CMSC 858M: Algorithmic Lower Bounds
Spring 2021
Chapter 1: Computational Complexity Crash Course

Instructor: Mohammad T. Hajiaghayi
Scribe: Anthony Ostuni

December 12, 2021

1 General Overview

I really enjoyed this chapter. It's pretty standard material, but I haven't seen most of it taught with this level of rigor and background information. My only minor concern (as detailed below) is that sometimes this focus on rigor makes it a bit difficult to grasp certain definitions, but it's never too too difficult, and of course this book is targeted at a more advanced audience, so I don't think this is a notable flaw.

2 Section 1.1

I like the specification of what *problem* actually means. It is so often glossed over that I originally thought defining such a convention was pedantic – that is until I realized I couldn't define it with any rigor. Overall, good overview of P.

3 Section 1.2

The section is titled 'Reductions', but you don't actually use that word (or any variant) anywhere in the section. May be helpful to note (either explicitly or by some example) how to properly use that terminology in context.

4 Section 1.3

I haven't seen that particular definition of NP before (it's more rigorous than what I've seen). I found the increased rigor interested, but I'm also appreciative of intuition discussion that immediately follows; it made it easier to comprehend the formal definition.

While interesting, I'm not sure 12 examples of NP are necessary - by the fifth or sixth one they already start feeling repetitive, and I'd be surprised if the average reader went through that whole list. (If you do end up removing some, do note that you reference this list later, so adjust any later references, as well).

5 Section 1.4

This was probably the most interesting section to me. I specifically had never know about the fact that $\mathsf{P} \neq \mathsf{NP}$ would settle many open problems in a similar fashion to what is discussed in point 3.

6 Section 1.5

I originally had some difficulty processing this section. My thought process (for VC for example) was "if VC_k can be done in polytime, then for the pair (G, k), why can't we just use the VC_k algorithm on G to do VC in polytime." I believe I now understand that you can't treat k as a constant in the latter case, so it wouldn't be polynomial anymore, but I do think this section could be either explained a bit more, or make it clear that this material will be explained more in whatever later chapters are relevant to it.

7 Section 1.6

Good - no comments.

8 Section 1.7

This section introduces a cool distinction that I was not aware of. No real critiques – I just think it's interesting.

9 Section 1.8

Good - no comments.

10 Section 1.9

I would have appreciated a bit of intuition for Π_1, Σ_2, and Π_2, similar to what was given when the formal definition for NP was given (perhaps even a bit briefer since I'm assuming the intuition is similar). However, relying on the intuition for the definition of NP, I don't think it's terrible to comprehend the definitions.

11 Section 1.10

I really enjoyed the discussion about factoring and graph isomorphism. I was relatively familiar with the state-of-the-art algorithms, but I was not aware of the ramifications of FACT and GRAPH ISOM being NPC.

12 Section 1.11

I'm unfamiliar with what a diagonalization argument means in this context. I would've liked a reference to either a paper/book or a later chapter that discusses it.

13 Important Related Problems

13.1 Problems in NP

There are several important problems in NP that may be worth mentioning in this first chapter.

Chromatic Number Problem: Given a graph and an integer c, does there exist a vertex coloring such that no two adjacent vertices receive the same color? This problem is one of Karp's 21 NP-complete problems [10], but its origins can be traced back at least as far as the famous four color conjecture in the mid-1800s. The conjecture states that any map can be colored in at most four colors, such that no two adjacent areas receive the same color. In graph theoretic language, the conjecture states the chromatic number of a planar graph is at most four. It was famously proven by Appel and Haken in one of the initial uses of computer-assisted proof [1].

Knapsack Problem: Given a set of items with weights and values, is it possible to obtain some value at least V without exceeding some weight W? While this is often listed as one of Karp's 21 NP-complete problems [10], Karp's version is closer to the subset sum problem case where the weights equal the values. The knapsack problem is particularly interesting, as the weights and values determine whether or not the problem is strongly NP-complete; if they are integers, the problems is weakly NP-complete, while if they are rational numbers, the problem is strongly NP-complete [12].

Traveling Salesman Problem: Given a (edge) weighted graph, does there exist a Hamiltonian cycle of weight at most W? The optimization version of this problem is one of the most famous problems in combinatorial optimization. Karp showed that checking if a graph contains a Hamiltonian cycle is NP-complete [10], and this complex problem also falls into that category. See [4] for a general overview of the problem and its history.

Dominating Set Problem: Given a graph and an integer D, does there exist a vertex set V of size at most D, such that every vertex not in V is adjacent to some vertex in V? This problem can be shown to be NP-complete through a reduction from the set cover problem, one of Karp's 21 NP-complete problems [10].

13.2 Strong NP-complete Examples

It would be nice to prove some examples of strong NP-complete problems when introducing the topic.

Maximum Independent Set Problem: Given a graph, find the largest vertex set such that no two vertices are adjacent. This is strong NP-complete [8]. While not one of Karp's 21 NP-complete problems [10], it has connections with two of them: vertex cover and clique. Indeed, it is easy to see that an independent set is a clique in the graph's complement, and the set's complement is a vertex cover.

3-Partition Problem: Given a list of integers, can they be partitioned into three groups with identical sums? This was originally shown to be NP-complete by Garey and Johnson using a reduction from 3-dimensional matching [7]. The 3-partition problem is used in many reductions.

Bin Packing Problem: Given a set of items of varying volumes and k bins of a fixed volume, can the items fit in the k bins? This problem can be shown to be NP-complete using a reduction from the previous 3-partition problem. Bin packing is rich with work on approximate algorithms; see [3] for a survey.

13.3 Possible NP-Intermediate Problems

Two more interesting possibly NP-Intermediate problems are as follows.

Discrete Logarithm Problem: Let a and b be elements of a group G, and let k be a positive integer. Determine k that solves the equation $b^k = a$. No polynomial time algorithm (for classical computers) is known, but the problem has not been shown to be NP-complete, either. There does exist a polynomial time quantum algorithm, however [11]. This problem is used in several popular public-key cryptography algorithms [5, 6].

Minimum Circuit Size Problem: Given the truth table of a boolean function f and an integer s, does f have a circuit with at most s logic gates? This is an important problem, as smaller circuits minimize resources required in the production of integrated circuits. It is unknown whether this problem is in P, nor whether it is NP-complete. Kabanets and Cai have provided arguments for why it may not be in either class [9].

13.4 Undecidable Problem Example

The following is a seemingly basic undecidable problem.

Mortal Matrix Problem: Given a finite set of $n \times n$ matrices, determine whether they can be multiplied in some order (possibly with repetition) to obtain the zero matrix. For certain parameters – a set of two 15×15 matrices for example – this is known to be undecidable [2].

References

[1] Kenneth Appel and Wolfgang Haken. *Every planar map is four colorable*, volume 98. American Mathematical Soc., 1989.

[2] Julien Cassaigne, Vesa Halava, Tero Harju, and François Nicolas. Tighter undecidability bounds for matrix mortality, zero-in-the-corner problems, and more. *arXiv preprint arXiv:1404.0644*, 2014.

[3] Edward Coffman Jr, Michael Garey, and David Johnson. Approximation algorithms for bin packing: A survey. *Approximation algorithms for NP-hard problems*, pages 46–93, 1996.

[4] William Cook. *In pursuit of the traveling salesman: mathematics at the limits of computation.* Princeton University Press, 2011.

[5] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[6] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[7] Michael Garey and David Johnson. Complexity results for multiprocessor scheduling under resource constraints. *SIAM Journal on Computing*, 4(4):397–411, 1975.

[8] Michael Garey and David Johnson. "Strong" NP-Completeness Results: Motivation, Examples, and Implications. *Journal of the ACM (JACM)*, 25(3):499–508, 1978.

[9] Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 73–79, 2000.

[10] Richard Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[11] Peter Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[12] Dominik Wojtczak. On strong NP-completeness of rational problems. In *International Computer Science Symposium in Russia*, pages 308–320. Springer, 2018.