Radixsort

Computational effort

Because the computational effort does not only depend on the number of elements n that are to be sorted but also depends on the maximum number of digits in a single element m, the complexity is denoted as $O(m \cdot n)$.

If the maximum number of digit is a known constant, the complexity can also be described as $O(n)$.

When does it make sense to use this algorithm?

Let's assume your job is to sort all the letters in a postoffice by postal code. If that is the case, the number of letters most likely exceeds the number of digits (in Germany that would be 5) significantly. Therefore you only have to do five iterations and in each iteration distribute n objects. Here the algorithm is a sound option.

If you work as an accountant and have to compare the net worth of your wealthiest two clients, you will be left with two very large numbers. For each of those digits, you would have to do one iteration and in each iteration distribute in two objects. At this point you would even be better off with the really impractical Selectionsort ($O(n^2)$ with $n = 2$).