MA/CSSE 474 Day 18 Announcements and Summary Solutions of in-class exercises

Announcements:

1) **Exam2 material:** 5.7-5.8, 6.1-6.4, 8.1-8.6, 9.1-9.2, 10, HW 5-9. Closed book and notes.

Main ideas from today

1. What is a decision procedure?

2. Given a DFSM M=(K,Σ, δ, s, A) and a string w ∈ Σ*, is w ∈ L(M)?

   I wrote this solution on the board in class. Also see Theorem 9.1 (P 188 in our textbook)

3. (I corrected the numbering)Given two regular expressions α_1 and α_2, is $(L(\alpha_1) \cap L(\alpha_2)) - \{\varepsilon\} \neq \varnothing$?
   (don't try to copy what's on the slide; use this space to make notes about those details).

   This one is in the slides.

   Also see page 194

4. With two other students, write (efficient?) decision procedures for as many of these problems as you have time for:
   a. Given a regular expression α and a string w, is w in L(α)?
      Construct a FSM that accepts L(α). Then use the algorithm from #2 above. (done in class)

      Also see p188

   b. Given a FSM M, is L(M) empty?
      State-based approach: Construct an equivalent minimal DFSM. The language is empty iff the minimal machine has no accepting states.

      String-based approach: Construct an equivalent DFSM, and let k be the number of states of that DFSM. If M accepts any strings, it has to accept at least one string whose length is <k. (If it accepts a string whose length is ≥ k, there has to be a cycle whose length is ≤ k. Skip the cycle and we get an accepted string whose length is shorter). SO we try all strings whose length is less than k. If M accepts any of them, the language is not empty. Otherwise it is empty.

      Also see pages 189-190

   c. Given a FSM M, does $L(M) = \Sigma_M^*$?
      Construct the DFSM that accepts the complement of L(M). Use the algorithm above to see whther the complement language is empty.

      Also see pages 190-191

d. Given a FSM M, is L(M) finite? [Try two different approaches]

   i. White box: Using full knowledge of the machine's transition graph.

   Deterministic L(M) is infinite iff its transition graph has a cycle, and we can get form one of those cycle states to an accepting state.

   1. Find all of the states that lead to accepting states.

      let LtoA = A

      repeat

        for each state q not in LtoA

          if there is an input symbol a such that δ(q, a) is in LtoA, add q to LtoA.

   2. Use depth-first search to determine whether there are any cycles in the graph. If not, the language is finite. If so, see if any states in a cycle are in LtoA. If so, language is infinite; if not, finite.

   Also see page 191

   ii. Gray box: Using only the number of states of M and a procedure for deciding whether M accepts a particular string w.

   Deterministic L(M) is infinite iff k-state M accepts some string whose length is at least k but less than 2k. Feed all of these strings to M and see whether it accepts any of them.

   Also see pages 191-192

e. Given a FSM M, is L(M) infinite?

   Run the algorithm for "is it finite?" and negate the answer.

f. Given two FSMs M1 and M2, are they equivalent? (I.e., do they accept the same language)

   For each machine, produce an equivalent DFSM, minimize it, and put the minimal machine in canonical form. Are the canonical form machi8nes equivalent?

   Also see pages 192-193

g. Given a DFSM M, is M minimal?

   Run the minimization algorithm. Does the minimized machine have the same number of states as M?

   Also see page 193