

# Limitations of Algorithm Power

Madiba Hudson-Quansah

# CONTENTS

## CHAPTER 1

**INTRODUCTION** \_\_\_\_\_ **PAGE 2** \_\_\_\_\_

## CHAPTER 2

***P*, *NP* AND *NP*-COMPLETE PROBLEMS** \_\_\_\_\_ **PAGE 3** \_\_\_\_\_

- 2.1 *P* and *NP* Problems \_\_\_\_\_ 3
- Non-deterministic Algorithms — 3 • *NP* Problems — 3
- 2.2 *NP*-Complete Problems \_\_\_\_\_ 4

# **Chapter 1**

## **Introduction**

# Chapter 2

## $P$ , $NP$ and $NP$ -Complete Problems

### Definition 2.0.1: Decision Problem

A problem that requires a yes or no answer.

### 2.1 $P$ and $NP$ Problems

#### Definition 2.1.1: Polynomial Time

A problem is said to be in  $P$  if there exists an algorithm that can solve the problem in polynomial time in the worst case, i.e the time complexity of the algorithm is of the form  $O(n^k)$  for some constant  $k$ . This class of problems is also called **tractable** problems.

#### Definition 2.1.2: $P$ Problems

A class of problems that can be solved in polynomial time by deterministic (non-random) algorithms. Also called **Polynomial Time** problems.

#### 2.1.1 Non-deterministic Algorithms

##### Definition 2.1.3: Non-deterministic Algorithm

An algorithm that takes in as input an instance  $I$  of a decision problem and does the following

**Guess** Generates an arbitrary string  $S$  that can be thought of as a candidate solution to the given instance  $I$ .

**Verification** Using a deterministic algorithm that takes in the candidate solution  $S$  and the problem instance  $I$  as input and outputs a boolean value indicating whether  $S$  actually represents a solution to the problem instance  $I$ .

A Non-deterministic algorithm is said to solve a decision problem if and only if for every instance  $I$  and candidate solution  $S$  it returns the correct answer, and never returns the wrong answer or a false positive, i.e. for a no instance it returns a no answer and for a yes instance it returns a yes answer.

#### 2.1.2 $NP$ Problems

##### Definition 2.1.4: $NP$ Problems

A class of decision problems that can be solved by a non-deterministic algorithm in polynomial time. The  $NP$  stands for Non-deterministic Polynomial time.

Most decision problems are in  $NP$ :

$$P \subseteq NP$$

This is true because if a problem is in  $P$ , we can use the deterministic polynomial time algorithm that solves it in the verification step of the non-deterministic algorithm bypassing the guessing step and as this algorithm used in the verification step is in polynomial time and is the only step in the non-deterministic algorithm, the entire non-deterministic algorithm for the problem instance  $I$  is in polynomial time.

$NP$  also contains problems that are not in  $P$ :

- Hamiltonian Circuit Problem
- Knapsack Problem
- Travelling Salesman Problem

This leads to the question:

$$P \stackrel{?}{=} NP$$

I.e. are all problems in  $NP$  also in  $P$  or is  $P$  a proper subset of  $NP$ ?

## 2.2 $NP$ -Complete Problems

### Definition 2.2.1: Polynomially Reducible

A decision problem  $D_1$  is said to be polynomially reducible to a decision problem  $D_2$ , if there exists a function  $t$  that transforms instances of  $D_1$  to instances of  $D_2$  such that:

1.  $t$  maps all yes instances of  $D_1$  to yes instances of  $D_2$ , and all no instances of  $D_1$  to no instances of  $D_2$ .
2.  $t$  is computable in polynomial time.

### Definition 2.2.2: $NP$ Complete

A decision problem  $D$  is  $NP$ -Complete if:

1. It belongs to the class  $NP$
2. Every problem in  $NP$  is polynomially reducible to  $D$ .