CSE221

Lecture 21:
Sorting

Acknowledgment: The content of this file is based on the slides of the textbook as well as the slides provided in former lectures at UNIST.

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

Search Records using a Key

- Sequential search in a[1:n]
  - Search successful at a[i] : i comparisons
  - Search unsuccessful : n comparisons
  - Average comparison : $(\sum_{1 \le i \le n} i) / n = (n+1)/2$
  - O(n)
- Searching in an ordered list
  - Binary search : O(log n)

we need sorting!

Insertion Sort

- Assume a[1:i] is ordered
- Insert a new record into the list to get a new ordered list a[1:i+1]

j	[1]	[2]	[3]	[4]	[5]
-	**5**	4	3	2	1
2	**4**	**5**	3	2	1
3	**3**	**4**	**5**	2	1
4	**2**	**3**	**4**	**5**	1
5	**1**	**2**	**3**	**4**	**5**

Elements in red are already sorted

Insertion Sort

```
void InsertionSort(*a, n)
{
    for(j=2;j<=n;j++)
    {
        temp = a[j];
        Insert(temp, a, j-1);
    }
}
```

```
void Insert(e, *a, i)
{
    a[0]=e;
    while(e<a[i])
    {
        a[i+1]=a[i];
        i--;
    }
    a[i+1]=e;
}
```

insert e to sorted list a[1:i] to make a[1:i+1]

		List L	*Priority Queue P*
Input		(7,4,8,2,5,3,9)	()
Phase 1	(a)	(4,8,2,5,3,9)	(7)
	(b)	(8,2,5,3,9)	(4,7)
	(c)	(2,5,3,9)	(4,7,8)
	(d)	(5,3,9)	(2,4,7,8)
	(e)	(3,9)	(2,4,5,7,8)
	(f)	(9)	(2,3,4,5,7,8)
	(g)	()	(2,3,4,5,7,8,9)

Insertion Sort

- Analysis
  - Worst case: i+1 comparison for Insert(e,a,i)

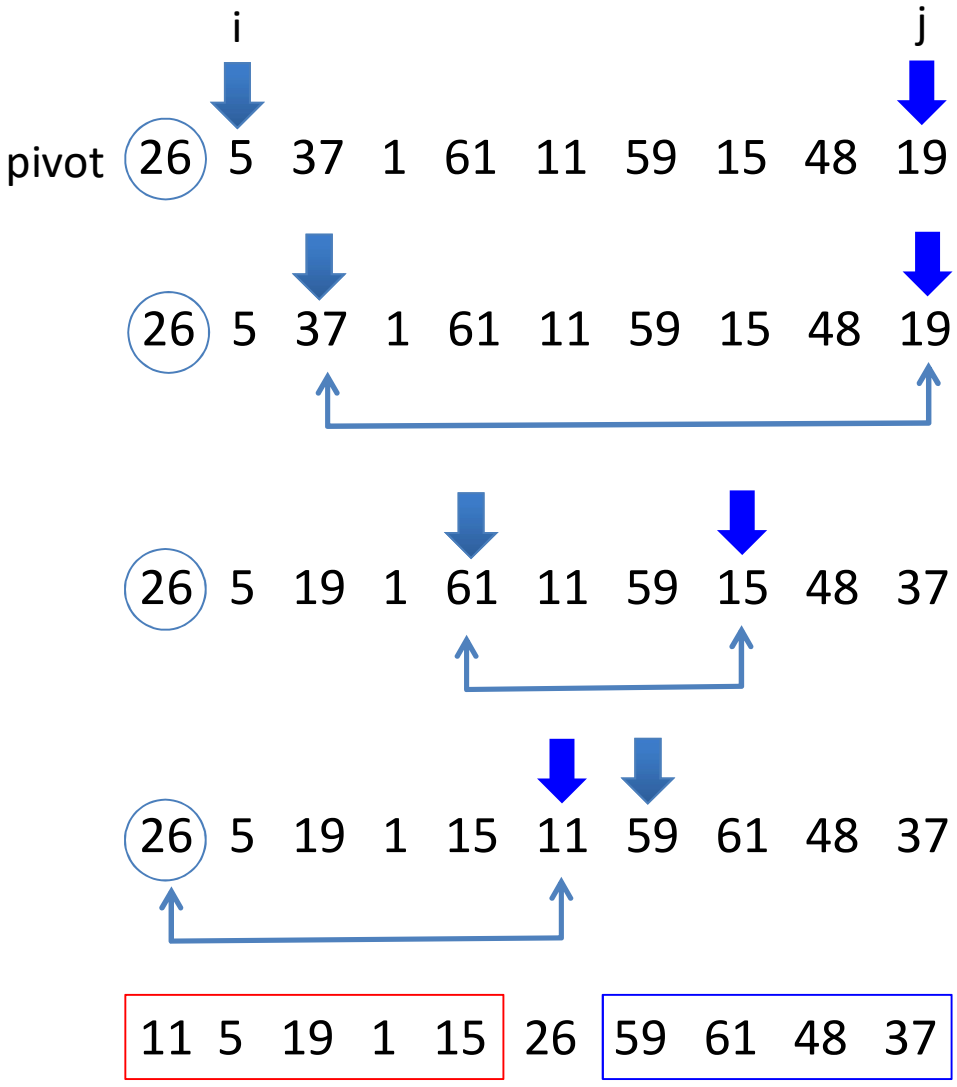
$$O(\sum_{i=1}^{n-1}(i+1)) = O(n^2)$$

  - Best case : O(n)
  - Average case : $O(n^2)$
  - Additional space requirement: O(1)
    - In-place swapping a pair of records

Quick Sort

- Partition the list into three sublists using pivot
  - Left, middle, right
  - Middle = pivot
  - Left <= pivot
  - Right >= pivot
- Recursively sort left and right sublists

Quick Sort



ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

Quick Sort

R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}
[26	5	37	1	61	11	59	15	48	19]
[11	5	19	1	15]	26	[59	61	48	37]
[1	5]	11	[19	15]	26	[59	61	48	37
1	5	11	[19	15]	26	[59	61	48	37]
1	5	11	15	19	26	[59	61	48	37]
1	5	11	15	19	26	[48	37]	59	[61]
1	5	11	15	19	26	37	48	59	[61]
1	5	11	15	19	26	37	48	59	61

Quick Sort

```
void QuickSort(*a, left, right)
{
    if(left<right) {
        int i = left, j=right+1, pivot=a[left];
        do {
            do i++; while(a[i]<pivot); // move i to right
            do j--; while(a[j]>pivot); // move j to left
            if(i<j) swap(a[i],a[j]);
        } while(i<j)
        swap(a[left],a[j]);
        QuickSort(a,left,j-1);
        QuickSort(a,j+1,right);
    }
}
```

Quick Sort

- Analysis
  - O(n) to partition a list with n records
  - Worst : $O(n^2)$
    - Either left or right sublist is empty
  - Best : O(n log n)
    - Left and right sublists are roughly same size
  - Average : O(n log n)
  - Additional space requirement : O(n) for stack

Questions?