CMPS 132 Final Review Sheet, W’06

The final exam will be comprehensive, although material after the midterm will be emphasized. You may have one 3x5 card of handwritten notes, but it will otherwise be a closed-book exam. The format and style of questions will be similar to that on the midterm. Plan on a 3-hour exam, although I may adjust the time after the exam is created. There will be a review session on Monday from 2:00 until 3:30 in E2 room 215.

Before the midterm we covered chapters 3, 4, and sections 5.1 (except for the section on reductions via computation histories) and 5.3 as well as things emphasized in lecture.

The topics include:

1. Decision problems and Languages
2. Turing Machines, transition tables, transition diagrams, configurations, accepting languages, deciding languages, Church-Turing thesis.
3. Combining Turing Machines and standard subroutines (insert, delete, copy, concatenate, addition, etc.).
4. Variations on Turing Machines (examples: Multiple tapes, Two way tape, A finite tape)
5. Non-deterministic Turing Machines and how to “simulate” them with deterministic Turing Machines.
6. TMs can simulate Random Access Machines (RAMs)
7. Universal Turing Machines, encoding letters, strings, and TMs.
8. The Turing recognizable and Turing decidable languages, and their basic definitions ($L$ accepted by TM, TM computes characteristic function for $L$).
9. The sets of regular languages, context free languages, decidable languages, Turing recognizable languages, and the containments between them. Example languages falling in each class.
10. Enumeration characterizations of the decidable and Turing recognizable languages (and why a language is recursively enumerable if and only if it is Turing recognizable).
11. Turing machine tricks: “Time-splicing” and the “triangle trick”
13. Properties of Turing recognizable Languages: closed under union, intersection, concatenation
14. if a language and its complement are Turing recognizable, then the language is decidable.
15. Countably infinite versus uncountable sets, the Real numbers are uncountable, the rational numbers are countable.
17. Languages NSA and SA (from lecture), $A_{TM}$, and other undecidable languages.

18. Constructive reductions (proofs by contradiction in section 5.1) including tricks for reductions: “brain surgery” on TM’s to add pre-processing, post-processing, etc.

19. Mapping reductions (Section 5.3) and their properties.

20. Neither Language $EQ_{TM}$ nor its complement is Turing recognizable.

After the midterm we covered Sections 5.1 (on reductions by computation histories), 5.2 (PCP), 6.1, and chapter 7. We also briefly mentioned topics in section 6.2. These topics include:

1. Using computation histories in reductions

2. Linearly bounded automata and using computation histories to show that $E_{LBA}$ is undecidable.

3. Using computation histories and PDAs to show that $ALL_{CFG}$ is undecidable.

4. Post’s correspondence problem and why it is undecidable (again, using computation histories)

5. Recursion theorem, how a TM can obtain a copy of its own description

6. Results using the recursion theorem: $MIN_{TM}$ is undecidable; there is no universal "bug inducer", a computable function $f$ that maps TM descriptions to modified TM descriptions such that for every TM $M$, $L(M) \neq L(f(M))$.

7. Logical theories (covered briefly): quantified logical statements over $\mathbb{N}, +$ are decidable, quantified logical statements over $\mathbb{N}, +, \ast$ are not. In every sound proof system there are some true statements over $\mathbb{N}, +, \ast$ are true, but not provable.

8. Time complexity, the running time of a (deterministic) TM and the running time of a non-deterministic TM, the worst case running times as a function of the input size $n$ (with review of big-0, little-o notation)

9. Complexity classes $TIME(f(n))$, $NTIME(f(n))$, P, and NP. Examples of languages in P and NP.

10. Robustness of P (the class P remains the same) over deterministic machine models (including RAMs) and input encodings.

11. Our simulation of non-deterministic TMs by deterministic TMs can increase the running time exponentially.

12. Speedup theorem (not in book) showing that if there is a TM deciding a language in time $f(n)$ then there is another more complicated TM deciding the language in time $n + 2 + \frac{1}{2} f(n)$. Therefore, there are generally no "fastest" TMs, and the constant factors are not (at least theoretically) important.

13. Polynomial time verifiers, and that any language $L$ has a polynomial time verifier if and only if $L$ is decided by a polynomial time non-deterministic TM.
14. Polynomial time reductions ($\le_p$) and their properties (transitivity and relating to P and NP).

15. What it means for a language to be NP-complete (definitions and relationship to the P=NP question).

16. Boolean formula satisfiability is the first NP-complete problem - how to create a formula (from an input $w$, a non-deterministic TM $M$, and a bound on $M$’s running time) that is satisfiable if and only $M$ accepts $w$.

17. The fact that 3SAT is NP-complete (the boolean formula above can be re-written as a 3SAT formula)

18. The Clique language is NP-complete (it is in NP and $3\text{SAT} \le_p \text{Clique}$)

19. The VertexCover Language is NP-complete

20. The SubSetSum Language is NP-complete

21. The Hamiltonian path Language is NP-complete (fact only, not the proof)