Algorithms and Data Structures II (CSC 226) Fall 2014

Dr. Ulrike Stege
Associate Professor
Chair, Department of Computer Science

Some of the Learning Outcomes

- learn skills required for advanced algorithm development for algorithms and data structures studied
- should be able to apply these same or similar steps to new applicable situations.
- be able to implement, in all major programming languages, algorithms and their data structures
    - preserving time and space complexity of the designed algorithm and data structures

Textbook

Assessment

- 5 assignments [30%]
- quizzes (in lab) [5%]
- midterm [20%]
- final exam [45%]

Course Admin

- Me
  - Ulrike Stege
  - email: ustege@uvic.ca
  - phone (during office hours): 250 472 5729
  - phone (otherwise): 250 472 5704
  - Office (during office hours): ECS 624
  - Office (otherwise): ask for me at the main office (ECS 5th floor)
  - Office hours: Monday 3:30-4:30pm; Thursday 2:00-3:00pm

When you send email ….

- ustege@uvic.ca
- use “Subject: CSC 226”
- use your UVic email address
- mention your name and V#

Course Admin

- Lab Instructor
  - Bill Bird
  - email: bbird@uvic.ca
  - Office hours: Wed 1:30-3:00 (subject to change)
    check connex for location, time

Office hours

- use them!!
- get to know lab instructor/instructor
- typically more efficient communication than email

Connex

- Try to login to connex this week
- Announcements, assignments etc will all be posted at UVic
- Assignment submission will be electronically (via conned)
- Always submit as instructed (typically pdf)
- Ensure after submission you received a confirmation, otherwise your submission is not valid

This course (226)

- first time offering
- required computer science course
- What about MATH 222?
- prerequisite: CSC 225
- students in this class are from a variety of CSC 225 offerings
- LTC grant/research study

Course Content

- Sorting revisited and lower bound for comparison based sorting; linear median algorithm
- Graphs, Minimum Spanning Trees and union find, Shortest Paths: Dijkstra
- Searching: Balanced binary search trees
- String matching: Longest Common Subsequence and KMP
- Network flow: Ford/Fulkerson & Emond Karp
- Backtracking: Vertex Cover, Independent Set, FPT and planar graphs

Before we start …

Warm up …

Warm up …

- Sorting

Sorting revisited

- Overview of sorting algorithms
- A lower bound for comparison based sorting

What sorting algorithms do you know?

Different types of Sorting Algorithms

- Comparison Based Sorting
  - sorting algorithm that sorts based only on comparisons
  - elements to be sorted must satisfy total order properties
- Integer Sorting
  - sorting algorithm that sorts a collection of data values by numeric keys, each of which is an integer

Linear Insertion/Insertion Sort (Pseudocode)

```
Algorithm Insertion Sort ($a$: sequence of comparable numbers) // sorts $a$ into increasing order

 for $i$ from 1 to $a$.length() do

  for $j$ from $i$ down to the 1st encounter of an $a[k]$
  with $a[k-1] < a[k]$; $j>0$ do
  // everything from $a[0]$ up to $a[k]$ is already sorted

   if $a[j] < a[j-1]$ then swap $a[j]$ & $a[j-1]$
```

Complexity of Insertion Sort

- sorting n elements with insertion sort requires
  - $O(n^2)$ time
  - $O(n)$ space

Some Properties of Insertion Sort

- Simple implementation
- Efficient for
  - (quite) small data sets
  - data sets of already substantially sorted sequences: more efficient in practice than most other simple quadratic algorithms.
- *Stable*: doesn’t change relative order of elements with equal keys
- *In-place*: only requires a constant amount of (additional) memory
- *Online*: updates sort as elements are received
- *Natural*: people often use an insertion like sorting in practice

Bubble Sort (Pseudocode)

```
Algorithm BubbleSort ($a$: sequence of comparable numbers)
// sorts $a$ into increasing order

 repeat
   swapped ← false
   $n$ ← $a$.length()-1
   for $i$ from 1 to $n$ do
     if $a[i-1] > a[i]$ then
       swap $a[i-1]$ & $a[i]$
       swapped ← true
   $n$ ← $n$-1
 until not swapped
```

Complexity of Bubble Sort

- Your turn!
  - Time
  - Space

Properties of Bubble Sort

- Your turn!
  - Simple implementation
  - Not efficient
    - even for small data sets
  - *Stable*: ??
  - *In-place*: ??
  - *Online*: ??

Sorting out Sorting

- http://www.youtube.com/watch?feature=player_detailpage&v=SJwEwA5gOkM
- List all sorting algorithms from movie
- What are worst case, best case complexity? Space complexity? Other noteworthy properties?

	Type of Sorting Algorithm	**Worst Case # comparisons**	**Worst Case # swaps**	**Worst Case Time**	**Best Case Performance**	**Average Case Performance**	**Properties**
Insertion Sort	Comparison Based Sorting						
Bubblesort	Comparison Based Sorting						
Selection Sort	Comparison Based Sorting						
Binary Insertion	Comparison Based Sorting						
Shakersort	Comparison Based Sorting						
Tree Selection	Comparison Based Sorting						
Shellsort	Comparison Based Sorting						
Quicksort	Comparison Based Sorting						
Heapsort	Comparison Based Sorting						
Mergesort	Comparison Based Sorting						
Bucketsort	Integer Sorting						
Radixsort	Integer Sorting						