Name: DAN PORTS

Theory of Computation
18.404/6.840
Tuesday, October 26, 2004

Michael Sipser

80 minute length

MidTerm Examination

INSTRUCTIONS: Answer all questions as clearly as you can. Be concise. If you get stuck, $\Longrightarrow$ **state what you are trying to do in order to have a chance at partial credit.**

You may use (without proof) any theorem that we have stated in lecture or recitation, or which has been assigned in a problem set, or which is proven in the text. You may not use the statements given in problems that appear without solutions in the text but that weren't assigned or covered in lecture, unless you prove them.

Open Book/Notes/Handouts.
Other books, bibles, or materials from outside of class not permitted.

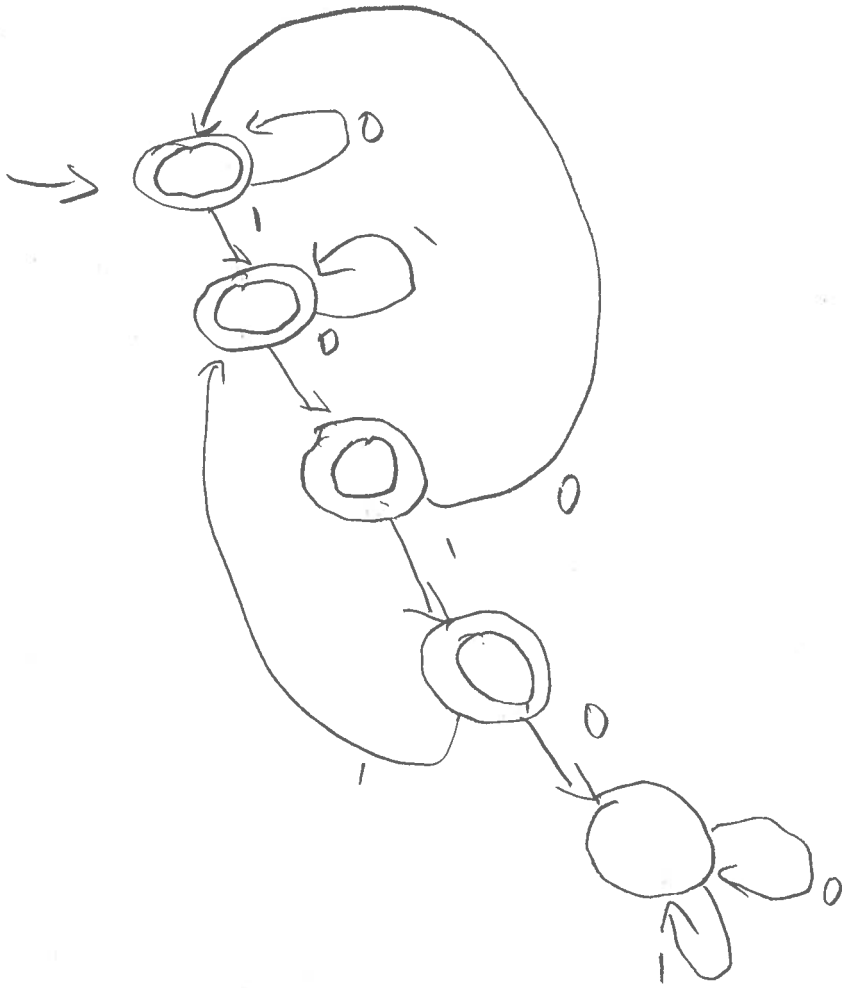
Problem	Possible	Actual
1	20	20
2	25	23
3	30	19
4	25	25
Total	100	87

1. (20 points)

(a) Is the class of languages that are recognized by <u>non</u>deterministic finite automata closed under complement? Explain why or why not.

Yes. Let L be a language recognized by a NFA N. Then N can be converted to an equivalent DFA M. By making all accept states reject states, and vice versa we create a DFA M' that recognizes $\overline{L}$. Regarding M' as a NFA, we have a NFA that recognizes $\overline{L}$.

(b) Let $\Sigma = \{0, 1\}$ and let $A = \{w | \ w$ does not contain 1010 as a substring$\}$. Give the state diagram of a deterministic finite automaton that recognizes A.



2. (25 points) Let $\Sigma = \{0, 1\}$.

(a) Let $A = \{w | \ w = 0^i 1 0^j 1 0^k$ for $i, j, k > 0$ and $i \leq k$ and $j \leq k\}$.
Prove that A is not a CFL.

Suppose A is a CFL. Then it has a pumping length p. Let $S = 0^p 1 0^p 1 0^p$. By the pumping lemma, $S = uvxyz$, with $|vy| > 0$ and $|vxy| \leq p$, and $\forall i \geq 0$ $uv^i xy^i z$. Note first that v and y cannot contain any 1s. If they did, we could pump up, and the resulting string would have more than two 1s, and so not be in A. Note also that v and y may not be in both the first and third sections of zeros. Then there are two cases:

1) v and y are contained entirely in the first two sections. Then we pump up, and the resulting string has either $i > k$ or $j > k$.

2) v or y contain part of the third section. Then we pump down and the resulting string has form (over →)

(b) Let $w \in \Sigma^*$. A *run of 0s* in w is a substring of one or more consecutive 0s that appears in w, where the substring cannot be extended by 0s. In other words, a run of 0s is bordered on both sides by a 1 or the end of w. For example, the string $s = 0011000100000$ has three runs of 0s which have lengths 2, 3, and 5. Let *LONG-AT-LAST* $= \{w | \ w$ contains a 0 and the rightmost run of 0s in w is at least as long as any other run of 0s in $w\}$.
Prove that *LONG-AT-LAST* is not a CFL. You may use the result of part (a).

Suppose that LONG-AT-LAST were a CFL. Note that A from above is the special case of LONG-AT-LAST with 3 runs, i.e. it is the intersection of LONG-AT-LAST with the RE $0^* 1 0^* 1 0^*$. But CFLs are closed under intersection w/ REs, so this means A is a CFL. Contradiction.

(-1)

(c) Prove that the complement of *LONG-AT-LAST* is a CFL.

The following grammar generates LONG-AT-LAST:

LONG-AT-LAST $\rightarrow \Sigma^* 1 T R \mid T R \mid R$

(-1)

$T \rightarrow 0 T 0 \mid 0 \Sigma^* 1$

$R \rightarrow 1R \mid \epsilon$

$\Sigma^* \rightarrow \Sigma^* 0 \mid \Sigma^* 1 \mid \epsilon$

(no ϵ-rules)

$0^p 1 0^i 1 0^k$ for $k < p$.

So the resulting string is not in A. —
Contradiction. A is not a CFL.

3. (30 points) Note: Avoid Rice's theorem. It's not helpful in either of the parts below.

(a) Let $LEX = \{\langle E \rangle | \ E$ is a Turing-enumerator that prints strings in lexicographic order$\}$. Show that LEX is undecidable.

Recall that a language is decidable iff an enumerator enumerates it in lexicographic order

Suppose LEX is decided by some machine R. Then define an enumerator $\mathcal{E}$ = "On any input

1) Obtain own description $\langle \mathcal{E} \rangle$
2) Simulate R on $\langle \mathcal{E} \rangle$ (this always completes)
3) If R accepts, print two strings that are not in order (say, 00 then 0)
   If R rejects, print some set of strings in lexicographic order (say, {0, 00, 000, ...})

Then $\mathcal{E}$ prints a set of strings in lexicographic order only if $\mathcal{E} \notin LEX$, Contradiction. So LEX is undecidable

15

(b) Let $INFINITE_{TM} = \{\langle M \rangle | \ M$ is a TM and $L(M)$ contains infinitely many strings$\}$. Show that $INFINITE_{TM}$ is not Turing-recognizable.

$INFINITE_{TM}$ is unrecognizable. Suppose R recognizes it. Then define

M = "Ignore input string
1) Obtain own description $\langle M \rangle$
2) Run R on $\langle M \rangle$
3) If R accepts, reject; otherwise accept."

4

If $\langle M \rangle \in INFINITE_{TM}$, then R accepts and M rejects every string (its language is empty, i.e. finite). This contradicts the definition of $INFINITE_{TM}$

What if R loops on $\langle M \rangle$? Then this implies $L(M)$ contains finitely many strings, which is true, since M loops on all strings. So no contradiction.

3b) INFINITE_TM is unrecognizable. Suppose it were, and R recognizes it. Then define

M = "On any input,

4. (25 points) Define the following two languages:

CH-CHECK$=\{\langle M, w, h\rangle \mid h$ is an accepting computation history for TM M on input $w\}$.

CH-SEARCH$=\{\langle M, w\rangle \mid$ there exists an accepting computation history for TM M on input $w\}$.

In each of the following three parts, *give a brief reason for your answer.* Detailed proofs are not necessary. You may assume that the TMs in the above languages are the single-tape, deterministic variety.

(a) Is *CH-CHECK* decidable?

Yes. This is the problem of verifying an accepting computation history for M, w. We can do this by generating a LBA that verifies computation histories for M, w (as in the proof of thm 5.9) and simulate it on h. We can perform the simulation because A_{LBA} is decidable.

(b) Is *CH-SEARCH* decidable?

No. If there is an accepting computation history, then M accepts w. So if we could solve CH-SEARCH, we could solve A_{TM}.

(c) Is *CH-CHECK* mapping reducible to *CH-SEARCH*? Explain why or why not.

Yes.

CH-CHECK is decidable, so it is recognizable. Any recognizable language is mapping reducible to A_{TM} (from PS3). So CH-CHECK $\leq_M A_{TM}$. CH-SEARCH is equivalent to A_{TM}, so

CH-CHECK $\leq_M$ CH-SEARCH

BLANK PAGE