COMP 550 - Assignment 1

Joe Puccio

September 4, 2014

2-3

(a) Because there is no programmatic recursion and only one loop, the answer is $\Theta(n)$.

(b)

$y = 0$
$k = 0$
for $k \leq n$ **do**
    $y \mathrel{+}= a_k \times x^k$
    $k\,{+}{+}$
end for

The running time of this algorithm is also $\Theta(n)$, however it's not as efficient of an algorithm as the implementation of Horner's rule even though they have the same Θ run time (constant differences).

(c) We can show that the invariant is true on the first iteration that actually returns a value for y, which is the $i = n - 1$ case. We see that our invariant results in simply a_n, which we can verify is the same result as our algorithm. Now, for the inductive step, we see that each decrement of i results in another term being added to the sum, specifically if the smallest term added in the previous iteration were a_j, the smallest term added in the next iteration would be a_{j-1}. Additionally, each term that existed in the previous iteration is multiplied by an additional power of x. This is the same result as the algorithm (in both cases, we are starting from the innermost layer and peeling outward, multiplying the inner layers by x at each step). Finally, at the last iteration when the program halts, $i = -1$, which means that y would be equal to $\sum_{k=0}^{n} a_k x^k$, which is exactly what the algorithm computes.

(d) Our code fragment evaluates exactly $a_0 + x(a_1 + x(a_2 + ... + x(a_{n-1} + xa_n)...))$ which, when multiplied out equals $a_0 + a_1x + a_2x^2 + a_3x^3 + ... + a_nx^n$, which is exactly a polynomial characterized by coefficients $a_0, a_1, ..., a_n$.

3-2

A	B	O	o	Ω	ω	Θ
$lg^k n$	n^{ϵ}	yes	yes	no	no	no
n^k	c^n	yes	yes	no	no	no
$\sqrt{n}$	$n^{\sin n}$	no	no	no	no	no
2^n	$2^{n/2}$	no	no	yes	yes	no
n^{logc}	c^{logn}	yes	no	yes	no	yes
$log(n!)$	$log(n^n)$	yes	no	yes	no	yes

3

$$\begin{aligned} T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + n \\ &\leq T(n/2) + T(\frac{n+1}{2}) + n \\ &\leq 2T(\frac{n+1}{2}) + n \end{aligned}$$

and we also have that $T(\frac{n+1}{2}) \leq 2T(\frac{\frac{n+1}{2}+1}{2}) + n$ (we also notice that if this internal pattern in T continues, then adding 1 to n would result in a collapse of these fractions where all the denominators of 2 combine and the result is $n/2^k + 1$. So, plugging in for $T(\frac{n+1}{2})$ and iteratively unpacking we get $T(n) \leq 2^k T(\frac{n}{2^k} + 1) + 2^{k-1} n$, because we know that T(1)=1, and letting $k = log(n)$ we can rewrite and simplify this to $T(n) \leq bn + a \times n \log(n)$, where b and a are constants. And we are done.

Collaborators: Sana Imam, Ryan Allan.