

# CSC373 Worksheet 7 Solution

August 14, 2020

## 1. My Work

The longest simple cycle problem is the problem of finding a cycle of maximum length in a graph [5].

The decision problem is, given  $k$ , to determine whether or not the instance graph has a simple cycle of length at least  $k$ . If yes, output 1. Otherwise, output 0.

## My Work

The language corresponding to the decision problem is as follows:

LONGEST-SIMPLE-CYCLE =  $\{ \langle G, v_0, v_1, \dots, v_k, k \rangle : G = (V, E) \text{ is an undirected graph}$   
 $k \geq 3 \text{ is an integer,}$   
 $v_0, v_1, \dots, v_k \in V \text{ are distinct,}$   
 $v_0 = v_k,$   
There should exist a simple cycle in  $G$   
with at least  $k$  edges }

## Correct Solution:

The problem LONGEST-SIMPLE-CYCLE is a relation that associates each instance of a graph with the longest simple cycle in that graph .

The decision problem is, given  $k$ , to determine whether or not the instance graph has a simple cycle of length at least  $k$ . If yes, output 1. Otherwise, output 0.

The language corresponding to the decision problem is as follows:

LONGEST-SIMPLE-CYCLE =  $\{\langle G, k \rangle : G = (V, E) \text{ is an undirected graph}$   
 $k \geq 0$  is an integer,  
 There should exist a simple cycle in  $G$   
 with at least  $k$  edges $\}$

### Notes

- **A Cycle in an Undirected Graph**
  - A path  $\langle v_0, v_1, \dots, v_k \rangle$  forms a cycle if  $k \geq 3$ , and  $v_0 = v_k$ .
- **Simple Cycle**
  - A cycle is simple if  $v_1, v_2, \dots, v_k$  are distinct
- **Decision Problem**
  - Is the problem with yes/no solution
- **Alphabet**
  - Is a finite set of symbols
  - Is denoted  $\Sigma$

#### Example:

For decision problem, its alphabet is:  $\Sigma = \{0, 1\}$

- \* 1 means 'yes'
- \* 0 means 'no'

- **Language**
  - Is any set of strings made of symbols from  $\Sigma$
  - Is denoted  $L$

#### Example:

$L = \{10, 11, 101, 111, 1011, 1101, 10001\}$

- Is denoted  $\Sigma^*$  for language of all strings over  $\Sigma$  plus empty string  $\epsilon$ .

#### Example:

$\Sigma^* = \{\epsilon, 0, 1, 00, 01, 11, 000, \dots\}$

#### Example 2:

The decision problem PATH has the corresponding language

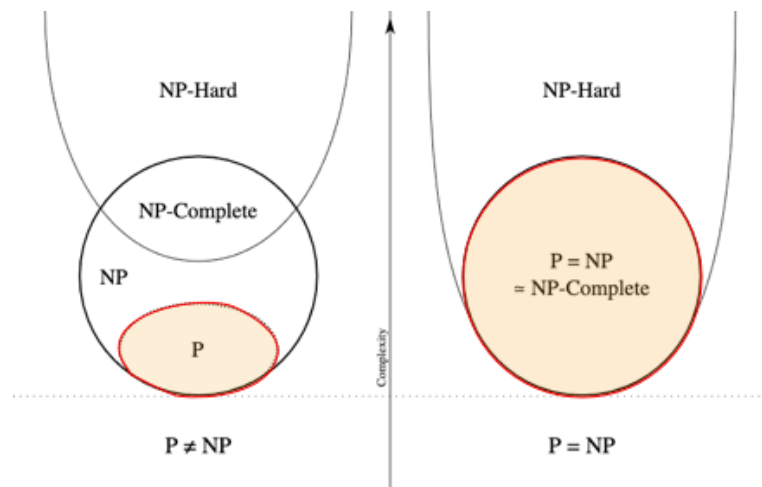
$$\text{PATH} = \{ \langle G, U, v, k \rangle : G = (V, E) \text{ is an undirected graph,} \\ u, v \in V, \\ k \geq 0 \text{ is an integer, and} \\ \text{there exists a path from } u \text{ to } v \text{ in } G \\ \text{consisting of at most } k \text{ edges} \}$$

- **P**

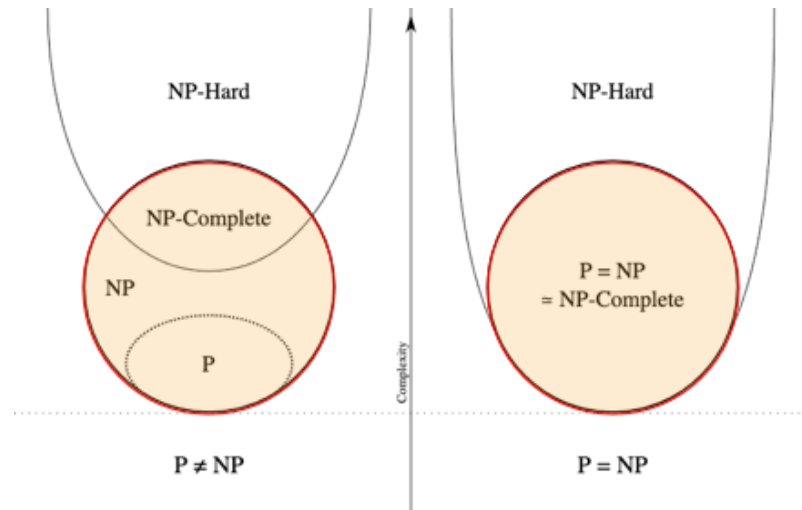
- Is set of problems that can be solved by a deterministic Turing machine in Polynomial time (i.e.  $\mathcal{O}(n^k)$ ) [2].

**Example:**

- 1) Shortest path problems
- 2) Calculating the greatest common divisor
- 3) Finding maximum bipartite matching



- **NP (Non-deterministic Polynomial):**

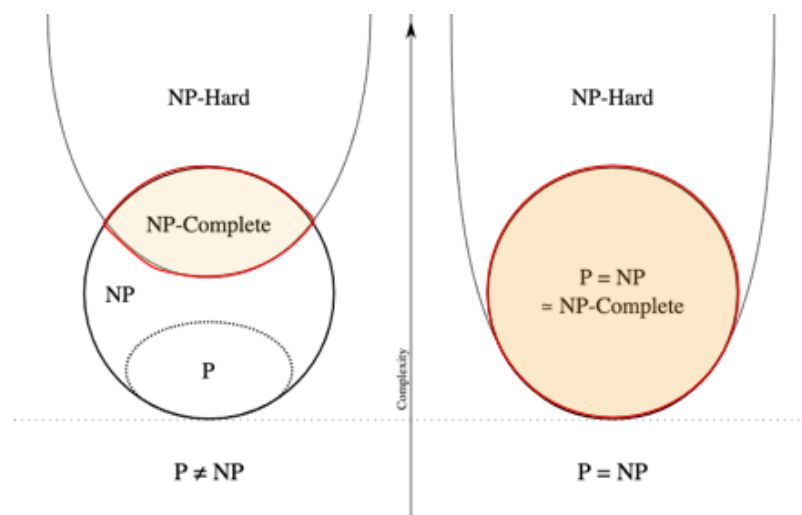


- Is set of decision problems that can be solved by a Non-deterministic Turing Machine in Polynomial time.<sup>[2]</sup>
- Has no particular rule is followed to make a guess <sup>[1]</sup>.
- Can be solved in polynomial time via a “lucky algorithm”, a magical algorithm that always make a right guess <sup>[2]</sup>
- $P \subseteq NP$

### Examples:

- Longest-path problems
- Hamiltonian Cycle
- Graph coloring

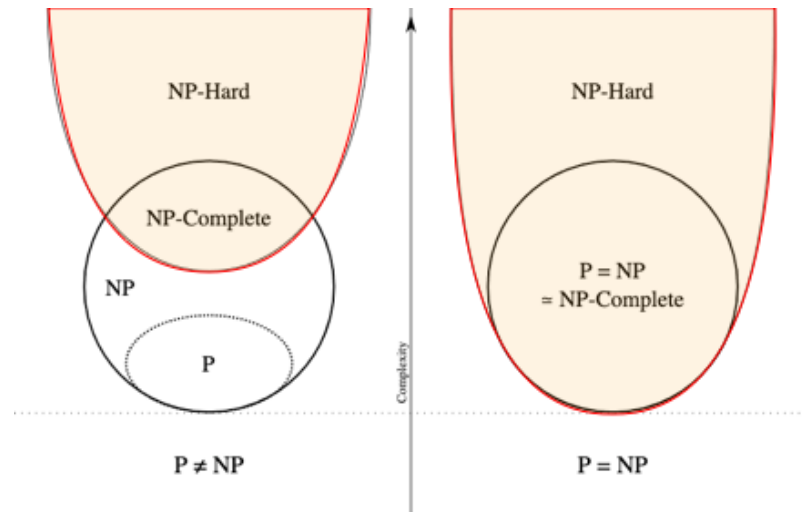
### • NP-Complete Problems:



- A decision problem A is NP-complete (NPC) if

- 1)  $A \in NP$  and
  - 2) Every (other) problems  $A'$  in NP is reducible to  $A$
- Has no efficient solution in polynomial number of steps (not yet) <sup>[3]</sup>
  - Is not likely that there is an algorithm to make it efficient <sup>[3]</sup>

• **NP-Hard:**



- A decision problem  $A$  is NP-hard if
  - 1)  $A \in NP$  (Not necessarily) and
  - 2) Every (other) problems  $A'$  in NP is reducible to  $A$
- NP-Hard means “at least as hard as any problems in NP”
- Does not have to be about decision problems

**Example:**

- 1) Alan Turing’s Halting Problem

**References**

- 1) Encyclopedia Britannica, NP-Complete Problem, [link](#)
- 2) Geeks for Geeks, NP-Completeness, [link](#)
- 3) Wikipedia, NP-complete, [link](#)
- 4) UCLA UC-Davis, ECS122A Handout on NP-Completeness, [link](#)