Harvard CS 121 and CSCI E-207

Lecture 25: Conclusions

Salil Vadhan

December 4, 2012

End of Term Miscellany

- PS10 due today (Dec. 4)
- No section this week.
- Office hours this week & next week: see Piazza
- Review sessions next week: see course schedule
- Pick up hardcopy-graded problem sets from Carol Harlow (MD 343)
- We will email you our records of your grades before the exam, please point out any discrepancies within 24 hours

Final Examination

- CS 121 exam: MONDAY, Dec. 17, 2–5pm, Geological Museum Lecture Hall (3 hours)
- CSCI E-207 exam: MONDAY, Dec. 17, 7:40–9:40pm, Sever 209 (2 hours)
- Closed book
- Old exams have been posted for practice, solutions are subject of CS 121 review

Reprise: Models of computation and formal systems

- DFAs, NFAs, REs, CFGs, PDAs, TMs, NTMs,...
- How to to formally model computation
- Asymptotic perspective (fixed program for all input lengths)
- Design your own models as circumstances demand (eg interactive/distributed computation, randomized computation, biological systems, economic systems)

Classification of computational problems

- Positive results: regular, context-free, polynomial-time, decidable, Turing-recognizable languages
- Negative results: non-regular, non-CF, NP-complete(?), undecidable, non-recognizable languages
- Notion of reduction between problems
- The systematic methodology for proving things impossible is one of the most important achievements of computer science.
- NP-completeness is one of the most important "exports" of computer science to the rest of science.

Understanding Intractability

- Many important problems are NP-complete (or even undecidable).
- But also some great positive results in algorithms design
  - E.g. poly-time algs for LINEAR PROGRAMMING, PRIMALITY TESTING, POLYNOMIAL FACTORIZATION, NETWORK FLOWS, ... (take CS124, CS222, CS223, ...)
- What does NP-completeness mean? (assuming P $\neq$ NP)
  - No algorithm can be guaranteed to solve the problem perfectly in polynomial time on all instances
  - Exhaustive search is often unavoidable
  - Mathematical nastiness: no nice, closed form solutions.

Coping with Intractability

- What if you need to solve an NP-complete (or undecidable) problem?
  - Ask your boss for a new assignment. :-)
  - Identify additional constraints that make the problem easier (eg bounded-degree graphs, ILP with fixed number of variables, 2-SAT)
  - Approximation algorithms, e.g. find a TSP tour of length at most 1.01 times the shortest.
  - Average-case analysis — analyze running time or correctness on "random" inputs. (Often hard to find distribution that models "real-life" inputs well.)

More attacks on intractable problems

- Heuristics — techniques that seem to work well in practice but do not have rigorous performance guarantees.
- Change the problem
  - Instead of verifying that general programs satisfy desired security properties (undecidable), ask programmers to supply programs with (easily verifiable) “proofs” that the properties are satisfied
  - Change the programming language (CS 152, CS 252r)

Theory of Computation after CS 121

- Algorithms
  - CS 124: Algorithms & Data Structures
  - CS 222: Algorithms at the End of the Wire
  - CS 223: Probabilistic Analysis and Algorithms
  - More coming next year
- Computational Complexity
  - CS 221: Computational Complexity
  - CS 225: Pseudorandomness

Theory of Computation after CS 121, cont.

- CS 120/220: Introduction to Cryptography
- CS 228: Computational Learning Theory
- CS 229r: Topics in the Theory of Computation (focus in Spring 2013: Data Privacy)
- AM 106: Applied Algebra & Combinatorics
- AM 107: Graph Theory & Combinatorics
- Logic: Math 141 (Mathematical Logic), 144 (Model Theory), EMR 17 (Deductive Logic), Phil 144 (Logic and Philosophy)
- Math 168: Computability Theory
- Many courses in CS & Math at MIT.

Theory Research Group

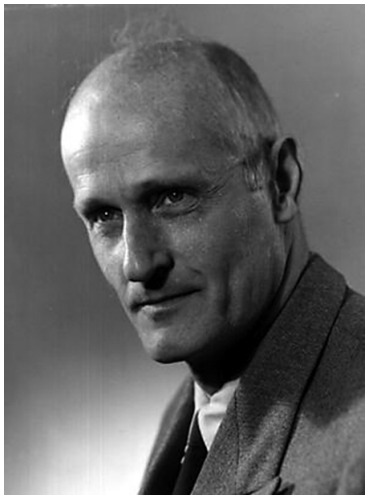
- Theory of Computation research group
  - Group webpage: `http://toc.seas.harvard.edu/`
  - Weekly seminar: Mondays at 5pm in MD119, usually with pizza!
  - `http://toc.seas.harvard.edu/seminar.html`
- Also MIT Theory of Computation group and its seminars
  - `http://theory.csail.mit.edu/`
- Many research opportunities

Connections to the Rest of CS (Partial List)

Circuit Design (CS 141)	Finite Automata
Parsing + Compiling (CS 153)	Context-free Languages Pushdown Automata
Programming Languages (CS 152)	Regular Expressions Formalization in General
Natural Language + Linguistics (CS 187)	Grammars Finite Automata
Program Analysis + Synthesis (CS 153)	Uncomputability
Artificial Intelligence (CS 181,182)	Formal Systems, Logic

Remember the People





Library of Congress