Introduction to Theory of Computation

Part IV-A:
Time Complexities
About this Part

• What is NP?
  • How to check if a problem is in NP?

• Cook-Levin Theorem
  • Showing one of the most difficult problem in NP

• Problem Reduction
  • Finding other most difficult problems
Decision Problems

• When we receive a problem, the first thing concern is: whether the problem has a solution or not

• E.g., Peter gives us a map $G = (V,E)$, and he asks us if there is a path from $A$ to $B$ whose length is at most 100

• E.g., Your sister gives you a number, say $11111111111111111$ (19 one’s), and asks you if this number is a prime
Decision Problems

• The problems in the previous page is called decision problems, because the answer is either YES or NO.

• Some decision problems can be solved efficiently, using time polynomial to the size of the input.

• We use $P$ to denote the set of all these polynomial-time solvable problems.
Decision Problems

E.g., For Peter’s problem, there is an $O(V \log V + E)$-time algorithm that finds the shortest path from $A$ to $B$;

- we can first apply this algorithm and then give the correct answer

- Peter’s problem is in $P$

• Can you think of other problems in $P$?
Decision Problems

- Another interesting classification of decision problems is to see if the problem can be verified in time polynomial to the size of the input.

- Precisely, for such a decision problem, whenever it has an answer YES, we can:
  1. Ask for a short proof, and
     /* short means: polynomial in size of input */
  2. Be able to verify the answer is YES.
Decision Problems

E.g., In Peter’s problem, if there is a path from $A$ to $B$ with length $\leq 100$, we can:

1. Ask for the sequence of vertices (with no repetition) in any path from $A$ to $B$ whose length $\leq 100$

2. Check if it is a desired path (in poly-time)

$\Rightarrow$ this problem is polynomial-time verifiable
Polynomial-Time Verifiable

More examples:

*Given a graph $G = (V, E)$, does the graph contain a Hamiltonian path?*

*Given a set of numbers, can be divide them into two groups such that their sum are the same?*
Now, imagine that we have a super-smart computer, such that for each decision problem given to it, it has the ability to guess a short proof (if there is one). With the help of this powerful computer, all polynomial-time verifiable problems can be solved in polynomial time (how?)
The Class NP

- Because of this, we use $\text{NP}$ to denote the set of polynomial-time verifiable problems.
  - $N$ stands for non-deterministic guessing power of our computer.
  - $P$ stands for polynomial-time solvable.
P and NP

• We can show that a problem is in $P$ implies that it is in $NP$ (why?)
  • Because if a problem is in $P$, and if its answer is $YES$, then there must be an algorithm that runs in polynomial-time to conclude $YES$ ...
  • Then, the execution steps of this algorithm can be used as a “short” proof
P and NP

• On the other hand, after many people’s efforts, some problems in $\text{NP}$ (e.g., finding a Hamiltonian path) do not have a polynomial-time algorithm yet ...

• Question: Does that mean these problems are not in $\text{P}$ ??

• The question whether $\text{P} = \text{NP}$ is still open

Clay Mathematics Institute (CMI) offers US$ 1 million for anyone who can answer this ...
P and NP

• So, the current status is:
  1. If a problem is in $P$, then it is in $NP$
  2. If a problem is in $NP$, it may be in $P$

• In the early 1970s, Stephen Cook and Leonid Levin (separately) discovered that:
  a problem in $NP$, called SAT, is very mysterious...
Cook-Levin Theorem

If SAT is in \( P \), then every problems in \( NP \) are also in \( P \)

- I.e., if SAT is in \( P \), then \( P = NP \)

// Can Cook or Levin claim the money from CMI yet?

- Intuitively, SAT must be one of the most difficult problems in \( NP \)
- We call SAT an NP-complete problem (most difficult in \( NP \))
Satisfiable Problem

• The SAT problem asks:
• Given a Boolean formula $F$, such as
  \[ F = (x \lor y \lor \neg z) \land (\neg y \lor z) \land (\neg x) \]
  is it possible to assign True/False to each variable, such that the overall value of $F$ is true?

Remark: If the answer is YES, $F$ is a satisfiable, and so it is how the name SAT is from
Other NP-Complete Problems

• The proofs made by Cook and Levin is a bit complicated, because intuitively they need to show that no problems in NP can be more difficult than SAT

• However, since Cook and Levin, many people show that many other problems in NP are shown to be NP-complete
  • How come many people can think of complicated proofs suddenly ??
Problem Reduction

• How these new problems are shown to be NP-complete rely on a new technique, called reduction (problem transformation)

• Basic Idea:
  • Suppose we have two problems, A and B
  • We know that A is very difficult
  • However, we know if we can solve B, then we can solve A
  • What can we conclude ??
Problem Reduction

• Now, consider
  \( A = \) an NP-complete problem (e.g., SAT)
  \( B = \) another problem in NP

• Suppose that we can show that:
  1. we can transform a problem of \( A \) into a problem of \( B \), using polynomial time
  2. We can answer \( A \) if we can answer \( B \)

\( \Rightarrow \) Then we can conclude \( B \) is NP-complete

(Can you see why??)
Example

- Let us define two problems as follows:
  - The **CLIQUE** problem:
    
    Given a graph $G = (V, E)$, and an integer $k$, does the graph contain a complete graph with at least $k$ vertices?
  
  - The **IND-SET** problem:
    
    Given a graph $G = (V, E)$, and an integer $k$, does the graph contain $k$ vertices such that there is no edge in between them?
Example

• Questions:
  1. Are both problems decision problems?
  2. Are both problems in \textit{NP}?

• In fact, \textit{CLIQUE} is \textit{NP}-complete
  
  • Can we use reduction to show that \textit{IND-SET} is also \textit{NP}-complete?

  [ transform which problem to which?? ]