WIA2005: Algorithm Design and Analysis

Semester 2, Session 2016/17

Lecture 2: Sorting Algorithm (Part 1)

Learning objectives

- Know what is:
  - Bubble sort
  - Counting sort
  - Radix sort
  - Bucket sort
  - Shell sort

Why sorting?

Many computer scientists consider sorting to be the most fundamental problem in the study of algorithms for the following reasons:

- Sometimes an application inherently needs to sort information.
- Algorithms often use sorting as a key subroutine.
- Based from among a wide variety of sorting algorithms, and they employ a rich set of techniques.

Why sorting?

- Many engineering issues come to the fore when implementing sorting algorithms.
- The fastest sorting program for a particular situation may depend on many factors, such as prior knowledge about the keys and satellite data, the memory hierarchy (caches and virtual memory) of the host computer, and the software environment.
- Many of these issues are best dealt with at the algorithmic level, rather than by "tweaking" the code.

Bubble sort

- Bubble sort is a popular, but inefficient, sorting algorithm. It works by repeatedly swapping adjacent elements that are out of order.

```
BUBBLESORT($A$)
for $i = 1$ to $A.length - 1$
    for $j = A.length$ downto $i + 1$
        if $A[j] < A[j-1]$
            exchange $A[j]$ with $A[j-1]$
```

Time complexity..

- What is the time complexity for Bubble sort?

Counting sort

- ***Counting sort*** algorithm sort elements based on numeric keys between a specific range.
- No comparison is done during sorting.
- Used as subroutine in other sorting algorithm.

```
COUNTING-SORT($A, B, k$)
let $C[0..k]$ be a new array
for $i = 0$ to $k$
    $C[i] = 0$
for $j = 1$ to $A.length$
    $C[A[j]] = C[A[j]] + 1$
// $C[i]$ now contains the number of elements equal to $i$.
for $i = 1$ to $k$
    $C[i] = C[i] + C[i-1]$
// $C[i]$ now contains the number of elements less than or equal to $i$.
for $j = A.length$ downto 1
    $B[C[A[j]]] = A[j]$
    $C[A[j]] = C[A[j]] - 1$
```

Time complexity..

- What is the time complexity for Counting sort?

Radix sort

- ***Radix sort*** is the algorithm used by the card-sorting machines you now find only in computer museums.
- Radix sort can sort n integers in base k with at most d digits.
- It does this by using counting sort to sort the n integers by digits, starting from the least significant digit (i.e. ones digit for integers) to the most significant digit.

```
RADIX-SORT(A, d)
1  for i = 1 to d
2      use a stable sort to sort array A on digit i
```

Time complexity..

- What is the time complexity for Radix sort?

Bucket sort

- ***Bucket sort*** algorithm creates buckets and put elements into them.
- Then using some sorting algorithm (e.g. Insertion sort) to sort elements in each bucket.
- Then the elements are taken out and joined to get the sorted result.

```
BUCKET-SORT(A)
1  let $B[0..n-1]$ be a new array
2  $n = A.length$
3  for $i = 0$ to $n-1$
4      make $B[i]$ an empty list
5  for $i = 1$ to $n$
6      insert $A[i]$ into list $B[\lfloor n A[i] \rfloor]$
7  for $i = 0$ to $n-1$
8      sort list $B[i]$ with insertion sort
9  concatenate the lists $B[0], B[1], \dots, B[n-1]$ together in order
```

Time complexity..

- What is the time complexity of Bucket sort?

Shell sort

- A generalization of the Insertion sort
  - sorting by comparing elements that are distant apart rather than adjacent.
- If we start comparing N elements that are at certain distant apart
  - value gap < N.
- In each pass, the value of gap is reduced until the lass pass where gap = 1.
- In the last pass, the sort is like insertion sort.

```
SHELL-SORT(A,n)
// we take gap sequence in order of |N/2|, |N/4|, |N/8|...1
    for gap=n/2; gap=0; gap/=2 do:
    //Perform gapped insertion sort for this gap size.

        for i=gap; i<n; i+=1 do: temp=A[i]

        // shift earlier gap-sorted elements up until
        // the correct location for a[i] is found
            for j=i; j>=gap && A[j-gap]>temp;j-=gap
                do:
                    A[j]= A[j-gap]
            end for
            // put temp in its correct location
            A[j]= temp;
        end for
    end for
end func
```

Time complexity..

- What is the time complexity of Bucket sort?

Choosing a sorting algorithm..

- Please refer to additional notes – “Choosing the Right Algorithm” by Lagoudakis, Littman and Parr.

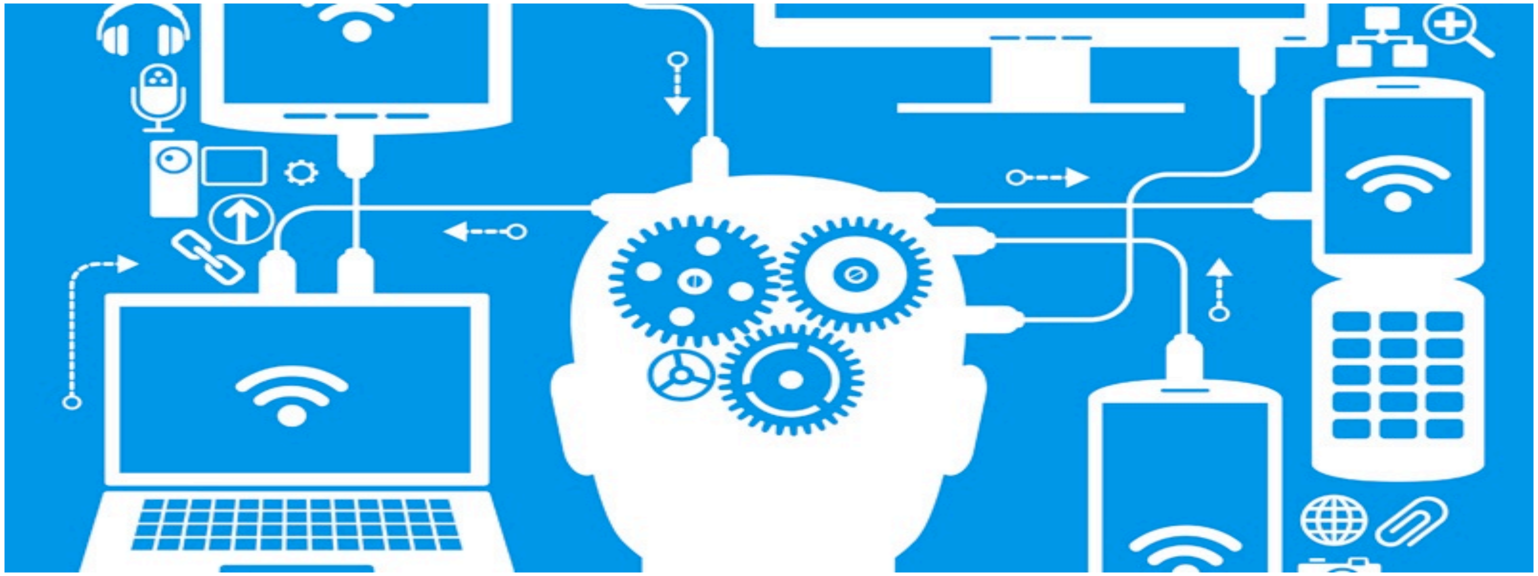
Key points

- There are several sorting sort that is used for solving many computational problem.
- Each has different time complexity.

References

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. 2009. Introduction to Algorithms, 3rd edition. MIT Press.
- Robert Sedgewick and Kevin Wayne. 2011. Algorithm. 5th Edition. Addison-Wesley.

In the next lecture..



Lecture 2: Divide and Conquer