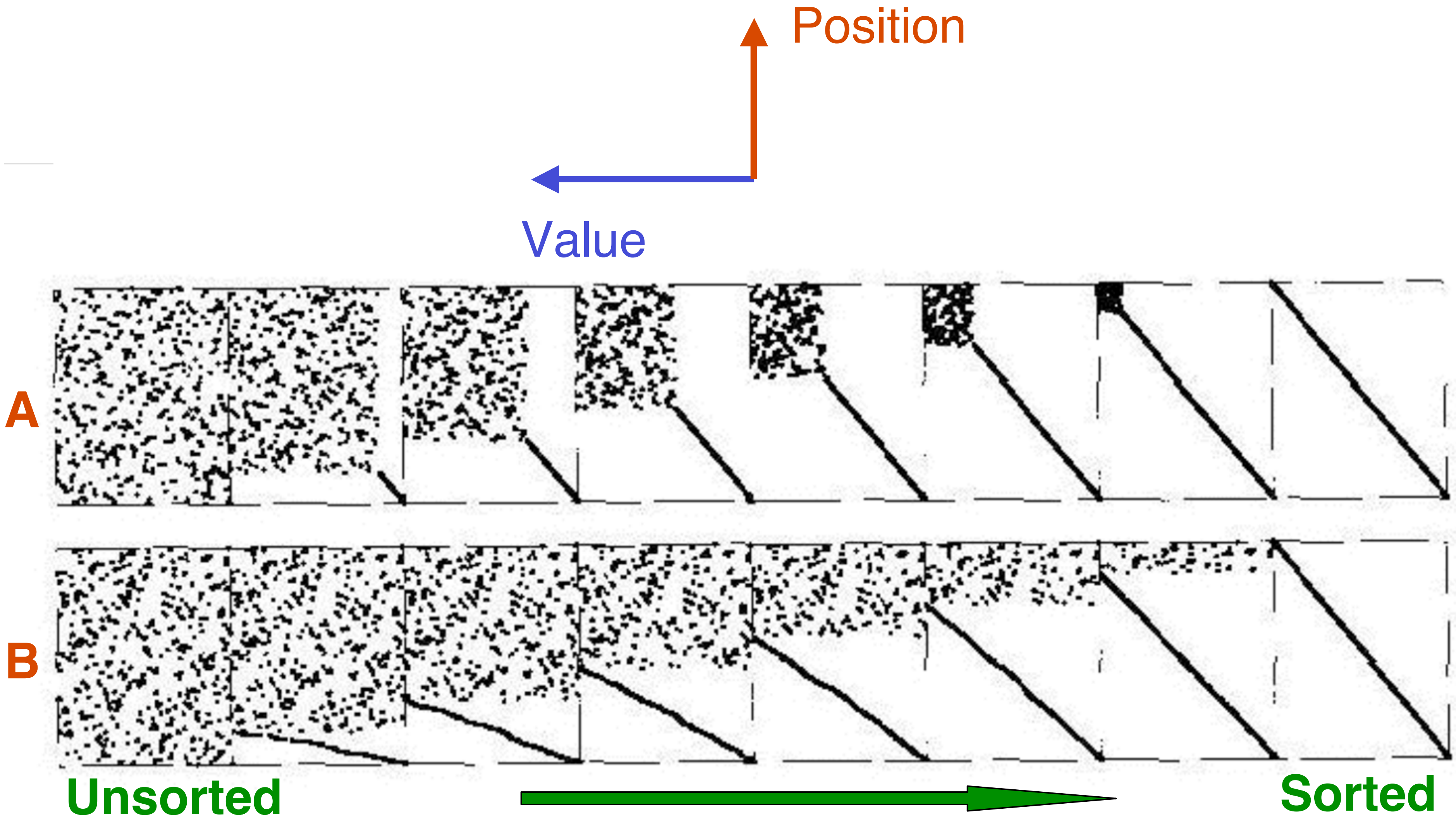
Sorting Algorithms

- Number of Comparisons
- Number of Data Movements
- Additional Space Requirements

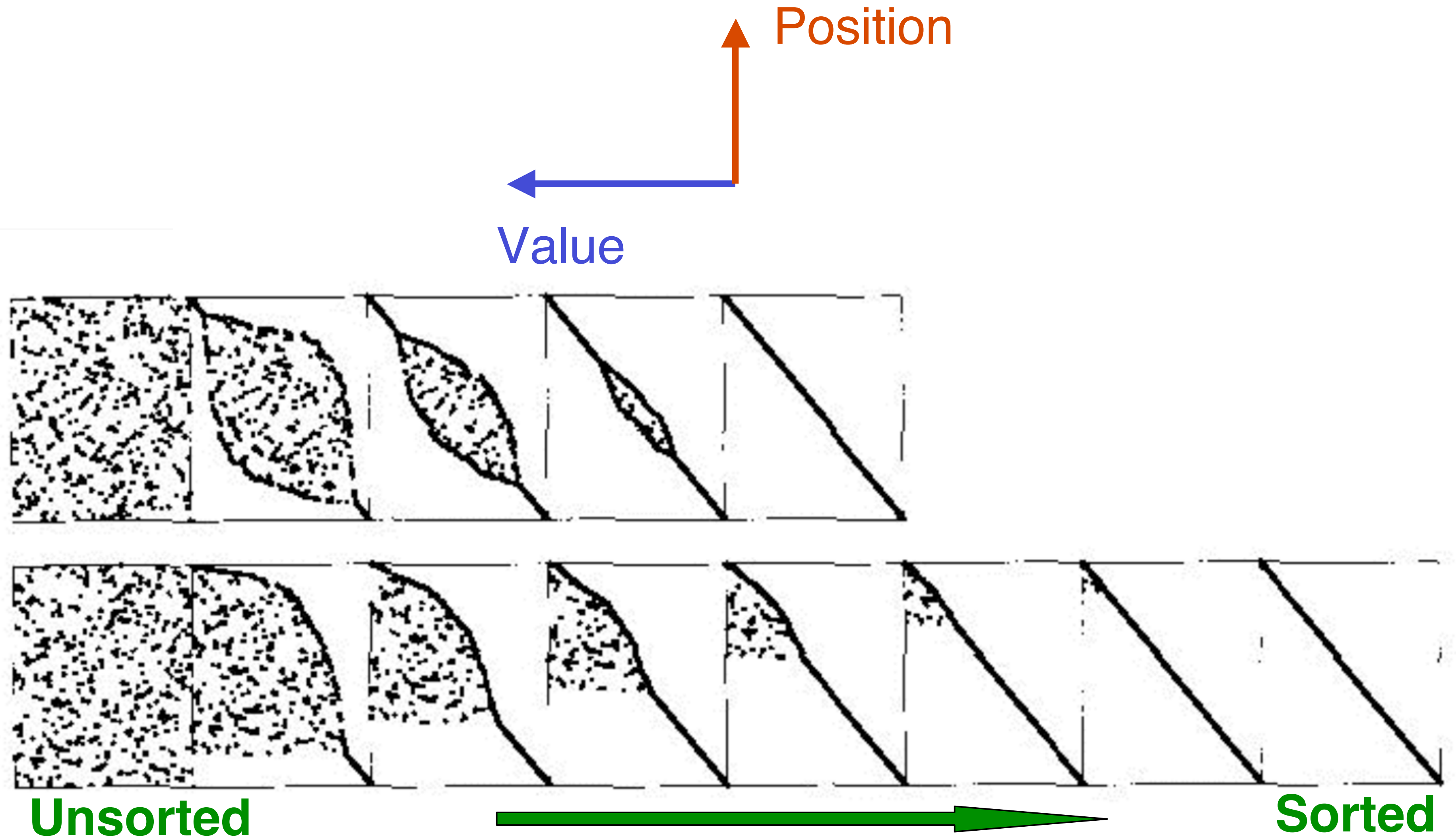
Sorting Algorithms

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

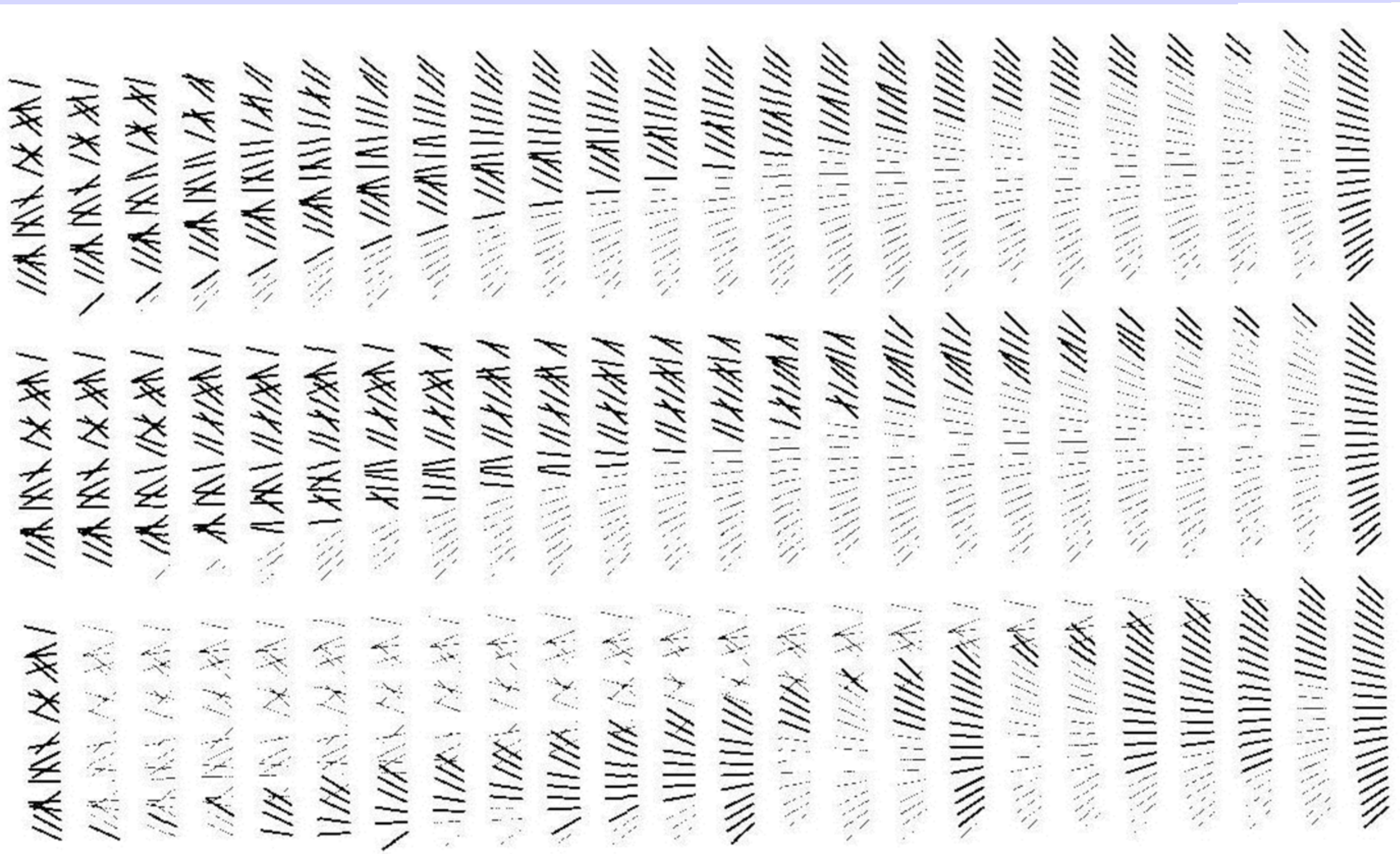
Visualizing Algorithms 1



Visualizing Algorithms 2



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, insertionsort is best.
- If data movements are very expensive, then among the $O(n^2)$ sorting algorithms, ?? is best.

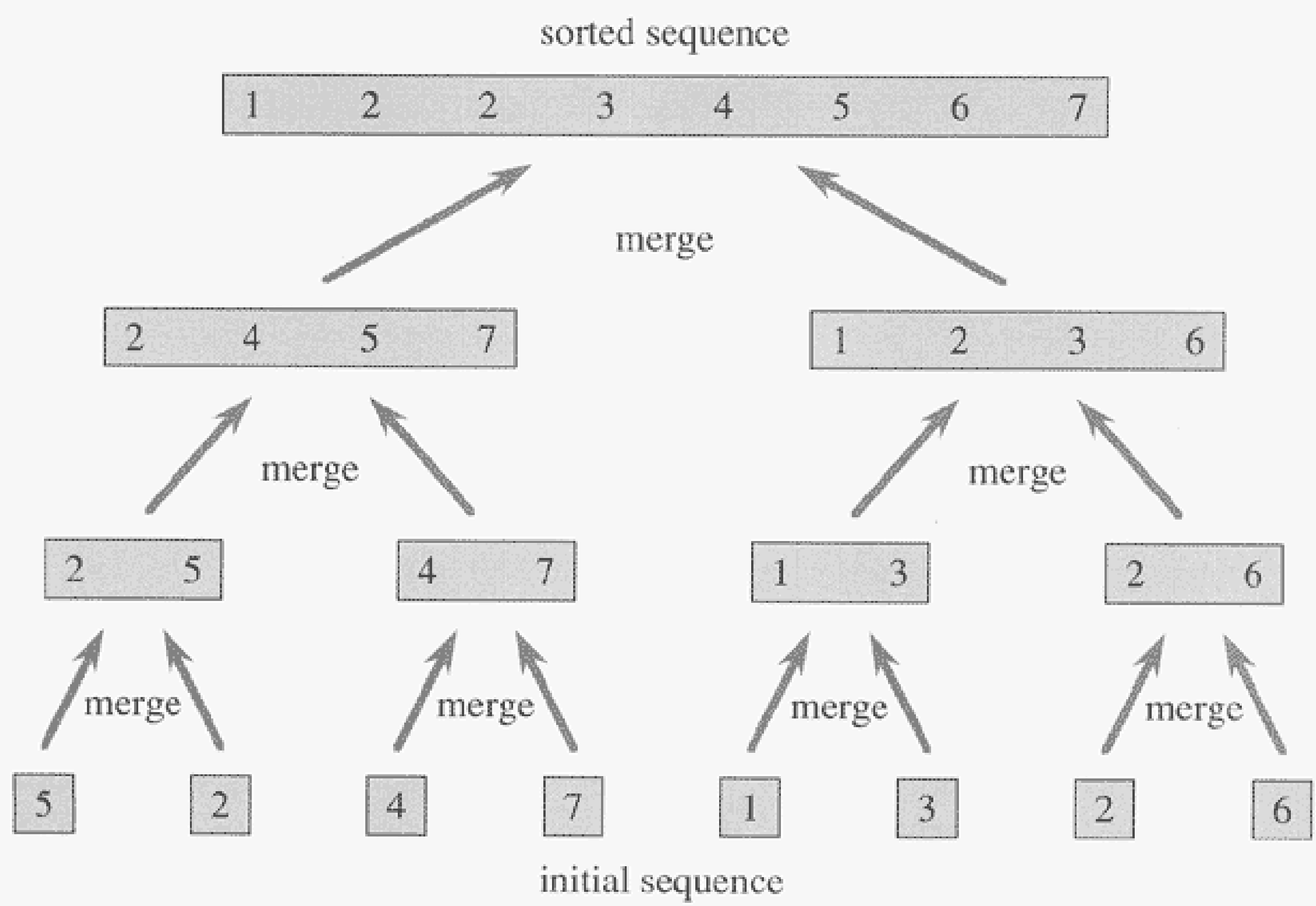
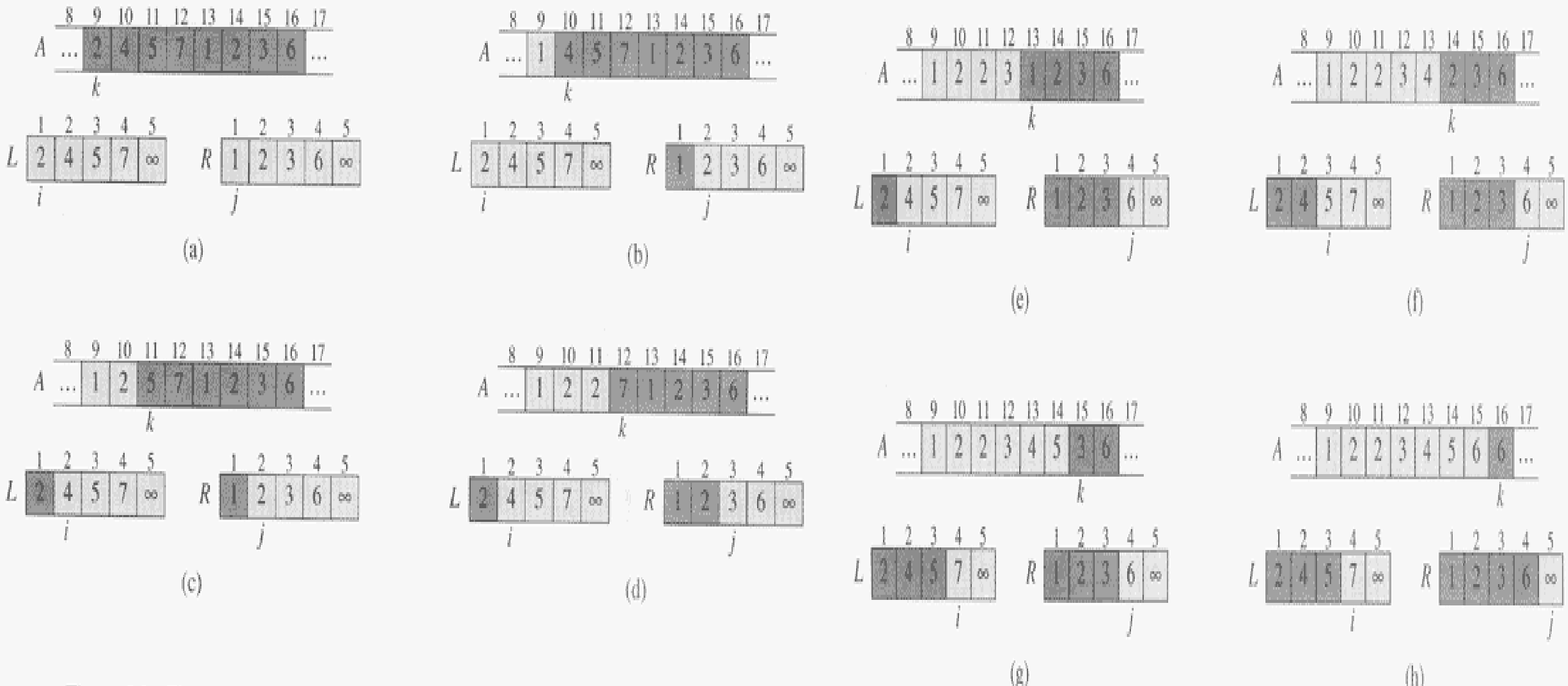


Figure 2.4 The operation of merge sort on the array $A = \langle 5, 2, 4, 7, 1, 3, 2, 6 \rangle$. The lengths of the sorted sequences being merged increase as the algorithm progresses from bottom to top.



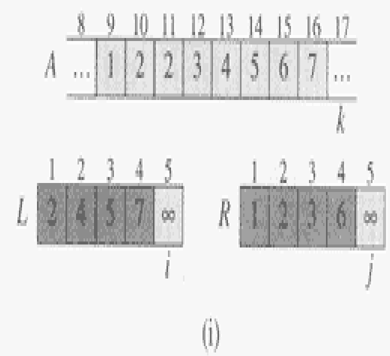


Figure 2.3 The operation of lines 10–17 in the call MERGE$(A, 9, 12, 16)$, when the subarray $A[9..16]$ contains the sequence $\langle 2, 4, 5, 7, 1, 2, 3, 6 \rangle$. After copying and inserting sentinels, the array L contains $\langle 2, 4, 5, 7, \infty \rangle$, and the array R contains $\langle 1, 2, 3, 6, \infty \rangle$. 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.

```
MERGE(A, p, q, r)
n1 ← q − p + 1
n2 ← r − q
create arrays L[1 .. n1 + 1] and R[1 .. n2 + 1]
for i ← 1 to n1
      do L[i] ← A[p + i − 1]
for j ← 1 to n2
      do R[j] ← A[q + j]
L[n1 + 1] ← ∞
R[n2 + 1] ← ∞
i ← 1
j ← 1
for k ← p to r
      do if L[i] ≤ R[j]
            then A[k] ← L[i]
                 i ← i + 1
            else A[k] ← R[j]
                 j ← j + 1
```

Assumption: Array A is sorted from positions p to q and also from positions q+1 to r.

MERGE-SORT(A, p, r)

if $p < r$
  then $q \leftarrow \lfloor (p + r)/2 \rfloor$
    MERGE-SORT(A, p, q)
    MERGE-SORT$(A, q + 1, r)$
    MERGE(A, p, q, r)

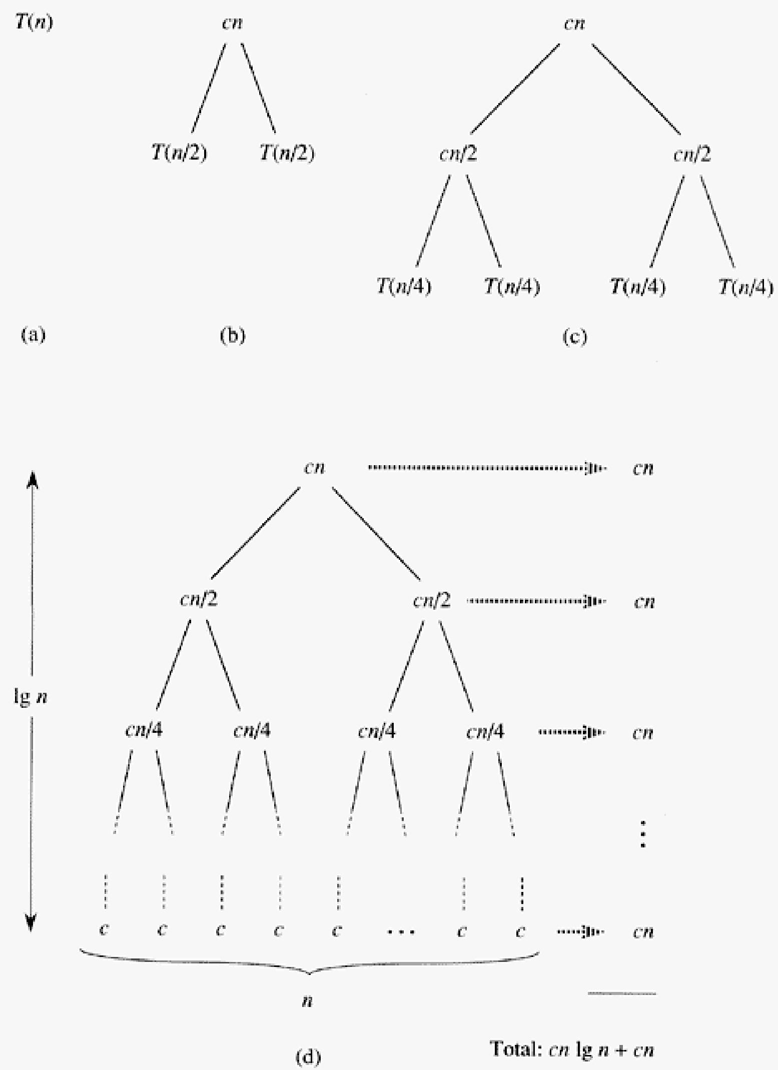


Figure 2.5 The construction of a recursion tree for the recurrence $T(n) = 2T(n/2) + cn$. Part **(a)** shows $T(n)$, which is progressively expanded in **(b)–(d)** to form the recursion tree. The fully expanded tree in part (d) has $\lg n + 1$ levels (i.e., it has height $\lg n$, as indicated), and each level contributes a total cost of cn. The total cost, therefore, is $cn \lg n + cn$, which is $\Theta(n \lg n)$.

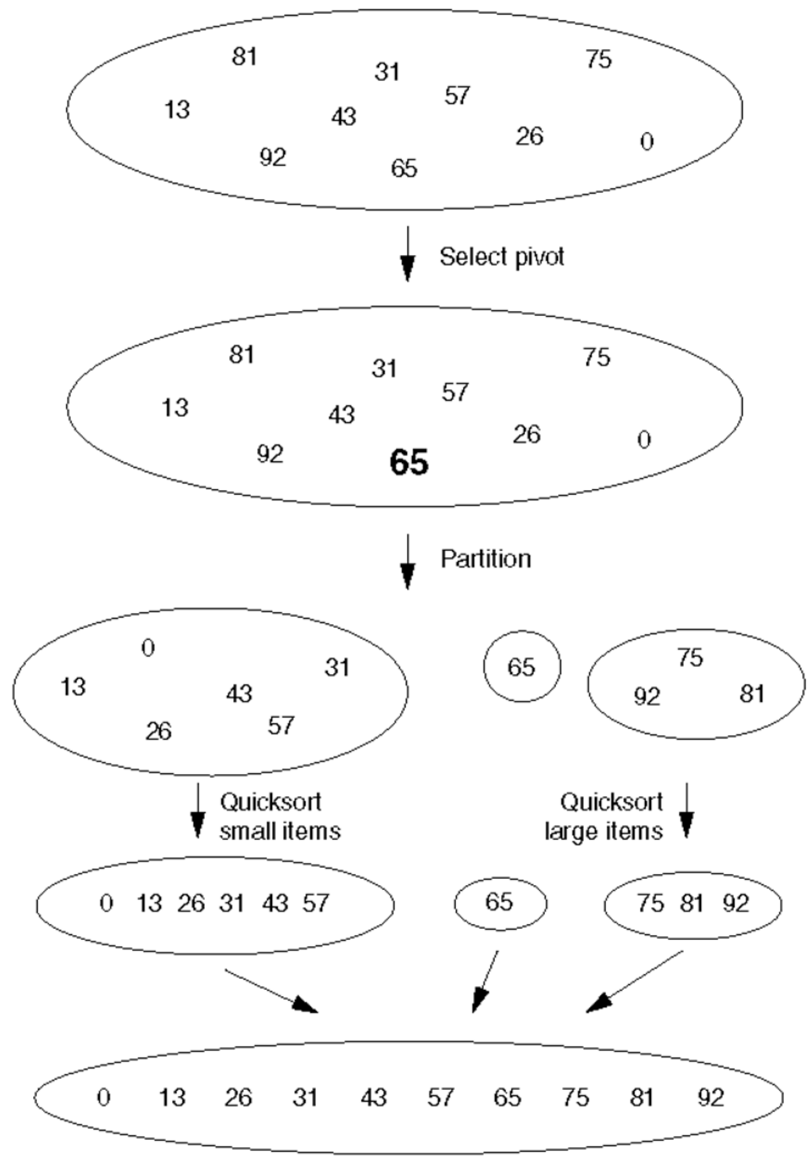
Merge: Algorithm Invariants

- Merge (many lists)
  - ??

Animation Demis

http://cg.scs.carleton.ca/~morin/misc/sortalg/

Figure 8.10 Quicksort



Partition

Figure A If 6 is used as pivot, the end result after partitioning is as shown in the Figure B.

2	1	4	5	0	3	9	8	7	6

Figure B Result after Partitioning

2	1	4	5	0	3	6	8	7	9

QuickSort

$\textsc{QuickSort}(array\ A, int\ p, int\ r)$

if $(p < r)$
    then $q \leftarrow \textsc{Partition}(A, p, r)$
        $\textsc{QuickSort}(A, p, q-1)$
        $\textsc{QuickSort}(A, q+1, r)$

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

$\textsc{Partition}(array\ A, int\ p, int\ r)$

$x \leftarrow A[r]$ $\quad\triangleright$ Choose **pivot**
$i \leftarrow p - 1$
for $j \leftarrow p$ **to** $r - 1$
    do if $(A[j] \leq x)$
        then $i \leftarrow i + 1$
            exchange $A[i] \leftrightarrow A[j]$
exchange $A[i+1] \leftrightarrow A[r]$
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?