MA/CSSE 474 Day 39 Summary

1) L_1 is *mapping reducible* to L_2 ($L_1 \leq_M L_2$) iff there exists some computable function *f* such that: $\forall x \in \Sigma^* (x \in L_1 \leftrightarrow f(x) \in L_2)$.
   a) To decide whether *x* is in L_1, we transform it using *f*, and ask whether $f(x)$ is in L_2.
   b) All of the "undecidability" reductions we have done so far are mapping reductions.
   c) Sometimes a mapping reduction is not sufficient.
      i) We need to use *Oracle* as a subroutine and then do other computations after *Oracle* returns.
   d) Example: {<M> : M accepts no even length strings}
      i) Do the "standard reduction" from H
         (1) R(<M,w>) → <M#>, where M# erases tape, writes w, runs M on w, accepts.
         (2) Oracle(R(<M,w>)) is "backwards"; <M,w>∈H ⇒ Oracle rejects <M#>; <M,w>∉H ⇒ Oracle accepts.
         (3) So we use ¬Oracle(R(<M,w>))

2) **Rice's Theorem:** Let P be any non-trivial property of the SD languages (true for some languages, false for others). Then {<*M*>: *P*(*L*(*M*)) = *True*} is not in D.
   a) This applies to many languages we have examined, but you should also know how to use reduction.
   b) {<M> : L(M) is regular} is not in D. Can show this by Rice's Theorem, or directly (reduction form H).
   c) Ditto {<M> : L(M) is context-free}

3) **{<M> : M accepts no even length strings}** Details on slides.
   a) Notes on reduction from H

   b) Notes on reduction from H with *not*

4) **Example:** L_2 = {<M, q> : M reaches state q on some input}
   a) Reduction diagram: Reduce H_{ANY} to L_2.

   b) R takes a TM description <M> as input and returns <M#, h>, where M# is defined by:

   c) C is correct: M# reaches it halting state h iff

i) If <M> ∈ H_{ANY} , then

ii) If <M> ∉ H_{ANY} , then

5) There is a computable function, *obtainSelf*. When any TM M calls it as a subroutine, it writes <M> on M's tape.
   a) This is a result of the Recursion Theorem, which is in Chapter 25.
   b) Quines are related to obtainSelf.
6) Details matter:
   a) L_1 = {<*M*>: *M* has an even number of states}.
   b) L_2 = {<*M*>: |<*M*>| is even}.
   c) L_3 = {<*M*>: |*L*(*M*)| is even}.
   d) L_4 = {<*M*>: *M* accepts all even length strings}.
7) **Non-SD languages.** Usually involve "double infinity"
   a) ¬H = {<*M*, *w*> : TM *M* does *not* halt on *w*}.
   b) {<*M*> : *L*(*M*) = Σ*}.
   c) {<*M*> : TM *M* halts on nothing}.
   d) Different ways to show non-SD:
      i) Contradiction
      ii) *L* is the complement of an SD/D Language.
      iii) Reduction from a known non-SD language.
8) If ¬*L* is in SD, and at least one of *L* or ¬*L* is not in D, then L is not in SD.
   a) Example that we have seen before: ¬H is in not in SD, since ¬(¬H) = H is in SD and not in D)
9) **Example:** A_{anbn} = {<*M*> : *L*(*M*) = A^nB^n}
   a) Rice's Theorem says "not in D". We want to show it's also not in SD. **Reduce ¬H to A_{anbn}**
   b) **Reduction 1:**

R(<*M*, *w*>) =
1. Construct the description <*M#*>, where *M#*(*x*) operates as follows:
   1.1. Erase the tape.
   1.2. Write *w* on the tape.
   1.3. Run *M* on *w*.
   1.4. Accept.
2. Return <*M#*>.

Reduction 2:

R(<*M*, *w*>) =
1. Construct the description <*M#*>, where *M#*(*x*) operates as follows:
   1.1 Copy the input *x* to another track for later.
   1.2. Erase the tape.
   1.3. Write *w* on the tape.
   1.4. Run *M* on *w*.
   1.5. Put *x* back on the tape.
   1.6. If *x* ∈ A^nB^n then accept, else loop.
2. Return <*M#*>.

Reduction 3:

R(<*M*, *w*>) reduces ¬H to A_{anbn}:
1. Construct the description <*M#*>:
   1.1. If *x* ∈ A^nB^n then accept. Else:
   1.2. Erase the tape.
   1.3. Write *w* on the tape.
   1.4. Run *M* on *w*.
   1.5. Accept.
2. Return <*M#*>.

10) $\mathbf{H_{ALL}}$ **= {<*M*> : TM halts on Σ^*}**