CLC 3: Merge Sort

Chris King && Ryan Scott

The Taco Bell Hierarchy

We will be sorting our favorite items from Taco Bell. It is important to acknowledge that this information is based on the premise that the “Hot” Sauce packets are being used. Here is the initial order:

1

Beef Quesarito

2

Beefy 5-Layer Burrito

3

Crunchwrap Supreme®

4

Nacho Fries

5

Doritos® Locos Tacos

6

Baja Blast®

7

Cinnabon Delights®

The only good hot sauce packet ---------> (Diablo’s okay too)

Time Complexity of Merge Sort

The time complexity of a recursive merge sort, in Big O Notation, is O(n(logn)). This is because the algorithm will always divide the array into two subarrays, and then merge the two, using Divide and Conquer. When it comes to the comparison of the Merge Sort algorithm in the context of larger data sets, merge sort is more efficient than the ladder. However, the speed in which merge sort is performed remains consistent on a data set of any size. When compared to sorting algorithms such as Quick Sort, with a worst-case time complexity of O(n^2), Merge Sort is the more efficient choice.

Visualized Comparison of Merge Sort

Initial Data Set:

Correct Data Set:

Behind the Scenes

Let’s take a look behind the curtains and see what’s really going on when we put the list back into order through the merge sort: