UPPSALA UNIVERSITET

Föreläsningsanteckningar

# Algoritmer & Datastrukturer

*Rami Abou Zahra*

Inlämningsdatum
September 1, 2023

Contents

# 1. Introduction

# 2. Algorithm Analysis

## 2.1. **Time complexity.**

It takes a lot longer *time* for an algorithm to solve a given problem.

- Size of the input
- Value of the input (could be the total number of bits in the 2 integers)

The *input size* depends on the problem being studied:

- The number of items in the input such as the size of the array being sorted
- Could be something else (bits)
- Could be described by more than one number (graph algorithms are expressed in terms of vertices and edges in input graph)

**Find runtime**
*Experimentation*: Run a clock, run algorithm, stop clock
Issues arise since it depends on the input, programming language, environment, etc. We need to ensure that we test on "difficult" examples

*Mathematically*: Using reasoning & logic to give an estimation of the algorithm runtinme in terms of the size of the input.
Does not depend on OS, CPU, etc.

**Does complexity matter?**
Recall the traveling salesperson problem (TSP). Assume we have an algorithm than enumerates all routes and chooses the shortest one. This is a natural occurring problem and can be adapted to a bunch of other industries.

For $n$ cities to visit, there are $n!$ possible routes.

**Performing Time Complexity Analysis**
Based on an abstract model of computation (*Random Access Machine* (mental model ofa computer))
Instructions are executed one after nother (no concurrency)
ELementary instructions can be perfomed in constant time (does not depend on the size of arguments)

- Arithmetical operations
- Assignment operations, access to array elements
- Control operations such as branching (if), loops

**Definition/Sats 2.1: Runtime of algorithm**

The runtime (running time)

$$\sum_{\text{all eLementary operations}} \text{(cost of operations)} \cdot \text{(times operations is executed)}$$