COMP3121: Assignment 1 – Q2

Gerald Huang
*z*5209342
Updated: June 15, 2020

In this question, we make a very important assumption. Each bolt's diameter must be unique and as a result, it is almost impossible to compare the sizes of the bolt to another bolt. We shall begin this question by considering a simpler case.

A simple case

Consider 5 bolts and some quantity S_k nuts, with every bolt occupying a different diameter. Label these bolts as B_1, B_2, B_3, B_4, B_5. It is carefully noted that the nuts aren't *necessarily* distinct in diameter.

Begin by selecting a random bolt and trying every nut, keeping track of whether the nut is too big, too small, or just right. Then pick a random nut that fits just right and see if it fits any of the other 4 bolts, keeping track of the result. One may observe that, since each bolt is unique in diameter, this bolt cannot possibly fit any of the other bolts available. Hence we have established our first nut-bolt pairing. In doing this, we have also constructed three separate groups of nuts and bolts; nuts that are too small, too big or just right to the original bolt that we have chosen. For example, let the pivot bolt be B_3, where B_1 and B_2 are smaller than B_3, while B_4 and B_5 are bigger than B_3. Then, for some subset of nuts S_i, this group of nuts will be smaller than the pivot bolt. Conversely, for some subset of nuts S_j, this new group of nuts will be bigger than the pivot bolt.

We will repeat this process with S_i nuts and the bolts B_1 and B_2. Picking a random pivot bolt B_1, we try every nut and record our findings. We repeat this process until all of the bolts have a nut pairing. We will now move onto the more general case.

A more general algorithm

Define A_k to be a set of bolts where $k \in [0, n]$ with differing diameters and define some quantity of nuts S_k. From the pool of bolts, we shall pick a random bolt to be the *pivot* bolt. Let that bolt be A_i for some $i \in [0, n]$. Compare the diameter of A_i with every nut, keeping track of whether the size of the nut is too small, too big or just right. This partitions the nuts into three groups: nuts that are too big, too small or fits the chosen bolt. Out of the nuts that fit the bolt, try every bolt on this nut and keep a record of whether the bolt is too big, too small, or just right. We make use of the assumption that every bolt is *unique*. This creates a one-to-one correspondence between a nut and a bolt. In other words, there can be no other bolt that matches the size of the given nut. Otherwise, we arrive at a contradiction of unique bolts since this would mean that two bolts share the same diameter. Hence this partitions our bolts into two groups: one that is smaller than A_i and one that is bigger than A_i in terms of its diameter. This algorithm runs in $O(n)$ time since we are comparing at most n bolts and nuts, with each comparison run in $O(1)$ time. Hence, at each level, we have $O(n)$ comparisons.

However, we have essentially split this problem into two smaller problems; we have split our nuts and bolts into two

separate groups. So we recurse through this problem using a **divide and conquer** method, similar to that of quick sort. In each level, we pick a random bolt from a particular group that we have partitioned to be our *pivot* and then split our corresponding nuts and bolts into two separate groups, recursing through the smaller and bigger groups. Since we're effectively dividing our original problem into two subproblems, this gives us a total of $\log n$ depths, and in each depth, we make $O(n)$ comparisons. So the expected time of execution is $O(n \log n)$ steps.