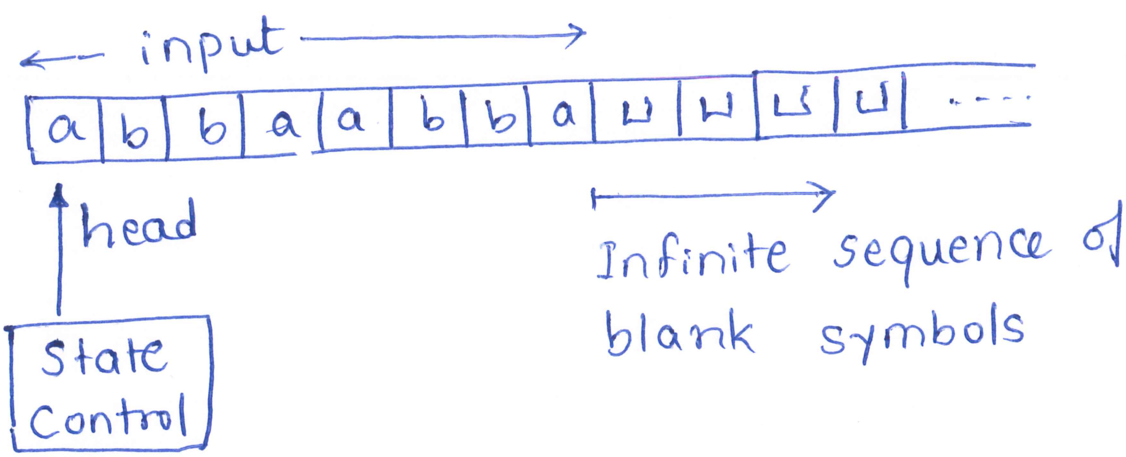
Computability Theory

- Turing machine model. Defined by Alan Turing in 1936.
- This turns out to be a model for what we now consider "real", "general purpose" computer.

Informal Description of TM model

Tape:



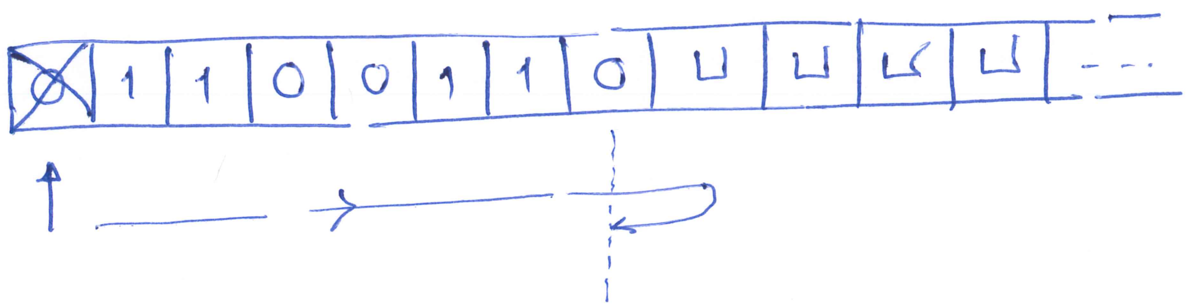
- Infinite tape. Initially, the input is written as a prefix on the tape followed by infinite seq. of blanks.
- Tape can be read or written using the "head". The head can move left or right (by one position in one step). Initially,

the head points to first/leftmost cell.

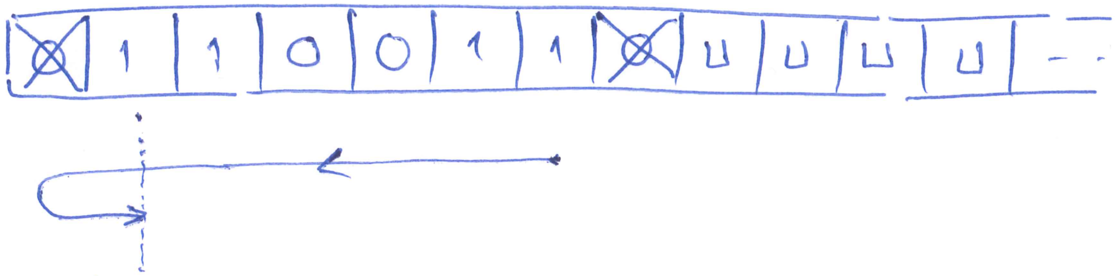
- Machine has an internal state that may change. It can use extra symbols.
- Two of the states are designated as Accept and Reject states. If/when the m/c enters these states, it halts and accepts or rejects the input respectively.

<u>Example</u> TM that recognizes palindromes.

$$L = \{ w \in \{0,1\}^* \mid w \text{ is a palindrome} \}.$$



M/c reads and remembers the first symbol, say '0'. It replaces that symbol by $\boxtimes$ symbol (for future convenience), moves to the right until it reads '⊔' signifying the end of the input, and turns left.



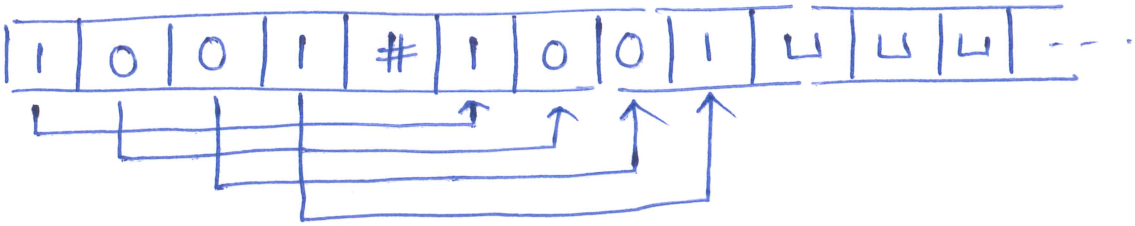
The symbol read by the head now is the last symbol of the input. If this symbol matches the symbol remembered, it is replaced by $\boxtimes$, the head moves to the left until it reads $\boxtimes$ and turns right. The head now points to the second symbol, say '1', of the input.

In similar manner, the m/c keeps going back and forth, matching corresponding symbols from the front and the end. If all symbols are matched successfully, m/c accepts. If an "unmatch" is detected at any step, the m/c rejects. ∎

Example TM that recognizes

$$L = \{ w\#w \mid w \in \{0,1\}^* \}.$$

(Note. L is not context free).



The symbols need to be matched as shown. Describe the m/c informally.

Formal Definition of Turing Machine

We consider a TM that is supposed to recognize a language over (input) alphabet Σ. There is one more blank symbol '$\sqcup$' as referred to before and moreover the m/c may use extra symbols for its convenience. The set of all symbols is denoted by Γ, the "tape alphabet". I.e.

$$\Sigma \subseteq \Gamma, \qquad '\sqcup' \in \Gamma \setminus \Sigma.$$

The m/c has a set Q of states. A typical move/instruction of the m/c specifies :

Given the current state and the tape symbol read by the head,

- (Possibly) change state
- (Possibly) replace the tape symbol
- Move the head to left or right by one position.

Therefore the set of all moves is described by the "transition function"

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$$

Left Right.

Finally, three of the states are specially designated as start, accept, reject states.

Def A Turing Machine M is a 7-tuple

$$M = (Q, \Sigma, \Gamma, \delta, q_{start}, q_{accept}, q_{reject})$$

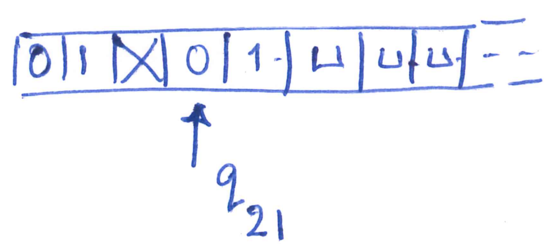
where

- Q is a finite set of states.
- Σ " input alphabet.
- Γ " tape alphabet, $\Sigma \subseteq \Gamma$, '$\sqcup$' $\in \Gamma \setminus \Sigma$.
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is transition f^n.
- q_{start} is the start state.
  q_{accept} " accept ".
  q_{reject} " reject ".

These are included in Q.

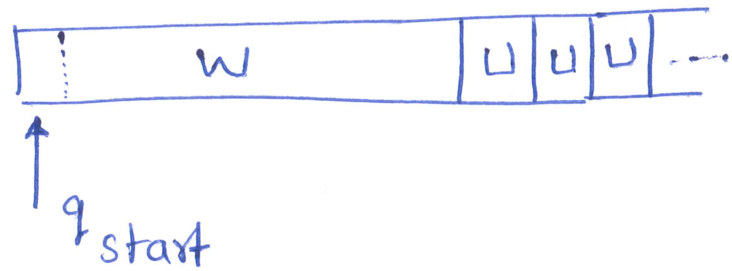
Def. A "configuration" of a TM consists of

- Entire content of the tape.
- The State.
- Position of the head.



Def. The initial configuration of a TM on input $w \in \Sigma^*$ consists of

- State is q_{start}

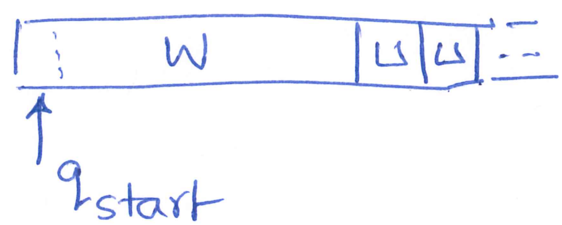


- Head points to the first (i.e. leftmost) cell.
- The input w is written as a prefix on the tape, followed by infinite seq. of '⊔'s.

—— x ——

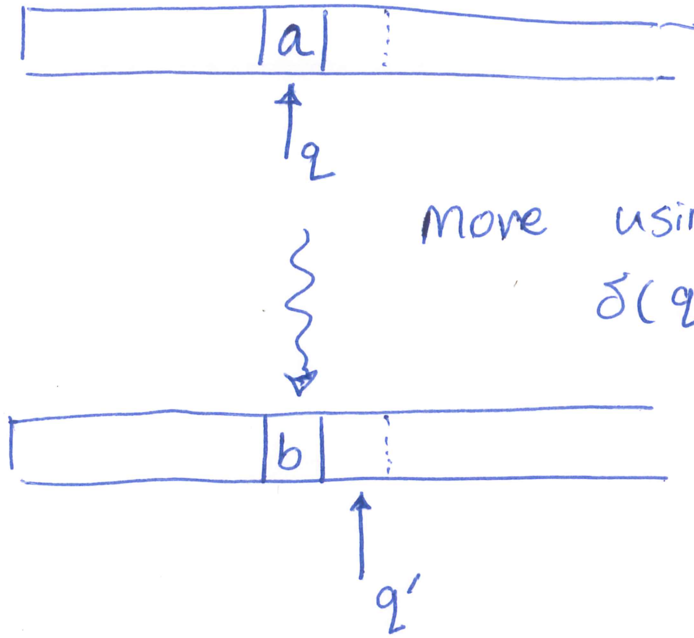
Computation of a TM on input $w \in \Sigma^*$

- M/c starts in the initial configuration.



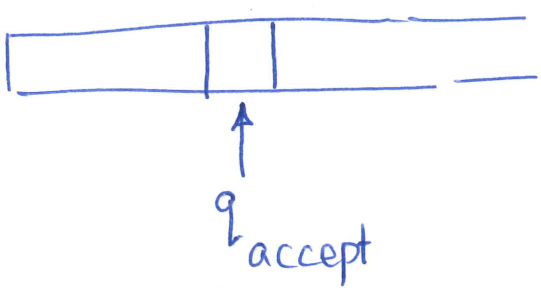
- M/c makes (deterministic) moves as long as the state is not q_{accept} or q_{reject}.

A typical move, along with intermediate configurations, looks like

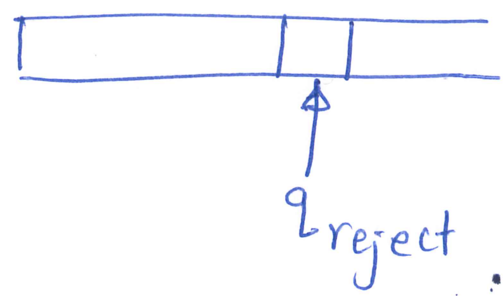


Move using instruction $\delta(q, a) = (q', b, R)$.

- If/when the state is q_{accept} or q_{reject}, the m/c halts and accepts or rejects accordingly. So the final/end configuration is



OR



.

Note It is possible that on certain input $w \in \Sigma^*$, the machine never halts and "runs forever".

Def For a TM M, the language $\boxed{\text{recognized}}$ by it, denoted $L(M)$ is :

$$L(M) = \left\{ w \in \Sigma^* \;\middle|\; \begin{array}{l} \text{On input } w,\ M \text{ (eventually)} \\ \text{halts and accepts} \end{array} \right\}.$$

Note - On inputs $w \notin L(M)$, M may never halt.

I.e.

$w \in L(M) \Rightarrow M$ halts and accepts on w.

$w \notin L(M) \Rightarrow M$ halts and rejects
<u>OR</u> never halts on w.

Def A language L is <u>Turing-recognizable</u> if it is recognized by some TM M.

I.e. if there is a TM M s.t.

$w \in L \Rightarrow M$ halts and accepts on w.

$w \notin L \Rightarrow M$ halts and rejects
<u>OR</u> never halts on w.

Example of a TM that never halts:

Consider the TM that stays in the start state and keeps moving the head to the right.

I.e. $\delta(q_{start}, a) = (q_{start}, a, R) \ \forall a \in \Gamma$.

For this m/c, $L(M) = \phi$.

It never halts no matter what the input is.

—— x ——

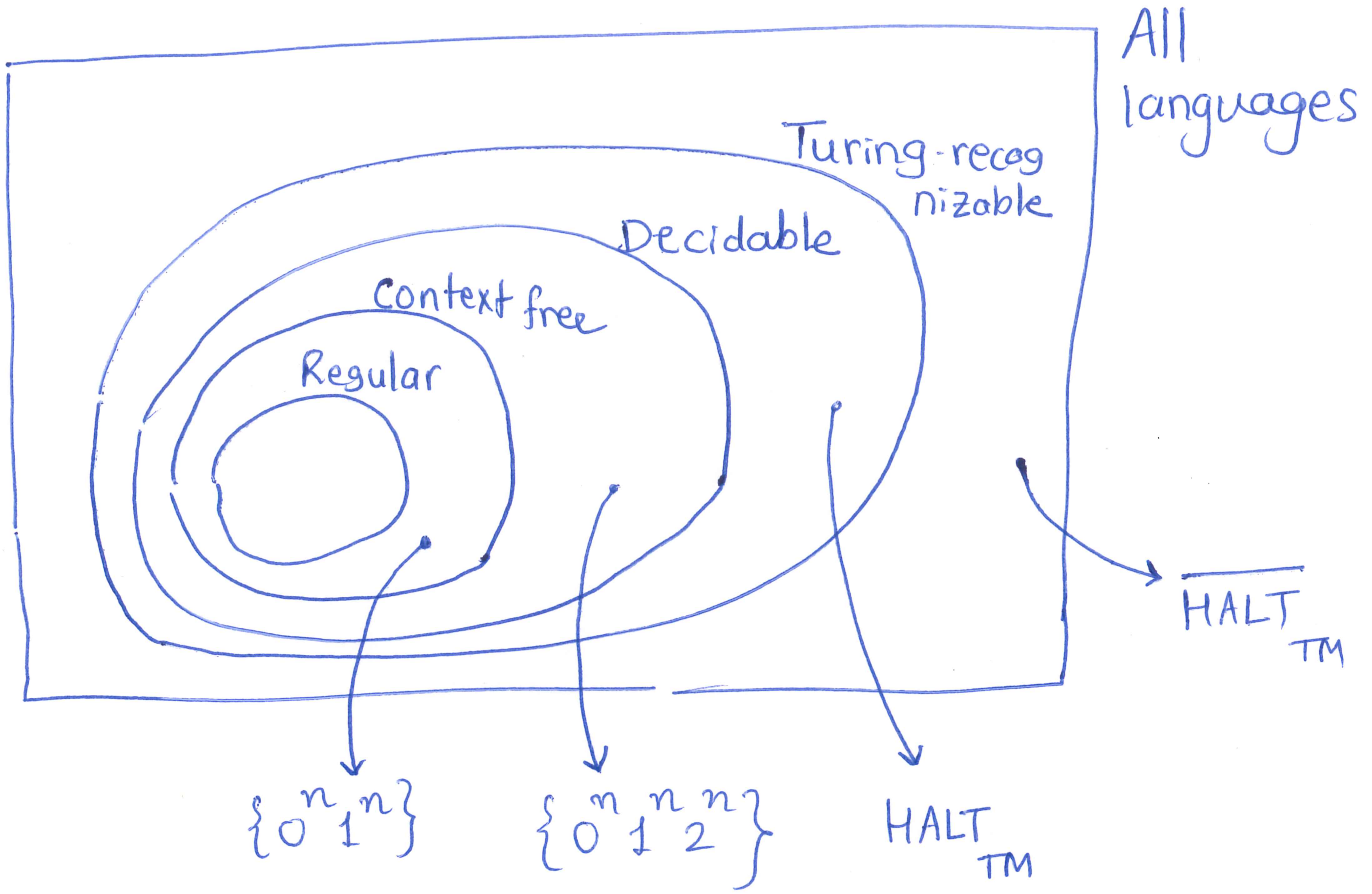
It may be the case that a TM M does halt on every input. In this case L(M) is said to be a decidable language (and decided by M).

Def. A language L is decidable if there is a TM M that halts on every input and

$w \in L \Rightarrow$ M halts and accepts w.

$w \notin L \Rightarrow$ M halts and rejects w.

Inclusion of Classes of Languages



This inclusion is strict:

$\{0^n 1^n\}$: Context-free but not regular.

$\{0^n 1^n 2^n\}$: Decidable but not context free.

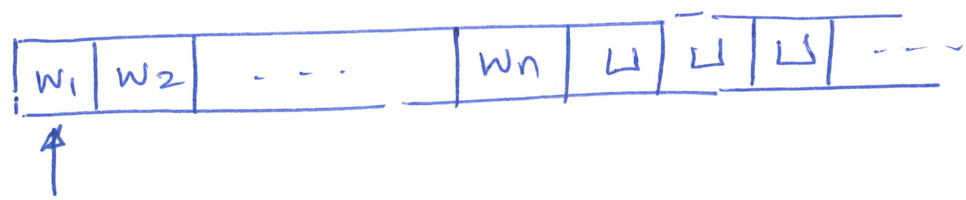
$HALT_{TM}$: Turing-recognizable but not decidable.
("Halting Problem", we will see it later).

$\overline{HALT_{TM}}$: Not Turing-recognizable.

Let's focus only on decidable languages for now and convince ourselves that a TM can accomplish "any" task that a "real" computer can.

<u>Example</u>. $\Sigma = \{a\}$.

$L = \{ a^n \mid n \text{ is even} \}$ is decidable.



The m/c simply keeps reading the input left to right, alternating its state between "even" and "odd" state, and halts when '$\sqcup$' is read (indicating end of the input).

Formally the m/c is described as:

$$Q = \{ q_{start} = q_{even}, q_{odd}, q_{accept}, q_{reject} \}$$

$$\delta(q_{even}, a) = (q_{odd}, a, R).$$

$$\delta(q_{odd}, a) = (q_{even}, a, R).$$

$$\delta(q_{even}, \sqcup) = (q_{accept}, \sqcup, R).$$

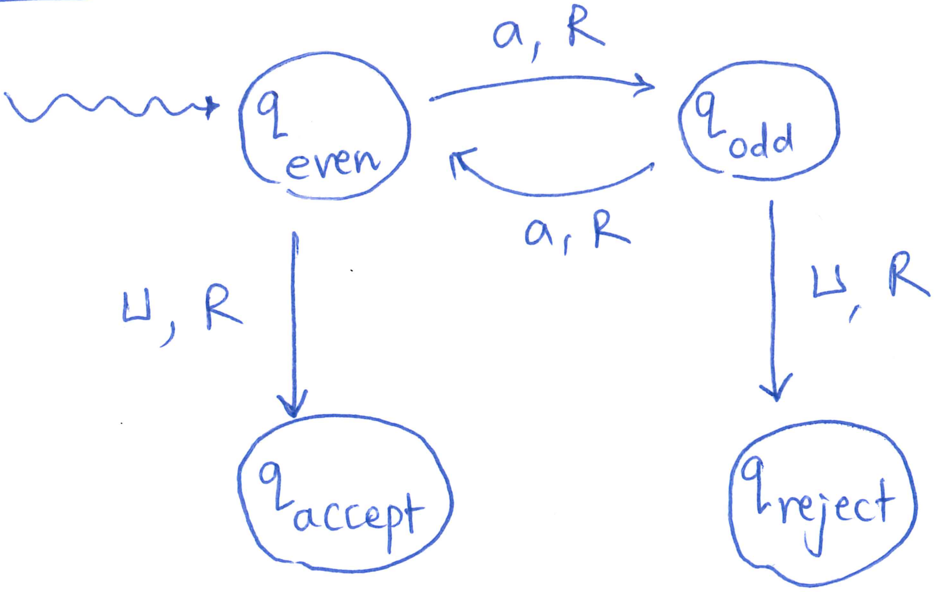
$$\delta(q_{odd}, \sqcup) = (q_{reject}, \sqcup, R).$$

The rest of the instructions, i.e. $\delta(q_{accept}, \ )$, $\delta(q_{reject}, \ )$, are irrelevant.

<u>State Diagram</u> An instruction $\delta(q, a) = (q', b, R)$ is represented as $q \xrightarrow{a \to b, R} q'$.

If $a = b$, i.e. if the tape symbol remains unchanged, one may simplify as $q \xrightarrow{a, R} q'$.

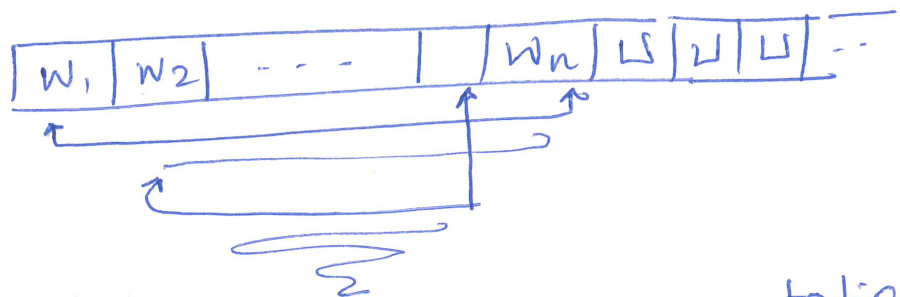
<u>Diagram for m/c above</u>



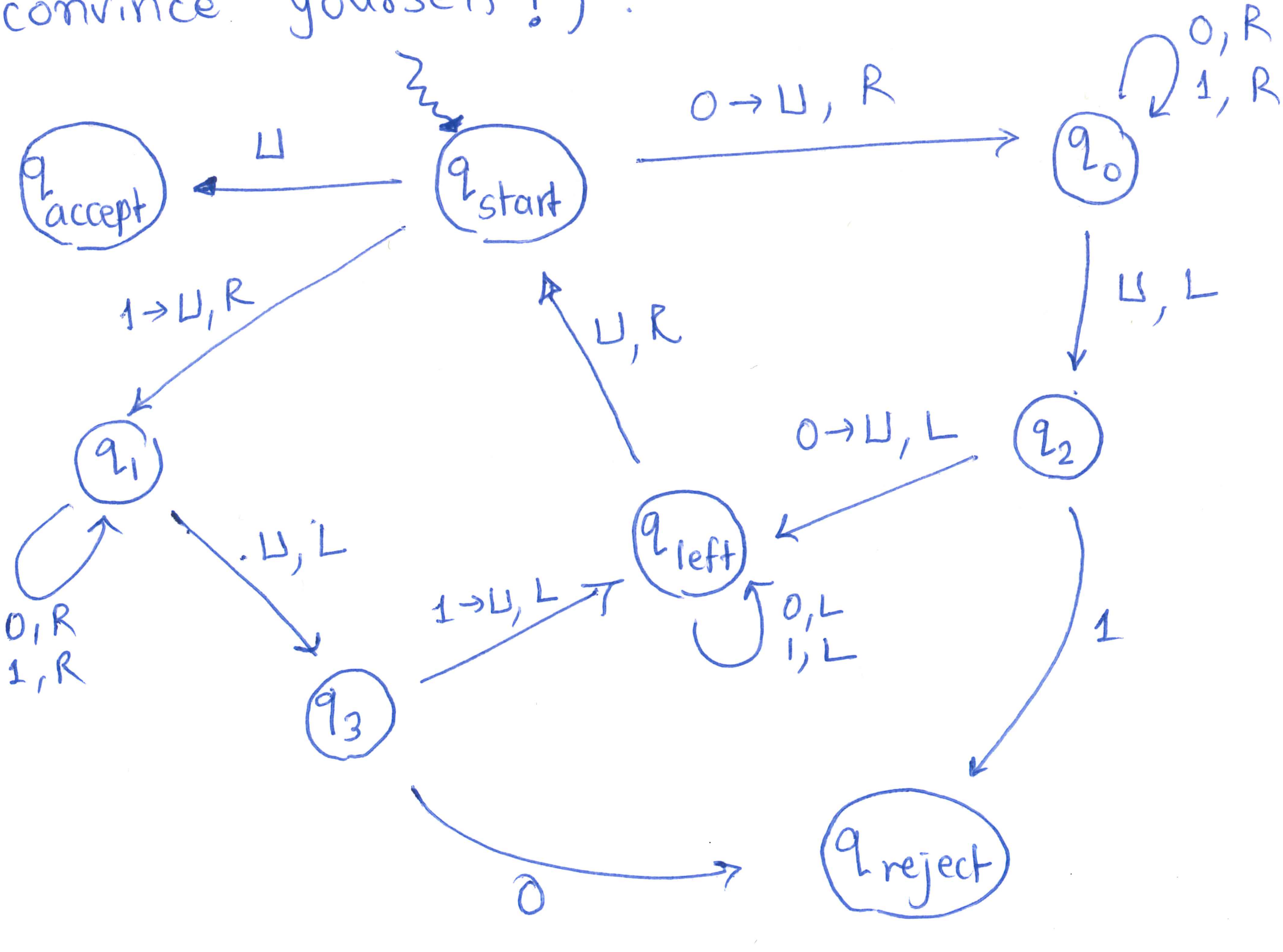
State Diagram of TM that decides

$L = \{ w \in \{0,1\}^* \mid w \text{ is a palindrome } \underline{\text{and}}\ |w| \text{ is even.} \}$.

$\Gamma = \{0, 1, \sqcup\}$.



The m/c below implements the computation that matches symbols from the front and the end, in back & forth fashion (convince yourself!).



Roughly speaking

- q_0 is the "mode" where the m/c has remembered a '0' and moving to the end.

- q_1 is the "mode" where the m/c has remembered a '1' and moving to the end.

- q_{left} is the "mode" where the m/c is moving back to the front.

In state q_2, the m/c is trying to match the '0' that is rememberd (hence reject if it reads '1').

In state q_3, the m/c is trying to match the '1' that is remembered (hence reject if it reads '0'). ▨

See textbook for more examples!

$L = \{ 0^{2^n} \mid n \geq 0 \}$ is decided by 7-state TM.

$L = \{ w\#w \mid w \in \{0,1\}^* \}$ " 14-State TM.