Recap: Recognizable versus Decidable Languages

- A language L is called Turing-Recognizable if there exists a TM M such that L(M) = L
  - Note: M need not halt on all inputs but it should halt and accept all and only those strings that are in L; it can reject strings by either going to q_{rej} or by looping forever
- A TM is a decider if it halts on all inputs
- A language L is decidable if there exists a *decider* D such that L(D) = L

Closure Properties of Decidable Languages

- Decidable languages are closed under $\cup$, $\circ$, *, $\cap$, and complement
- Example: Closure under $\cup$
- Need to show that union of 2 decidable L's is also decidable

  Let M1 be a decider for L1 and M2 a decider for L2

  A decider M for L1 $\cup$ L2:

  On input w:

  1. Simulate M1 on w. If M1 accepts, then ACCEPT w. Otherwise, go to step 2 (because M1 has halted and rejected w)
  2. Simulate M2 on w. If M2 accepts, ACCEPT w else REJECT w.

  M accepts w iff M1 accepts w OR M2 accepts w

  i.e. L(M) = L1 $\cup$ L2

Closure Properties

- Consider the proof for closure under ∪

  A decider M for L1 ∪ L2:

  On input w:

  1. Simulate M1 on w. If M1 accepts, then ACCEPT w. Otherwise, go to step 2 (because M1 has halted and rejected w)
  2. Simulate M2 on w. If M2 accepts, ACCEPT w else REJECT w.

  M accepts w iff M1 accepts w OR M2 accepts w

  i.e. L(M) = L1 ∪ L2

Will this proof work for showing Turing-recognizable languages are closed under ∪? Why/Why not?

Closure for Recognizable Languages

- Turing-Recognizable languages are closed under ∪, °, *, and ∩ (but not complement! We will see this later)
- Example: Closure under ∩

  Let M1 be a TM for L1 and M2 a TM for L2 (both may loop)

  A TM M for L1 ∩ L2:

  On input w:

  1. Simulate M1 on w. If M1 halts and accepts w, go to step 2. If M1 halts and rejects w, then REJECT w. (If M1 loops, then M will also loop and thus reject w)
  2. Simulate M2 on w. If M2 halts and accepts, ACCEPT w. If M2 halts and rejects, then REJECT w. (If M2 loops, then M will also loop and thus reject w)

  M accepts w iff M1 accepts w AND M2 accepts w i.e. L(M) = L1∩L2

Suppose you want a decider TM for deciding whether a DFA D accepts an input string w

How do we encode a given DFA D as input to a TM?

How does the TM decide if D accepts a given w?

(On-board solution: binary encoding of DFA/CFG/TM and three-tape decider TM)