Paranoid Quicksort

Here we prove that the runtime of paranoid quicksort is $O(n \log^2 n)$ *with high probability*. Let us recap what paranoid quicksort does:

Algorithm 1: $\textsc{ParanoidQuicksort}(A)$

if $|A| \leq 1$ **then**
  return A;
repeat
  Select a pivot $p \in A$ uniformly at random;
  $L \leftarrow$ elements smaller than p in A;
  $R \leftarrow$ elements larger than p in A;
until $|L| \geq |A|/3$ *and* $|R| \geq |A|/3$;
return $\textsc{ParanoidQuicksort}(L) + p + \textsc{ParanoidQuicksort}(R)$

It is easy to see that the recursion depth in the algorithm is $O(\log n)$ (where $n = |A|$) since at each level, the input list size decreases by a factor of $2/3$ (i.e. $|L| \leq 2|A|/3$ and $|R| \leq 2|A|/3$).

Therefore, we only need to bound the number of iterations where we pick a pivot uniformly at random. Assume for simplicity that $|A|$ is a multiple of three[1]. For sake of analysis let Y be the sorted version of A. We partition Y into three equal parts (Y_1, Y_2, Y_3).

We say a pivot is *good* if it belongs to Y_2. Since we pick a pivot uniformly at random, the event that a pivot is good happens with probability $1/3$. Note that if we pick a good pivot, we immediately exit the loop, and go to the next recursion level.

Thus, the probability of choosing a bad pivot is $2/3$. If we have $t + 1$ iterations, one must pick a bad pivot t times in a row. This occurs with probability

$$= \mathrm{P}[(1^{st}\text{pivot bad}) \cap (2^{nd}\text{pivot bad}) \cap \cdots \cap (t^{th}\text{pivot bad})] = (2/3)^t$$

Observe that the recursion tree is a binary tree of depth $\log_{3/2} n$, so there are at most $2^{\log_{3/2} n} = O(n)$ nodes in the recursion tree. Therefore we can use the union bound to show that

$$\mathrm{P}[\exists \text{ a bad node in the recursion tree}] \leq O(n) \cdot (2/3)^t.$$

Setting $t = 3 \log_{3/2} n = O(\log n)$ makes this probability at most $1/n$.

Therefore, we have that with probability $1 - 1/n$, $t = O(\log n)$ for each recursive call. This implies that the time required to pick a good pivot in single recursive call is $O(n \log n)$. Therefore, with probability $1 - 1/n$, the running time of $\textsc{ParanoidQuicksort}$ is given by the following recursive relation:

$$T(n) \leq 2 \cdot T(2n/3) + O(n \log n)$$

And by solving this relation, we get that the total running time is $O(n \log^2 n)$ with high probability[2].

[1] We can remove this assumption at the expense of making the analysis a bit uglier. We can also satisfy this assumption by adding some dummy elements with value ∞ to A and delete them later

[2] by setting $t = (c + 2) \log_{3/2} n$ for some constant $c \geq 1$, we can increase the success probability to $1 - 1/n^c$