15-212
Spring 1998
Midterm Exam 1: Solution set
10 Feb 1998
Adam Berger

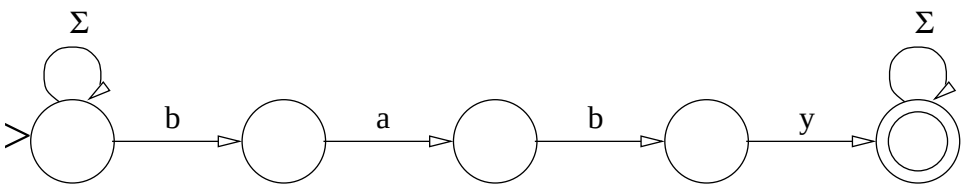
1. (*10 points*) List the following machines or languages in increasing order of power (*i.e.*, ability to represent languages that others cannot). (List equivalent ones on the same line):

   - Context-free grammars (CFG)
   - Turing machines (TM)
   - Deterministic finite state automata (DFSA)
   - Regular expressions (RE)
   - Deterministic push down automata (DPDA)
   - Non-deterministic push down automata (NPDA)
   - Non-deterministic finite state automata (NFSA)

   $DFSA{=}NFSA{=}RE$
   $DPDA$
   $NPDA{=}CFG$
   TM

   Some of you thought a DPDA and NPDA were equally powerful. But remember that a DPDA actually cannot represent all context-free grammars. The example to keep in mind is $L = ww^R$. *How would a DPDA know when* w^R *begins?*

2. (*10 points*) Briefly describe what is accepted by this FSA, and write the language accepted by it as a regular expression:



   Any string containing the substring "baby". That is, $\Sigma^* baby \Sigma^*$.

3. (*10 points*) If a DFSM F accepts a (regular) language L, then we can construct another one to accept L's complement by interchanging the accepting and non-accepting states of F. Explain why this construction fails if F were non-deterministic.

   Say we interchange the accepting and non-accepting states of F *to create* $\overline{F}$. *A string* w *should be accepted by* $\overline{F}$ *only if* all *the clones of* F *wind up in a non-accepting state after reading* w. *But by the*

given construction, w will be accepted by $\overline{F}$ if even a single clone of F winds up in a non-accepting state. In short, $\overline{F}$ accepts too much.

There were other acceptable solutions as well: many of you used the fact that epsilon transitions can cause trouble, for instance.

4. (*15 points*) If L is context-free, is $L' = \{w : w^R \text{ in } L\}$ also context-free? (w^R is the reverse of w.) Give an informal argument justifying your answer.

4

Yes. Given a context-free language L, there exists a CFG G which generates strings in the language. The "reverse" language L' is context-free if we can demonstrate a CFG G' which generates strings in it. We can generate G' from G simply by reversing each term on the righthand side of all the productions in G. So, for example, the grammar

$S \rightarrow SS \mid e \mid (S)$

becomes

$S \rightarrow SS \mid e \mid)S($

The parse trees generated by the new grammar will be "mirror images" of the trees generated by the old grammar.

5. (*15 points*)

We know that the language of balanced parenthesis strings (), (())(), (()()), etc. is context-free but not regular. If we limit the nesting of parentheses to some constant K (i.e., at any point in a string there are at most K left parentheses pending to be closed), do we then get a regular language? Why or why not?

Yes: this language is indeed regular. If the nesting level is never more than K, then we can create a DFA with $K+2$ states which accepts the language. The machine behaves as follows. It starts in state s_0. From state s_i, reading a (drives the machine to state s_{i+1}; reading a) drives the machine to state s_{i-1}. The states s_{-1} and s_{K+1} are non-accepting "black hole" states: once the machine enters them, it can't leave.

(For what it's worth, by combining s_{-1} and s_{K+1} we could save a state.)

6. (*15 points*) How many languages are there over a finite alphabet? How many Turing machines are there? What does this imply?

There are uncountably many languages over a finite alphabet. Why is this? Assume there are only countably many languages. Then we could write them down in a chart, as follows:

Language	*strings in $\Sigma^\star$*									
1	*yes*	*no*	*no*	*no*	*yes*	*no*	*yes*	*no*	*no*	*yes*
2	*no*	*no*	*yes*	*no*	*yes*	*no*	*no*	*yes*	*no*	*yes*
3	*yes*	*yes*	*no*	*yes*	*yes*	*yes*	*yes*	*no*	*yes*	*no*
4	*yes*	*yes*	*yes*	*no*	*yes*	*yes*	*yes*	*yes*	*no*	*yes*
$\vdots$					$\vdots$					

In this chart, each row contains the behavior of a particular language on all strings. So, for example, the second language does not accept the first and second strings, but it does accept the third.

Now, the complement of the diagonal, which starts "no, yes, yes, yes", is certainly a legitimate language. However, it cannot possibly be a row in the above chart: it differs from row i in the i'th position, for every i. So the assumption that we could actually write down all the languages, one per row, must have been false—i.e., there are uncountably many languages.

As for Turing machines, there are only countably many. Think of it this way: each Turing machine could be written down like a (Java) program, as a sequence of bytes. This sequence of bytes is just a long number; that is, there's at most one Java program for each number. Thus they are countable.

Since there are more languages than Turing machines, there must be some language that is not computable (decidable) by a Turing machine.

7. (*15 points*) Write a CFG over $\Sigma = \{a, b\}$ for $L = \{xayb : x, y \text{ in } \Sigma^*, \text{ and } |x| = |y|\}$. ($|x|$ is the length of string x.)

$$\begin{array}{lcl} S & \rightarrow & Tb \\ T & \rightarrow & aTa \mid aTb \mid bTa \mid bTb \mid a \end{array}$$

8. (*10 points*) Show that $L = \{wxw^R : w \text{ and } x \text{ are strings in } \Sigma^*\}$ is regular. (w^R is the reverse of w.)

$L = \Sigma^\star$, *which is regular. Do you see why?*