CS 463
Section 01
Spring 2021
Christopher K. Schmitt Assignment 2

Problem 1

The order of growth is $\Theta\left(n^3\right)$. The innermost operation, $RESULT++$, is constant or $\Theta(1)$. It is contained within a loop which is grows linearly with respect to j. This loop is therefore $\Theta(j)$. The middle loop is linear with respect to i, and is therefore $\Theta(i)$. The outermost loop is linear with respect to n, and is therefore $\Theta(n)$. This gives us $\Theta(n * i * j * 1)$, or (with substitution) $\Theta(n * n * n * 1) = \Theta\left(n^3\right)$. This can also be found by solving the summation expression.

$$\sum_{i=0}^{n+1}\sum_{j=2}^{i}\sum_{k=3}^{j-1} 1$$

Simplifying

$$\sum_{i=0}^{n+1}\sum_{j=2}^{i}\sum_{k=3}^{j-1} 1$$

$$\sum_{i=0}^{n+1}\sum_{j=2}^{i} j-3$$

$$\sum_{i=0}^{n+1} \frac{i^2}{2}+\frac{i}{2}-4$$

$$\frac{n^3}{6}+\frac{3n^2}{4}+\frac{13n}{12}-\frac{7}{2}$$

Problem 2 (a)

Proof.
Start by breaking the expression down into a geometric series and a simple sum. Then combine terms.

$$\sum_{i=0}^{n-1} 2i(i+1) = 2\sum_{i=0}^{n-1} i^2 + 2\sum_{i=0}^{n-1} i = \frac{2}{3}(n-1)\cdot(n^2+n)$$

Next, rearrange the expression such that it is a sum of powers, leading with the highest order term.

$$\frac{2}{3}(n-1)\cdot(n^2+n) = \frac{2n^3}{3} - \frac{2n}{3}$$

Since the right-hand side of the sum is both lower-order and negative, it can be ignored, leaving us with $\frac{2}{3}n^3$, therefore the expression is $\Theta\left(n^3\right)$.

□

Problem 2 (b)

Proof.
We start by solving the innermost sum in terms of i. This gives us

$$\sum_{j=0}^{i} i + j = \frac{3i^2 + 3i}{2}$$

Substituting this expression in for the innermost sum and expanding gives

$$\sum_{i=1}^{n+1} \frac{3i^2 + 3i}{2} = \frac{n^3 + 6n^2 + 11n + 6}{2}$$

The highest order term in the polynomial is $\frac{3}{2} \cdot n$. Therefore the order of growth is $\Theta\left(n^3\right)$.

$\square$

Problem 3 (a)

$$\begin{aligned} T(n) = T(n-1) + 10 &= T(n-2) + 20 \\ &= T(n-3) + 30 \\ &= T(n-4) + 40 \\ &\vdots \\ &= T(n-k) + 10k \end{aligned}$$

Setting the initial conditions and solving gives us

$$T(n) = 10n - 10 = \Theta(n)$$

Problem 3 (b)

$$\begin{aligned} T(n) = 2T(n-1) &= 4T(n-2) \\ &= 8T(n-3) \\ &= 16T(n-4) \\ &\vdots \\ &= 2^k T(n-k) \end{aligned}$$

Setting the initial conditions and solving gives us

$$T(n) = 2^{n+1} = \Theta\left(2^{n+1}\right)$$

Problem 3 (c)

$$\begin{aligned}T(n) = T(n-1) + n &= T(n-2) + n - 1 \\ &= T(n-3) + n - 2 \\ &= T(n-4) + n - 3 \\ &\vdots \\ &= T(n-k) + (n-k+1)\end{aligned}$$

Setting the initial conditions and solving gives us

$$T(n) = \frac{n^2 + n}{2} = \Theta\left(n^2\right)$$

Problem 3 (d)

$$\begin{aligned}T(n) = 4T(\frac{n}{2}) &= 16T(\frac{n}{4}) \\ &\vdots \\ &= n^2 T(\frac{n}{(n-1)^2})\end{aligned}$$

Setting the initial conditions and solving gives us

$$T(n) = n^2 = \Theta\left(n^2\right)$$

Problem 3 (e)

This is case one of the master theorem. The relation is in the form $aT(\frac{n}{b}) + f(n)$, where $a = 3$, $b = 3$, and $f(n) = n$. The master theorem tells us that this is $\Theta\left(n^{\log_3 3}\right)$ or $\Theta\left(n^1\right) = \Theta\left(n\right)$

Problem 4

The algorithm recursively divides the list in two, giving us $2T\left(\frac{n}{2}\right)$. Merging two sub-lists takes n time. The recurrence relation is as follows

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

Solving the relation

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2}\right) + n \\ &= 4T\left(\frac{n}{4}\right) + 2n \\ &= 8T\left(\frac{n}{8}\right) + 3n \\ &\vdots \\ &= nT\left(\frac{n}{n}\right) + n\log n \end{aligned}$$

$$T(n) = 2T\left(\frac{n}{2}\right) + n = \Theta\left(n\log n\right)$$