Theory of Algorithms

Introduction

Mechanics

- Outline
- Exams, Dates, Homework, Grading
- No Programming Exercises
- Required Books
- Class Handouts
- Previous Exams
- Web Sites

Why Study Algorithms

- The One Constant in A Changing Universe
- Techniques are Useful, and Required in Most other Research Areas
- Some Exposure to Theory is Necessary at the Graduate Level
- The Material is Interesting and Challenging in its Own Right

What We will Study

- Speed
  - Time Complexity
- Memory Usage
  - Space Complexity
- Usage of Communication Network
  - Communication Complexity
- Sometimes: How to do stuff

Time Complexity

- How “Fast” an Algorithm Runs
- An Algorithm is Not a Program
- Results Should Be Applicable to *All* Machines
- Results Should Apply Regardless of Input Size
- Are Some Algorithms *Inherently* Better than Others?

How to Measure Speed

- For Each Algorithm Find a Function $f(n)$
- The Argument n is the Size of the Input
- The Function $f(n)$ Gives the Amount of Time Required to Process the Input
- For Any Machine there Must be a Constant K such that $Kf(n)$ is Close to the Real Run Time on Machine M.
- K Also Depends on the Algorithm

Some Observations About $f(n)$

- Because of the Machine and Algorithm Dependent Constant K, $f(n)$ is the same as $Cf(n)$ for any Positive Constant C
- For Small Values of n, Initialization Dominates, so $f(n)$ May Be Inaccurate
- Only “Large” Values of n are “Interesting”

Theory of Algorithms

Introduction

Mechanics

- Outline
- Exams, Dates, Homework, Grading
- No Programming Exercises
- Required Books
- Class Handouts
- Previous Exams
- Web Sites

Why Study Algorithms

- The One Constant in A Changing Universe
- Techniques are Useful, and Required in Most other Research Areas
- Some Exposure to Theory is Necessary at the Graduate Level
- The Material is Interesting and Challenging in its Own Right

What We will Study

- Speed
  - Time Complexity
- Memory Usage
  - Space Complexity
- Usage of Communication Network
  - Communication Complexity
- Sometimes: How to do stuff

Time Complexity

- How “Fast” an Algorithm Runs
- An Algorithm is Not a Program
- Results Should Be Applicable to *All* Machines
- Results Should Apply Regardless of Input Size
- Are Some Algorithms *Inherently* Better than Others?

How to Measure Speed

- For Each Algorithm Find a Function $f(n)$
- The Argument n is the Size of the Input
- The Function $f(n)$ Gives the Amount of Time Required to Process the Input
- For Any Machine there Must be a Constant K such that $Kf(n)$ is Close to the Real Run Time on Machine M.
- K Also Depends on the Algorithm

Some Observations About $f(n)$

- Because of the Machine and Algorithm Dependent Constant K, $f(n)$ is the same as $Cf(n)$ for any Positive Constant C
- For Small Values of n, Initialization Dominates, so $f(n)$ May Be Inaccurate
- Only “Large” Values of n are “Interesting”