22C:44 Homework 2 Solutions

1. Consider the following "strange" function:

```
   Strange(n) {
1.     for i ← 1 to n do
2.         if (n mod i == 0) then
3.             for j ← 1 to n do
4.                 print(i, j);
   }
```

Let $T(n)$ be the running time of Strange(n). While the behavior of $T(n)$ is strange, its behavior for certain values of n is easy to predict. For example, when n is a prime, the condition in Line 2 is true only twice, once for $i = 1$ and once for $i = n$. Therefore, for any prime n, $T(n) \leq an$ for some constant $a > 0$. Similarly, suppose that $n = 2^k$, for some non-negative integer k, then n has $(k+1)$ distinct factors $2^0, 2^1, \ldots, 2^{k-1}, 2^k$ and therefore $T(n) \geq b \cdot n \cdot k = bn \lg n$.

(a) **Friend's Claim:** $T(n) = \Theta(n^2)$.
So the friend is claiming that there exist positive constants n_0, c_1, c_2 such that $c_1 n^2 \leq T(n) \leq c_2 n^2$ for all $n \geq n_0$. This means that for any prime $n \geq n_0$, $c_1 n^2 \leq an$. This is equivalent to saying that for any prime $n \geq n_0$, $n \leq a/c_1$. Since there are infinite primes, we can pick a prime $n > \max\{n_0, a/c_1\}$ that will make this claim nonsense.

(b) **Brother's Claim:** $T(n) = \Theta(n)$.
So the friend is claiming that there exist positive constants n_0, c_1, c_2 such that $c_1 n \leq T(n) \leq c_2 n$ for all $n \geq n_0$. This means that for any $n \geq n_0$ that is a power of 2, $bn \lg n \leq c_2 n$. This is equivalent to saying that for any $n \geq n_0$, that is a power of 2, $\lg n \leq c_2/b$. Now choosing $n > \max\{n_0, 2^{c_2/b}\}$ will make this claim nonsense.

(c) Line 1 contains a loop that executes n times and it immediately follows that $T(n) = \Omega(n)$. For any positive integer n, let $f(n)$ be the number of factors of n. Then, Line 1 takes $\Theta(n)$ time, Line 2 takes $\Theta(n)$ time, Line 3 is executed $f(n)$ times for a total of $\Theta(nf(n))$ time, and Line 4 takes $\Theta(nf(n))$. Thus the total running time is $\Theta(nf(n))$. Since $f(n) \leq n$, $T(n) = O(n^2)$.

2. (a) The recurrence is

$$T(n) = \Theta(1) + n[\Theta(1) + T(n-1)] = nT(n-1) + \Theta(n)$$

for any $n \geq 1$ and $T(0) = \Theta(1)$.

(b) After 1 iteration the right hand side of the recurrence expands to

$$T(n) = n[(n-1)T(n-2) + \Theta(n-1)] + \Theta(n) = n(n-1)T(n-2) + \Theta(n(n-1)) + \Theta(n).$$

After 2 iterations the right hand side of the recurrence expands to

$$\begin{aligned} T(n) &= n(n-1)[(n-2)T(n-3) + \Theta(n-2)] + \Theta(n(n-1)) + \Theta(n) \\ &= n(n-1)(n-2)T(n-3) + \Theta(n(n-1)(n-2)) + \Theta(n(n-1)) + \Theta(n). \end{aligned}$$

After k iterations, the right hand side expands to

$$\begin{aligned} T(n) &= n(n-1)\cdots(n-k)T(n-k-1) + \Theta(n(n-1)\cdots(n-k)) \\ &+ \Theta(n(n-1)\cdots(n-(k-1))) + \cdots + \Theta(n(n-1)) + \Theta(n). \end{aligned}$$

Letting $k = (n-1)$ we get

$$\begin{aligned} T(n) &= n!T(0) + \Theta(n!) + \Theta\left(\frac{n!}{1!}\right) + \Theta\left(\frac{n!}{2!}\right) + \cdots + \Theta\left(\frac{n!}{(n-2)!}\right) + \Theta\left(\frac{n!}{(n-1)!}\right) \\ &= n!\Theta(1) + \Theta\left(n! \sum_{i=0}^{n-1} \frac{1}{i!}\right). \end{aligned}$$

Now $\sum_{i=0}^{n-1} 1/i! \geq 1/0! = 1$. An upper bound on this sum can be obtained by recalling that

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \cdots + \frac{x^k}{k!} + \cdots$$

Setting $x = 1$ gives $e = \sum_{i=0}^{\infty} \frac{1}{i!}$ and therefore $\sum_{i=0}^{n-1} 1/i! \leq e$. Using these bounds on the sum we get

$$T(n) = \Theta(n!) + \Theta(n!) = \Theta(n!).$$

(c) `Stranger(A, 1)` prints the $n!$ permutations of the sequence stored in array `A`. When `A` contains the sequence $1, 2, \ldots, n$ this is simply the sequence of all permutations of the first n natural numbers.

3. (a)

```
Merge(A, p, q, r) {
        left ← max{p, r-c+1};
        right ← max{q, r+c};
        Sort(A, left, right);
}
```

Here `Sort` is any sorting function and `Sort(A, left, right)` takes as input the subarray `A[left..right]` and returns it sorted.

Explanation: From the condition that any pair of "out-of-order" elements `A[i]` and `A[j]` satisfy $|i - j| \leq c$ it follows that there are at most c elements in `A[p..r]` that are larger than some element in `A[r+1..q]`. Similarly, there are at most c elements in `A[r+1..q]` that are smaller than some element in `A[p..r]`. In the input to `Merge`, the subarray `A[p..r]` is sorted and the subarray `A[r+1..q]` is sorted. Hence all the elements in `A[p..r]` that are larger than some element in `A[r+1..q]` occur in the subarray `A[r-c+1..r]`. Similarly, all the elements in `A[r+1..q]` that are smaller than some element in `A[p..r]` occur in the subarray `A[r+1..r+c]`. This means that only the $2c$ elements in the subarray `A[r-c+1..r+c]` need to be merged. This can be done by simply sorting this subarray using *any* sorting technique. Since c is a constant, this subarray contains $\Theta(1)$ elements and hence any sorting algorithm takes $\Theta(1)$ time to sort this.

(b) The new `MergeSort` recurrence is $T(n) = 2T(n/2) + \Theta(1)$ for all $n > 1$ and $T(n) = \Theta(1)$ for $n \leq 1$. Using the iteration method and iterating $(k-1)$ times expands the above recurrence to

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + \Theta\left(\sum_{i=0}^{k-1} 2^i\right) = 2^k T\left(\frac{n}{2^k}\right) + \Theta\left(2^k - 1\right).$$

Choose k such that the conditions $n/2^k \leq 1$ and $n/2^{k-1} > 1$ are satisfied. This gives

$$T(n) = \Theta(n)\Theta(1) + \Theta(n) = \Theta(n).$$

4. Solve the following recurrence relations using the *iteration method*. For each problem, assume that $T(n) = \Theta(1)$ for $n \leq 1$ and $T(n)$ for $n > 1$ is given below.

(a) $T(n) = aT(n/b) + \Theta(n)$. Here a and b are positive integers. Iterating $(k-1)$ times and expanding the right hand side of the recurrence gives

$$T(n) = a^k T\left(\frac{n}{b^k}\right) + \Theta\left(n \sum_{i=0}^{k-1} \left(\frac{a}{b}\right)^i\right).$$

Choose k satisfying $n/b^k \le 1$ and $n/b^{k-1} > 1$. This implies that k satisfies

$$n \le b^k < bn \qquad \log_b n \le k < \log_b n + 1.$$

This also implies that for any x,

$$n^{\log_b x} \le x^k < x \cdot n^{\log_b x}.$$

We now consider the cases $a = b$ and $a \ne b$ separately.

Case 1: $a = b$ The above recurrence now simplifies to

$$T(n) = b^k T\left(\frac{n}{b^k}\right) + \Theta(nk).$$

Substituting the inequalities involving k into this we get

$$T(n) = \Theta(n)\Theta(1) + \Theta(n \log n) = \Theta(n \log n).$$

Case 2: $a \ne b$ In this case the above recurrence simplifies to

$$T(n) = a^k T\left(\frac{n}{b^k}\right) + \Theta\left(n\left(\frac{a}{b}\right)^k\right).$$

Substituting the inequalities involving k in the above recurrence we get

$$\begin{aligned} T(n) &= \Theta(n^{\log_b a})\Theta(1) + \Theta\left(n^{\log_b (a/b)+1}\right) \\ &= \Theta(n^{\log_b a}) + \Theta(n^{\log_b a}) = \Theta(n^{\log_b a}). \end{aligned}$$

(b) $T(n) = 5T(n/5) + n^2$. Iterating $(k-1)$ times gives the recurrence

$$T(n) = 5^k T\left(\frac{n}{5^k}\right) + n^2 \sum_{i=0}^{k-1} \frac{1}{5^i}.$$

Choose k satisfying $n/5^k \le 1$ and $n/5^k > 1$, note that $\sum_{i=0}^{k-1} 1/5^i = \Theta(1)$, and substitute to get

$$T(n) = \Theta(n)\Theta(1) + n^2\Theta(1) = \Theta(n^2).$$

(c) $T(n) = T(n/2) + T(n/3) + n$. Iterating once yields

$$T(n) = T\left(\frac{n}{2^2}\right) + 2T\left(\frac{n}{2 \cdot 3}\right) + T\left(\frac{n}{3^2}\right) + \left(\frac{5}{6}\right) n + n.$$

Iterating a second time yields

$$T(n) = T\left(\frac{n}{2^3}\right) + 3T\left(\frac{n}{2^2 \cdot 3}\right) + 3T\left(\frac{n}{2 \cdot 3^2}\right) + T\left(\frac{n}{3^3}\right) + \left(\frac{5}{6}\right)^2 n + \left(\frac{5}{6}\right) n + n.$$

Iterating $(k-1)$ times yields

$$T(n) = \sum_{j=0}^{k} \binom{k}{j} T\left(\frac{n}{2^j 3^{k-j}}\right) + n \sum_{i=0}^{k-1} \left(\frac{5}{6}\right)^i.$$

Here $\binom{k}{j}$ is binomial number that represents the number of ways of choosing j objects from k objects. Now we use the fact that $T(n)$ is monotonically increasing to obtain the following bounds on the summation:

$$\sum_{j=0}^{k} \binom{k}{j} T\left(\frac{n}{3^k}\right) + n \sum_{i=0}^{k-1} \left(\frac{5}{6}\right)^i \leq T(n) \leq \sum_{j=0}^{k} \binom{k}{j} T\left(\frac{n}{2^k}\right) + n \sum_{i=0}^{k-1} \left(\frac{5}{6}\right)^i.$$

We now use the fact that $\sum_{j=0}^{k} \binom{k}{j} = 2^k$ and the fact that

$$1 \leq \sum_{i=0}^{k-1} \left(\frac{5}{6}\right)^i \leq \sum_{i=0}^{\infty} \left(\frac{5}{6}\right)^i = 6$$

to substitute into and simplify the above inequalities to get:

$$2^k T\left(\frac{n}{3^k}\right) + n\Theta(1) \leq T(n) \leq 2^k T\left(\frac{n}{2^k}\right) + n\Theta(1).$$

To simplify the left hand side of the above inequality we chose k such that $n/3^k \leq 1$ and $n/3^{k-1} > 1$. This implies that $n \leq 3^k < 3n$ and $\log_3 n \leq k \leq 1 + \log_3 n$. This further implies that $n^{\log_3 2} \leq 2^k \leq 2 \cdot n^{\log_3 2}$. Substituting into the left hand side of the inequality we get

$$\begin{aligned} \Theta(n^{\log_3 2})\Theta(1) + \Theta(n) &\leq T(n) \\ \Theta(n) &\leq T(n) \end{aligned}$$

To simplify the right hand side of the inequality we chose k such that $n/2^k \leq 1$ and $n/2^{k-1} > 1$. This implies that $n \leq 2^k < 2n$ and substituting into the inequality we get

$$T(n) \leq \Theta(n)\Theta(1) + n\Theta(1) = \Theta(n).$$

The fact that $\Theta(n) \leq T(n) \leq \Theta(n)$ implies that $T(n) = \Theta(n)$.

(d) $T(n) = T(n-2) + 7$. Iterating $(k-1)$ times yields

$$T(n) = T(n-2k) + 7k.$$

Choosing k such that

$$\frac{n-1}{2} \leq k < \frac{n+1}{2}$$

and substituting in the above recurrence gives us

$$T(n) = \Theta(1) + 7\Theta(n) = \Theta(n).$$

(e) $T(n) = nT(n-1) + 1$. Iterating $(k-1)$ times yields

$$T(n) = n(n-1)(n-2)\cdots(n-(k-1))T(n-k) + k.$$

Choosing k such that $n-1 \leq k < n$ yields

$$T(n) = n!T(1) + n = n!\Theta(1) + n = \Theta(n!).$$
