A Brief Proof to QuickSort Tail Recursion

Steven

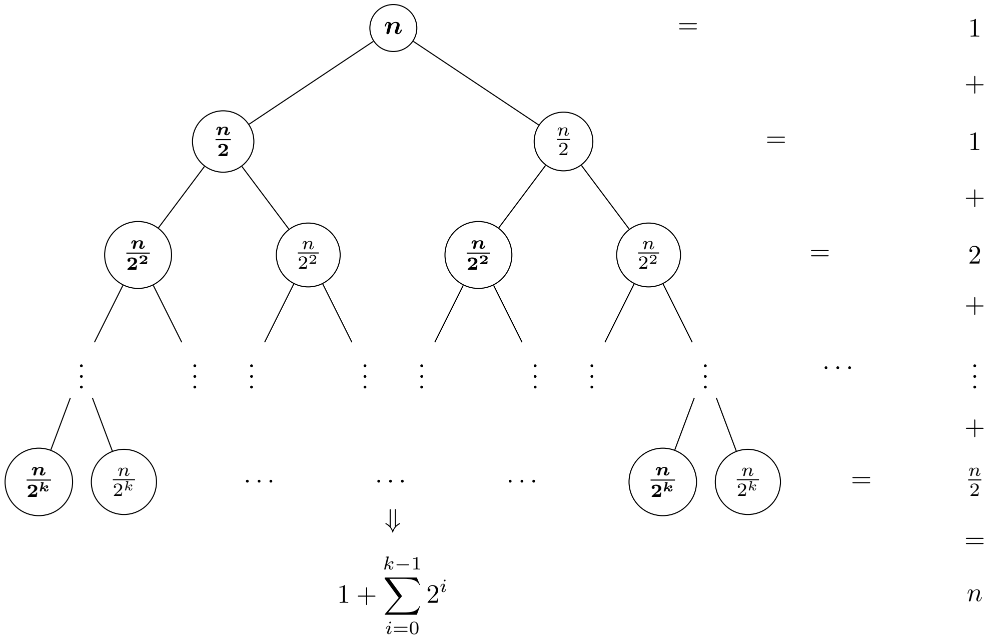
1 Original Algorithm

In the classical QuickSort algorithm, we recursively apply the QuickSort to the sub-range up to the point of division and to the sub-range after it util there is only one element in the sub-range. So we call QuickSort function for $1 + 2 + 4 + \cdots + n = 2n - 1$ times.

2 Tail Recursion Optimization

Inspired by the tail recursion, we've got the optimized algorithm as follows.

```
void QuickSort(int *array, int p, int n) {
    while (n > 1) {
        int r = partition(array, p, n);
        QuickSort(array, p, r);
        *array += r + 1;
        n -= r - 1;
    }
}
```



For each child node pair, only the left node will call QuickSort function, the other one's function call in original algorithm is replaced by the update of *array and n. And take a view of the bottom nodes, once we complete the calculation of the left node, the right one will break the loop because $n \not> 1$, and return to their parent node immediately. And if the parent node is also a right child node, it will complete function execution for the same reason.