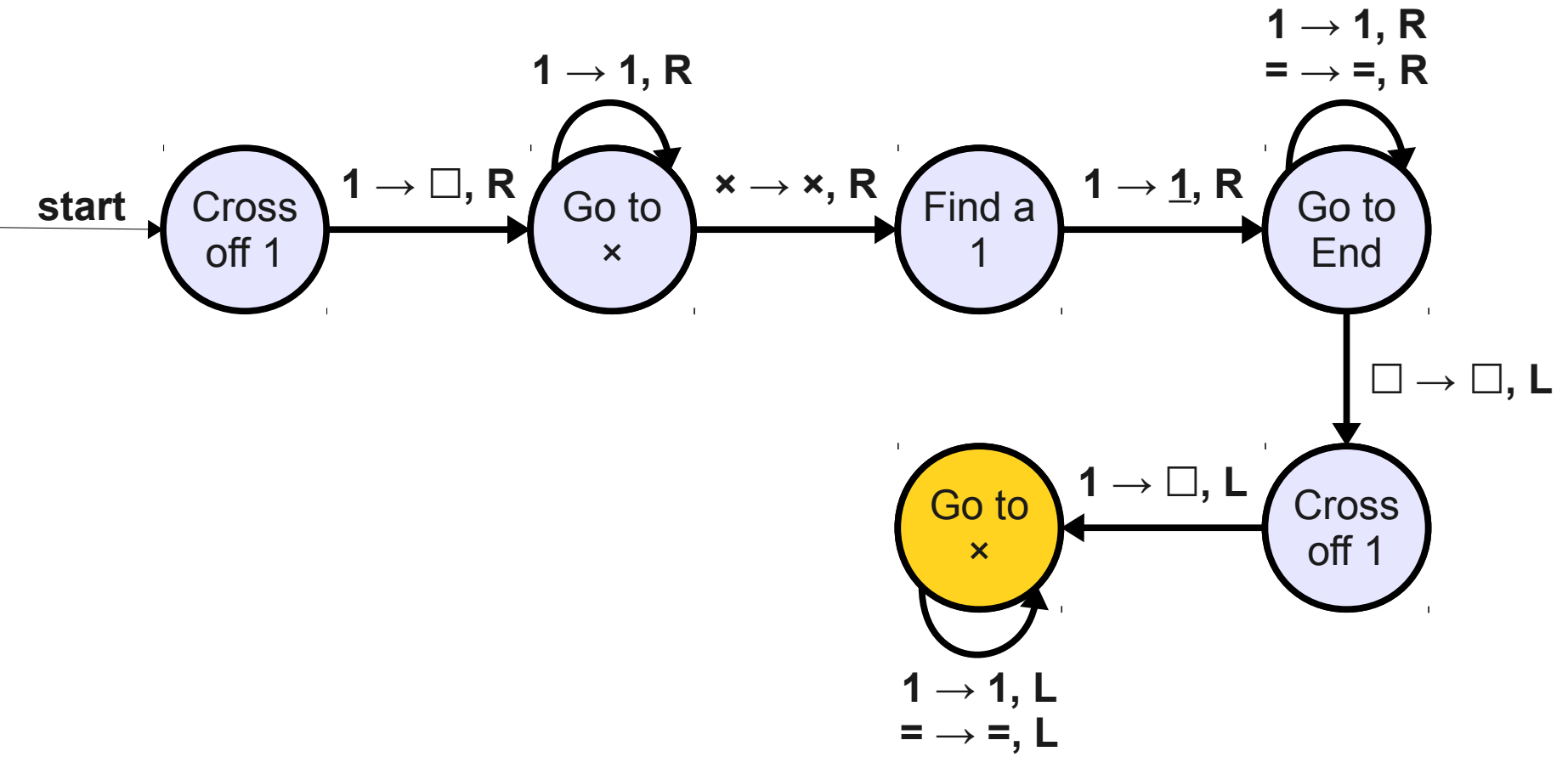

... 1 × 1 1 = 1 1 1 ...

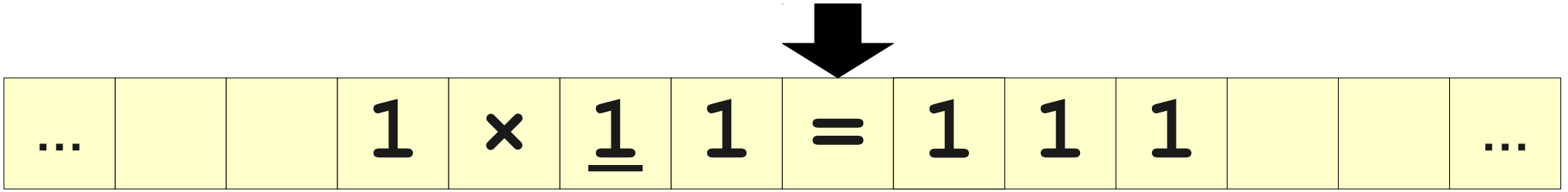

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L

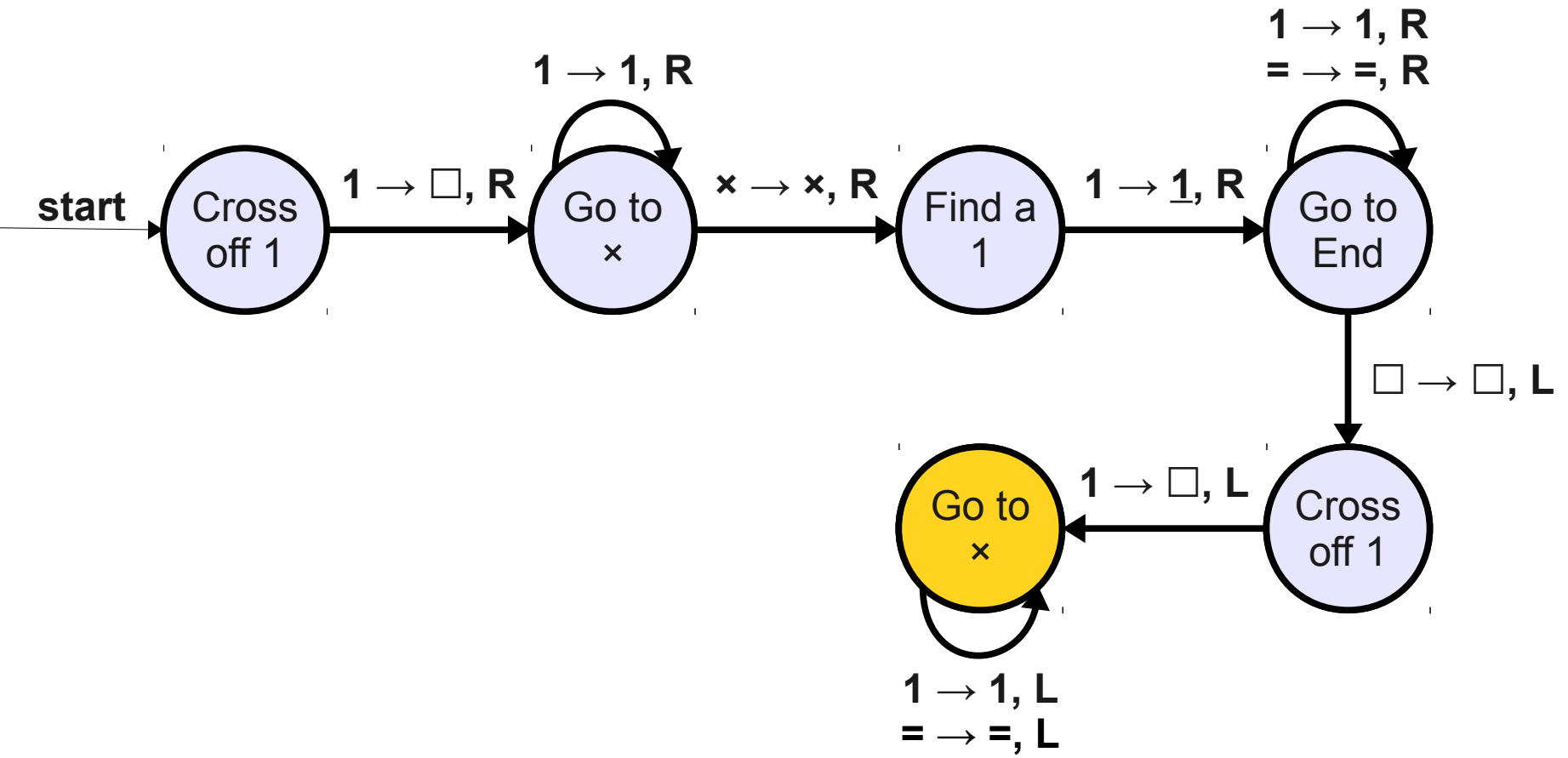

...
1 × 1 1 = 1 1 1
...

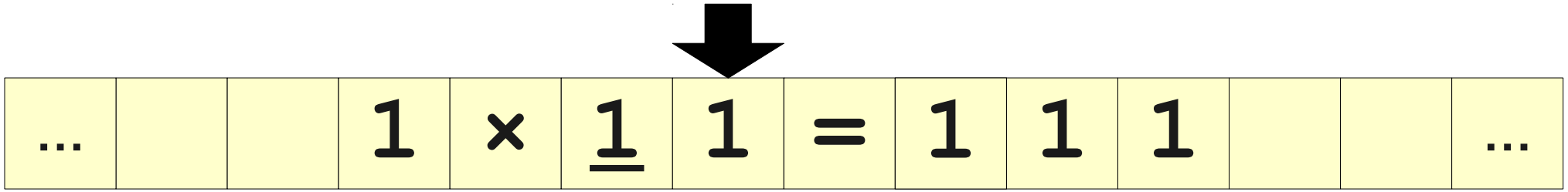

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L

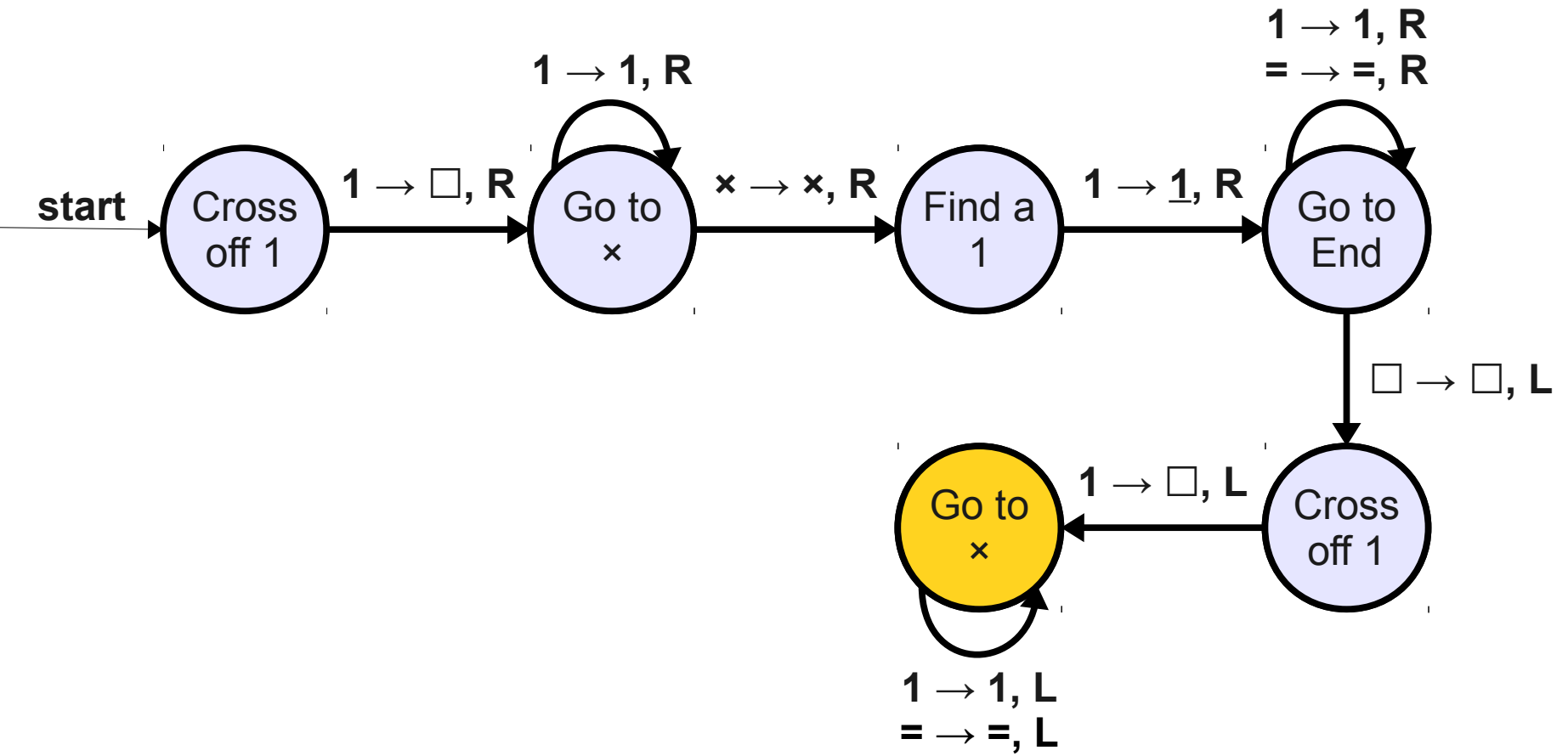

...
1 × 1 1 = 1 1 1
...

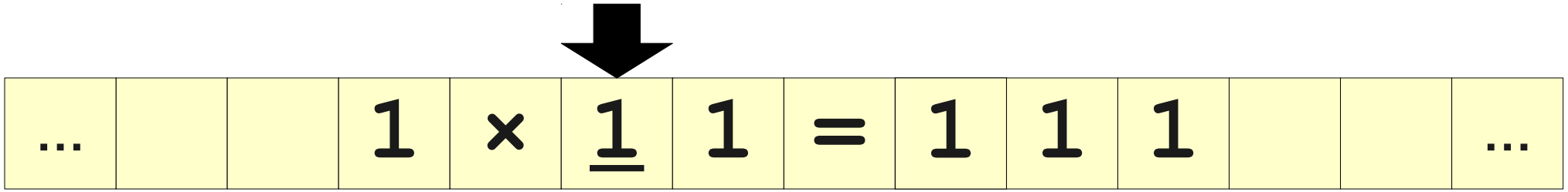

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L
= → =, L

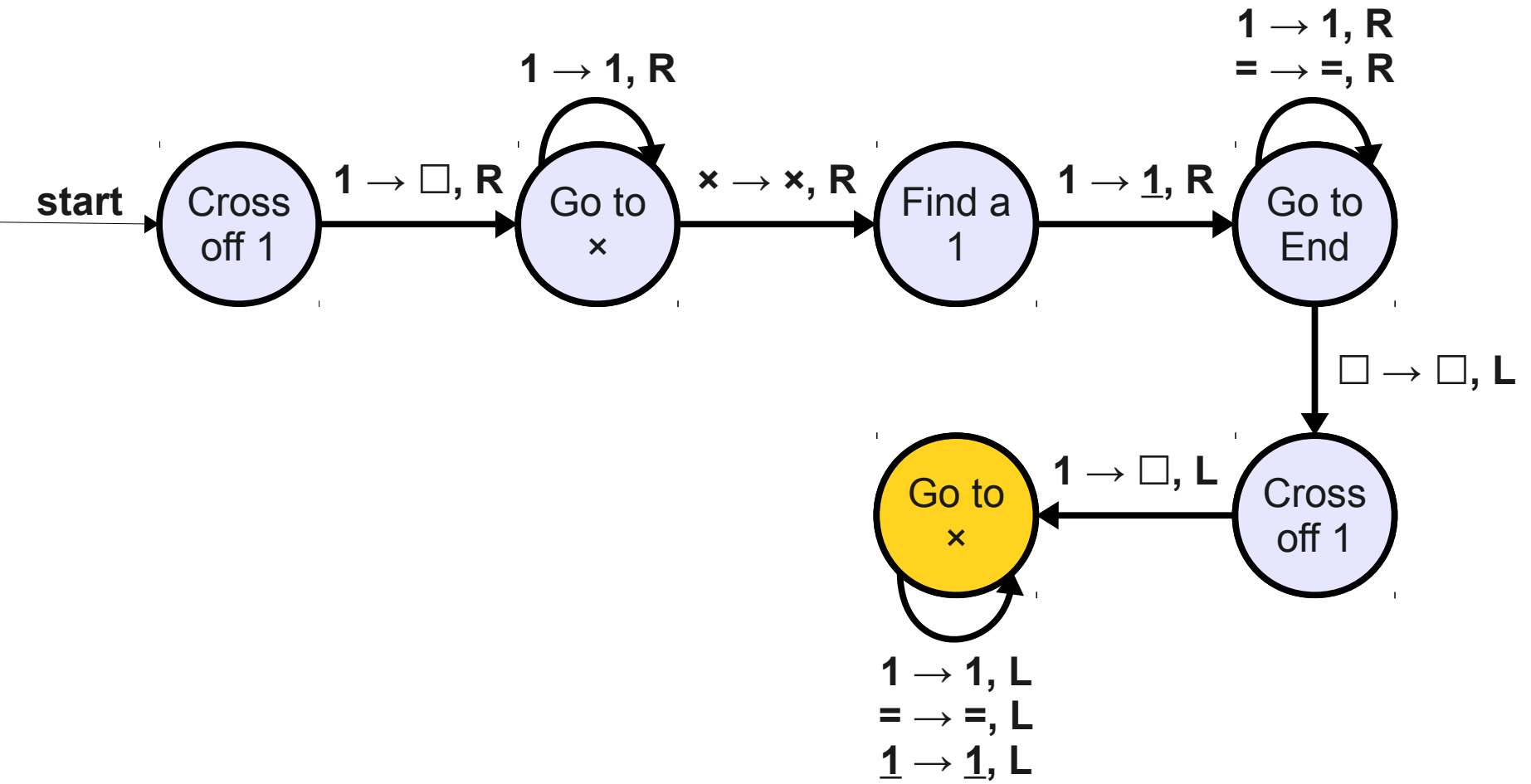

...
1 × 1 1 = 1 1 1
...

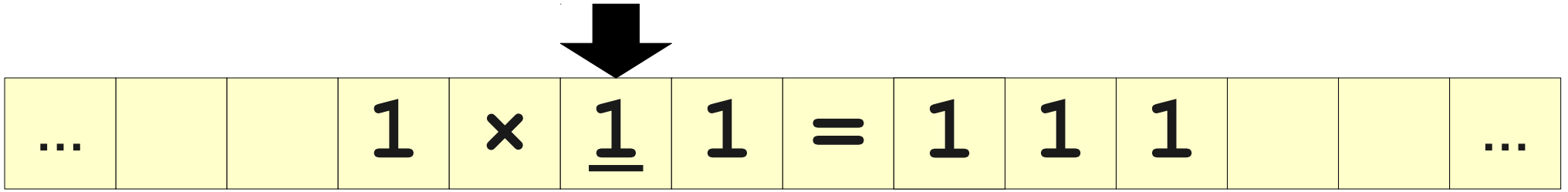

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L
= → =, L

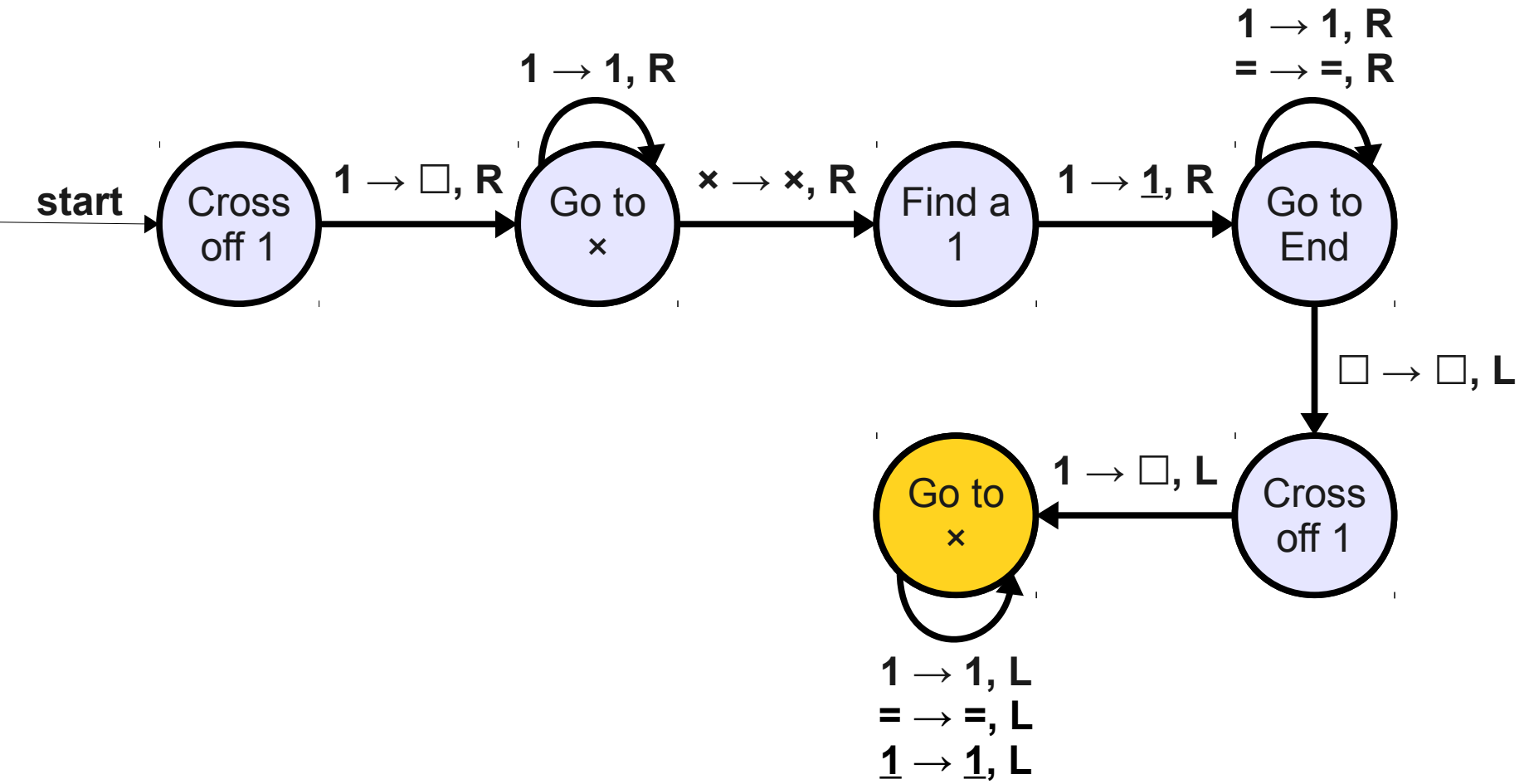

...
1 × 1 1 = 1 1 1
...

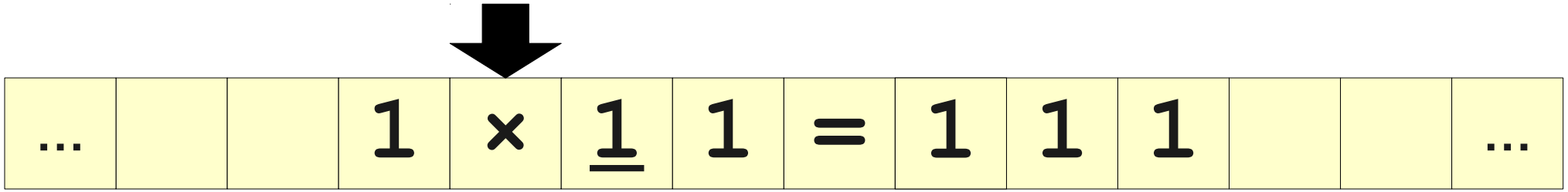

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L
= → =, L

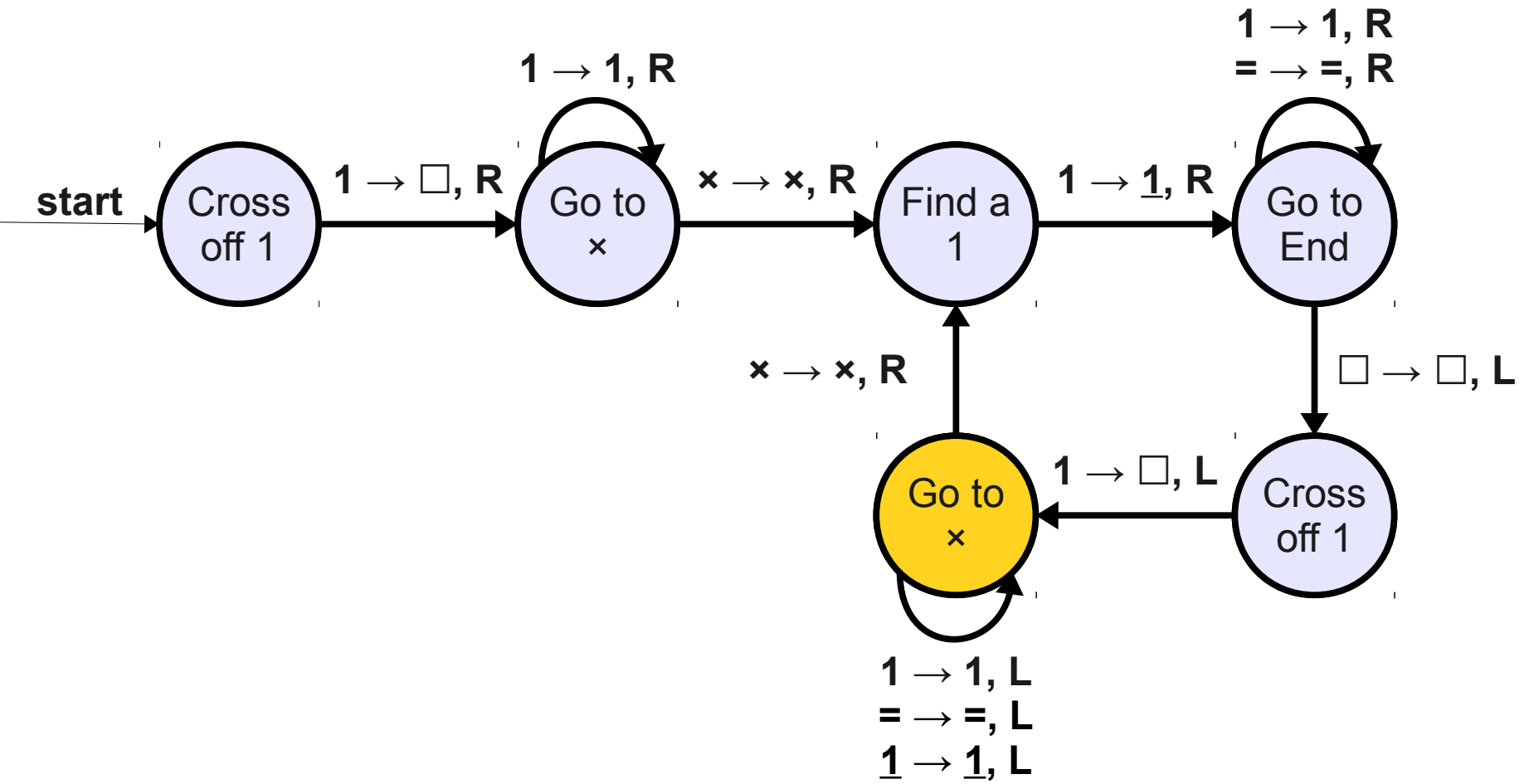

...
1
×
1
1
=
1
1
1
...

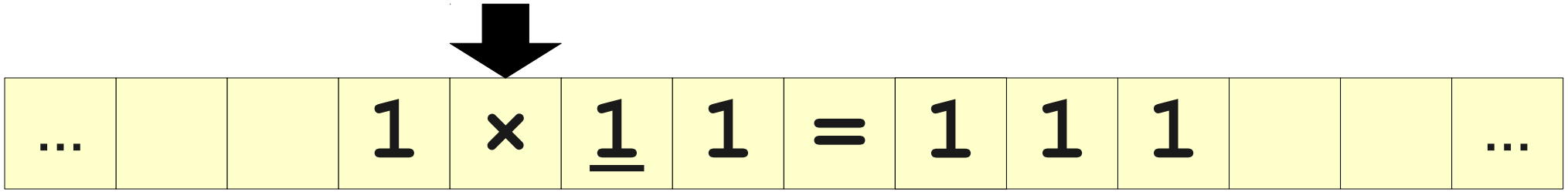

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L
= → =, L
1 → 1, L

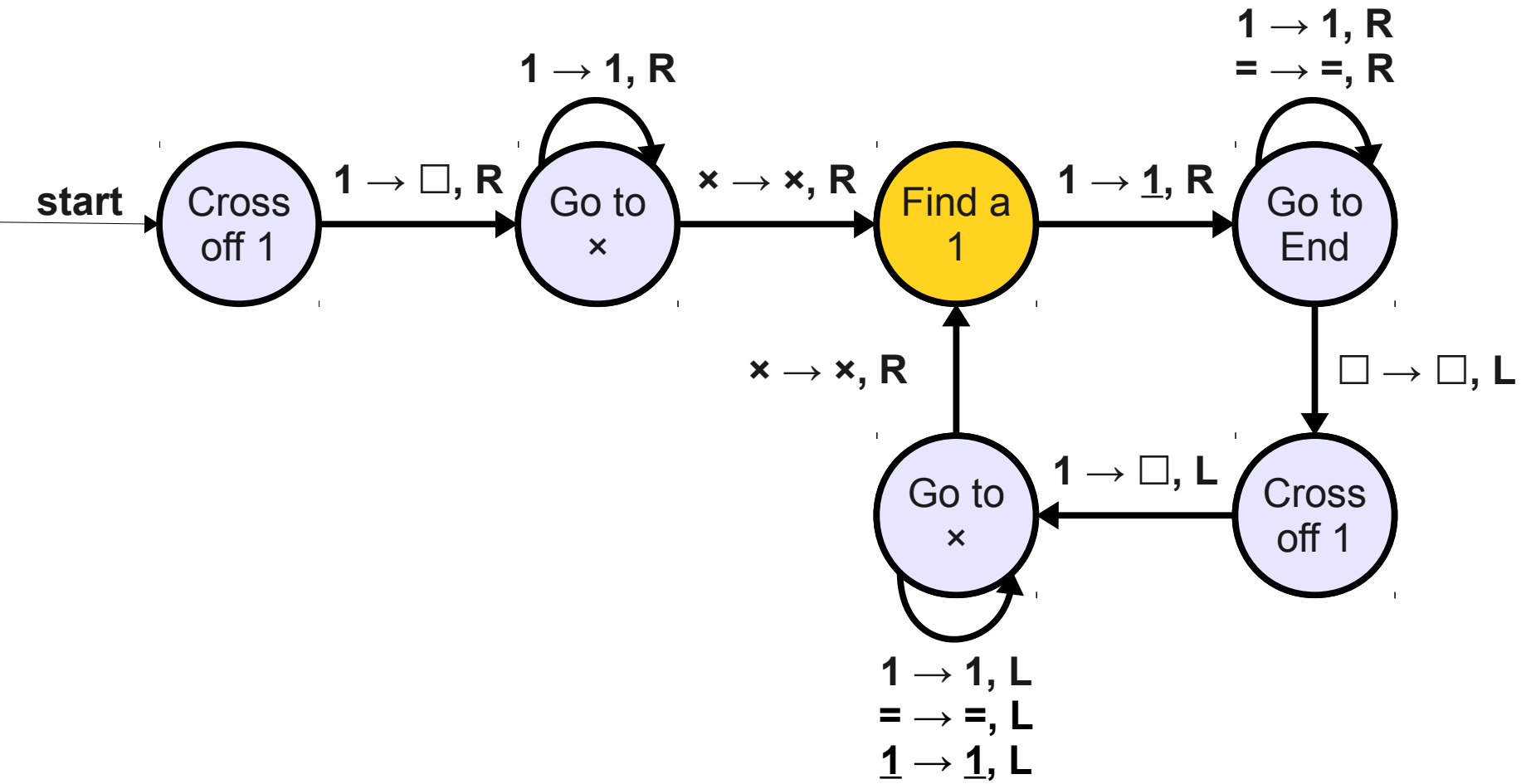

... 1 × 1 1 = 1 1 1 ...

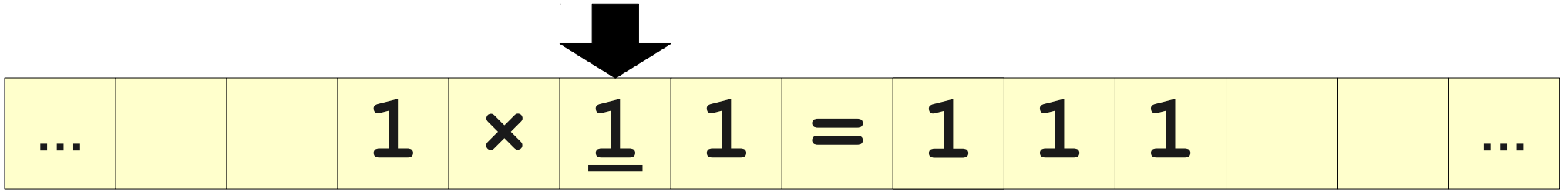

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
1 → 1, L
= → =, L
1 → 1, L

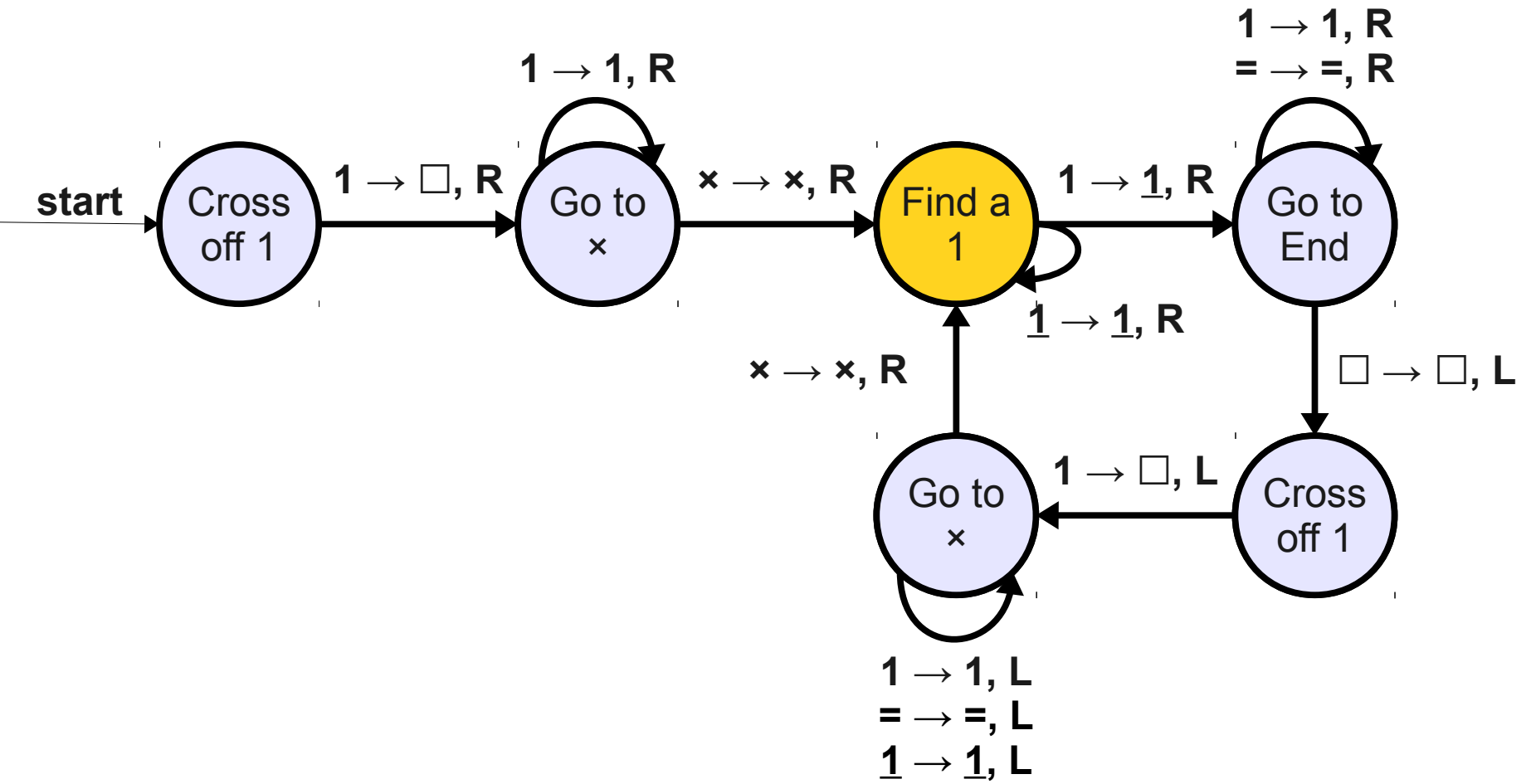

... 1 × 1 1 = 1 1 1 ...

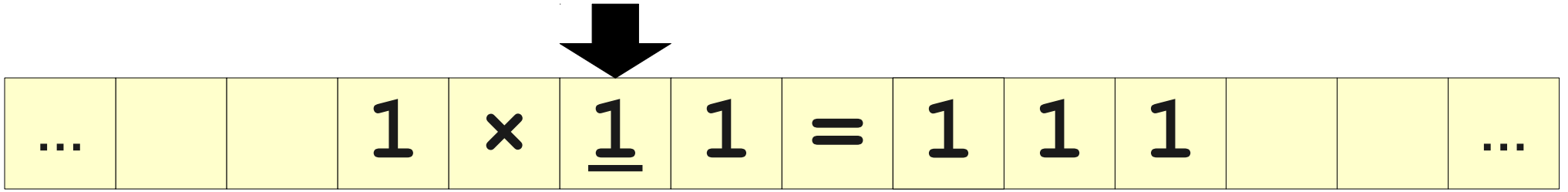

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

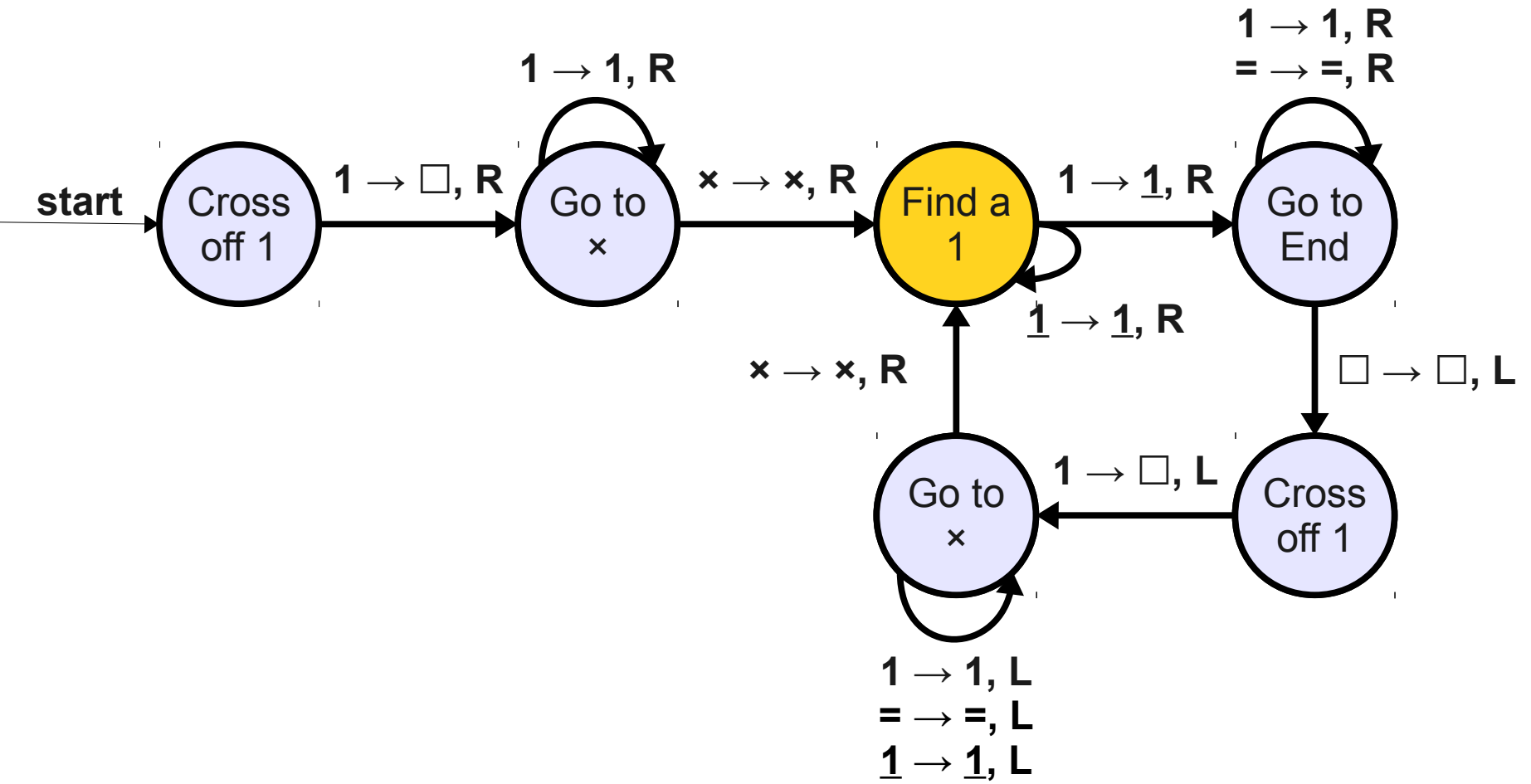

... 1 × 1 1 = 1 1 1 ...

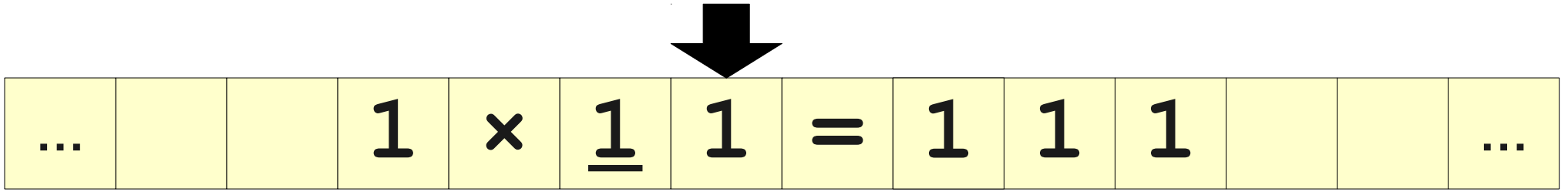

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

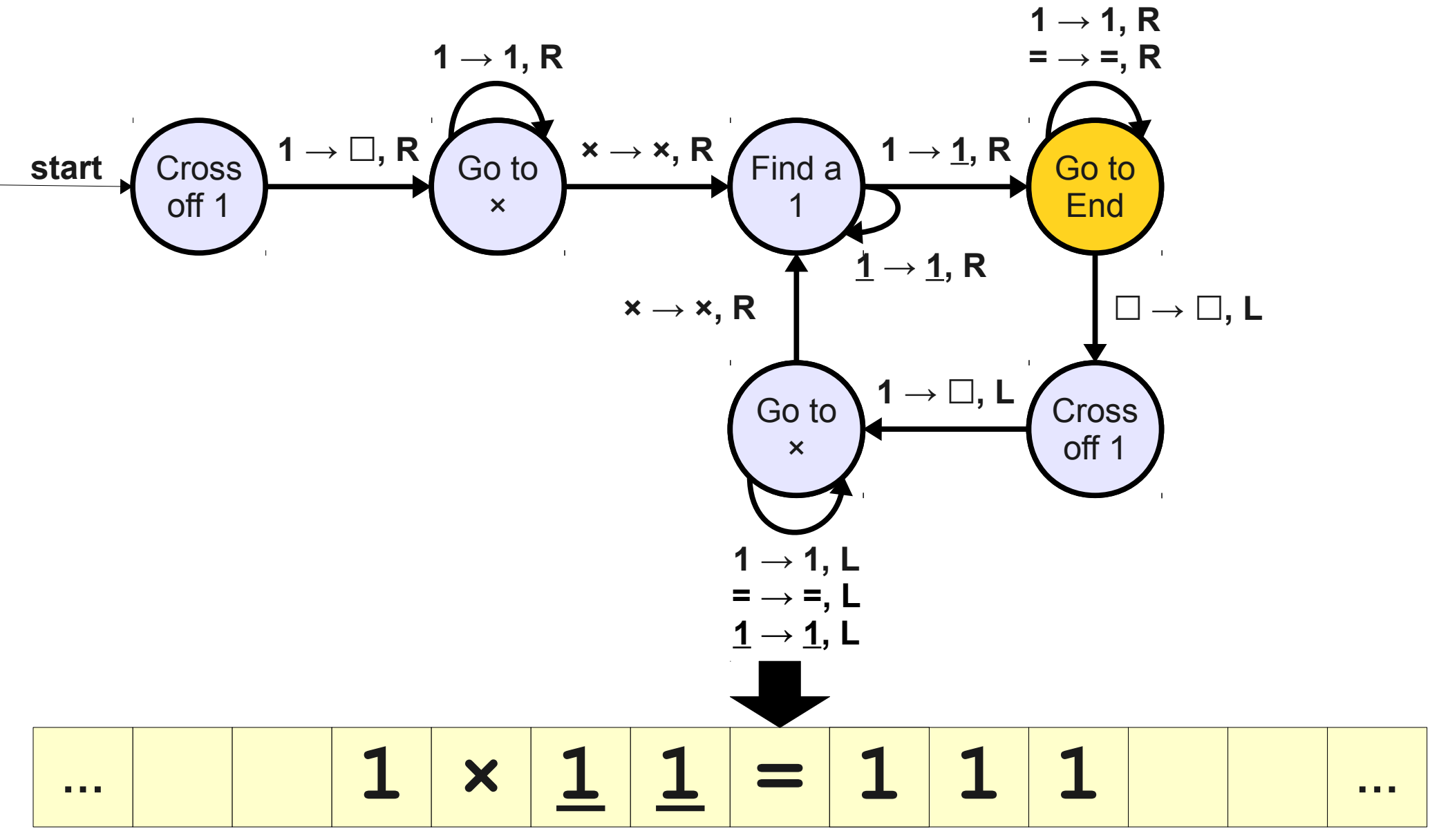

...
1 × 1 1 = 1 1 1
...

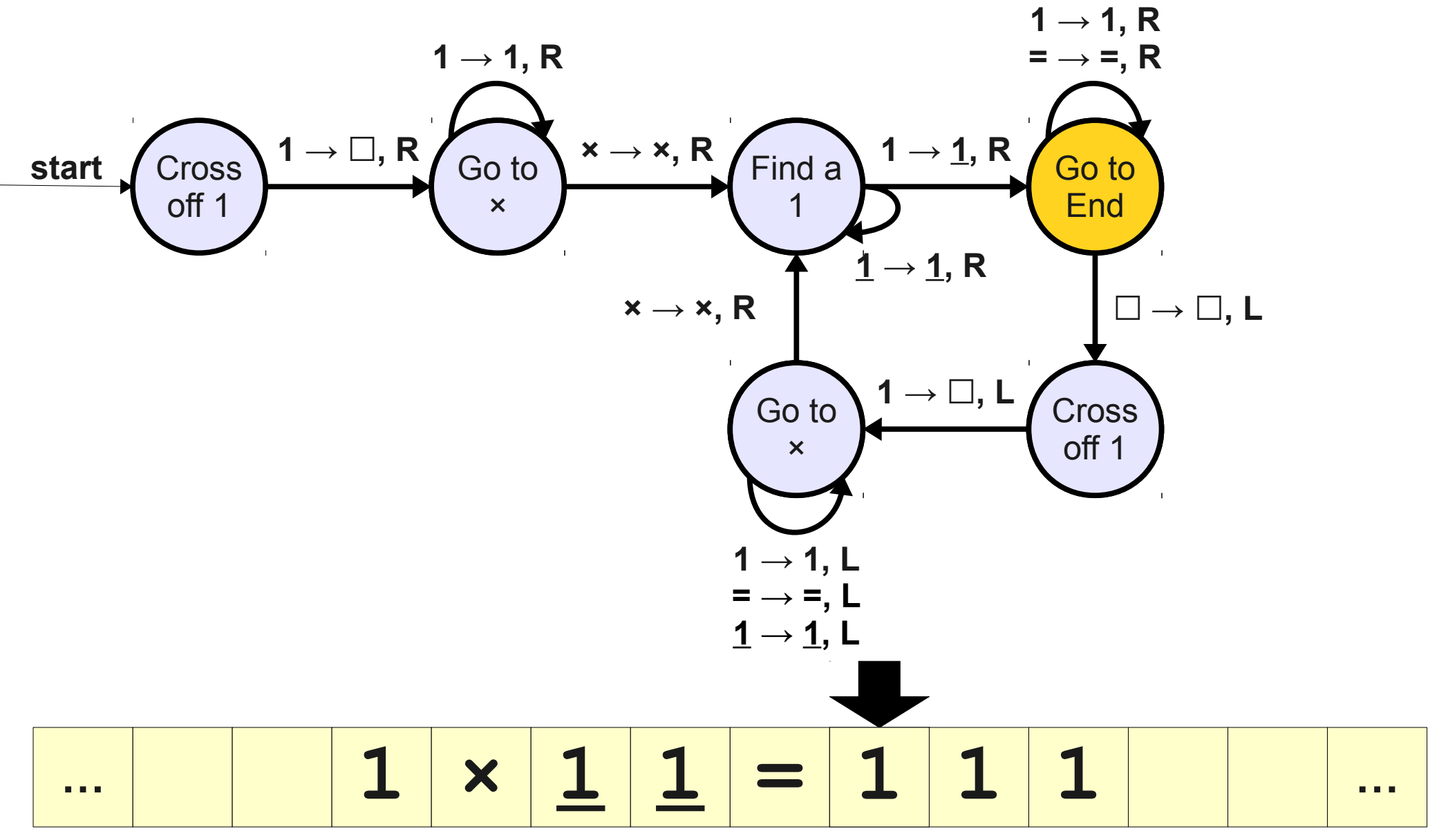

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
1 → 1, R
Go to End
1 → 1, R
= → =, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

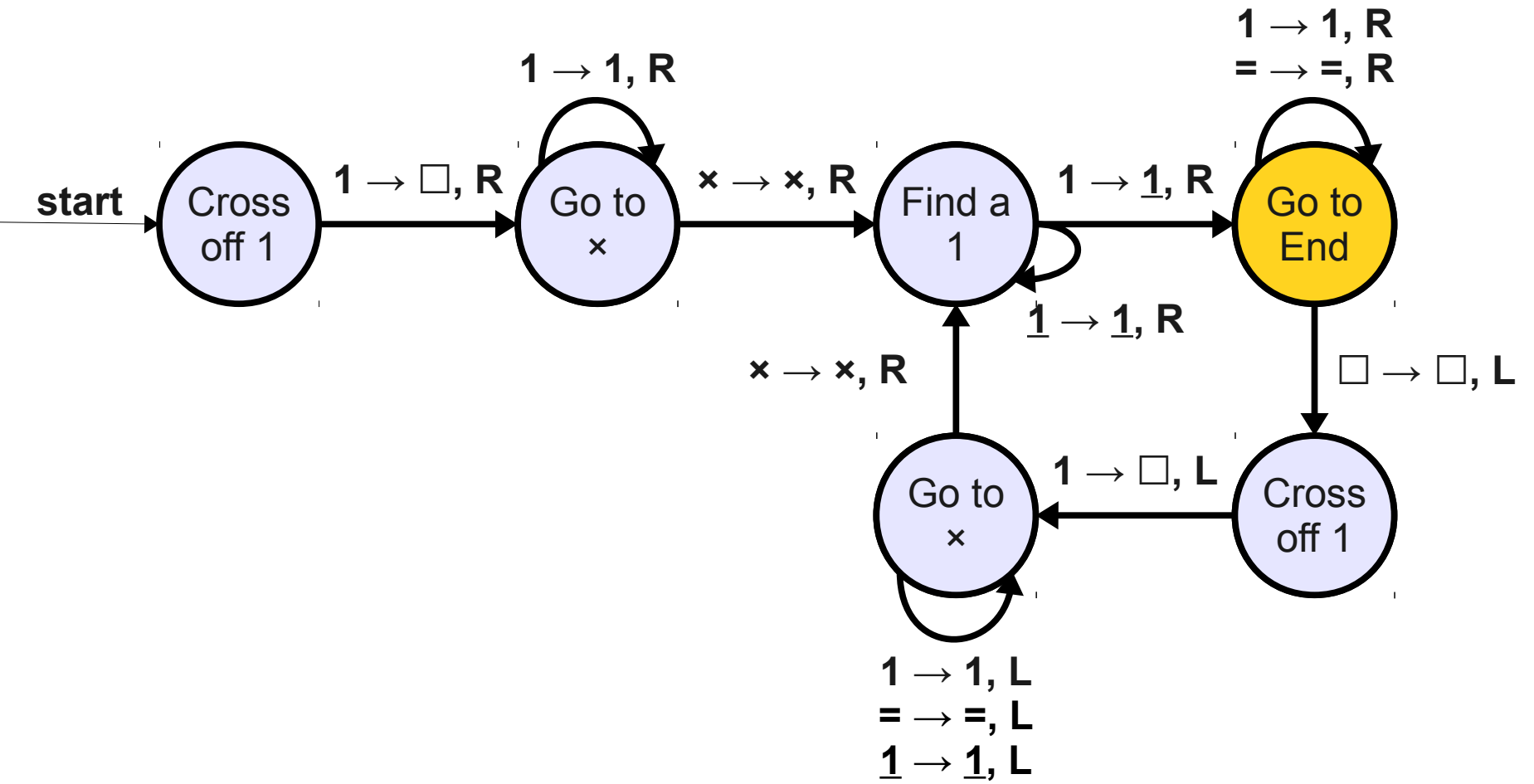

...
1 × 1 1 = 1 1 1
...

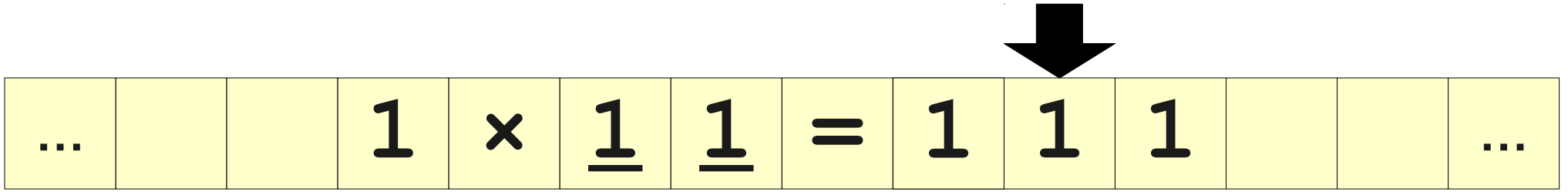

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
1 → 1, R
= → =, R
Go to End
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

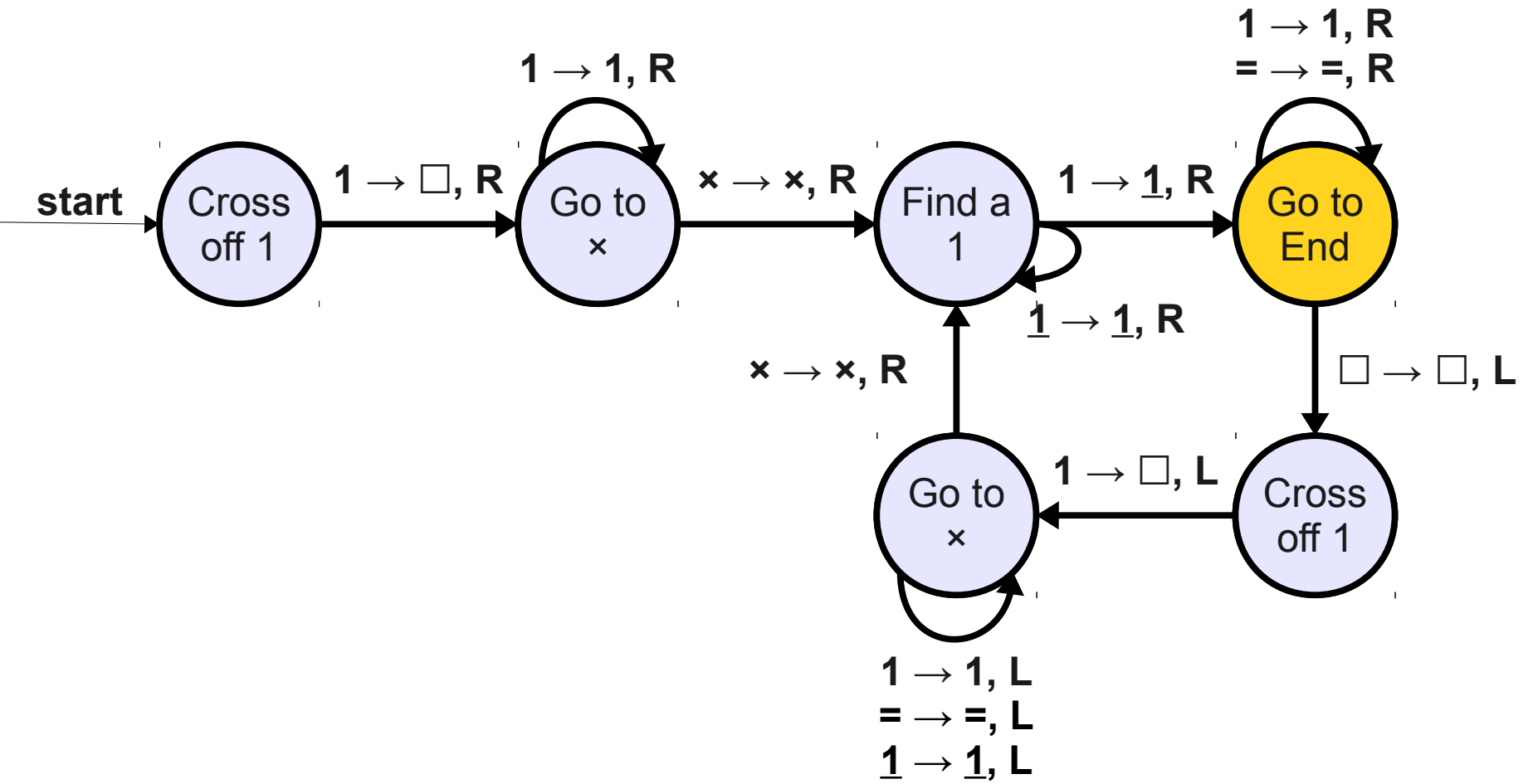

...
1 × 1 1 = 1 1 1
...

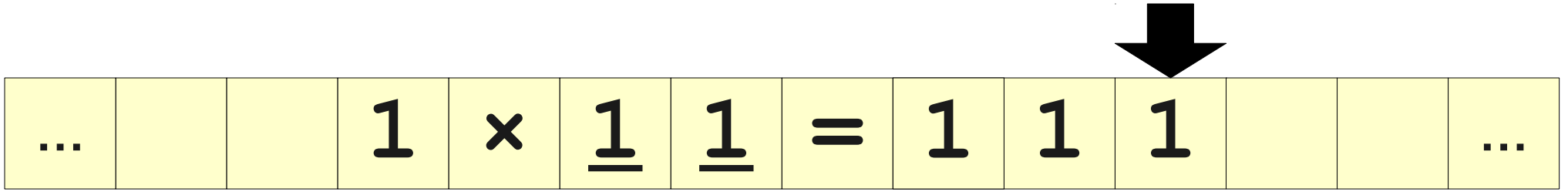

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 1 ...

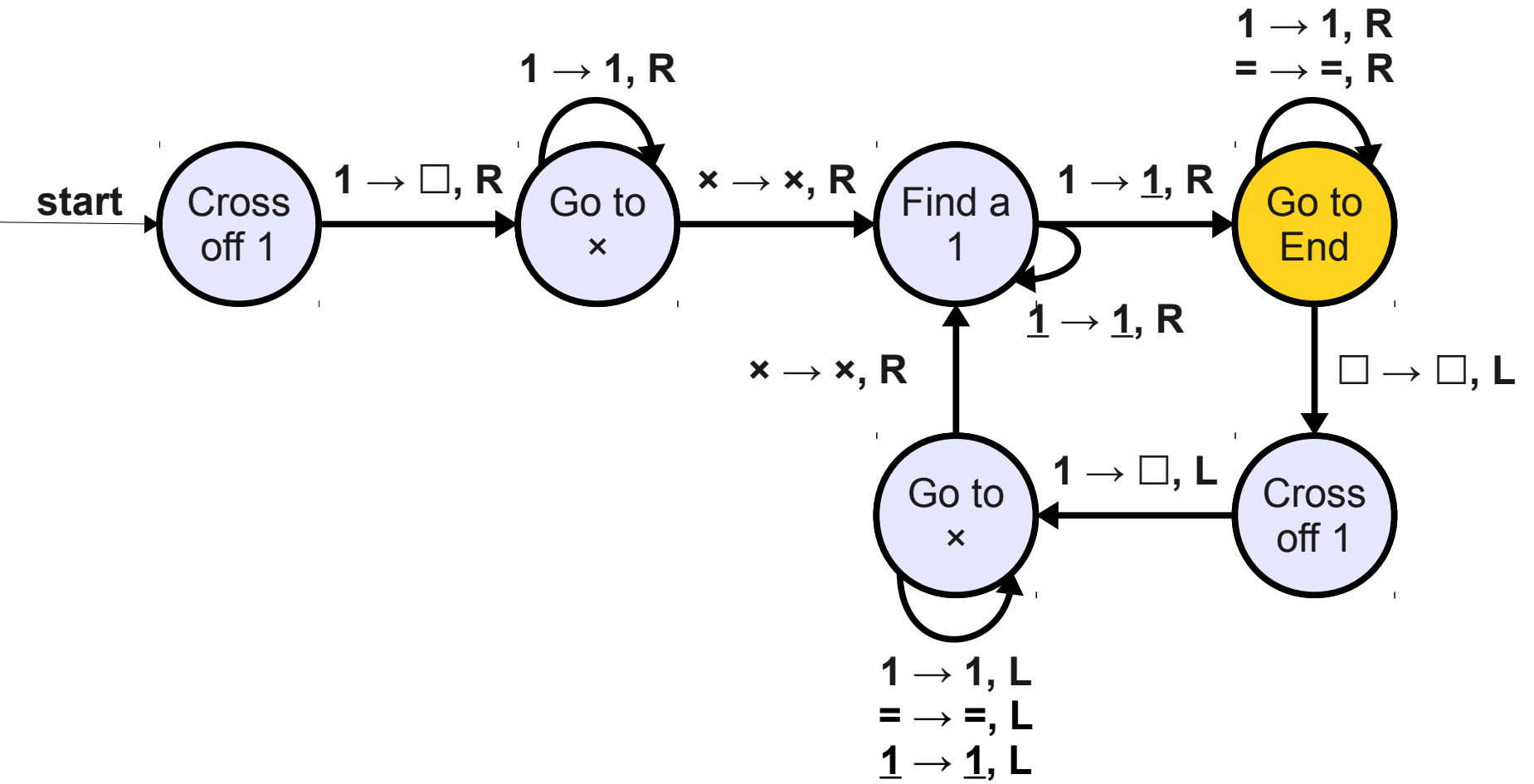

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 1 ...

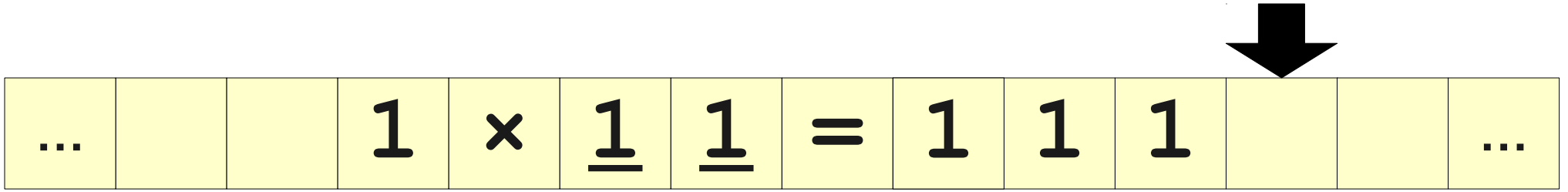

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L

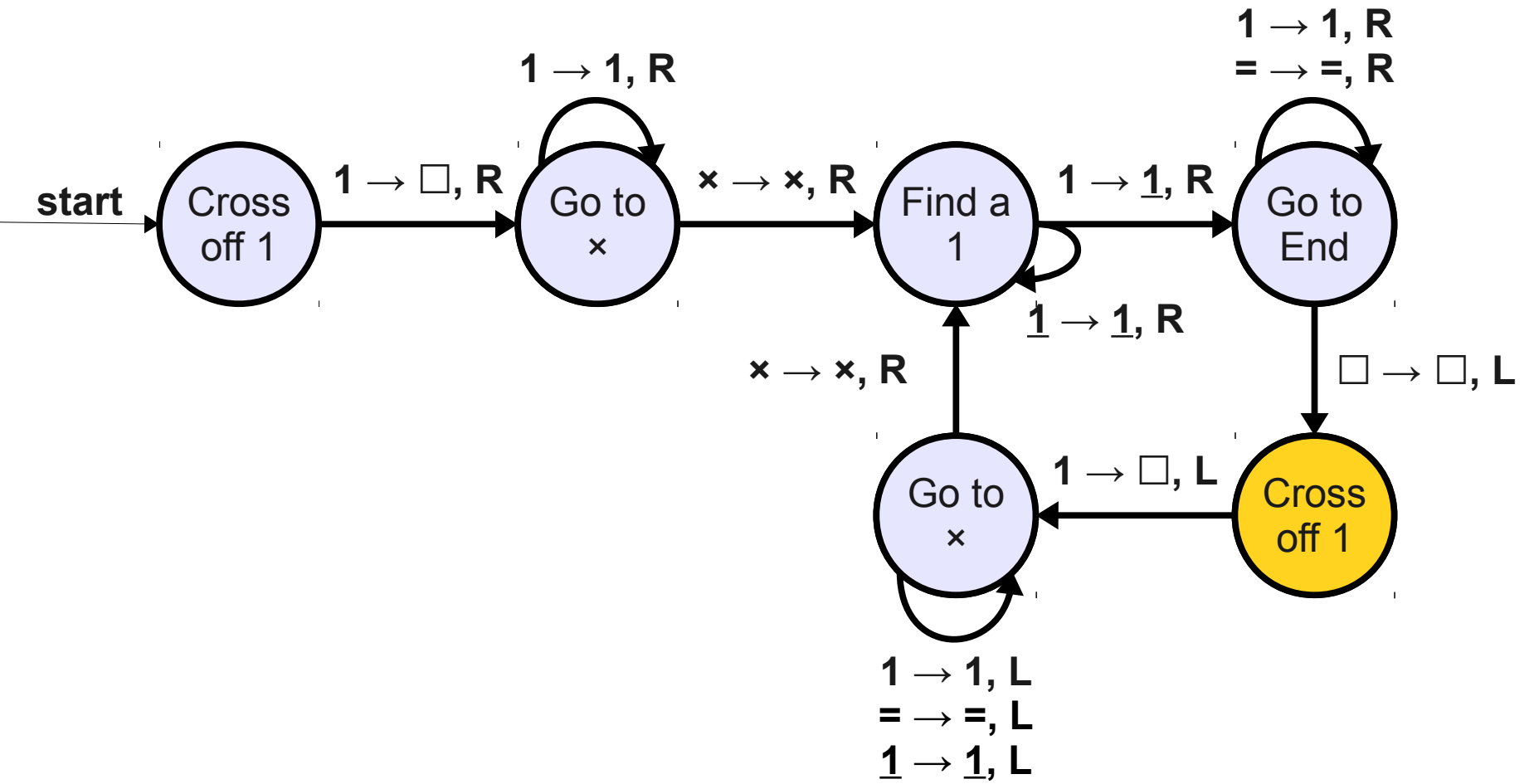

...
1 × 1 1 = 1 1 1
...

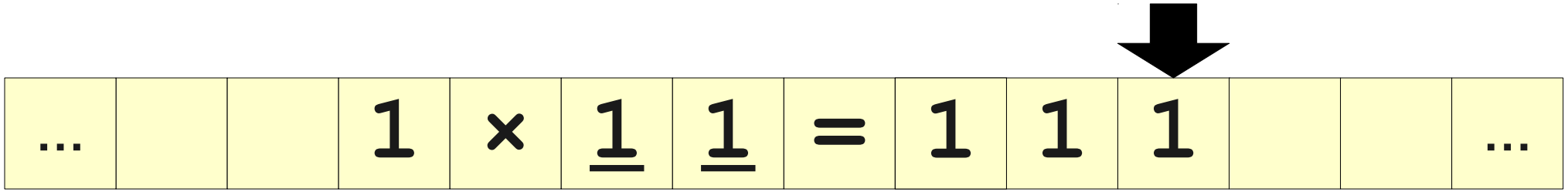

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
1 → 1, R
= → =, R
Go to End
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

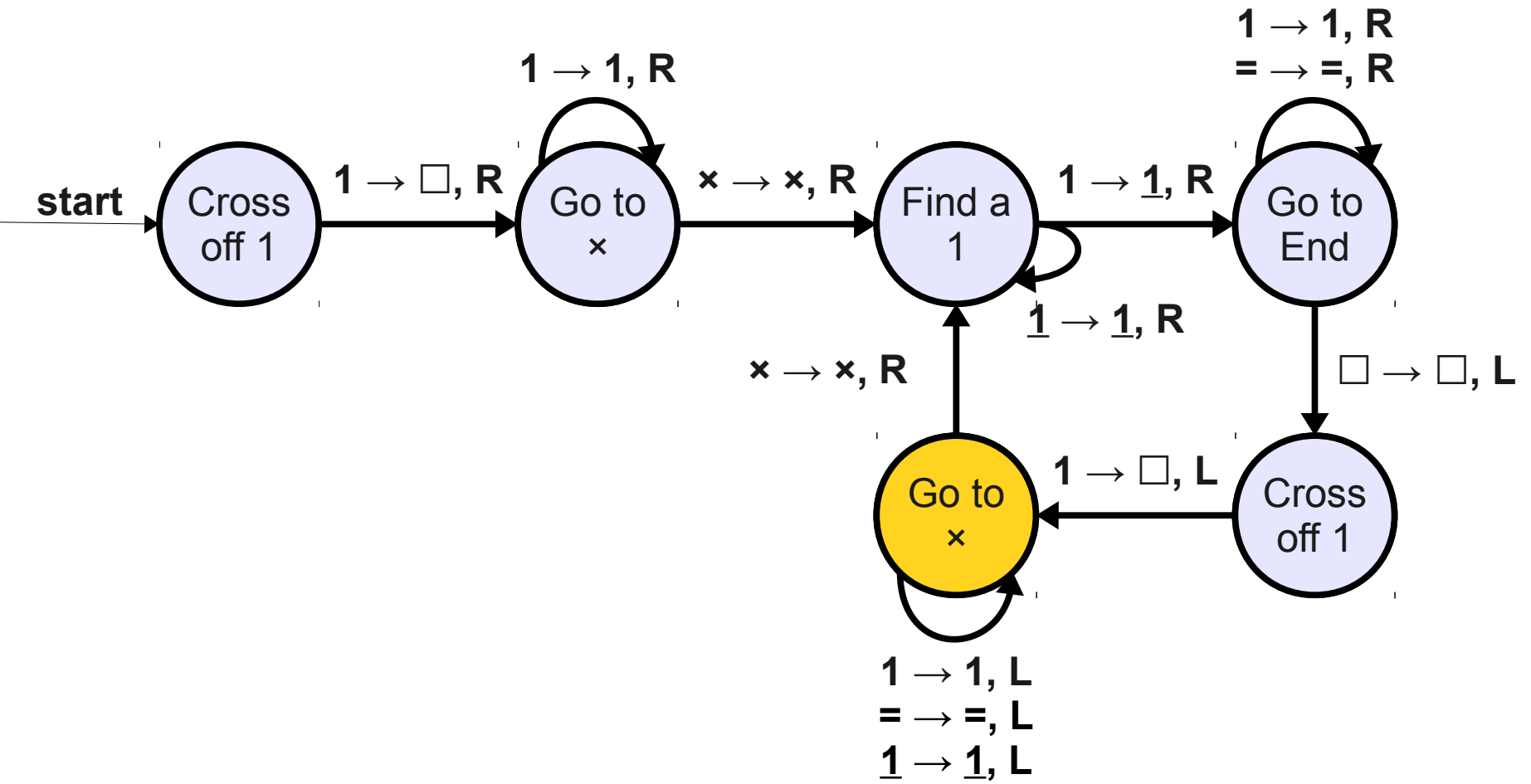

...
1 × 1 1 = 1 1 1
...

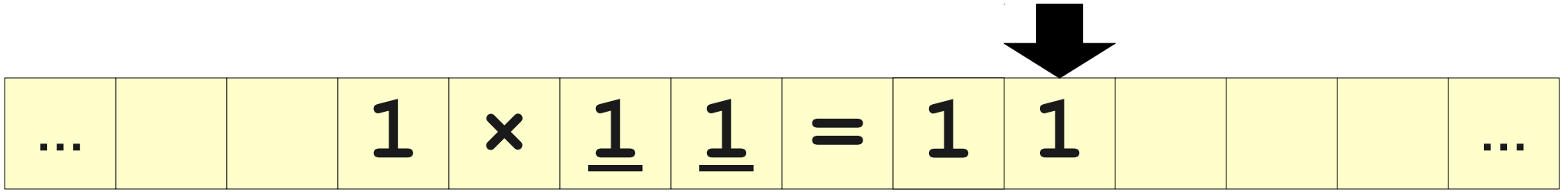

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
1 → 1, R
= → =, R
Go to End
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

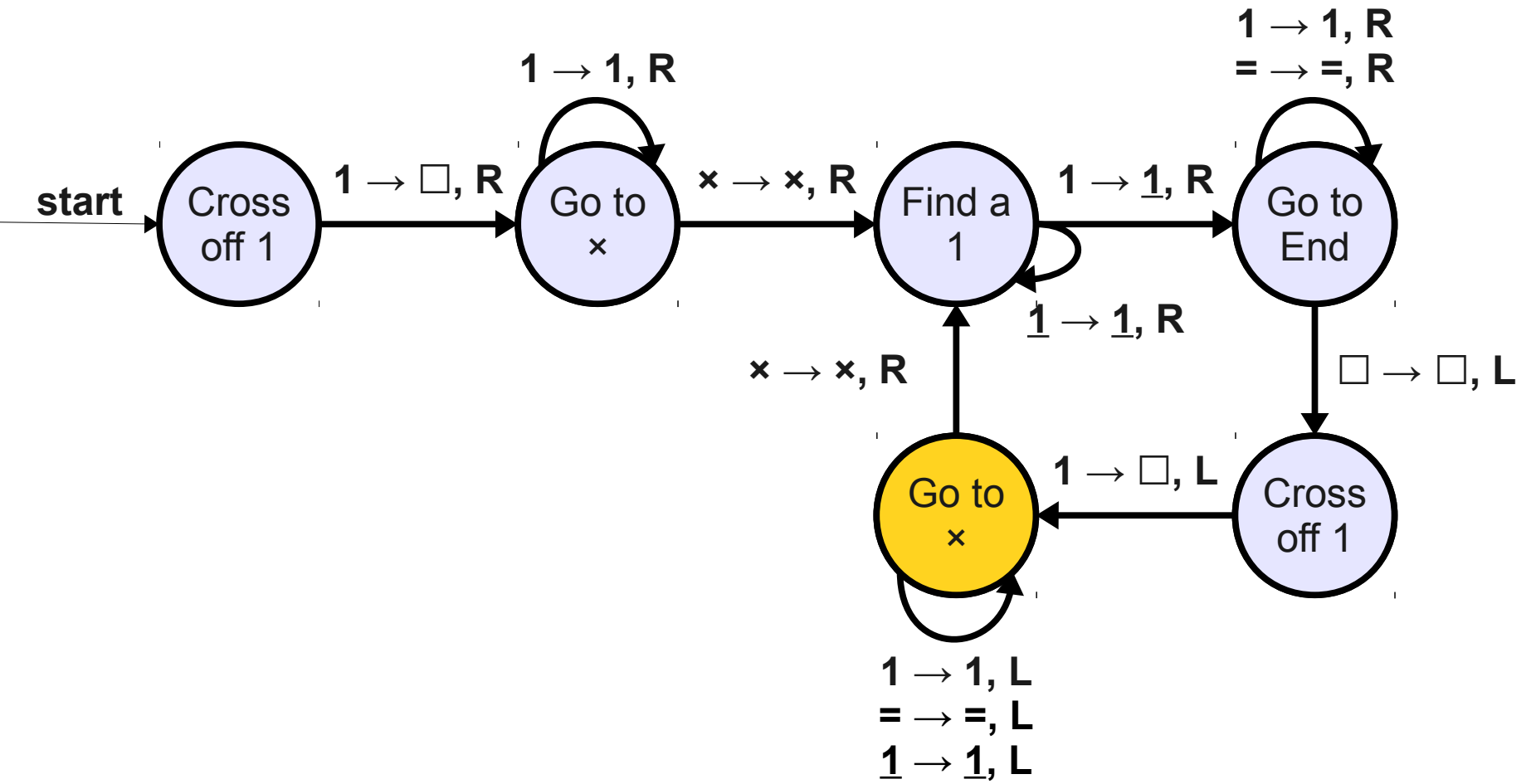

...
1 × 1 1 = 1 1 1
...

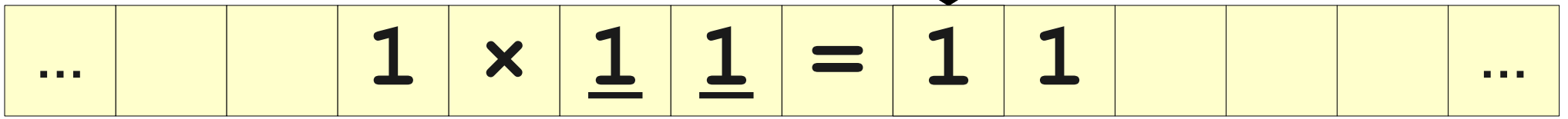

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

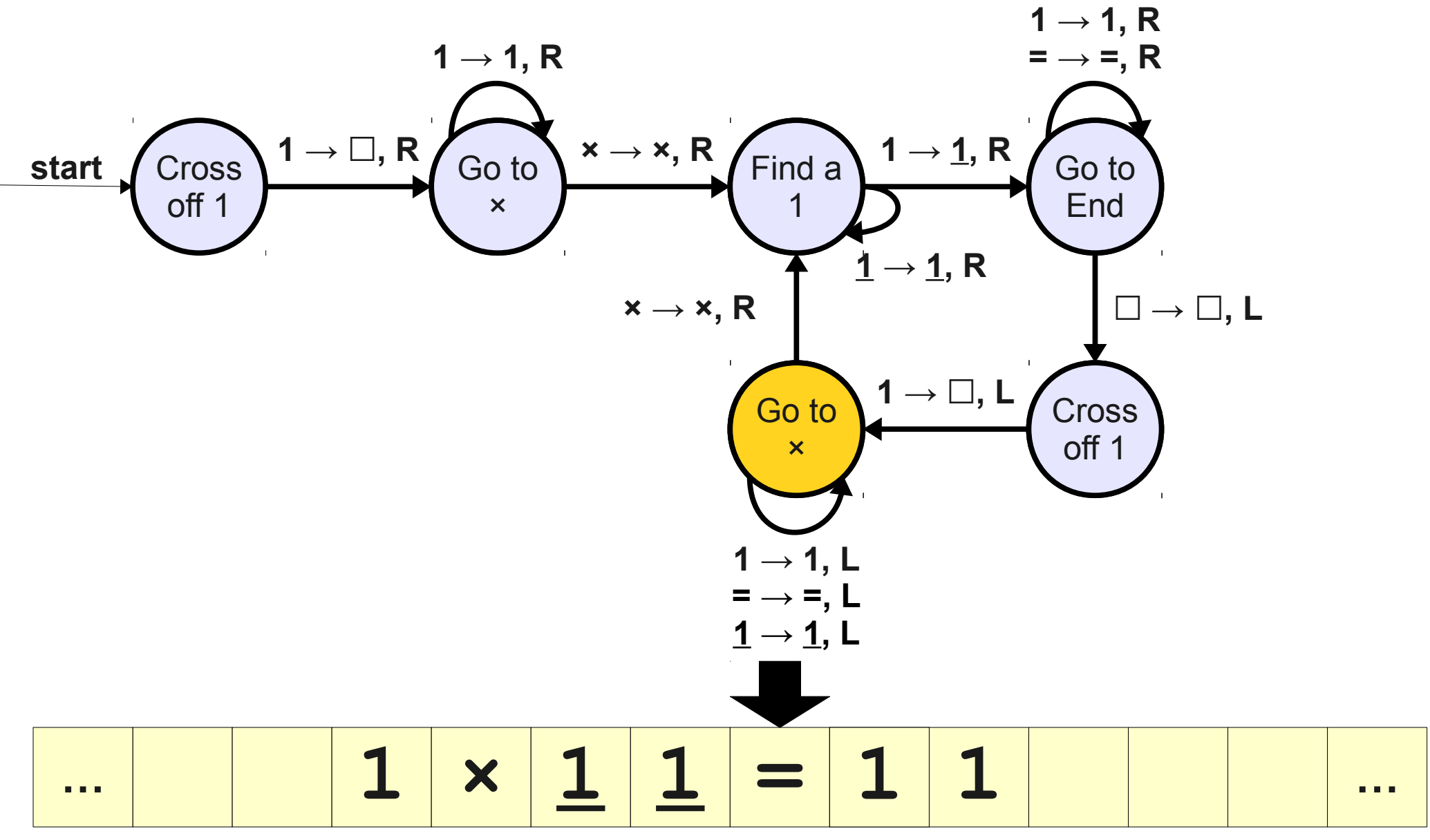

...
1 × 1 1 = 1 1 1
...

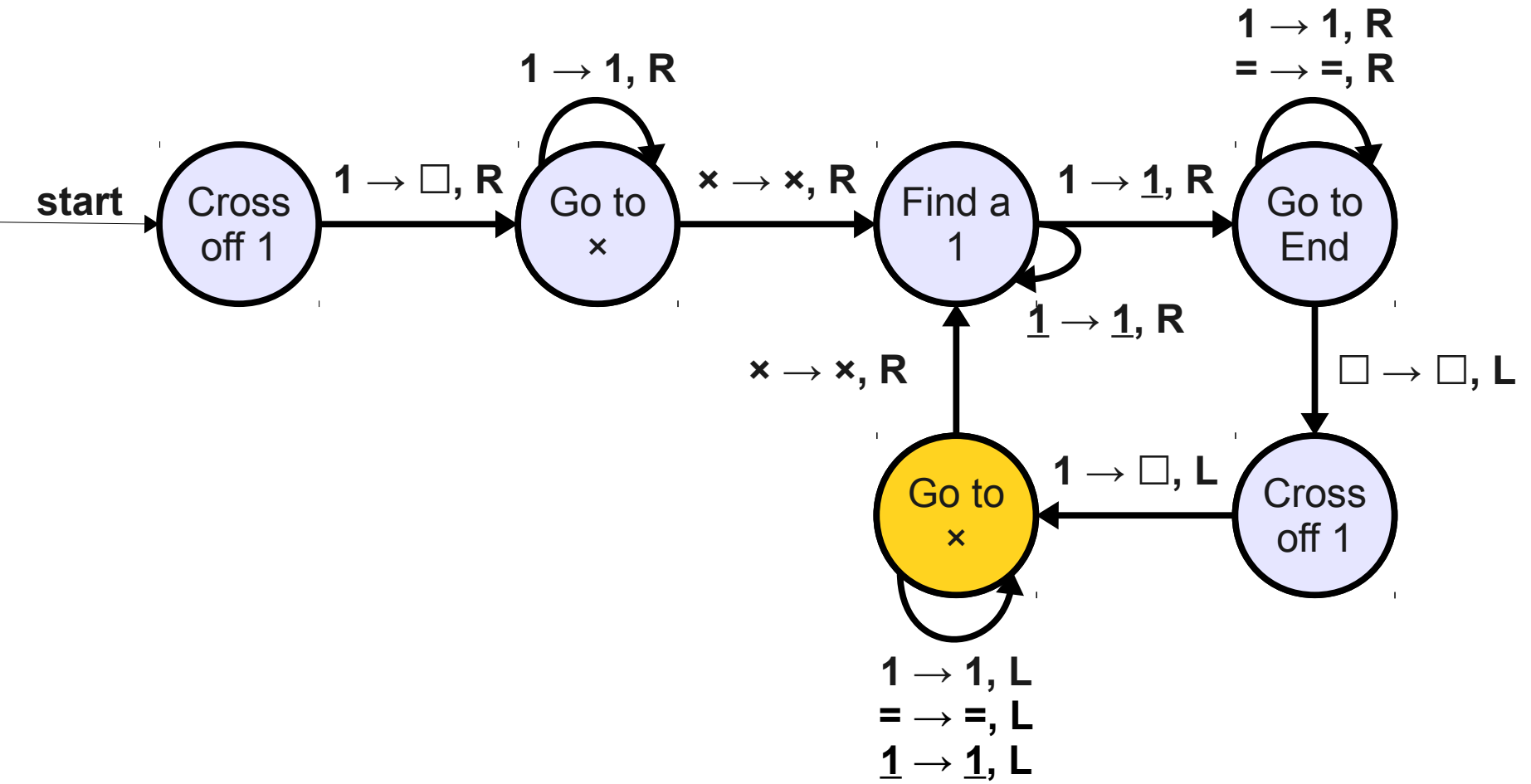

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

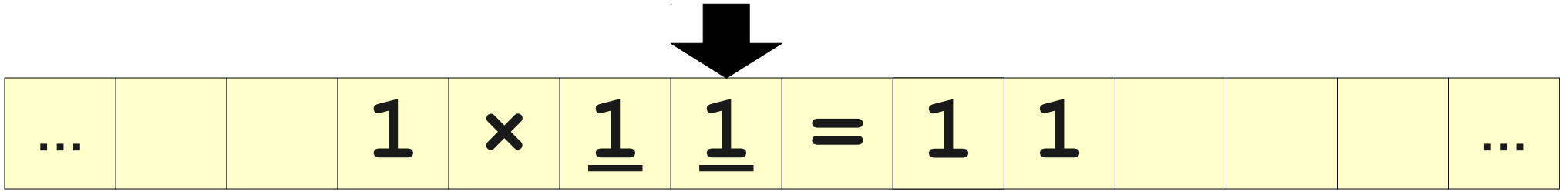

...
1
×
1
1
=
1
1
...

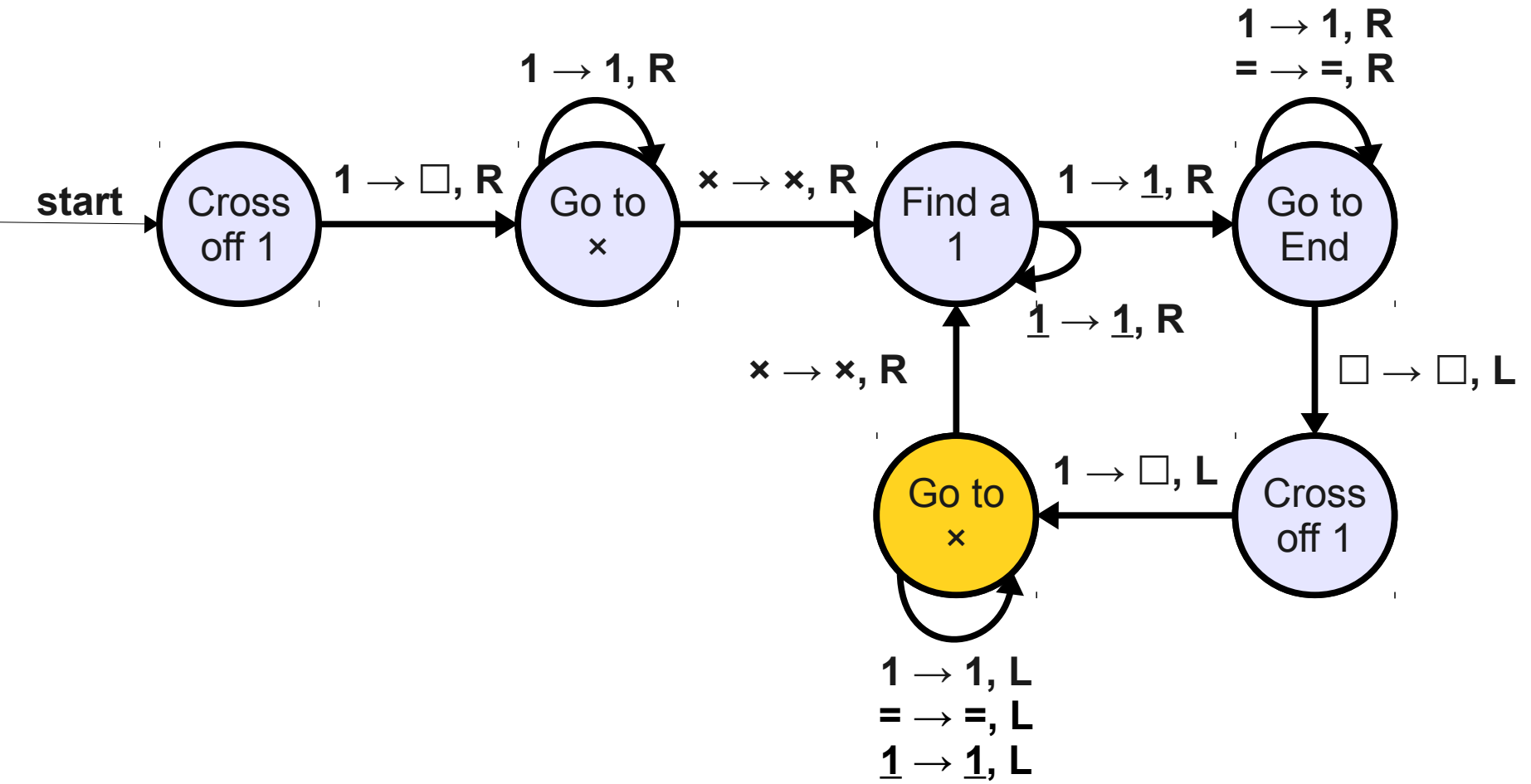

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L

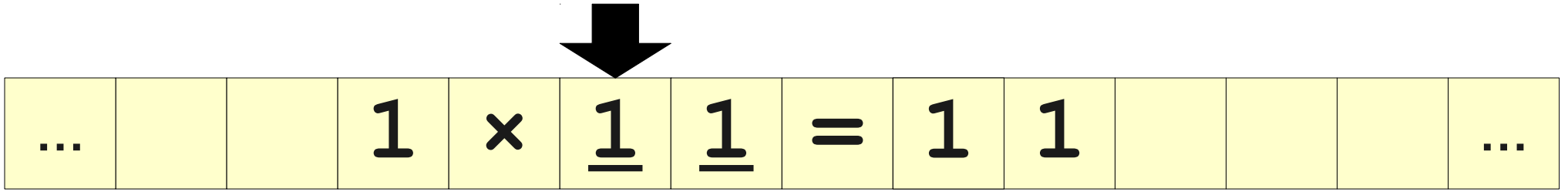

...
1
×
1
1
=
1
1
...

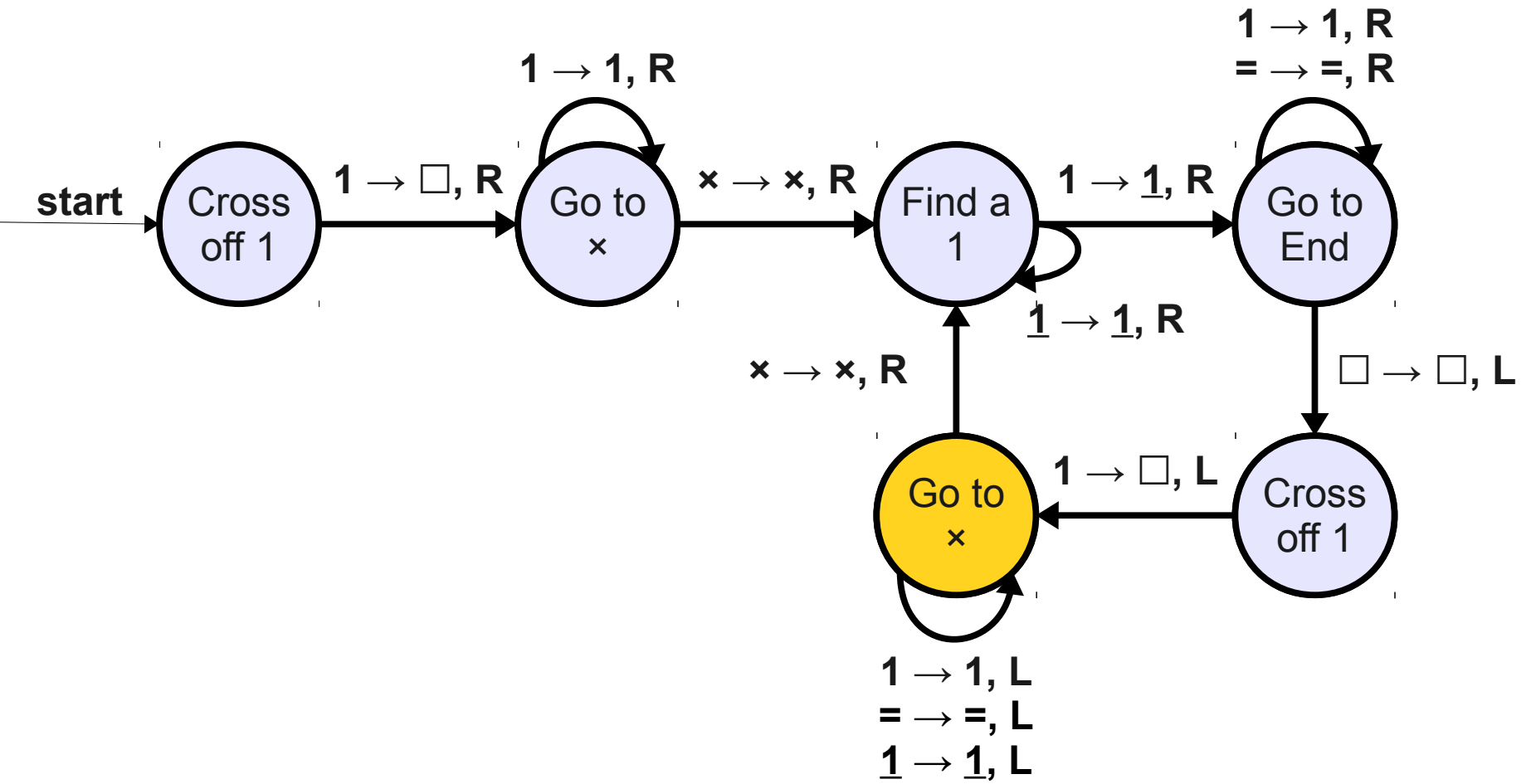

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

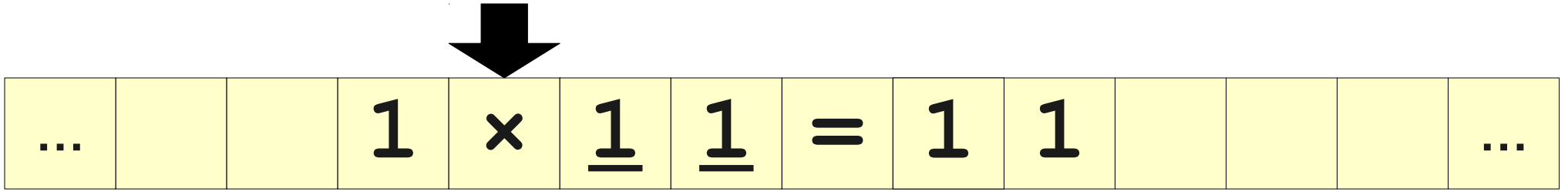

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

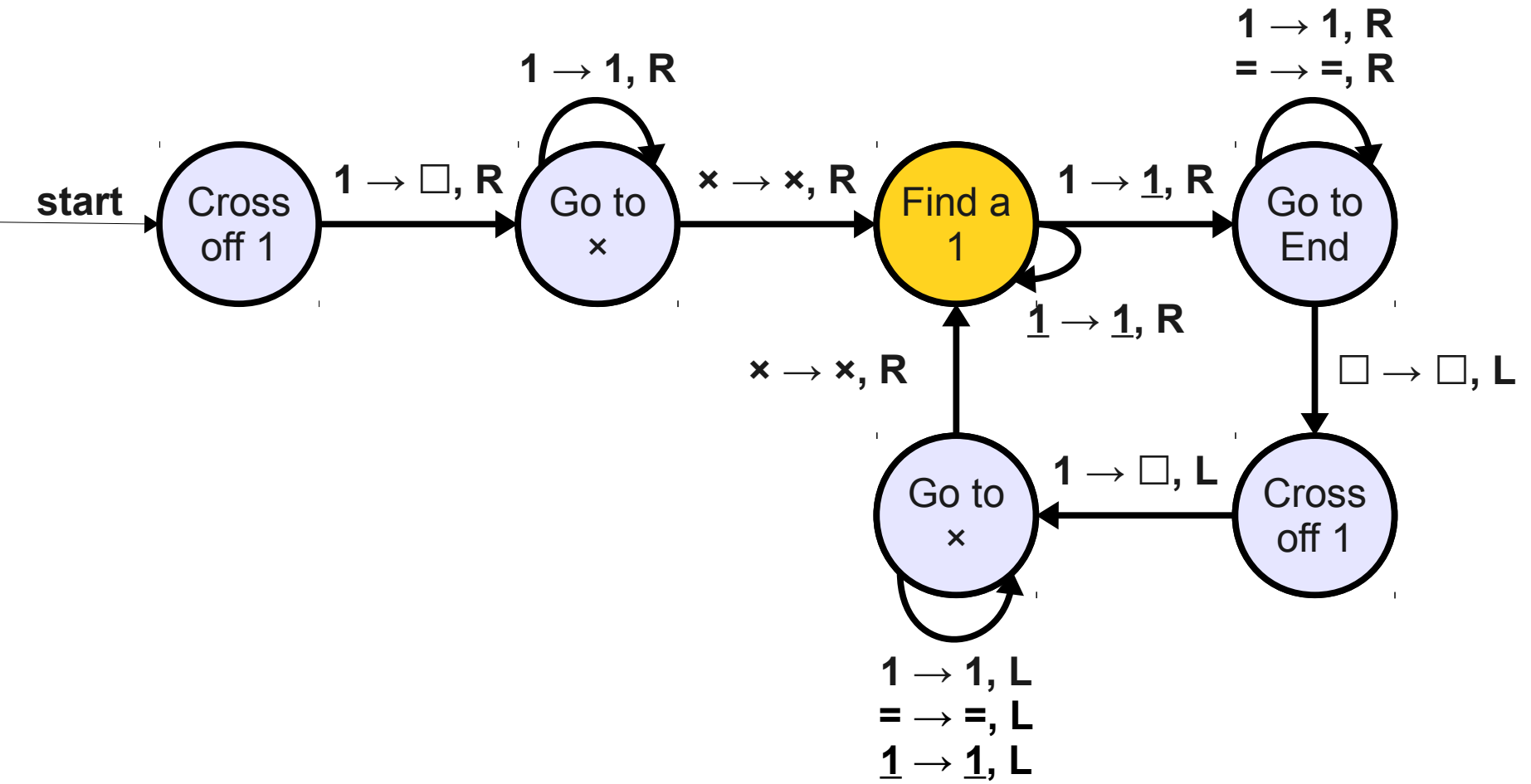

...
1 × 1 1 = 1 1
...

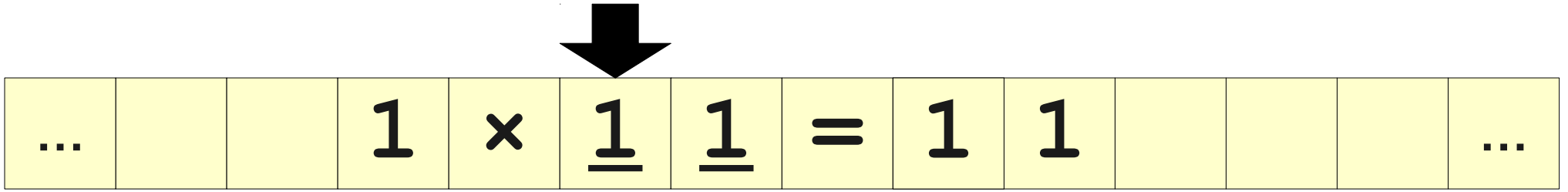

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

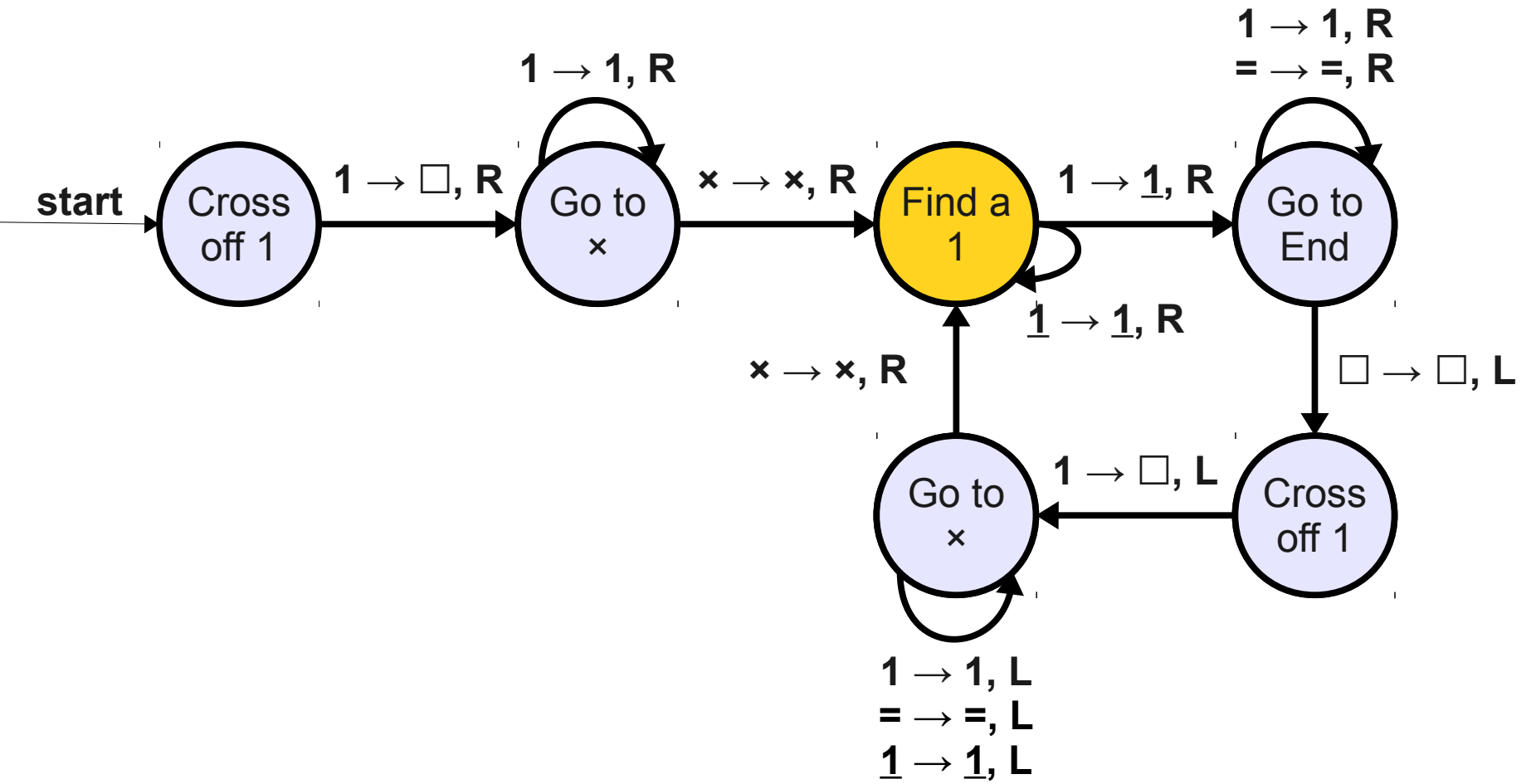

...
1
×
1
1
=
1
1
...

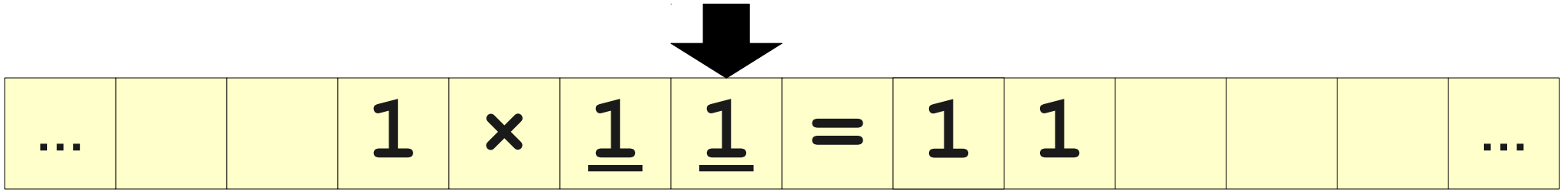

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

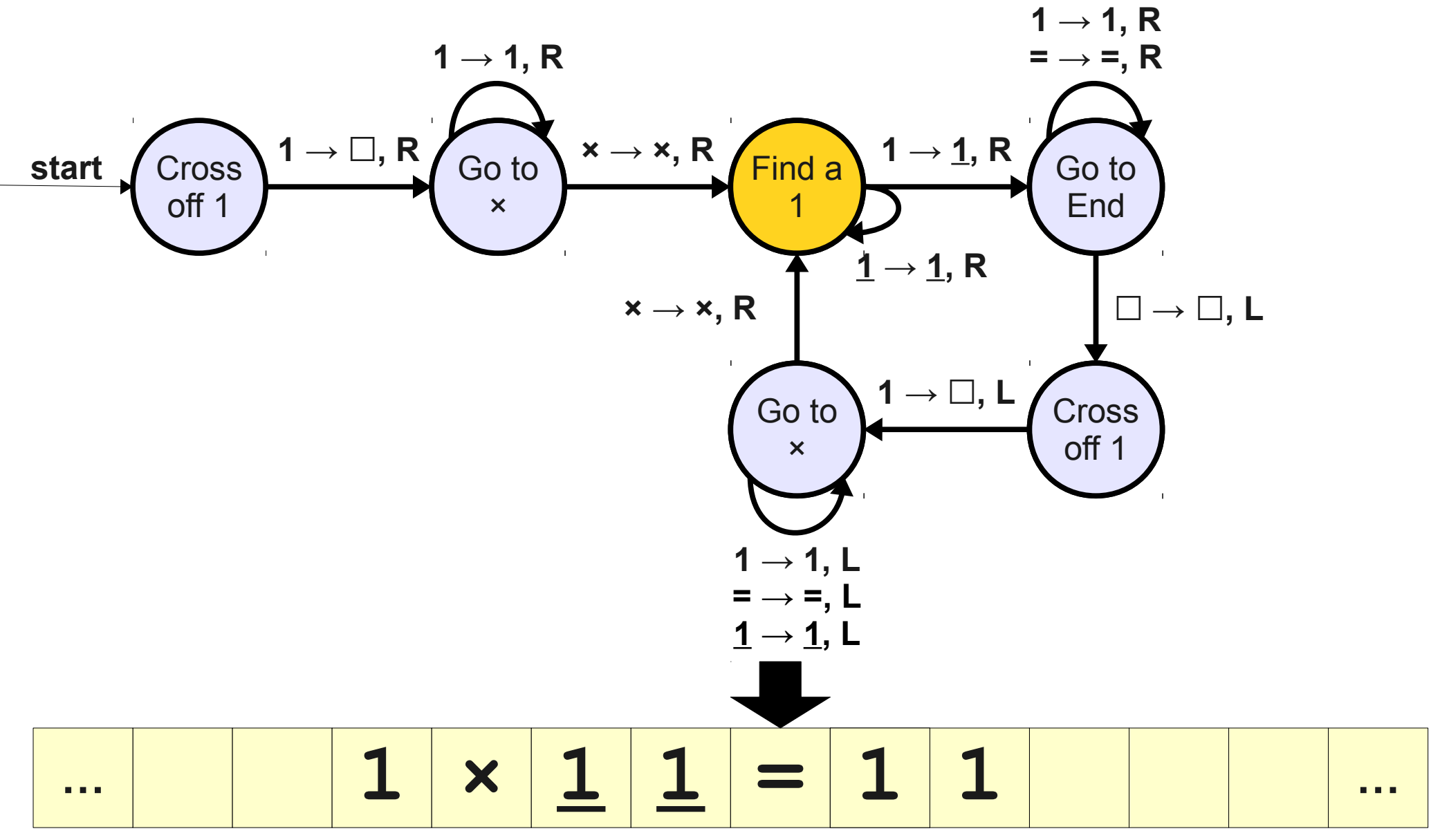

...
1
×
1
1
=
1
1
...

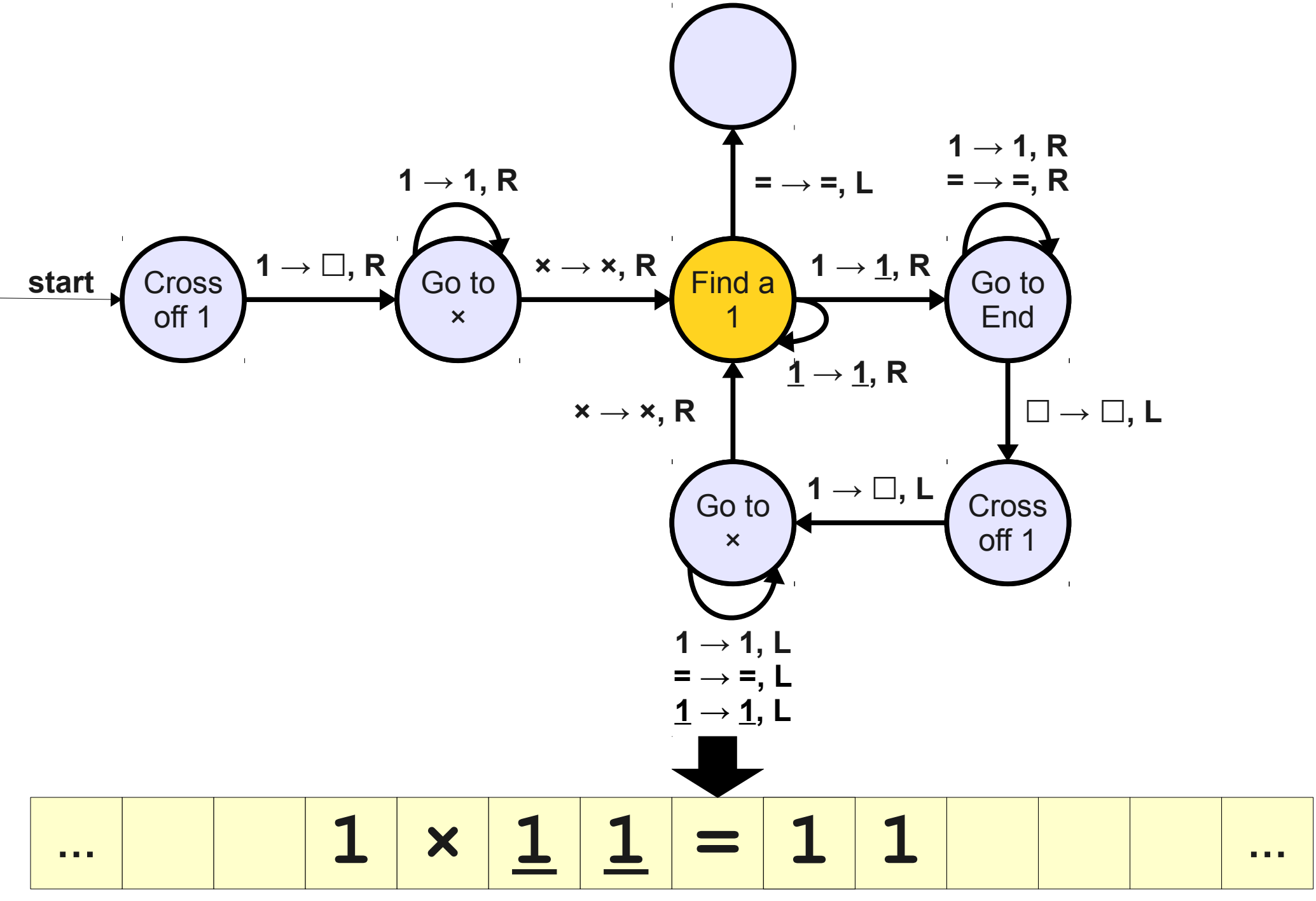

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

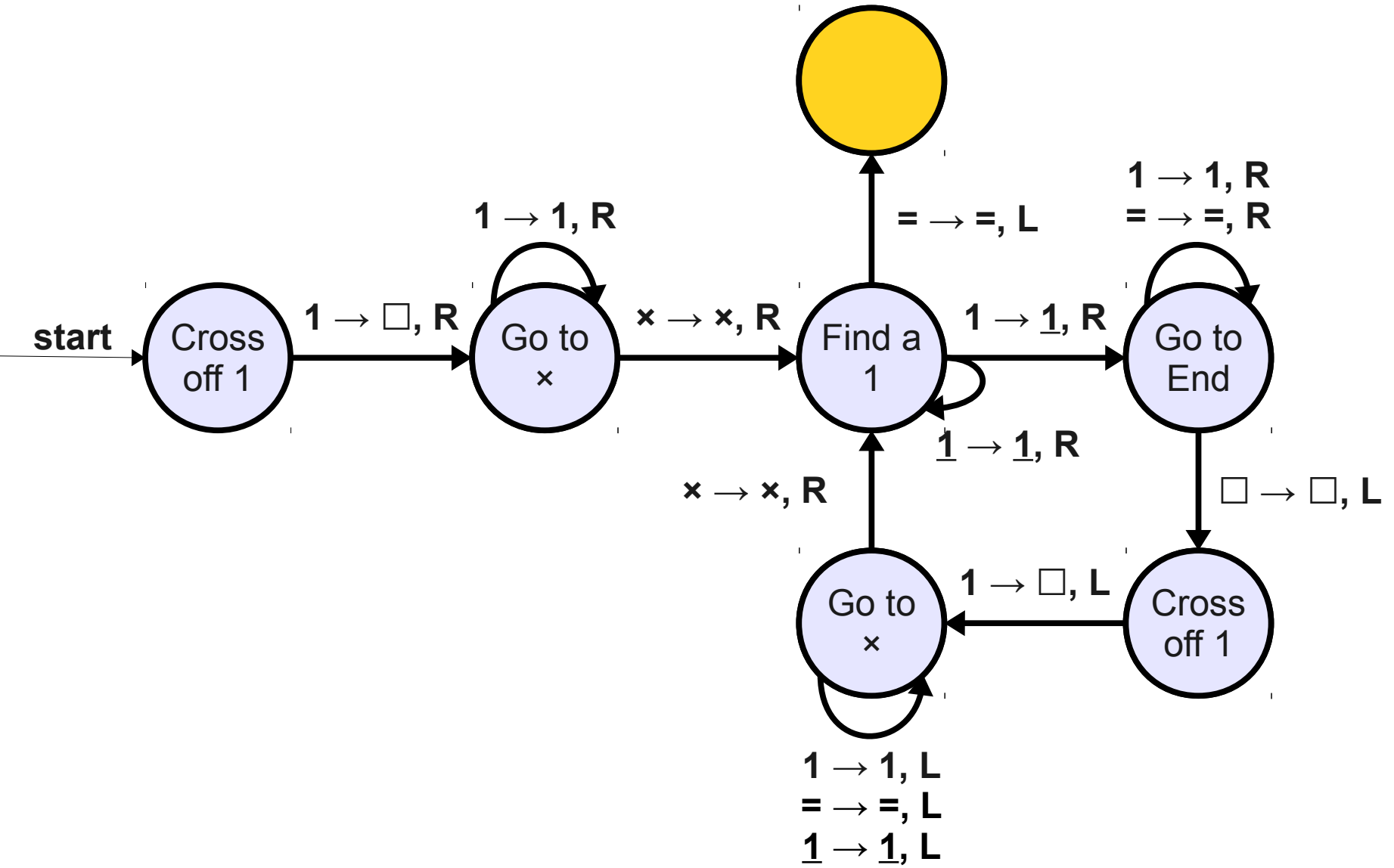

...
1
×
1
1
=
1
1
...

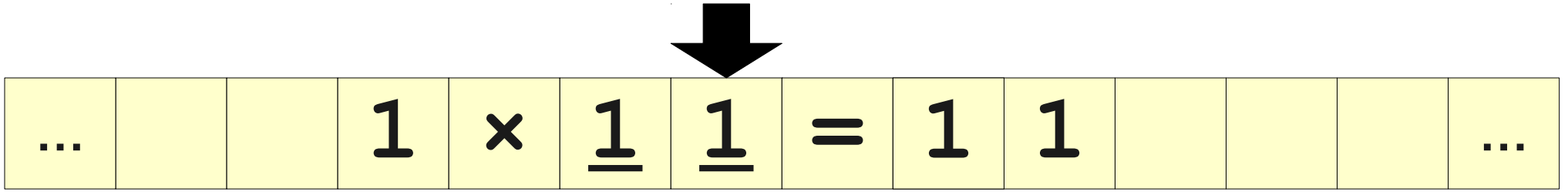

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

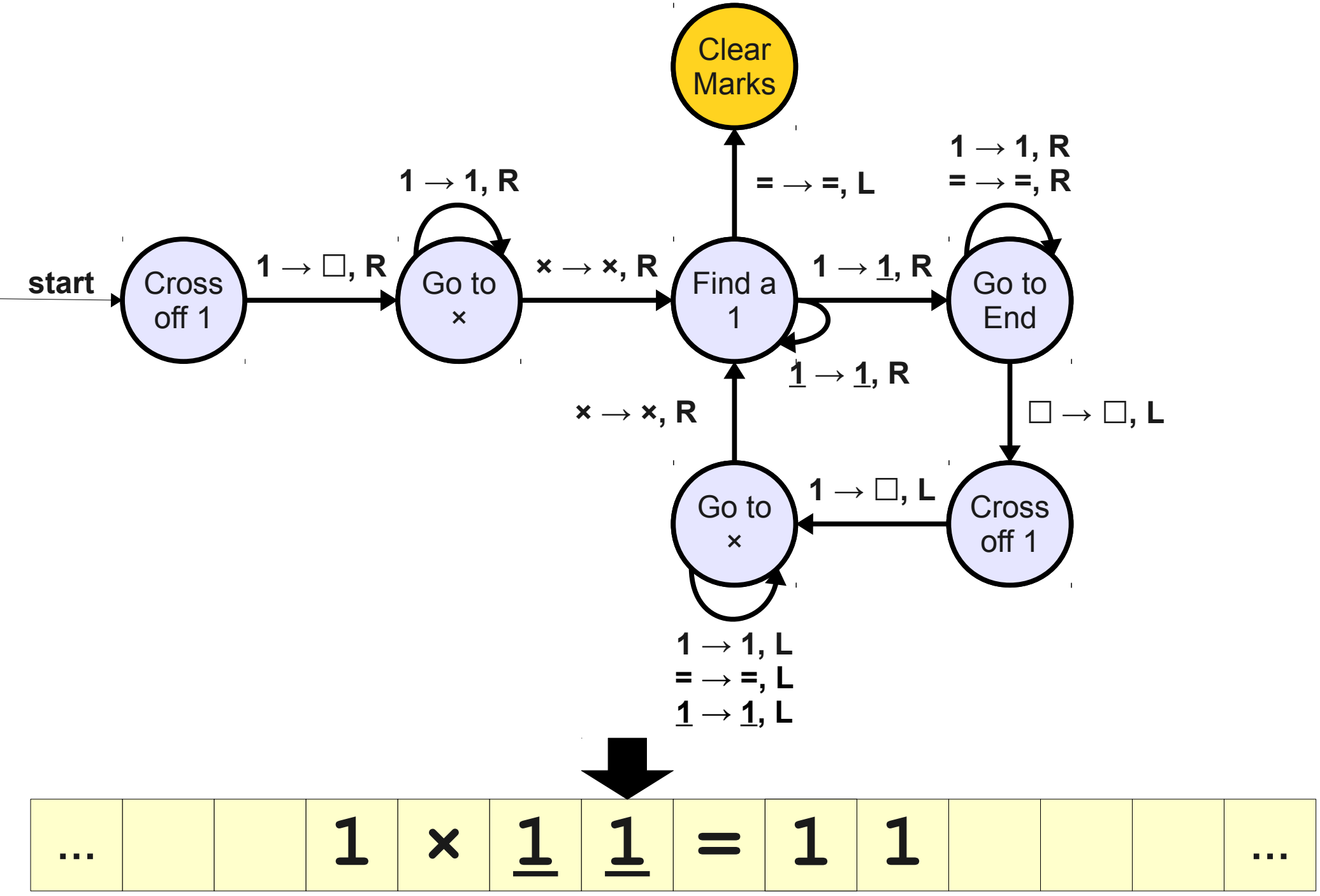

...
1
×
1
1
=
1
1
...

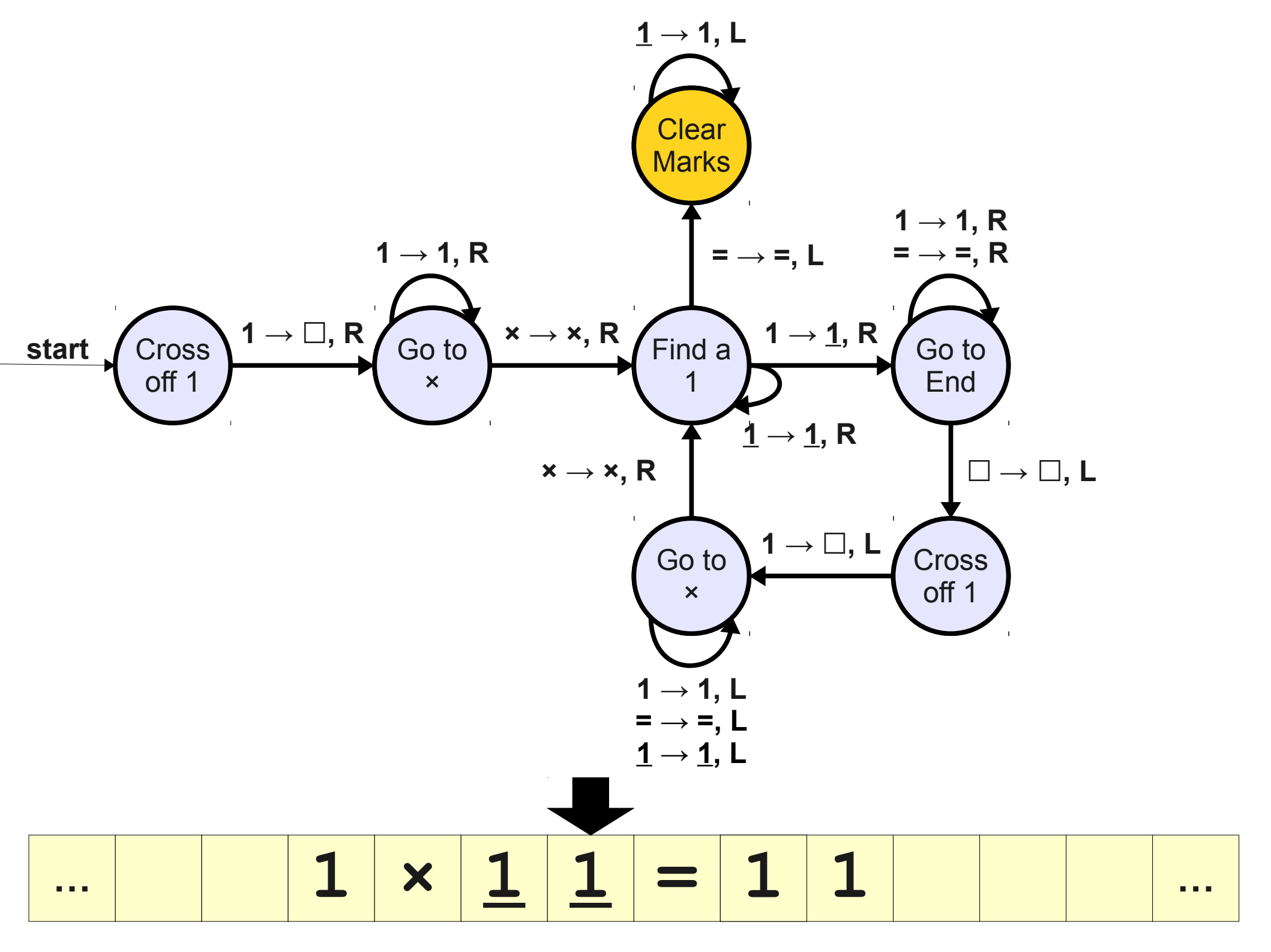

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
1 → 1, R
1 → 1, R
= → =, R
Go to End
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

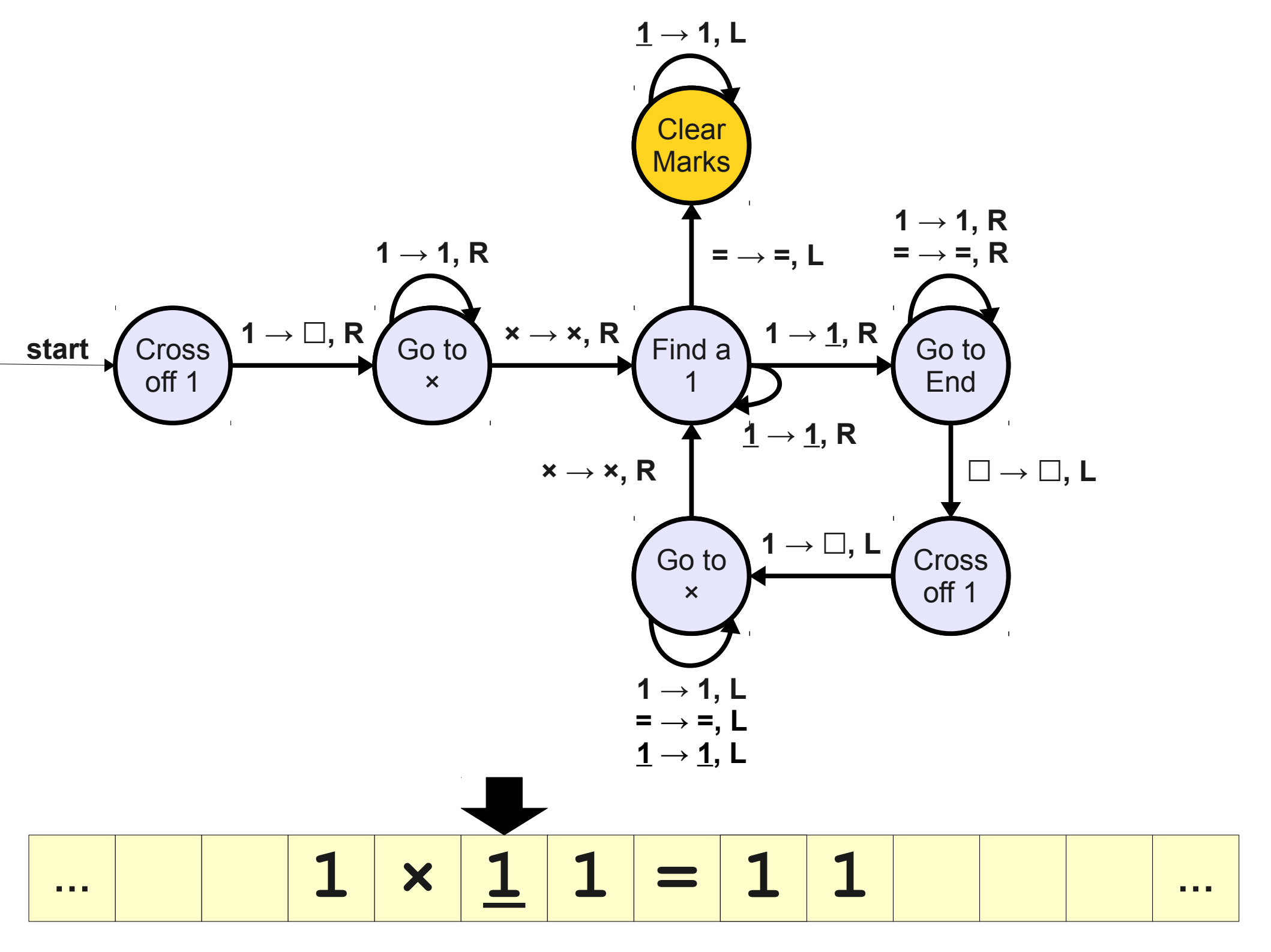

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
= → =, L
1 → 1, R
Go to End
1 → 1, R
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L
… 1 × 1 1 = 1 1 …

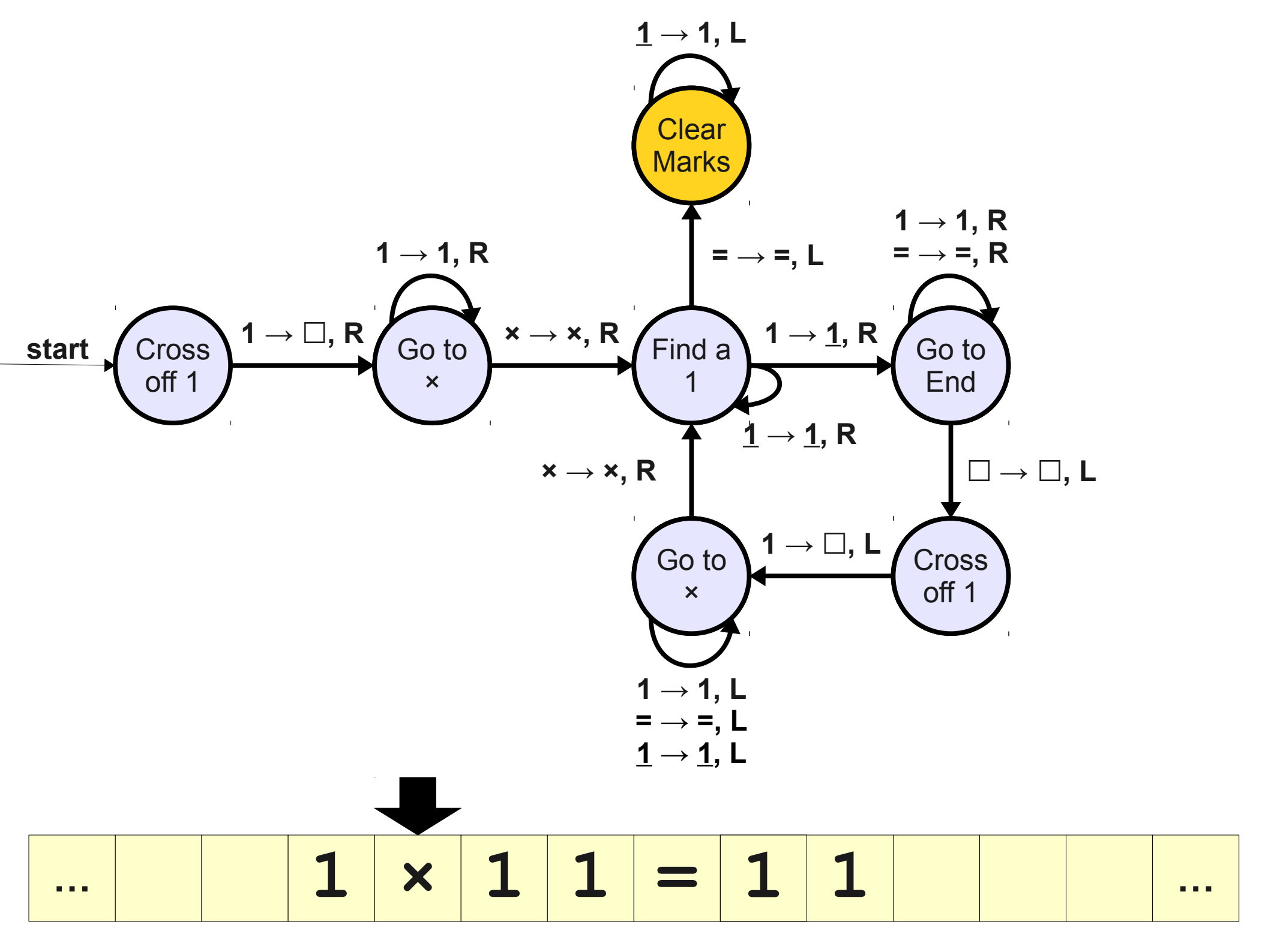

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
= → =, L
1 → 1, R
1 → 1, R
= → =, R
Go to End
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L

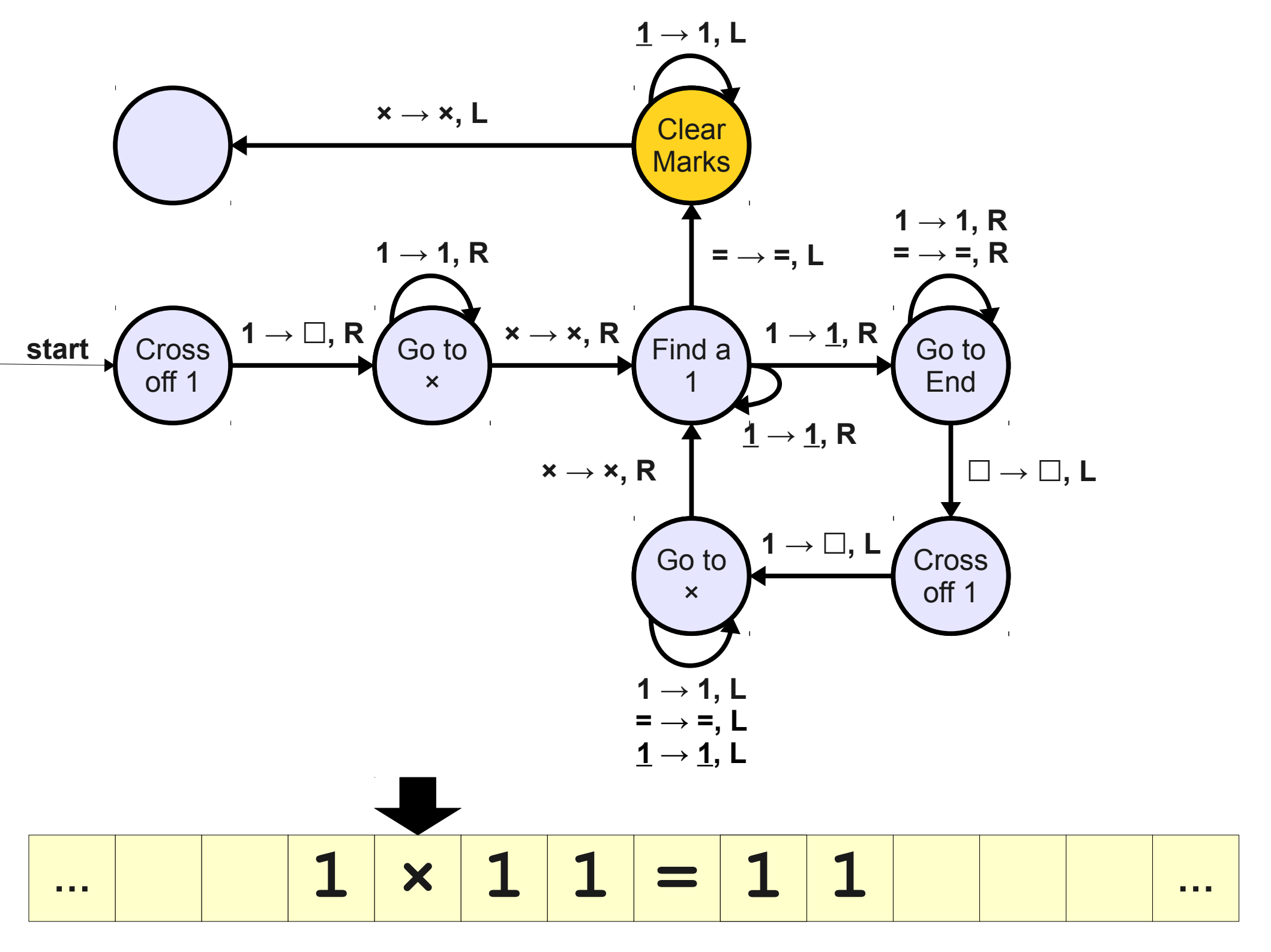

...
1 × 1 1 = 1 1
...

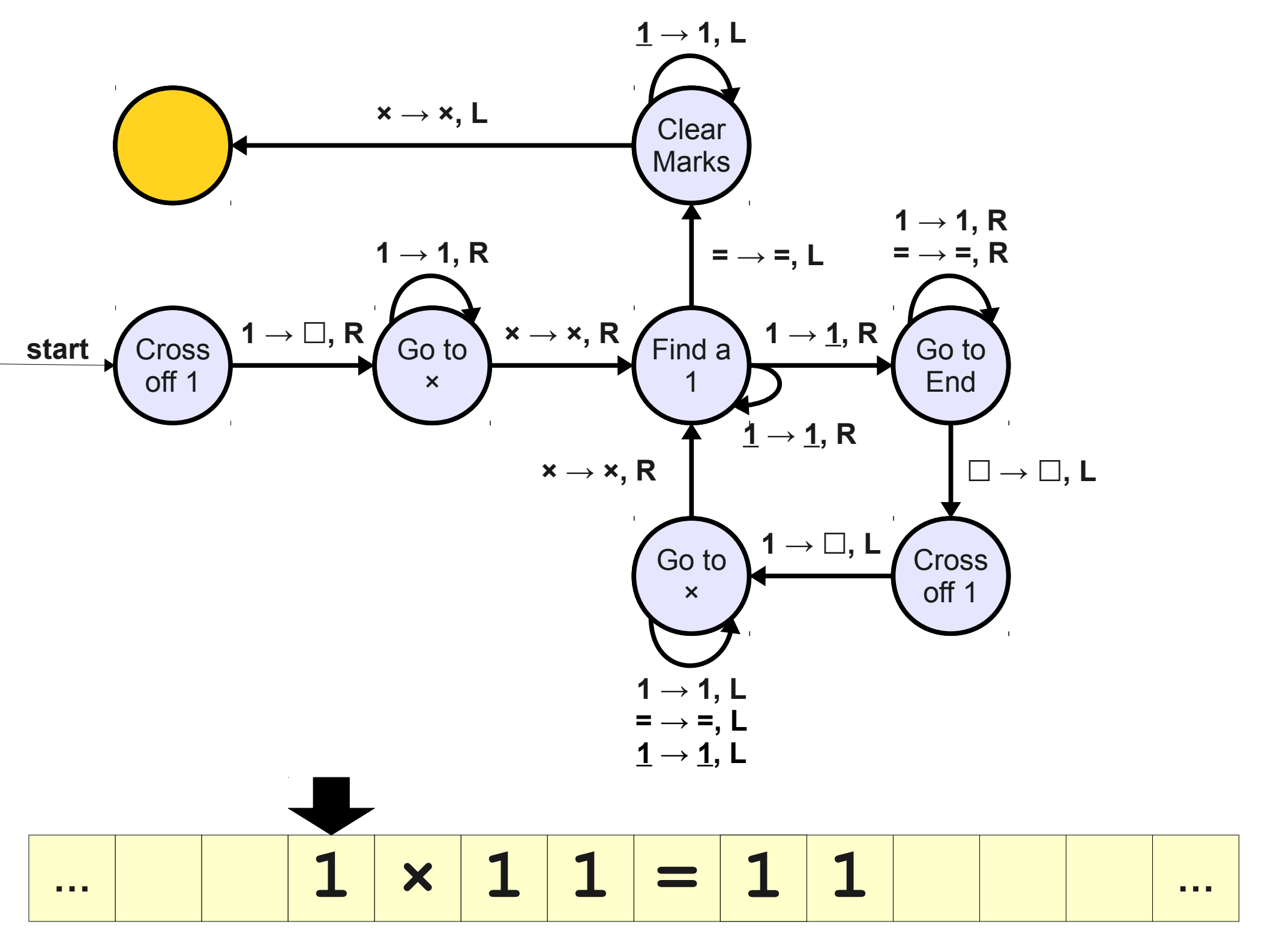

Clear Marks
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

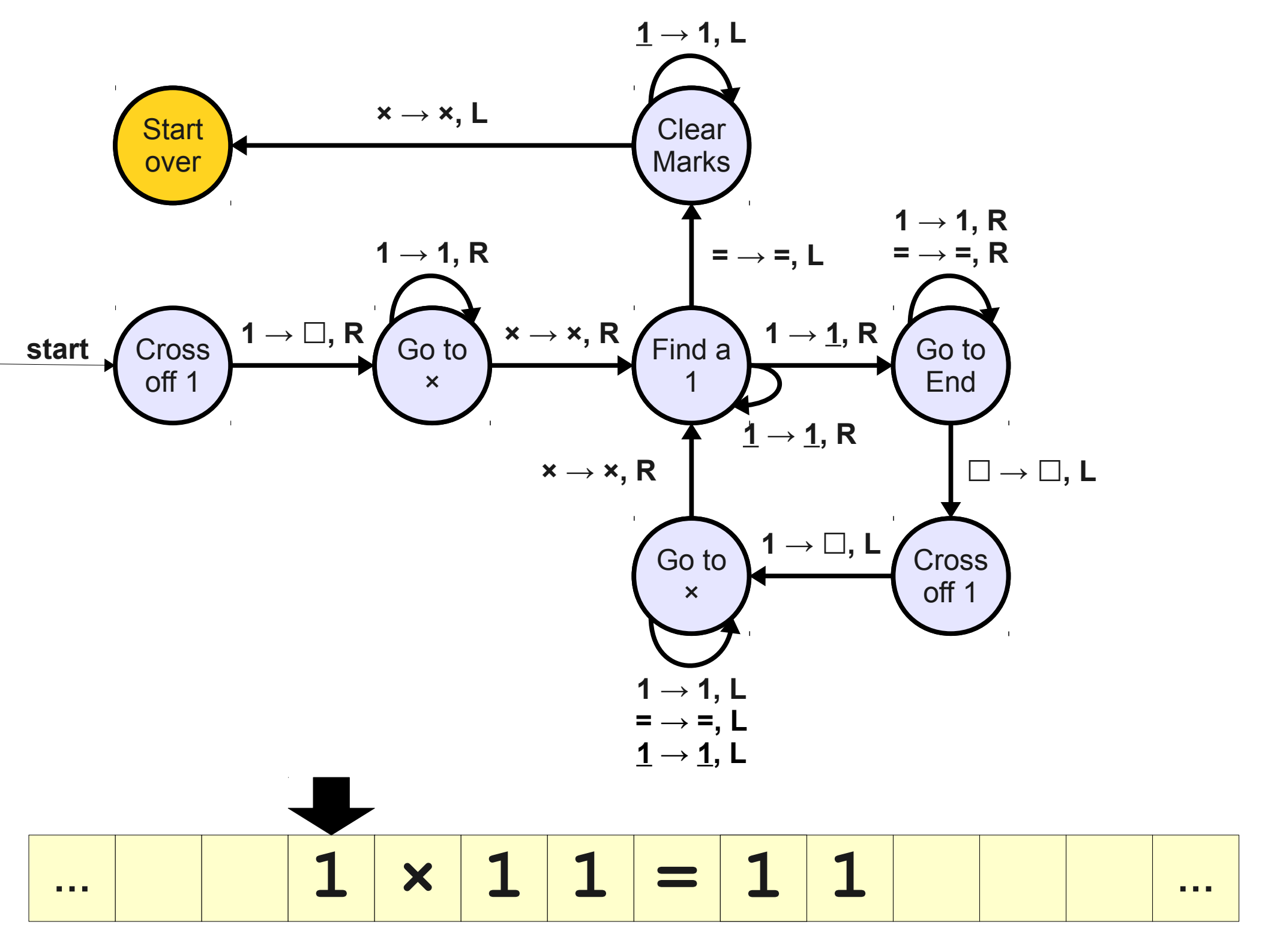

1 → 1, L
Clear Marks
= → =, L
1 → 1, R
= → =, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

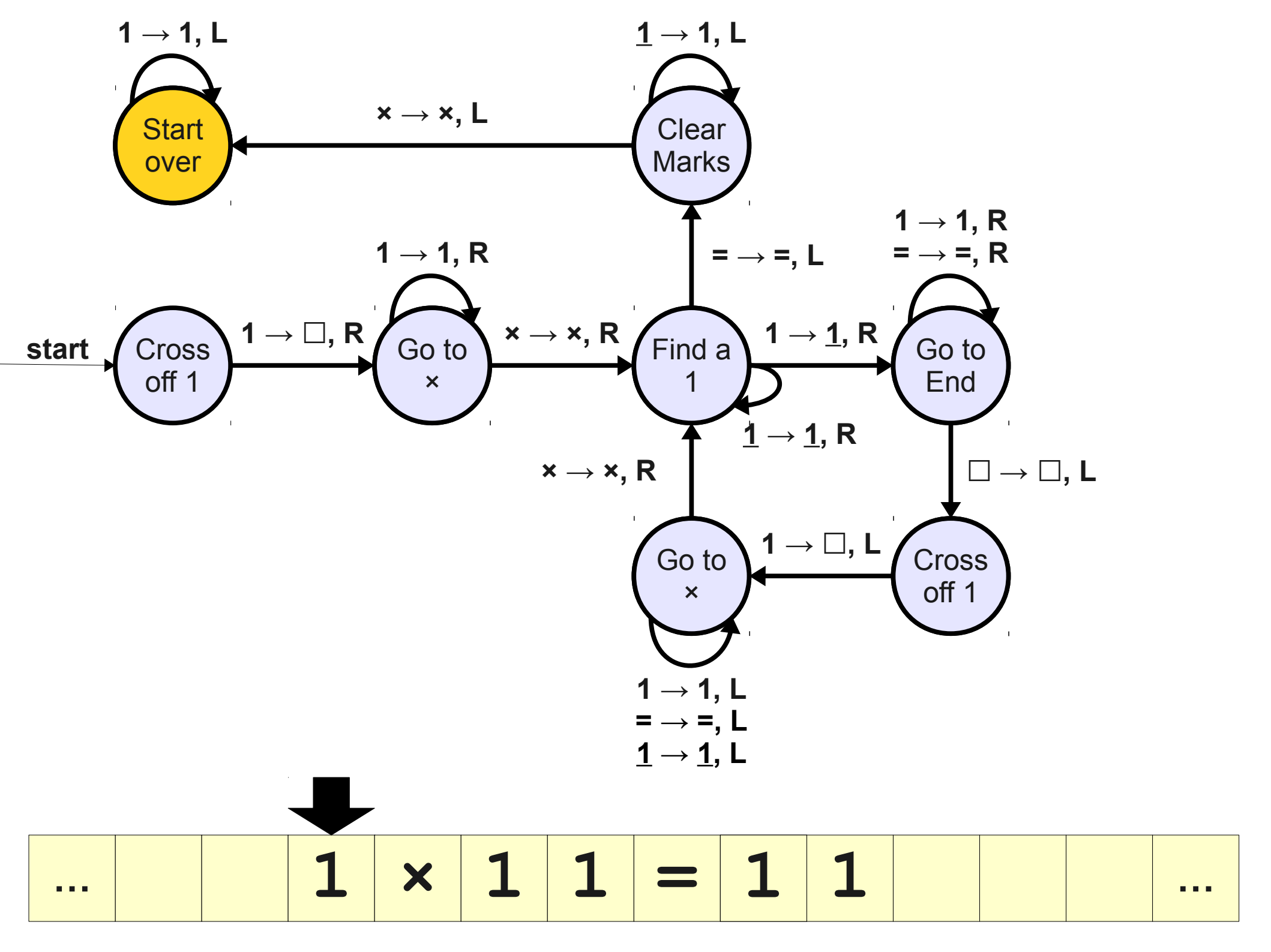

1 → 1, L
Clear Marks
= → =, L
1 → 1, R
= → =, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

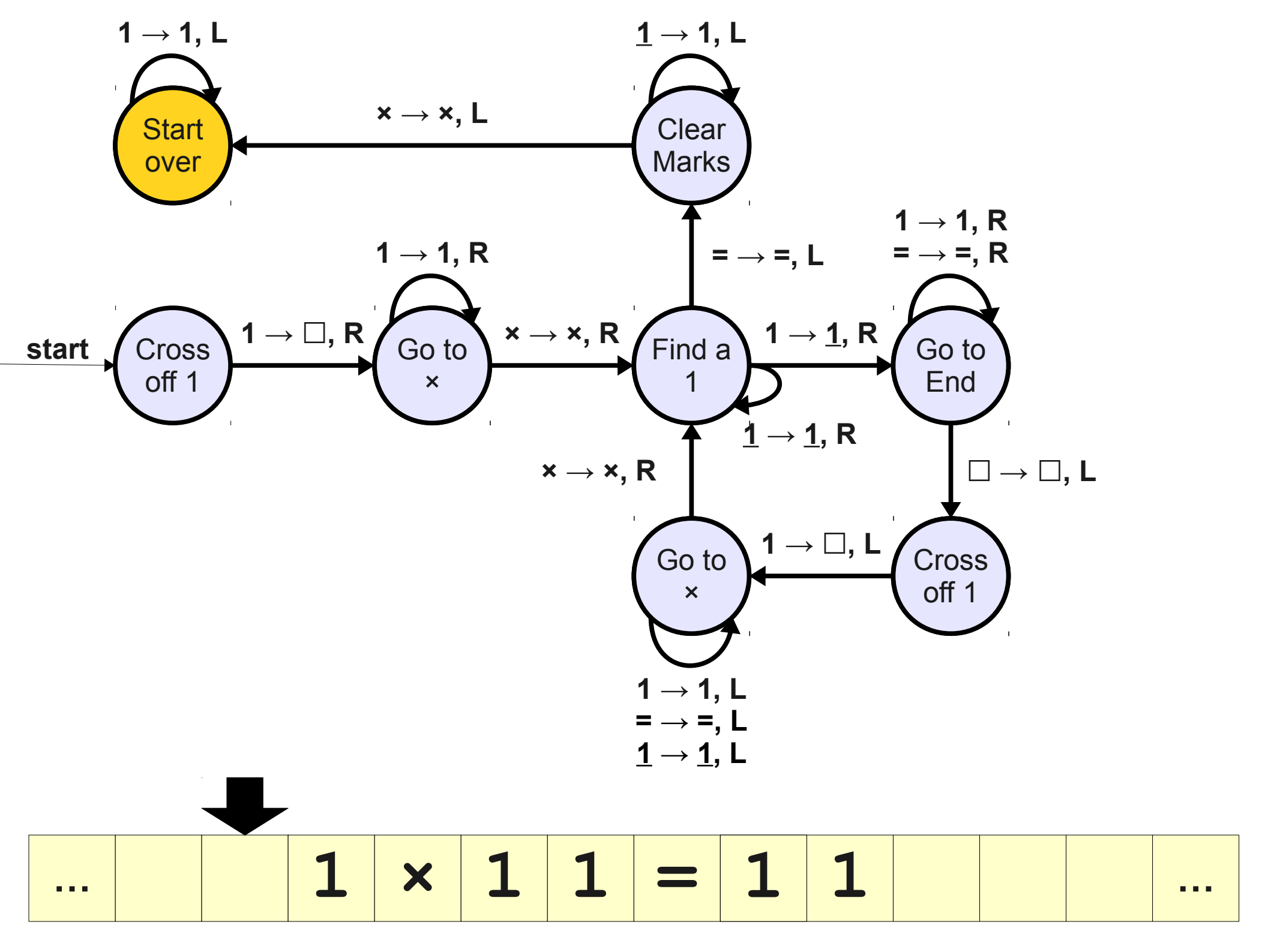

1 → 1, L
Clear Marks
= → =, L
1 → 1, R
= → =, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

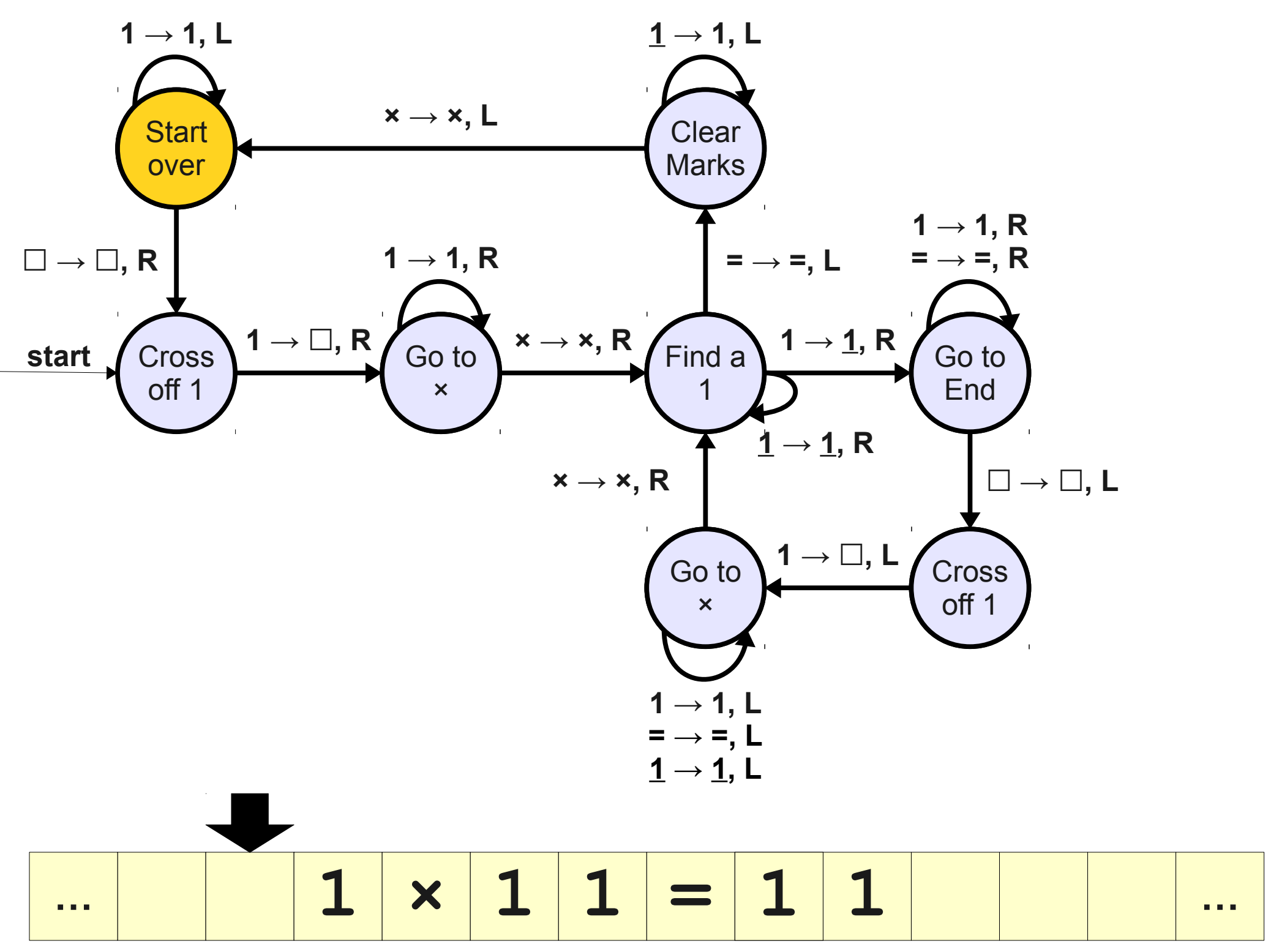

1 → 1, L
× → ×, L
Clear Marks
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

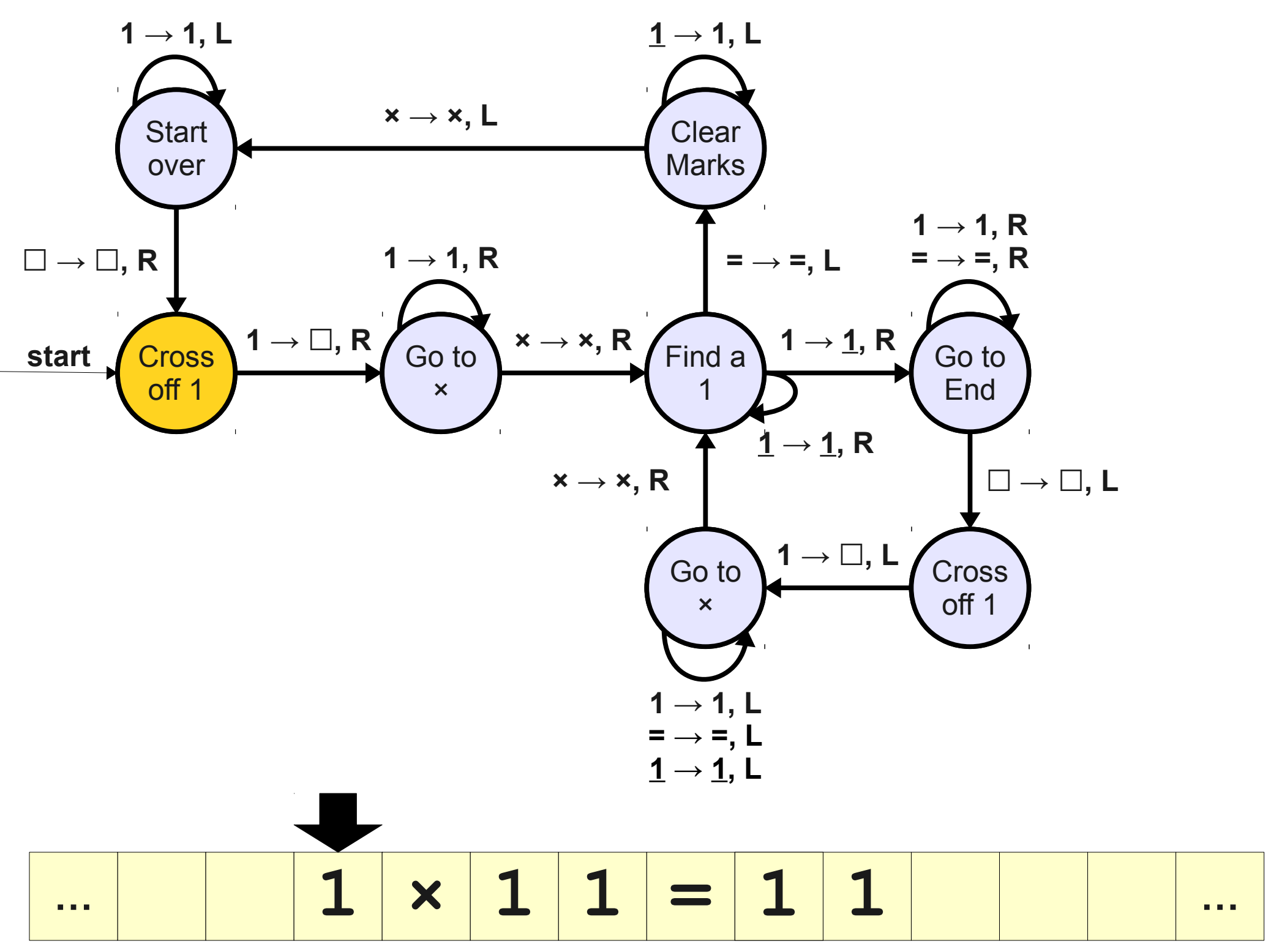

1 → 1, L
× → ×, L
Clear Marks
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

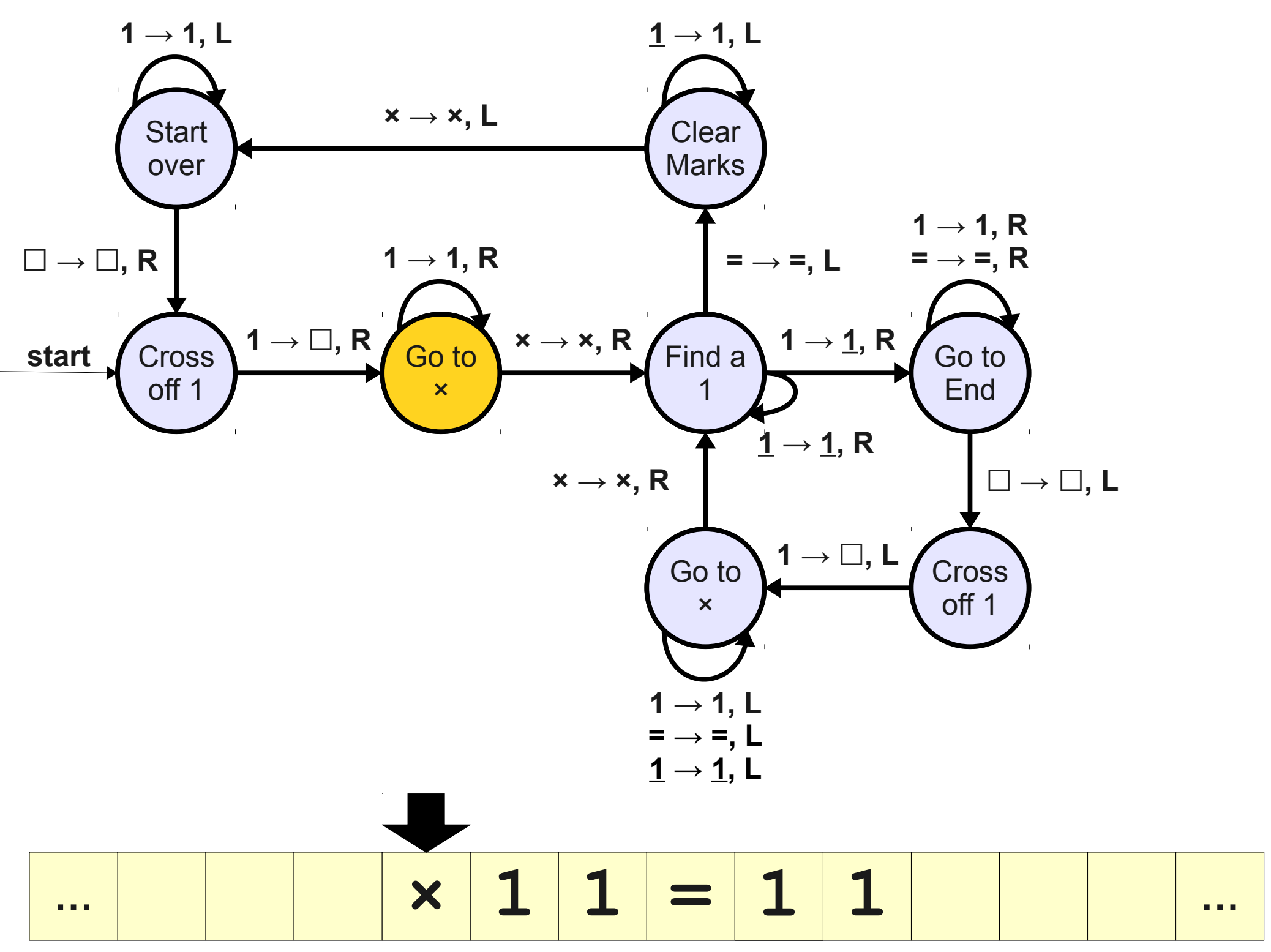

start
Cross off 1
1 → □, R
Go to ×
1 → 1, R
× → ×, R
Find a 1
= → =, L
Clear Marks
× → ×, L
Start over
1 → 1, L
1 → 1, R
Go to End
= → =, R
1 → 1, R
□ → □, L
Cross off 1
1 → □, L
Go to ×
× → ×, R
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

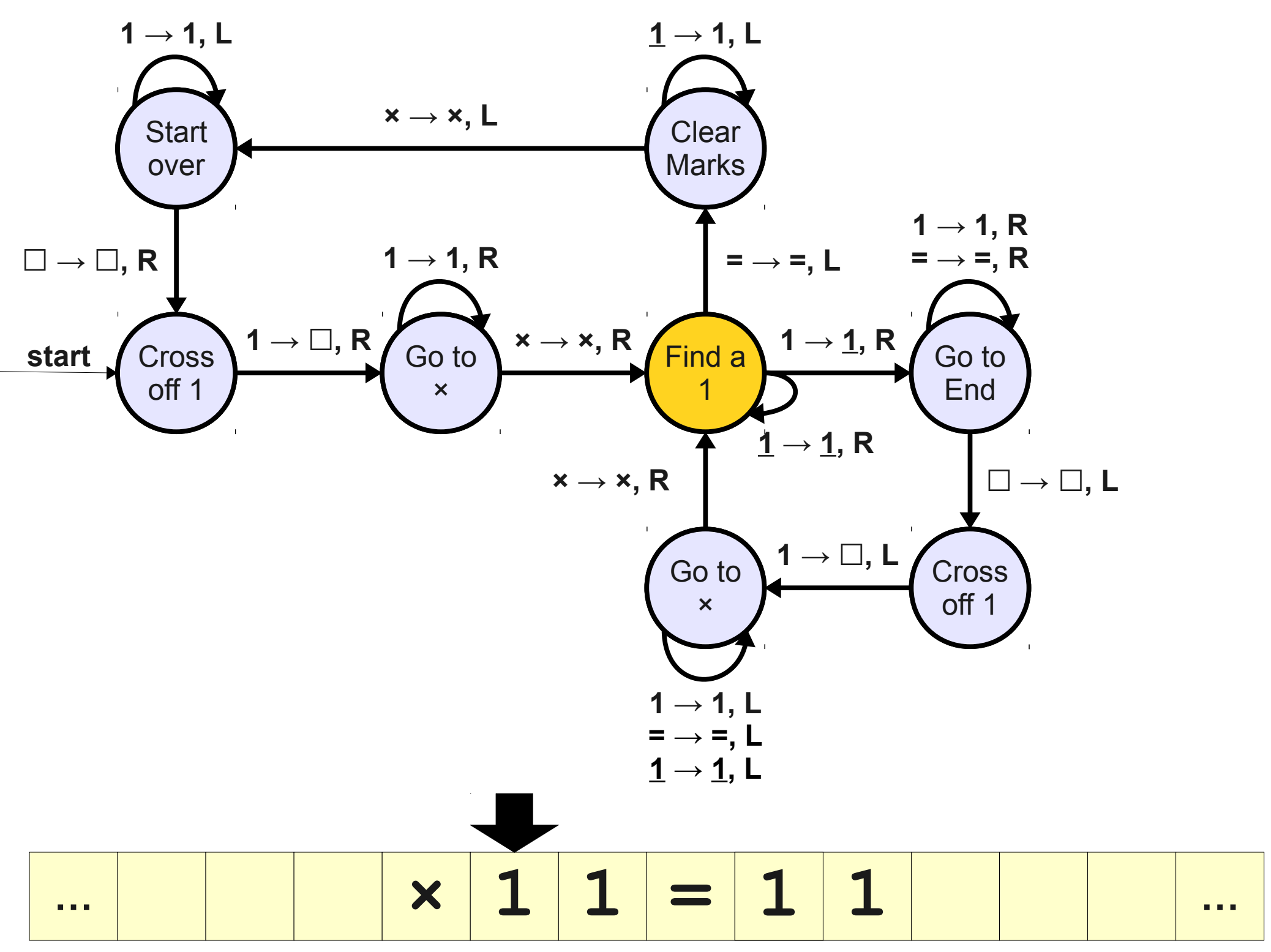

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
= → =, L
1 → 1, R
= → =, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

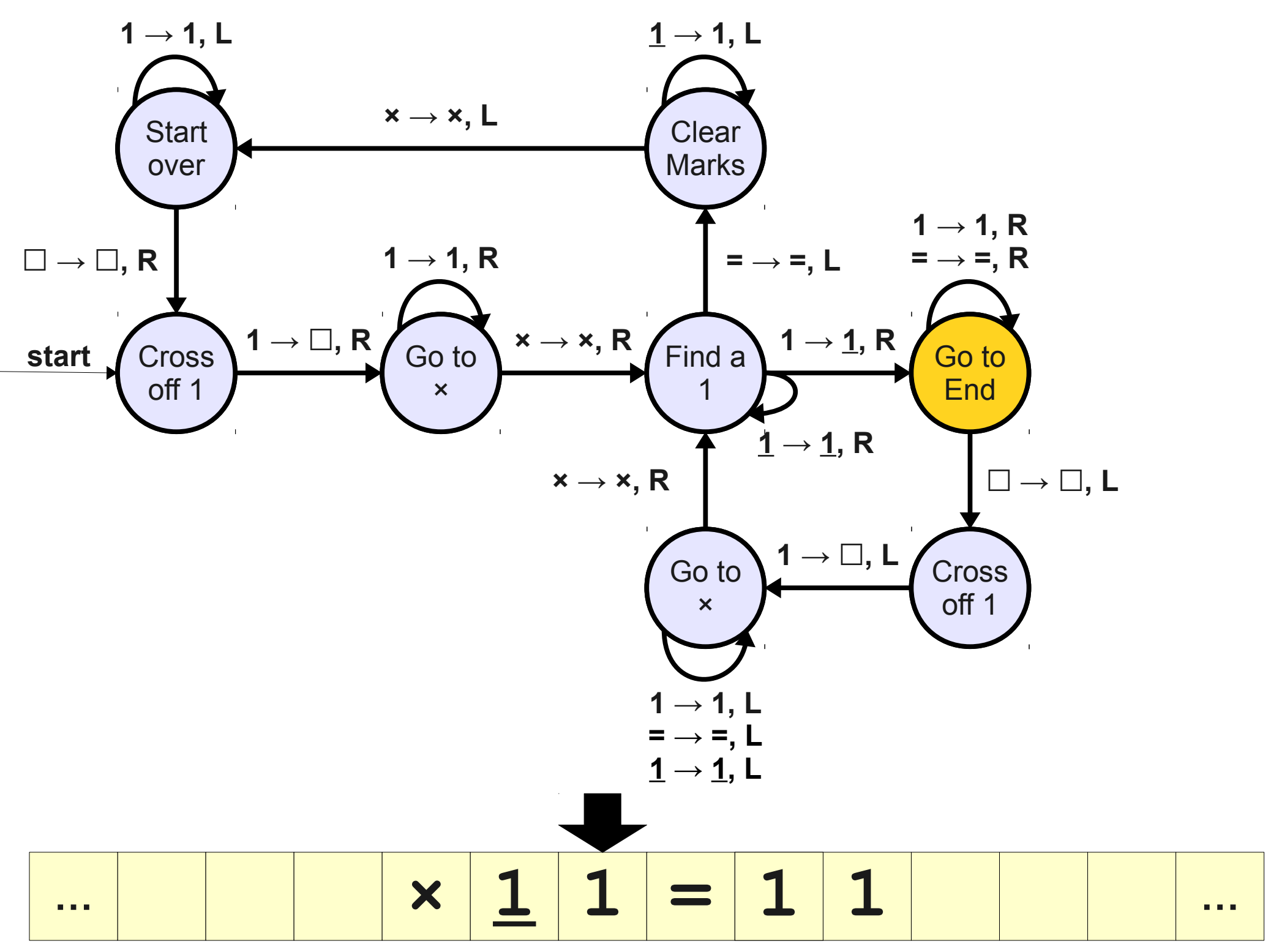

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
= → =, L
1 → 1, R
= → =, R
1 → 1, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

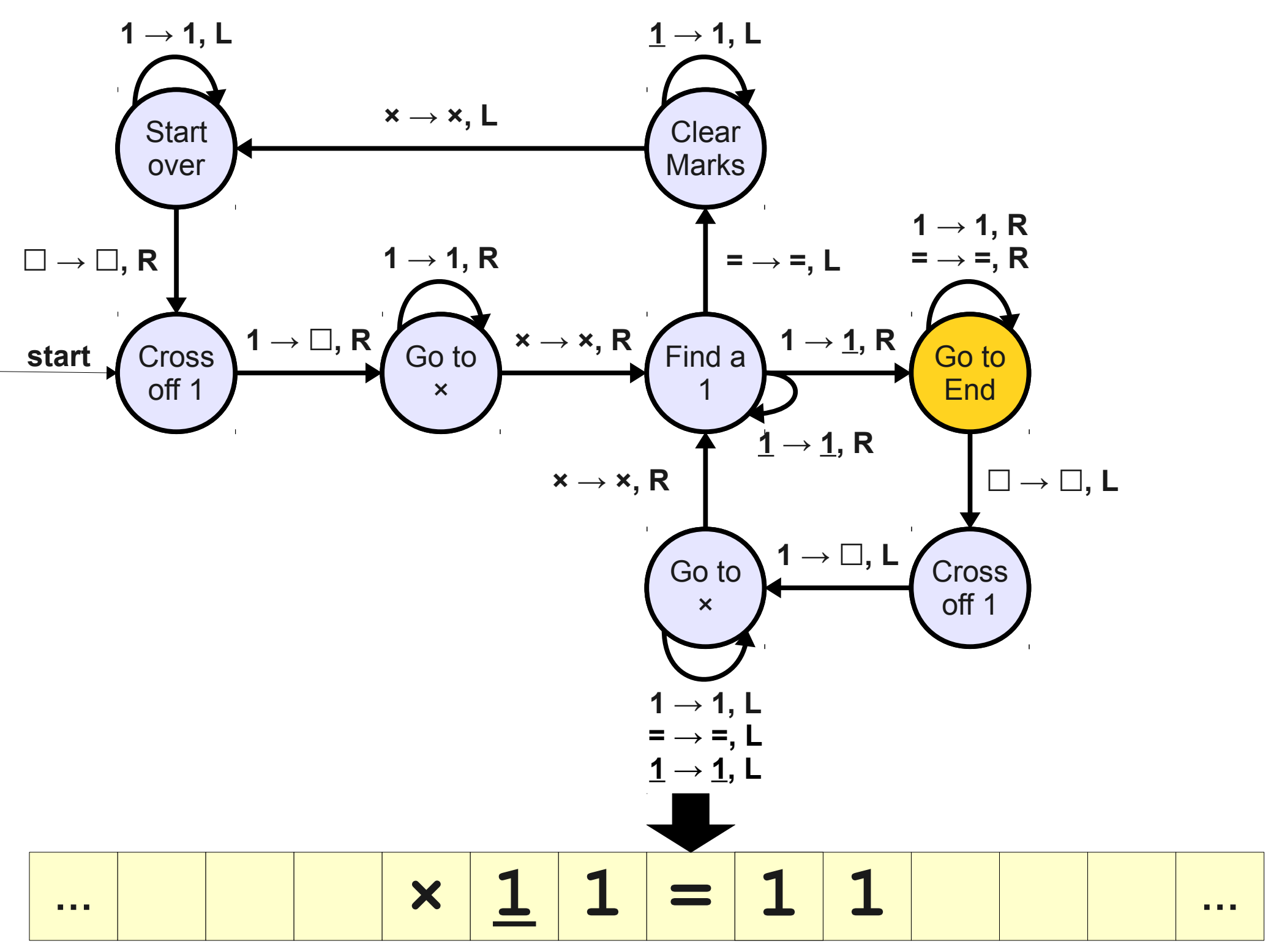

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

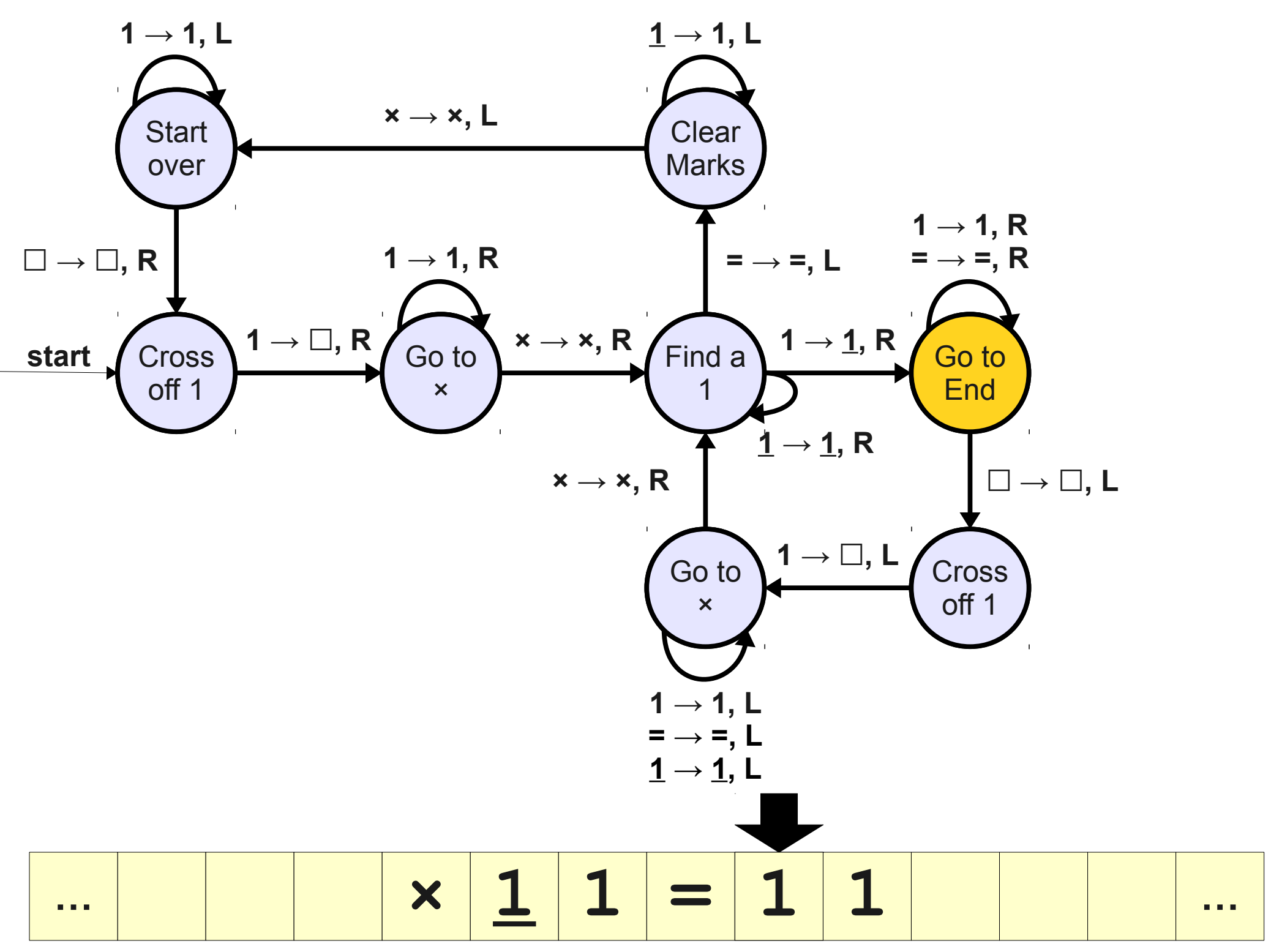

1 → 1, L
1 → 1, L
Start over
× → ×, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 1 ...

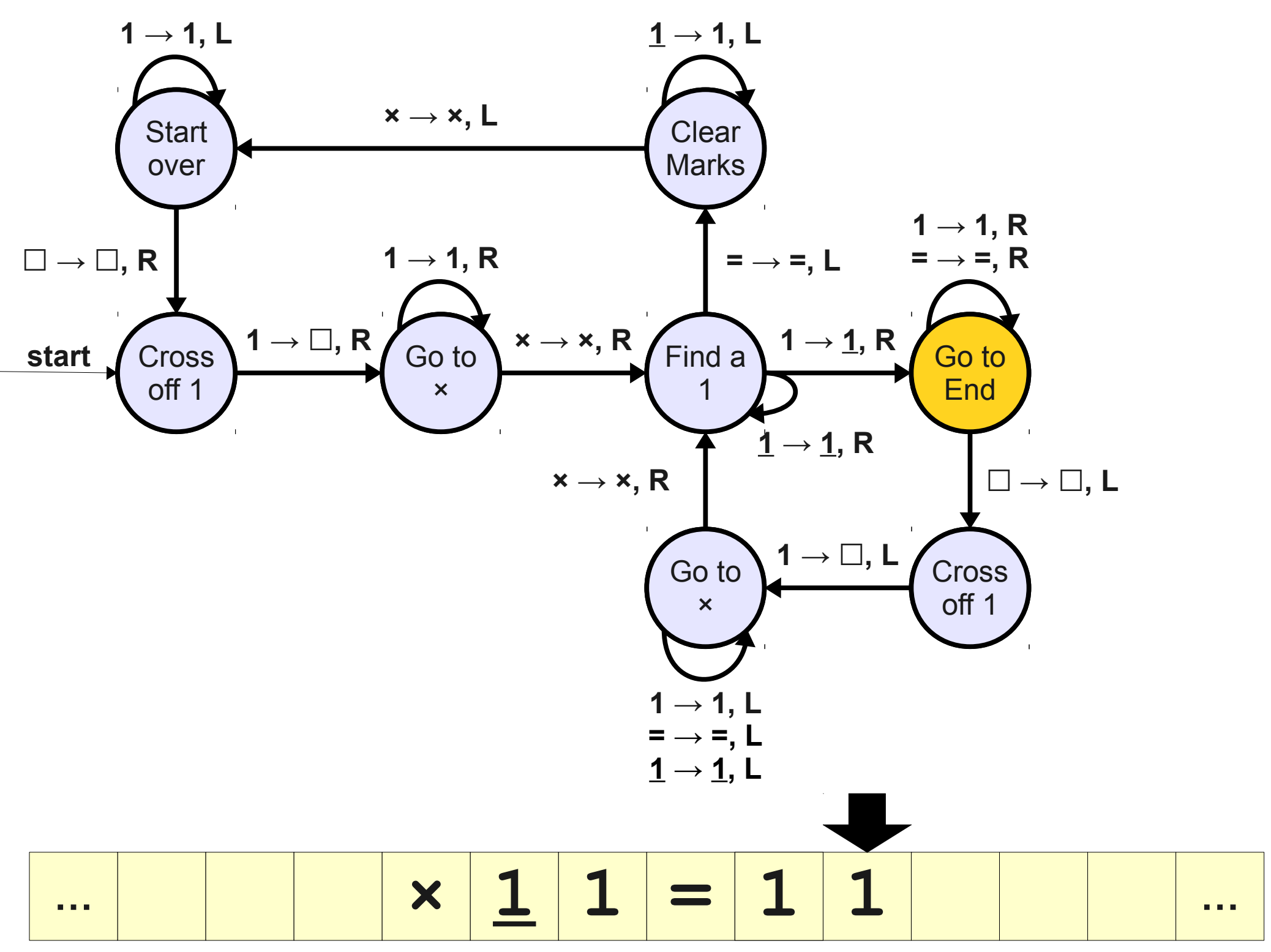

1 → 1, L
1 → 1, L
Start over
× → ×, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

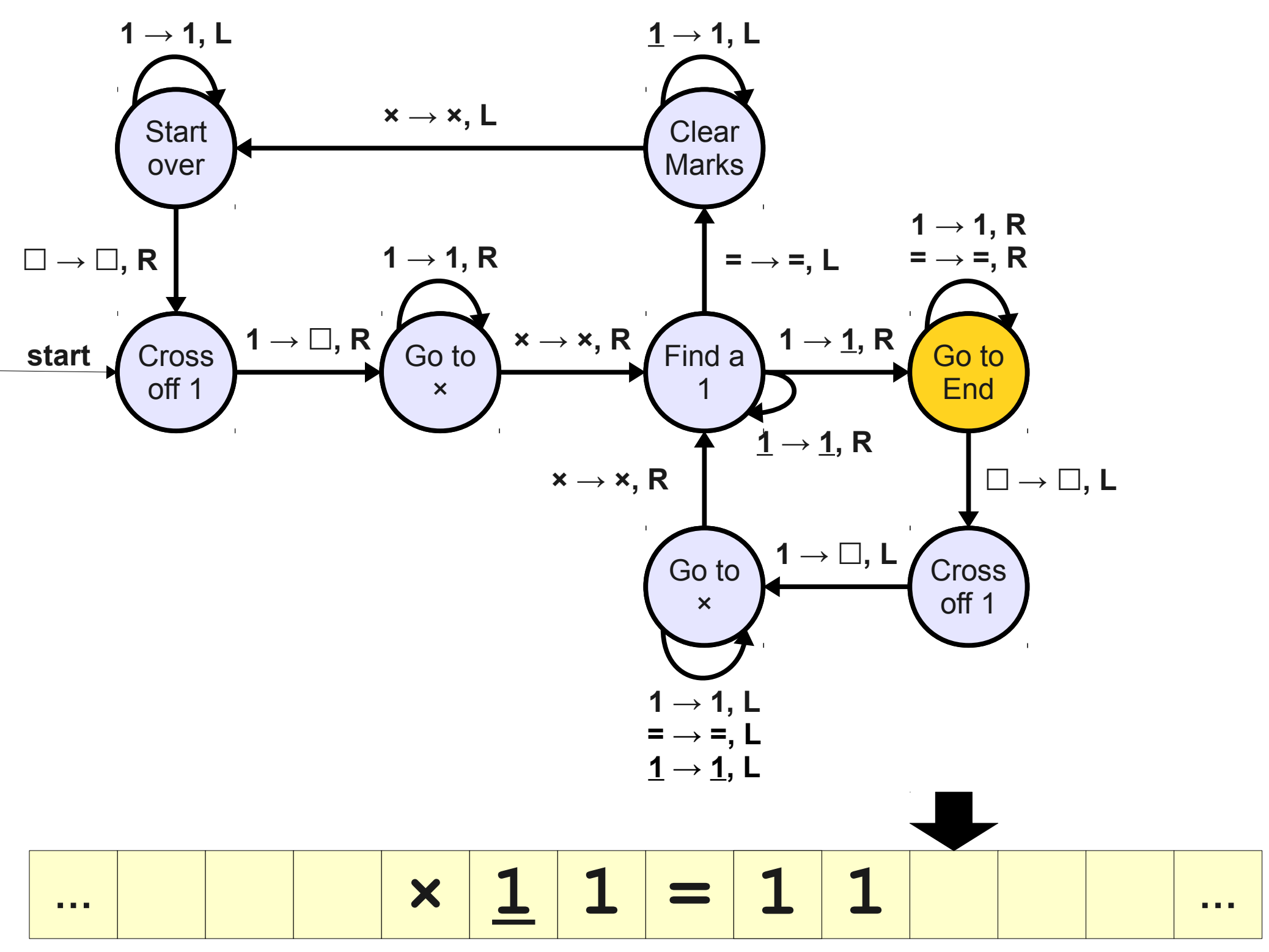

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

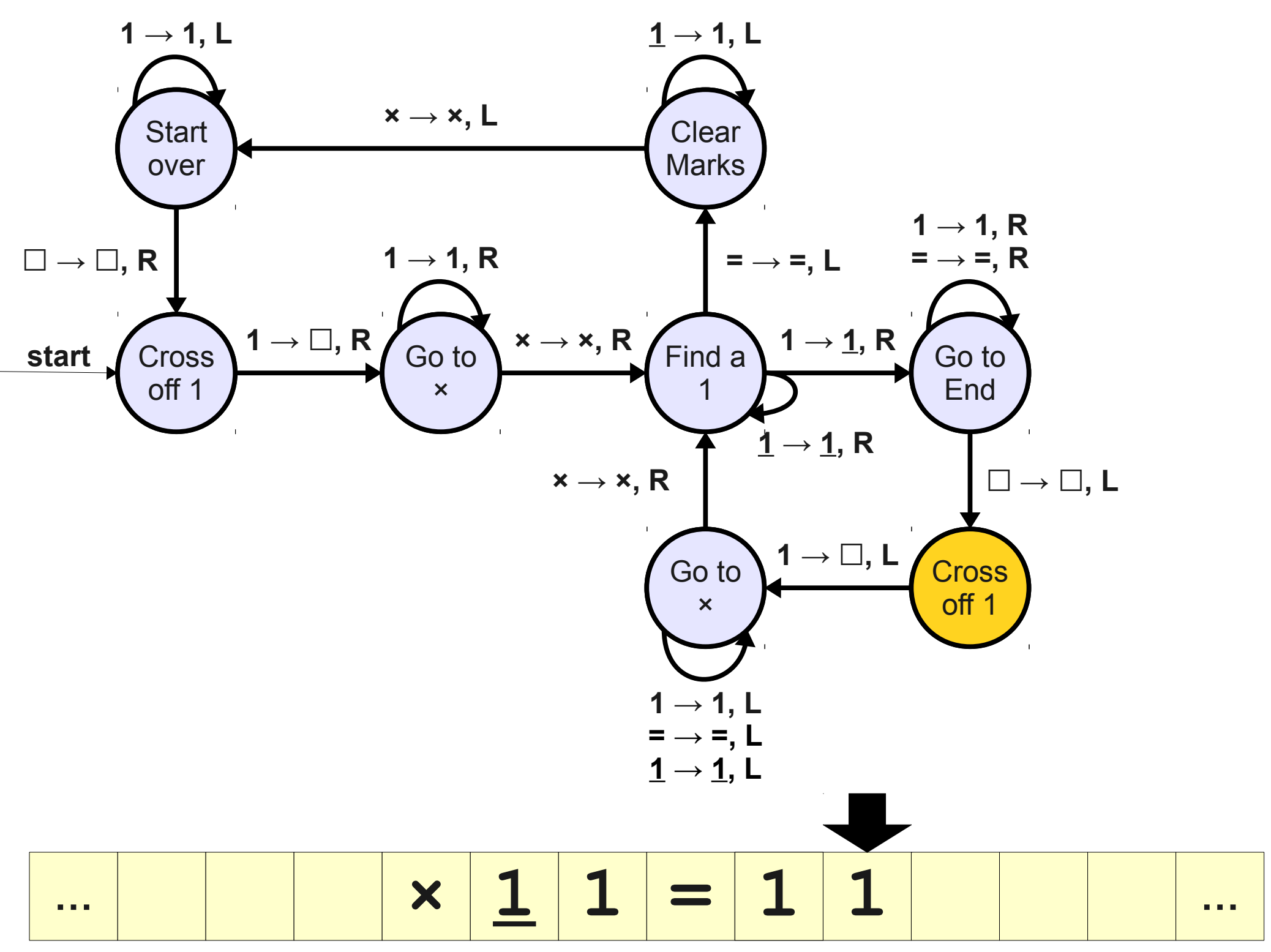

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

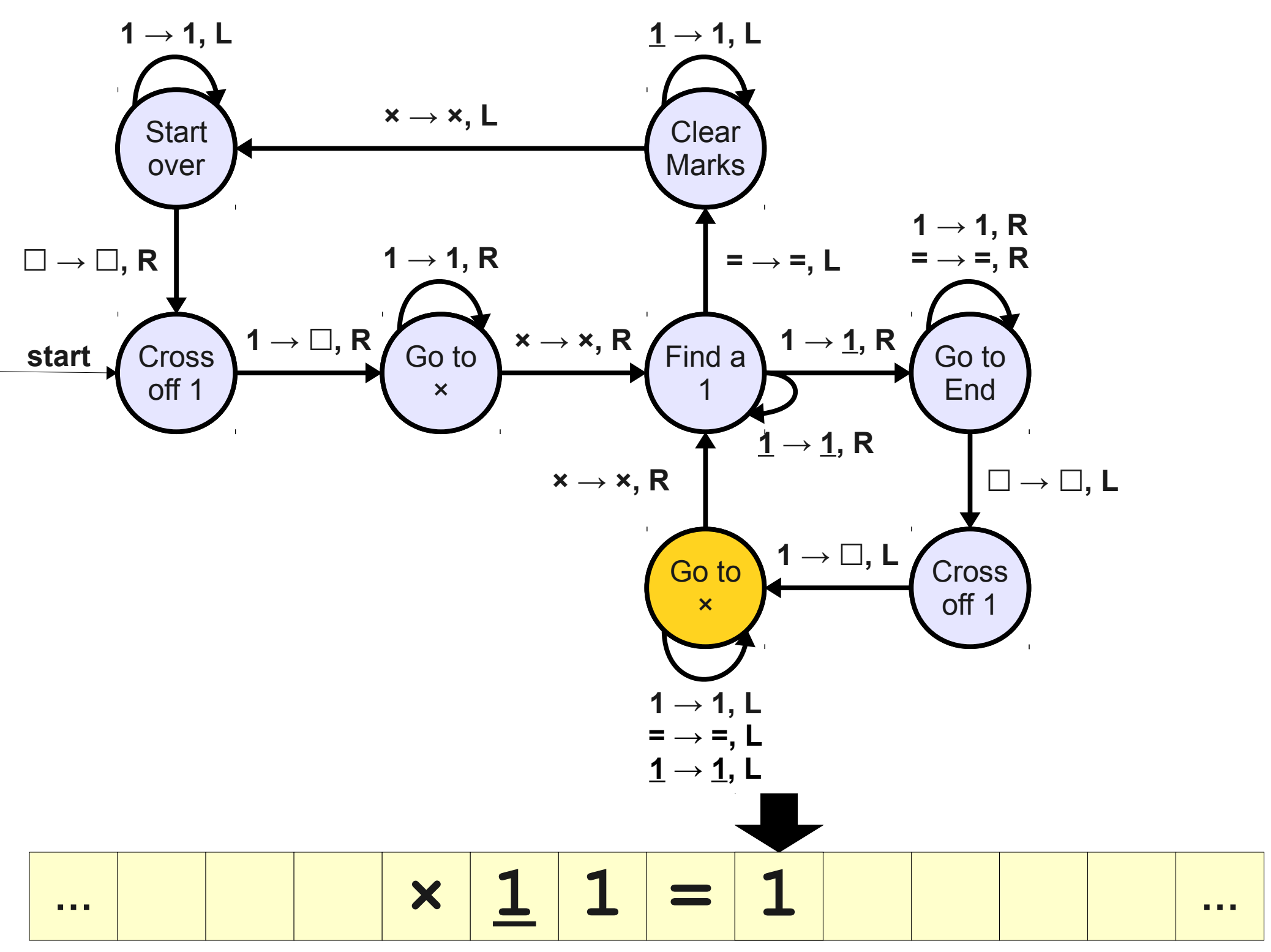

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

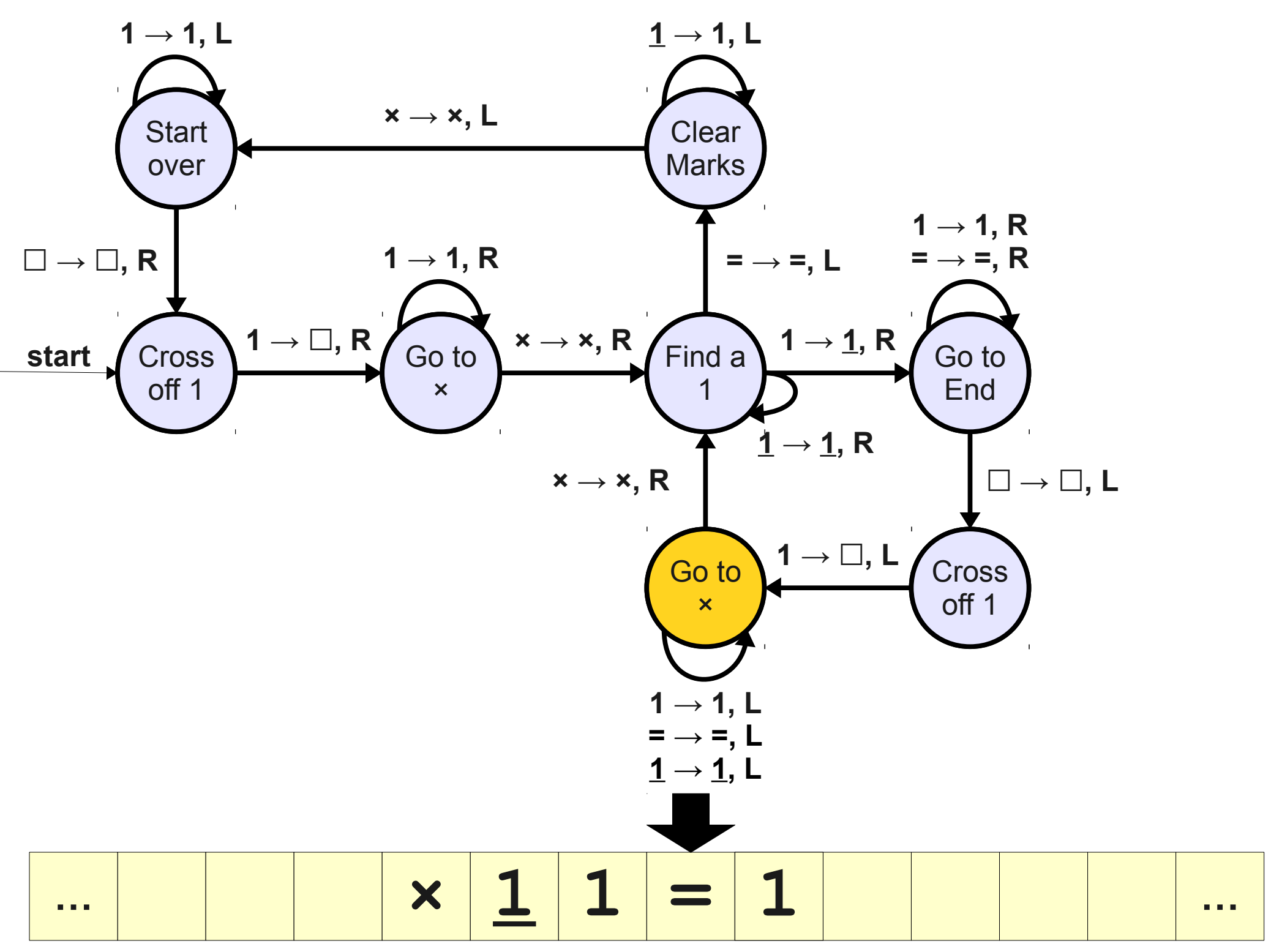

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

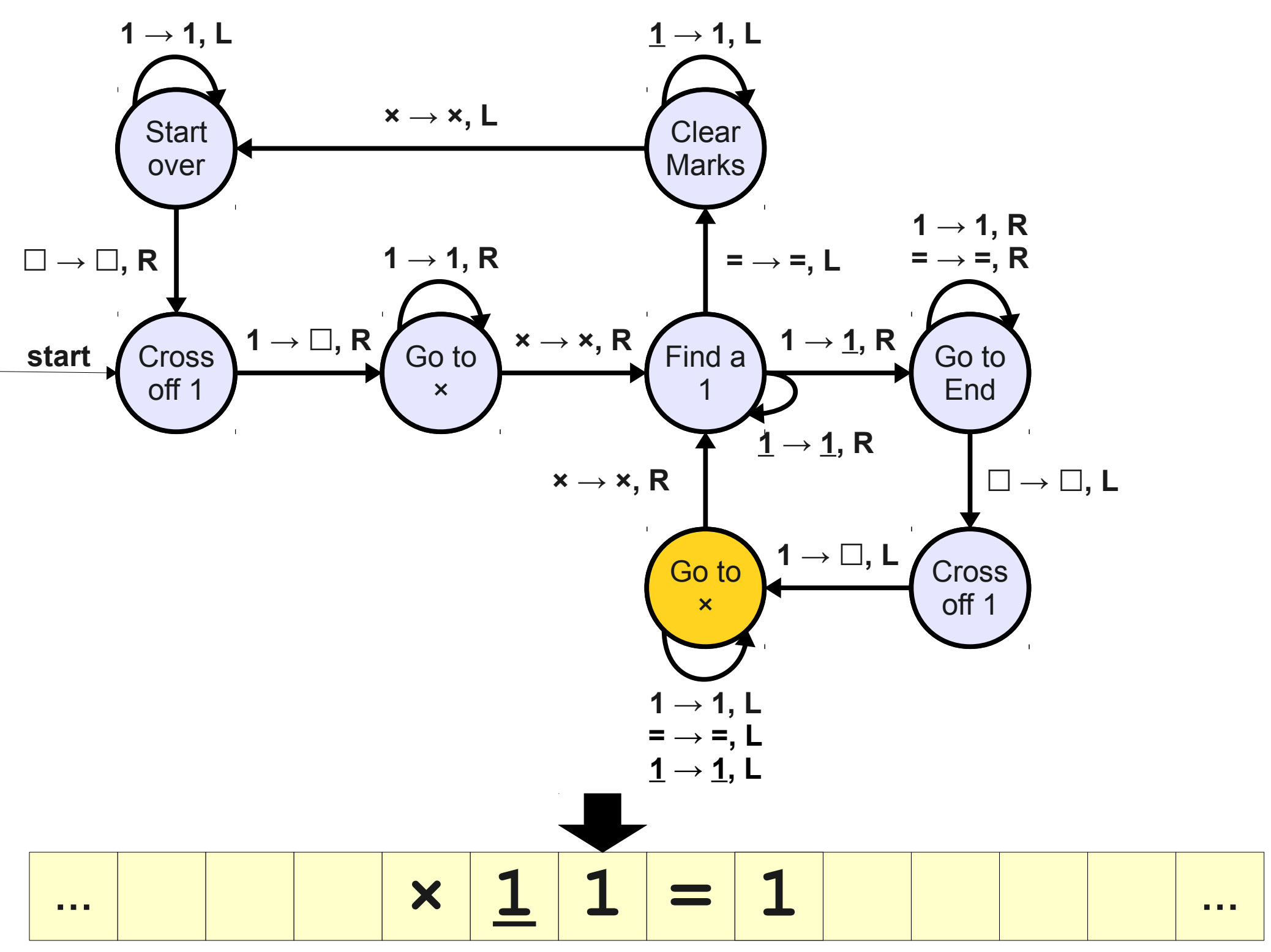

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

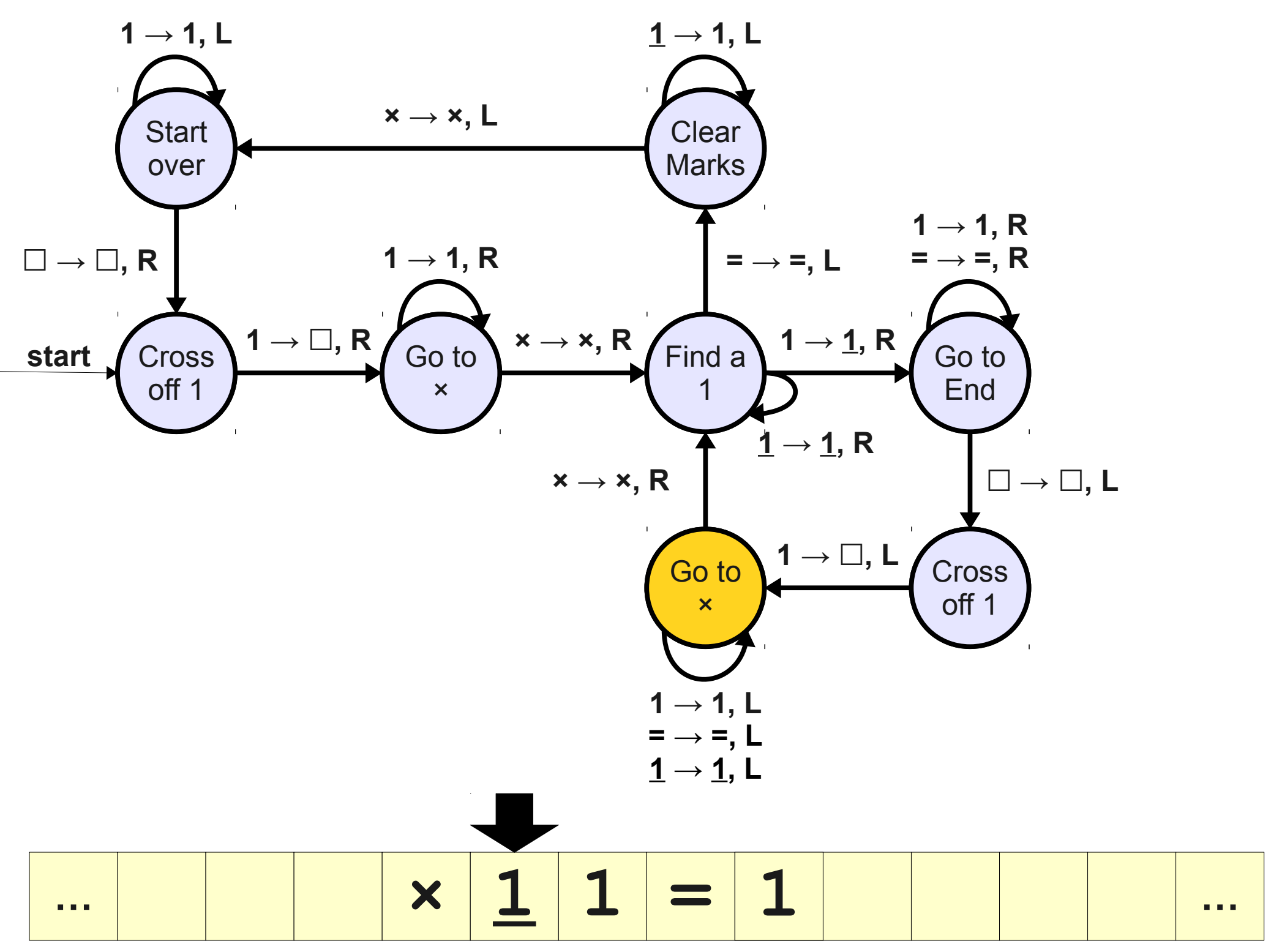

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

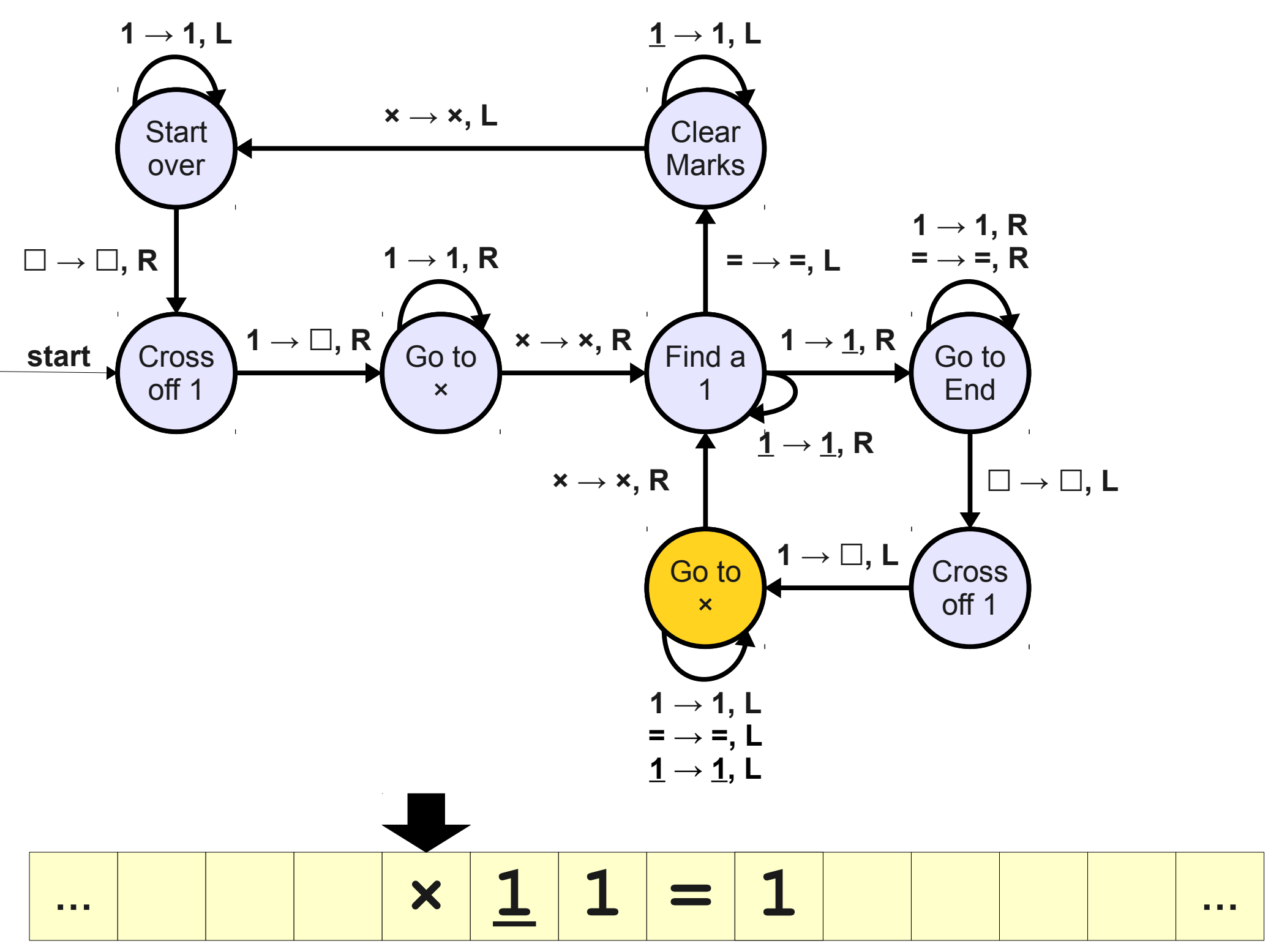

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 1 ...

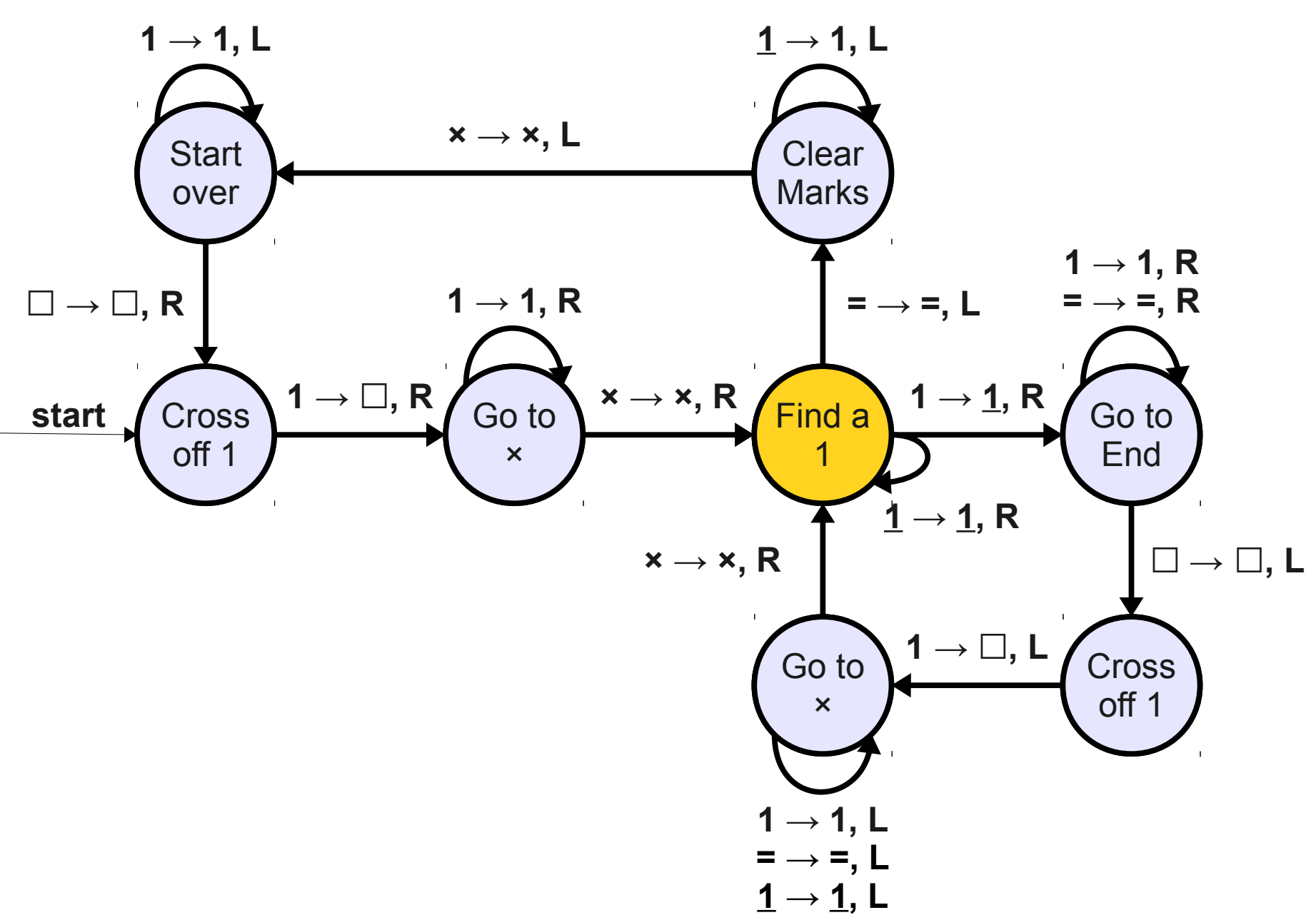

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

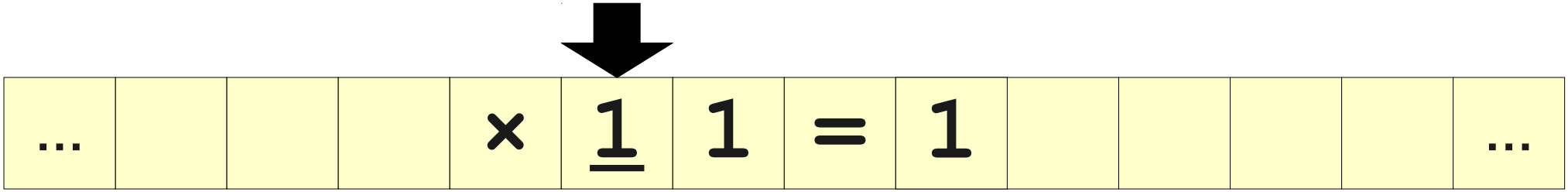

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

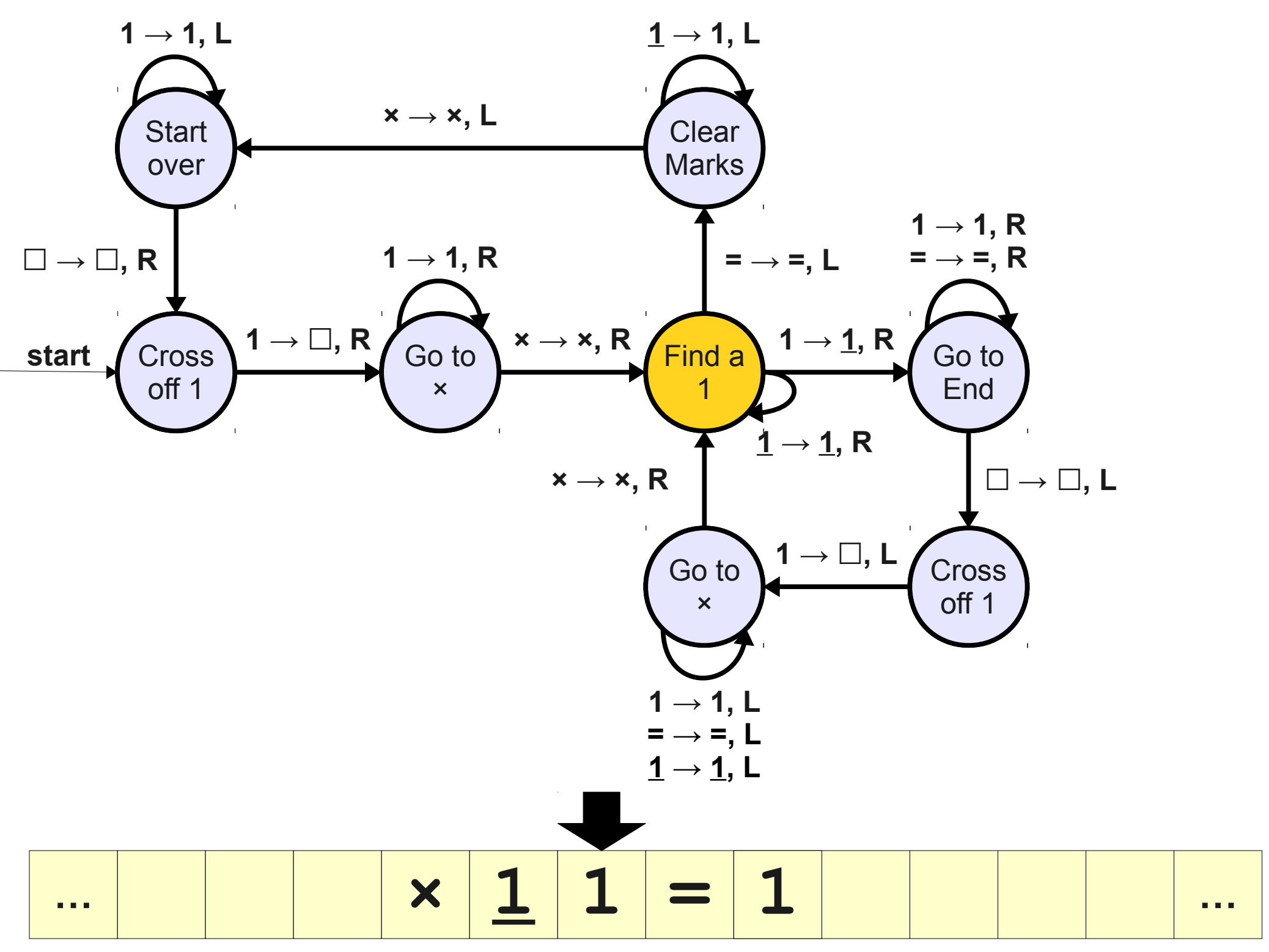

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

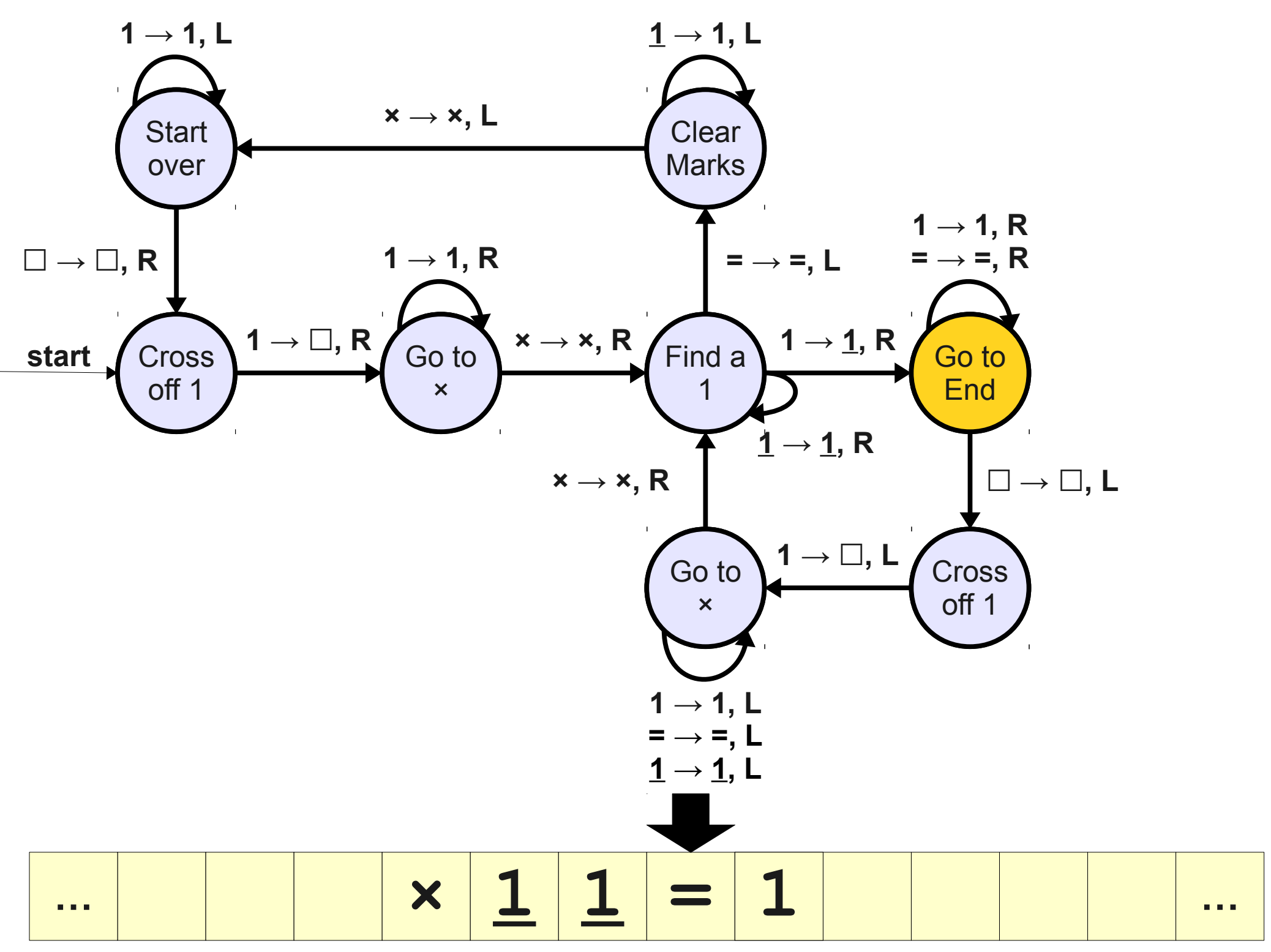

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

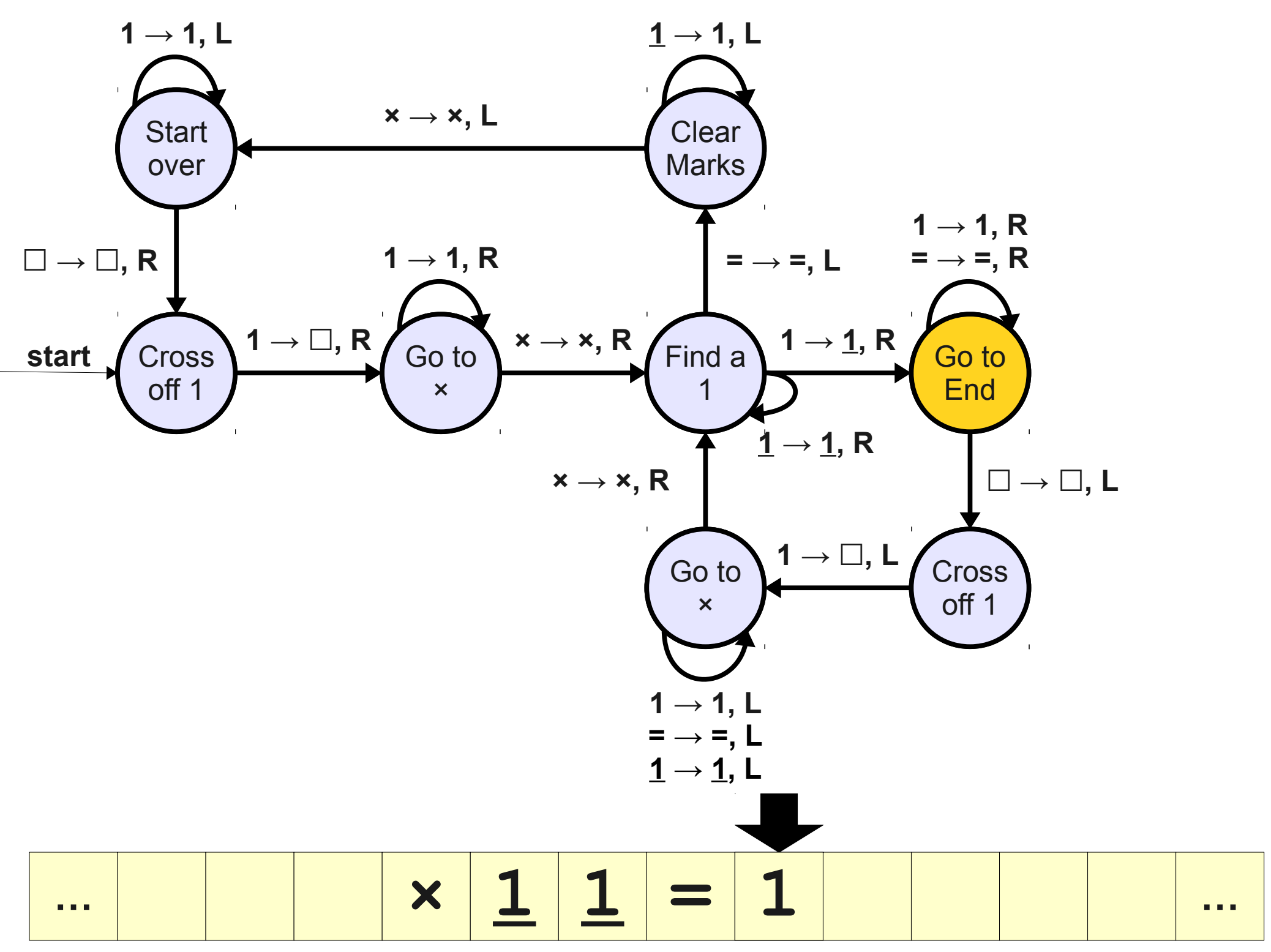

1 → 1, L
1 → 1, L
Start over
× → ×, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

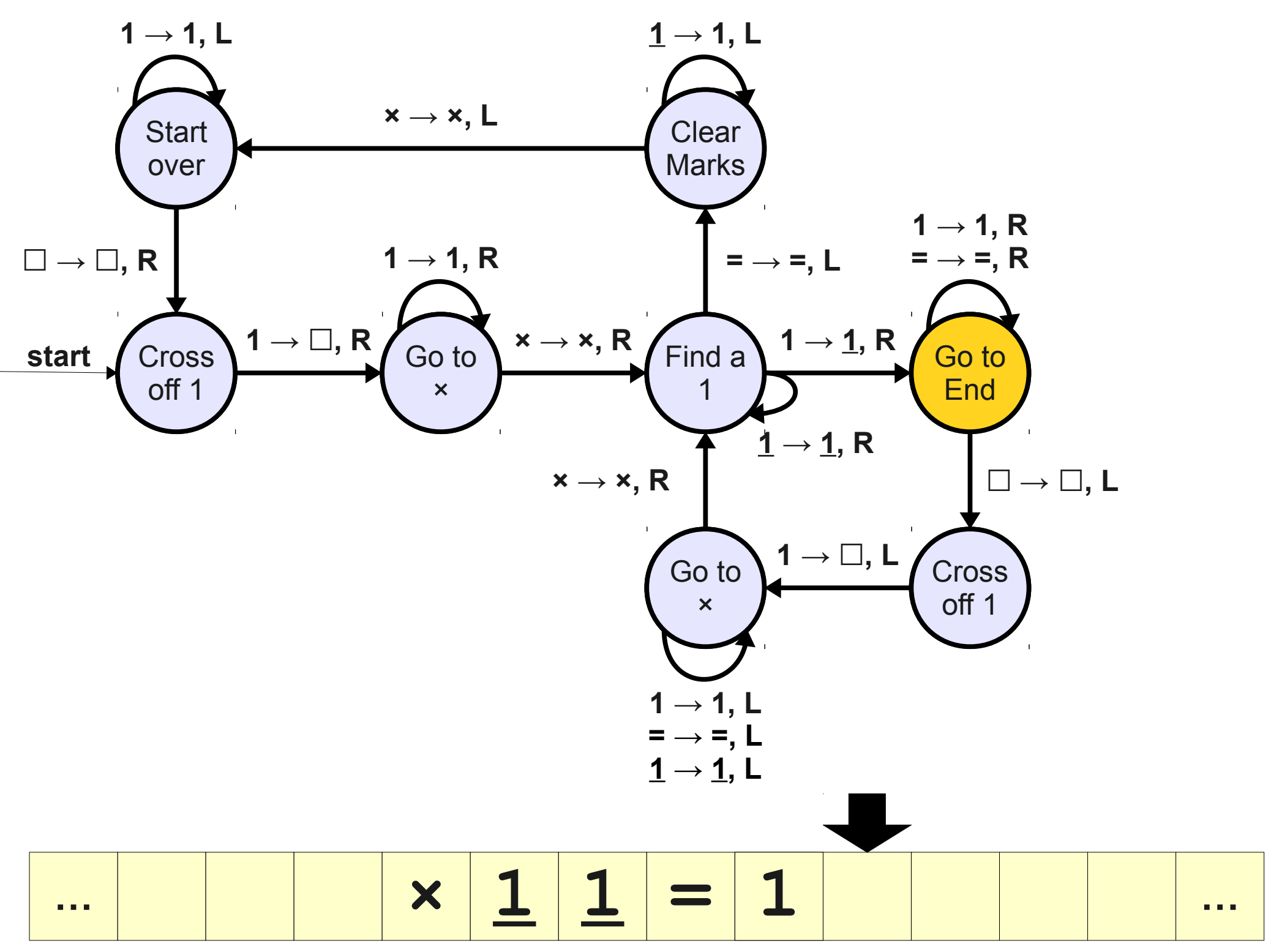

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L

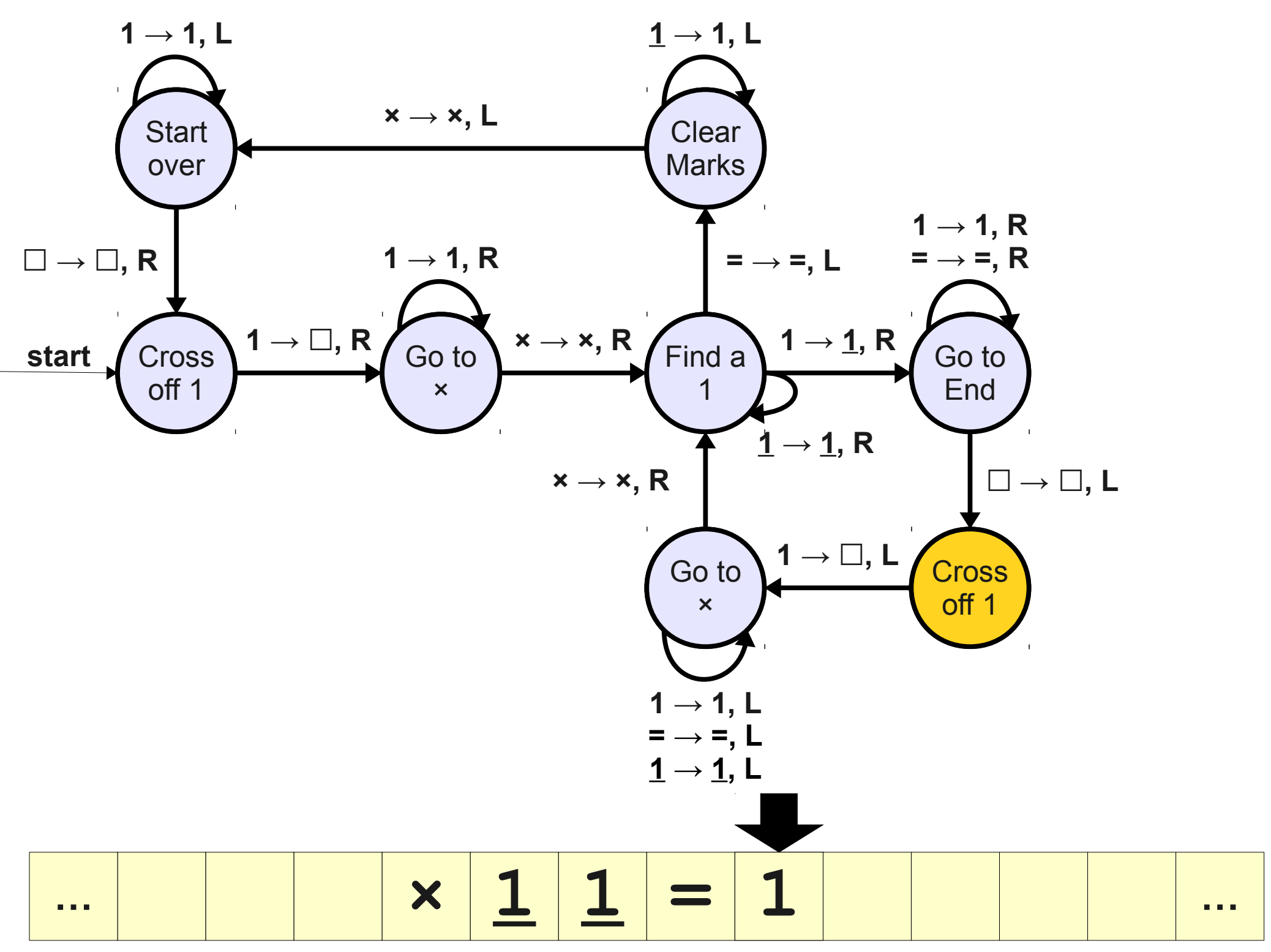

...
×
1
1
=
1
...

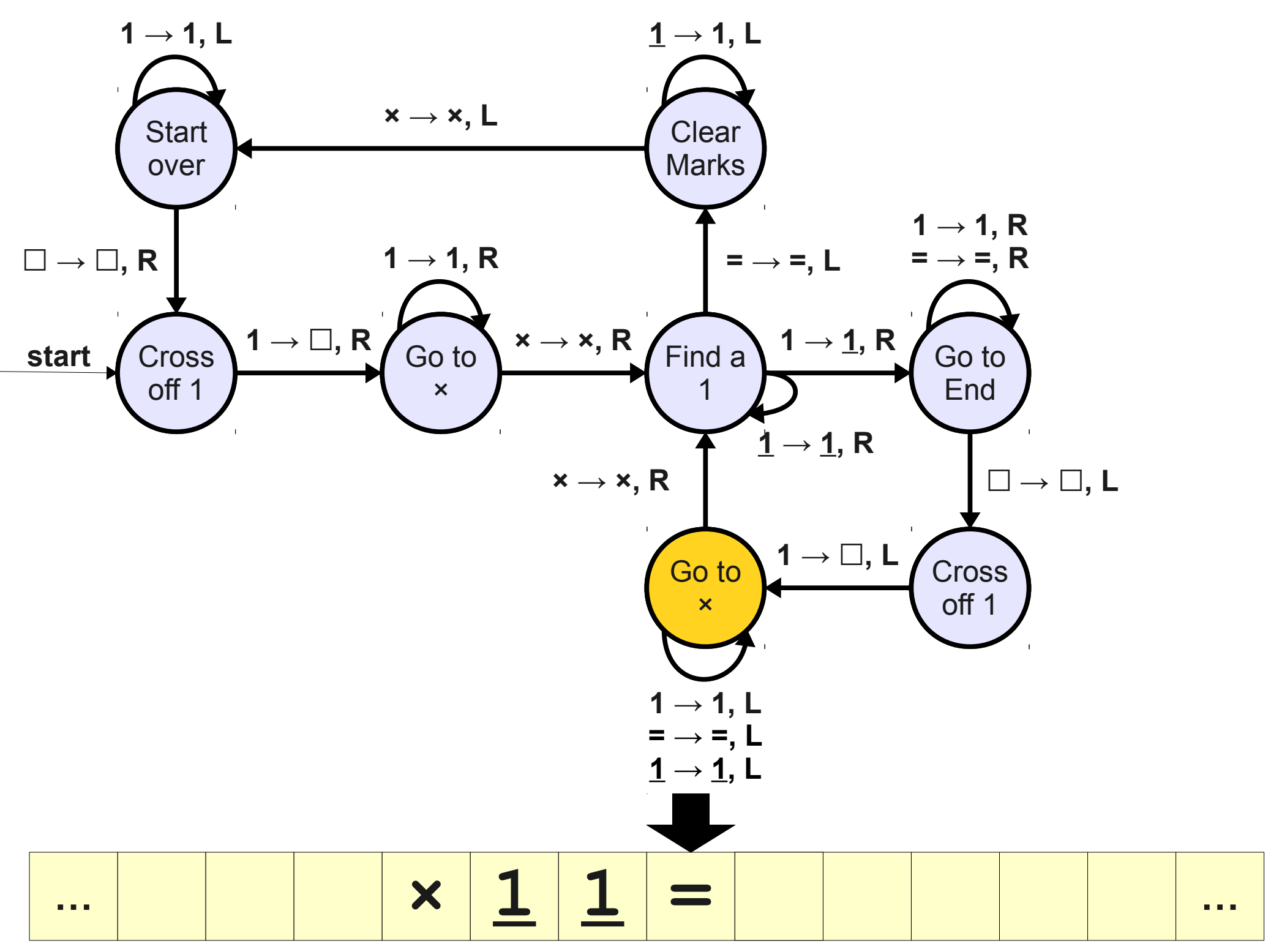

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

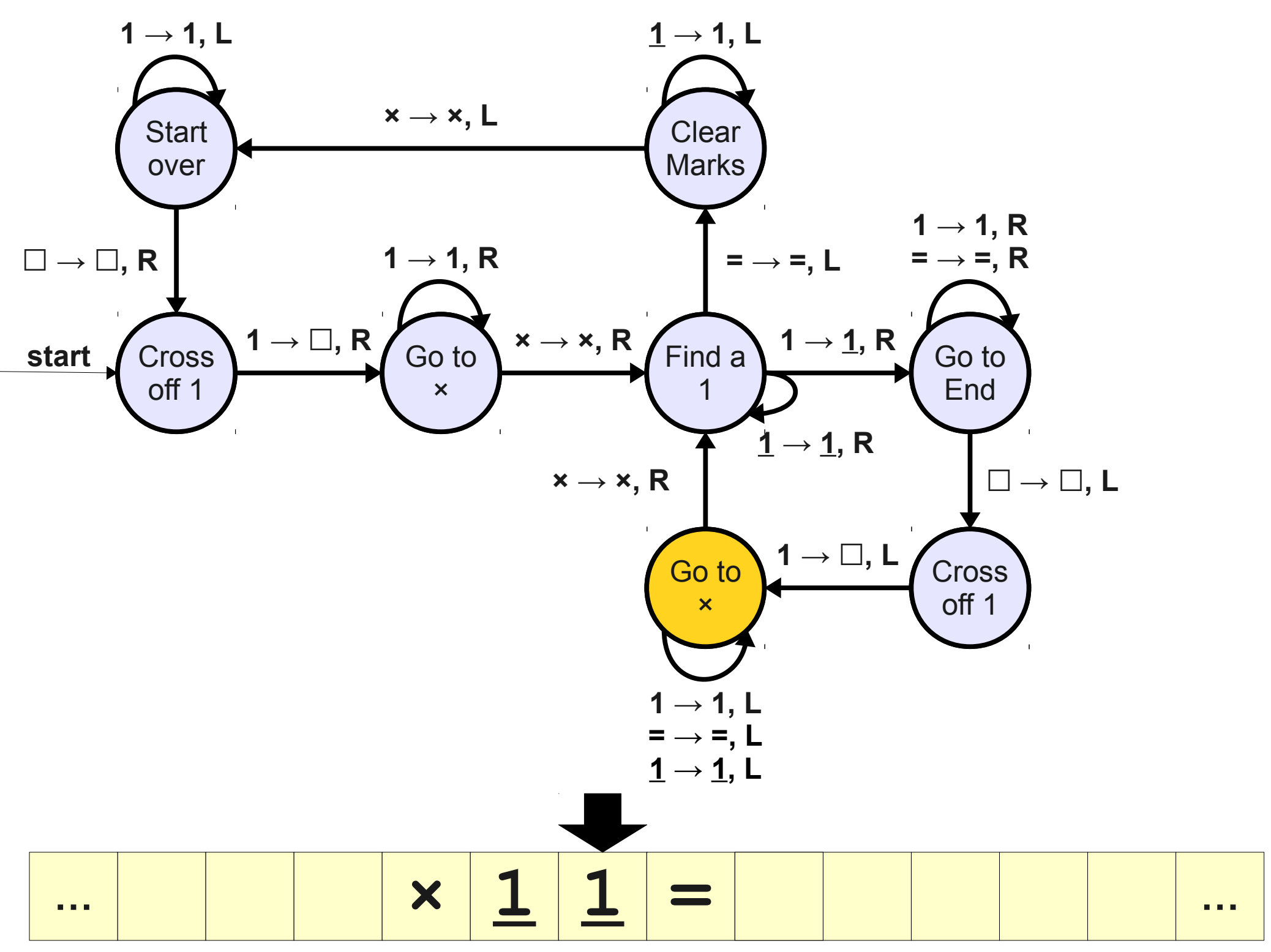

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

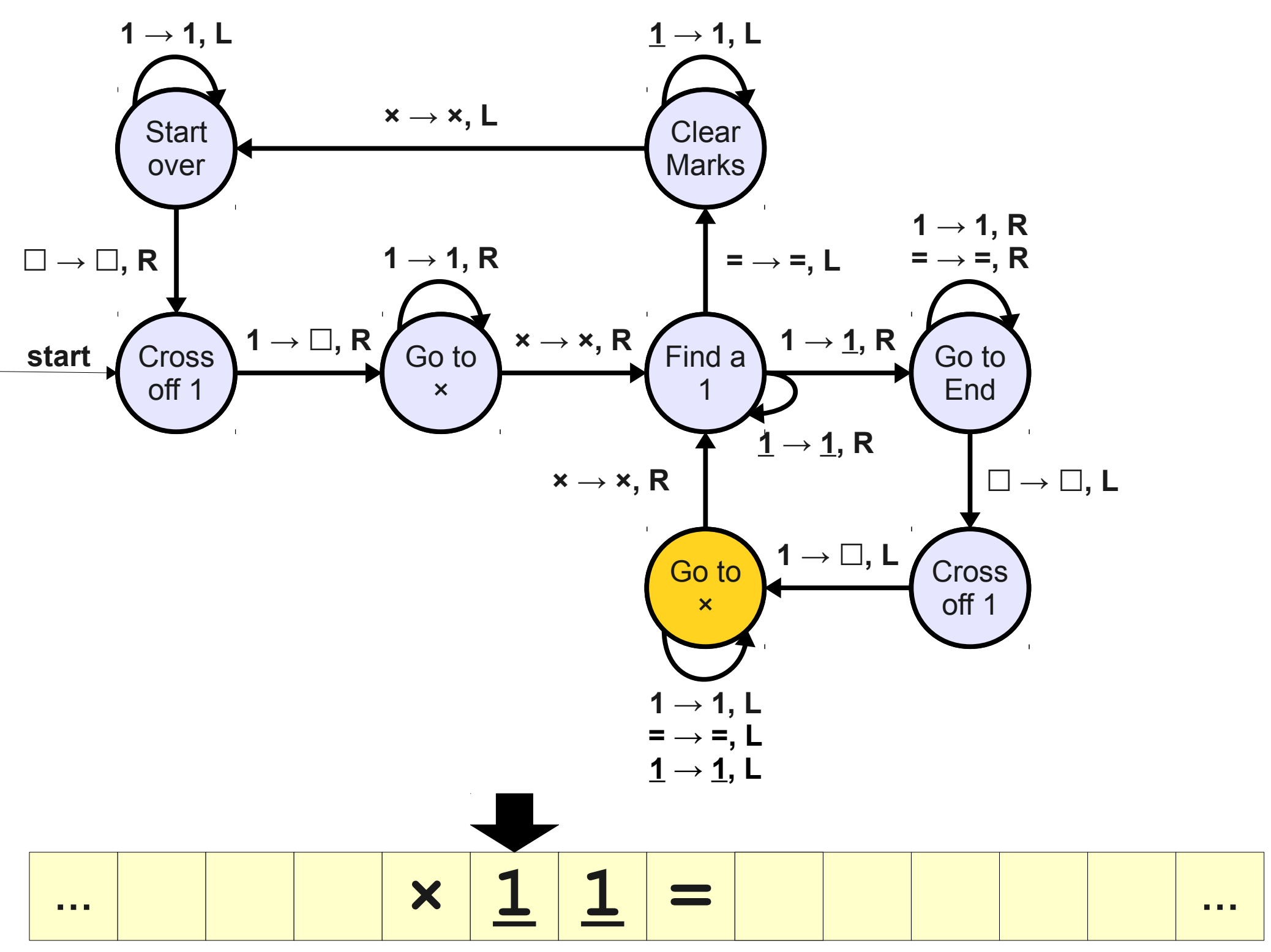

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

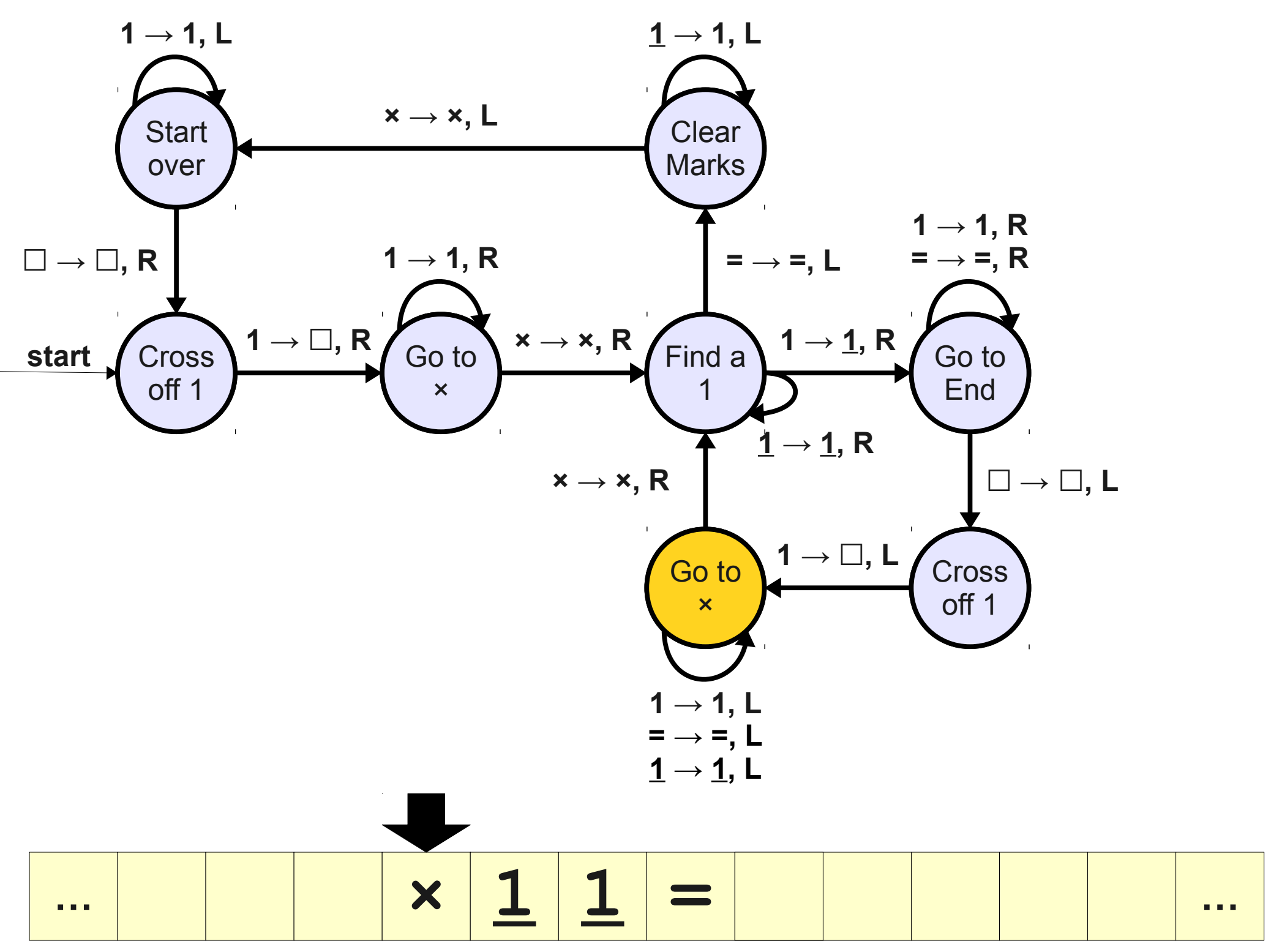

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

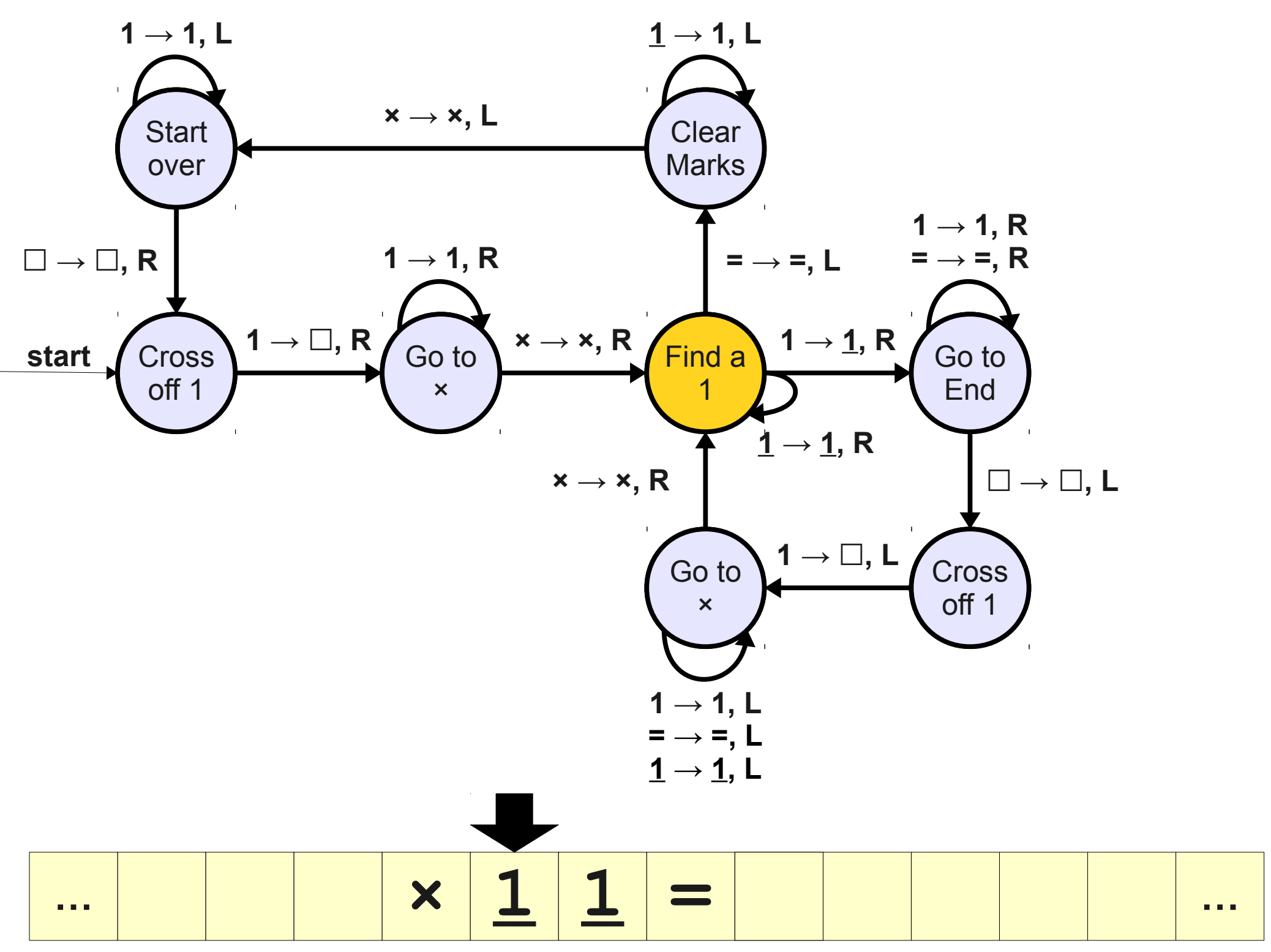

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

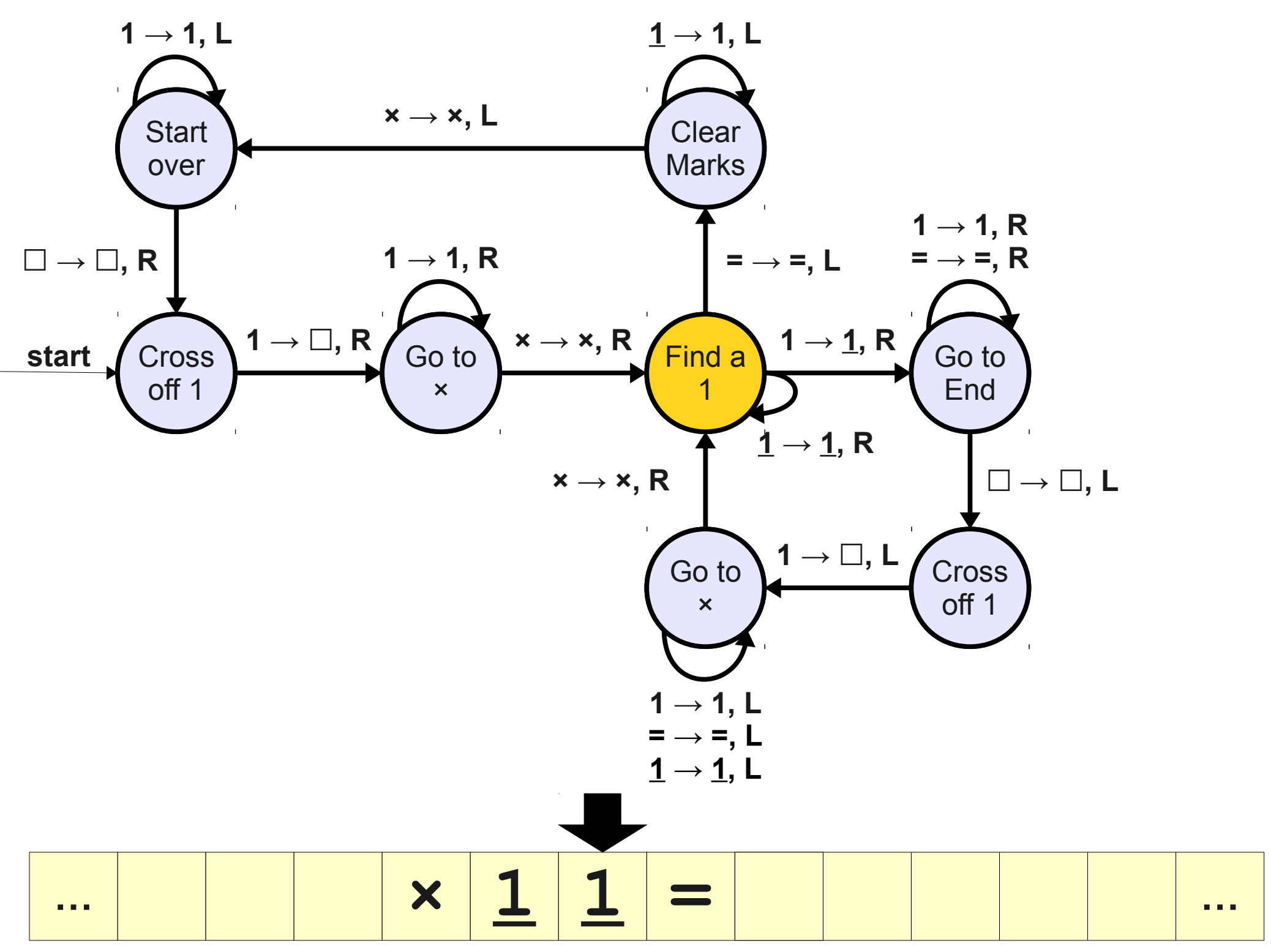

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

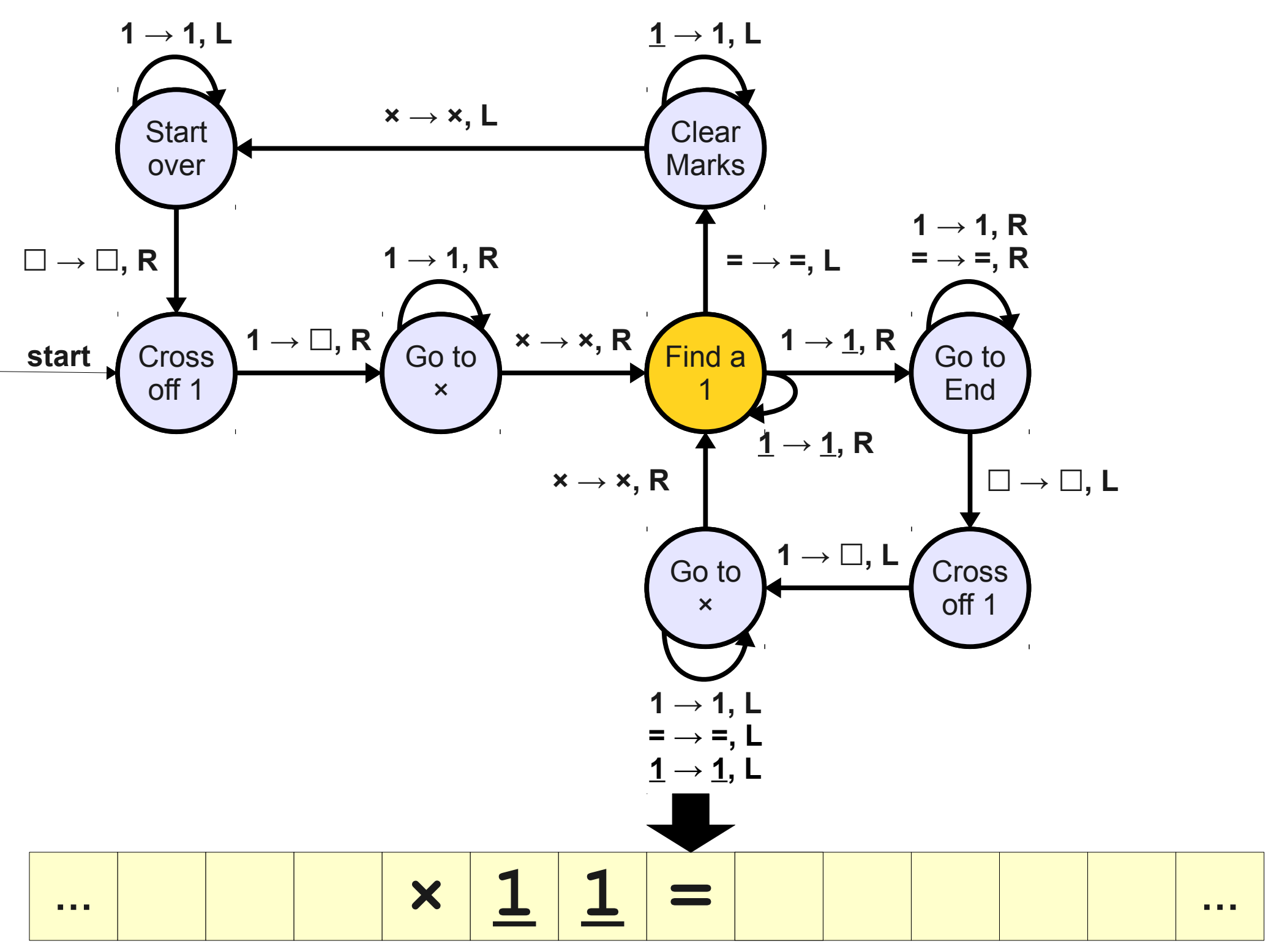

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

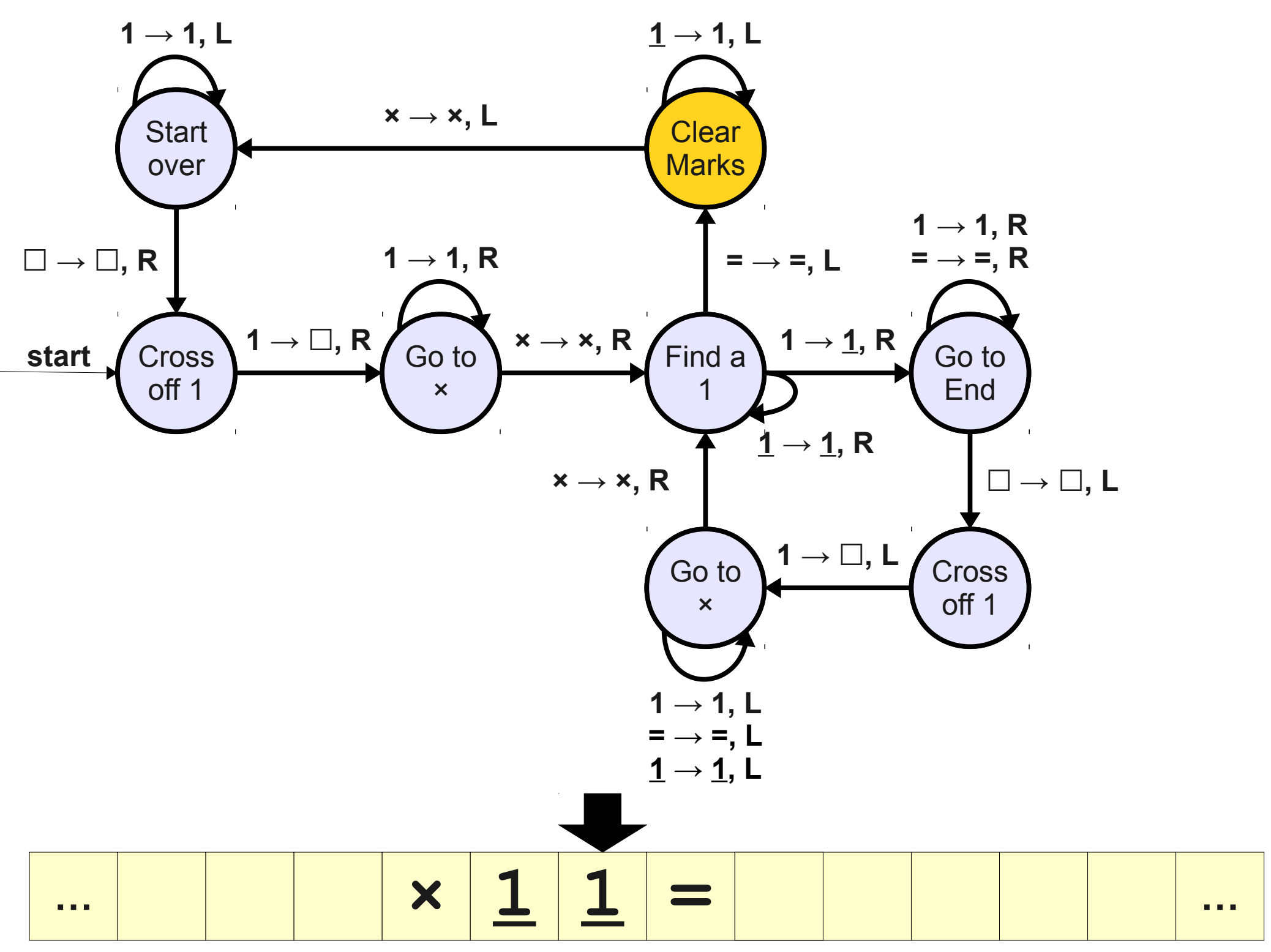

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

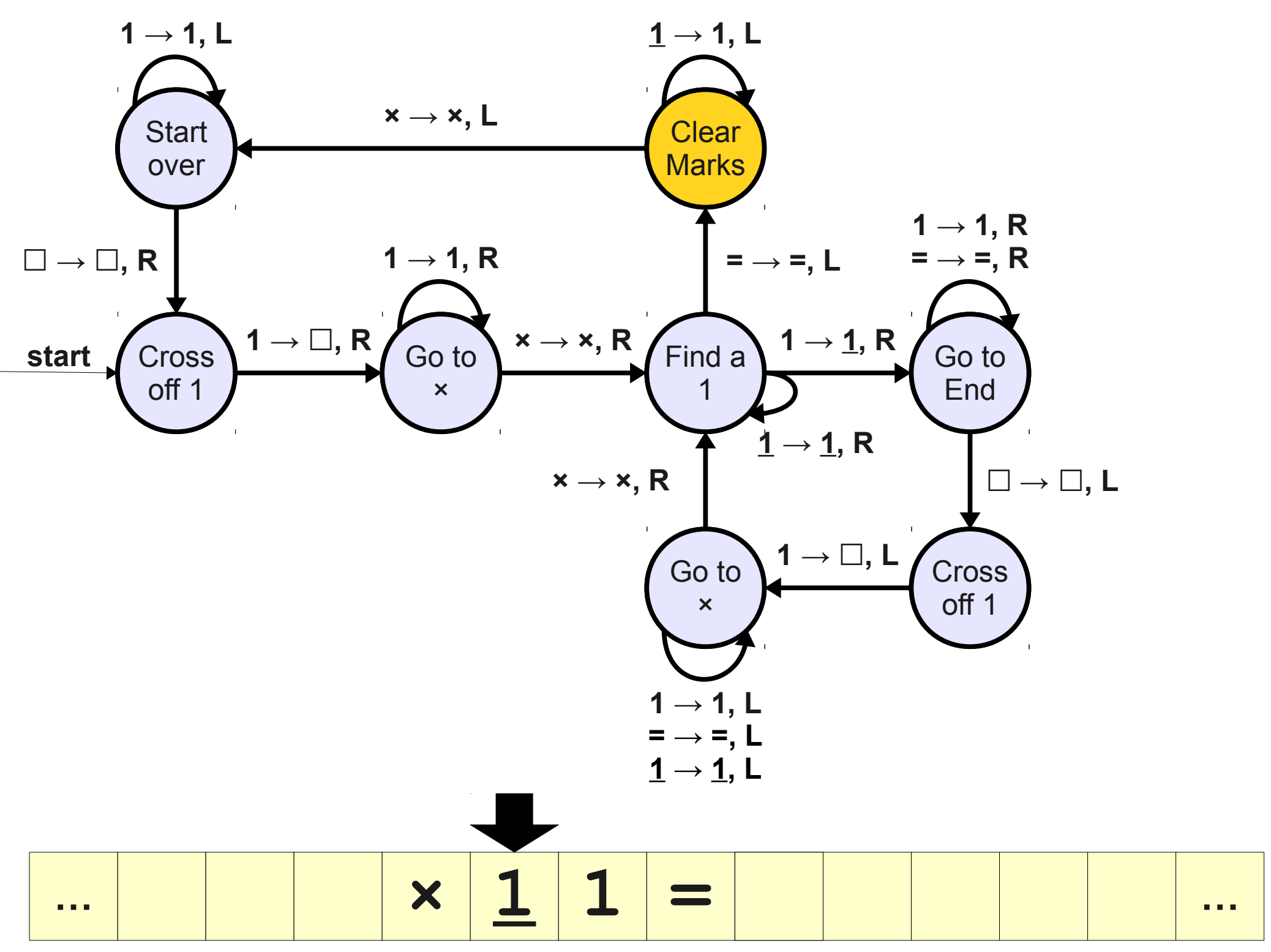

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

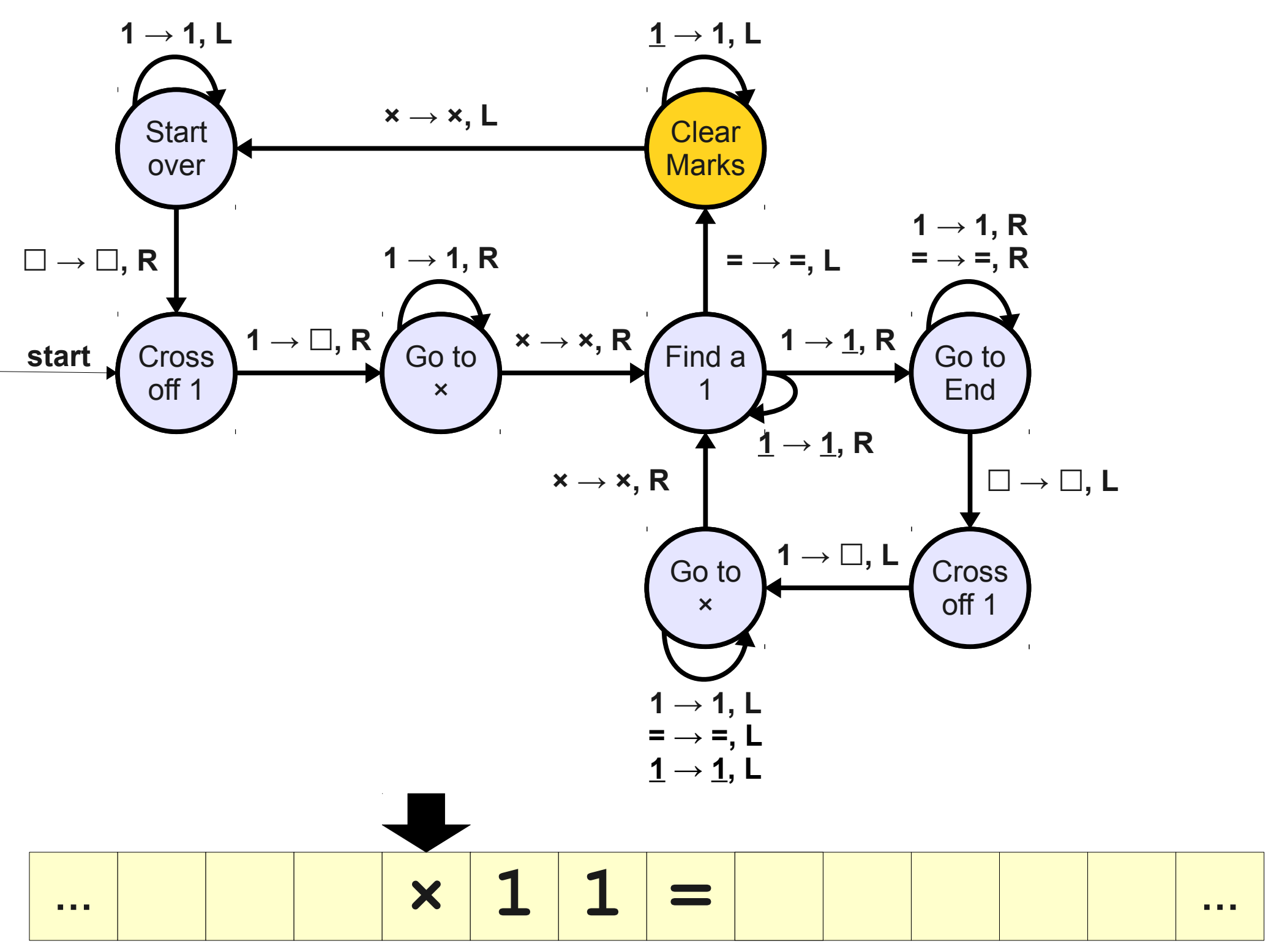

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

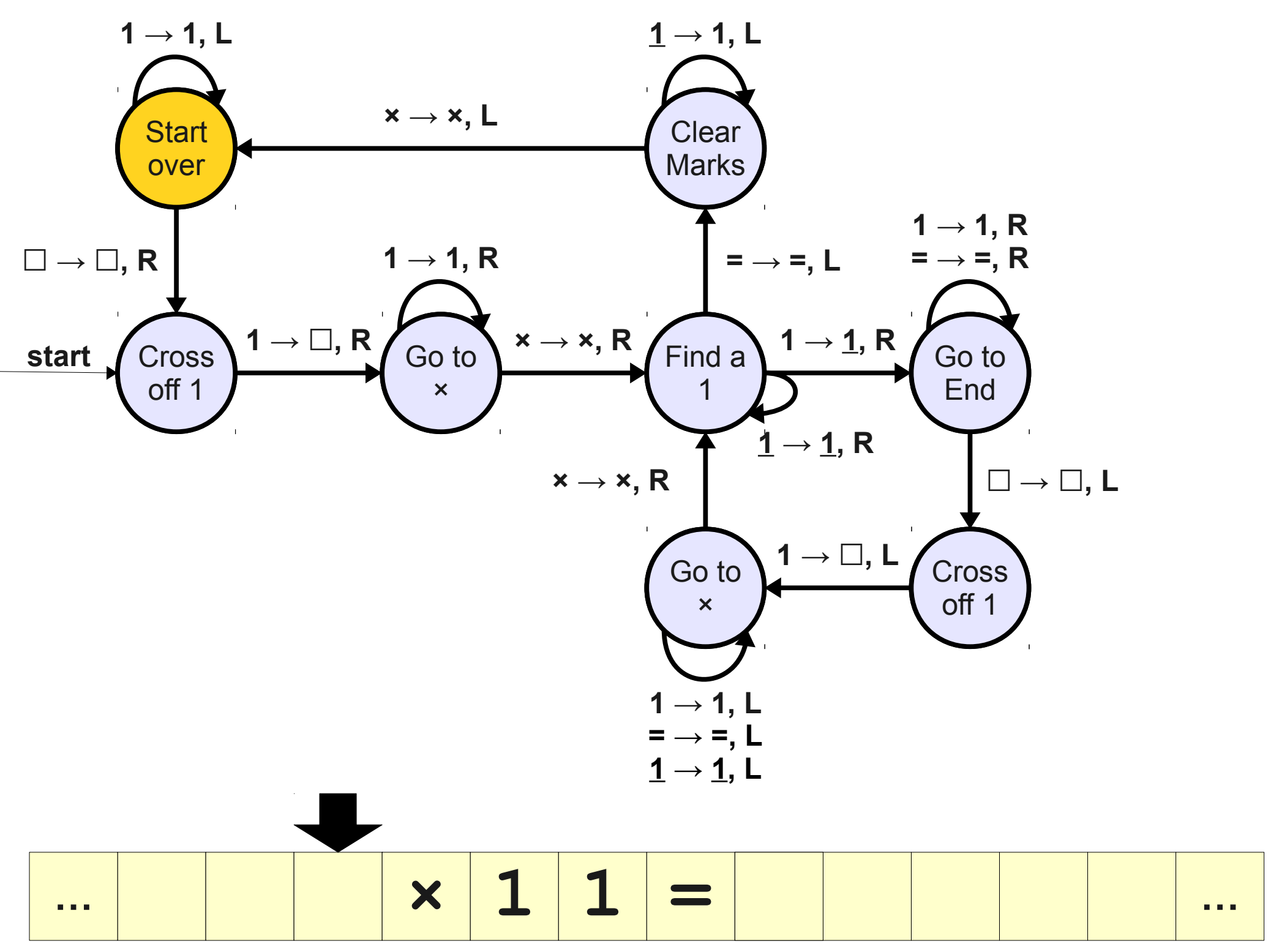

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

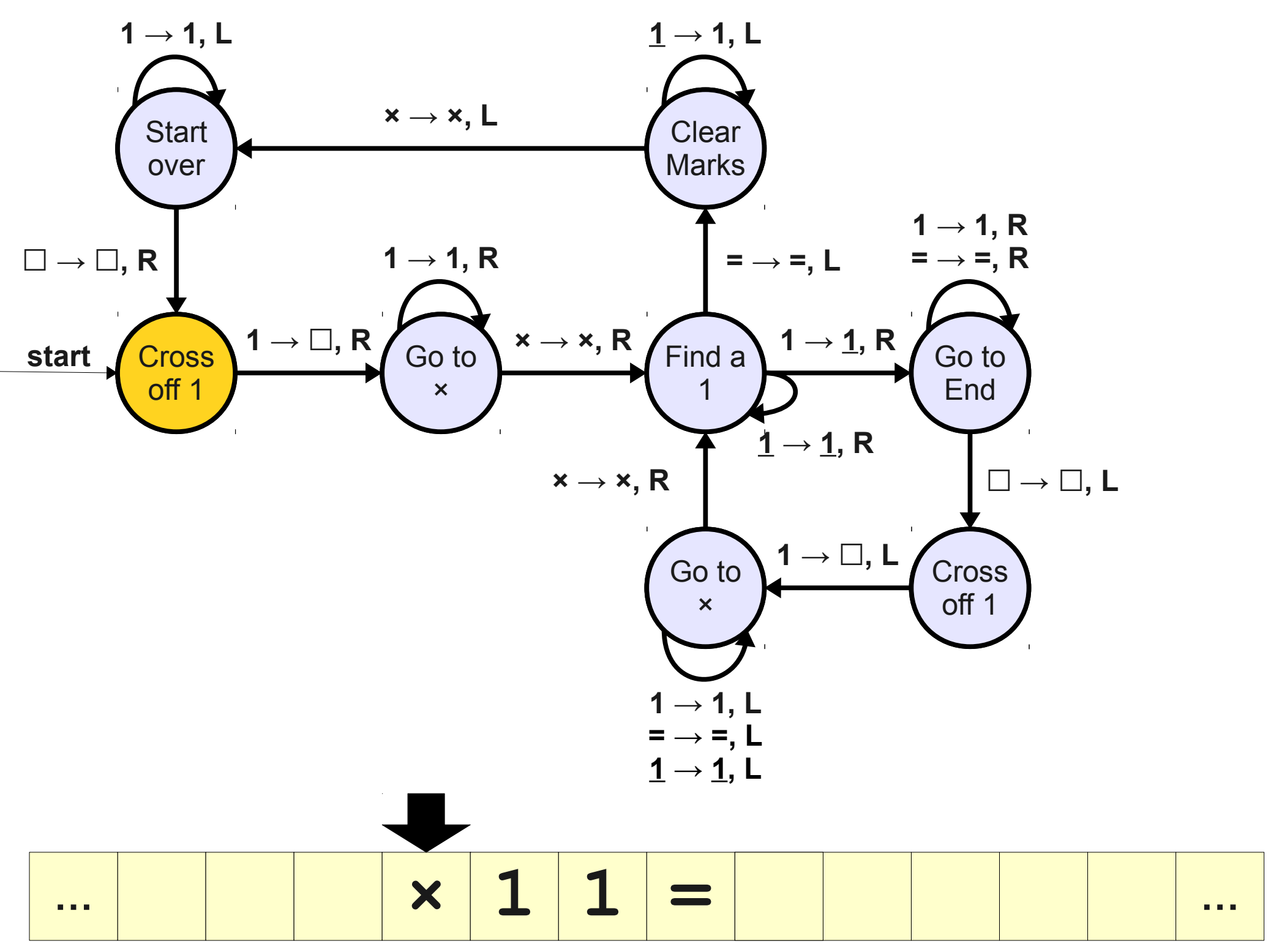

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

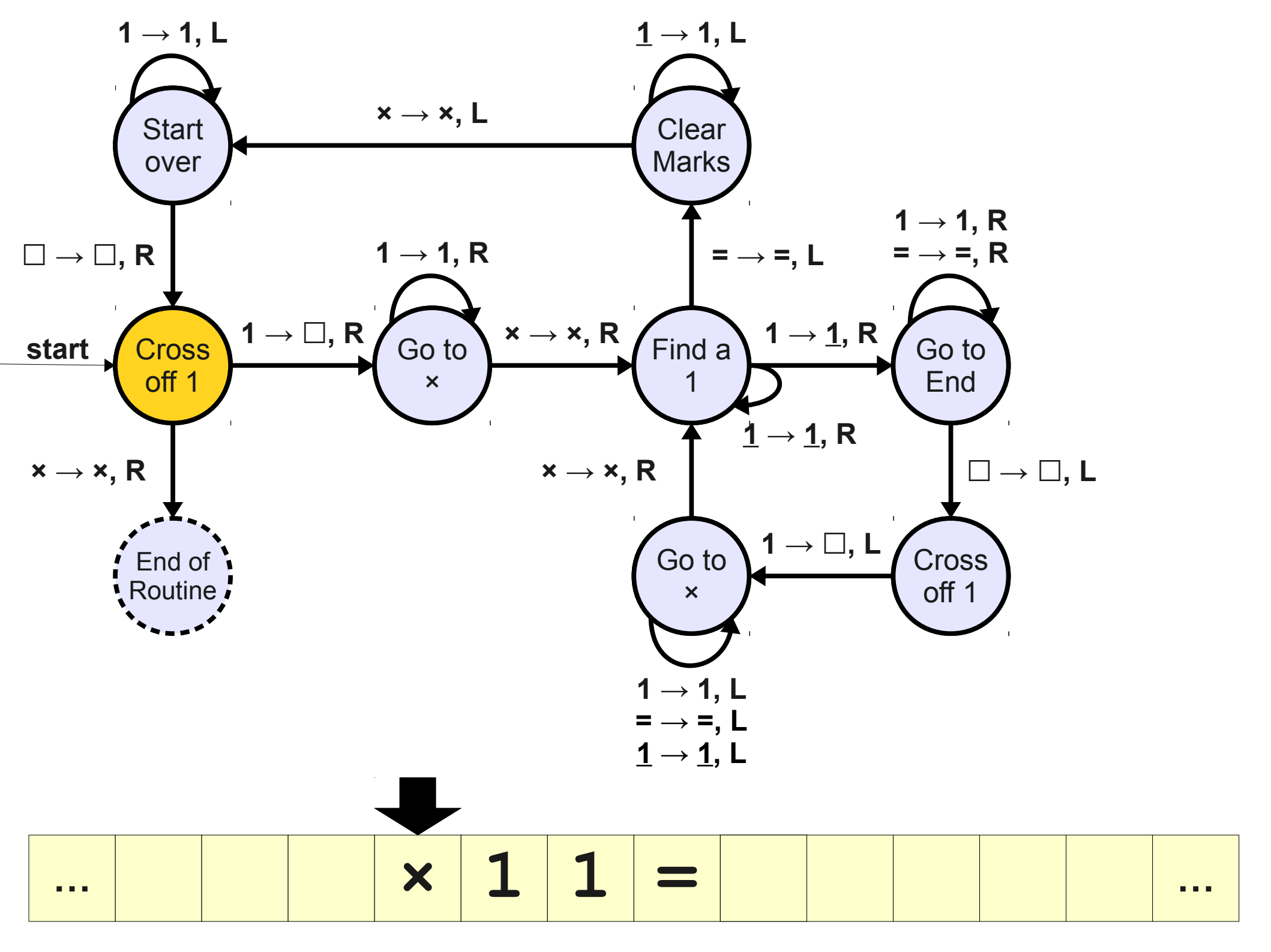

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

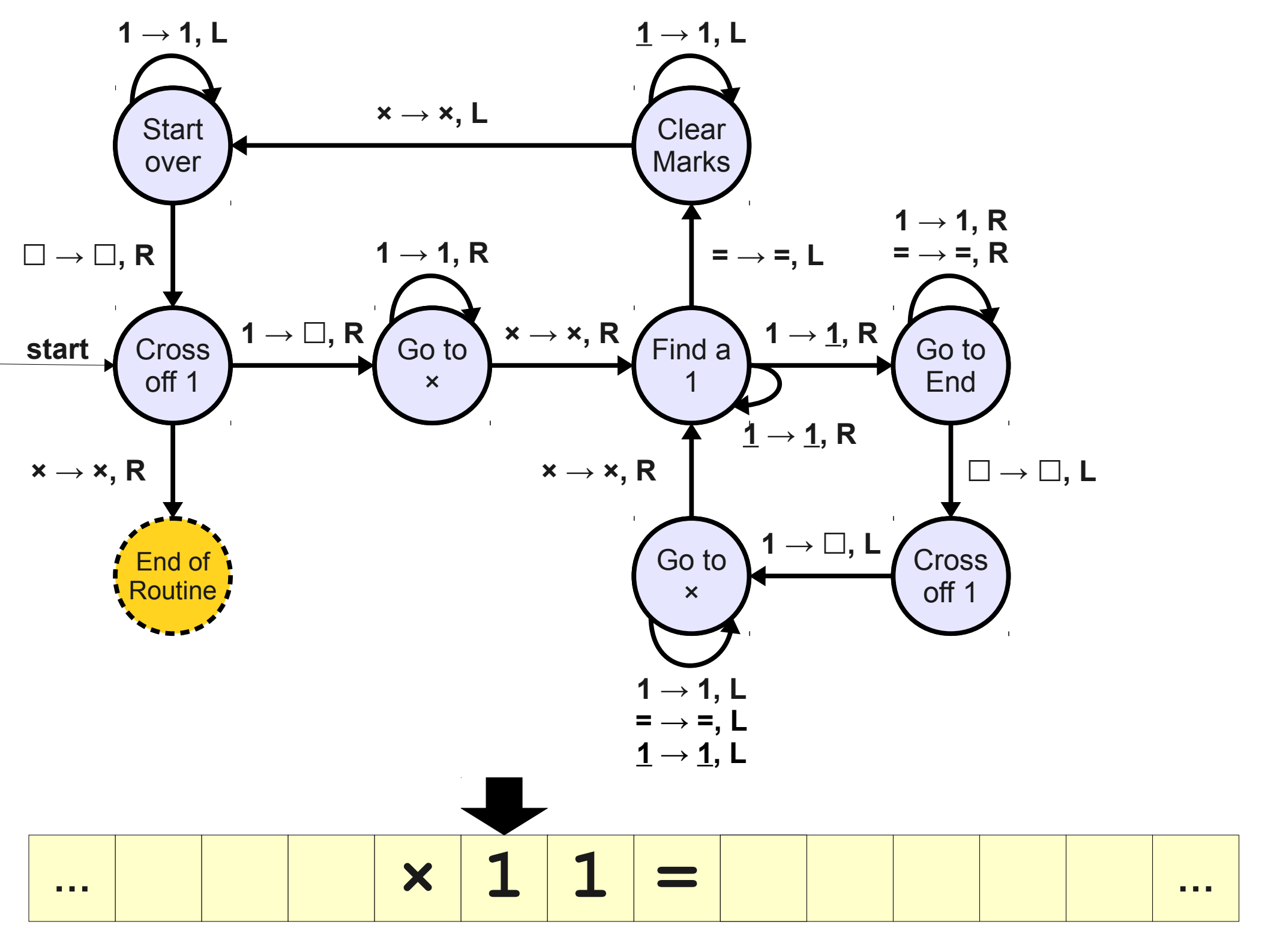

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

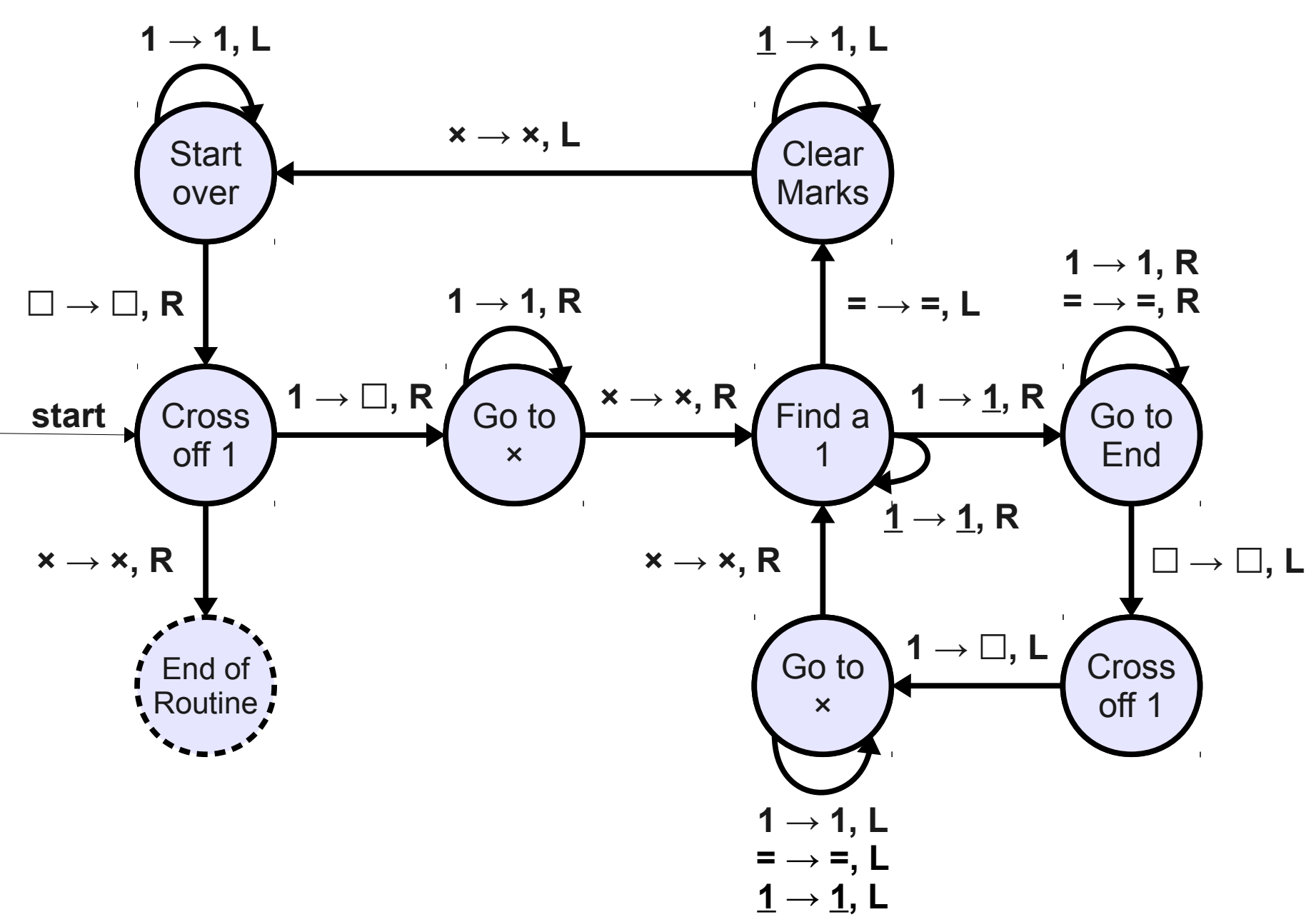

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

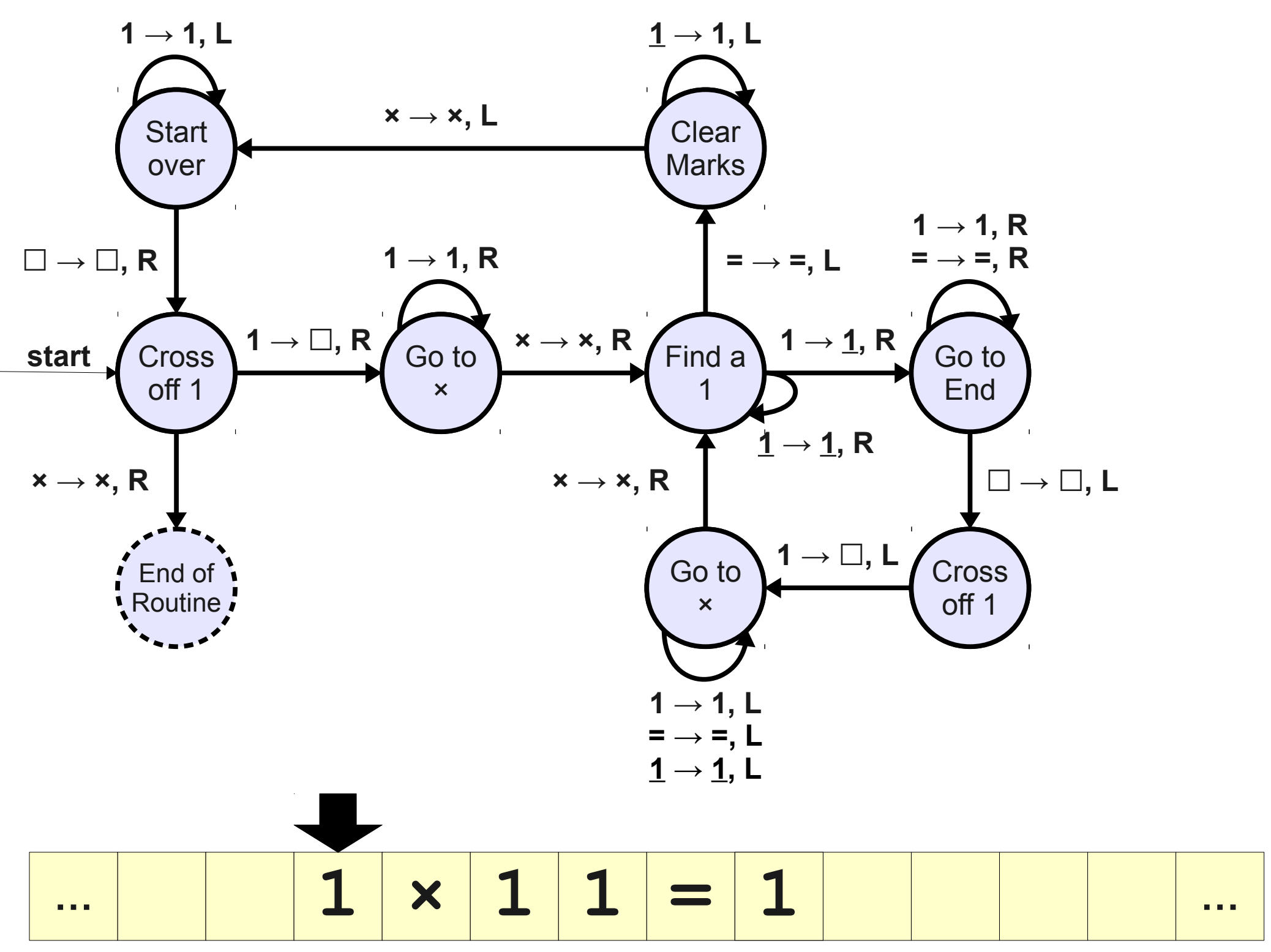

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

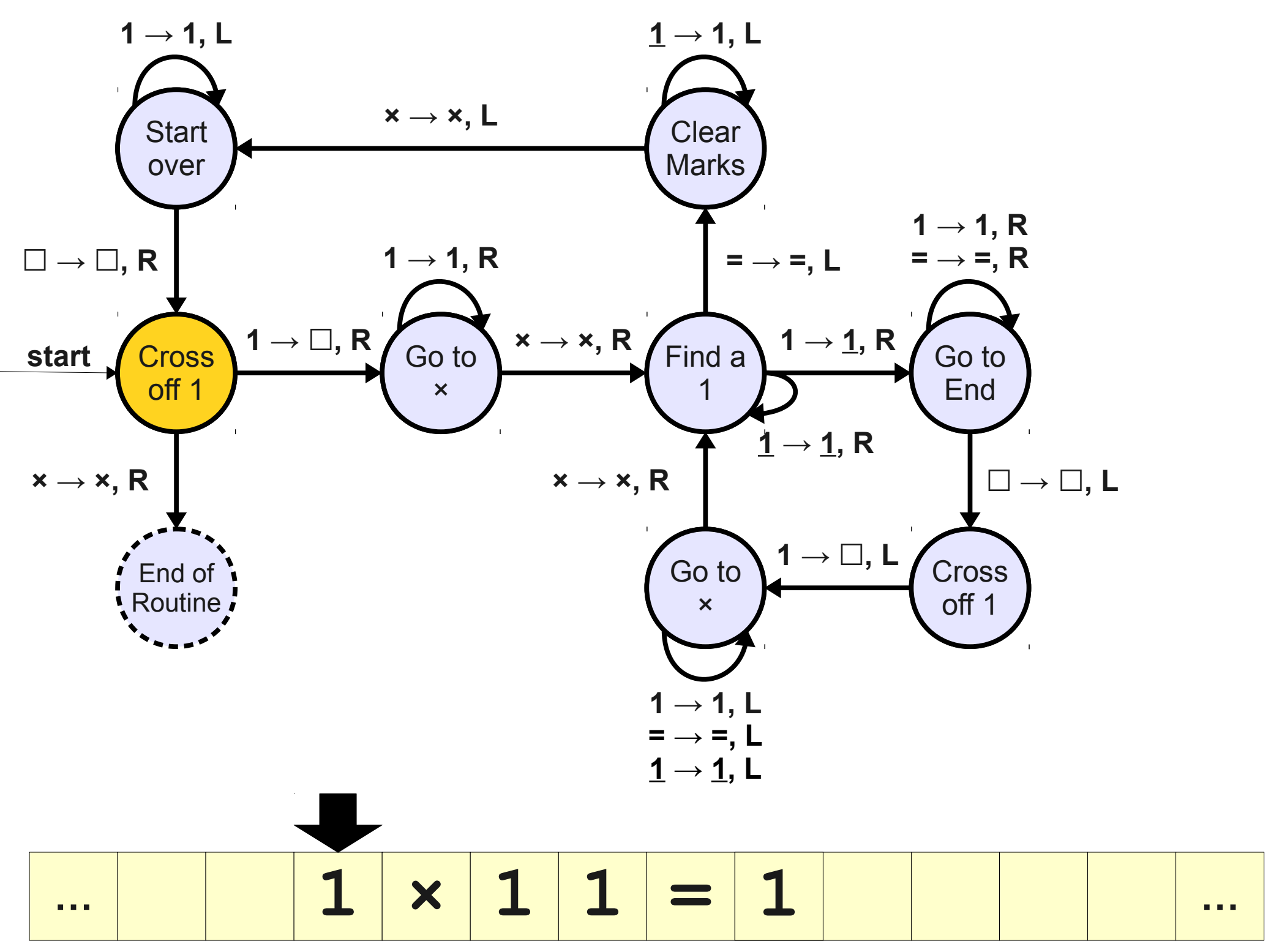

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
□ → □, L
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

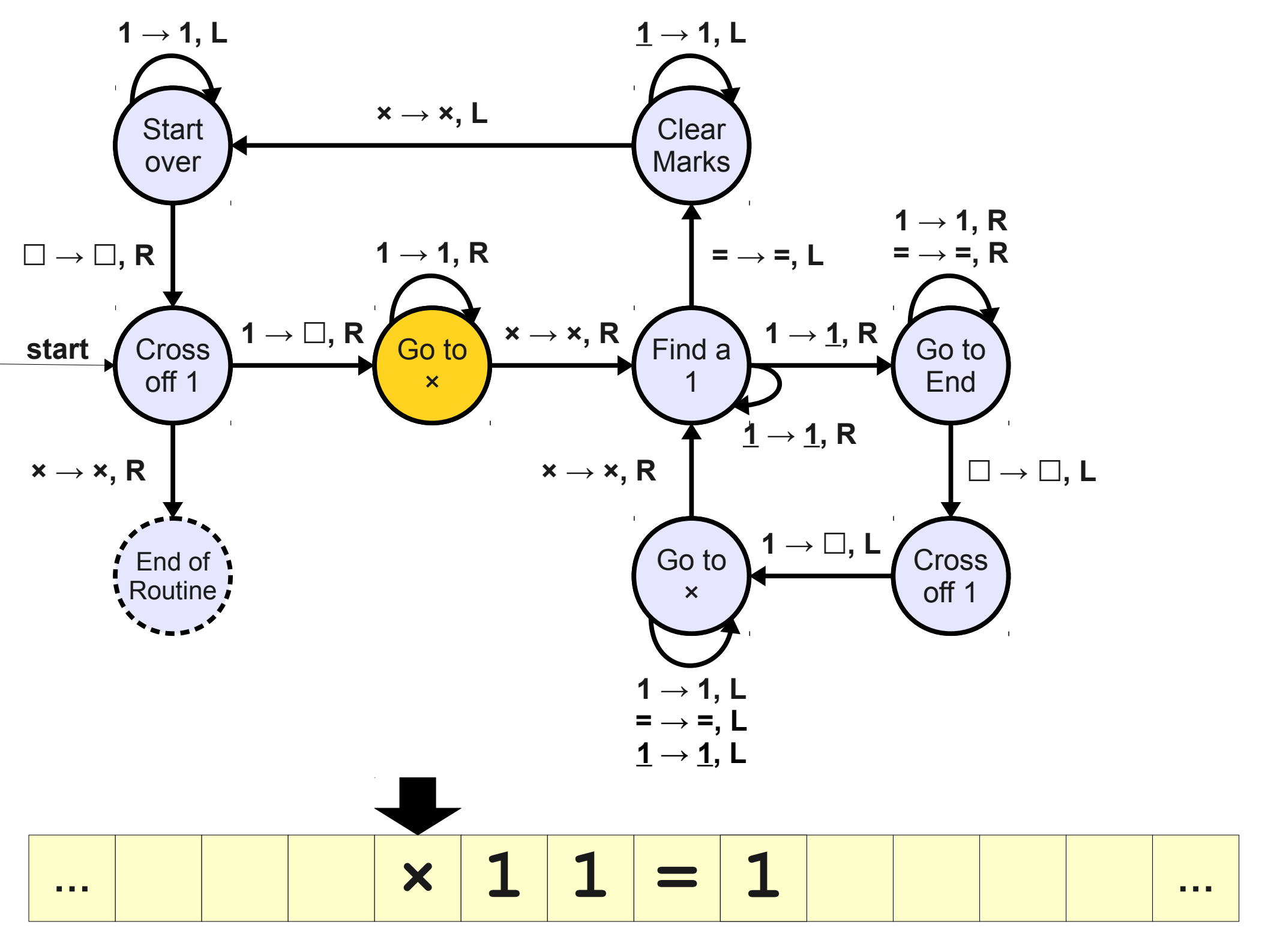

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

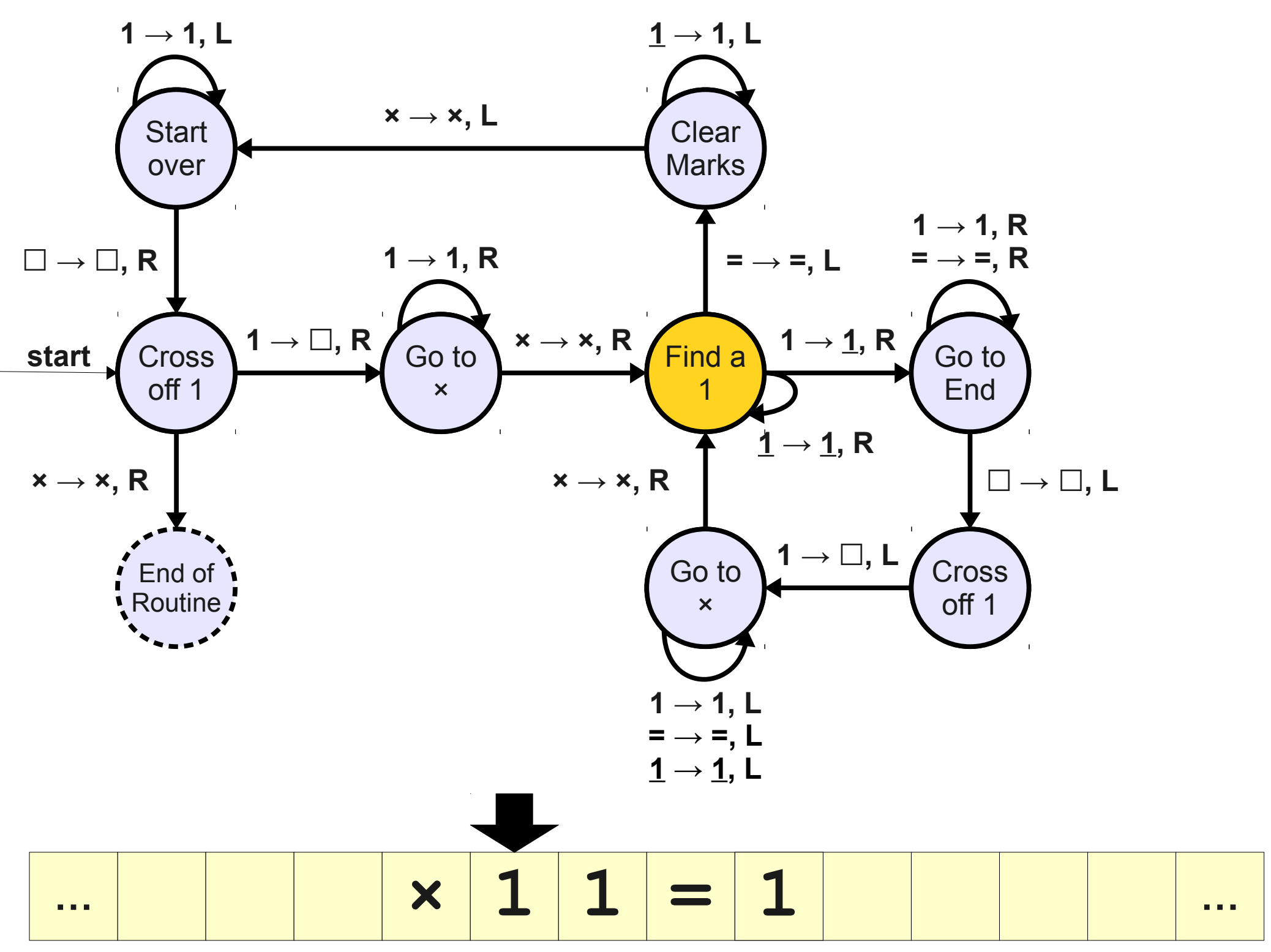

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

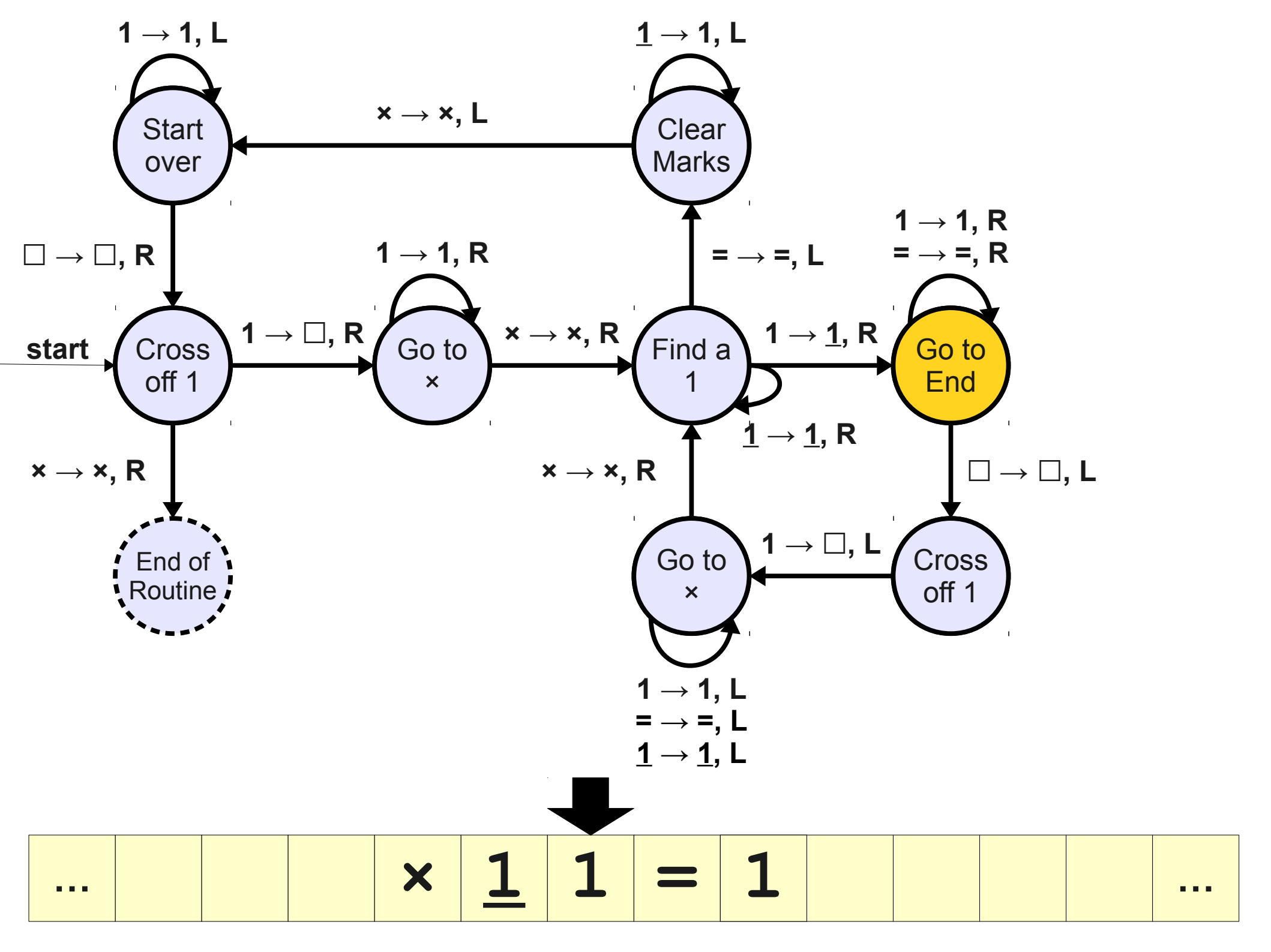

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L

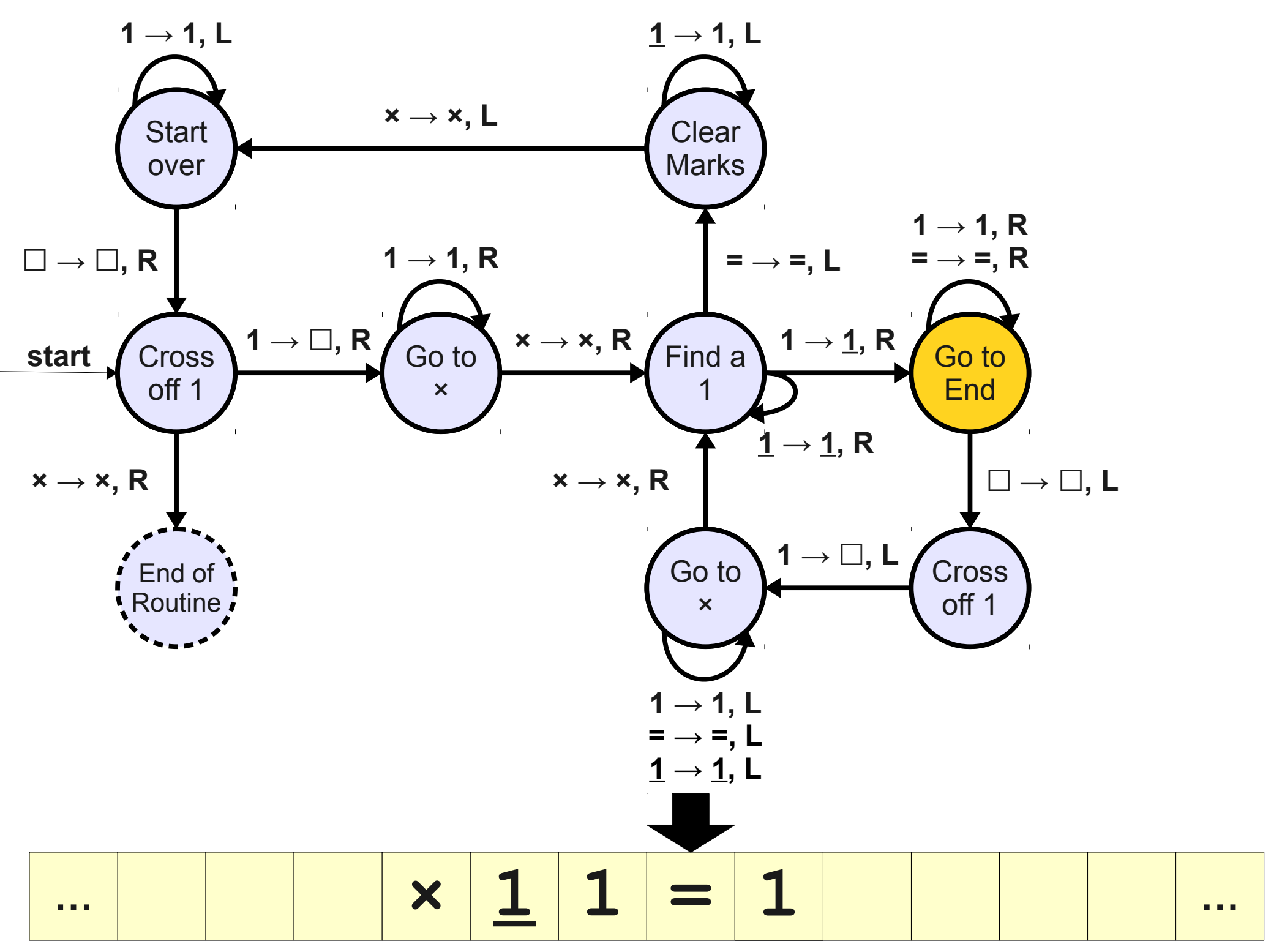

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 ...

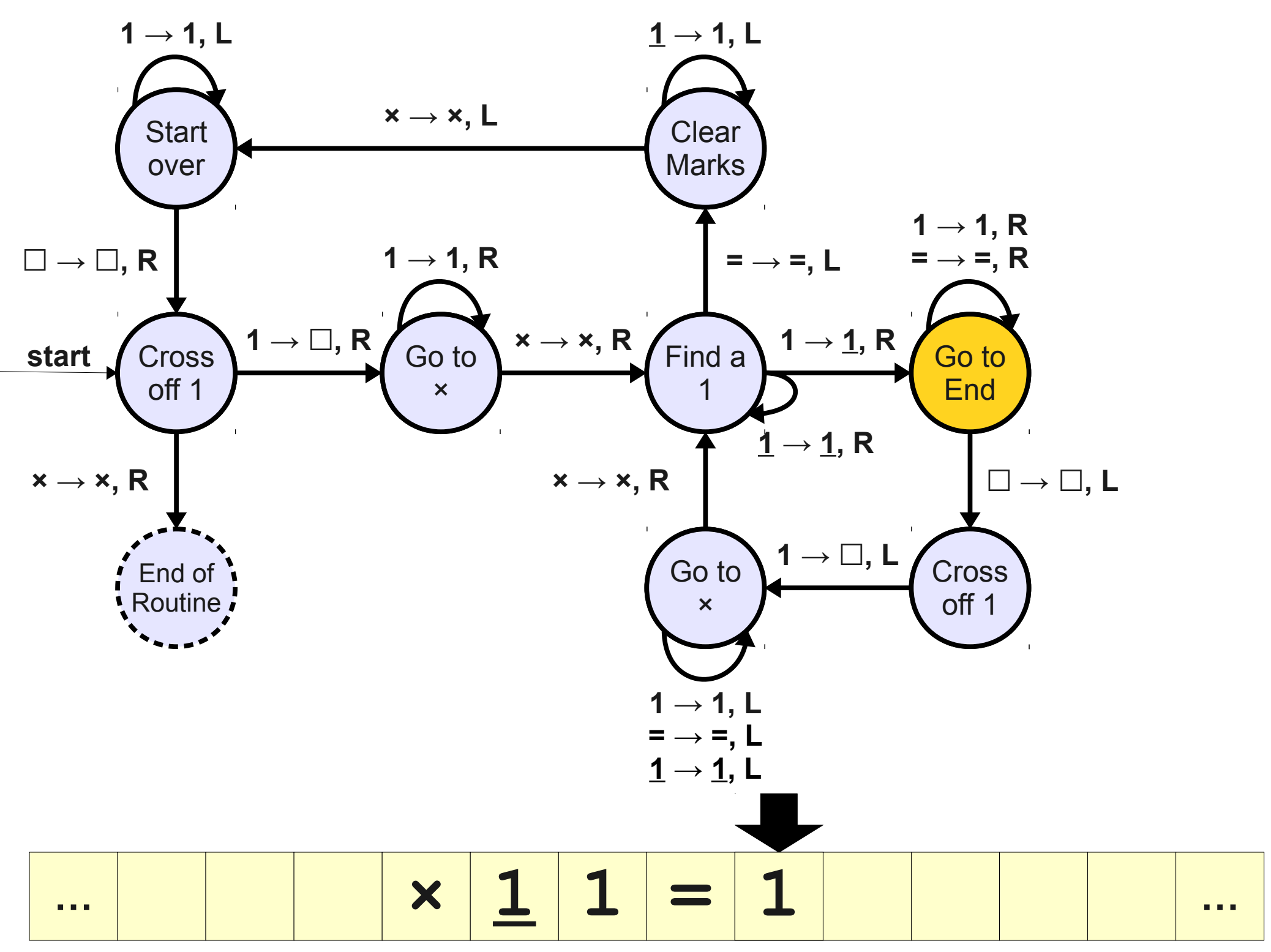

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... 1 × 1 1 = 1 ...

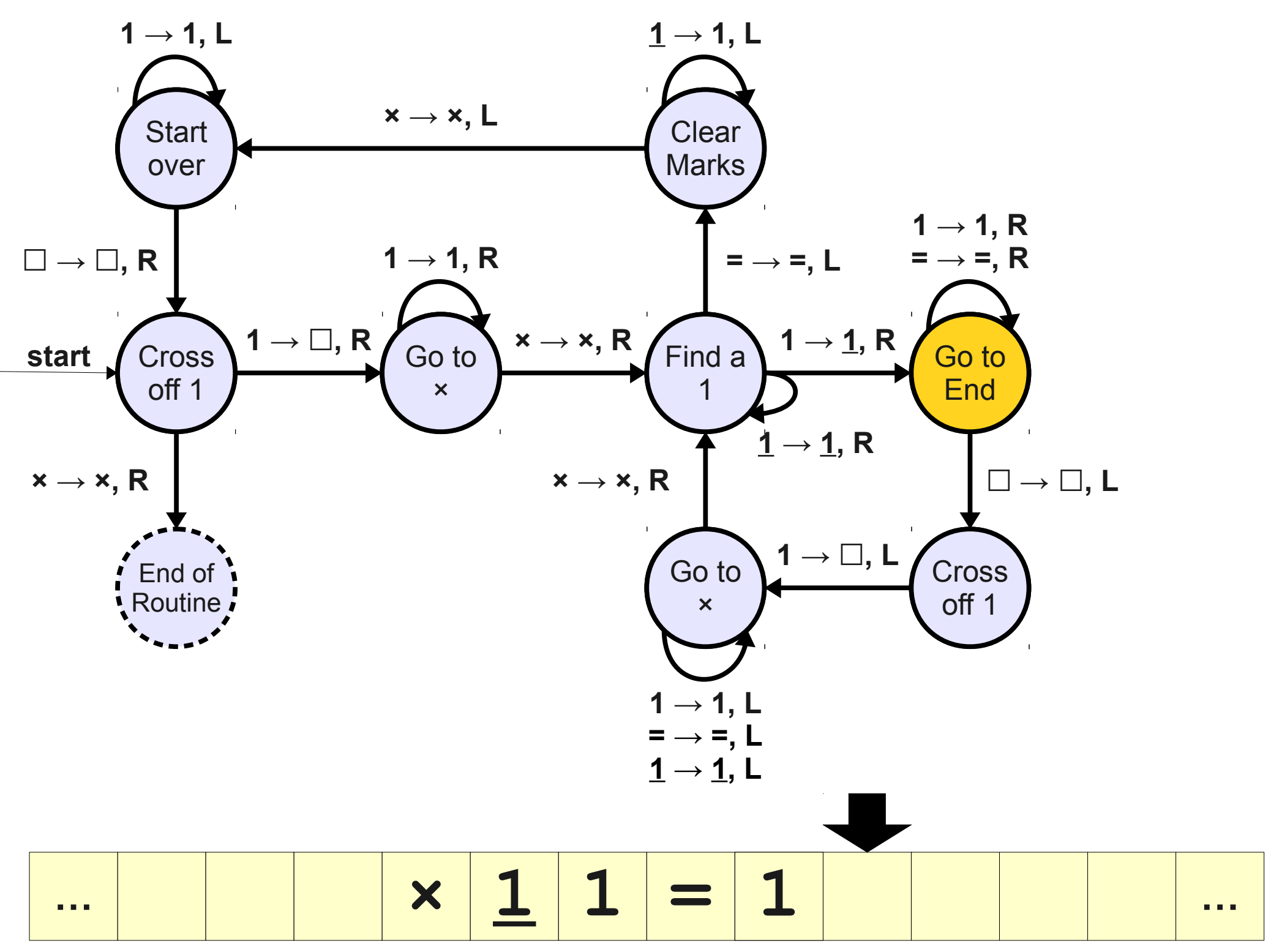

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

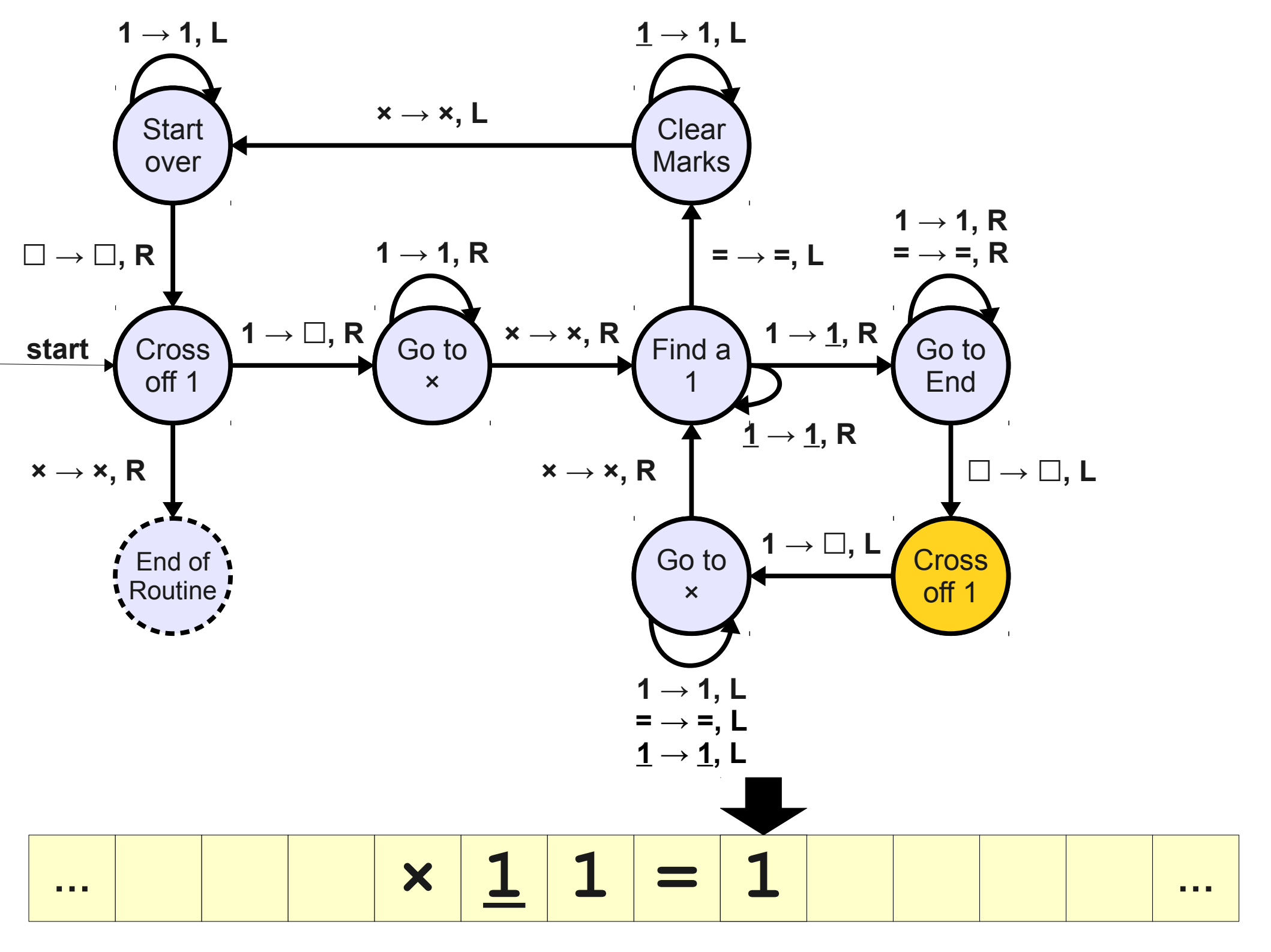

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

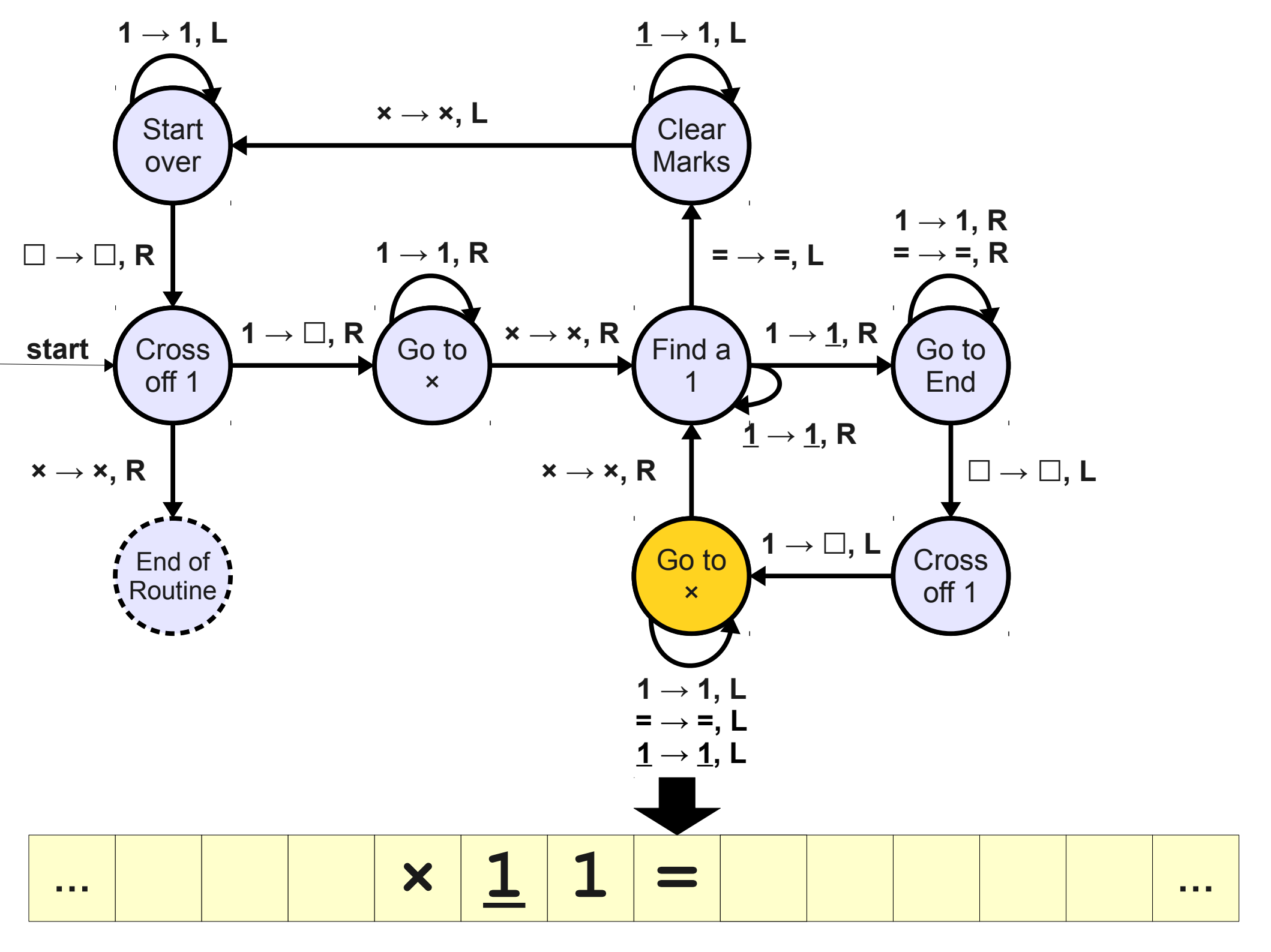

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

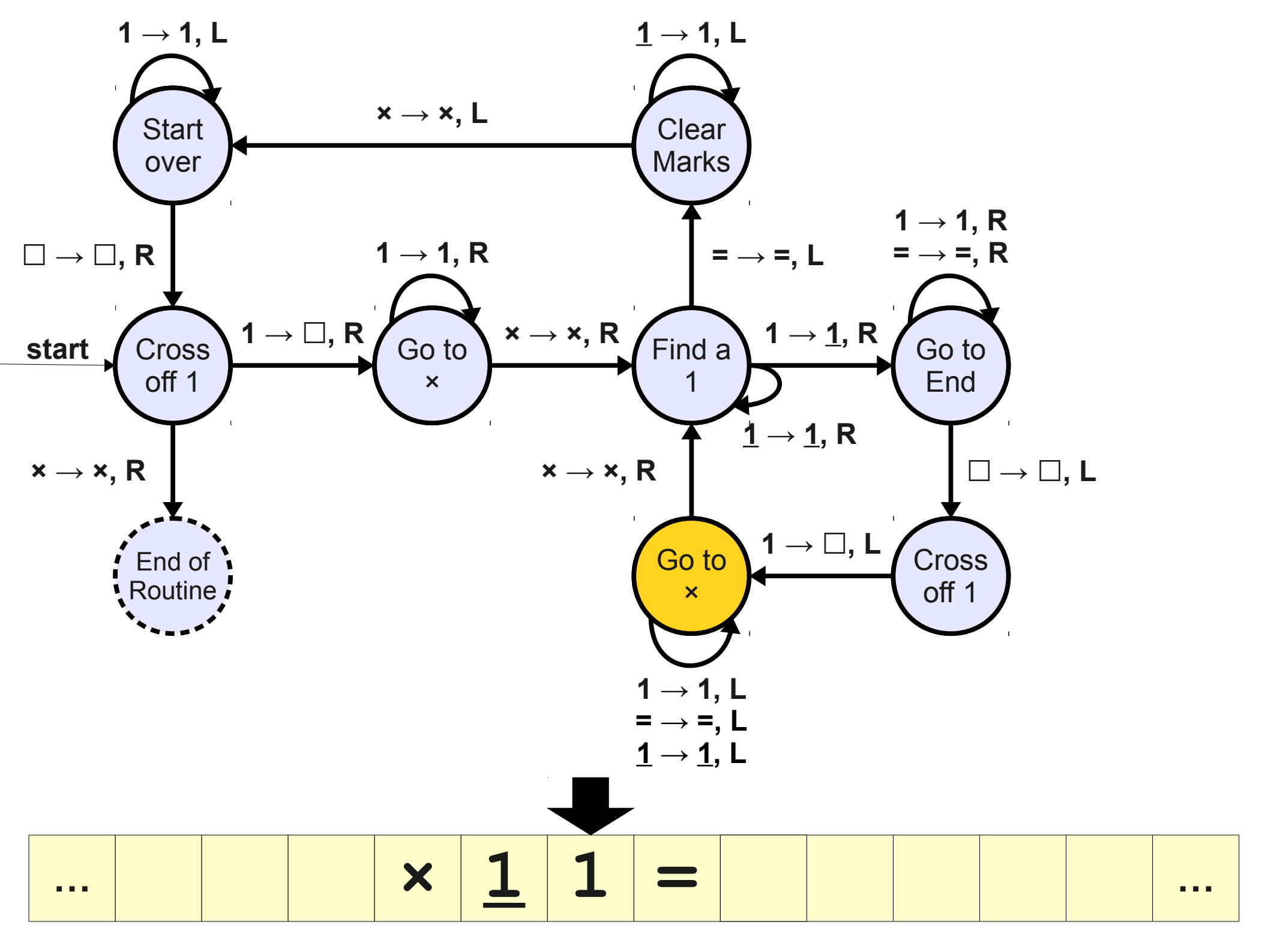

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

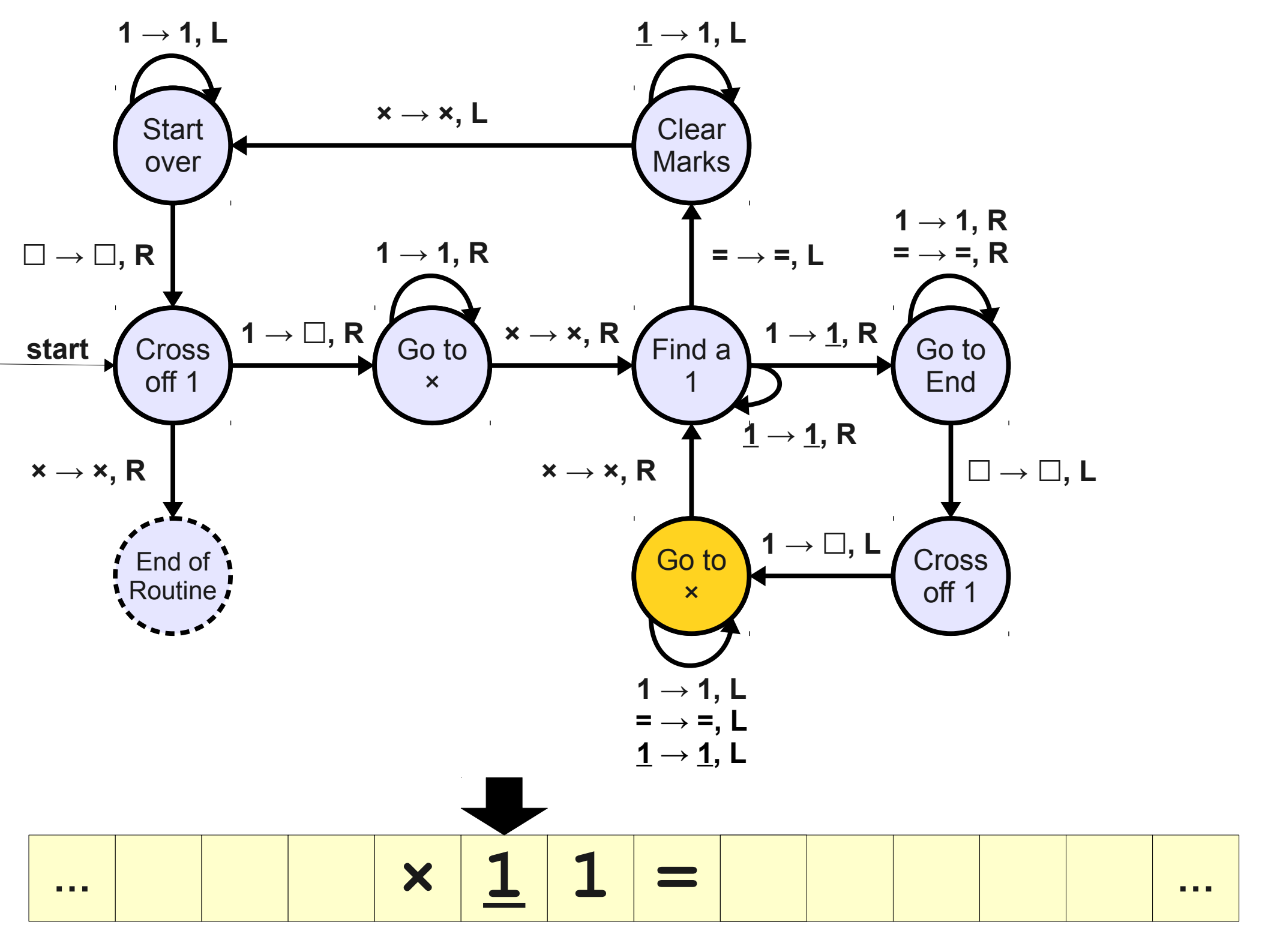

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

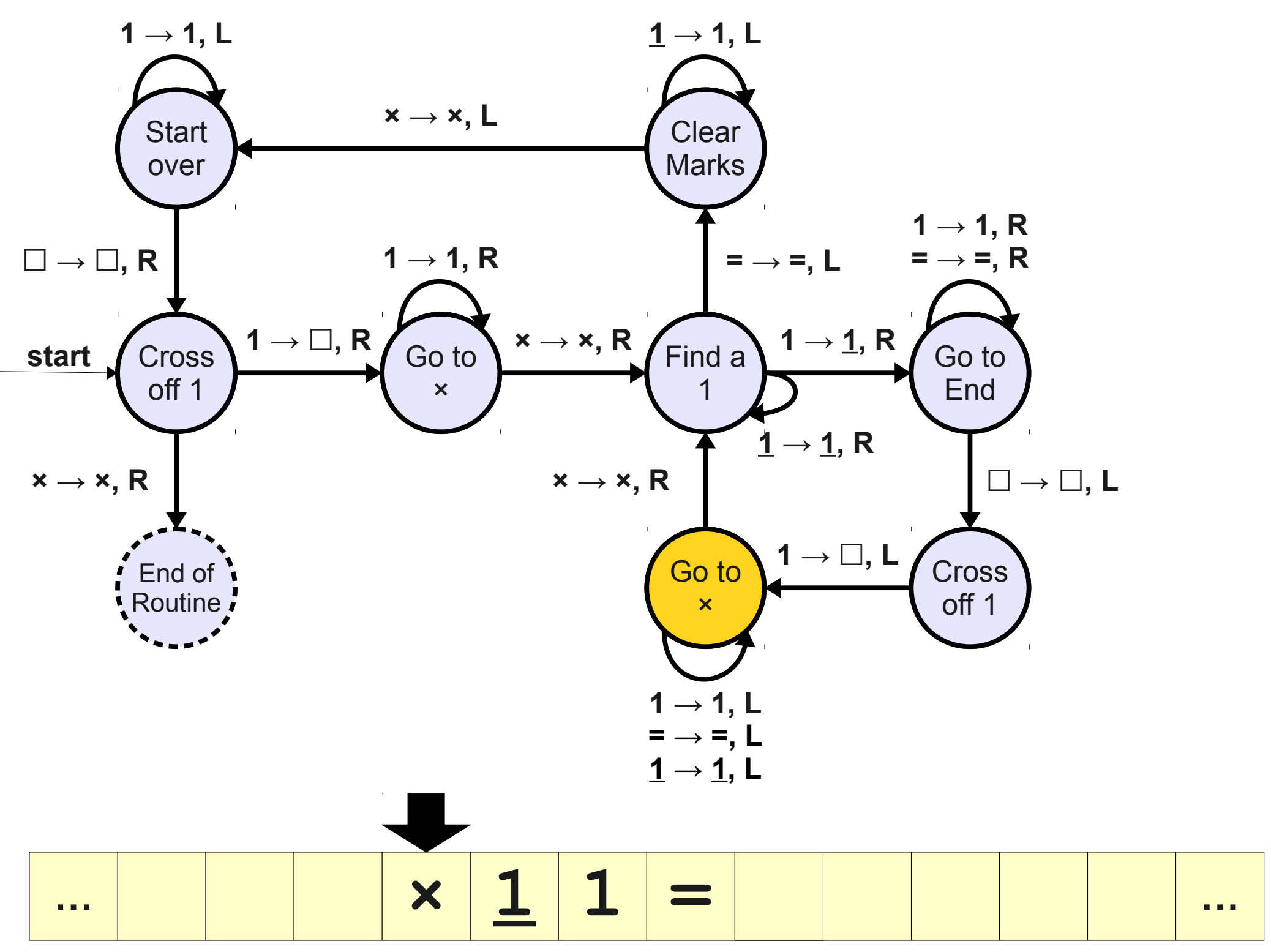

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

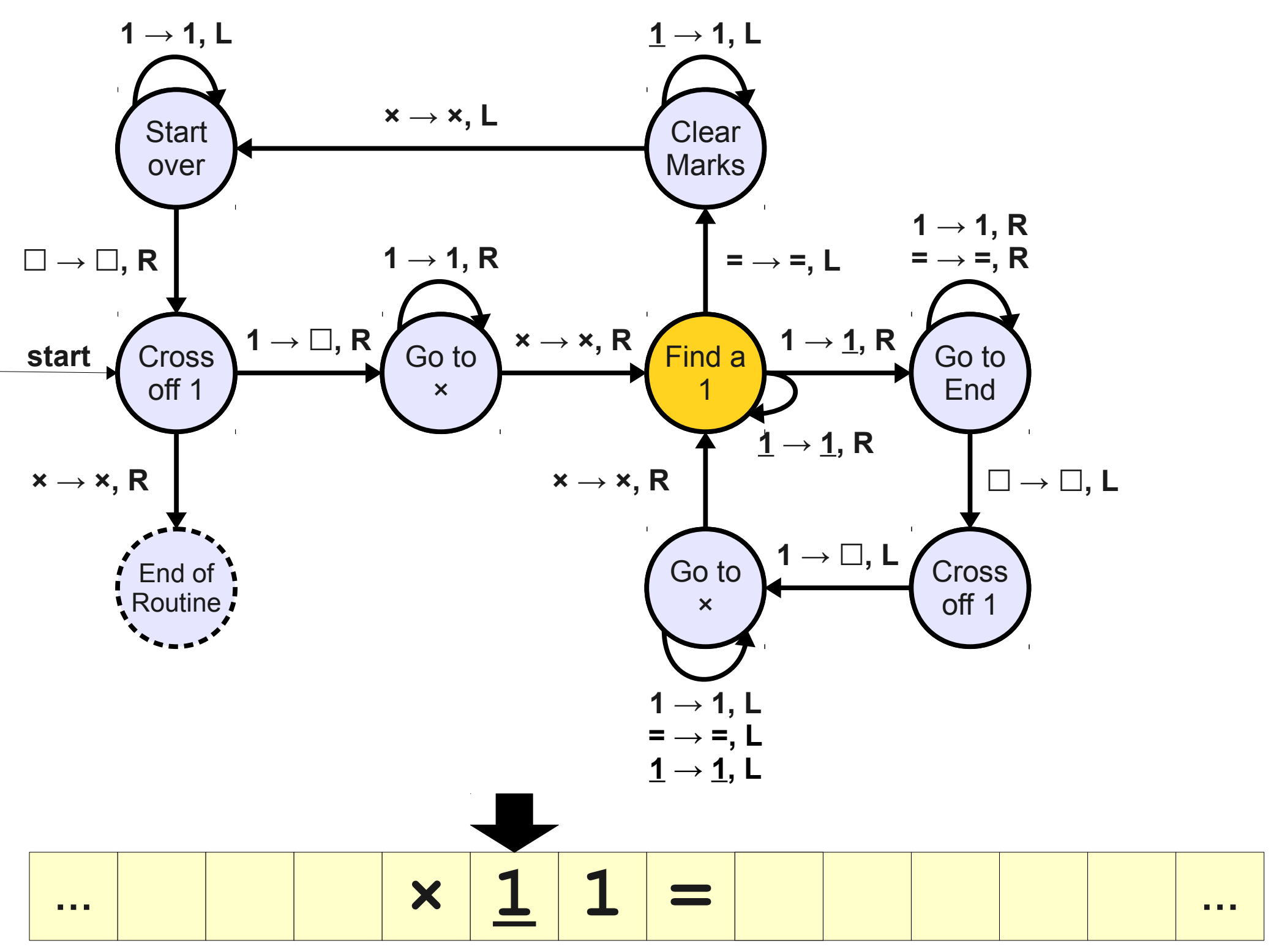

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = 1 ...

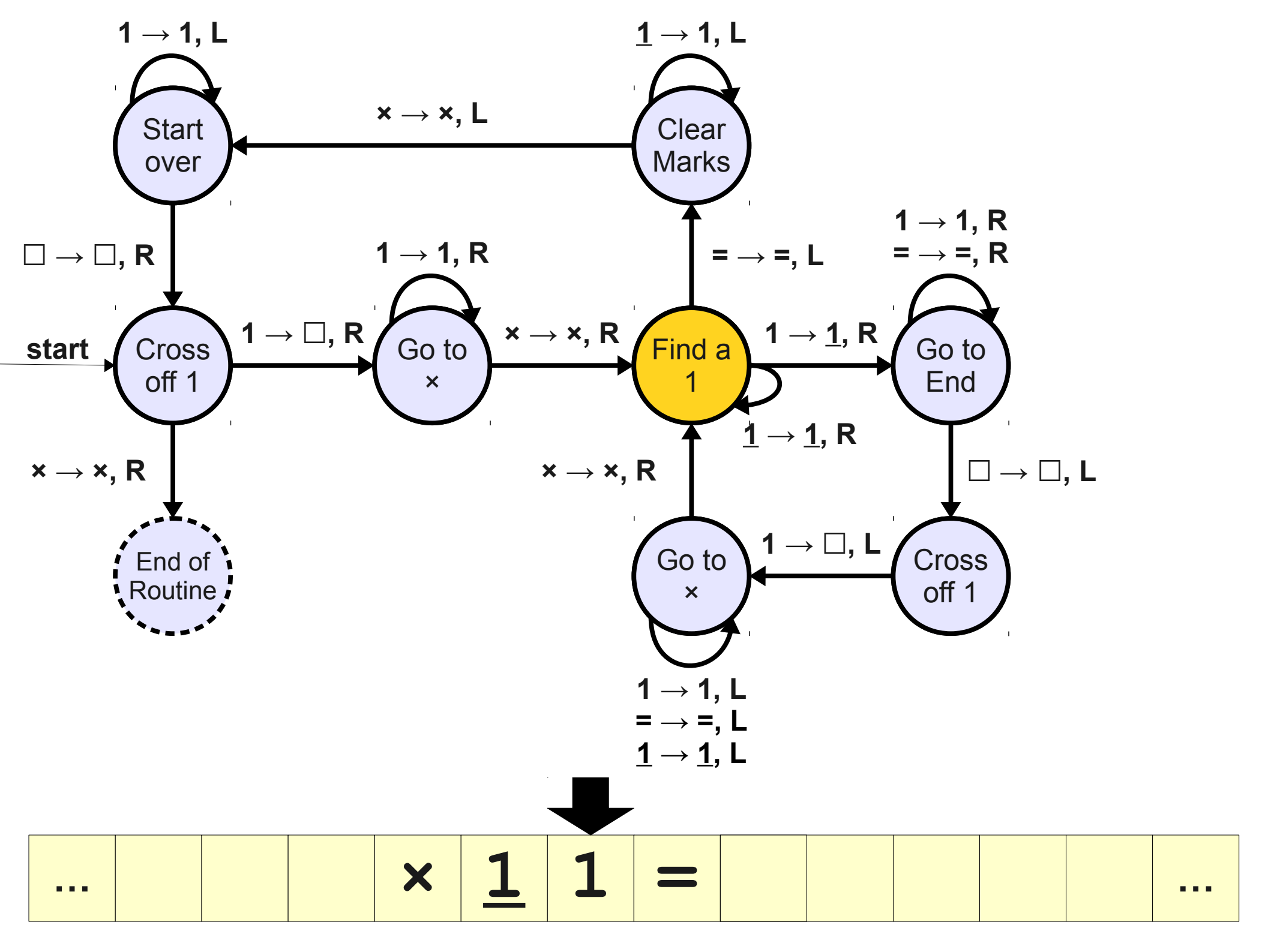

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

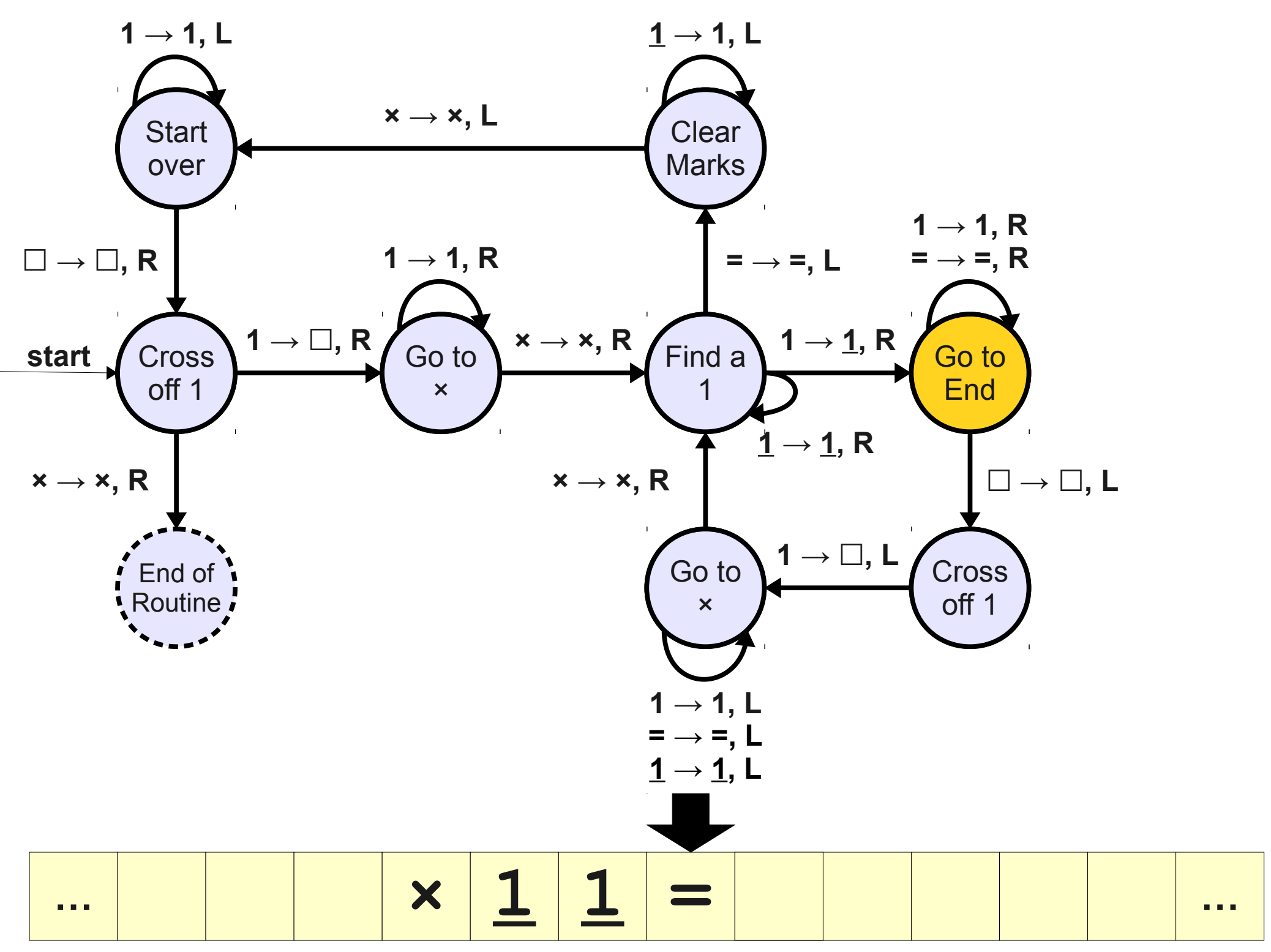

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

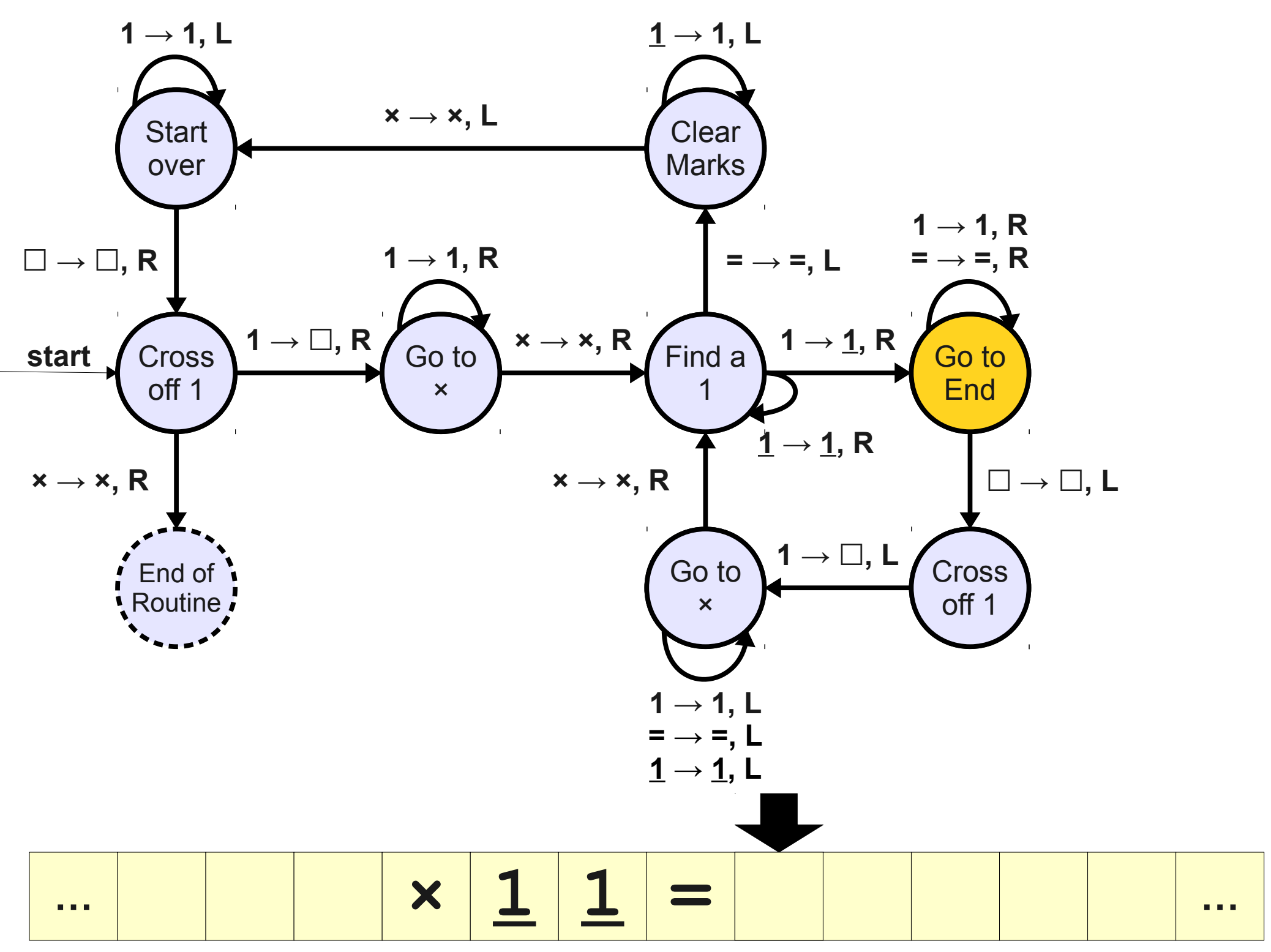

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

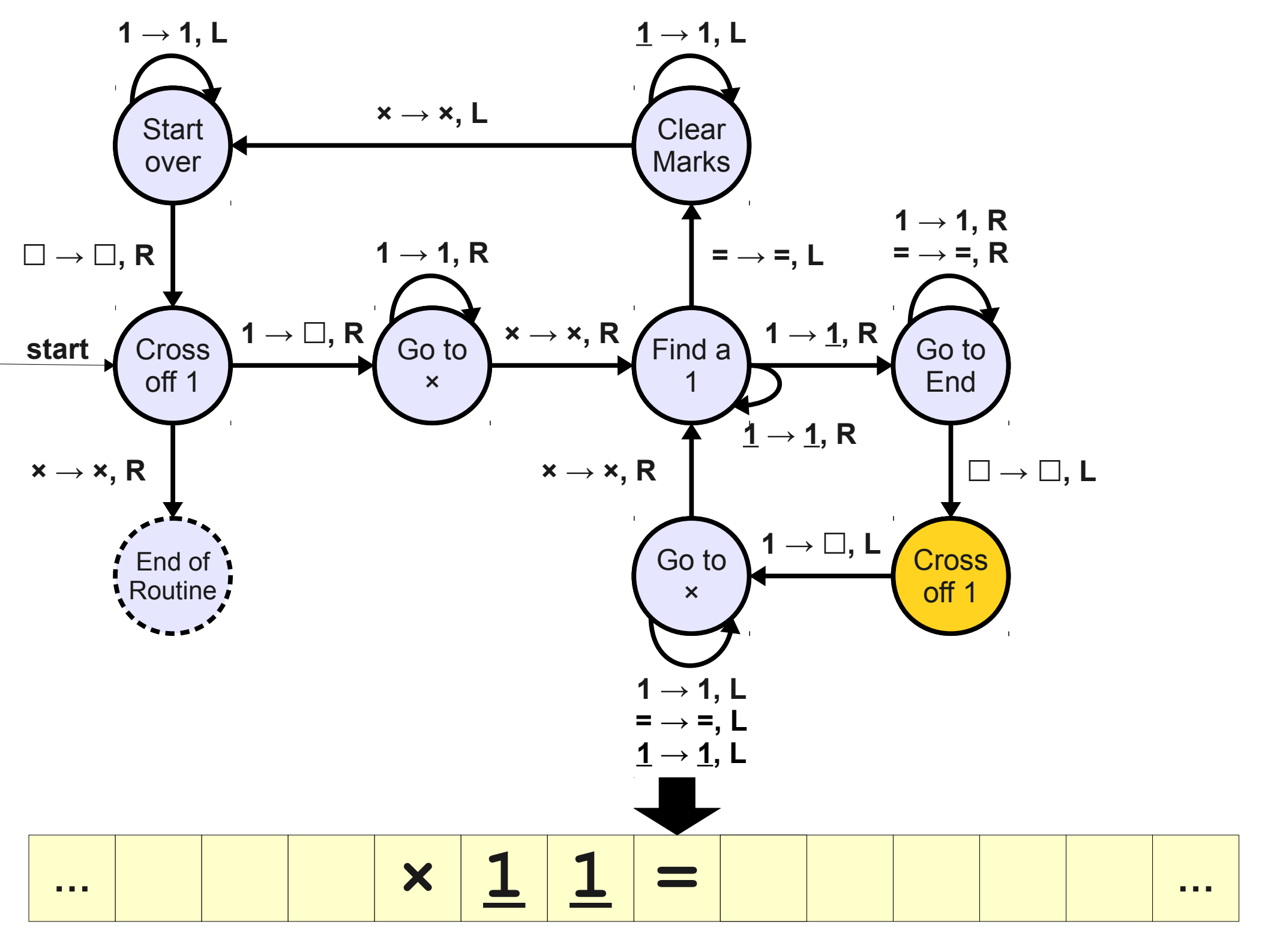

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

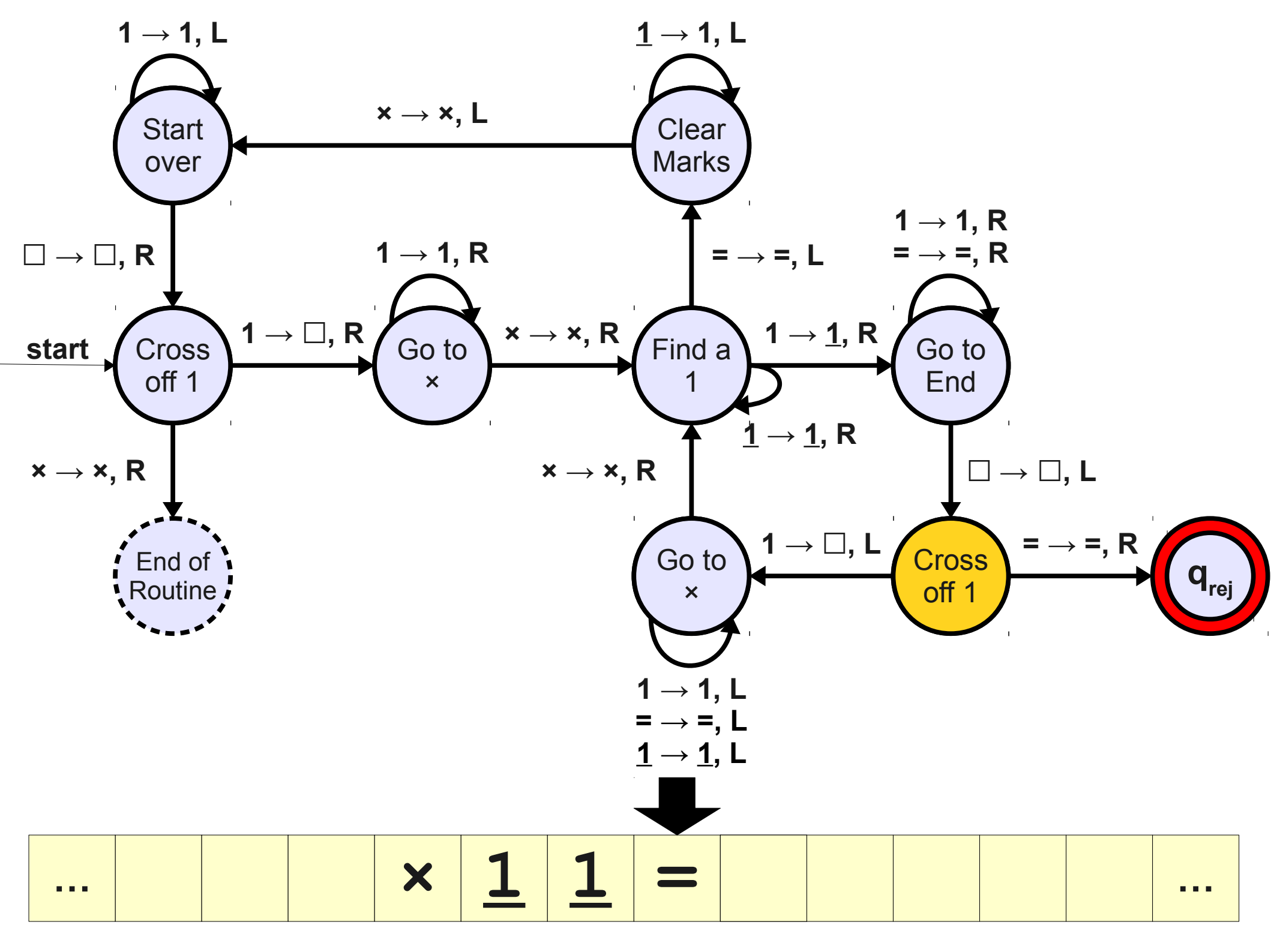

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

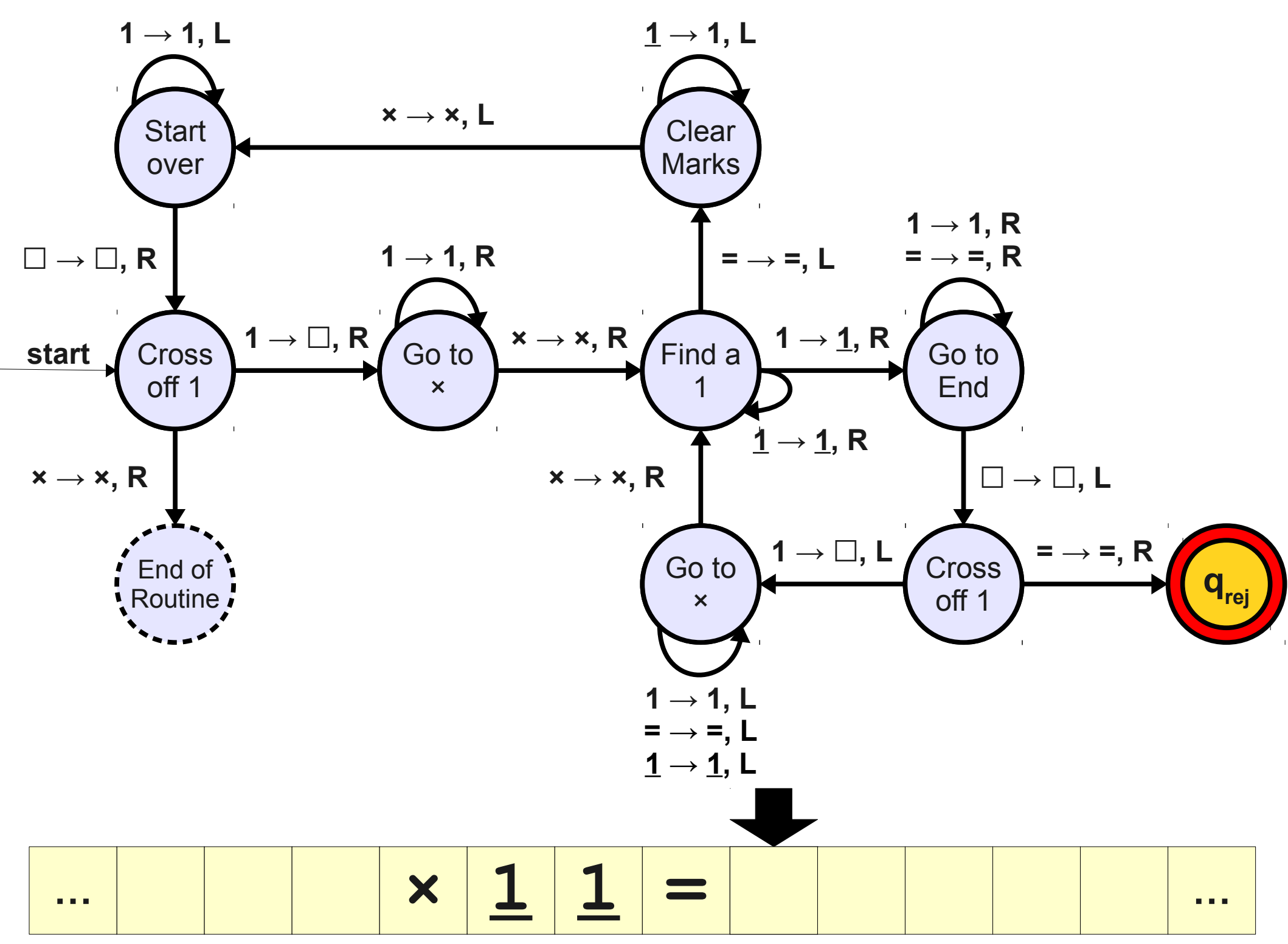

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

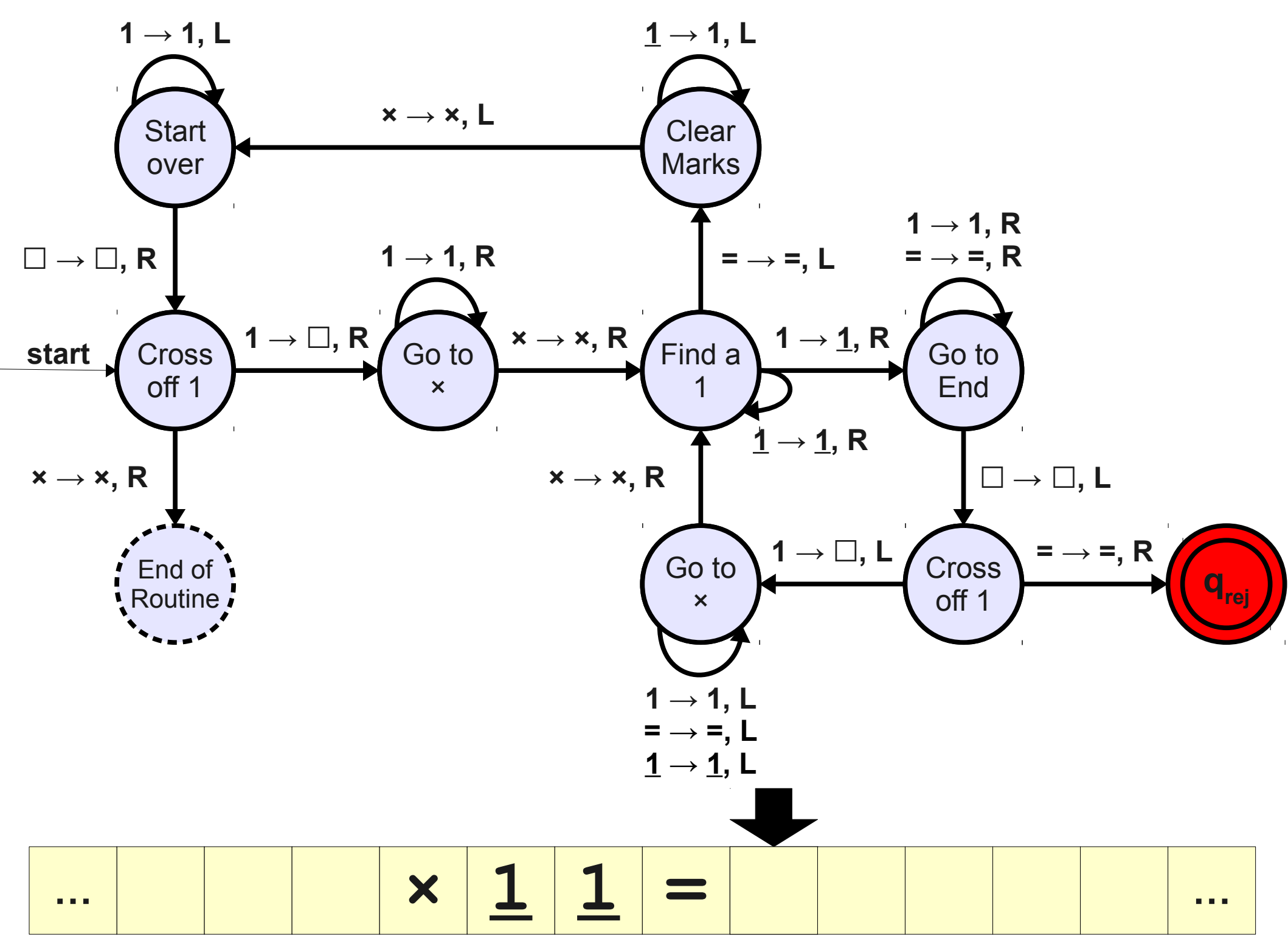

1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...


1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...


1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...


1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
= → =, R
q_rej
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...


1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
= → =, R
q_{rej}
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...


1 → 1, L
Start over
× → ×, L
1 → 1, L
Clear Marks
□ → □, R
1 → 1, R
= → =, L
1 → 1, R
= → =, R
start
Cross off 1
1 → □, R
Go to ×
× → ×, R
Find a 1
1 → 1, R
Go to End
1 → 1, R
× → ×, R
× → ×, R
□ → □, L
End of Routine
Go to ×
1 → □, L
Cross off 1
= → =, R
q_{rej}
1 → 1, L
= → =, L
1 → 1, L
... × 1 1 = ...

The Final Piece

- If $m = 0$, we need to check that $p = 0$.
- Input has form $\texttt{×1}^n\texttt{=1}^p$.
- In other words, accept iff string matches the regular expression $\texttt{×1}^*\texttt{=}$.
- Exercise: Build a TM to check this!

Turing Machines and Math

- Turing machines are capable of performing
  - Addition
  - Subtraction
  - Multiplication
  - Integer division
  - Exponentiation
  - Integer logarithms
  - **Plus a whole lot more…**

Next Time

- **More Turing Machines**
  - Worklist approaches.
- **Nondeterministic Turing Machines**
  - Turing machines with Magic Superpowers!
  - How powerful are they?
- **The Church-Turing Thesis**
  - Just how powerful *are* Turing machines?