J. Edward Black, Jr.

Dr. William Bares

November 29, 2013

# Sort Analysis

## Selection Sort

### Best Case Analysis

Input List: a,b,c,d,e,f,g,h,i,j

Key Compares Big(O): 100 (n^2)

Assignments Big(O): 30 (3n)

Key Comparisons (actual): 45

Assignment Operations (actual): 27

Explanation: For the first iteration, the code compares data[0] with the 9 other elements in the array. After each iteration, the code then no longer has to compare the previous element with any of the others. So, 9+8+7+6+5+4+3+2+1 = 45. As for assignments, it does the a "swap" (3 assignment operations) for each of the first 9 array elements. 3*9 = 27.

### Worst Case Analysis

Input List: j,i,h,g,f,e,d,c,b,a

Key Compares Big(O): 100 (n^2)

Assignments Big(O): 30 (3n)

Key Comparisons (actual): 45

Assignment Operations (actual): 27

Explanation: There is no difference in actual efficiency between best and worst case with selection sort.

## Merge Sort

### Best Case Analysis

Input List: a,b,c,d,e,f,g,h,i,j

Key Compares Big(O): 33 (n log n)

Assignments Big(O): 33 (n log n)

Key Comparisons (actual): 19

Assignment Operations (actual): 19

Explanation: The list is already sorted. Therefore, every sublist is already sorted. The algorithm still has to compare to see this, but not every item will be compared with every other because the algorithm just “pastes” in the remainder of a sorted sublist once it’s sister sublist has been depleted.

### Worst Case Analysis

Input List: a,j,b,i,c,h,d,g,e,f

Key Compares Big(O): 33 (n log n)

Assignments Big(O): 33 (n log n)

Key Comparisons (actual): 23

Assignment Operations (actual): 32

Explanation: There is never a sublist that can be just “pasted” in the the final array. Every element in a sublist must be compared to every element in it’s sister sublist.

## Bubble Sort

### Best Case Analysis

Input List: a,b,c,d,e,f,g,h,i,j

Key Compares Big(O): 100 (n^2)

Assignments Big(O):  constant

Key Comparisons (actual): 45

Assignment Operations (actual): 0

Explanation: The list is already sorted. The loop will still run n^2 times, but no swaps will be done.

### Worst Case Analysis

Input List:j,k,i,h,g,f,e,d,c,b,a

Key Compares Big(O): 100 (n^2)

Assignments Big(O): 300 (3n)

Key Comparisons (actual): 45

Assignment Operations (actual): 135

Explanation: There same number of comparisons is done, but every time the loop runs a swap is performed (except for the last element, then second to last, etc…)

# Quick Sort

## Best Case Analysis

Input List: a,b,c,d,e,f,g,h,i,j

Key Compares Big(O): 33 (n log n)

Assignments Big(O): 33 (n log n)

Key Comparisons (actual): 12

Assignment Operations (actual): 18

Explanation: The list is already sorted. Pivots are still chosen and the algorithm still has to verify that each element is on the correct side of the pivot.

## Worst Case Analysis

Input List: a,j,b,c,g,d,h,e,f,i

Key Compares Big(O): 33 (n log n)

Assignments Big(O): 33 (n log n)

Key Comparisons (actual): 22

Assignment Operations (actual): 30

Explanation: A randomized list will make the quick sort as inefficient as it can be because pivots aren’t strategic.