BU CS 332 – Theory of Computation

Lecture 24:

- Final review

Reading:

Sipser Ch 7.1-8.3, 9.1

Mark Bun

April 29, 2020

Final Topics

Everything from Midterms 1 and 2

- Midterm 1 topics: DFAs, NFAs, regular expressions, pumping lemma, context-free grammars, pushdown automata, pumping lemma for CFLs

  (more detail in lecture 9 notes)

- Midterm 2 topics: Turing machines, TM variants, Church-Turing thesis, decidable languages, countable and uncountable sets, undecidability, reductions, unrecognizability, mapping reductions

  (more detail in lecture 17 notes)

Time Complexity (7.1)

- Asymptotic notation: Big-Oh, little-oh, Big-Omega, little-omega, Theta
- Know the definition of running time for a TM and of time complexity classes (TIME / NTIME)
- Understand how to simulate multi-tape TMs and NTMs using single-tape TMs and know how to analyze the running time overhead

P and NP (7.2, 7.3)

- Know the definitions of P and NP as time complexity classes
- Know how to analyze the running time of algorithms to show that languages are in P / NP
- Understand the verifier interpretation of NP and why it is equivalent to the NTM definition
- Know how to construct verifiers and analyze their runtime
- Understand the surprising implications of P = NP, esp. how to show that search problems can be solved in poly-time

NP-Completeness (7.4, 7.5)

- Know the definition of poly-time reducibility
- Understand the definitions of NP-hardness and NP-completeness
- Understand the statement of the Cook-Levin theorem (don’t need to know its proof)
- Understand several canonical NP-complete problems and the relevant reductions: SAT, 3SAT, CLIQUE, INDEPENDENT-SET, VERTEX-COVER, HAMPATH, SUBSET-SUM

Space Complexity (8.1)

- Know the definition of running space for a TM and of space complexity classes (SPACE / NSPACE)
- Understand how to analyze the space complexity of algorithms (including SAT, NFA analysis)

PSPACE and PSPACE-Completeness (8.2, 8.3)

- Know the definitions of PSPACE and NPSPACE
- Know why they're equivalent (statement of Savitch's Theorem)
- Understand how to show that languages are in PSPACE
- Know the definition of PSPACE-completeness
- You will not be asked anything about the PSPACE-complete language TQBF, or to show that any specific language is PSPACE-complete

Hierarchy Theorems (9.1)

- Know that we can prove, unconditionally, that P ≠ EXP and that PSPACE ≠ EXPSPACE
- You will not be asked about the formal statements of the time/space hierarchy theorems, but should understand how they generalize the above statements

Things we didn't get to talk about

- Additional classes between NP and PSPACE (polynomial hierarchy)
- Logarithmic space
- Relativization and the limits of diagonalization
- Boolean circuits
- Randomized algorithms / complexity classes
- Interactive proof systems
- Complexity of counting

https://cs-people.bu.edu/mbun/courses/535_F20/

Tips for Preparing Exam Solutions

Designing (nondeterministic) time/space-bounded deciders

The following algorithm decides EC in polynomial time:

"On input $\langle A, C, e, p \rangle$, four binary integers:

1. Let $r \leftarrow 1$.

...

6. If $C = r \mod p$, *accept*; otherwise *reject*."

The algorithm is called *repeated squaring*.

Let $T(d)$ denote a polynomial upper bound on the running time of basic procedures for multiplication and modular operations on d-bit numbers. Then steps 4 and 5 of the algorithm each take at most $O(T(\log A) + T(\log p))$ time because r is never larger than p. In addition, the total number of multiplication and modular operations is $O(k) = O(\log e)$. Therefore, the total running time of the algorithm is polynomial in $O((\log e) \cdot (T(\log A) + T(\log p)))$ which is polynomial in n. Hence, the total running time is polynomial. Note that without performing mod p operation in Steps 4 and 5,

- Key components: High-level description of algorithm, analysis of running time and/or space usage
- A good idea: Explain correctness of your algorithm

Designing NP verifiers

We give a poly-time verifier for $TEAM$. A certificate c for our verifier is a subset of M of size k.

"On input $\langle n, X, Y, Z, M, k; c\rangle$ where $\langle n, X, Y, Z, M, k\rangle$ is a $TEAM$ instance and c is a certificate:

1. If $k > \min(n, |M|)$, *reject*.
2. Check whether $|c| = k$ and $c \subseteq M$.
3. Check whether all elements of triples in c are different.
4. If any of these checks fails, *reject*; otherwise, *accept*."

Step 1 is performed to ensure that the running time is polynomial in n even for large k. Step 2 can be run in $O(k \cdot |M|) = O(|M|^2)$ time, by iterating through M and marking elements. Step 3 can be implemented to run in $O(|c| \log |c|)$ time by first sorting the elements of c. This verifier runs in polynomial time; hence, $TEAM \in$ NP.

- Key components: Description of certificate, high-level description of algorithm, analysis of running time
- A good idea: Explain correctness of your algorithm

NP-completeness proofs

To show a language L is NP-complete:

1) Show L is in NP (follow guidelines from previous two slides)

2) Show L is NP-hard (usually) by giving a poly-time reduction $A \leq_p L$ for some NP-complete language A

   - High-level description of algorithm computing reduction
   - Explanation of correctness: Why is $w \in A$ iff $f(w) \in L$ for your reduction f?
   - Analysis of running time

Practice Problems

P

Give examples of the following languages: 1) A language in P. 2) A decidable language that is not in P. 3) A language for which it is unknown whether it is in P.

Give an example of a problem that is solvable in polynomial-time, but which is not in P

Let $L = \{\langle w_1, w_2 \rangle | \exists \text{ strings } x, y, z \text{ such that } w_1 = xyz \text{ and } w_2 = xy^R z\}$. Show that $L \in \mathrm{P}$.

Which of the following operations is P closed under? Union, concatenation, star, intersection, complement.

NP and NP-completeness

Prove that $LPATH = \{\langle G, s, t, k\rangle | G$ is an undirected graph containing a simple path from s to t of length $\geq k\}$ is in NP

Prove that $LPATH$ is NP-hard

Which of the following operations is NP closed under? Union, concatenation, star, intersection, complement.

Show that if P = NP, there is a polynomial-time decider for $USAT = \{\langle \phi \rangle | \phi$ is a formula with exactly one satisfying assignment$\}$

Space Complexity

Which of the following statements are true?

- $SPACE(2^n) = SPACE(2^{n+1})$
- $SPACE(2^n) = SPACE(3^n)$
- $NSPACE(n^2) = SPACE(n^5)$

Consider the inheritance problem from HW9, except Alice and Bob now take turns drawing bags from boxes. Alice's goal is to assemble a complete collection of marbles, and Bob's is to thwart her. Prove that determining whether Alice has a winning strategy is in PSPACE.