Sorting Algorithms

- SelectionSort
- InsertionSort
- BubbleSort
- ShakerSort
- QuickSort
- MergeSort
- HeapSort
- Bucket & Radix Sort
- Counting Sort

Upper and Lower Bounds

- Define an upper bound on the time complexity of a problem. The upper bound on the time complexity of a problem is $T(n)$ if $\exists$ an algorithm that solves the problem with time complexity $O(T(n))$.
- Clearly upper bound on the time complexity for sorting is $O(n \log n)$.
- Define a lower bound on the time complexity of a problem. The lower bound on the time complexity of a problem is $T(n)$ if $\forall$ algorithms that solve the problem, their time complexity is $\Omega(T(n))$.
- It can be proved that the upper bound is tight! In other words, it can be mathematically proved that the lower bound for sorting is $\Omega(n \log n)$.

Bucket Sort

- N values in the range [a..a+m-1]
- For e.g., sort a list of 50 scores in the range [0..9].
- Algorithm
  - Make m buckets [a..a+m-1]
  - As you read elements throw into appropriate bucket
  - Output contents of buckets [0..m] in that order
- Time O(N+m)

Stable Sort

- A sort is stable if equal elements appear in the same order in both the input and the output.
- Which sorts are stable? Homework!

Radix Sort

3	5	9		3	5	9		3	3	6		3	3	6
3	5	7		3	5	7		3	5	9		3	5	1
3	5	1		3	5	1		3	5	7		3	5	5
7	3	9	→	3	3	6	→	3	5	1	→	3	5	7
3	3	6		3	5	5		3	5	5		3	5	9
7	2	0		7	3	9		7	2	0		7	2	0
3	5	5		7	2	0		7	3	9		8	3	9

Algorithm

for i = 1 **to** d **do**

sort array A on digit i using any sorting algorithm

Time Complexity: $O((N+m) + (N+m^2) + \ldots + (N+m^d))$

Space Complexity: $O(m^d)$

Radix Sort

3 2 9		7 2 0		7 2 0		3 2 9
4 5 7		3 5 5		3 2 9		3 5 5
6 5 7		4 3 6		4 3 6		4 3 6
8 3 9	→	4 5 7	→	8 3 9	→	4 5 7
4 3 6		6 5 7		3 5 5		6 5 7
7 2 0		3 2 9		4 5 7		7 2 0
3 5 5		8 3 9		6 5 7		8 3 9

Algorithm

for i = 1 **to** d **do**

  sort array A on digit i using a stable sort algorithm

Time Complexity: O((n+m)d)

Counting Sort

Initial Array

1	2	3	4	5	6	7	8
2	5	3	0	2	3	0	3

Counts

0	1	2	3	4	5
2	0	2	3	0	1

Cumulative Counts

0	1	2	3	4	5
2	2	4	7	7	8

External Sorting Methods

- Assumptions:
  - data is too large to be held in main memory;
  - data is read or written in blocks;
  - 1 or more external devices available for sorting
- Sorting in main memory is cheap or free
- Read/write costs are the dominant cost
- Wide variety of storage types and costs
- No single strategy works for all cases

External Merge Sort

- Initial distribution pass
- Several multi-way merging passes

```
ASORTINGANDMERGINGEXAMPLEWITHFORTYFIVERECORDS.$
=================================================================
AOS.DMN.AEX.FHT.ERV.$
IRT.EGR.LMP.ORT.CEO.$
AGN.GIN.EIW.FIY.DRS.$
=================================================================
AAGINORST.FFHIORTTY.$
DEGGIMNNR.CDEEORRSV.$
AEEILMPWX.$
=================================================================
AAADEEEGGGIIILMMNNNOPRRSTWX.$
CDEEFFHIOORRRSTTVY.$
=================================================================
AAACDDEEEEEEFFGGGHIIIILMMNNNOOOPRRRRRSSTTTWXY.$
=================================================================
```

With 2P external devices
Space for M records in main memory
Sorting N records needs
$1 + \log_P(N/M)$ passes

Order Statistics

- Maximum, Minimum n-1 comparisons

7	3	1	9	4	8	2	5	0	6

- MinMax
  - 2(n-1) comparisons
  - 3n/2 comparisons
- Max and 2ndMax
  - (n-1) + (n-2) comparisons
  - ???