CSE 105
THEORY OF COMPUTATION

Spring 2018

http://cseweb.ucsd.edu/classes/sp18/cse105-ab/

Today's learning goals

Sipser Section 3.1

- Trace the computation of a Turing machine on given input
- Describe the language recognized by a Turing machine
- Determine if a Turing machine is a decider
- Give an implementation-level description of a Turing machine

Turing machine computation

- Read/write head starts at leftmost position on tape
- Input string written on leftmost squares of tape, rest is blank
- Computation proceeds according to transition function:
  - Given current state of machine, and current symbol being read
  - the machine
    - transitions to new state
    - writes a symbol to its current position (overwriting existing symbol)
    - moves the tape head L or R (if possible)
- Computation ends **if and when** machine enters either the **accept** or the **reject** state.

Language of a Turing machine

L(M) = { w | computation of M on w halts after entering the **accept** state}

i.e. L(M) = { w | w is accepted by M}

Language of a TM

Sipser p. 144

$L(M) = \{ w \mid M \text{ accepts } w\}$

Which of the following is not always true?

A. If w is in L(M) then the computation of M on w halts and accepts.
B. If the computation of M on w halts and rejects, then w is not in L(M).
C. If w is not in L(M) then the computation of M on w halts and rejects.

Configuration

To trace DFAs: enough to list states.
To trace NFAs: tree of possible current states (incl. spontaneous moves)
To trace PDAs: tree of possible computations incl. state + stack

- Current state
- Current tape contents up to (finite) point after which all blank
- Current location of read/write head

$$u\ q\ v$$

current state is q
current tape contents are uv (and then all blanks)
current head location is first symbol of v

Special configurations

For input string w

- Starting configuration $q_0\ w$
- Accepting configuration $u\ q_{acc}\ v$
- Rejecting configuration $u\ q_{rej}\ v$

$\left.\begin{matrix} u\ q_{acc}\ v \\ u\ q_{rej}\ v \end{matrix}\right\}$ Halting configurations

current state is q
current tape contents are uv (and then all blanks)
current head location is first symbol of v

Language of a TM

Sipser p. 144

$L(M) = \{ w \mid M \text{ accepts } w\}$

= { w | there is a sequence of configurations of M

where C_1 is start configuration of M on input w,

each C_i yields C_{i+1} and C_k is accepting configuration}

"The language of M"

"The language recognized by M"

An example

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

We already know that L is

- not regular
- not context-free

We will prove that L is

the language of some Turing machine

Implementation-level description

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

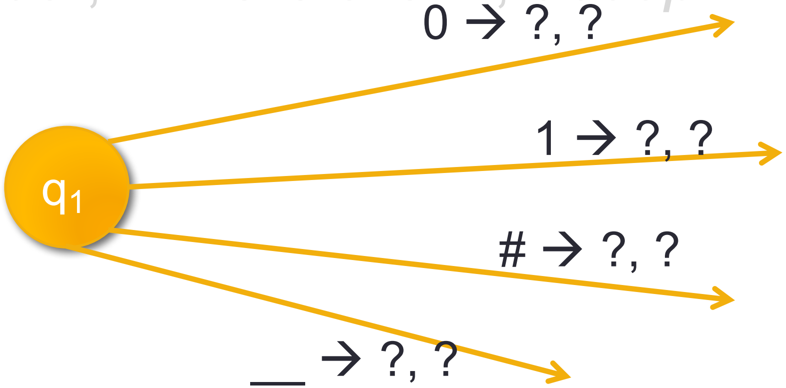
Idea for Turing machine

- **Zig-zag across tape** to corresponding positions on either side of '#' to check whether these positions agree. If they do not, or if there is no '#', *reject*. If they do, cross them off.
- Once all symbols to the left of the '#' are crossed off, check for any un-crossed-off symbols to the right of '#': if there are any, *reject*; if there aren't, *accept*.

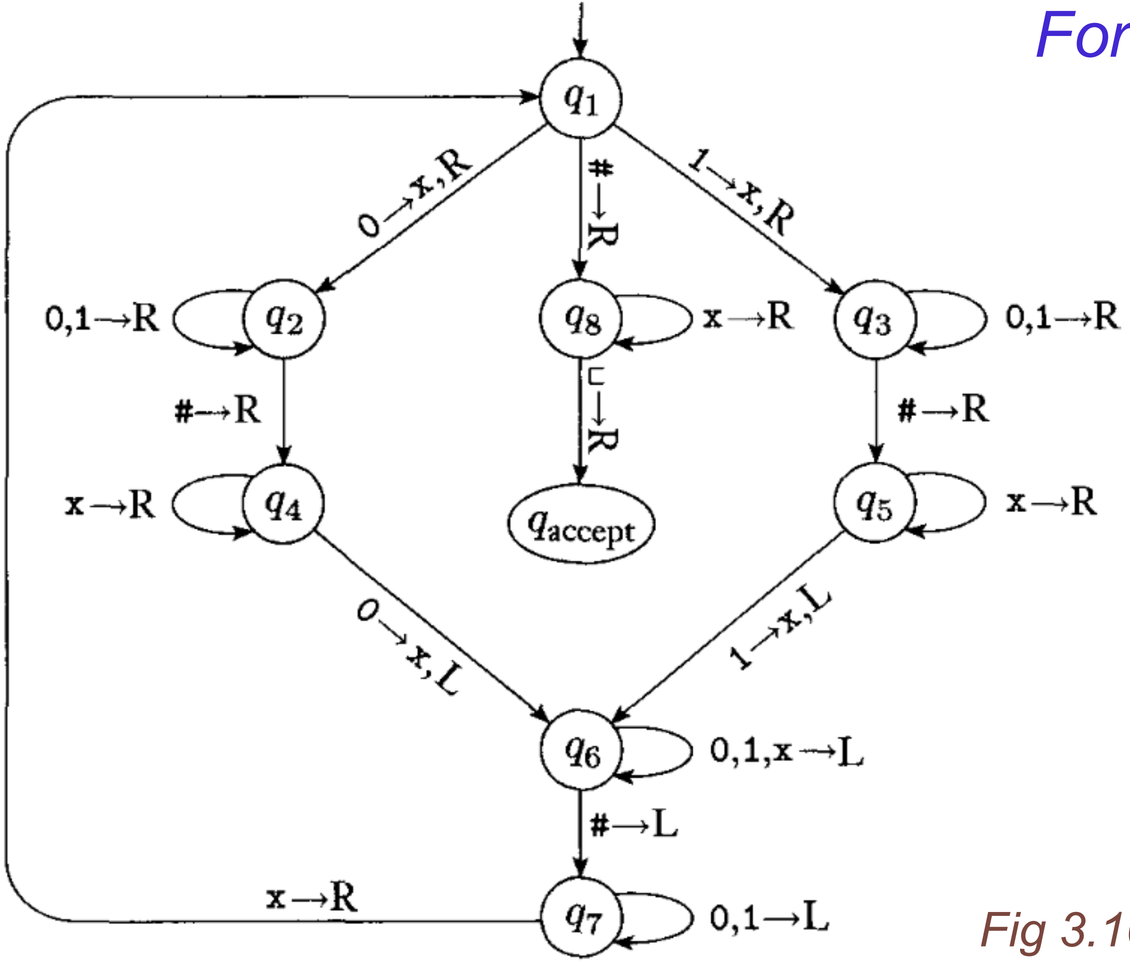
Implementation-level description

Zig-zag across tape to corresponding positions on either side of '#' to check whether these positions agree. If they do not, or if there is no '#', *reject*. If they do, cross them off.

Once all symbols to the left of the '#' are crossed off, check for any un-crossed-off symbols to the right of '#': if there are any, *reject*; if there aren't, *accept*.



State diagram



Formal definition

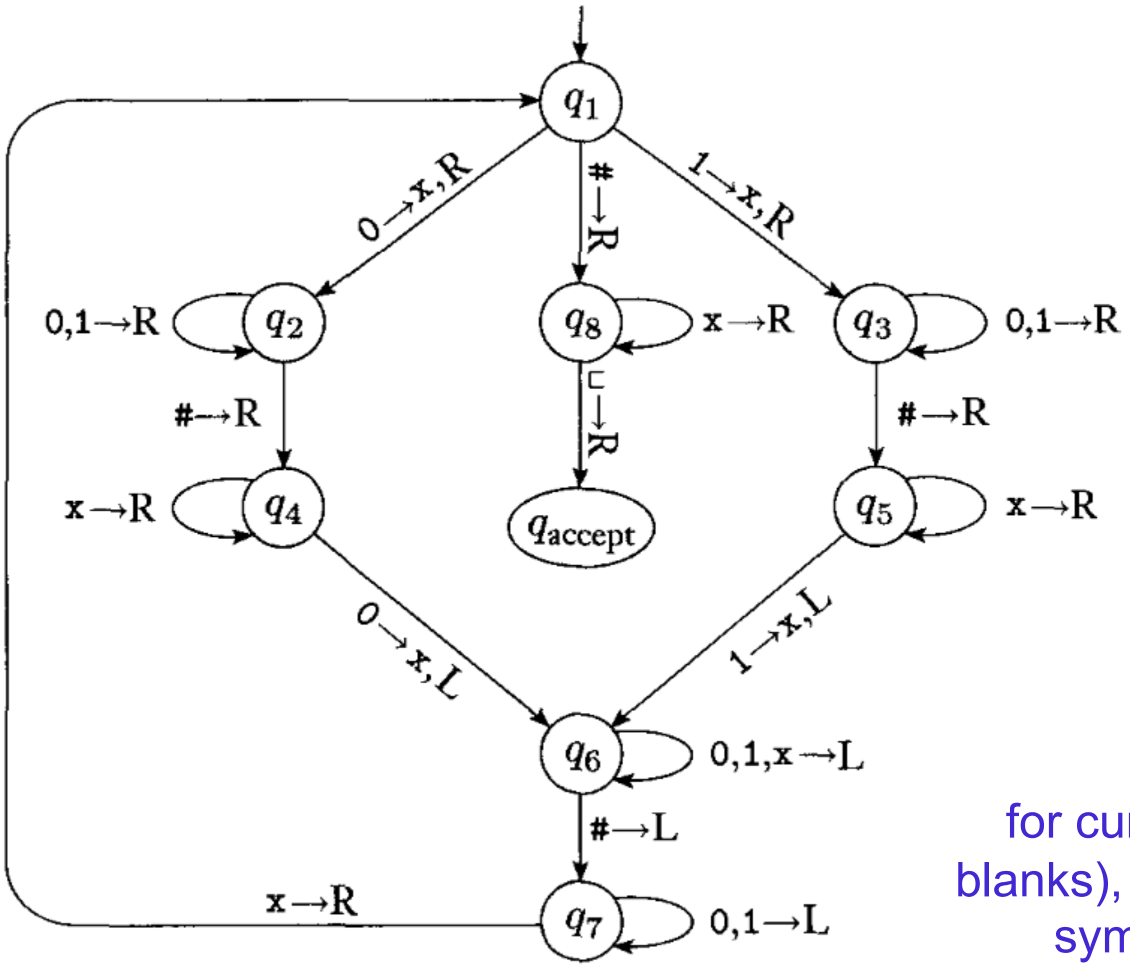
$Q=\{q1,q2,q3,q4,q5,q6,q7,q8,q_{accept},q_{reject}\}$

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

$\Gamma = \{0,1,\#,x,\_\}$

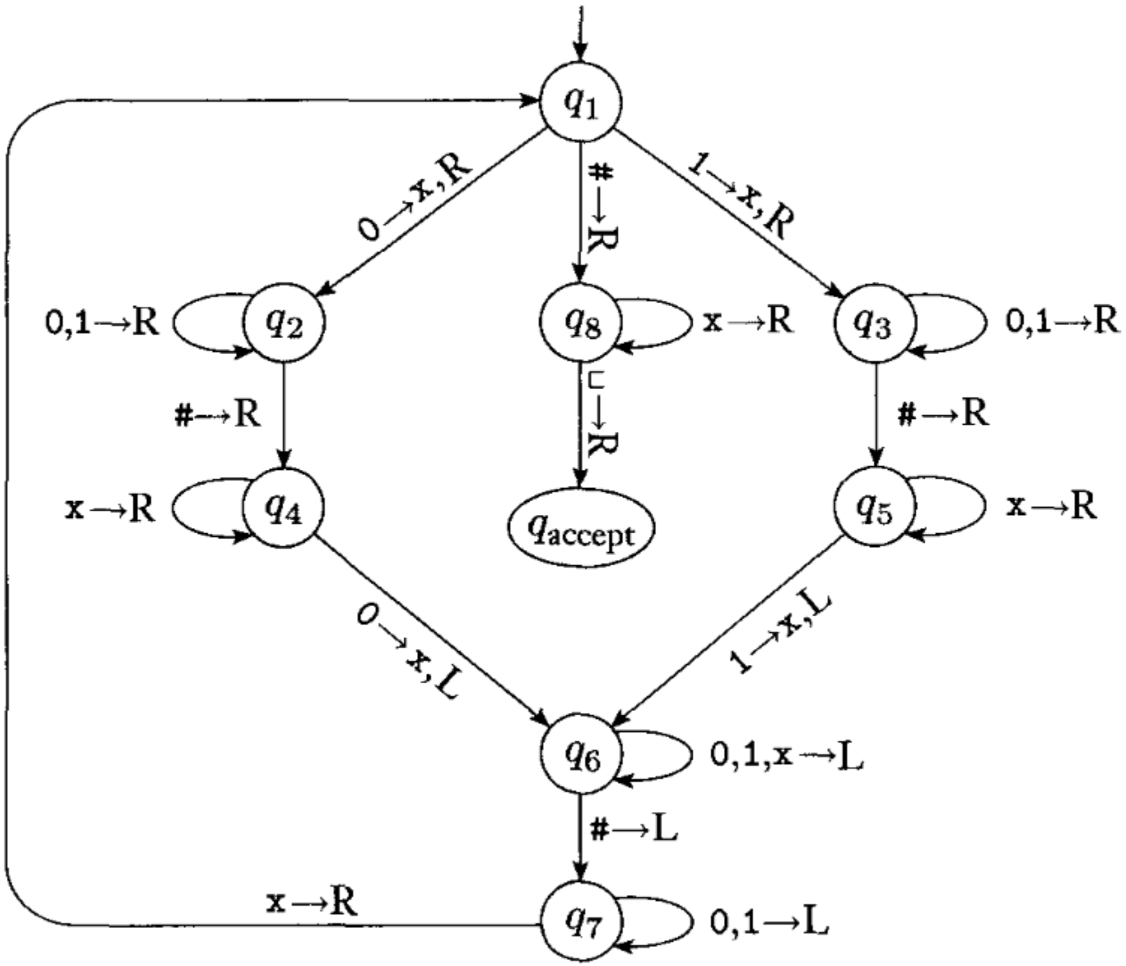
All missing transitions have output (q_{reject}, _, R)

Fig 3.10 in Sipser



Computation on input 0#0?

Configuration **u q v**
for current tape uv (and then all blanks), current head location is first symbol of v, current state q



Computation on input 0# ?

For next time

GroupHW4 due Saturday, May 12

For Friday, pre-class reading:

* Bottom of page 166 and top of page 167 (high-level and implementation level definitions of Turing machines)

* Terminology for describing Turing machines pages 184-185