474 Instructor Notes from Day 17 slides:

Slide 7: Using the Closure Properties

Answer: a*b*

Slide 10: ***L* = {a^ib^j: *i, j* ≥ 0 and *i* ≠ *j*}**

*If ¬L were regular, then the intersection of ¬L and a*b* would be regular. In aprevious slide, we showed that it is not regular.*

Slide 21: **Defining Functions from one Language to Another**

Let w be any string from a*db*. |w| is even, it contributes nothing to chop(L).
If |w| odd, and #a's = #b's. then w contribtes a^nb^n to chop(L), for some L.
otherwise, |w| is odd, and d is not in the middle, so chop removes an a or b.
Since |W| is odd difference between #a and #b must be at least 2, so chopped string also has different number of a's and b's.
Chop(a*db*) contains all strings from A^nB^n, plus some strings in {a*bb* whose length is even}.
Can it be regular? If so, its intersection with a*b* would be regular. But that intersection is A^nB^n

Slide 29: Totality

Construct *M´* to accept ¬*L*(*M*).

2. Return *emptyFSM*(*M´*).

Slide 30: Finiteness:

The mere presence of a loop does not guarantee that *L*(*M*) is infinite. The loop might be:

- labeled only with ε,
- unreachable from the start state, or
- not on a path to an accepting state.

1. *M´* = *ndfsmtodfsm*(*M*).
2. *M´´* = *minDFSM*(*M´*).
3. Mark all states in *M´´* that are on a path to an accepting state.
4. Considering only marked states, determine whether there are any cycles in *M´´*.
5. If there are cycles, return *True*. Else return *False*.

The simulation approach:

M´ = *ndfsmtodfsm*(*M*).

2. For each string *w* in Σ* such that ________________ do: **[answer: $|K_M{'}| \le w \le 2 \cdot |K_M{'}| - 1$]**
   Run *decideFSM*(*M´*, *w*).
3. If *M´* accepts at least one such string, return *False*.
   Else return *True*.

Slide 34: Minimality

M´ = *minDFSM*(*M*).

2. If $|K_M| = |K_{M'}|$ return *True*; else return *False*.