Theory of Computation
Tutorial IV

Speaker: Yu-Han Lyu

November 14, 2006

A Short History of Computational Complexity

- http://people.cs.uchicago.edu/~fortnow/beatcs/column80.pdf
- Lance Fortnow
  - http://people.cs.uchicago.edu/~fortnow/
  - http://weblog.fortnow.com/

Birth of Computational Complexity

- 1930~1965
  - Recursion theory
  - Computation theory
- Juris Hartmanis and Richard Stearns 1965
  - On the Computational Complexity of Algorithms, *Transactions of the AMS*
- Measure resources, time and memory, as a function of the size of the input problem

Complexity in the ’60s

- Better simulations and hierarchies
- Relationship between time and space, deterministic and nondeterministic
  - Savitch’s Theorem

P versus NP

- Gödel to von Neumann letter in 1956.
- Cook showed Boolean formula satisfiability NP-complete in 1971
- Levin also proved the same result
- Karp in 1972 showed several important combinatorial problems were NP-complete
- Industry in the 1970's of showing that problems were NP-complete

Different Models

- As technology changes so does the notion of “efficient computation”
  - Randomized, Parallel, Distributed, Approximation
- Complexity theorists tackle these issues by defining models and proving relationships between these classes and more traditional models

Approximation complexity

- Decision problem -> Optimization problem
- Find the suboptimal solution
- Focus on NP-Complete problem
  - Some problem cannot be approximated
- Complexity class
  - PTAS, APX, NPO

Probabilistic complexity

- We can flip a coin in the algorithm, and select the next action.
- Determining the primality of a number in 1976 by Miller and Rabin
- Probabilistic Turing machine
- Probability and approximation
- Probabilistic Class
  - ZPP, BPP, PP

Circuit Complexity

- Computers are built from electronic devices wired together in a design called a digital circuit.
- Many circuit can processing in the same time – parallel computing
- Complexity class
  - NC, AC, P-Complete

Communication Complexity

- Alice has string x and Bob has string y, how can they compute f(x, y) with the least communication.
- Introduced by Yao in 1979
- Distributed algorithm
- Complexity class
  - CC

Online Algorithms

- Online algorithm is one that can process its input piece-by-piece, without having the entire input available from the start
- New performance measures have to be introduced
  - Competitive analysis
- Complexity class
  - none

The Role of Mathematics

- Discrete Mathematics
- Logic
- Probability
  - Probabilistic method
- Algebra
  - Coding theory
- Information theory

HW2 Problem 3

- Let A = {wtw^R | w,t $\in${0,1}* and |w| = |t|}. Prove that A is not a context-free language
- Assume p exists, let s=$0^p1^p0^p$
  - Then uvy can not in 1^p, because when we pump down, |w|<|t|
  - Why is this argument false?

Homework 3

- Due
  - 2:10 pm, November 28, 2006 (before class)
- Problem 3 is the easiest (although it has *)
- Problems 2 and 4 are easier
- Problem 1 is harder
- Problem 5 is the hardest
- As simple as possible, but not simpler

Problem 1

- Show that single-tape TMs that cannot write on the portion of the tape containing the input string recognize only regular languages.
- Prove that it equals to read-only
- Proof 1: Read-only TM’s index is finite
- Proof 2: Read-only TM is equal to 2DFA, then prove 2DFA is equal to DFA

Problem 2

- Let A be a Turing-recognizable language consisting of descriptions of Turing machines, {<M_1>, <M_2>, …}, where every M_i is a decider. Prove that some decidable language D is not decided by any decider M_i whose description appears in A.
- Diagonalization technique

Problem 3

- Let PAL_{DFA}={<M>| M is a DFA that accepts some string with more 1s than 0s}. Show that PAL_{DFA} is decidable
- Context Free Language's property

Problem 4

- Let C be a language. Prove that C is Turing-recognizable if and only if a decidable language D exists such that $C=\{x| \exists_y(<x,y> \in D)\}$
- If $x \in C$, then there exists a configuration transition from initial to accept state.
- D is the verifier
- NP's another definition!!

Problem 5

- Show that the problem of determining whether a CFG generates all string in 1* is decidable. In other words, show that {<G>| G is a CFG over {0,1} and $1^* \subseteq L(G)$} is a decidable language.
- No Hint, but some facts. If G is a CFL and R is a regular language, the following is undecidable:
  - Is L(G) = L(R)?
  - Is L(G) = T* for some alphabet T
  - Is $L(R) \subseteq L(G)$