MA/CSSE 474 Day 08 Summary

Recall

1. $x \approx_L y$ iff $\forall z \in \Sigma^*$ ($xz \in L$ iff $yz \in L$)

Main ideas from today:

1. If L = {aa}{b}*{a}, what are the equivalence classes of $\approx_L$? (Do this one for practice later)
   Σ = {a, b}.L = {$w \in \Sigma^*$: $|w|$ is even} (Do this one for practice later)

2. L={$w \in$ {a, b} *: no two adjacent chars are the same} shows: multiple equivalence classes may be subsets of L. (equivalence classes are on the slides)

3. If $L = A^nB^n = \{a^nb^n: n \geq 0\}$, what are the equivalence classes of $\approx_L$?

   Things that we will state today and probably prove on Monday (4-11 below):

4. L regular →, # equivalence classes of $\approx_L$ is a lower bound on # states in any DFSM M such that L = L(M).
5. ***Theorem*:** Let L be a regular language over some alphabetΣ. Then there is a DFSM M that accepts L and that has precisely n states where n is the number of equivalence classes of $\approx_L$. Any other FSM that accepts L must either have more states than M or it must be equivalent to M except for state names.
6. **Construction:** $M = (K, \Sigma, \delta, s, A)$, where:
   - K contains n states, one for each equivalence class of $\approx_L$.
   - $s = [\varepsilon]$, the equivalence classε under $\approx_L$ that contains ε.
   - $A = \{[x] : x \in L\}$.
   - $\delta([x], a) = [xa]$. In other words, if M is in the state (equiv. class) that contains some string x, then, after reading the next symbol, a, it will be in the state that contains xa.
7. Three things to show (but we won't show them today):
   a. K is finite.
   b. **δ is a well-defined function.** i.e., δ is defined for all *(state, input)* pairs and produces, for each pair, a unique value.
   c. ***L* = *L*(*M*).** To prove this, we must first show that $\forall s,t \in \Sigma^*(([\varepsilon], st) \vdash_M^* ([s], t))$.
      We do this by induction on $|s|$. **The base case is trivial**.
8. There exists no smaller machine *M#* that also accepts L.
9. There is no different machine *M#* that also has n states and that also accepts L.

   Example:
   $L = \{w \in \Sigma^*$: no two adjacent characters are the same}

10. **Myhill-Nerode Theorem:** A language is regular iff the number of equivalence classes of $\approx_L$ is finite.
11. **Myhill-Nerode Theorem:** A language is regular iff the number of equivalence classes of $\approx_L$ is finite.
12. ***Two approaches to minimizing a given DFSM:***
    a. ***Start with separate states and merge.***

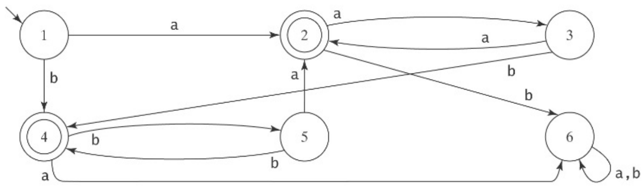
    b. ***Start with overclustering, and then separate distinguishable states. Successively approximate.***

13. Define ≡ (a relationship on states of a DFSM M) by p≡q iff for every string w ∈ Σ*, either w takes M to an accepting state from both p and q, or w takes M to a rejecting state from both p and q.

14. We construct ≡ as the limit of a sequence of approximating equivalence Relations $\equiv^n$
    a. p $\equiv^0$ q iff configurations (p, ε) and (q, ε) either both lead to accepting states or both lead to non-accepting states. I.e., ____'
    b. For all k >= 0, p $\equiv^{k+1}$ q iff both of the following are true:
    p $\equiv^k$ q
    For all a ∈ Σ, $\delta(p, a) \equiv^k \delta(q, a)$

Can you express the meaning of $\equiv^n$ non-recursively in concise English?

15. Example:



16. We can put any DFSM into a *canonical form* that makes it easy for us to tell whether two machines are "the same". **Canonical form.** Given a connected DFSM, we can systematically number the states in such a way that any other equivalent machine will have the same numbering and thus be identical. If we apply this to a minimal-state DFSM for language L, we see that there is a unique canonical minimal machine for each regular language.