Turing Machine

Language of TM

- Yield: configuration C_1 yields C_2 if
  the TM can go from C_1 to C_2 in one step.
  e.g. If $\delta(q, a) = (r, b, L)$, then $ub\,q\,av \vdash u\,r\,bav$
- A TM M accepts a string w iff
  M goes from the start configuration to an accepting configuration on input w, i.e.,
  $\exists \alpha, \beta \in \Gamma^*$, such that $q_0 w \ \vdash^* \ \alpha\, q_{accept}\, \beta$
- The language of M:
  $L(M) = \{\ w \mid w \in \Sigma^*$ and M accepts $w\ \}$

Turing-recognizable Languages

- Definition: A language is Turing-recognizable (recursively enumerable, r.e.) if some Turing machine recognizes it, i.e.,
  $\{$ L : $\exists$ TM M, such that L = L(M) $\}$
- A TM has three possible outcomes: *accept, reject, or not halt*
- If L is r.e., we can determine if $w \in L$,
  but not necessarily if $w \notin L$.
  Thus, we don't have an algorithm for determining whether or not $w \in L$.

Turing-decidable Languages

- Decider: TMs that halt on all inputs.
- Decide: A TM M decides a language L if M recognizes L and M is a decider.
- Definition: A language is Turing-decidable (recursive) if some TM decides it, i.e.,

  $\{$ L : $\exists$ TM M, such that L = L(M) and M halts on all inputs $\}$
- If L is recursive, then $\exists$ an algorithm (TM) for determining whether or not $w \in L$.
- Example:
- CFL $\subset$ recursive $\subset$ R.E.

Computing with TMs

- Advantage: simplicity and uniformity
  - one type of instruction
  - time complexity: $\#$ of steps
  - space complexity: total number of tape squares used
- Disadvantage: not efficient
- Picture notation for TMs
- Examples:
  - decide $\{w\#w \mid w \in \{0,1\}^* \}$
  - add two unary numbers

Variants of TMs

- Multi-tape TMs
- Nondeterministic TMs
- Enumerators
- All these models are equivalent in power.