Sorting Algorithms

• Number of Comparisons
• Number of Data Movements
• Additional Space Requirements
Sorting Algorithms

- SelectionSort
- InsertionSort
- BubbleSort
- ShakerSort
- QuickSort
- MergeSort
- HeapSort
- Bucket & Radix Sort
- Counting Sort
Visualizing Algorithms 1
Visualizing Algorithms 2

Position

Value

Unsorted

Sorted
Visualizing Comparisons 3
Animation Demos

http://cg.scs.carleton.ca/~morin/misc/sortalg/
Comparing $O(n^2)$ Sorting Algorithms

- InsertionSort and SelectionSort (and ShakerSort) are roughly twice as fast as BubbleSort for small files.
- InsertionSort is the best for very small files.
- $O(n^2)$ sorting algorithms are **NOT** useful for large random files.
- If *comparisons* are very expensive, then among the $O(n^2)$ sorting algorithms, insertion sort is best.
- If *data movements* are very expensive, then among the $O(n^2)$ sorting algorithms, ?? is best.
Figure 2.4  The operation of merge sort on the array $A = \langle 5, 2, 4, 7, 1, 3, 2, 6 \rangle$. The lengths of the sorted sequences being merged increase as the algorithm progresses from bottom to top.
Figure 2.3 The operation of lines 10–17 in the call `MERGE(A, 9, 12, 16)`, when the subarray \( A[9..16] \) contains the sequence \( (2, 4, 5, 7, 1, 2, 3, 6) \). After copying and inserting sentinels, the array \( L \) contains \( (2, 4, 5, 7, \infty) \), and the array \( R \) contains \( (1, 2, 3, 6, \infty) \). Lightly shaded positions in \( A \) contain their final values, and lightly shaded positions in \( L \) and \( R \) contain values that have yet to be copied back into \( A \). Taken together, the lightly shaded positions always comprise the values originally in \( A[9..16] \), along with the two sentinels. Heavily shaded positions in \( A \) contain values that will be copied over, and heavily shaded positions in \( L \) and \( R \) contain values that have already been copied back into \( A \). (a)–(h) The arrays \( A, L, \) and \( R, \) and their respective indices \( k, i, \) and \( j \) prior to each iteration of the loop of lines 12–17. (i) The arrays and indices at termination. At this point, the subarray in \( A[9..16] \) is sorted, and the two sentinels in \( L \) and \( R \) are the only two elements in these arrays that have not been copied into \( A \).
Assumption: Array A is sorted from positions p to q and also from positions q+1 to r.

```
MERGE(A, p, q, r)
 1  n_1 ← q − p + 1
 2  n_2 ← r − q
 3  create arrays L[1..n_1 + 1] and R[1..n_2 + 1]
 4  for i ← 1 to n_1
 5      do L[i] ← A[p + i − 1]
 6  for j ← 1 to n_2
 7      do R[j] ← A[q + j]
 8  L[n_1 + 1] ← ∞
 9  R[n_2 + 1] ← ∞
10  i ← 1
11  j ← 1
12  for k ← p to r
13      do if L[i] ≤ R[j]
14         then A[k] ← L[i]
15            i ← i + 1
16         else A[k] ← R[j]
17            j ← j + 1
```
\textbf{MERGE-SORT}(A, p, r)

1 \textbf{if} \ p < r

2 \quad \textbf{then} \ q \leftarrow \lfloor (p + r)/2 \rfloor

3 \quad \textbf{MERGE-SORT}(A, p, q)

4 \quad \textbf{MERGE-SORT}(A, q + 1, r)

5 \quad \textbf{MERGE}(A, p, q, r)
Figure 2.5 The construction of a recursion tree for the recurrence $T(n) = 2T(n/2) + cn$. Part (a) shows $T(n)$, which is progressively expanded in (b)–(d) to form the recursion tree. The fully expanded tree in part (d) has $\lg n + 1$ levels (i.e., it has height $\lg n$, as indicated), and each level contributes a total cost of $cn$. The total cost, therefore, is $cn \lg n + cn$, which is $\Theta(n \lg n)$.
Merge: Algorithm Invariants

• Merge (many lists)
  - ??
Animation Demis

http://cg.scs.carleton.ca/~morin/misc/sortalg/
Figure 8.10  Quicksort
Figure A  If 6 is used as pivot, the end result after partitioning is as shown in the Figure B.

Figure B  Result after Partitioning

2 1 4 5 0 3 6 8 7 9
QuickSort

\[
\text{QuickSort}(\text{array } A, \text{ int } p, \text{ int } r)
\]

1. If \((p < r)\)
2. Then \(q \leftarrow \text{Partition}(A, p, r)\)
3. QuickSort\((A, p, q - 1)\)
4. QuickSort\((A, q + 1, r)\)

To sort array call \textbf{QuickSort}\((A, 1, \text{length}[A])\).

Partition

\[
\text{Partition}(\text{array } A, \text{ int } p, \text{ int } r)
\]

1. \(x \leftarrow A[r]\) \quad \triangleright \text{Choose pivot}
2. \(i \leftarrow p - 1\)
3. For \(j \leftarrow p \text{ to } r - 1\)
4. Do if \((A[j] \leq x)\)
5. Then \(i \leftarrow i + 1\)
7. Exchange \(A[i + 1] \leftrightarrow A[r]\)
8. Return \(i + 1\)

Page 146, CLRS
Problems to think about!

• What is the least number of comparisons you need to sort a list of 3 elements? 4 elements? 5 elements?

• How to arrange a tennis tournament in order to find the tournament champion with the least number of matches? How many tennis matches are needed?