CS3933 Winter 2007 – Group Assignment 1
Due Wednesday September 19, 2007, at 5pm.
Sample Solution

Groups can be of up to 3 people; please list all names of the group members at the top of the assignment.

1. Consider the Stable Matching Problem and its algorithm, as discussed in class.

   Can you construct an input instance such that there is a woman w and a man m where: if the men propose, w will reject m, but if the women propose, w will propose to m? Either construct and demonstrate such an instance, or explain why no such instance exists. You can use the various properties discussed.

   When the men propose, w can only reject m if she has a better offer. So w will end up paired with m', where she prefers m' to m. m' is thus a valid partner for w that she ranks above m.

   When the women propose, w will only propose to m if she is rejected by all men she prefers to m, so she will have been rejected by m'. w will thus end up with a partner that she ranks below m' (either m or someone she ranks below m). However, the women-propose results are women-optimal, where each woman gets her best valid partner. This contradicts with m' being a valid partner for w that she ranks above m.

2. For each of the following running times (assuming each function calculates the total number of steps), determine what is the largest size (value of n) whose computation will finish in 10^6 steps.

   (a) $\log_2 n$
   $n = 2^{(10)^6} \;=\; 2^{(1000000)}$

   (b) $500n$
   $n = 2000$

   (c) $20n^2$
   $n = 223$

   (d) n^6
   $n = 10$

   (e) 2^n
   $n = 19$

   (f) $n!$
   $n = 9$

3. Prove that $2^n \in \Omega(n^2)$, by: applying the definition of $\Omega()$, experimenting to find a suitable c and n_0, and proving the resulting inequality by induction.

   To show that $2^n \in \Omega(n^2)$, we need to show that there are constants $c > 0$, $n_0 \geq 0$ such that $2^n \geq c \cdot n^2$, $\forall n \geq n_0$.

   Let $c = 1$ and $n_0 = 5$ (other pairs are possible).

   Then we need to prove that $2^n \geq n^2$, $\forall n \geq 5$.

   Proof by induction:
   Base case: $n = 5$
   RHS $= n^2 = 5^2 = 25$
   LHS $= 2^n = 2^5 = 32 < 25 =$RHS.

   Inductive step: assume that $2^k \geq k^2$, for some arbitrary $k \geq 5$.
   Prove that $2^{(k+1)} \geq (k+1)^2$.
   Since $k > 3$, $2k + 1 < 3k < k^2$, so $k^2 > 2k + 1$.
   Therefore LHS $= 2^{(k+1)} = 2 \cdot 2^k = 2^k + 2^k \ \geq 2k^2$ (substituting the hypothesis)
   $= k^2 + k^2 > k^2 + 2k + 1 = (k+1)^2 \ =$ RHS.

   Therefore, for all $n \geq 5$, $2^n \geq n^2$, so $2^n \in \Omega(n^2)$.

4. Consider the following algorithm:
   Algorithm *Triangle(A,n)*:
   Input: Array A of n real numbers.

```
Let S be a numerical array of length n.
for i from 0 to n − 1 do
    S[i] ← 0
    for j from 0 to i do
        for k from j to i do
            S[i] ← S[i] + A[k]
return array S
```

   Analyse the running time of this algorithm, using O, Ω, and Θ, as appropriate.

   Upper bound:
   The outer for loop (on i) iterates exactly n times. For each of these iterations, the middle loop (on j) iterates at most n times. For each of these iterations, the inner loop iterates at most n times. The rest of the algorithm runs in constant time, so the algorithm as a whole runs in time $\in O(n^3)$.

Lower bound:
Consider values for i such that $i \geq \lfloor \frac{n}{2} \rfloor$. There are at least $\lfloor \frac{n}{2} \rfloor - 1$ such values for i.
Consider values for j such that $j \leq \lfloor \frac{n}{4} \rfloor$. There are at least $\lfloor \frac{n}{4} \rfloor$ such values for j. All these i and j values satisfy $j \leq i$, as required since j must be between 0 and i.
Since $j \leq \lfloor \frac{n}{4} \rfloor$ and $i \geq \lfloor \frac{n}{2} \rfloor$, there are at least $\lfloor \frac{n}{4} \rfloor$ values for k that satisfy $j \leq k \leq i$, for each pair of these i and j values.

So, the outer loop iterates at least $\lfloor \frac{n}{2} \rfloor - 1$ times (with $i \geq \lfloor \frac{n}{2} \rfloor$); for each of these iterations, the middle loop iterates at least $\lfloor \frac{n}{4} \rfloor$ times (with $j \leq \lfloor \frac{n}{4} \rfloor$); and for each of those iterations, the inner loop iterates at least $\lfloor \frac{n}{4} \rfloor$ times.

Thus the algorithm runs in time $\in \Omega((\lfloor \frac{n}{2} \rfloor - 1) \cdot (\lfloor \frac{n}{4} \rfloor) \cdot (\lfloor \frac{n}{4} \rfloor)) \;=\; \Omega(n^3)$.

Since this algorithm runs in time $\in O(n^3)$ and $\in \Omega(n^3)$, it runs in time $\in \Theta(n^3)$.