Counting Sort assumes that the range of the input elements is known and relatively small. This is because it uses an array to store the counts of each element, and the size of this array depends on the range of input. If the range is too large, the array used for counting becomes too big, which can cause memory issues. Additionally, Counting Sort assumes that the input elements are non-negative integers.

Bucket Sort assumes that the input elements are uniformly distributed over a range. This is because the buckets are created based on a hashing function that maps each element to a specific bucket. If the input elements are not evenly distributed, some buckets may end up with too many elements, while others may end up with too few. This can lead to poor performance of the algorithm. Bucket Sort also assumes that the sorting algorithm used to sort the individual buckets is efficient and stable.

If these assumptions are not satisfied, the performance and correctness of these algorithms may be impacted. For example, if the input range is too large for Counting Sort, it may not be able to allocate enough memory for the counting array, leading to a memory error. Similarly, if the input elements are not uniformly distributed for Bucket Sort, some buckets may become very large, leading to a high time complexity. Therefore, it's important to understand these assumptions and use these algorithms only when the input satisfies them.