

BU CS 332 – Theory of Computation

Lecture 25:

- Final review

Reading:

Sipser Ch 7.1-8.2, 9.1

Mark Bun

April 28, 2021

Final Topics

Everything from Midterms 1 and 2

- **Midterm 1 topics:** DFAs, NFAs, regular expressions, distinguishing set method
(more detail in lecture 8 notes)
- **Midterm 2 topics:** Turing machines, TM variants, Church-Turing thesis, decidable languages, countable and uncountable sets, undecidability, reductions, unrecognizability
(more detail in lecture 16 notes)

Mapping Reducibility (5.3)

- Understand the definition of a computable function
- Understand the definition of a mapping reduction
- Know how to use mapping reductions to prove decidability, undecidability, recognizability, and unrecognizability

Time and Space Complexity (7.1)

- Asymptotic notation: Big-Oh, little-oh
- Know the definition of running time and space for a TM and of time and space complexity classes (TIME / NTIME / SPACE / NSPACE)
- 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

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

Hierarchy Theorems (9.1)

- Formal statements of time and space hierarchy theorems and how to apply them
- How to use hierarchy theorems to prove statements like $P \neq EXP$

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 and probabilistic proof systems
- Complexity of counting

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

Theory and Algorithms Courses after 332

- Algorithms
 - CS 530/630 (Advanced algorithms)
 - CS 531 (Optimization algorithms)
 - CS 537 (Randomized algorithms)
- Complexity
 - CS 535 (Complexity theory)
- Cryptography
 - CS 538 (Foundations of crypto)
- Topics (CS 591)

E.g., Privacy in machine learning, algorithms and society, sublinear algorithms, new developments in theory of computing, communication complexity

Algorithms and Theory Research Group

- <https://www.bu.edu/cs/research/theory/>
- Weekly seminar: Mondays at 11
<https://www.bu.edu/cs/algorithms-and-theory-seminar/>

Great way to learn about research in theory of computation!

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 $C = r \pmod p$, *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 $\pmod p$ operation in Steps 4 and 5,**

- Key components: High-level description of algorithm, explanation of correctness, analysis of running time and/or space usage

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, explanation of correctness, analysis of running time

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

Use a mapping reduction to show that $ALL_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \Sigma^*\}$ is co-unrecognizable

Use a mapping reduction to show that $ALL_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \Sigma^*\}$ is unrecognizable

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 $L =$

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

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

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

Prove that *LPATH* is NP-hard

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

Which of the following statements are true?

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

- $SPACE(2^n) = SPACE(3^n)$

- $NSPACE(n^2) = SPACE(n^5)$

