CS3933 Winter 2007 – Group Assignment 2
Due Wednesday February 7, 2007, at 5pm.

1. (6 marks) For the overall input S to have an equivalence class of size $> \frac{n}{2}$, that equivalence class must contain more than $\frac{n}{4}$ items from one half of S. For our algorithm to run in $O(n \log n)$ time, we need a linear time combine step.

```
Algorithm Equiv(set S, size n): returns (x, count, flag) where flag = 0 if S' does not exist;
    if S' does exist, flag = 1, count = |S'|, and x is an element of S'
if n > 1
    split S into two halves, S_1 (size ⌈n/2⌉) and S_2 (size ⌊n/2⌋)
    (x_1, c_1, f_1) = Equiv(S_1, ⌈n/2⌉)
    (x_2, c_2, f_2) = Equiv(S_2, ⌊n/2⌋)
    flag = 0
    x = null
    if f_1 = 1
      count = c_1
      for all elements i in S_2
            if i is equivalent to x_1
                count = count + 1
      if count > n/2
            flag = 1
            x = x_1
    if flag = 0 and f_2 = 1
      count = c_2
      for all elements i in S_1
            if i is equivalent to x_2
                count = count + 1
      if count > n/2
            flag = 1
            x = x_2
else (n=1)
    x = sole element of S
    count = 1
    flag = 1
return (x, count, flag)
```

2. You are analysing a divide & conquer algorithm, and have determined that it has the following characteristics:

- The algorithm does not recurse if $n = 1$
- The algorithm splits the data into three parts, each of size $\frac{n}{3}$, in linear time ($\in \Theta(n)$)
- The algorithm recurses four times (each time on a part of size $\frac{n}{3}$)
- the algorithm merges the results from the recursion in linear ($\in O(n)$) time

(a) (6 marks) Complete the asymptotic analysis of this algorithm, by setting up a recurrence for $T(n)$ and solving it. Use the recursion tree analysis method (in which you would sum up all levels of the recursion tree). You can assume that n is a power of 3.

$$T(n) = \begin{cases} 0 & \text{if } n = 1 \\ 4 \cdot T(\frac{n}{3}) + c \cdot n & \text{otherwise} \end{cases}$$

The recursion tree for this recurrence will have deepest level $\log_3 n$, since it divides the size by 3 until it is reduced to 1 . At level k, where level 0 is the root, there will be 4^i nodes, each of size $\frac{n}{3^k}$. Since the amount of work done for each node is linear, then the amount of time needed at level k is

$$W(k) \;=\; 4^k \cdot c \cdot \frac{n}{3^k} \;=\; cn \cdot \left(\frac{4}{3}\right)^k$$

Summing over all levels, the total time taken for the algorithm is

$$\begin{aligned} T(n) &= \sum_{k=0}^{\log_3 n} W(k) \\ &= \sum_{k=0}^{\log_k n} cn \cdot \left(\frac{4}{3}\right)^k \\ &= cn \cdot \left(\sum_{k=0}^{\log_3 n} \left(\frac{4}{3}\right)^k \right) \\ &= cn \cdot \left(\frac{\left(\frac{4}{3}\right)^{\log_3 n+1} - 1}{\frac{4}{3} - 1} \right) \\ &= 3 \cdot \frac{4}{3} cn \left(\frac{4}{3}\right)^{\log_3 n} - 3cn \end{aligned}$$

$$
\begin{aligned}
&= 4cn \cdot n^{\log_3\left(\frac{4}{3}\right)} - 3cn \\
&= 4cn \cdot n^{\log_3 4 - 1} - 3cn \\
&= 4cn^{\log_3 4} - 3cn
\end{aligned}
$$

Since $\log_3 4 > 1$, $T(n) \in \Theta(n^{\log_3 4})$, which is approximately $\Theta(n^{1.26})$.

(b) (4 marks) Consider a change in the algorithm so that the merge of the results takes $\Theta(n^2)$ time. State the new recurrence for $T(n)$ and complete the recursion tree analysis to find the time complexity.

$$
T(n) = \begin{cases} 0 & \text{if } n = 1 \\ 4 \cdot T(\frac{n}{3}) + c \cdot n^2 & \text{otherwise} \end{cases}
$$

The recursion tree for this recurrence will have deepest level $\log_3 n$, since it divides the size by 3 until it is reduced to 1. At level k, where level 0 is the root, there will be 4^i nodes, each of size $\frac{n}{3^k}$. Since the amount of work done for each node is cn^2, then the amount of time needed at level k is

$$
W(k) = 4^k \cdot c \cdot \left(\frac{n}{3^k}\right)^2 = cn^2 \cdot \left(\frac{4}{9}\right)^k
$$

Summing over all levels, the total time taken for the algorithm is

$$
\begin{aligned}
T(n) &= \sum_{k=0}^{\log_3 n} W(k) \\
&= \sum_{k=0}^{\log_k n} cn^2 \cdot \left(\frac{4}{9}\right)^k \\
&= cn^2 \cdot \left(\sum_{k=0}^{\log_3 n} \left(\frac{4}{9}\right)^k\right) \\
&= cn^2 \cdot \left(\frac{1 - \left(\frac{4}{9}\right)^{\log_3 n+1}}{1 - \frac{4}{9}}\right) \\
&= \frac{9}{5} \cdot cn^2 \cdot \left(1 - \left(\frac{4}{9}\right)^{\log_3 n+1}\right) \\
&= \frac{9}{5} \cdot cn^2 - \frac{9}{5} \cdot cn^2 \cdot \left(\frac{4}{9}\right)^{\log_3 n+1}
\end{aligned}
$$

Since $\left(\frac{4}{9}\right)^{\log_3 n+1}$ is a decreasing function, we can conclude that $T(n) \in \Theta(n^2)$.