Universidad Nacional de Rosario
Facultad de Ciencias Exactas,
Ingeniería y Agrimensura
Departamento de Ciencias de la
Computación

Estructuras de Datos y Algoritmos 2

Trabajo Práctico 2

Cassinerio Marcos, Cerruti Lautaro

Junio de 2022

1. Instancia de Secuencias para Listas

1.1. mapS

Se tiene la siguiente definición para mapS:

```
mapS f [] = emptyS
mapS f (x:xs) = let (y,ys) = f x ||| mapS f xs
                in y:ys
```

Sea s la secuencia sobre la que se aplica *mapS*, f la función con la cual se aplica *mapS* y n el largo de la secuencia s.

1.1.1. Trabajo

A partir de la definición se puede ver lo siguiente:

$W_{mapS}(0) = c_0$

$W_{mapS}(n) = c_1 + W_f(s_0) + W_{mapS}(n-1)$

Donde la constante c_1 corresponde al Cons de listas y al uso de *let*, $W_f(s_0)$ al Trabajo de la función f y $W_{mapS}(n-1)$ al Trabajo de *mapS*.

Se probará utilizando el método de substitución el costo del mismo. Se adivina que:

$$W_{mapS} \in O(\sum_{i=0}^{n-1} W_f(s_i))$$

Se prueba por inducción.

Demostración.

Para poder probar esto se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}/\forall n \geq n_0$ se cumpla lo siguiente:

$$0 \leq W_{mapS}(n) \leq c\sum_{i=0}^{n-1} W_f(s_i)$$

Caso base: $n = 1$

$$W_{mapS}(1) = c_1 + W_f(s_0) + W_{mapS}(0) = c_1 + W_f(s_0) + c_0$$

$$\leq W_f(s_0)c_1 + W_f(s_0) + W_f(s_0)c_0 \leq (c_1 + 1 + c_0)W_f(s_0) \leq c * W_f(s_0)$$

Donde $c \geq 1 + c_0 + c_1$.

Se define a continuación la Hipótesis Inductiva.

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq W_{mapS}(n) \leq c\sum_{i=0}^{n-1} W_f(s_i)$$

con $n > 1$.

Teniendo esto, se quiere probar que vale para $n+1$.

Por definición de $mapS$ se tiene:

$$W_{mapS}(n+1) = c_1 + W_f(s_0) + W_{mapS}(n)$$

Aplicando la Hipótesis Inductiva se tiene

$$W_{mapS}(n+1) \leq c_1 + W_f(s_0) + c\sum_{i=0}^{n-1} W_f(s_i)$$

Se puede definir una constante c' de forma que $c'W_f(s_0) \geq c_1 + W_f(s_0)$.

Es decir, cualquiera de la forma $c' \geq 1 + \frac{c_1}{W_f(s_0)}$

Volviendo a la ecuación anterior se tiene

$$W_{mapS}(n+1) \leq c_1 + W_f(s_0) + c\sum_{i=0}^{n-1} W_f(s_i) \leq c'W_f(s_0) + c\sum_{i=0}^{n-1} W_f(s_i)$$

Ahora tomando $c \geq c'$ se llega a lo siguiente

$$W_{mapS}(n+1) \leq c'W_f(s_0) + c\sum_{i=0}^{n-1} W_f(s_i) \leq c\sum_{i=0}^{n} W_f(s_i)$$

Tomando una constante c lo suficientemente grade tal que

$$c \geq max(1 + c_0 + c_1, c')$$

y $n \geq n_0$ con $n_0 = 1$ se cumple que

$$0 \leq W_{mapS}(n) \leq c\sum_{i=0}^{n-1} W_f(s_i)$$

y finalmente, aplicando la definición de O, tenemos

$$W_{mapS} \in O(\sum_{i=0}^{n-1} W_f(s_i))$$

$\blacksquare$

1.1.2. Profundidad

A partir de la definición se puede ver lo siguiente:

$S_{mapS}(0) = c_0$

$S_{mapS}(n) = c_1 + max(S_f(s_0), S_{mapS}(n-1))$

Donde la constante c_1 corresponde al Cons de listas y al uso de *let*, $S_f(s_0)$ a la Profundidad de la función f y $S_{mapS}(n-1)$ a la Profundidad de *mapS*.

Se probará utilizando el método de substitución el costo de la misma. Se adivina que:

$$S_{mapS} \in O(\max_{i=0}^{n-1} S_f(s_i) + n)$$

Se prueba por inducción.

Demostración.

Nuevamente se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumpla lo siguiente:

$$0 \leq S_{mapS}(n) \leq c(\max_{i=0}^{n-1} S_f(s_i) + n)$$

Caso base: $n = 1$

$$S_{mapS}(1) = c_1 + max(S_f(s_0), c_0)$$

Aquí se tienen 2 casos:

Caso 1

$max(S_f(s_0), c_0) = S_f(s_0)$

En este caso, $S_f(s_0) = \max_{i=0}^{0} S_f(s_i)$, y se tiene que como $n = 1$, $c_1 = n.c_1$. Si se toma a $c \geq max(c_1, 1)$ resulta en

$$S_{mapS}(1) = c_1 + max(S_f(s_0), c_0) = c_1 n + \max_{i=0}^{0} S_f(s_i)$$

$$\leq c(\max_{i=0}^{0} S_f(s_i) + n)$$

Caso 2

$$max(S_f(s_0), c_0) = c_0$$

De donde se tiene que $S_{mapS}(1) = c_1 + c_0$ que es una constante, por lo que podemos tomar $c \geq c_0 + c_1$ y al ser $n = 1$ concluir que

$$S_{mapS}(1) = c_1 + max(S_f(s_0), c_0) = c_1 + c_0 \leq c(\max_{i=0}^{0} S_f(s_i) + n)$$

Se define a continuación la Hipótesis Inductiva

Se supone que $\exists c \in \mathbb{R}^+$

$$0 \leq S_{mapS}(n) \leq c(\max_{i=0}^{n-1} S_f(s_i) + n)$$

con $n > 1$.

Teniendo esto, se quiere probar que vale para n + 1.

Por definición de mapS se tiene:

$$S_{mapS}(n+1) = c_1 + max(S_f(s_0), S_{mapS}(n))$$

Aplicando la Hipótesis Inductiva se tiene

$$S_{mapS}(n+1) \leq c_1 + max(S_f(s_0), c(\max_{i=0}^{n-1} S_f(s_i) + n))$$

Se vuelven a tener 2 casos

Caso 1

$$max(S_f(s_0), c(\max_{i=0}^{n-1} S_f(s_i) + n)) = S_f(s_o)$$

Ya que $S_f(s_0)$ esta incluido en el conjunto, se tiene

$$S_f(s_0) \leq \max_{i=0}^{n} S_f(s_i)$$

De esta manera se deriva lo siguiente

$$S_{mapS}(n+1) \leq c_1 + S_f(s_0) = c_1 + \max_{i=0}^{n} S_f(s_i)$$

$$\leq c_1(n+1) + \max_{i=0}^{n} S_f(s_i) \leq c'(\max_{i=0}^{n} S_f(s_i) + (n+1))$$

Tomando $c' \geq max(c_1, 1)$

De esta forma tenemos la cota buscada.

Caso 2

$$max(S_f(s_0), c(\max_{i=0}^{n-1} S_f(s_i) + n)) = c(\max_{i=0}^{n-1} S_f(s_i) + n)$$

De aquí se llega a la siguiente ecuación

$$S_{mapS}(n+1) \leq c_1 + c(\max_{i=0}^{n-1} S_f(s_i) + n)$$

Trivialmente

$$\max_{i=0}^{n-1} S_f(s_i) \leq \max_{i=0}^{n} S_f(s_i)$$

Por lo que se puede seguir la ecuación de la siguiente manera

$$S_{mapS}(n+1) \leq c_1 + c(\max_{i=0}^{n} S_f(s_i) + n) \leq c + c(\max_{i=0}^{n} S_f(s_i) + n)$$
$$= c(1 + \max_{i=0}^{n} S_f(s_i) + n) = c(\max_{i=0}^{n} S_f(s_i) + (n+1))$$

Con $c \geq c_1$

Asi llegando a la cota buscada.

De esta forma, tomando $c \geq max(c_0 + c_1, 1)$

y $n \geq n_0$ con $n_0 = 1$ se cumple que

$$0 \leq S_{mapS}(n) \leq c(\max_{i=0}^{n-1} S_f(s_i) + n)$$

Y aplicando la definición de O, se llega a

$$S_{mapS} \in O(\max_{i=0}^{n-1} S_f(s_i) + n)$$

■

1.2. appendS

Se tiene la siguiente definición para *appendS*:

```
appendS [] ys = ys
appendS xs [] = xs
appendS (x:xs) ys = x:(appendS xs ys)
```

Para el calculo del Trabajo y de la Profundidad, definiremos la recurrencia respecto al largo de la primer lista.

Sea n el largo de la primer lista y ys la segunda lista.

1.2.1. Trabajo

A partir de la definición se puede ver lo siguiente:

$W_{appendS}(0) = c_0$

$W_{appendS}(n) = c_0$ (si $|ys| = 0$)

$W_{appendS}(n) = c_1 + W_{appendS}(n-1)$ (si $|ys| > 0$)

Suponemos que $|ys| > 0$ ya que en caso contrario siempre se tiene un Trabajo de orden constante.

Se probará utilizando el método de substitución el costo del mismo. Se adivina que:

$$W_{appendS} \in O(n)$$

Se prueba por inducción.

Demostración.

Se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumpla lo siguiente:

$$0 \leq W_{appendS}(n) \leq cn$$

Caso base: $n = 1$

$$W_{appendS}(1) = c_1 + W_{appendS}(0) = c_1 + c_0$$

Tomando $c \geq c_1 + c_0$ se tiene

$$0 \leq W_{appendS}(1) \leq cn$$

Se define a continuación la Hipótesis Inductiva.

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq W_{appendS}(n) \leq cn$$

con n >1.

Teniendo esto, se quiere probar que vale para $n + 1$.

Por definición de *appendS* se tiene:

$$W_{appendS}(n+1) = c_1 + W_{appendS}(n)$$

Aplicando la Hipótesis Inductiva se tiene

$$W_{appendS}(n+1) \leq c_1 + c.n$$

Tomando $c \geq c_1$

$$W_{appendS}(n+1) \leq c_1 + cn \leq c(n+1)$$

Con esto se llegó a lo que se quería probar.

Con $c \geq c_1 + c_0$ y $n \geq n_0$ con $n_0 = 1$ se llegó a que

$$0 \leq W_{appendS}(n) \leq cn$$

Y finalmente aplicando la definición de O, se tiene

$$W_{appendS} \in O(n)$$

■

1.2.2. Profundidad

A partir de la definición se puede ver lo siguiente:

$S_{appendS}(0) = c_0$

$S_{appendS}(n) = c_0$ (si $|ys| = 0$)

$S_{appendS}(n) = c_1 + S_{appendS}(n-1)$ (si $|ys| > 0$)

Podemos ver que las ecuaciones de la Profundidad son iguales a las del Trabajo y no hay nada paralelizable. Ya se demostró que $W_{appendS}$ esta acotado superiormente por O(n), por lo que

$$S_{appendS} \in O(n)$$

1.3. reduceS

Se tienen las siguientes definiciones para *reduceS*:

```
contraerSL f [] = []
contraerSL f l@[x] = l
contraerSL f (x:y:xs) = let (z, zs) =  f x y ||| contraerSL f xs
                        in z:zs
reduceS f a [] = a
reduceS f a [x] = f a x
reduceS f a xs = reduceS f a (contraerSL f xs)
```

Sea s la secuencia sobre la cual se aplica *reduceS*, f la función con la cual se aplica *reduceS*, a el valor por defecto y n el largo de la secuencia s.

Se supone $W_f \in O(1)$ y $S_f \in O(1)$

1.3.1. Trabajo

A partir de la definición se tiene:

$W_{reduceS}(0) = c_0$

$W_{reduceS}(1) = W_f(a, s_0) = c_f$

$W_{reduceS}(n) = c_1 + W_{reduceS}(\left\lceil \frac{n}{2} \right\rceil) + W_{contraerSL}(n)$

Primero se analizará el Trabajo de *contraerSL*. A partir de la definición se tiene:

$W_{contraerSL}(0) = c_2$

$W_{contraerSL}(1) = c_3$

$W_{contraerSL}(n) = c_4 + W_f(s_0, s_1) + W_{contraerSL}(n-2) = c_4 + c_f + W_{contraerSL}(n-2)$

Se demostrara que

$$W_{contraerSL} \in \Theta(n)$$

Demostración.

$\underline{W_{contraerSL} \in O(n)}$

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq W_{contraerSL}(n) \leq cn$$

Caso Base: $n = 1$

$$W_{contraerSL}(1) = c_3 \leq cn$$

Tomando $c \geq c_3$ se tiene

$$0 \leq W_{contraerSL}(1) \leq cn$$

Caso Base: $n = 2$

$$W_{contraerSL}(2) = c_4 + c_f + W_{contraerSL}(0) = c_4 + c_f + c_2 < (c_4 + c_f + c_2)2 \leq c2 = cn$$

Tomando $c \geq c_4 + c_f + c_2$ se tiene

$$0 \leq W_{contraerSL}(2) \leq cn$$

Se define la Hipótesis Inductiva como

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq W_{contraerSL}(n) \leq cn$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de contraer se tiene

$$W_{contraerSL}(n+1) = c_4 + c_f + W_{contraerSL}(n-1)$$

aplicando la Hipótesis Inductiva

$$W_{contraerSL}(n+1) \leq c_4 + c_f + c(n-1) = c_4 + c_f + cn - c$$

Tomando $c \geq max(c_4, c_f)$

$$W_{contraerSL}(n+1) \leq c + c + cn - c = cn + c = c(n+1)$$

Con esto se llegó a lo que se quería probar.

Con $c \geq max(c_4 + c_f + c_2, c_3)$ y $n \geq n_0$ con $n_0 = 1$ se llega a que

$$0 \leq W_{contraerSL}(n) \leq cn$$

Y finalmente aplicando la definición de O

$$W_{contraerSL} \in O(n)$$

<u>$W_{contraerSL} \in \Omega(n)$</u>

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq cn \leq W_{contraerSL}(n)$$

Caso Base: $n = 1$

$$W_{contraerSL}(1) = c_3 \geq cn$$

Tomando $c \leq c_3$ se tiene

$$0 \leq cn \leq W_{contraerSL}(1)$$

Caso Base: $n = 2$

$$W_{contraerSL}(2) = c_4 + c_f + W_{contraerSL}(0) = c_4 + c_f + c_2 \geq c2 = cn$$

Tomando $c \leq \frac{(c_4+c_f+c_2)}{2}$ se tiene

$$0 \leq cn \leq W_{contraerSL}(2)$$

Ahora se define la Hipótesis Inductiva de la siguiente manera

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq cn \leq W_{contraerSL}(n)$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de contraer se tiene

$$W_{contraerSL}(n+1) = c_4 + c_f + W_{contraerSL}(n-1)$$

Aplicando la Hipótesis Inductiva

$$W_{contraerSL}(n+1) \geq c_4 + c_f + c(n-1) = c_4 + c_f + cn - c$$

Tomando $c \leq min(c_4, c_f)$

$$W_{contraerSL}(n+1) \geq c + c + cn - c = cn + c = c(n+1)$$

Con esto se llegó a lo que se quería probar. Con $c \leq min(c_4, c_f, c_3, \frac{c_4+c_f+c_2}{2})$ y $n \geq n_0$ con $n_0 = 1$ se llego a que

$$0 \leq cn \leq W_{contraerSL}(n)$$

Aplicando la definición de Ω

$$W_{contraerSL} \in \Omega(n)$$

$$\therefore W_{contraerSL} \in \Theta(n)$$

■

Estando esto demostrado se puede seguir con el Trabajo de *reduceS*

Demostración.

Se supone $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$W_{reduceS}(n) = c_1 + W_{reduceS}(\left\lceil \frac{n}{2} \right\rceil) + W_{contraerSL}(n) = c_1 + W_{reduceS}(\frac{n}{2}) + W_{contraerSL}(n)$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el tercer caso del mismo Tomando $a = 1$, $b = 2$ y $f' = W_{contraerSL}(n) + c_1$

Como $W_{contraerSL} \in \Theta(n)$, trivialmente $f' \in \Theta(n)$

$\exists e > 0, e = 1 : f' \in \Omega(n^{log_b a + e}) = \Omega(n^{log_2 1+1}) = \Omega(n^{0+1}) = \Omega(n)$

y

$\exists c < 1, N \in \mathbb{N}, N > 1, \forall n > N, af'(\frac{n}{b}) \leq cf'(n)$

Si y solo si $W_{contraerSL}(\frac{n}{2}) \leq cW_{contraerSL}(n)$

$$W_{contraerSL}(\frac{n}{2}) = \frac{n}{2} = \frac{1}{2}n \leq cW_{contraerSL}(n)$$

$$\therefore W_{reduceS} \in \Theta(f') = \Theta(2^k)$$

Como $W_{reduceS}$ es eventualmente no decreciente y es suave, podemos concluir que

$$W_{reduceS} \in \Theta(n)$$

■

1.3.2. Profundidad

A partir de la definición se tiene:

$S_{reduceS}(0) = c_0$

$S_{reduceS}(1) = S_f(a, s_0) = c_f$

$S_{reduceS}(n) = c_1 + S_{reduceS}(\lceil \frac{n}{2} \rceil) + S_{contraerSL}(n)$

Nuevamente se analizará la Profundidad de *contraerSL*. A partir de la definición se tiene:

$S_{contraerSL}(0) = c_2$

$S_{contraerSL}(1) = c_3$

$S_{contraerSL}(n) = c_4 + \text{máx}(S_f(s_0, s_1), S_{contraerSL}(n-2)) = c_4 + \text{máx}(c_f, S_{contraerSL}(n-2))$

Se demostrará que

$$S_{contraerSL} \in \Theta(n)$$

Demostración.

$\underline{S_{contraerSL} \in O(n)}$

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq S_{contraerSL}(n) \leq cn$$

Caso Base: $n = 1$

$$S_{contraerSL}(1) = c_3 \leq cn$$

Tomando $c \geq c_3$ se tiene

$$0 \leq S_{contraerSL}(1) \leq cn$$

Caso Base: $n = 2$

$$S_{contraerSL}(2) = c_4 + max(c_f, S_{contraerSL}(0)) = c_4 + max(c_f, c_2)$$

Sea $c \geq max(c_f, c_4, c_2)$
$$0 \leq S_{contraerSL}(2) \leq c + c = c2 = cn$$

Se define la Hipótesis Inductiva

Se supone que $\exists c \in \mathbb{R}^+$
$$0 \leq S_{contraerSL}(n) \leq cn$$

con $n > 2$.

Teniendo esto, se quiere probar para $n + 1$.

Por definición de contraer

$$S_{contraerSL}(n + 1) = c_4 + \text{máx}(c_f, S_{contraerSL}(n - 1))$$

Aplicando la Hipótesis Inductiva

$$S_{contraerSL}(n + 1) \leq c_4 + max(c_f, c(n - 1))$$

Tomando $c \geq max(c_f, c_4)$

$$S_{contraerSL}(n + 1) \leq c_4 + max(c_f, c(n - 1))$$

$$= c_4 + c(n - 1) \leq c_4 + cn \leq c + cn = c(n + 1)$$

Con esto se llego a lo que se quería probar.

Con $c \geq max(c_f, c_2, c_3, c_4)$ y $n \geq n_0$ con $n_0 = 1$ se llega a

$$0 \leq S_{contraerSL}(n) \leq cn$$

Y por definición de O
$$S_{contraerSL} \in O(n)$$

$\underline{S_{contraerSL} \in \Omega(n)}$

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}/\forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq cn \leq S_{contraerSL}(n)$$

Caso Base: $n = 1$
$$S_{contraerSL}(1) = c_3 \geq cn$$

Tomando $c \leq c_3$ se tiene
$$0 \leq cn \leq S_{contraerSL}(1)$$

Caso Base: $n = 2$

$$S_{contraerSL}(2) = c_4 + max(c_f, S_{contraerSL}(0)) = c_4 + max(c_f, c_2) \geq c2 = cn$$

Tomando $c \leq \frac{c_4 + max(c_f, c_2)}{2}$ se tiene

$$0 \leq cn \leq S_{contraerSL}(2)$$

Ahora se define la Hipótesis Inductiva de la siguiente manera Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq cn \leq S_{contraerSL}(n)$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de contraer se tiene

$$S_{contraerSL}(n+1) = c_4 + max(c_f, S_{contraerSL}(n-1))$$

Aplicando la Hipotesis Inductiva

$$S_{contraerSL}(n+1) \geq c_4 + max(c_f, c(n-1))$$

Aqui se tienen 2 casos

Caso 1: $max(c_f, c(n-1)) = c_f$

Tomando $c \leq \frac{c_4}{2}$

$$S_{contraerSL}(n+1) \geq c_4 + c_f \geq c_4 + c(n-1) \geq 2c + c(n-1) = c(n+1)$$

Caso 2: $max(c_f, c(n-1)) = c(n-1)$

Tomando $c \leq \frac{c_4}{2}$

$$S_{contraerSL}(n+1) \geq c_4 + c(n-1) \geq 2c + c(n-1) = c(n+1)$$

Con un valor de $c \leq min(c_3, \frac{c_4}{2})$ y $n \geq n_0$ con $n_0 = 1$ queda demostrado que

$$0 \leq cn \leq S_{contraerSL}(n)$$

Aplicando definición de Ω

$$S_{contraerSL} \in \Omega(n)$$

$$\therefore S_{contraerSL} \in \Theta(n)$$

■

Continuando con la Profundidad de *reduceS*, se ve que esta no paraleliza nada con respecto al Trabajo, y como la Profundidad de *contraerSL* es del mismo orden que su Trabajo, la demostración de la Profundidad de *reduceS* se realiza aplicando el Teorema Maestro de forma análoga a su Trabajo. Quedando de la siguiente manera

$$S_{reduceS} \in \Theta(n)$$

1.4. scanS

Se tienen las siguientes definiciones para *scanS*:

```
contraerSL f [] = []
contraerSL f l@[x] = l
contraerSL f (x:y:xs) = let (z, zs) =  f x y ||| contraerSL f xs
                        in z:zs

expandirSL _ [] _ = []
expandirSL _ [_] ys = ys
expandirSL f (x:_:xs) (y:ys) = let (z, zs) = (f y x) ||| expandirSL f xs ys
                               in y:z:zs

scanS f a [] = ([], a)
scanS f a [x] = (singletonS a, f a x)
scanS f a xs = let (is, r) = scanS f a (contraerSL f xs)
               in (expandirSL f xs is, r)
```

Sea f la función con la cual se aplica *reduceS*, a el valor por defecto, s la secuencia sobre la cual se aplica *reduceS* y n el largo de la secuencia s. Para la función *expandirSL* también definiremos una secuencia q que sera la devuelta por *scanS* como primer elemento de la tupla.

Para el calculo del Trabajo y de la Profundidad de *expandirSL*, definiremos la recurrencia respecto al largo de la primer lista.

Podemos ver que la función *contraerSL* es la misma previamente utilizada en *reduceS* y sus costos ya fueron demostrados.

Se supone $W_f \in O(1)$ y $S_f \in O(1)$

1.4.1. Trabajo

A partir de la definición se tiene:

$W_{scanS}(0) = c_0$

$W_{scanS}(1) = c_1 + W_f(a, s_0) = c_1 + c_f$

$W_{scanS}(n) = c_2 + W_{expandirSL}(n) + W_{scanS}(\lceil \frac{n}{2} \rceil) + W_{contraerSL}(n)$

Primero se analizara el Trabajo de *expandirSL*. A partir de la definición se tiene:

$W_{expandirSL}(0) = c_3$

$W_{expandirSL}(1) = c_4$

$W_{expandirSL}(n) = W_f(s_0, q_0) + W_{expandirSL}(n-2) + c_5 = c_5 + c_f + W_{expandirSL}(n-2)$

Se demostrara que

$$W_{expandirSL}(n) \in \Theta(n)$$

Demostración.

<u>$W_{expandirSL} \in O(n)$</u>

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}/\forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq W_{expandirSL}(n) \leq cn$$

Caso Base: $n = 1$

$$W_{expandirSL}(1) = c_4 \leq cn$$

Tomando $c \geq c_4$ se tiene

$$0 \leq W_{expandirSL}(1) \leq cn$$

Caso Base: $n = 2$

$$W_{expandirSL}(2) = c_f + c_5 + W_{expandirSL}(0) = c_5 + c_f + c_3 < (c_5 + c_f + c_3)2 \leq c2 = cn$$

Tomando $c \geq c_5 + c_f + c_3$ se tiene

$$0 \leq W_{expandirSL}(2) \leq cn$$

Se define la Hipótesis Inductiva como

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq W_{expandirSL}(n) \leq cn$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de expandir se tiene

$$W_{expandirSL}(n + 1) = c_5 + c_f + W_{expandirSL}(n - 1)$$

Aplicando la Hipótesis Inductiva

$$W_{expandirSL}(n + 1) \leq c_5 + c_f + c(n - 1) = c_5 + c_f + cn - c$$

Tomando $c \geq max(c_5, c_f)$

$$W_{expandirSL}(n + 1) \leq c + c + cn - c = cn + c = c(n + 1)$$

Con esto se llegó a lo que se quería probar.

Con $c \geq max(c_4, c_5 + c_f + c_2)$ y $n \geq n_0$ con $n_0 = 1$ se llega a que

$$0 \leq W_{expandirSL}(n) \leq cn$$

Y finalmente aplicando la definición de O

$$W_{expandirSL} \in O(n)$$

$W_{expandirSL} \in \Omega(n)$

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}/\forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq cn \leq W_{expandirSL}(n)$$

Caso Base: $n = 1$

$$W_{expandirSL}(1) = c_4 \geq cn$$

Tomando $c \leq c_4$ se tiene

$$0 \leq cn \leq W_{expandirSL}(1)$$

Caso Base: $n = 2$

$$W_{expandirSL}(2) = c_5 + c_f + W_{expandirSL}(0) = c_5 + c_f + c_3 \geq c2 = cn$$

Tomando $c \leq \frac{(c_5+c_f+c_3)}{2}$ se tiene

$$0 \leq cn \leq W_{expandirSL}(2)$$

Ahora se define la Hipótesis Inductiva de la siguiente manera

Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq cn \leq W_{expandirSL}(n)$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de contraer se tiene

$$W_{expandirSL}(n+1) = c_5 + c_f + W_{expandirSL}(n-1)$$

Aplicando la Hipótesis Inductiva

$$W_{expandirSL}(n+1) \geq c_5 + c_f + c(n-1) = c_5 + c_f + cn - c$$

Tomando $c \leq min(c_5, c_f)$

$$W_{expandirSL}(n+1) \geq c + c + cn - c = cn + c = c(n+1)$$

Con esto se llegó a lo que se quería probar. Con $c \leq min(c_4, c_5, c_f, \frac{c_5+c_f+c_3}{2})$ y $n \geq n_0$ con $n_0 = 1$ se llego a que

$$0 \leq cn \leq W_{expandirSL}(n)$$

Aplicando la definición de Ω

$$W_{expandirSL} \in \Omega(n)$$

$$\therefore W_{expandirSL} \in \Theta(n)$$

$\blacksquare$

Estando esto demostrado se puede seguir con el Trabajo de *scanS*

Demostración.

Se supone que $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$W_{scanS}(n) = c_2 + W_{expandirSL}(n) + W_{contraerSL}(n) + W_{scanS}(\left\lceil \frac{n}{2} \right\rceil)$$
$$= c_2 + W_{expandirSL}(n) + W_{contraerSL}(n) + W_{scanS}(\frac{n}{2})$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el tercer caso del mismo

Tomando $a = 1, b = 2, f' = W_{expandirSL}(n) + W_{contraerSL}(n) + c_2$

Como $W_{contraerSL} \in \Theta(n)$ y $W_{expandirSL} \in \Theta(n)$, trivialmente $f' \in \Theta(n)$

Se puede ver que

$\exists e > 0, e = 1 : f' \in \Omega(n^{log_b a+e}) = \Omega(n^{log_2 1+1}) = \Omega(n^{0+1}) = \Omega(n)$

y

$\exists c < 1, N \in \mathbb{N}, N > 1, \forall n > N, af'(\frac{n}{b}) \leq cf'(n)$

si y solo si $W_{contraerSL}(\frac{n}{2}) + W_{expandirSL}(\frac{n}{2}) + c_2 \leq c(W_{contraerSL}(n) + W_{expandirSL}(n) + c_2)$

$$W_{contraerSL}(\frac{n}{2}) + W_{expandirSL}(\frac{n}{2}) + c_2$$
$$\leq \frac{W_{contraerSL}(n) + W_{expandirSL}(n) + c_2}{2} = c(W_{contraerSL}(n) + W_{expandirSL}(n) + c_2)$$

Con $c = \frac{1}{2}$

$$\therefore W_{scanS} \in \Theta(f') = \Theta(2^k)$$

Como W_{scanS} es eventualmente no decreciente y $f'(n) = n$ es suave, podemos concluir que

$$W_{scanS} \in \Theta(n)$$

■

1.4.2. Profundidad

A partir de la definición se tiene:

$S_{scanS}(0) = c_0$

$S_{scanS}(1) = c_1 + S_f(a, s_0) = c_1 + c_f$

$S_{scanS}(n) = c_2 + S_{expandirSL}(n) + S_{scanS}(\left\lceil \frac{n}{2} \right\rceil) + S_{contraerSL}(n)$

Primero se analizara la Profundidad de $expandirSL$. A partir de la definición

$S_{expandirSL}(0) = c_3$

$S_{expandirSL}(1) = c_4$

$S_{expandirSL}(n) = \text{máx}(S_f(s_0, q_0), S_{expandirSL}(n-2)) + c_5 = c_5 + \text{máx}(c_f, S_{expandirSL}(n-2))$

Se demostrará que

$$S_{contraerSL} \in \Theta(n)$$

Demostración.

$\underline{S_{expandirSL} \in O(n)}$

Primero se busca probar que para algun $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}$

$$0 \leq S_{expandirSL}(n) \leq cn$$

Caso Base: $n = 1$

$$S_{expandirSL}(1) = c_4 \leq cn$$

Tomando $c \geq c_4$ se tiene

$$0 \leq S_{expandirSL}(1) \leq cn$$

Caso Base: $n = 2$

$$S_{expandirSL}(2) = c_5 + max(c_f, S_{expandirSL}(0)) = c_5 + max(c_f, c_3)$$

Sea $c \geq max(c_f, c_5, c_3)$

$$0 \leq S_{expandirSL}(2) \leq c + c = c2 = cn$$

Se define la Hipótesis Inductiva

Se supone que $\exists c \in \mathbb{R}^+$

$$0 \leq S_{expandirSL}(n) \leq cn$$

con $n > 2$.

Teniendo esto, se quiere probar para $n + 1$.

Por definición de expandir

$$S_{expandirSL}(n+1) = c_5 + \text{máx}(c_f, S_{expandirSL}(n-1))$$

Aplicando la Hipótesis Inductiva

$$S_{expandirSL}(n+1) \leq c_5 + max(c_f, c(n-1))$$

Tomando $c \geq max(c_f, c_5)$

$$S_{expandirSL}(n+1) \leq c_5 + max(c_f, c(n-1))$$

$$= c_5 + c(n-1) \leq c_5 + cn \leq c + cn = c(n+1)$$

Con esto se llego a lo que se quería probar.

Con $c \geq max(c_f, c_5, c_3, c_4)$ y $n \geq n_0$ con $n_0 = 1$ se llega a

$$0 \leq S_{expandirSL}(n) \leq cn$$

Y por definición de O

$$S_{expandirSL} \in O(n)$$

$\underline{S_{expandirSL} \in \Omega(n)}$

Primero se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumple lo siguiente:

$$0 \leq cn \leq S_{expandirSL}(n)$$

Caso Base: $n = 1$

$$S_{expandirSL}(1) = c_4 \geq cn$$

Tomando $c \leq c_4$ se tiene

$$0 \leq cn \leq S_{expandirSL}(1)$$

Caso Base: $n = 2$

$$S_{expandirSL}(2) = c_5 + max(c_f, S_{expandirSL}(0)) = c_5 + max(c_f, c_3) \geq c2 = cn$$

Tomando $c \leq \frac{c_5 + max(c_f, c_3)}{2}$ se tiene

$$0 \leq cn \leq S_{expandirSL}(2)$$

Ahora se define la Hipótesis Inductiva de la siguiente manera Se supone que $\exists c \in \mathbb{R}^+$ tal que

$$0 \leq cn \leq S_{expandirSL}(n)$$

con $n > 2$

Se prueba para $n + 1$.

Por definición de contraer se tiene

$$S_{expandirSL}(n+1) = c_5 + max(c_f, S_{expandirSL}(n-1))$$

Aplicando la Hipótesis Inductiva

$$S_{expandirSL}(n+1) \geq c_5 + max(c_f, c(n-1))$$

Aqui se tienen 2 casos

Caso 1: $max(c_f, c(n-1)) = c_f$

Tomando $c \leq \frac{c_5}{2}$

$$S_{expandirSL}(n+1) \geq c_5 + c_f \geq c_5 + c(n-1) \geq 2c + c(n-1) = c(n+1)$$

Caso 2: $max(c_f, c(n-1)) = c(n-1)$

Tomando $c \leq \frac{c_5}{2}$

$$S_{expandirSL}(n+1) \geq c_5 + c(n-1) \geq 2c + c(n-1) = c(n+1)$$

Con un valor de $c \leq min(c_4, \frac{c_5}{2})$ y $n \geq n_0$ con $n_0 = 1$ queda demostrado que

$$0 \leq cn \leq S_{expandirSL}(n)$$

Aplicando la definición de Ω

$$S_{expandirSL} \in \Omega(n)$$

$$\therefore S_{expandirSL} \in \Theta(n)$$

■

Continuando con la Profundidad de $scanS$, se ve que esta no paraleliza nada con respecto al Trabajo, y como las Profundidades de $contraerSL$ y de $expandirSL$ son del mismo orden que sus Trabajos respectivamente, la demostración de la Profundidad de $scanS$ se realiza aplicando el Teorema Maestro de forma análoga a su Trabajo. Quedando de la siguiente manera

$$S_{scanS} \in \Theta(n)$$

2. Instancia de Secuencias para Arrays

2.1. mapS

Se tienen las siguientes definiciones:

```
tabulateS = A.tabulate

nthS = (!)

lengthS = A.length

mapS f xs = tabulateS (\i -> f (nthS xs i)) (lengthS xs)
```

Sea s la secuencia sobre la que se aplica $mapS$, f la función con la cual se aplica $mapS$ y n el largo de la secuencia s. Definimos $f' = \lambda i \rightarrow f(nthS\ s\ i) = \lambda i \rightarrow f((!)\ s\ i)$

2.1.1. Trabajo

A partir de la definición se tiene

$W_{mapS}(n) = W_{lengthS}(n) + W_{tabulateS}(f', n) = W_{A.length}(n) + W_{A.tabulate}(f', n)$

A partir de la tabla de costos ya conocidos se puede ver que

$W_{A.length}(n) \in O(1)$

$W_{A.(!)}(i) \in O(1), \forall i \in \mathbb{N} : 0 \leq i < n$

$W_{A.tabulate}(f', n) \in O(\sum_{i=0}^{n-1} W_{f'}(i))$

Vamos a demostrar que

$$W_{mapS} \in O(\sum_{i=0}^{|s|-1} W_f(s_i))$$

Demostración.

Primero se analizará el Trabajo de f'

$$W_{f'}(i) = W_f(s_i) + W_{nthS}(i)$$

Como sabemos que $WnthS \in O(1)$

$$W_{f'}(i) = W_f(s_i) + c_{nthS}$$

y tomando $c \geq 1 + c_{nthS}$ se tiene

$$W_{f'}(i) = W_f(s_i) + c_{nthS} \leq c.W_f(s_i)$$

$\forall i \in \mathbb{N}$

Por lo que por definición de O, $W_{f'}(i) \in O(W_f(s_i))$

Ya habiendo hecho esto, para poder probar el Trabajo de $mapS$ se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n \geq n_0$ se cumpla lo siguiente

$$0 \leq W_{mapS}(n) \leq c(\sum_{i=0}^{n-1} W_f(s_i))$$

Se comienza con

$$W_{mapS}(n) = W_{A.length}(n) + W_{A.tabulate}(f', n) = c_0 + W_{A.tabulate}(f', n)$$

$$= c_0 + \sum_{i=0}^{n-1} W_{f'}(i) \leq c' \sum_{i=0}^{n-1} W_{f'}(i)$$

Tomando $c' \geq c_0 + 1$ y como $W_{f'}(i) \in O(W_f(s_i))$

$$W_{mapS}(n) \leq c' \sum_{i=0}^{n-1} W_{f'}(i) \leq c' \sum_{i=0}^{n-1} c'' W_f(s_i) = c'c'' \sum_{i=0}^{n-1} W_f(s_i)$$

Finalmente, con $c \geq c'c''$

$$W_{mapS}(n) \leq c \sum_{i=0}^{n-1} W_f(s_i)$$

Con $n \geq n_0$ tal que $n_0 = 0$ se cumple

$$0 \leq W_{mapS}(n) \leq c \sum_{i=0}^{n-1} W_f(s_i)$$

y finalmente, aplicando la definición de O, tenemos

$$W_{mapS} \in O(\sum_{i=0}^{n-1} W_f(s_i))$$

■

2.1.2. Profundidad

A partir de la definición se tiene

$S_{mapS}(n) = S_{lengthS}(n) + S_{tabulateS}(f', n) = S_{A.length}(n) + S_{A.tabulate}(f', n)$

A partir de la tabla de costos ya conocidos se puede ver que

$S_{A.length}(n) \in O(1)$

$S_{A.(!)}(i) \in O(1), \forall i \in \mathbb{N} : 0 \leq i < n$

$S_{A.tabulate}(f', n) \in O(\max_{i=0}^{n-1} S_{f'}(i))$

Vamos a demostrar que

$$S_{mapS} \in O(\max_{i=0}^{|s|-1} S_f(s_i))$$

Demostración.

Al igual que en el Trabajo de $mapS$, siendo

$$S_{f'}(i) = S_f(s_i) + S_{nthS}(i)$$

ya que las Profundidades de f y de $nthS$ son iguales a sus Trabajos, se tiene que

$$S_{f'} \in O(S_f(s_i))$$

Se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N}/\forall n \geq n_0$ se cumpla lo siguiente

$$0 \leq S_{mapS}(n) \leq c(\max_{i=0}^{n-1} S_f(s_i))$$

Se comienza con

$$S_{mapS}(n) = S_{A.length}(n) + S_{A.tabulate}(f', n) = c_0 + S_{A.tabulate}(f', n)$$

$$= c_0 + \max_{i=0}^{n-1} S_{f'}(i) \leq c' \max_{i=0}^{n-1} S_{f'}(i)$$

Tomando $c' \geq c_0 + 1$ y como $S_{f'}(i) \in O(S_f(s_i))$

$$S_{mapS}(n) \leq c' \max_{i=0}^{n-1} S_{f'}(i) \leq c' \max_{i=0}^{n-1} c'' S_f(s_i) = c'c'' \max_{i=0}^{n-1} S_f(s_i)$$

Finalmente, con $c \geq c'c''$

$$S_{mapS}(n) \leq c \max_{i=0}^{n-1} S_f(s_i)$$

Con $n \geq n_0$ tal que $n_0 = 0$ se cumple

$$0 \leq S_{mapS}(n) \leq c \max_{i=0}^{n-1} S_f(s_i)$$

y finalmente, aplicando la definición de O, tenemos

$$S_{mapS} \in O(\max_{i=0}^{n-1} S_f(s_i))$$

■

2.2. appendS

Se tienen las siguientes definiciones:

```
fromList = A.fromList

joinS = A.flatten

appendS xs ys = joinS (fromList [xs,ys])
```

Sean s_1 y s_2 las secuencias sobre la que se aplica *appendS* ,y n_1 y n_2 el largo de las mismas.

Definimos $s = fromList(s_1, s_2)$ siendo esta una secuencia que tiene como únicos elementos, s_1 y s_2 con $n = 2$ el largo de s. Y $s' = [s_1, s_2]$.

2.2.1. Trabajo

De la definición

$W_{appendS}(n_1, n_2) = W_{fromList}(s') + W_{joinS}(s) = W_{A.fromList}(s') + W_{A.flatten}(s)$

A partir de la tabla de costos se tiene

$W_{A.fromList}(s') \in O(|s'|)$

$W_{A.flatten}(s) \in O(|s|) + \sum_{i=0}^{|s|-1} O(|s!i|)$

Se busca demostrar que

$$W_{appendS} \in O(n_1 + n_2)$$

Demostración.

Se comienza con el Trabajo de $A.fromList$

$$W_{A.fromList}(s') \in O(|s'|) = O(2)$$

$$\therefore W_{A.fromList}(s') \in O(1)$$

Ahora se verá el de $A.flatten$

$$W_{A.flatten}(s) \in O(|s|) + \sum_{i=0}^{|s|-1} O(|s!i|)$$

Como $|s| = n = 2, O(|s|) = O(1)$

$$W_{A.flatten}(s) \in \sum_{i=0}^{|s|-1} O(|s!i|) = O(|s!0|) + O(|s!1|) = O(n_1) + O(n_2)$$

Por propiedades de O,

$$W_{A.flatten}(s) \in O(n_1 + n_2)$$

Ya habiendo hecho esto, se puede probar el Trabajo de *appendS*. Se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n_1 \geq n_0, n_2 \geq n_0$ se cumpla lo siguiente

$$0 \leq W_{appendS}(n_1, n_2) \leq c(n_1 + n_2)$$

Teniendo

$$W_{appendS}(n_1, n_2) = W_{A.fromList}(s') + W_{A.flatten}(s) = c_0 + W_{A.flatten}(s) \leq c_0 + c'(n_1 + n_2)$$

Tomando $c \geq c_0 + c'$ se llega a

$$0 \leq W_{appendS}(n_1, n_2) \leq c(n_1 + n_2)$$

Por definición de O

$$W_{appendS}(n_1, n_2) \in O(n_1 + n_2)$$

■

2.2.2. Profundidad

A partir de la definición se tiene

$S_{appendS}(n_1, n_2) = S_{fromList}(s') + S_{joinS}(s) = S_{A.fromList}(s') + S_{A.flatten}(s)$

A partir de la tabla de costos se tiene $S_{A.fromList}(s') \in O(|s'|)$

$S_{A.flatten}(s) \in O(lg|s|)$

Se busca demostrar que

$$S_{appendS} \in O(1)$$

Demostración.

Se comienza con la Profundidad de $A.fromList$

$$S_{A.fromList}(s') \in O(|s'|) = O(2)$$

$$\therefore S_{A.fromList}(s') \in O(1)$$

Ahora se verá la de $A.flatten$

$$S_{A.flatten}(s) \in O(lg\ |s|)$$

Como $|s| = n = 2, O(lg\ |s|) = O(lg\ 2) = O(1)$

$$S_{A.flatten}(s) \in O(1)$$

Ya habiendo hecho esto, se puede probar la Profundidad de *appendS*. Se busca probar que para algún $c \in \mathbb{R}^+$ y $n_0 \in \mathbb{N} / \forall n_1 \geq n_0, n_2 \geq n_0$ se cumpla lo siguiente

$$0 \leq S_{appendS}(n_1, n_2) \leq c$$

Teniendo

$$S_{appendS}(n_1, n_2) = S_{A.fromList}(s') + S_{A.flatten}(s) = c_0 + c_1$$

Tomando $c \geq c_0 + c_1$ se llega a

$$0 \leq S_{appendS}(n_1, n_2) \leq c$$

Por definición de O

$$S_{appendS}(n_1, n_2) \in O(1)$$

■

2.3. reduceS

Se tienen las siguientes definiciones:

```
contraerSA :: (a -> a -> a) -> A.Arr a -> A.Arr a
contraerSA f xs | even l = A.tabulate contraerSP half
                | otherwise = A.tabulate contraerSI (half+1)
                where
                     l = A.length xs
                     half = div l 2
                     contraerSP i = f (xs!(2*i)) (xs!((2*i) + 1))
                     contraerSI i | i == half = xs!(2*half)
                                  | otherwise = contraerSP i
reduceS f a xs | l == 0 = a
               | l == 1 = f a (nthS xs 0)
               | otherwise = reduceS f a (contraerSA f xs)
               where
                    l = lengthS xs
```

Sea s la secuencia sobre la que se aplica *reduceS*, f la función con la cual se aplica *reduceS* y n el largo de la secuencia s.

Se supone $W_f \in O(1)$ y $S_f \in O(1)$

2.3.1. Trabajo

De la definición

$W_{reduceS}(0) = c_0$

$W_{reduceS}(1) = W_f(a, s_0) + W_{nthS}(0) = c_f + c_{nthS}$

$W_{reduceS}(n) = c_1 + W_{contraerSA}(n) + W_{reduceS}(\left\lceil \frac{n}{2} \right\rceil)$

Primero se analizará el Trabajo de *contraerSA*.

Se demostrará que

$$W_{contraerSA} \in \Theta(n)$$

A partir de la definición se tiene:

$W_{contraerSA}(n) = c_2 + W_{A.tabulate}(f', \frac{n}{2})$.

Donde f' sera definido como *contraerSP* o *contraerSI* dependiendo de la paridad de n.

Demostración.

Si n par

$W_{f'}(i) = W_{contraerSP}(i) = W_f(s_{2i}, s_{2i+1}) + 2c_{nthS} = c_f + 2c_{nthS} \in \Theta(1)$

Si n impar

$W_{f'}(i) = W_{contraerSI}(i) = W_{nthS}(n-1) = c_{nthS} \in \Theta(1)$ si $i = \frac{n-1}{2}$

$W_{f'}(i) = W_{contraerSI}(i) = W_{contraerSP}(i) \in \Theta(1)$ en otro caso

Como se puede ver, en cualquier caso, $W_{f'} \in \Theta(1)$

Ahora viendo $W_{contraerSA}$

$$W_{contraerSA}(n) = c_2 + W_{A.tabulate}(f', \frac{n}{2}) \in \Theta(\sum_{i=0}^{\frac{n}{2}} W_{f'}(i)) = \Theta(\sum_{i=0}^{\frac{n}{2}} \Theta(1)) = \Theta(\frac{n}{2})$$

$$\therefore W_{contraerSA} \in \Theta(n)$$

■

Ahora podemos continuar con el Trabajo de *reduceS*

Se demostrará que

$$W_{reduceS} \in \Theta(n)$$

Demostración.

Se supone $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$W_{reduceS}(n) = c_1 + W_{contraerSA}(n) + W_{reduceS}(\left\lceil \frac{n}{2} \right\rceil) = c_1 + W_{contraerSA}(n) + W_{reduceS}(\frac{n}{2})$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el tercer caso del mismo

Tomando $a = 1, b = 2$ y $f'' = W_{contraerSA}(n) + c_1$

Como $W_{contraerSA} \in \Theta(n)$, trivialmente $f'' \in \Theta(n)$

$\exists e > 0, e = 1 : f'' \in \Omega(n^{log_b a+e}) = \Omega(n^{log_2 1+1}) = \Omega(n^{0+1}) = \Omega(n)$

y

$\exists c < 1, N \in \mathbb{N}, N > 1, \forall n > N, af''(\frac{n}{b}) \leq cf''(n)$

Si y solo si $W_{contraerSA}(\frac{n}{2}) \leq cW_{contraerSA}(n)$

$$W_{contraerSA}(\frac{n}{2}) = \frac{n}{2} = \frac{1}{2}n \leq cW_{contraerSA}(n)$$

$$\therefore W_{reduceS} \in \Theta(f'') = \Theta(2^k)$$

Como $f'' \in \Theta(n)$ es suave y $W_{reduceS}$ es eventualmente no decreciente, por la regla de suavidad

$$W_{reduceS} \in \Theta(n), \forall n > 0$$

■

2.3.2. Profundidad

A partir de la definición se tiene

$S_{reduceS}(0) = c_0$

$S_{reduceS}(1) = S_f(a, s_o) + S_{nthS}(0) = c_f + c_{nthS}$

$S_{reduceS}(n) = c_1 + S_{contraerSA}(n) + S_{reduceS}(\left\lceil \frac{n}{2} \right\rceil)$

Primero se analizará la Profundidad de $contraerSA$ $S_{contraerSA}(n) = c_2 + S_{A.tabulate}(f', \frac{n}{2})$. Donde f' sera definido como $contraerSP$ o $contraerSI$ dependiendo de la paridad de n.

Se demostrará que

$$S_{contraerSA}(n) \in \Theta(1)$$

Demostración.

Estando f' formado a partir de f y $nthS$, ya que el Trabajo de estas 2 es igual a su Profundidad, el análisis que se realiza sobre f' es el mismo que se realizó previamente en el Trabajo y como resultado tenemos que $S_{f'} \in \Theta(1)$.

Ahora viendo $S_{contraerSA}$

$$S_{contraerSA}(n) = c_2 + S_{A.tabulate}(f', \frac{n}{2}) \in \Theta(\max_{i=0}^{\frac{n}{2}} S_{f'}(i)) = \Theta(\max_{i=0}^{\frac{n}{2}} \Theta(1)) = \Theta(1)$$

$$\therefore S_{contraerSA} \in \Theta(1)$$

■

Con esto se puede ver la Profundidad de $reduceS$

Se demostrará que

$$S_{reduceS} \in \Theta(log\, n)$$

Demostración.

Se supone $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$S_{reduceS}(n) = c_1 + S_{contraerSA}(n) + S_{reduceS}(\left\lceil \frac{n}{2} \right\rceil) = c_1 + S_{contraerSA}(n) + S_{reduceS}(\frac{n}{2})$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el segundo caso del mismo

Tomando $a = 1, b = 2$ y $f'' = S_{contraerSA}(n) + c_1$

Como $S_{contraerSA} \in \Theta(1)$, trivialmente $f'' \in \Theta(1)$

$f'' \in \Theta(n^{log_b a}) = \Theta(n^{log_2 1}) = \Theta(n^0) = \Theta(1)$

y aplicando el teorema queda

$$S_{reduceS} \in \Theta(n^{log_b a}\, log\, n) = \Theta(n^{log_2 1}\, log\, n) = \Theta(log\, n)$$

Como $f'' \in \Theta(1)$ es suave y $S_{reduceS}$ es eventualmente no decreciente, por la regla de suavidad

$$S_{reduceS} \in \Theta(log\, n), \forall n > 0$$

■

2.4. scanS

Se tienen las siguientes definiciones:

```
contraerSA :: (a -> a -> a) -> A.Arr a -> A.Arr a
contraerSA f xs | even l = A.tabulate contraerSP half
                | otherwise = A.tabulate contraerSI (half+1)
                where
                      l = A.length xs
                      half = div l 2
                      contraerSP i = f (xs!(2*i)) (xs!((2*i) + 1))
                      contraerSI i | i == half = xs!(2*half)
                                   | otherwise = contraerSP i

expandirSA :: (a -> a -> a) -> A.Arr a -> A.Arr a -> A.Arr a
expandirSA f xs is = A.tabulate
                        (\i -> if even i then (is!(div i 2))
                                         else f (is!(div i 2)) (xs!(i-1)))
                        (A.length xs)

singletonS a = fromList [a]

scanS f a xs | l == 0 = (emptyS, a)
             | l == 1 = (singletonS a, f a (nthS xs 0))
             | otherwise = let (is, r) = scanS f a (contraerSA f xs)
                           in (expandirSA f xs is, r)
             where
                  l = lengthS xs
```

Sea s la secuencia sobre la que se aplica $scanS$, f la función con la cual se aplica $scanS$ y n el largo de la secuencia s. Para la función $expandirSA$ también definiremos una secuencia q que sera la devuelta por $scanS$ como primer elemento de la tupla.

Se supone $W_f \in O(1)$ y $S_f \in O(1)$

2.4.1. Trabajo

De la definición se tiene

$W_{scanS}(0) = c_0$

$W_{scanS}(1) = W_{singletonS}(1) + W_f(a, s_0) + W_{nthS}(0) = c_{singletonS} + c_f + c_{nthS}$

$W_{scanS}(n) = c_1 + W_{expandirSA}(n) + W_{contraerSA}(n) + W_{scanS}(\lceil \frac{n}{2} \rceil)$

La función $contraerSA$ ya fue analizada por lo que resta ver el Trabajo de $expandirSA$

A partir de la definición se tiene:

$W_{expandirSA}(n) = W_{A.tabulate}(f', n) + W_{A.length}(n)$

Se demostrara que

$$W_{expandirSA} \in \Theta(n)$$

Donde f' será definido dependiendo de la paridad del índice pasado.

Demostración.

Si el índice pasado a la función f' es par, entonces

$W_{f'}(i) = c_2 + W_{(!)}(\frac{i}{2}) = c_2 + c_{(!)} \in \Theta(1)$

En cambio, si es impar

$W_{f'}(i) = c_2 + W_f(q_{\frac{i}{2}}, s_{i-1}) + W_{(!)}(\frac{i}{2}) + W_{(!)}(i-1) = c_2 + c_f + 2c_{(!)} \in \Theta(1)$

Se puede ver que en cualquier caso $W_{f'} \in \Theta(1)$

Volviendo a $expandirSA$

$$W_{expandirSA}(n) = W_{A.tabulate}(f', n) + W_{A.length}(n) \in \Theta(\sum_{i=0}^{n-1} W_{f'}(i)) + \Theta(1) = \Theta(\sum_{i=0}^{n-1} \Theta(1)) = \Theta(n)$$

$$\therefore W_{expandirSA} \in \Theta(n)$$

■

Ahora se puede demostrar el trabajo de $scanS$

Se demostrará que

$$W_{scanS} \in \Theta(n)$$

Demostración.

Se supone $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$W_{scanS}(n) = c_1 + W_{expandirSA}(n) + W_{contraerSA}(n) + W_{reduceS}(\left\lceil \frac{n}{2} \right\rceil)$$

$$= c_1 + W_{expandirSA}(n) + W_{contraerSA}(n) + W_{reduceS}(\frac{n}{2})$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el tercer caso del mismo

Tomando $a = 1, b = 2$ y $f'' = W_{expandirSA}(n) + W_{contraerSA}(n) + c_1$

Como $W_{contraerSA} \in \Theta(n)$ y $W_{expandirSA} \in \Theta(n)$, trivialmente $f'' \in \Theta(n)$

$\exists e > 0, e = 1 : f'' \in \Omega(n^{log_b a + e}) = \Omega(n^{log_2 1 + 1}) = \Omega(n^{0+1}) = \Omega(n)$

y

$\exists c < 1, N \in \mathbb{N}, N > 1, \forall n > N, af''(\frac{n}{b}) \leq cf'(n)$

Si y solo si $W_{contraerSA}(\frac{n}{2}) + W_{expandirSA}(\frac{n}{2}) \leq c(W_{contraerSA}(n) + W_{expandirSA}(n))$

$$W_{contraerSA}(\frac{n}{2}) + W_{expandirSA}(\frac{n}{2}) + c_1$$

$$\leq \frac{W_{contraerSA}(n) + W_{expandirSA}(n) + c_1}{2} = c(W_{contraerSA}(n) + W_{expandirSA}(n) + c_1)$$

Con $c = \frac{1}{2}$

$$\therefore W_{scanS} \in \Theta(f'') = \Theta(2^k)$$

Como $f'' \in \Theta(n)$ es suave y W_{scanS} es eventualmente no decreciente, por la regla de suavidad

$$W_{scanS} \in \Theta(n), \forall n > 0$$

■

2.4.2. Profundidad

A partir de la definición se tiene

$S_{scanS}(0) = c_0$

$S_{scanS}(1) = S_{singletonS}(1) + S_f(a, s_0) + S_{nthS}(0) = c_{singletonS} + c_f + c_{nthS}$

$S_{scanS}(n) = c_1 + S_{expandirSA}(n) + S_{contraerSA}(n) + S_{scanS}(\left\lceil \frac{n}{2} \right\rceil)$

La Profundidad función $contraerSA$ ya fue analizada por lo que resta ver la Profundidad de $expandirSA$

A partir de lo que se tiene:

$S_{expandirSA}(n) = S_{A.tabulate}(f', n) + S_{A.length}(n)$

Se demostrara que

$$S_{expandirSA} \in \Theta(1)$$

Demostración.

Estando f' formado a partir de f y $nthS$, ya que el Trabajo de estas 2 es igual a su Profundidad, el análisis que se realiza sobre f' es el mismo que se realizo previamente en el Trabajo y como resultado tenemos que $S_{f'} \in \Theta(1)$.

Ahora viendo $S_{expandirSA}$

$$S_{expandirSA}(n) = S_{A.tabulate}(f', n) + S_{A.length}(n) \in \Theta(\max_{i=0}^{n-1} S_{f'}(i)) + \Theta(1)$$

$$= \Theta(\max_{i=0}^{n-1} S_{f'}(i)) = \Theta(\max_{i=0}^{n-1} \Theta(1)) = \Theta(1)$$

$$\therefore S_{expandirSA}(n) \in \Theta(1)$$

■

Con esto se puede ver la Profundidad de $scanS$

Se demostrará que

$$S_{scanS} \in \Theta(log\, n)$$

Demostración.

Se supone $n = 2^k$ con $k \in \mathbb{N}$

Con esto se tiene

$$S_{scanS}(n) = c_1 + S_{contraerSA}(n) + S_{expandirSA}(n) + S_{scanS}(\left\lceil \frac{n}{2} \right\rceil)$$
$$= c_1 + S_{contraerSA}(n) + S_{expandirSA}(n) + S_{scanS}(\frac{n}{2})$$

Se puede tratar de aplicar el Teorema Maestro, mas específicamente el segundo caso del mismo

Tomando $a = 1, b = 2$ y $f'' = c_1 + S_{contraerSA}(n) + S_{expandirSA}(n)$

Como $S_{contraerSA} \in \Theta(1)$ y $S_{expandirSA} \in \Theta(1)$, trivialmente $f'' \in \Theta(1)$

$f'' \in \Theta(n^{log_b a}) = \Theta(n^{log_2 1}) = \Theta(n^0) = \Theta(1)$

y aplicando el teorema queda

$$S_{scanS} \in \Theta(n^{log_b a}\, log\, n) = \Theta(n^{log_2 1}\, log\, n) = \Theta(log\, n)$$

Como $f'' \in \Theta(1)$ es suave y S_{scanS} es eventualmente no decreciente, por la regla de suavidad

$$S_{scanS} \in \Theta(log\, n), \forall n > 0$$

$\blacksquare$