CSC373 - Problem Set 1

Authors: Luke Bacchus, Naslin Rahman, Zhuoqian Li

January 18, 2021

Question 1

a. We have n piles of m papers. As merging two sorted piles can be done it time proportional to the size of the resulting pile, following the given algorithm the first merge will take $2m$ time, the second merge will take $3m$ time, etc. The running time of this algorithm can be found:

$$\begin{aligned} &= 2m + 3m + 4m + ... + nm \\ &= \sum_{k=2}^{n} km \\ &= m(\sum_{k=2}^{n} k) \\ &= m(\sum_{k=1}^{n} k - 1) \\ &= m(\frac{(n+1)(n)}{2} - 1) \end{aligned}$$

This is $\mathcal{O}(mn^2)$.

b. If you have exactly two piles, merge them as in a. If you have more than two piles, divide into two sets of piles of equal size (we can do this because n is an exact power of two). Then, recursively merge each set of piles until you obtain two piles $p1$ and $p2$. Then merge $p1$ and $p2$. The recurrence relation is $T(n) = 2T(n/2) + mn$. By the master theorem, this is $O(mn \log n)$.

Question 2

a. Divide the array into two subarrays: $s1 = A[0 : \frac{n}{2}]$ and $s2 = A[\frac{n}{2} + 1 : n]$. Recursively find S_{ij} such that $S_{ij} = \sum_{r=i}^{j} A[r]$ is the maximum sum over all possible i and j, $1 \le i \le j \le n$ for both subarrays. Then linearly scan left and right from the midpoint to find the maximum subarray beginning in $s1$ and ending in $s2$. Finally, take the maximum of all three maximum sums.

```
# Assume arrays start at index 1
function func(A, i, j):
    if i == j
        return A[i]
    mid = (i + j)/2
    leftSum = func(A, i, mid)
    rightSum = func(A, mid+1, j)
    crossSum = helper(A, i, j, mid)

    return max(leftSum, rightSum, crossSum)


function helper(A,start,end, mid):
    leftSum = 0
    rightSum = 0
    totalSum  = 0

    for i = mid downto start
        sum += A[i]
        if sum > leftSum
            leftSum = sum

    sum = 0
    for j = mid +1 to end
        sum += A[j]
        if sum > rightSum
            rightSum = sum

    return leftSum + rightSum
```

The recurrence relation is $T(n) = 2T(n/2) + n$. By the master theorm this is $O(nlogn)$

b.