Rashid Lasker

Kate Kostelnik

ENWR 2520

25 April 2017

P ≟ NP

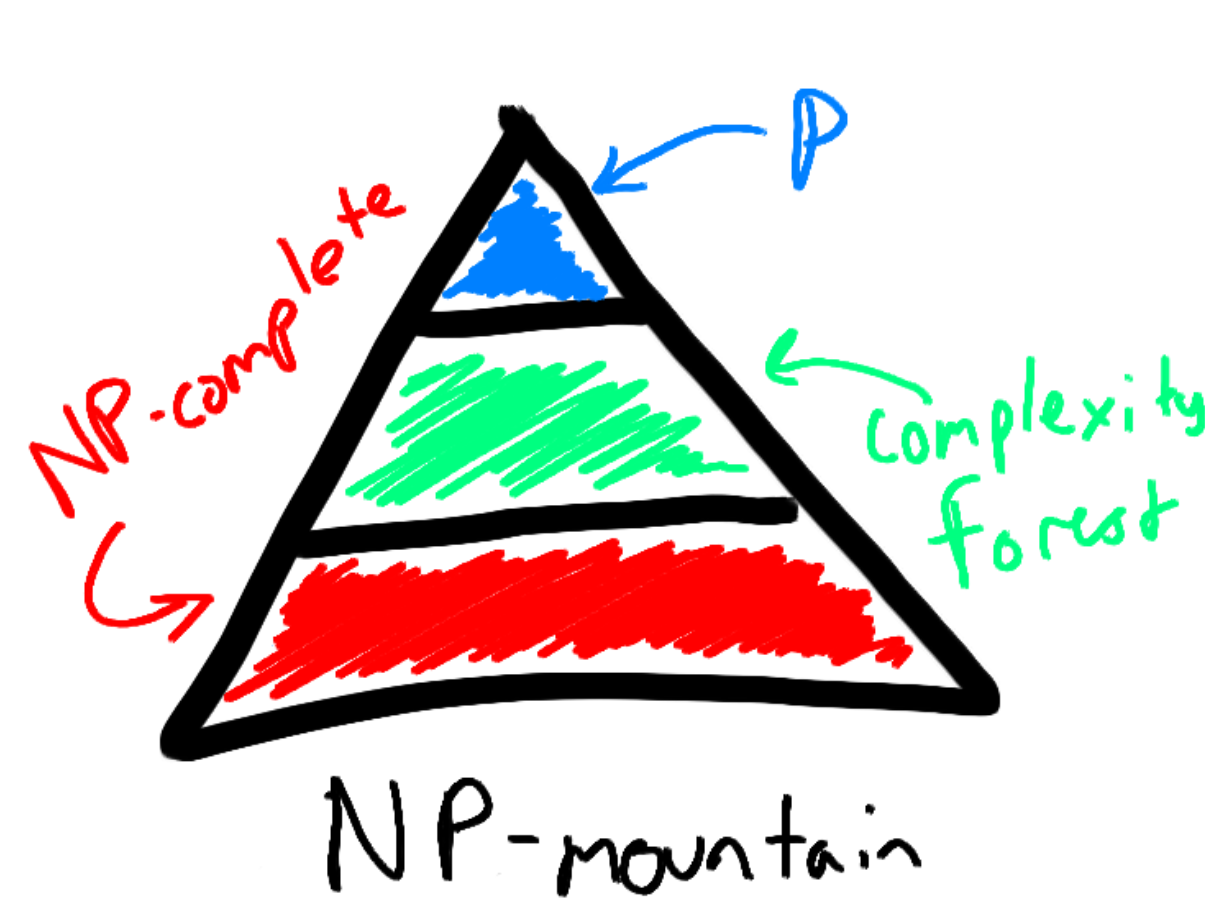
Introduction/Hook:

If you have ever spent a lazy Sunday afternoon playing with puzzles, you have most likely come across a puzzle known as Sudoku. The goal is simply to fill a 9 by 9 grid with single digit numbers in a way that each digit is not repeated in specific columns, rows, and 3 by 3 squares. The underlying concept of this puzzle is not profoundly complex. All it is rearranging numbers in a grid. However, it still presents a challenging mind game as far as puzzles go. Beyond that, it also plays a key role in one of computer science’s most interesting problems, P vs. NP.

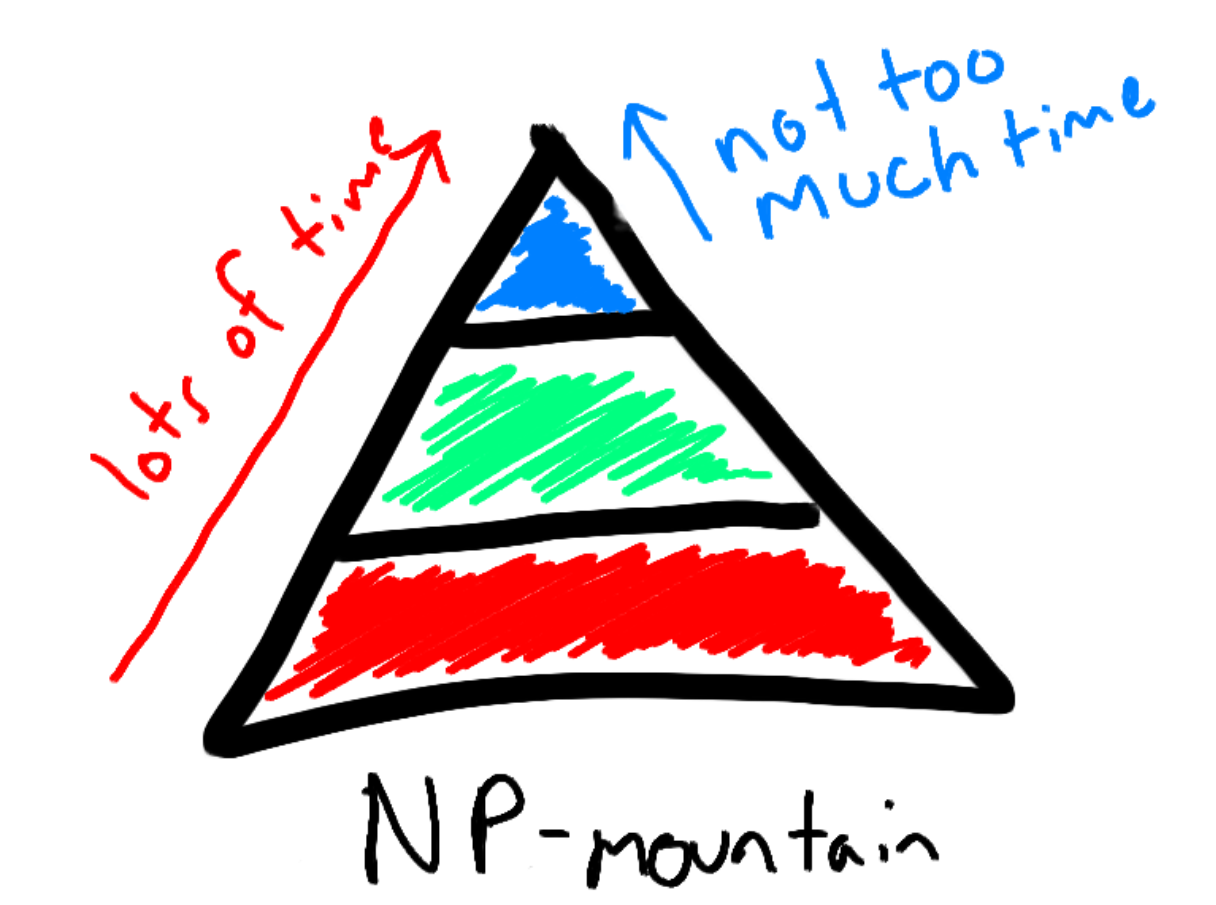
Solving sudoku puzzles is far more difficult than it seems at first glance. As you scale up the difficulty, going beyond 16x16 grids, the problem becomes exponentially more complex. In fact, solving sudoku is an NP-complete problem, which we will go over later (Yato, 2003). This means sudoku puzzles are on the same plane of complexity as protein folding problems. Somehow, this simple puzzle is as difficult as something we are using to understand and combat cancer. This wide spectrum of problems within the same complexity level is why the P vs. NP problem is so important.

An analogy:

In an attempt to visualize the P vs. NP problem, let’s imagine ourselves on a mountain. We might not be mountaineers in our free time, but our goal here is to reach the summit. The distance to the summit varies on where we start. Let’s imagine the mountain in the following way:



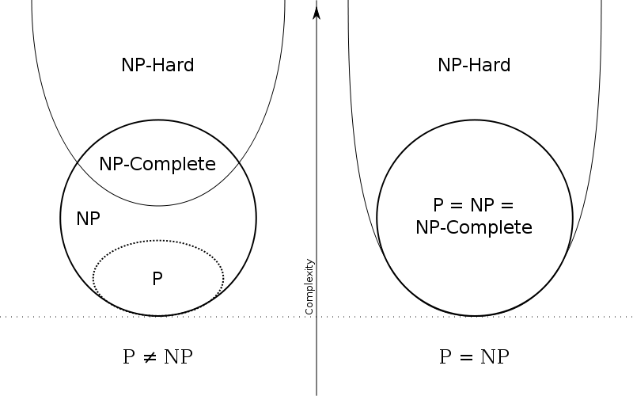
The three zones are separated as follows. The P-zone is the section closest to the summit. From here, it’s possible to reach the top of the summit in a reasonable amount of time. The other two zones are the NP-complete-zone and the Complexity Forest. Essentially, the Complexity Forest separates the P-zone and the NP-complete-zone. As a result, if you start off in the NP-complete-zone, it takes a significantly longer time to reach the summit.

 From both zones, it’s easy to check if you’re at the summit; you just look down at your feet. However, the time it takes to reach the summit from either zone is very different. Herein lies the main difference between P and NP-complete problems. Both types of problems are easy to check if you’re given a solution already. However, it is far easier to find a solution to a P problem, compared to an NP-complete problem.

The above analogy assumes that P ≠ NP. In this scenario, there’s no easy way to bypass the Complexity Forest. However, there is alternative possibility, where P = NP. This would mean that there is not a difference between P and NP problems. Going back to the analogy, if P = NP, there would be a ski-lift that goes over the Complexity Forest entirely. For every NP-complete problem, there would be a shortcut that turns it into a P problem. Therefore, all NP-complete problems would be able to be solved faster, which has widespread ramifications in today’s world.



Formalization of the problem:



* Definitions

History of the problem:

* Computer Scientists in the 70s trying to categorize problems
* Unsure if certain problems had a shortcut.

What makes P ≟ NP so difficult to prove?

* Proving P vs. NP one way or another is an NP-complete problem in it of itself.

The future of P ≟ NP:

There is a school of thought that believes that the overall result of the P vs. NP problem will be inconsequential in the end. This is because of sizeable majority of computer scientists believe that P ≠ NP as is. If P ≠ NP is proven, then nothing will change because of the assumptions already made within the computer science industry. Advances in theoretical computation will thereby rely on things like quantum computing to radically disrupt the speed at which we solve certain problems.

However, there are several computer scientists that are optimistic about the potential about the P = NP problem:

* “If P = NP, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in "creative leaps," no fundamental gap between solving a problem and recognizing the solution once it's found.” – Scott Aaronson, MIT
* “The main argument in favor of P ≠ NP is the total lack of fundamental progress in the area of exhaustive search. This is, in my opinion, a very weak argument. The space of algorithms is very large and we are only at the beginning of its exploration. [...] The resolution of Fermat's Last Theorem also shows that very simple questions may be settled only by very deep theories.” — Moshe Y. Vardi, Rice University
* [the above two quotes will be displayed uniquely on the website]

Ultimately, we continue to wait on the answer to the P vs. NP problem.

Resources

Aaronson, Scott. “P ≟ NP.” *The University of Texas*, 2017.

Sipser, Michael. “The History and Status of the P versus NP Question.” *Massachusetts Institute of Technology*, 1992.

Yato, Takayuki. “Complexity and Completeness of Finding Another Solution and its Application to Puzzles.” *The University of Tokyo*, 2003.