Illinois Institute of Technology
Department of Computer Science

Homework 4 Solutions

CS 430 Introduction to Algorithms
Spring Semester, 2014

1. **Problem 1 Solution:** Example of quick sort taking$O(n^2)$ steps: When the array is sorted and the pivot element is either the first or last element then quicksort takes$O(n^2)$ time.
   $T(n) = c(n + n - 1 + n - 2 + ... + 1) + n.T(0)$
   $= c.\frac{(n+1)n}{2} + 0 = O(n^2)$

2. **Problem 2. Solution:**

   (a) Prove A is a Monge array
   $= \forall i, j\ddot{k}, l\ddot{A}[i, j] + A[k, l] \leq A[i, l] + A[k, j]$
   $= A[i, j] + A[i + 1, j + 1] \neq A[i, j + 1] + A[i + 1, j]$
   Since this is a if and only if statement we have to prove it both ways.
   A is a Monge array
   $= \forall i, j\ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$
   If A is a Monge array then $\forall i, j\ddot{k}, l\ddot{A}[i, j] + A[k, l] \leq A[i, l] + A[k, j]$ is true. $\forall i, j\ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j +$
   is just a special case of the above property where
   $k = i + 1, l = j + 1$
   Hence proved.
   To prove that if $\forall i, j\ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$
   is true then A is a monge array.
   Proof by induction:
   Base Case : 2×2 submatrix of A Yes this is true beacuse $k = i+1, l = j+1$ are the only possible values.
   Assume true for $2n - 1$ say A' submatrix of A
   To prove true for $2n$ if $j, l \in A$ then by inductive hypothesis
   $\forall i, jk, lA[i, j] + A[k, l] \leq A[i, l] + A[k, j]$ is true.
   If $j \in A$ and $l \in A$ then we have
   Assume $j = n - 1, l = n, i = 1, k = 2$ So we get
   $A[1, n1] + A[2, n] \leq A[1, n] + A[2, n1]$
   This is true from
   $\forall i, j\ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$
   say $T1$.
   Also
   $A[1, 1] + A[2, n1] \leq A[1, n1] + A[2, 1]$ is true from assumption that A' is true say $T2$. Adding $T1$ and $T2$ we get:
   $A[1, 1] + A[2, n] \leq A[1, n] + A[2, 1]$ Which proves for $2 \times n$
   Now we consider the rows:
   Base Case : $2 \times n$ submatrix of A
   Yes this is true beacuse we already proved it above.
   Assume true for $m1 \times n$ say A' submatrix of A
   To prove true for $m \times n$ Assume $k = n1$, $l = i$, $r = m1$.
   Let$A[1, k]$, $A[1, l]$, $A[r, k]$,
   $A[r, l]$, $A[m, k]$, $A[m, l]$ be a, b, c, d, e, f respectively.
   So we get:
   $c+f \leq e+d$ from $\forall i, j\ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$ and the previous induction on columns
   $a + d \leq b + c$ From $m1 \times n$ case and the previous induction on columns

Adding these 2 terms we get: $a + f \leq e + b$
which proves true for $m \times n$
Hence A is a Monge array

(b) From the previous part we know that an matrix is a monge array iff
$\forall i, j \ddot{A}[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$
The only element not satisfying that property is $A[1, 3] = 22$, so we replace 22 by any number > 24 say 26.

(c) Proof by contradiction:
Assume $f(i) > f(i + 1)$
Let $m = f(i), n = f(i + 1)$
Therefore, $A[i, m] + A[i + 1, n]$ must be the smallest sum between any 2 numbers in the row i and $i + 1$.
But by monge array property $A[i, m] + A[i + 1, n] \neq A[i, n] + A[i + 1, m]$
Contradiction. Therefore, $f(i)f(i + 1)$

(d) Let a_i be the index of the minimum number in row i
Assume m is an even number
$T(m/2) = T(1) + T(3) + .. + T(m1)$
$= (m) - f(0) + m/2$
$= n + m/2$
$= O(n + m)$

(e) $T[s] = T[m/2] + O(m + n)$
$= T[m/4] + m + m/2 + 2n$
$= n + n \log m + 2m$
$= n(1 + \log m) + 2m$
$= O(m + n \log m)$

3. **Problem 3**. **Solution:** When the input array is already sorted in increasing order, HEAPSORT takes $\Omega(nlgn)$ time since each of the $n - 1$ calls to MAX-HEAPIFY takes $\Omega(lgn)$time.

4. **Problem 4**. **Solution:** IN radix sort we start with the least significant digits first. So at the i^{th} iteration of radix sort, the numbers are sorted with respect to the i^{th} least significant digits.
So, we can output any number as soon as we consider all its digits we can put it in the right place in the sorted array.
k is the total number of digits on all numbers.
n is the size of the list.
To output the numbers whose $length = i$ after i^{th} iteration, we need to check the length of each number before putting it in a bucket at round$i + 1$.
If number length is less than i then move the number to the result list.
So the time complexity is $O(n + k)$

5. **Problem 5**. **Solution:** To get the lower bound on the solution we need to consider a binary tree with subsequences containing $\log n$ elements as leaf nodes. Now there are $\log n!$ different ways to sort each of the subsequences. There are $\frac{n}{\log n}$ such subsequences so $(\log n!)^{n/\log n}$. Any decision tree which does this takes $\log((\log n!)^{n/\log n})$ So we have
$= \frac{n}{\log n} \log \log n!$
$= \frac{n. \log n}{\log n} \log \log n$
$= n \log \log n$