Algorithms: Assignment One

Deadline: Oct 5, 2020

1. Consider a tournament with n players, with every pair of players playing one game between them. Thus, the number of games played is $\binom{n}{2}$. Every game is decisive (i.e. one player wins). Suppose you are given access to the results of the games. Describe an algorithm to label the players as $1, 2, \ldots, n$ in such a way that Player i won against Player $i + 1$ for every $i \leq n - 1$.

2. This problem is from Jeff Erickson's notes.

   The input is a sorted array of n distinct numbers that has been right-rotated k steps, for an unknown integer k between 1 and $n - 1$. That is, $A[1, 2, \ldots, k]$ and $A[k + 1, \ldots, n]$ are in increasing order, with $A[n] < A[1]$.

   An example is the sequence $10, 14, 17, 24, 25, 3, 4, 7$ where $n = 8$ and $k = 3$.

   Describe an algorithm that to find the unknown value of k, given an input array of the form described above.

3. Consider the following problem.

   Input: An array A[1,2,...,n] with all the m+n elements of A union B being distinct. Output: The number of pairs (x,y) such that x in A and y in B and x¡y.

   Eg: A: 1,3,5,7,19 and B: 2,4,10,14 Output: 11

   The above problem can be solved in O(m+n) time by modifying the merge procedure (see sample pseudocode for this below).

   ```
   Merge(A[1,2,,n],B[1,2,m]):
   ```

```
i=1, j=1, k=0, count=0
While ( i<n and j<m)
If (A[i]<B[j])
Set C[k]=A[i] and increment i.
Else
Set C[k]=B[j] and increment j.
If i equals n, copy remaining elements of B to C
If j equals m, Copy remaining elements of A to C

Output count.
```

Explain briefly (in a sentence or two) how to modify the merge procedure to solve the above problem.

4. Given an array $A[1, 2, \ldots, n]$ of distinct elements, an inversion is a pair (i, j) of indices such that $i < j$ and $A[i] > A[j]$. Eg: The sequence 3,8,0,-4,1 has 7 inversions, namely the pairs (1,3), (1,4), (1,5), (2,3), (2,4), (2,5), (3,4).

   Describe an algorithm to count the number of inversions of a given array, and running in $O(n \log n)$ time. Hint: Modify the merge-sort algorithm suitably.