
2. Describe a worst-case input array of length \( n \) for selection sort and bubble sort. Then, argue that the worst-case time complexity of these sorting algorithms is in \( \Theta(n^2) \).

3. Use an invariant to show the correctness of selection sort.

4. An inversion in a sequence of integers is a pair \( i \) and \( j \) such that \( i \) occurs before \( j \) and \( i > j \).
   (a) Give the inversions in \([2, 3, 8, 6, 1]\).  
   (b) Give a permutation of \([1, \ldots, 10]\) with as many inversions as possible. 
   (c) What can be said about insertion sort and the inversions in the input?

5. Give pseudo-code for a recursive version of insertion sort.
   Give and solve a recurrence equation for the worst-case running time.

6. Give the merge sort tree for \([0, 9, 7, 1, 4, 2, 5, 3, 6, 8]\).

7. Give an example of an input sequence that has running time in \( \Theta(n \log(n)) \) for merge sort, but in \( \Theta(n) \) for insertion sort.

8. Solve the following two recurrence equations:
   (a) (see merge sort)

   \[
   T(n) = \begin{cases} 
   1 & \text{if } n = 1 \\
   2T\left(\frac{n}{2}\right) + n & \text{if } n > 1
   \end{cases}
   \]

   (b) (see tiling example)

   \[
   T(n) = \begin{cases} 
   1 & \text{if } n = 1 \\
   4T\left(\frac{n}{2}\right) & \text{if } n > 1
   \end{cases}
   \]

9. Is an algorithm with a worst-case time complexity in \( \mathcal{O}(n) \) always faster than an algorithm with a worst-case time complexity in \( \mathcal{O}(n^2) \)?