6.045J/18.400J: Automata, Computability and Complexity Nancy Lynch

Handout 15: Final Exam Information

May 11, 2005

1 General Information on the Final Exam

The final exam will take place from 9:00AM to 12:00PM on **Friday, May 20.**
People who need a makeup exam because of a conflict should see the course staff for information about time and place.

The final will be *cumulative*, covering everything that was covered by the three quizzes, plus the space complexity and probabilistic complexity material covered in the last two weeks of the course. Coverage for the first three units of the course is described in the information handouts for the three quizzes, with handout numbers 2, 7, and 12. These handouts are available on the MIT server.

For the last two weeks, you are responsible for Chapter 8 and Section 10.2 on Probabilistic Algorithms (omitting the subsection on primality) of Sipser's text, as well as all material covered in the last four lectures.

The final is OPEN BOOK, OPEN NOTES. You may bring your copy of Sipser's text with you to class to use during the quiz. You may also bring all course handouts, and all of your own notes. No other material is permitted.

2 Material from the last two weeks that you should know for the final

The following is a list of important topics and ideas we covered in our study of space complexity.

- **Deterministic Space Complexity.** You should know how to measure the space complexity of a Turing machine. You should understand the notion of a deterministic space-bounded complexity class. You should be able to show good space bounds for particular problems, e.g., for CLIQUE and other NP-complete problems.
- **Nondeterministic Space Complexity.** You should understand the notion of a nondeterministic space-bounded complexity class. You should be able to show good nondeterministic space bounds for particular problems, e.g., for the totality problem for NFAs.
- **Deterministic vs. Nondeterministic Space Classes.** You should know the statement of Savitch's theorem and understand its proof. In particular, you should know how to bound the space complexity of machines (like the one in this proof) that implement recursive algorithms. You should understand the modifications to the proof for $f(n) \geq n$ that are required to extend the result to smaller space bounds.
- **The Class PSPACE.** You should know the definition of the complexity class PSPACE. You should know how to show that particular languages are in PSPACE.
- **PSPACE-Completeness.** You should know the definition of PSPACE-completeness. You should know what TQBF is, and should understand the proof that it is PSPACE-complete (both parts).
- **Log Space.** You should know the definition of the language class L, based on two-tape Turing machines. You should be able to describe log space bounded deciders for certain problems, e.g., problems involving counting and matching, and should be able to explain why these in fact halt within log space. You should understand the notion of a configuration for a log-space bounded Turing machine. You should understand why L is a subset of P.
- **Nondeterministic Log Space.** You should know the definition of the language class NL. You should be able to describe nondeterministic log space deciders for basic problems like PATH, and should be able to explain why these halt within log space.
- **Log-Space Reducibility.** You should know the definition of log-space reducibility $\leq_L$, based on log-space transducers. You should understand why log-space transducers run in polynomial time. You should understand why log-space reducibility satisfies basic properties such as transitivity.

- **NL-completeness.** You should know the definition of NL-completeness, and its basic properties. You should understand the proof that PATH is NL-complete, and should know why this implies that $NL \subseteq P$.

- **Complexity Class Relationships.** You should be familiar with the known relationships among the classes L, NL, coNL, P, NP, coNP, PSPACE, and EXPTIME. In particular, you should know that NL = coNL, and should understand the proof reasonably well.

And for probabilistic complexity:

- **Probabilistic Algorithms.** You should understand the notion of probabilistic computation. You should understand the definitions of the various probabilistic complexity classes. In particular, you should be familiar with the definitions and known relationships among the classes BPP, RP, coRP, PP as well as their relations to L, NL, coNL, P, NP, coNP, PSPACE, and EXPTIME.