CS014 Introduction to Data Structures and Algorithms

Instructor: Ahmed Eldawy

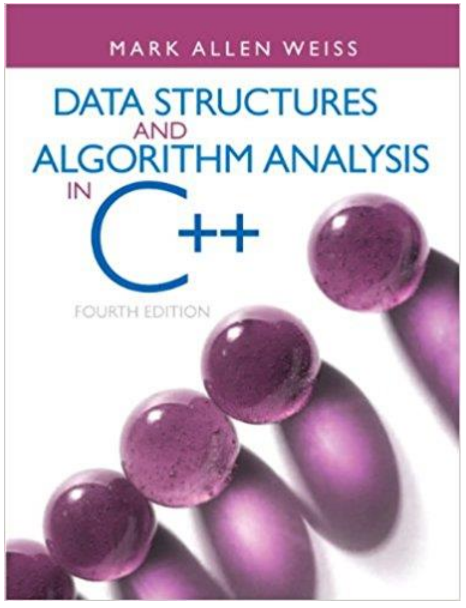
Welcome (back) to UCR!

Class information

- Classes: Tuesday and Thursday 2:10 PM – 3:30 PM at MSE 104
- Instructor: Ahmed Eldawy
- Office hours:
  Tuesday and Thursday 1:00 PM – 2:00PM @357 WCH. Conflicts?
- TAs: Saheli Ghosh, Zacharias (Harry) Chasparis, and Tin Vu

Textbook

- Data Structures and Algorithm Analysis in C++, Fourth Edition
- By Mark Allen Weiss
- http://www.facultybookshelf.org/course/13395



Course goals

- What are your goals?

- Analyze and compare algorithms
- Familiarize yourself with fundamental data structures and algorithms
- Use basic data structures to solve problems
- Develop your own data structure or algorithm

Course work

- Five assignments (10%) – Lowest grade does not count
  - Late policy: 20% for each calendar day up to four days
- Lab work (35%) – Lowest two grades do not count
- Midterm exam (15%)
- Final exam (40%)
  - Friday, December 15 11:30 a.m. - 2:30 p.m.

Experience with C++

- We will use C++ but we will not teach it in class or labs!
- We may use the following features and more:
  - Recursion
  - C++ classes (declaration, definition, constructor, destructor …)
  - Arrays (both single- and multi-dimensional)
  - Pointers (allocation, deallocation, dereference, …)
  - File manipulation and streams (read and write, random access)
  - Templates

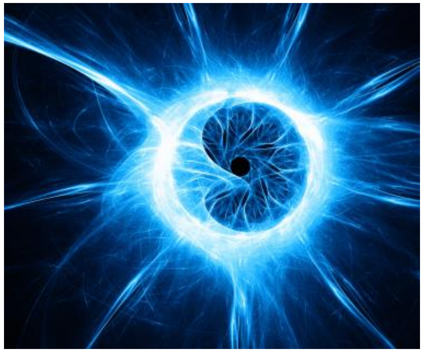
Covered topics

- Analysis of algorithms
- Abstract data types (ADT)
- Lists, stack, and queues
- Search trees
- Heaps
- Sorting algorithms
- Hash tables
- Graphs

Introduction

Performance of algorithms

- Conservation of Energy
  - “Energy can neither be created nor destroyed”
  - But it can be wasted!



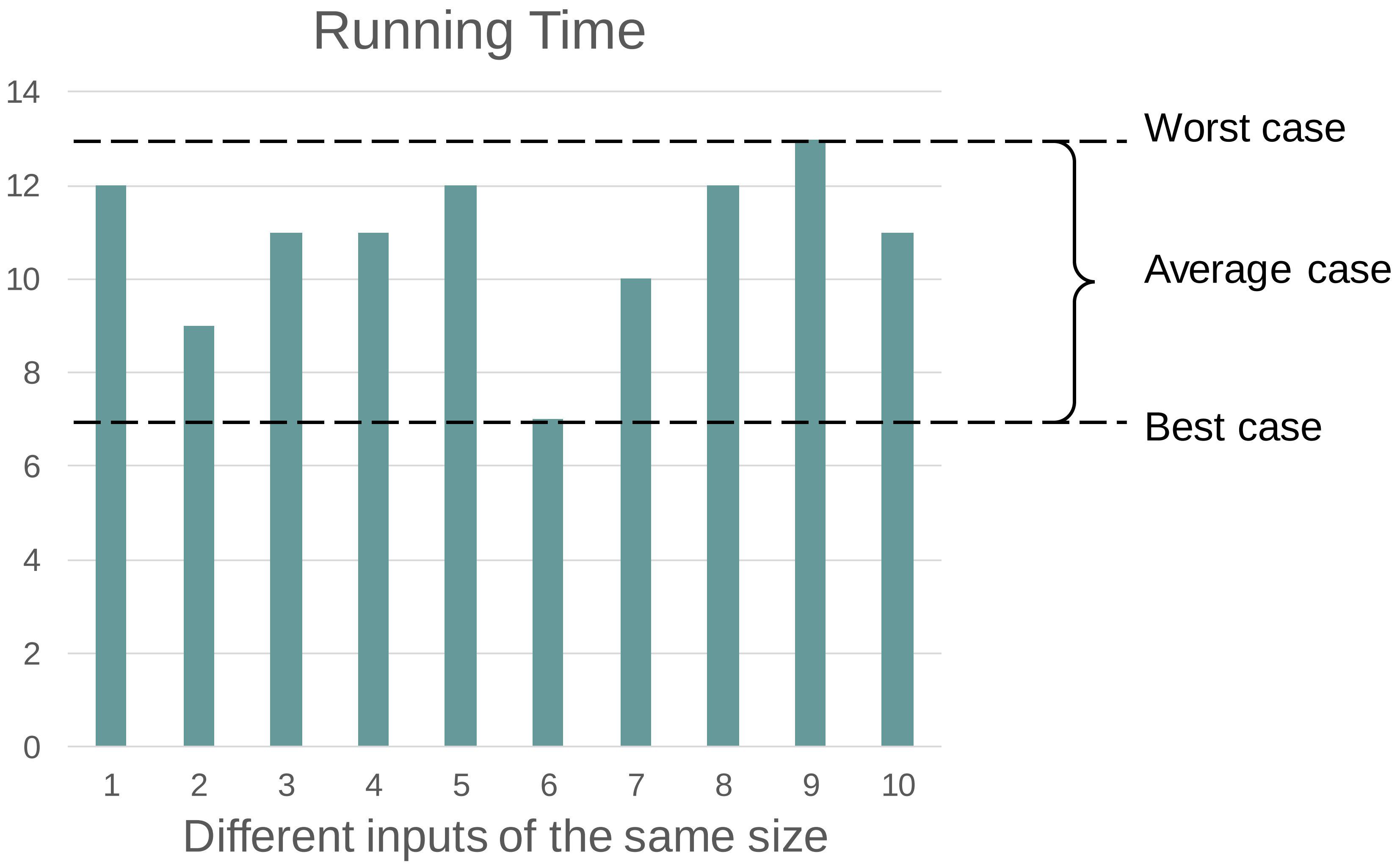
- In algorithms, you can think of energy as:
  - Running time
  - Disk IO
  - Network IO
  - …
- How to get the job done efficiently!

Criteria of Analysis

- Which criteria should be taken into account?

- Running time
- Memory footprint
- Disk IO
- Network bandwidth
- Power consumption
- Lines of codes
- …

Average Case Vs Worst Case



Case Study: Insertion Sort

INSERTION-SORT(A, n)	*cost*	*times*
for $j = 2$ **to** n	c_1	n
$\quad key = A[j]$	c_2	$n - 1$
$\quad$ // Insert $A[j]$ into the sorted sequence $A[1 \,..\, j - 1]$.	0	$n - 1$
$\quad i = j - 1$	c_4	$n - 1$
$\quad$ **while** $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
$\qquad A[i + 1] = A[i]$	c_6	$\sum_{j=2}^{n} (t_j - 1)$
$\qquad i = i - 1$	c_7	$\sum_{j=2}^{n} (t_j - 1)$
$\quad A[i + 1] = key$	c_8	$n - 1$

Running time analysis

- $T(n) = c_1 n + (c_2 + c_4 + c_8)(n-1) + c_5 \sum_{j=2}^{n} t_j + (c_6 + c_7) \sum_{j=2}^{n} t_j - 1$
- Worst case ($t_j = j$)
- $T(n) = c_1 n + (c_2 + c_4 + c_8)(n-1) + c_5 \left(\frac{n(n+1)}{2} - 1 \right) + (c_6 + c_7) \frac{n(n-1)}{2} = an^2 + bn + c$
- Best case ($t_j = 1$)
- $T(n) = c_1 n + (c_2 + c_4 + c_8)(n-1) + c_5(n-2) = \mathrm{an} + \mathrm{b}$

Growth of Functions

- It is hard to compute the actual running time
- The cost of the worst-case is a good measure
- The *growth* of the function is what interests us (Big Data)
- We are more concerned with comparing two functions, e.g., two algorithms.