1 End of sorts, other algorithms, introduction of the `class` statement

1.1 Simple questions

If, given an array of n `doubles`, we want to find the couple so that the sum is maximal, what is the best strategy ? And if we want to find the two elements so that the absolute value of their difference is the smallest ?

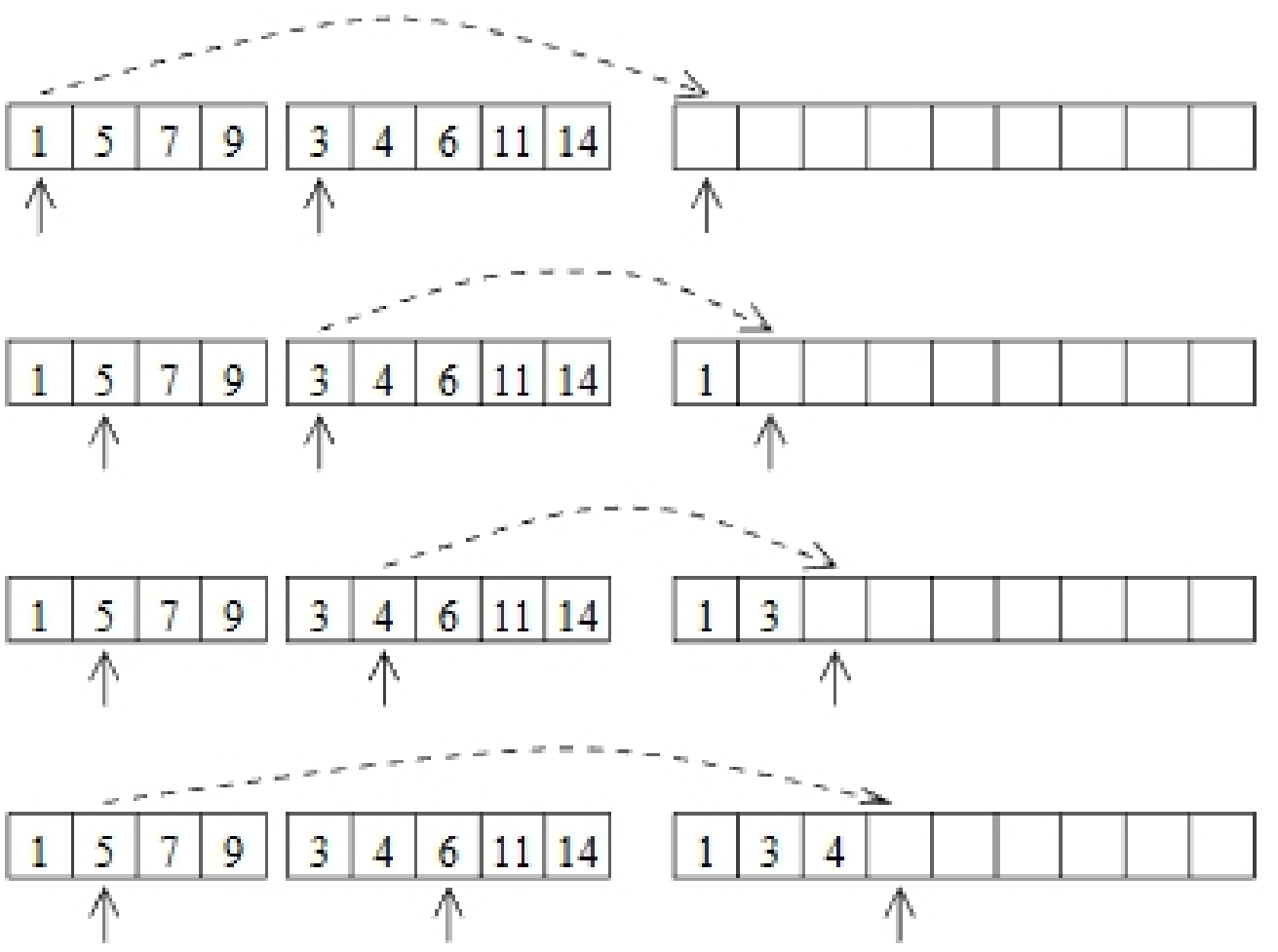
What is the best costs you can imagine for those two problems?

1.2 Fusion sort

The usual dumb algorithms for sorting things require a number of operations proportional to the square of the number of elements to sort ($O(n^2)$). In practice, the used algorithms need a number of operations proportional to $n \times \log n$.

The first one is the **fusion sort**.

The main point is that given two sorted list of numbers, generating the sorted merged list needs a number of operations proportional to the size of this result list. Two index indicate the next elements to take from each list, and one indicates where to store the smallest of the two :



This process can be iterated, starting with packets of size 1 (which are *already* sorted ...) and merging them each time two by two. After k iterations of that procedure, the packets are of size 2^k, so the number of iterations for this process is $\log_2 n$ where n is the total number of objects to sort.

Each step of this main process cost the sum of the sizes of the resulting packets, which is n. Finally the total number of operations is $\sum_{i=1}^{\log_2 n} n = n \times \log_2 n$.