COMP3600/6466 – Algorithms

Introduction

[CLRS ch. 1, sec. 2.1, 2.2]

Hanna Kurniawati

comp_3600_6466@anu.edu.au

Australian National University | School of Computing

Topics

✓What is Algorithms?

- Setting up the stage
  - The Problems we'll focus on:
    - Sorting
    - Searching
  - Model of Computation
  - Math refresher

The Problems

- Recall: An algorithm is a well-defined procedure to solve well-specified problems
- In this class, we will focus on 2 classes of problems:
  - Sorting
  - Searching

  These problems are building blocks of almost any program you see in practice today

Sorting – Problem Specification

- Input: A sequence of n numbers $[a_1, a_2, \ldots, a_n]$
- Output: A reordering $[a_1', a_2', \ldots, a_n']$ of the input sequence such that $a_1 \leq a_2 \leq \ldots \leq a_n$ (for ascending order)

Where is sorting used?

- Often sorting is used as a building block of a bigger program. For instance:
  - Searching. Binary search can run faster once the keys are sorted. Internet search (e.g., Google) also uses sort to present the most relevant results first
  - Closest pair. Given n numbers, which two have the smallest different between them? Once the numbers are sorted, the closest pair must be next to each other. Therefore, a linear-time scan will find the solution. Closest pair problem is often encountered in various applications, including in machine learning, computational geometry, etc.

Searching – Problem Specification

- Input: A bounded set X with relation R among its elements, and a solution criteria C
- Output: An element of X, $x \in X$, that satisfies the criteria C
- Note: The above definition is very generic. In many problems, we'll be more specific. For instance:
  - What type of space is X? (e.g., is it countable or uncountable?) What type of relation exists among the elements? Different algorithms are more suitable for different types of X and R.
  - The solution criteria will also influence the type of algorithms we should use to be efficient.

Searching – A Problem Specification

- An example for a specific search problem that will be used often in this class
- Input: A sequence of n numbers $A = [a_1, a_2, \ldots, a_n]$ and a value v
- Output: An index i such that $v = A[i]$ or the special value null if v does not appear in A

Where is searching used?

- More visible than sorting. For instance:
  - Internet search, search in your email, search in a document, etc.
  - Optimization. Computation wise, optimization problem is a search problem (i.e., finding a value that is the largest/smallest). Many are search in continuous space (e.g., in real number space).

  Since almost (if not all) problems can be framed as an optimization problem, they will eventually become a search problem ☺

A bit more about problems

- Recall: An algorithm solves well-specified problems
- In reality, problems do not come in well-specified form. Someone (usually the algorithm designer) needs to formulate the problem into a well-specified problem that an algorithm can solve
- In fact, good problem formulation is usually half the solution
- In this class, esp. from tutorial questions and assignments, we'll learn how to formulate problems too
    - Yes, you'll need to do this in your assignments ☺

Today

- ✓What is Algorithms?
- Setting up the stage
  - ✓The Problems we'll focus on:
    - ✓Sorting
    - ✓Searching
  - Model of Computation
  - Math refresher

Analysing Algorithms

- Recall that analysing algorithms mean computing computational resources (typically, time and space) required to run the algorithm on a certain input
- To do the above analysis, we need to first know the machine where the algorithms are executed
  - Model of computation

Model of Computation

- In this class, we'll assume:
  - An abstract generic one-processor Random Access Machine (RAM)
    - Abstract: We don't consider memory swapping, garbage collection, etc.
    - One-processor RAM: Instructions are executed one after another (no concurrent operations)

    - To store numbers, RAM has integer and float data type. Each data type in RAM has a limit. We will introduce as necessary

Model of Computation

- Have the following primitive instructions (each primitive instruction takes constant time)

  Arithmetic: Add, subtract, multiply, divide, mod, floor, ceil

  Data movement: Load, store, copy

  Control: Conditional & unconditional branching, subroutine call and return

Analysing Algorithms Executed in a RAM: Example on Time Analysis

- Sorting problem:
  - Input: A sequence of n numbers $[a_1, a_2, \ldots, a_n]$
  - Output: A reordering $[\, a_1', a_2', \ldots, a_n']$ of the input sequence such that $a_1 \leq a_2 \leq \ldots \leq a_n$

- InsertionSort(A)

```
for j = 2 to A.length
    Key = A[j]
    i = j-1
    While i > 0 and A[i] > key
        A[i+1] = A[i]
        i = i-1
    A[i+1] = key
```

Intuitively:

- Separate the sequence of numbers / array into: Sorted and unsorted.
- At each iteration take an element from the unsorted part and put it in the right place in the sorted part of the sequence

Illustration of Insertion Sort

Sorted | Unsorted

	20	5	10	8	7	15	12	3

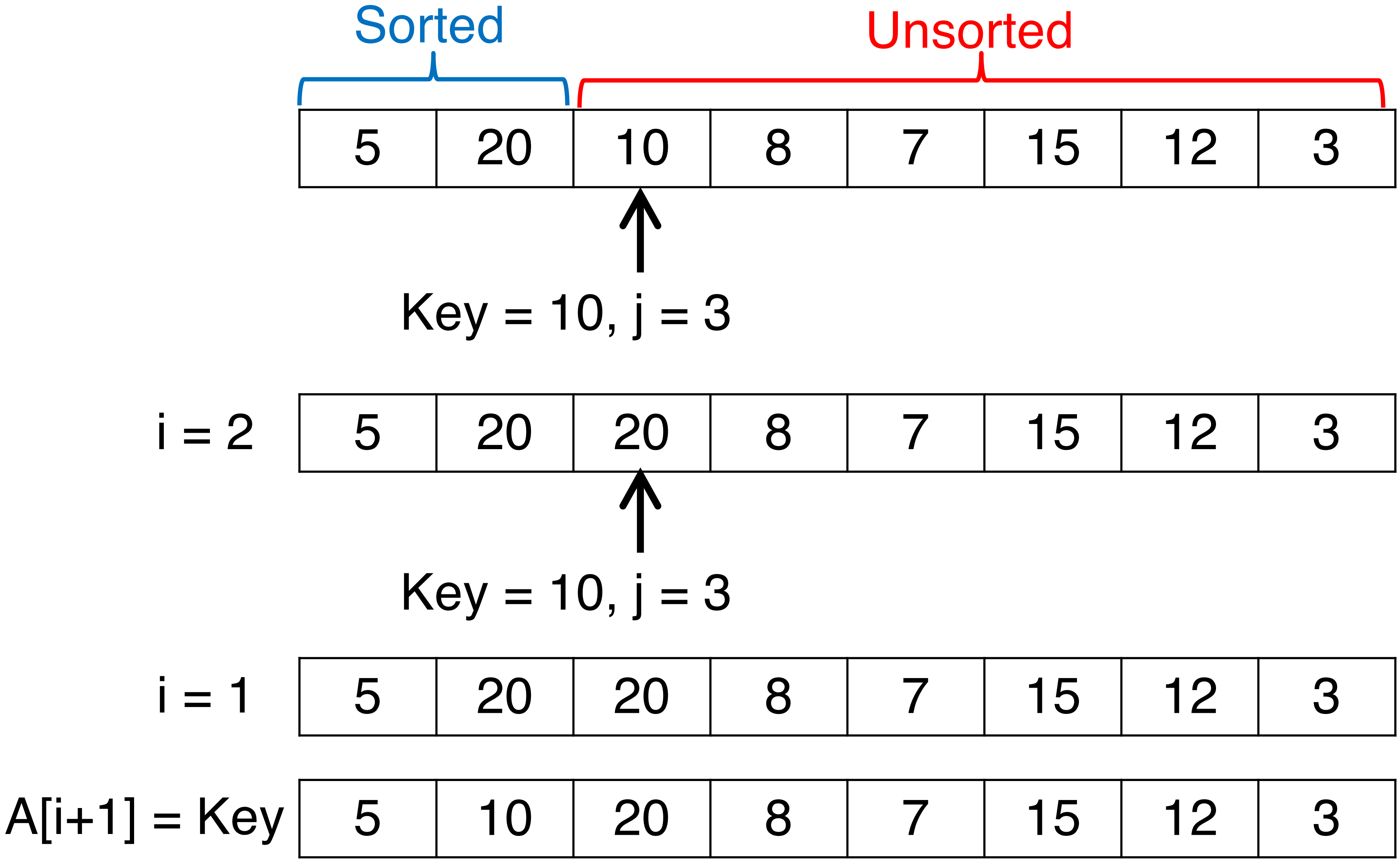
Key = 5, j = 2

i = 1	20	20	10	8	7	15	12	3

Key = 5, j = 2

i = 0	20	20	10	8	7	15	12	3
A[i+1] = Key	5	20	10	8	7	15	12	3

Illustration of Insertion Sort



Correctness? We'll discuss after Asymptotic Analysis

Back to time analysis

InsertionSort		Cost	Times
1	for j = 2 to A.length		
2	Key = A[j]		
3	i = j-1		
4	While i > 0 and A[i] > key		
5	A[i+1] = A[i]		
6	i = i-1		
7	A[i+1] = key		

Total time $T(n)$: sum of cost X times

Today

- ✓What is Algorithms?
- Setting up the stage
  - ✓The Problems we'll focus on:
    - ✓Sorting
    - ✓Searching
  - ✓Model of Computation
  - Math refresher

Important Summation Formulas

1. $\sum_{i=l}^{u} 1 = \underbrace{1+1+\cdots+1}_{u-l+1 \text{ times}} = u - l + 1$ (l, u are integer limits, $l \le u$); $\quad \sum_{i=1}^{n} 1 = n$

2. $\sum_{i=1}^{n} i = 1 + 2 + \cdots + n = \frac{n(n+1)}{2} \approx \frac{1}{2}n^2$

3. $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6} \approx \frac{1}{3}n^3$

4. $\sum_{i=1}^{n} i^k = 1^k + 2^k + \cdots + n^k \approx \frac{1}{k+1}n^{k+1}$

5. $\sum_{i=0}^{n} a^i = 1 + a + \cdots + a^n = \frac{a^{n+1}-1}{a-1}$ $(a \ne 1)$; $\quad \sum_{i=0}^{n} 2^i = 2^{n+1} - 1$

6. $\sum_{i=1}^{n} i2^i = 1 \cdot 2 + 2 \cdot 2^2 + \cdots + n2^n = (n-1)2^{n+1} + 2$

7. $\sum_{i=1}^{n} \frac{1}{i} = 1 + \frac{1}{2} + \cdots + \frac{1}{n} \approx \ln n + \gamma$, where $\gamma \approx 0.5772\ldots$ (Euler's constant)

8. $\sum_{i=1}^{n} \lg i \approx n \lg n$

Source: [Lev] Appendix A

Properties of logarithms

1. $\log_a 1 = 0$
2. $\log_a a = 1$
3. $\log_a x^y = y \log_a x$
4. $\log_a xy = \log_a x + \log_a y$
5. $\log_a \frac{x}{y} = \log_a x - \log_a y$
6. $\boxed{a^{\log_b x} = x^{\log_b a}}$
7. $\log_a x = \frac{\log_b x}{\log_b a} = \log_a b \log_b x$

In this class, unless otherwise stated, the base of the log is 2, i.e.: $\log x = \log_2 x$

A function where the exponent is log can be transformed into a polynomial function

Source: [Lev] Appendix A

Examples

- Can we transform $2^{3 \log \sqrt{n}}$ into a polynomial form?

- Can we transform $n^{\sqrt{n}}$ into the form of a power of 2?

You need to be comfortable with the basic math (basic transformations as in previous slides, basic arithmetic and algebra + basic calculus + basic probability).

If not, please catch up **NOW**!!!

To catch up, see the notes of Week-1 in Wattle

Today

- ✓What is Algorithms?
- ✓Setting up the stage
  - ✓The Problems we'll focus on:
    - ✓Sorting
    - ✓Searching
  - ✓Model of Computation
  - ✓Math refresher

Next: Asymptotic Analysis