CSE596 **Final Exam** **Dec. 14, 2012**

Open book, open notes, closed neighbors, 170 minutes. The exam has seven problems and totals 240 pts., subdivided as shown. *Show your work*—this may help for partial credit. *Please write in the exam books only* (if you enclose separate scratch sheets, they may—or may not—be ignored). You may cite any major relevant theorem or fact or definition covered in the course without needing to give a justification (unless one is specifically asked for).

Notation: The names of complexity classes and languages are either completely standard or explained in the questions themselves; for future reference let's still clarify $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ and $\mathsf{EXP} = \mathsf{DTIME}[2^{n^{O(1)}}]$. The notation M_e refers to a fixed acceptable recursive enumeration of RE by non-oracle deterministic Turing machines.

Especially for Problem (1), we make some special notations. Suppose $\Sigma' = \Sigma \cup \{\,\#\,\}$ where '#' is a divider character not in Σ. Then given any language $L \subseteq \Sigma'^*$ define

$$\#\mathit{Prefs}(L) = \{\, x \in \Sigma'^* : (\exists y \in \Sigma'^*) x\#y \in L \,\}.$$

That is, we care about prefixes only up to dividers. With this in mind, we code languages of pairs or tuples using divider signs, for instance

$$\begin{aligned} \mathit{CVP} &= \{\, C\#x : C(x) = 1 \,\} \\ \mathsf{KleeneT} &= \{\, e\#x\#c : T(e,x,c) \,\}. \end{aligned}$$

Here C stands for a Boolean circuit (encoded over Σ without using '#') and T for the Kleene T-predicate; by convention C has n input gates where n is the length of x. Also as-usual G stands for a graph, undirected unless otherwise specified. Other unspecified notions take their definitions in the textbook, for instance what it means for a graph to be k-colored.

Time Guideline: Up to 45 minutes for Problem (1), then 15 minutes each on (2), (3), and (4), 30 on (5), 20 on (6), and 15–30 minutes on the essay (7).

(1) (60 pts.)

Classify each of the following languages $L_1, \ldots, L_{12}$ according to the classification on the next page. Please write your answers in this form: if L_{13} were the language of the Halting Problem, you would write "13. h" or "13. (h)", perhaps adding the words "c.e. but undecidable" to guard against silly errors. *No justifications are needed*, but may help for partial credit. There is a unique best answer for each language, and some answer(s) may be unused. In L_7 and L_8, the Boolean circuits have some number n of input gates, and just one output gate. *The class divisions and languages are overleaf.*

(a) regular

(b) in logspace but not regular.

(c) in NL but not known/believed to be in logspace.

(d) in P and not known to be—or believed not to be—in NL.

(e) in NP but not known/believed to be in co-NP.

(f) in PSPACE but not known/believed to be in NP.

(g) decidable but known to be not in PSPACE;

(h) c.e. but not decidable;

(i) co-c.e. but not c.e.

(j) in Π_2 but neither c.e. nor co-c.e.

1. $L_1 = \bigcup_{m=1}^{\infty} (a+b+c)^{2m}$.
2. $L_2 = \bigcup_{m=1}^{\infty} (a+b)^m (b+c)^m$.
3. $L_3 = \{ \lambda \}$.
4. $L_4 = \{ e : L(M_e) = \{\lambda\} \}$.
5. $L_5 = \{ G : G$ can be 2-colored $\}$.
6. $L_6 = \{ G : G$ can be 4-colored $\}$.
7. $L_7 = \{ C :$ every x makes $C(x) = 0 \}$.
8. $L_8 = \{ C :$ every m makes $C(0^m 1^{n-m}) = 0 \}$.
9. $L_9 =$ a diagonal language with respect to (some recursive enumeration of) EXP.
10. $L_{10} = \{ d\#e : L(M_d) \cap L(M_e) = \emptyset \}$.
11. $L_{11} = \#\mathit{Prefs}(CVP)$.
12. $L_{12} = \#\mathit{Prefs}(\text{KleeneT})$.

(2) 30 pts. total

For each statement about complexity classes, all of whose truth is currently unknown, write as many unknown consequences as you can (without being redundant). Ten correct consequences total bring full credit. An incorrect consequence deducts 3, while having two redundant equalities or two redundant inequalities deducts 1—but even with deductions, you can still reach 30 with more good ones, and there is no penalty for having more good ones than you need.

For example, suppose the statement is $\mathsf{P} = \mathsf{NP}$. Then valid consequences include $\mathsf{PH} = \mathsf{P}$, $\mathsf{E} = \mathsf{NE}$, and $\mathsf{NP} \neq \mathsf{EXP}$. Although the last is implied by $\mathsf{PH} = \mathsf{P}$, it is not considered redundant because $\mathsf{NP} \neq \mathsf{EXP}$ is an inequality and $\mathsf{PH} = \mathsf{P}$ is an equality. But writing also $\mathsf{co\text{-}NP} = \mathsf{P}$ would be redundant with $\mathsf{PH} = \mathsf{P}$, since both are equalities and one follows from the other—in fact, they are equivalent. (It is also OK to list the same consequence for different questions below.)

(a) $\mathsf{NP} = \mathsf{co\text{-}NP}$.

(b) $\mathsf{Factoring} \notin \mathsf{P}$.

(c) $\mathsf{DLBA} \subset \mathsf{NP}$.

(d) $\mathsf{L} = \mathsf{P}$. (The original had $\mathsf{NL} = \mathsf{P}$.)

(e) $\mathsf{NTIME}[O(n)] \subseteq \mathsf{DTIME}[O(n^2)]$.

(3) $6 \times 5 = 30$ pts. True/false *with justifications*: Please write out the words `true` and `false` *in full*, for 3 points, and for the remaining 2 points, write a relevant justification (need not be a full proof, and should be brief).

(a) If f many-one reduces a language A to a language B, and $C \subseteq A$ and $B \subseteq D$, then f many-one reduces C to D as well.

(b) On current knowledge, it is unknown whether $\mathsf{P} \neq \mathsf{NL}$ or $\mathsf{P} \neq \mathsf{PSPACE}$, but it is currently known that at least one of those two inequalities must hold.

(c) The class $\prod_2$ of the arithmetical hierarchy has non-c.e. languages that are complete for it under polynomial-time many-one reductions.

(d) If R is a routine that runs in polynomial time given some size-n graph as input, and you put R inside a for-loop that loops from 1 to n and exits, the new routine R' runs in polynomial time.

(e) If NP is closed downward under polynomial time Turing reductions then $\mathsf{NP} = \mathsf{P}$.

(f) The language $\{\, M\#x : M$ is a deterministic Turing machine and M accepts $x \,\}$ is complete for the class of r.e. languages under *polynomial time* many-one reductions.

(4) (24 pts.)

Let $L \subseteq \{0,1\}^*$ be a language that contains exactly one string of each length, and such that whenever $x \in L$, exactly one of the strings $x0$ and $x1$ belongs to L. Do ONE of the following:

(a) Prove that if L is r.e., then L is decidable; OR

(b) Prove that if $L \in \mathsf{NP}$, then $L \in \mathsf{NP} \cap \mathsf{co\text{-}NP}$.

(5) (36 pts. total)

Prove that the following decision problem is NP-complete.

Two-Coloring With Faults
Instance: An undirected graph G, an integer $k \geq 1$.
Question: Can G be colored with 2 colors so that at most k edges have both nodes of the same color?

Note: You may use any variant of (3)SAT that was covered in the course, including the "2-3" hybrid used in the Cook-Levin proof given in lecture, or forms such as "not-all-equal 3SAT." Or you may use a problem proved NP-complete on homeworks. But you are not allowed to use "any" problem from an Internet or otherwise unofficial source—and it is strongly recommended that you use the "ladder/clause-gadget" architecture covered in the course for reductions from (3)SAT.

(6) (18 + 12 = 30 pts.)

Let $A = \bigcup_{m=1}^{\infty}(a+b)^m c(b+c)^m$. The middle '$c$' is not a typo—this is just a bit of extra-help before we consider the related language used on problem (1).

(a) Sketch the design of a deterministic multi-tape Turing machine M such that $L(M) = A$, and M runs in **both** linear time and logarithmic space, with a read-only input tape. It is **not** necessary to give full arc-node detail—it is enough to state some ingredients of your M and show why it operates within the stated time and space bounds.

(b) Now sketch how a **single-tape** deterministic Turing machine M' can recognize A within time $O(n \log n)$. Now the input tape is not read-only, and technically the space is no longer logarithmic (but you're not required to have that here). (For 3 points exam extra-credit, say whether you can still do this even if the middle 'c' is removed, making this the same as L_2 in problem (1).)

(7) (18 + 12 = 30 pts.) ***Short essay answers.***

Most critics of the Church-Turing thesis have argued that it is too strong—i.e. imposes too strong a limitation on human thought process that they contend cannot be modeled by a Turing machine. (The famous physicist Roger Penrose is one of them, in his books *The Emperor's New Mind* and *Shadows of the Mind*.) *Argue the opposite*—argue that the model corresponding to human cognition should be *weaker* than the general Turing machine, such as a Turing machine restricted in its manner of operation or in allotted computational resources.

Write a brief discussion of major points relevant to the topic. Three major and relevant points or examples will suffice for full credit. Your answer should fit within two exam book pages—there is no need to drag it out longer once you show a strong understanding of the issues. This is for 18 points.

Then for the final 12 points, discuss Turing's "other" thesis—namely that via various mathematical systems of logic, human beings can *define* numbers and languages that we cannot *compute*. Give an example of such a number and such a language. Does your argument in the first part make this observation stronger, or weaker?

End of Exam.