Due: February 25/26, 1998

Read Sections 3.1, 3.2,

Do Exercises 3.2, 3.5, 3.8, 3.20, 3.31 from GKP. Do not submit your solutions, but check them with the answers from the back of the text.

1. (12 points) 3-WAY MERGESORT

Since classical (2-WAY MERGESORT) is so efficient, consider the following variation which also sorts $n$ elements.

3-WAY MERGESORT:

IF $n > 1$

Partition the $n$ elements into three lists, as equal in size as possible

3-WAY MERGESORT each of these three lists

Merge the smaller two of these three lists, yielding a list of size approximately $2n/3$

Merge this list and the largest of the three original lists

Assume that merging sorted lists of lengths $p$ and $q$ into a sorted list of length $p+q$ requires $p+q-1$ comparisons. As an example, a list of 11 elements would first be partitioned into lists of 3, 4 and 4 elements. Each of these lists would be 3-WAY MERGESORTed (using a certain number of comparisons), and then two sorted lists of lengths 3 and 4 would be merged into a sorted list of length 7 (using 6 comparisons), and then this list of length 7 would be merged with the final list of length 4 into a sorted list of length 11 (using 10 comparisons). The total number of comparisons for $n=11$ would be 16 plus the total numbers of comparisons to 3-WAY MERGESORT lists of lengths 3, 4 and 4.

(A) Assuming that $n$ is a power of 3 (that is, assuming there exists integer $k$ such that $n=3^k$), develop and solve (using characteristic equations) a recurrence for $f(n)$, the number of comparisons used by 3-WAY MERGESORT for $n$ elements. Some small values are $f(1)=0$, $f(3)=3$ and $f(9)=22$.

(B) Assuming that $n$ is not necessarily a power of 3, develop a recurrence for $g(n)$, the exact number of comparisons used by 3-WAY MERGESORT for $n$ elements. Some small values are $g(1)=0$, $g(2)=1$, $g(3)=3$, $g(4)=5$. What are $g(6)$, $g(7)$ and $g(8)$? Develop a recurrence for $h(n)=g(n)-g(n-1)$. The function $g(n)$ should not appear in the recurrence for $h(n)$. You needn’t solve the recurrences for $g(n)$ and $h(n)$.
2. (7 points) Suppose you want to sort $n$ distinct numbers where each of the $n!$ input permutations is equally likely. The expected number of pairwise comparisons to INSERTIONSORT the $n$ elements is $\frac{n(n+1)}{4} - \frac{1}{2}$. We also saw that the expected time to QUICKSORT the $n$ elements is about $2(\ln n + 1) H_n$. However, for small values of $n$ (say all $n$ less than or equal to some $\theta$), INSERTIONSORTing is faster than QUICKSORTing. Consider the hybrid algorithm, QUICKINSORT, which acts like QUICKSORT for large values of $n$ but INSERTIONSORT for small values of $n$.

```c
void QUICKINSORT(int \theta, lo, hi)
{
    int Partition();
    int pivot;
    if (hi-lo<=\theta) INSERTIONSORT(lo,hi);
    else {
        pivot = Partition(lo, hi);
        QUICKINSORT(\theta,lo, pivot - 1);
        QUICKINSORT(\theta,pivot+1, hi);
    }
}
```

Describe and solve (find a closed form for) a recurrence for $f(\theta,n)$, the expected number of pairwise comparisons to QUICKINSORT $n$ distinct numbers where each of the $n!$ input permutations is equally likely.