pp. 165-175. **Turing Machines** (Sec. 3.1)

- (p. 166) Difference from DFA and PDA
  - 1-sided infinite **Tape** instead of (infinite) stack
    - One symbol fits in a cell
    - Initially input string starts on left edge and extends right
      - 1st **blank □** to right of tape marks end of input string
    - Tape cells to right of 1st □ go on forever with more □s
    - Any tape cell can be modified
  - **Tape head** initially on leftmost symbol on tape
    - Can move head left or right one cell
  - **Accept** and **reject** signaled by entering designated states
  - (p. 167) Sample TM for {w#w|w in {0,1}*} (non-CFL)
- Formal Definition: $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject}\}$
  - Q = set of states
  - ∑ = **input alphabet**, not including □
    - Characters that make up tape at start
  - Γ = **tape alphabet**, symbols that can be on tape cell
    - □ in Γ, ∑ subset of Γ
    - Characters that can be written to tape
  - δ: QxΓ -> QxΓx{L,R}
    - Where L & R signal which direction to move tape
  - q_0 = start state; q_{accept} is accept state; q_{reject} is reject state

- **Computation**:
  - Input string w = $w_1, w_2, \ldots w_n$ on left of tape, followed by □s
  - Tape head starts at leftmost cell (i.e. where w_1 is)
  - Computation step
    - Reads cell under head
    - Combine with current state to determine which transition rule applys (note no εs!)
    - Set state to new value from transition rule
    - Write symbol from rule to cell
    - Move tape head either left or right as specified
      - Cannot move beyond leftmost cell
  - Repeat until accept or reject
    - Possible for machine to loop forever
- **Configuration**:
  - Current state, tape contents, head location
  - Written as u q v
    - q is current state
    - Current tape holds string uv
    - Tape head is over *leftmost symbol in string v*
  - Start configuration: q_0 w (u is empty string)
  - (p.169) Fig. 3.4 Example configuration
    - TM that accepts in in Fig. 3.10 p. 173 (discussed later)

- (p. 169) Configuration C1 **yields** C2 if M can legally go from C1 to C2 in 1 step
  - if $\delta(q_i, b) = (q_j, c, L)$ then ua q_i bv yields u q_j acv
    - If tape head at left end (ua = ε), then q_i bv yields q_j cv
  - $\delta(q_i, b) = (q_j, c, R)$ then ua q_i bv yields uac q_j v
    - If tape head at current rightmost end (b = □),
    - then ua q_i □ yields uac q_j □
      - Note former blank now occupied
  - **Accepting configuration** u q_{accept} v
  - **Rejecting configuration** u q_{reject} v
  - Accepting and Rejecting configurations called **halting configurations** because no further configurations possible
- (p.170) M **accepts** w if
  - A sequence C1, C2, … Ck exists
  - C1 = start configuration q_0 w
  - Each C_i yields C_{i+1}
  - C_k is accepting configuration: u q_{accept} v
    - Strings u and v are arbitrary

- (p. 170) TMs and Languages
  - L(M) = set of strings accepted by TM M
  - L is **Turing-recognizable** if some TM M accepts it
  - When M started, 3 outcomes: Accept, Reject, Loops
    - M can fail to accept if it enters q_{reject} or loops
  - (p. 170) M is a **decider** is it never loops
    - I.E. always stops, regardless of input string
    - I.e. always ends up in either q_{accept} or q_{reject}
  - (p. 170) L is **Turing-decidable** (or simply **decidable**) if some Turing Machine decides it.
- Examples
  - (p. 171 Ex. 3.7) A = $(0^k \mid k=2^n, n \geq 0\}$
    - Multiple iterations, each cuts # 0s in half
  - (p.173 Ex. 3.9) B = {w#w | w in {0,1}*}
  - (p. 174 Ex. 3.11) C = $\{a^i b^j c^k \mid ixj=k, i,j,k \geq 1\}$
  - (p.175 Ex. 3.12} E = $\{\#x_1\#x_2\# \ldots\#x_l \mid$ no two x's are equal}