Treasure Hunting

Zheng Zangwei*

April 19, 2018

Problem 1

1. *Analyze the worst case $W(n)$ and the best case $B(n)$ time complexity of mergesort as accurately as possible.*
   Explore the relation between them and the binary representations of numbers.
   Plot $W(n)$ and $B(n)$ and explain what you observe.

2. *Analyze the average case $A(n)$ time complexity of mergesort.*
   Plot $A(n)$ and explain what you observe.

3. *Prove that: The minimum number of comparisons needed to merge two sorted arrays of equal size m is $2m-1$.*

Algorithm 1

```
Subroutine Mergesort(A,p,r)
    if (p<r)
        q = [(p+r)/2];
        Mergesort(A,p,q);
        Mergesort(A,q+1,r);
        Merge(A,p,q,r);
```

1 Problem 1.1

We know from the algorithm, $W(0) = 0, W(\lfloor \frac{n}{2} \rfloor) + W(\lceil \frac{n}{2} \rceil) + n - 1$; We have following lemma which I omit the proof:

*E-mail: zzw@smail.nju.edu.com Student ID: 171860658

Lemma 1

$$\lceil \frac{n}{2} \rceil = \lfloor \frac{n+1}{2} \rfloor$$

Lemma 2

If $T(n) = 1 + T(\lfloor \frac{n}{k} \rfloor)$ *we have* $T(n) = \lceil log_k(n+1) \rceil$. *This can be derived from k-base number.*

Now consider W(n+1)-W(n), we have

$$W(n+1) - W(n) = W(\lfloor \frac{n}{2} \rfloor + 1) - W(\lfloor \frac{n}{2} \rfloor) + 1.$$

Let $f(n) = W(n+1) - W(n)$ we have $f(n) = 1 + f(\lfloor \frac{n}{2} \rfloor)$ so that $W(n+1) - W(n) = f(n) = \lceil log_k(n+1) \rceil$. By summation, $W(n) = \sum_{k=1}^{n} \lceil log_2 k \rceil$.

This introduced an interesting property which I want but cannot find a intuitive proof to it.

Corollary 1

If n is even, W(n) denotes the number of bits in binary representations, then

$$\sum_{k=0}^{\frac{n}{2}-1} W(k) + n - 1 = \sum_{k=\frac{n}{2}+1}^{n} W(k)$$

Remark

This property is the same as the one has been discussed on class, which is illustrated in following picture.

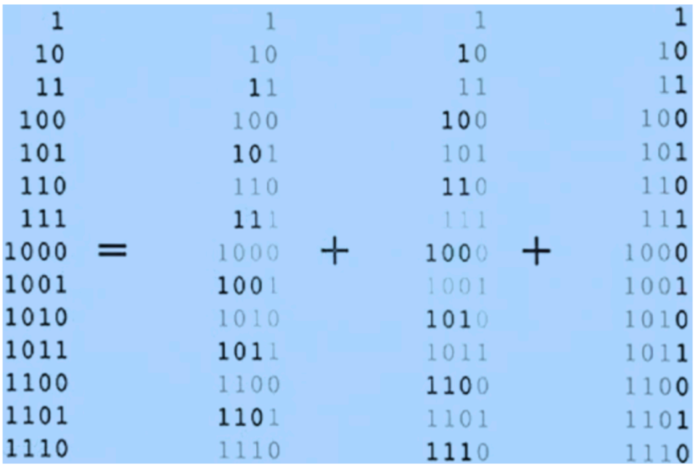

Figure 1: Property

Then from the great *Concrete Mathematics* problem 3.34 we have the following calculation. Let $m = \lceil log_2 n \rceil$,

$$\begin{aligned} W(n) + (2^m - n)m &= \sum_{k=1}^{2^m} \lceil log_2 k \rceil \\ &= \sum_{j,k} j[j = \lceil log_2 k \rceil][1 \leq k \leq 2^m] \\ &= \sum_{j,k} j[2^{j-1} < k \leq 2^j][1 \leq jm] \\ &= \sum_{j=1}^{m} j * 2^{j-1} = 2^m(m-1) + 1 \end{aligned}$$

Thus $W(n) = n\lceil log_2 n \rceil - 2^{\lceil log_2 n \rceil} + 1$.

2 Problem 1.2

Still, we know $B(0) = 0, B(\lfloor \frac{n}{2} \rfloor) + B(\lceil \frac{n}{2} \rceil) + \lfloor \frac{n}{2} \rfloor$.

Definition 1

If $T(n) = [2 \not| n] + T(\lfloor \frac{n}{2} \rfloor)$ we define $T(n) = Dn$. Dn is the number of 1s in the binary representations.

Now consider B(n+1)-B(n), we have

$$B(n+1) - B(n) = B(\lfloor \frac{n}{2} \rfloor + 1) - B(\lfloor \frac{n}{2} \rfloor) + [2 \not| n].$$

B(n) is waiting to be counted.

3 Problem 1.3

Here is the plot of W(n) and B(n) along with A(n).

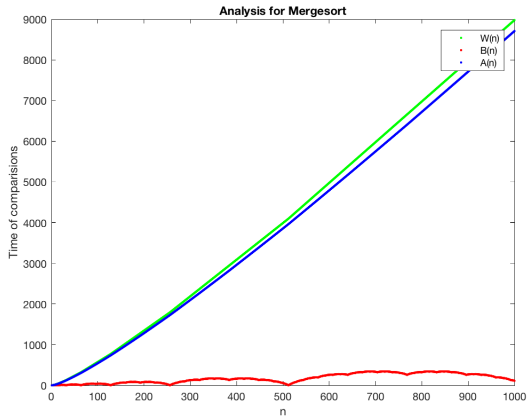

Figure 2: n=1000

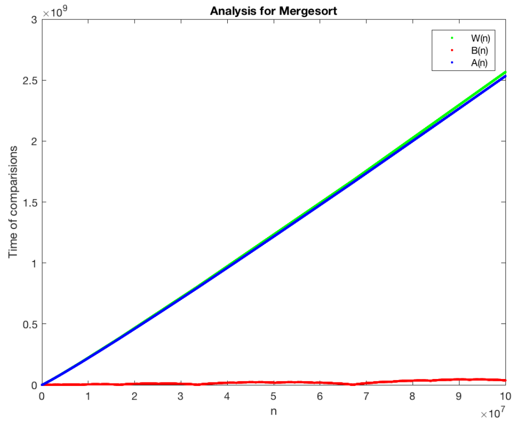

Figure 3: n=100000000

From the plot, we know that W(n) and A(n) are close when n is large. A(n) remains small even when n is very large.

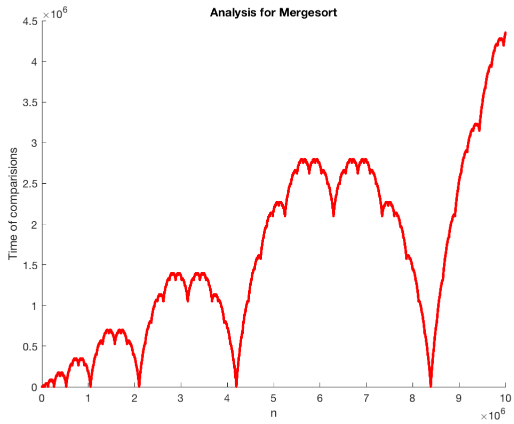

Figure 4: B(n)

For B(n), it is cool.

4 Problem 2

Definition 2

R is the number of elements left in the non-empty array by MERGE(A,B) on A of size a and B of size b.

Theorem 4.1

$$P(R \geq r) = \frac{C^a_{a+b-r}}{C^a_{a+b}} + \frac{C^b_{a+b-r}}{C^b_{a+b}},\ E(R) = \sum_r P(R \geq r) = \frac{a}{b+1} + \frac{b}{a+1}$$

Proof

For the case R>=r, there are two possible situation. We consider that A is not empty. Since there are a+b number at all, then $B_b < A_r$ which means B_b is at most the (a+b-r)th most number. Thus there are C^a_{a+b-r} possible cases. There are C^a_{a+b} in all. Thus the first formula is generated.

$E(R) = \sum_r r * P(r) = (1 * P(1) + 2 * P(2) + ...) + (2 * P(2) + 3 * P(3) + ...) + ... = \sum_r P(R \geq r)$.

Q:Is there a formal way to get this equation? ☐

Then we have $A(n) = A(\lfloor \frac{n}{2} \rfloor) + A(\lceil \frac{n}{2} \rceil) + (n - E(R)) = A(\lfloor \frac{n}{2} \rfloor) + A(\lceil \frac{n}{2} \rceil) + (n - (\frac{\lfloor \frac{n}{2} \rfloor}{\lceil \frac{n}{2} \rceil + 1} + \frac{\lceil \frac{n}{2} \rceil}{\lfloor \frac{n}{2} \rfloor + 1}))$.

5 Problem 3

Theorem 5.1

Each of the 2m-1 comparisions $B_1 : A_1, A_1 : B_2, B_2 : A_2, ..., Bm : A_m$ *must be made for list A:A1 <· · ·<Am B:B1 <· · ·<Bm subject to B1 <A1 <B2 <A2 <· · ·<Bm <Am.*

Proof

Suppose there are less than 2m-1 comparisions between A and B. Then at least two B_i *is compared less than two time. Then at least one comparsion is not* $B_1 < A_1$*. Without loss of generality, we suppoese the comparision is* $B_i < A_j$*. Thus* $A_{j-2} < B_{i-1} < B_i < A_j$*. Altough we may have the comparison* $B_{i_1} < A_{j-1}$*, we can exchange* B_i *with* A_{j-1} *and get another valid permutation. Thus we need at least 2m-1 comparisions.* □

Since there exists two array that must be compared 2m-1 times, thus the minimal comparison time is 2m-1.