How hard is it to determine how to win?

Mario Sanchez

Group: U2

Mr. Wylie

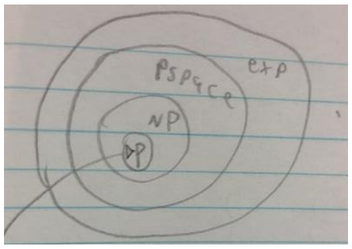
CSCI 4341 Games & Computation

2/27/2022

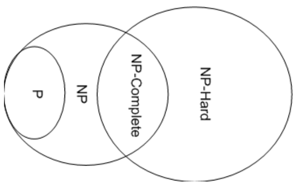
How hard is it to determine how to win? This was the question that led to knowing what types of problems are there and their respective difficulties eventually leading to P vs NP (One of the seven millennium prize problems) to know if a game has a difficulty at P, NP, or worst. To better understand this idea of difficulty, it is necessary to define what is P and NP. (P) Polynomial Time is the amount of steps you need to solve a problem which could be said to be the amount of time it takes to solve the problem s = f(N) where s is the # of steps, f is the polynomial function, and N is the size. One brief example would be 2N – 3 and could cover problems like mazes and multiplication. (NP) Non-deterministic Polynomial Time is checking all paths at the same time and therefore getting the answer in polynomial time, as stated by Hazel (2014). In class it was explained using a tree diagram following multiple paths at the same time and where only one branch would end up having the correct solution.

In terms of difficulty, it is good to know that P refers to those problems where there is a fast way to achieve a solution to a particular problem. While NP, refers to a problem where there is a way to verify the solution to a problem. An example of a P type of problem would be multiplying five times twenty. A solution can be achieved by simply doing it in our heads or by paper in just some seconds. As for NP, even if an answer cannot be achieved in polynomial time, the solution can be verified in polynomial time. For example, the subset sum problem whereas the name indicates, you are supposed to find a subset where all of its components add up to a desired number. Let's take a set of numbers where the set S = {-1, 2, 7, 10, 6, 2, 1} and you take two sets A = {-1, 6} and B = {2, 6}, hopping that the sum of each set is equal to five. In this case A is correct while B is wrong and both answers can be verified in polynomial time once the answer has been given to us.

In class we went through a diagram that showed the different levels of complexity (from P to EXP) where for every level moving out, the complexity would grow.



Lucky for us, we only need to take in consideration the first four levels which we will go a bit more in detail of the classifications of P and NP.



As we already know, P problems are those problems that can be realistically solved in polynomial time. NP don't need a solution but just a way to prove that solution to be true or false. Then there is the NP-hard problem whereas stated by Mann (2017, p.1) there is no known optimal polynomial time algorithm to solve it. In other words, NP-hard are those problems with no proven optimized algorithm and therefore we could say that there is no known correct way to do them. Everything above NP-hard is something that won't be discussed in this paper. Finally, the last level in NP that we got to see more in detail at class was NP-Complete. These NP-complete problems don't have an optimized polynomial time algorithm, but they can be verified in polynomial time. In class, we covered the 3 SAT were SAT stands for satisfiability. SAT is used to "decide whether a formula consisting of Boolean variables, negation, disjunction, and conjunction operators evaluates to true for some assignment of logical values to the variables" (Mann, 2017, p. 6). The version 3 SAT uses Boolean equations that contains clauses with three inputs each and you can tell it is a clause by the number of inputs it contains and the parenthesis. For example: (A V B V C) ^ (A V !B V !C) ^ (!A V !B V !C). for this NP-complete problem, the only question we have is if there is an assignment to the number of variables that would make this Boolean equation satisfiable (true)?

What is the big question P vs NP? Why is it such a big deal? The problem P vs NP is so problematic that is part of the seven millennium prize problems, and the problem is still a big thing today because no one has a proven answer. P vs NP starts by saying that P type problems are in every NP type of problems because if you can find an answer in a reasonable time then we can verify the answer in a reasonable time. The problem starts when you try the opposite. Is NP in P? it is crucial to know that because you can prove that something is correct, it doesn't mean that you can find the correct answer in a reasonable time or even find one. In other words, the problem P vs NP asks if P is equal to NP? I would give the answer but just like many others, I don't know the solution.

Things to remember:

1. P type problems are those that we can find an answer in a decent amount of time.

2. NP type problems are those where we can verify the answer since getting and answer and proving an answer is not the same thing.

3. Everything above NP-hard is crazy hard. Don't go there.

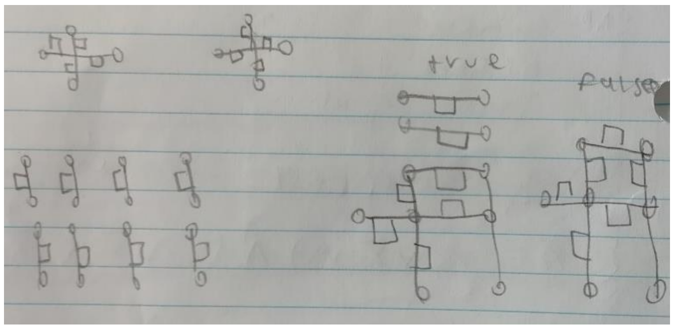
4. NP-hard are those problems where there is no known optimized polynomial time algorithm to solve said problem.

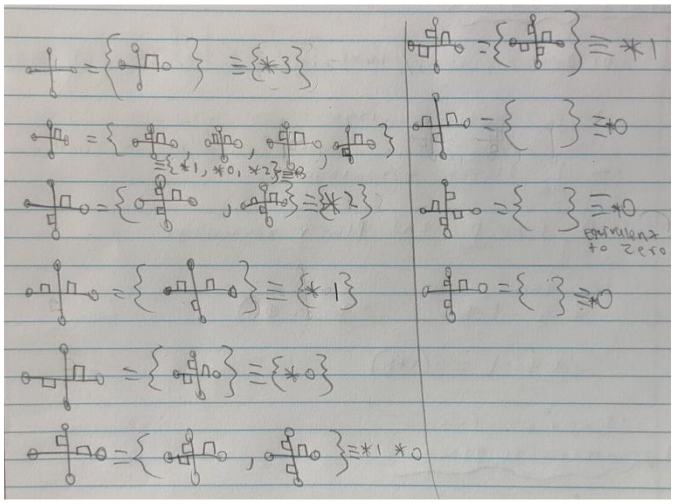
5. NP-complete is the combination of both NP and NP-hard. Answer is verifiable, but no known optimized algorithm to solve the problem in polynomial time.

6. no one is sure of how to solve the "P = NP" problem or even prove it.

Games we went through in class:

The naming the streets game was basically about giving a tag to each street without running out of space and being able to name all of them at the end.





Hope these notes help to whoever confused :b

References

Mann, Z. (2017) ‘The Top Eight Misconceptions about NP-Hardness’. *IEEE Computer*, *50*(5) pp. 72-79.

https://www.researchgate.net/publication/316897910_The_Top_Eight_Misconceptions_about_NP-Hardness

Hazel, Steven. [heckerdashery]. (08/26/2014). P vs. NP and the Computational Complexity Zoo [Video]. You tube.

https://youtu.be/YX40hbAHx3s