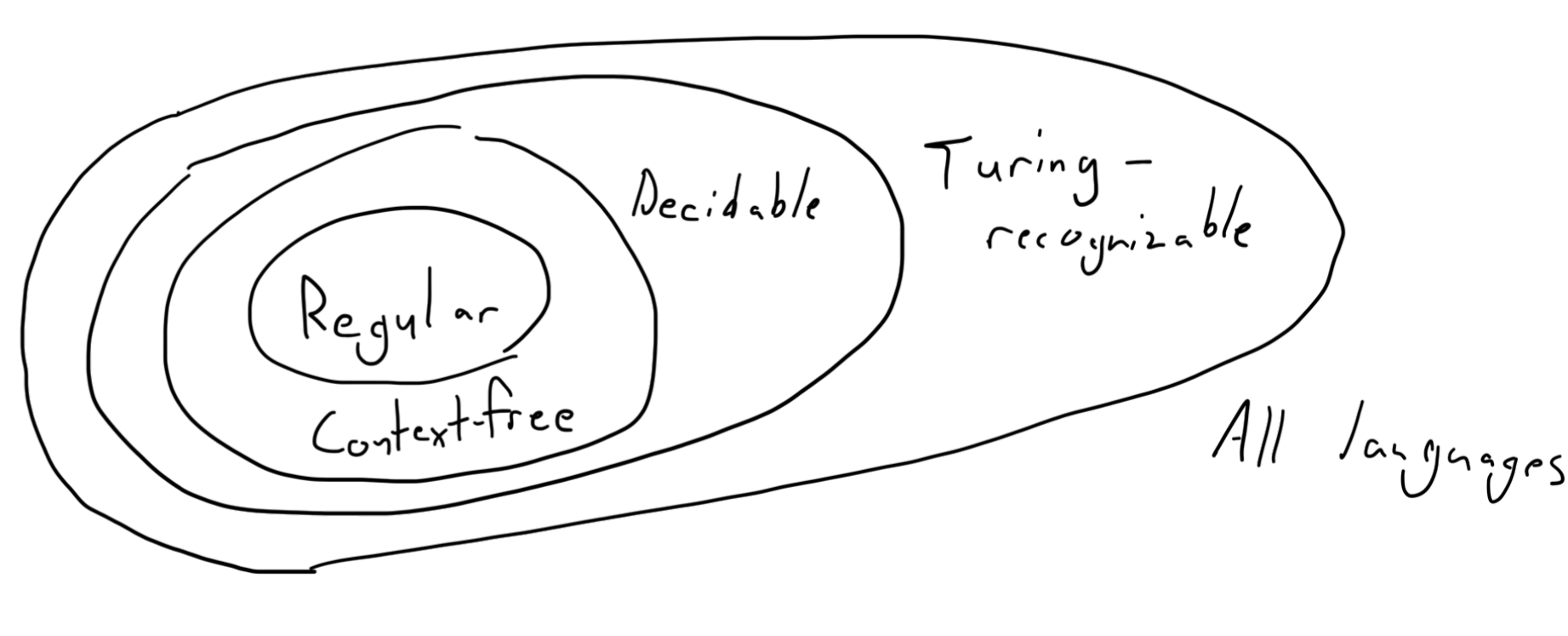


Ch.3 Church-Turing Thesis



1930's: What does "computable" mean?

Alan Turing was the PhD student of Alonzo Church. They concluded that

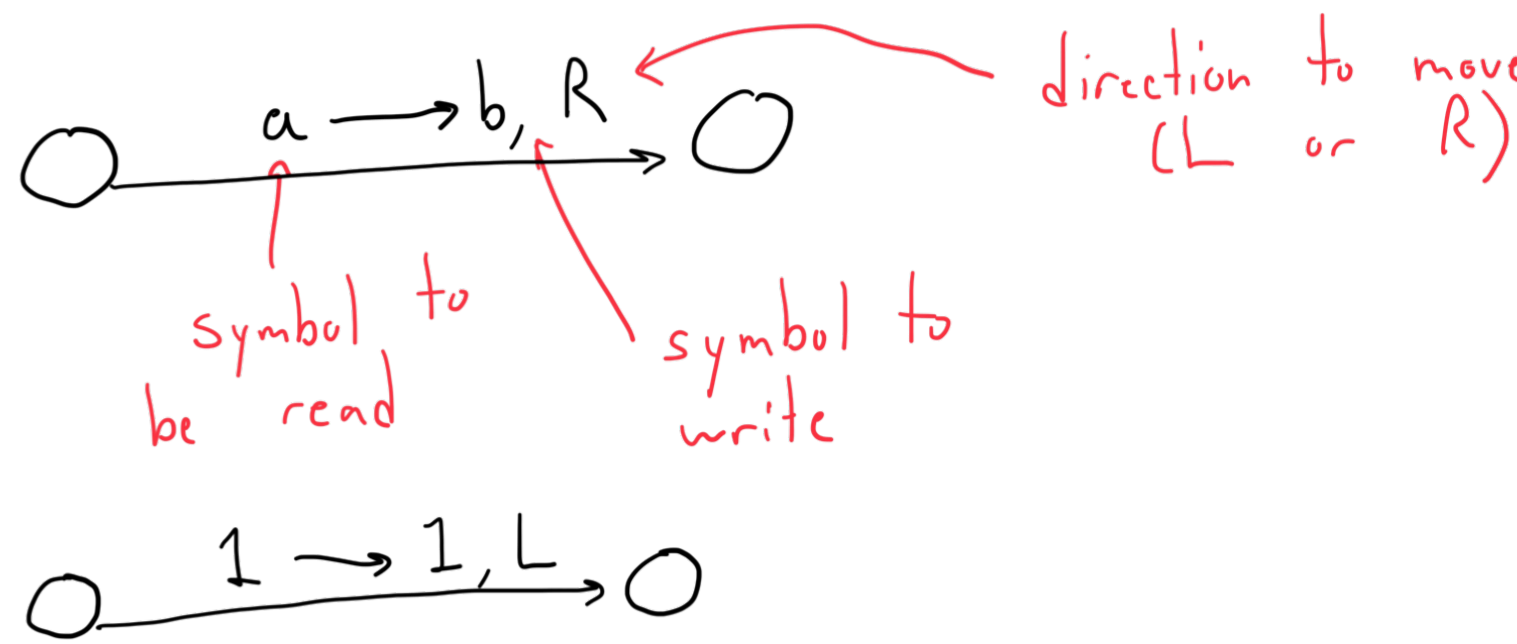
"Algorithmically computable" equals
"Computable by a Turing Machine (TM)"

A TM uses an infinite tape initially containing only the input string and blank everywhere else.



Rules of operation) At each step of computation:

1. Read current symbol
2. Update (i.e. write) to the same index
3. Move exactly one index (cell) either left or right. If we are at the left-most index and try to move left, do not move.



Formal Definition) A Turing Machine (TM) is a

7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where

Q, Σ, Γ are all finite states and:

1. Q is a set of states
2. Σ is the input alphabet, not including the blank symbol $\sqcup$
3. Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$
4. $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is the transition function
5. $q_0 \in Q$ is the start state.
6. $q_{\text{accept}} \in Q$ is the accept state.
7. $q_{\text{reject}} \in Q$ is the reject state where $q_{\text{reject}} \neq q_{\text{accept}}$.

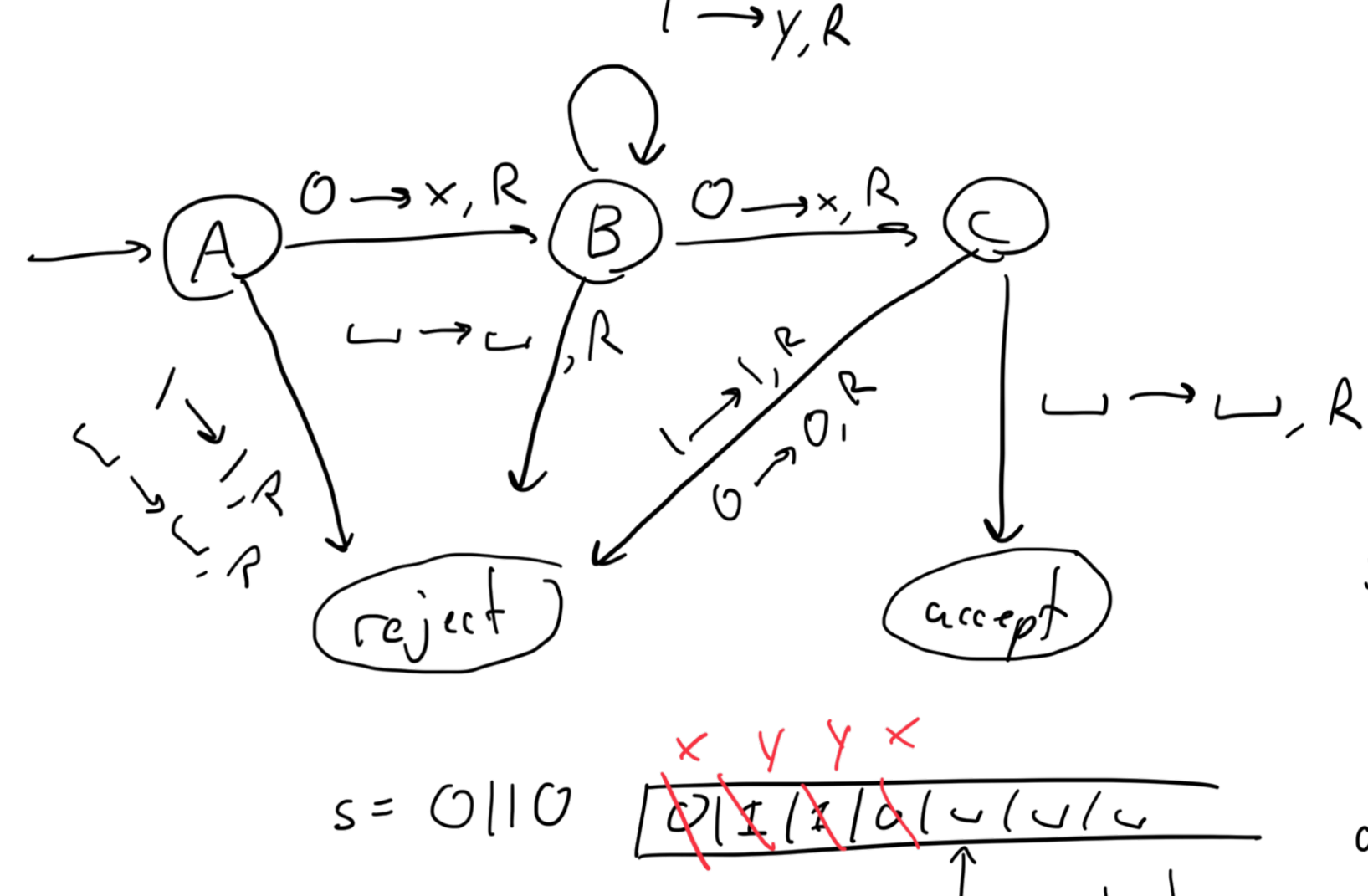
Computation can

- Halt and accept. When a TM enters the accept state, computation immediately halts.
- Halt and reject (... enters a reject state...)
- Loop. It fails to halt.

Given language L , if a TM "halts & accepts" on all $s \in L$ and "halts & rejects" on all $s \notin L$, then L is decidable and the TM is a "decider."

Given language L , if a TM always halts and accepts when $s \in L$, then L is Turing-recognizable. And, the TM is a "recognizer."

Ex 1 $L = 01^*0$



Any string in L leads to a "halt & reject" or "halt & accepts", so the TM is a decider. This proves L is decidable (because at least one TM decides L).

How do you prove a language is decidable?

Construct some TM that is a decider!

So, the TM "halts and accepts" on every $s \in L$ and "halts and rejects" on every $s \notin L$.

How do you prove a language is Turing-recognizable?

Construct some TM that is a recognizer!

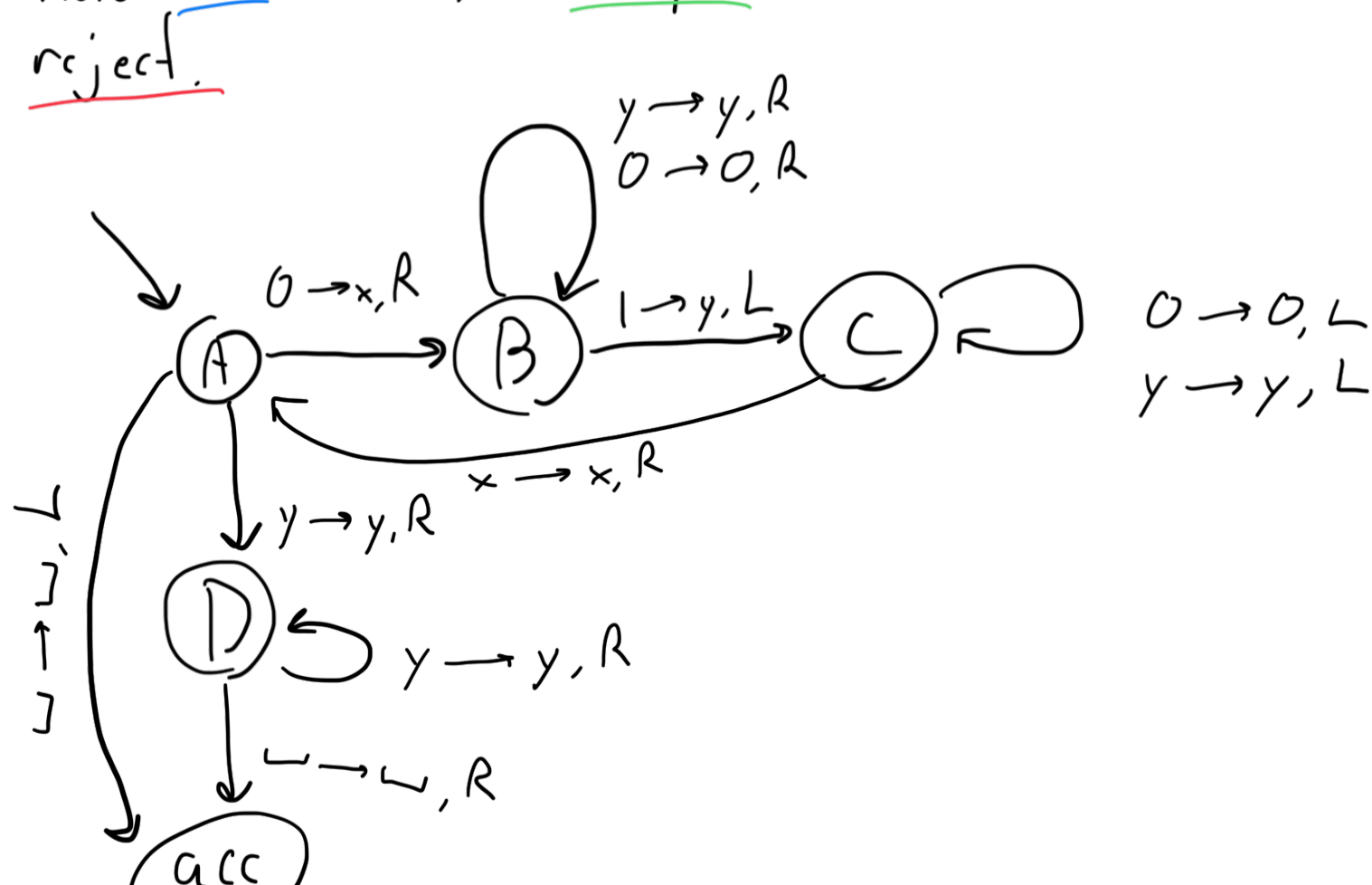
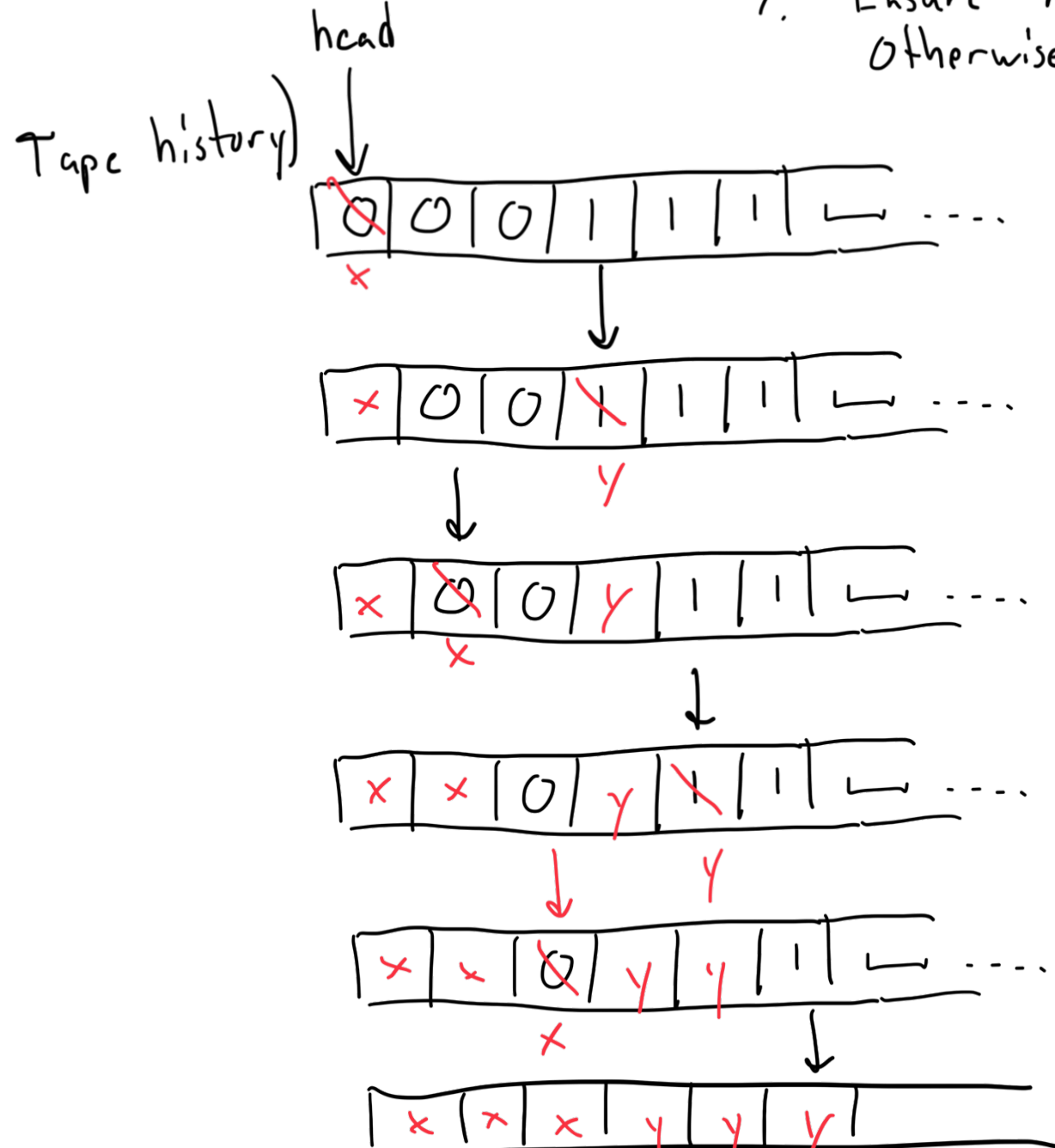
So, the TM "halts and accepts" iff $s \in L$. (if and only if)

Ex 2 $L = 0^n 1^n$ s.t. $n \geq 0$

$\Sigma = \{0, 1\}$

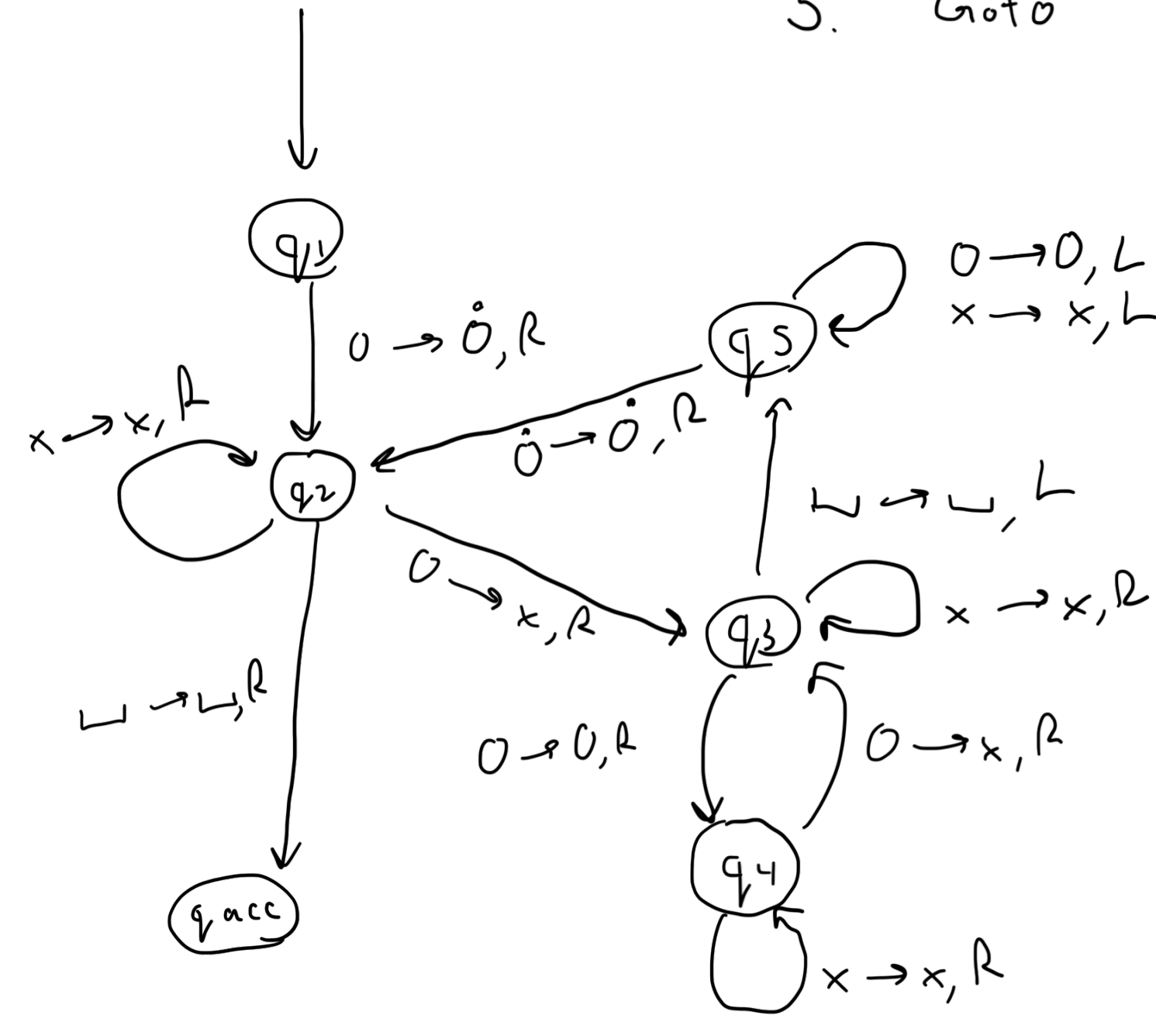
$\Gamma = \{0, 1, x, y, \sqcup\}$

- TM Algorithm)
1. If there are no symbols, accept.
 2. If we have a 0, change it to x.
↳ if not, reject
 3. Move tape head right to first 1.
↳ if it does not exist, reject
 4. Change 1 to y.
 5. Move left to left-most 0.
 6. Goto step 1 if 0's remain
 7. Ensure no more 1's remain, the accept. Otherwise, reject.



Ex 3 $A = \{0^{2^n} : n \geq 0\}$ M: "On input string w :

1. Sweep left to right, crossing off every other 0.
2. If in stage 1 the tape contains a single 0, accept.
3. If in stage 1 the tape contains more than a single 0, and the # of 0's was odd, reject.
4. Return to left end of tape.
5. Goto stage 1.



configuration

0000
↑ tape head
q₁ 0000

0 q₂ 000

0 x q₃ 00

0 x 0 q₄ 0

0 x 0 x q₃ x

0 x 0 q₂ x

0 x q₃ 0 x

0 q₃ x x

q₃ 0 x x x

0 q₂ x x x

0 x q₂ x x

0 x x q₂ x

0 x x x q₂ x

0 x x x x q_{acc}