Solution to HW 4

February 28, 2013

1. (a) Prove A is a Monge array
$=> \forall i, jk, l A[i,j] + A[k,l] \leq A[i,l] + A[k,j]$
$=> A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$
Since this is a if and only if statement we have to prove it both ways.

A is a Monge array $\implies \forall i, j A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$
If A is a Monge array then $\forall i, jk, l A[i,j] + A[k,l] \leq A[i,l] + A[k,j]$ is true.
$\forall i, j A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$ is just a special case of the above property where$k = i+1, l = j+1$.
Hence proved.
To prove that if $\forall i, j A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$ 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 $2 \times n-1$ say A' submatrix of A
To prove true for $2 \times n$
if $j, l \in A'$ then by inductive hypothesis $\forall i, jk, l A[i,j] + A[k,l] \leq A[i,l] + A[k,j]$ 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
$A[1,n-1] + A[2,n] \leq A[1,n] + A[2,n-1]$
This is true from $\forall i, j A[i,j] + A[i+1,j+1] \leq A[i,j+1] + A[i+1,j]$ say $T1$.
Also
$A[1,1] + A[2,n-1] \leq A[1,n-1] + A[2,1]$ 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 $m - 1 \times n$ say A' submatrix of A
To prove true for $m \times n$
Assume$k = n - 1, l = i, r = m - 1$.
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 $\forall i, j A[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$ and the previous induction on columns
$a + d \leq b + c$
From $m - 1 \times n$ 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 $\forall i, j A[i, j] + A[i + 1, j + 1] \leq A[i, j + 1] + A[i + 1, j]$
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 $A[i, m]+A[i+1, n] \leq A[i, n]+A[i+1, m]$
Contradiction. Therefore, $f(i) \leq f(i + 1)$

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

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

2. 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.

$T(n) = c(n + n - 1 + n - 2 + ... + 1) + n.T(0)$
$\implies c.\frac{(n+1)n}{2} + 0$
$\implies O(n^2)$

3. Best Possible partititon is $n/5$
   So the tree will be of degree 5 i.e each node will have 5 children.
   The height of the tree$i = log_5 n$
   As we keep increasing the number of partitions the tree height decreases but the contant factor increases as the the number of children increase. So it is not better than a binary tree. Thus time complexity will be $O(nlog_5 n)$

   Algorithm will be the same in the book except we will be partitioning the array into 5 parts instead of 2 in every step.

```
Partition(i,j, p, A)
   x <- A[p]

   while true do
      repeat
         j <- j + 1
      until A[j] <= x &&  i<=j

      repeat
         i <- i + 1
      until A[i] >= x && j>=i

      if i < j then
         exchange(A[i], A[j])
      else
         return j


Quicksort(A, f, len)
 i <- f
j<- len+1 -f
f1= partition(i,j, p1, A)
f2= partition(i,j, p2, A)
f3= partition(i,j, p3, A)
f4= partition(i,j, p4, A)
Quicksort(A, f, f1-f+1)
Quicksort(A, f1+1, f2-f1)
Quicksort(A, f2+1, f3-f)
Quicksort(A, f3+1, f4-f)
Quicksort(A, f4+1, len-f4-1)
```