



Estructuras de Datos y Algoritmos II

Trabajo práctico 2

Bautista Marelli

Aldana Zarate

Junio 2020

1. Cálculo de costos

1.1. Implementación de Seq para listas

mapS

Trabajo

Tenemos mapS definida como:

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

Por lo tanto, podemos desplegar los siguientes casos en el análisis de costo:

$$\begin{aligned} W(\text{mapS } f \ 0) &= O(1) \\ W(\text{mapS } f \ n) &= O(1) && \text{(factores constantes de let y cons)} \\ &+ W(f \ n_0) && \text{(llamada izquierda a f)} \\ &+ W(\text{mapS } f \ n - 1) && \text{(llamada derecha)} \end{aligned}$$

Demostremos por inducción que

$$W(\text{mapS } f \ n) \in \sum_{i=0}^{n-1} W(f \ i)$$

Usando notación O vemos que

$$\begin{aligned} &W(\text{mapS } f \ n) \\ = &\langle \text{def. } W \rangle \\ &O(1) + W(f \ n_0) + W(\text{mapS } f \ n - 1) \\ = &\langle \text{HI} \rangle \\ &O(1) + W(f \ n_0) + O\left(\sum_{i=0}^{n-2} W(f \ i)\right) \\ = &\langle \text{como } O(1) \text{ está acotado por } W(f \ n_0) \rangle \\ &O(W(f \ n_0)) + \sum_{i=1}^{n-2} W(f \ i) \\ = &\langle \text{agrupando la sumatoria} \rangle \\ &O\left(\sum_{i=1}^{n-1} W(f \ i)\right) \end{aligned}$$

■

Profundidad

En este caso, de nuestro modelo de costo y de la implementación de mapS podemos obtener las siguientes ecuaciones:

$$\begin{aligned} S(\text{mapS } f \ 0) &= O(1) \\ S(\text{mapS } f \ n) &= O(1) \quad (\text{factores constantes de let y cons}) \\ &\quad + \text{máx}\{S(f \ n_0), S(\text{mapS } f \ n - 1)\} \quad (\text{llamada a izquierda y derecha}) \end{aligned}$$

Demostremos por inducción que

$$S(\text{mapS } f \ n) \in O(\text{máx}_{i=0}^{n-1} S(f \ i) + n)$$

Usando notación O vemos que

$$\begin{aligned} &S(\text{mapS } f \ n) \\ = &\langle \text{def. } S \rangle \\ &O(1) + \text{máx}\{S(f \ n_0), S(\text{mapS } f \ n - 1)\} \\ \leq &\langle \text{HI} \rangle \\ &O(1) + \text{máx}\{S(f \ n_0), O(\text{máx}_{i=0}^{n-2} S(f \ i) + (n - 1))\} \\ \leq &\langle (1) \rangle \\ &O(1) + O(\text{máx}_{i=0}^{n-1} S(f \ i) + (n - 1)) \\ = &\langle \text{Como } O(n - 1) \text{ y } O(1) \text{ están acotados por } O(n) \rangle \\ &O(\text{máx}_{i=0}^{n-1} S(f \ i) + n) \end{aligned}$$

(1) máx en este paso puede tener dos resultados:

- $\text{máx}\{S(f \ n_0), O(\text{máx}_{i=0}^{n-2} S(f \ i) + (n - 1))\} = S(f \ n_0)$. En este caso, queda acotado por lo expuesto dado que además de estar incluido en $\text{máx}_{i=0}^{n-1} S(f \ i)$, se le está sumando un factor lineal.
- $\text{máx}\{S(f \ n_0), O(\text{máx}_{i=0}^{n-2} S(f \ i) + (n - 1))\} = O(\text{máx}_{i=0}^{n-2} S(f \ i) + (n - 1))$. Este caso está acotado trivialmente por lo expuesto al estar incluido en el conjunto.

appendS

Trabajo

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

Para calcular el trabajo del appendS definimos la recurrencia en relación de la longitud de xs:

$$\begin{aligned} T(0) &= c1 \\ T(n) &= T(n-1) + c2 \end{aligned}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq T(n) \leq c \cdot n$

$$\begin{aligned} &T(n+1) = T(n) + c2 \\ = \langle \text{HI} \rangle & \\ &c \cdot n + c2 \\ \leq \langle \text{Tomando } c \geq c2, m_0 \geq 0 \rangle & \\ &c \cdot (n+1) \end{aligned}$$

Caso Base: Se verifica trivialmente
 $\therefore T \in O(n) \implies W(\text{appendS}) \in O(n)$

Profundidad

En este caso, de nuestro modelo de costo y de la implementación de appendS podemos obtener las siguientes ecuaciones:

$$\begin{aligned} S\text{appendS}(0) &= c1 \\ S\text{appendS}(n) &= 1 + S\text{appendS}(n-1) \end{aligned}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq S\text{appendS}(n) \leq c \cdot n$

$$\begin{aligned} &S\text{appendS}(n+1) = 1 + S\text{appendS}(n) \\ = \langle \text{HI} \rangle & \\ &1 + c \cdot n \\ \leq \langle \text{Tomando } c \geq 1, m_0 \geq 0 \rangle & \\ &c \cdot (n+1) \end{aligned}$$

Caso Base: Se verifica trivialmente
 $\therefore S\text{appendS} \in O(n)$

reduceS

Trabajo

Tenemos a reduceS definida como:

```
contract :: (a -> a -> a) -> [a] -> [a]
contract _ [] = []
contract _ [x] = [x]
contract f (x:y:xs) = let (z, zs) = f x y ||| contract f xs
                      in z:zs

reduceS_ _ e [] = e
reduceS_ f e [x] = f e x
reduceS_ f e xs = let ctr = contract f xs
                  ys = reduceS_ f e ctr
                  in id ys
```

Suponemos que $W(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista (1):

$$\begin{aligned} W(\text{reduceS } 0) &= O(1) \\ W(\text{reduceS } 1) &= W(f \ x_1 \ x_2) = O(1) \\ W(\text{reduceS } n) &= W(\text{contract } n) + W(\text{reduceS } \lceil \frac{n}{2} \rceil) + W(\text{id } n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $W\text{contract}(n)$

$$\begin{aligned} W\text{contract}(0) &= a1 \\ W\text{contract}(1) &= a2 \\ W\text{contract}(n) &= a3 + W\text{contract}(n-2) \end{aligned}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq W\text{contract}(n) \leq c \cdot n$

$$W\text{contract}(n+1) = W\text{contract}(n-1) + a3$$

$\leq \langle \text{HI} \rangle$

$$c \cdot (n-1) + a3$$

$\leq \langle \text{Tomando un } c \geq a3, m_0 \geq 0 \rangle$

$$c \cdot (n+1)$$

$$\therefore W\text{contract} \in O(n) \quad (2)$$

2. $W(\text{id } n)$

Si buscamos la definición en el Preludio tenemos definida a id como:

```
id           :: a -> a
id x        =  x
```

$$\therefore W(\text{id } n) = O(1) \quad (3)$$

Por lo tanto, si volvemos a (1) con lo obtenido en (2) y (3) tenemos que:

$$\begin{aligned} W(\text{reduceS } 0) &= O(1) \\ W(\text{reduceS } 1) &= O(1) \\ W(\text{reduceS } n) &= O(n) + W(\text{reduceS } \lceil \frac{n}{2} \rceil) + O(1) \end{aligned}$$

Por suavidad y dado que $O(1)$ está acotado por $O(n)$ queda

$$W(\text{reduceS } n) = W(\text{reduceS } \frac{n}{2}) + O(n)$$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con dichos valores y $\epsilon = 1$ estamos en condiciones de utilizar el 3^{er} caso de este teorema.

$$\therefore W(\text{reduceS } n) \in O(n)$$

Profundidad

Suponemos que $S(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista (1):

$$\begin{aligned} S(\text{reduceS } 0) &= O(1) \\ S(\text{reduceS } 1) &= W(f \ x_1 \ x_2) = O(1) \\ S(\text{reduceS } n) &= S(\text{contract } n) + S(\text{reduceS } \lceil \frac{n}{2} \rceil) + S(\text{id } n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $S(\text{contract } n)$

$$\begin{aligned} S\text{contract}(0) &= O(1) \\ S\text{contract}(1) &= O(1) \\ S\text{contract}(n) &= O(1) \quad (\text{cons}) \\ &\quad + \text{máx}\{S(f \ x \ y), S(\text{contract } n - 2)\} \end{aligned}$$

Demostremos por inducción que

$$S(\text{contract } n) \in O(n)$$

$$S(\text{contract } n)$$

$$= \langle \text{def. S} \rangle$$

$$O(1) + \text{máx}\{S(f \ x \ y), S(\text{contract } n - 2)\}$$

$$= \langle \text{HI} \rangle$$

$$O(1) + \text{máx}\{S(f \ x \ y), O(n - 2)\}$$

$$= \langle f \in O(1) \rangle$$

$$O(1) + \text{máx}\{O(1), O(n - 2)\}$$

$$= \langle \text{def. máx} \rangle$$

$$O(1) + O(n - 2)$$

$$= \langle O(n - 2) \text{ y } O(1) \text{ están acotados por } O(n) \rangle$$

$$O(n) \quad (2)$$

■

2. $S(\text{id } n)$

$$\text{En este caso también obtenemos trivialmente que } S(\text{id } n) \in O(1) \quad (3)$$

Por lo tanto, si volvemos a (1) con lo obtenido en (2) y (3) tenemos que:

$$\begin{aligned} S(\text{reduceS } 0) &= O(1) \\ S(\text{reduceS } 1) &= O(1) \\ S(\text{reduceS } n) &= O(n) + S(\text{reduceS } \lceil \frac{n}{2} \rceil) + O(1) \end{aligned}$$

Por suavidad y dado que $O(1)$ está acotado por $O(n)$ queda

$$S(\text{reduceS } n) = S(\text{reduceS } \frac{n}{2}) + O(n)$$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con dichos valores y $\epsilon = 1$ estamos en condiciones de utilizar el 3^{er} caso de este teorema.

$$\therefore S(\text{reduceS } n) \in O(n)$$

scanS

Trabajo

Tenemos a scanS definida como:

```
contract :: (a -> a -> a) -> [a] -> [a]
contract _ [] = []
contract _ [x] = [x]
contract f (x:y:xs) = let (z, zs) = f x y ||| contract f xs
                      in z:zs

scanS_ _ e [] = ([], e)
scanS_ f e [x] = ([e], f e x)
scanS_ f e xs = let ctr = contract f xs
                (ys, y) = scanS_ f e ctr
                in (buildList f xs ys False, y)

where
  buildList f [] _ _ = []
  buildList f _ [] _ = []
  buildList f [x] [y] _ = [y]
  buildList f l1@(x:z:xs) l2@(y:ys) flag | flag = (f y x) : buildList f xs ys False
                                          | otherwise = y : buildList f l1 l2 True
```

Suponemos que $W(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista:

$$\begin{aligned} W_{\text{scanS}}(0) &= O(1) \\ W_{\text{scanS}}(1) &= O(1) \\ W_{\text{scanS}}(n) &= W_{\text{contract}}(n) + W_{\text{scanS}}(\lceil \frac{n}{2} \rceil) + W_{\text{buildList}}(n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $W_{\text{contract}}(n)$

$$\begin{aligned} W_{\text{contract}}(0) &= a1 \\ W_{\text{contract}}(1) &= a2 \\ W_{\text{contract}}(n) &= a3 + W_{\text{contract}}(n - 2) \end{aligned}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq Wconcata(n) \leq c \cdot n$

$$Wcontract(n+1) = Wcontract(n-1) + a3$$

$$\leq \langle HI \rangle$$

$$c \cdot (n-1) + a3$$

$$\langle \text{Tomando un } c \geq a3, m_0 \geq 0 \rangle$$

$$c \cdot (n+1)$$

$$\therefore Wcontract \in O(n)$$

2. $WbuildList(n)$

$$WbuildList(0) = b1$$

$$WbuildList(1) = b2$$

$$WbuildList(n) = \begin{cases} b3 + WbuildList(n-2) & flag == True \\ b3 + WbuildList(n) & flag == False \end{cases}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq WbuildList(n) \leq c \cdot n$

Para calcular el costo, tenemos que separar en 2 casos:

■ $flag == True$

$$WbuildList(n+1) = b3 + WbuildList(n-1)$$

$$= \langle HI \rangle$$

$$b3 + c \cdot (n-1)$$

$$\leq \langle \text{Tomando } c \geq b3, m_0 \geq 0 \rangle$$

$$c \cdot (n+1)$$

■ $flag == False$

$$WbuildList(n+1) = b3 + WbuildList(n+1)$$

$$= \langle flag = True \rangle$$

$$2 \cdot b3 + WbuildList(n-1)$$

$$= \langle HI \rangle$$

$$2 \cdot b3 + c \cdot (n-1)$$

$$\leq \langle \text{Tomando } c \geq 2 \cdot b3, m_0 \geq 0 \rangle$$

$$c \cdot (n+1)$$

$$\therefore WbuildList \in O(n)$$

Ahora que ya tenemos los costos, pasamos a calcular el costo de $WscanS(n)$:

$$WscanS(0) = O(1)$$

$$WscanS(1) = O(1)$$

$$WscanS(n) = O(n) + WscanS(\lceil \frac{n}{2} \rceil) + O(n)$$

Por suavidad queda $WscanS(n) = WscanS(\frac{n}{2}) + 2 \cdot O(n)$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con estos valores y $\epsilon = 1$ estamos en condiciones de utilizar el 3^{er} caso del teorema.

$\therefore WscanS \in O(n)$

Profundidad

Suponemos que $S(f) \in O(1)$. Según la definición y nuestro modelo de costos podemos ver que:

$$\begin{aligned} SscanS(0) &= O(1) \\ SscanS(1) &= O(1) \\ SscanS(n) &= Scontract(n) + SscanS(\lceil \frac{n}{2} \rceil) + SbuildList(n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $Scontract(n)$

$$\begin{aligned} Scontract(0) &= O(1) \\ Scontract(1) &= O(1) \\ Scontract(n) &= O(1) \quad (cons) \\ &\quad + \text{máx}\{S(f \ x \ y), S(contract \ n - 2)\} \end{aligned}$$

Demostremos por inducción que

$$Scontract(n) \in O(n)$$

$$Scontract(n)$$

$$= \langle \text{def. S} \rangle$$

$$O(1) + \text{máx}\{S(f \ x \ y), Scontract(n - 2)\}$$

$$= \langle \text{HI} \rangle$$

$$O(1) + \text{máx}\{S(f \ x \ y), O(n - 2)\}$$

$$= \langle f \in O(1) \rangle$$

$$O(1) + \text{máx}\{O(1), O(n - 2)\}$$

$$= \langle \text{def. máx} \rangle$$

$$O(1) + O(n - 2)$$

$$= \langle O(n - 2) \text{ y } O(1) \text{ están acotados por } O(n) \rangle$$

$$O(n)$$

2. *SbuildList*(*n*)

$$\begin{aligned}
SbuildList(0) &= b1 \\
SbuildList(1) &= b2 \\
SbuildList(n) &= \begin{cases} b3 + SbuildList(n-2) & flag == True \\ b3 + SbuildList(n) & flag == False \end{cases}
\end{aligned}$$

Supongamos que $\exists c \in \mathbb{R}^+, m_0 \in \mathbb{N}_0 / \forall n \geq m_0, 0 \leq SbuildList(n) \leq c \cdot n$

Para calcular el costo, tenemos que separar en 2 casos:

- `flag == True`

$$\begin{aligned}
&SbuildList(n+1) = b3 + SbuildList(n-1) \\
&= \langle HI \rangle \\
&\qquad b3 + c \cdot (n-1) \\
&\leq \langle Tomando \ c \geq b3, m_0 \geq 0 \rangle \\
&\qquad c \cdot (n+1)
\end{aligned}$$

- `flag == False`

$$\begin{aligned}
&SbuildList(n+1) = b3 + SbuildList(n+1) \\
&= \langle flag = True \rangle \\
&\qquad 2 \cdot b3 + SbuildList(n-1) \\
&= \langle HI \rangle \\
&\qquad 2 \cdot b3 + c \cdot (n-1) \\
&\leq \langle Tomando \ c \geq 2 \cdot b3, m_0 \geq 0 \rangle \\
&\qquad c \cdot (n+1)
\end{aligned}$$

$$\therefore SbuildList \in O(n)$$

Por suavidad queda $SscanS(n) = SscanS(\frac{n}{2}) + 2 \cdot O(n)$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con estos valores y $\epsilon = 1$ estamos en condiciones de utilizar el 3^{er} caso del teorema.

$$\therefore SscanS \in O(n)$$

1.2. Implementación de Seq para arreglos

mapS

Trabajo

Tenemos mapS definida como:

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

De esta definición podemos ver que:

$$\begin{aligned}
W(mapS \ f \ n) &= O(1) && \text{(factor constante del costo de length)} \\
&+ W(tabulate \ g \ n) && \text{(llamado a tabulate)}
\end{aligned}$$

Donde

$g = \lambda i \rightarrow (f (nthS\ xs\ i))$

Por lo tanto,

$$W(g) = W(f\ p) + W(nthS\ n\ i)$$

Y al tener $W(nthS\ n\ i) \in O(1)$, el costo de g queda dominado por el costo de f

$$\therefore W(g) \in O(W(f\ p)) \quad (1)$$

Demostremos que

$$W(\text{mapS}\ f\ n) \in \sum_{i=0}^{n-1} W(f\ i)$$

Usando notación O vemos que

$$W(\text{mapS}\ f\ n)$$

$$= \langle \text{def. } W \rangle$$

$$O(1) + W(\text{tabulate}\ g\ n)$$

$$= \langle \text{Costo de tabulate} \rangle$$

$$O(1) + O\left(\sum_{i=0}^{n-1} W(g\ i)\right)$$

$$= \langle O(1) \text{ queda dominado por el costo de tabulate y (1)} \rangle$$

$$O\left(\sum_{i=0}^{n-1} W(f\ i)\right)$$

■

Profundidad

Según la definición y nuestro modelo de costos podemos ver que:

$$\begin{aligned} S(\text{mapS}\ f\ n) &= O(1) && \text{(factor constante del costo de length)} \\ &+ S(\text{tabulate}\ g\ n) && \text{(llamado a tabulate)} \end{aligned}$$

Demostremos que

$$S(\text{mapS}\ f\ n) \in \max_{i=0}^{n-1} S(f\ i)$$

Usando notación O vemos que

$$S(\text{mapS}\ f\ n)$$

$$= \langle \text{def. } S \rangle$$

$$O(1) + S(\text{tabulate}\ g\ n)$$

$$= \langle \text{Costo de tabulate} \rangle$$

$$O(1) + O(\max_{i=0}^{n-1} S(g\ i))$$

= $\langle O(1) \rangle$ queda dominado por el costo de tabulate y (1) ya que la profundidad de g es análoga a su trabajo \rangle

$$O(\max_{i=0}^{n-1} S(f\ i))$$

■

appendS

Trabajo

Tenemos a appendS definida como:

```
appendS_ xs ys = tabulateS f (n + m)
  where
    n = lengthS xs
    m = lengthS ys
    f i | i < n      = nthS xs i
        | otherwise  = nthS ys (i - n)
```

Por lo tanto, queremos demostrar que $WappendS \in O(n + m)$, donde n, m son las longitudes de los arreglos. Usando la notación O , vemos que:

$$WappendS(n + m) = WtabulateS(n + m) = O\left(\sum_{i=0}^{(n+m)-1} W(f(i))\right)$$

$$= \langle f \in O(1) \rangle \langle W(f) = W(nthS(n)) = O(1) \rangle$$

$$O\left(\sum_{i=0}^{(n+m)-1} O(1)\right)$$

$$= \langle O(1) \cdot (n + m) = O(n + m) \rangle$$

$$O(n + m)$$

$$\therefore WappendS \in O(n + m)$$

Profundidad

En este caso, de nuestro modelo de costo y de la implementación de appendS podemos obtener la siguiente ecuación:

$$SappendS(n) = StabulateS(n)$$

$$SappendS(n + m) = StabulateS(n + m)$$

$$= \langle \text{def. } StabulateS \rangle$$

$$O\left(\max_{i=0}^{n+m-1} S(f(i))\right)$$

$$= \langle f \in O(1) \rangle \langle S(f) = S(nthS(n)) = O(1) \rangle$$

$$O(1)$$

$$\therefore SappendS \in O(1)$$

reduceS

Trabajo

Tenemos a reduceS definida como sigue:

```
contract :: (a -> a -> a) -> A.Arr a -> A.Arr a
contract f xs = if even len then tabulateS f' m else tabulateS f' (m+1)
  where
    len = lengthS xs
    m = div len 2
    f' i = if (2*i) == (len - 1) then (nthS xs (len - 1))
      else f (nthS xs (2 * i)) (nthS xs (2 * i + 1))

reduceS_ f e xs | len == 0      = e
                | len == 1      = f e (nthS xs 0)
                | otherwise     = let ctr = contract f xs
                                ys = reduceS_ f e ctr
                                in id ys
  where len = lengthS x
```

Suponemos que $W(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista (1):

$$\begin{aligned} W(\text{reduceS } 0) &= O(1) \\ W(\text{reduceS } 1) &= W(f \ x_1 \ x_2) + O(1) = O(1) + O(1) = O(1) \\ W(\text{reduceS } n) &= W(\text{contract } n) + W(\text{reduceS } \lceil \frac{n}{2} \rceil) + W(\text{id } n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $W(\text{contract } n)$

$$\begin{aligned} W(\text{contract } n) &= O(1) && (\text{factor constante de even, lengthS, div, +1}) \\ &+ W(\text{tabulate } f' \ \lceil \frac{n}{2} \rceil) \end{aligned}$$

Por suavidad, tenemos que:

$$W(\text{contract } n) = O(1) + W(\text{tabulate } f' \ \frac{n}{2})$$

Para simplificar, veamos que a los fines de calcular el costo de tabulate, $\frac{n}{2} \approx n$.

$$\text{Sabemos que } W(\text{tabulate } f' \ n) \in O(\sum_{i=0}^{n-1} W(f' \ i)) \quad (2)$$

Si observamos la definición de f' , podemos ver que $f' \in O(1)$, ya que su costo total está compuesto por suma de costos constantes:

$$\begin{aligned} W(f' \ n) &= O(1) && (2*i) == (len - 1) \\ &+ O(1) + O(1) + O(1) && (f \ (\text{nthS } xs \ (2 * i)) \ (\text{nthS } xs \ (2 * i + 1))) \end{aligned}$$

$$\therefore W(f' \ i) \in O(1)$$

Si volvemos a (2) reemplazando por lo obtenido, tenemos que:

$$W(\text{tabulate } f' \ n) \in O(\sum_{i=0}^n O(1)) \implies W(\text{tabulate } f' \ n) \in O(n) \implies W(\text{contract } n) = O(1) + O(n)$$

Dado que $O(1)$ se encuentra acotado por $O(n)$,

$$\therefore Wcontract \in O(n) \quad (3)$$

2. $W(id\ n)$

Si buscamos la definición en el Preludio tenemos definida a id como:

```
id           :: a -> a
id x        =  x
```

$$\therefore W(id\ n) = O(1) \quad (4)$$

Por lo tanto, si volvemos a (1) con lo obtenido en (3) y (4) tenemos que:

$$\begin{aligned} W(reduceS\ 0) &= O(1) \\ W(reduceS\ 1) &= O(1) \\ W(reduceS\ n) &= O(n) + W(reduceS\ \lceil \frac{n}{2} \rceil) + O(1) \end{aligned}$$

Por suavidad y dado que $O(1)$ está acotado por $O(n)$ queda

$$W(reduceS\ n) = W(reduceS\ \frac{n}{2}) + O(n)$$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con dichos valores y $\epsilon = 1$ estamos en condiciones de utilizar el 3^{er} caso de este teorema.

$$\therefore W(reduceS\ n) \in O(n)$$

Profundidad

Suponemos que $S(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista (1):

$$\begin{aligned} S(reduceS\ 0) &= O(1) \\ S(reduceS\ 1) &= S(f\ x_1\ x_2) + O(1) = O(1) + O(1) = O(1) \\ S(reduceS\ n) &= S(contract\ n) + S(reduceS\ \lceil \frac{n}{2} \rceil) + S(id\ n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $Scontract(n)$

$$\begin{aligned} S(contract\ n) &= O(1) && (\text{factor constante de even, lengthS, div, +1}) \\ &+ S(tabulate\ f'\ \lceil \frac{n}{2} \rceil) \end{aligned}$$

Por suavidad, tenemos que:

$$S(contract\ n) = O(1) + S(tabulate\ f'\ \frac{n}{2})$$

Para simplificar, veamos que para el cálculo de costos $\frac{n}{2} \approx n$.

$$\text{Sabemos que } S(tabulate\ f'\ n) \in O(\max_{i=0}^{n-1} S(f'\ i)) \quad (2)$$

Si observamos la definición de f' , podemos ver que $S(f'\ i) \in O(1)$, ya que su costo total está compuesto por suma de costos constantes:

$$S(f' n) = O(1) \quad (2^*i) == (\text{len} - 1) \\ + O(1) + O(1) + O(1) \quad (f \text{ (nthS xs (2^* i)) (nthS xs (2^* i + 1)))$$

$$\therefore S(f'i) \in O(1)$$

Si volvemos a (2) reemplazando por lo obtenido, tenemos que:

$$S(\text{tabulate } f' \ n) \in O(\max_{i=0}^{n-1} O(1)) \implies S(\text{tabulate } f' \ n) \in O(1) \implies S(\text{contract } n) = O(1) + O(1) = O(1)$$

$$\therefore S_{contract} \in O(1) \quad (3)$$

2. $S(id\ n)$

Si buscamos la definición en el Preludio tenemos definida a id como:

```
id      :: a -> a
id x    = x
```

$$\therefore S(id\ n) = O(1) \quad (4)$$

Por lo tanto, si volvemos a (1) con lo obtenido en (3) y (4) tenemos que:

$$\begin{aligned} S(reduceS\ 0) &= O(1) \\ S(reduceS\ 1) &= O(1) \\ S(reduceS\ n) &= O(1) + S(reduceS\ \lceil \frac{n}{2} \rceil) + O(1) \end{aligned}$$

Por suavidad y dado que $O(1) + O(1) = O(1)$

$$W(\text{reduceS } n) = W(\text{reduceS } \frac{n}{2}) + O(1)$$

Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$. Con dichos valores estamos en condiciones de utilizar el 2^{do} caso de este teorema.

$$\therefore W(\text{reduceS } n) \in O(\lg n)$$

scanS

Trabajo Tenemos a scanS definida como sigue:

```

contract :: (a -> a -> a) -> A.Arr a -> A.Arr a
contract f xs = if even len then tabulateS f' m
                else tabulateS f' (m + 1)
  where
    len = lengthS xs
    m = div len 2
    f' i = if (2 * i) == (len - 1) then (nthS xs (len - 1))
        else f (nthS xs (2 * i)) (nthS xs (2 * i + 1))

scanS_ f e xs | len == 0 = (emptyS, e)

```

```

| len == 1          = (singletonS_ e, f e (nthS xs 0))
| otherwise         = let ctr = contract f xs
                      (ys, y) = scanS_ f e ctr
                      in (buildList f xs ys, y)
where
  len = lengthS xs
  buildList f xs ys = tabulateS (\i -> if even i then (nthS ys (div i 2))
                                     else f (nthS ys (div i 2)) (nthS xs (i - 1)))
                                len

```

Suponemos que $W(f) \in O(1)$ y definimos la recurrencia en relación a la longitud de la lista:

$$\begin{aligned}
WscanS(0) &= O(1) \\
WscanS(1) &= O(1) \\
WscanS(n) &= Wcontract(n) + WscanS(\lceil \frac{n}{2} \rceil) + WbuildList(n)
\end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $Wcontract(n)$

$$\begin{aligned}
Wcontract(n) &= O(1) && \text{(factor constante de even, lengthS, div, +1)} \\
&+ W(tabulate f' \lceil \frac{n}{2} \rceil)
\end{aligned}$$

Por suavidad, tenemos que:

$$Wcontract(n) = O(1) + W(tabulate f' \frac{n}{2})$$

Para simplificar, veamos que a los fines de calcular el costo de tabulate, $\frac{n}{2} \approx n$.

$$Sabemos \text{ que } W(tabulate f' n) \in O(\sum_{i=0}^{n-1} W(f' i)) \quad (2)$$

Si observamos la definición de f' , podemos ver que $f' \in O(1)$, ya que su costo total está compuesto por suma de costos constantes:

$$\begin{aligned}
W(f' n) &= O(1) && (2*i) == (len - 1) \\
&+ O(1) + O(1) + O(1) && (f (nthS xs (2 * i)) (nthS xs (2 * i + 1)))
\end{aligned}$$

$$\therefore W(f'i) \in O(1)$$

Si volvemos a (2) reemplazando por lo obtenido, tenemos que:

$$W(tabulate f' n) \in O(\sum_{i=0}^{n-1} O(1)) \iff W(tabulate f' n) \in O(n) \implies Wcontract(n) = O(1) + O(n)$$

Dado que $O(1)$ se encuentra acotado por $O(n)$,

$$\therefore Wcontract \in O(n)$$

2. $WbuildList(n)$

$$WbuildList(n) = W(tabulate f' n)$$

$$\langle W(tabulate f' n) \in O(\sum_{i=0}^{n-1} W(f' i)) \rangle$$

$$O(\sum_{i=0}^{n-1} W(f' i))$$

Ahora demostremos que $f' \in O(1)$:

```
f' = (\i -> if even i then (nthS ys (div i 2))
                                     else f (nthS ys (div i 2)) (nthS xs (i - 1)))
```

$$W(f' n) = \begin{cases} O(1) & i \text{ es par} \\ O(1) & i \text{ es impar} \end{cases}$$

Queda claro que $W(f' n) \in O(1)$ ya que todas las funciones son de orden $O(1)$

Seguimos con la demostración de $WbuildList$:

$$= \langle W(f'i) \in O(1) \rangle$$

$$\sum_{i=0}^{n-1} O(1)$$

$$= \langle O(1) \cdot n = O(n) \rangle$$

$$O(n)$$

$$\therefore WbuildList \in O(n)$$

Por suavidad y por lo que demostramos anteriormente, nos queda: $WscanS(n) = 2 \cdot O(n) + WscanS(\lceil \frac{n}{2} \rceil)$. Aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$ y definiendo $\epsilon = 1$, caemos en el 3^{er} caso del teorema.

$$\therefore WscanS \in O(n)$$

Profundidad

Suponemos que $S(f) \in O(1)$. Según la definición y nuestro modelo de costos podemos ver que:

$$\begin{aligned} SscanS(0) &= O(1) \\ SscanS(1) &= O(1) \\ SscanS(n) &= Scontract(n) + SscanS(\lceil \frac{n}{2} \rceil) + SbuildList(n) \end{aligned}$$

Ahora que definimos la recurrencia, tenemos que calcular:

1. $Scontract(n)$

$$S(contract n) = O(1) + S(tabulate f' \lceil \frac{n}{2} \rceil) \quad (\text{factor constante de even, lengthS, div, +1})$$

Por suavidad, tenemos que:

$$Scontract(n) = O(1) + S(tabulate f' \frac{n}{2})$$

Para simplificar, veamos que para el cálculo de costos $\frac{n}{2} \approx n$.

$$\text{Sabemos que } S(tabulate f' n) \in O(\max_{i=0}^{n-1} S(f' i)) \quad (2)$$

Si observamos la definición de f' , podemos ver que $S(f'i) \in O(1)$, ya que su costo total está compuesto por suma de costos constantes:

$$\begin{aligned} S(f' n) &= O(1) + O(1) + O(1) + O(1) \quad (2*i) == (len - 1) \\ &\quad (f (nthS xs (2 * i)) (nthS xs (2 * i + 1))) \end{aligned}$$

$$\therefore S(f'i) \in O(1)$$

Si volvemos a (2) reemplazando por lo obtenido, tenemos que:

$$S(\text{tabulate } f' \ n) \in O(\max_{i=0}^{n-1} O(1)) \implies S(\text{tabulate } f' \ n) \in O(1) \implies S\text{contract}(n) = O(1) + O(1) = O(1)$$

$$\therefore S\text{contract} \in O(1)$$

2. $S\text{buildList}(n)$

$$S\text{buildList}(n) = S(\text{tabulate } f' \ n)$$

$$= \langle S(\text{tabulate } f' \ n) \in O(\max_{i=0}^{n-1} S(f' \ i)) \rangle$$

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

Ahora calculamos $S(f' \ i)$:

```
f' = (\i -> if even i then (nthS ys (div i 2))
      else f (nthS ys (div i 2)) (nthS xs (i - 1)))
```

Con la definición tenemos:

$$S(f' \ n) = O(1) \quad \text{even } i \\ + O(1) + O(1) + O(1) \quad f \ (nthS \ ys \ (div \ i \ 2)) \ (nthS \ xs \ (i - 1))$$

$$\therefore S(f' \ i) \in O(1)$$

Seguimos con el calculo de la profundidad de scanS y por lo que demostramos anteriormente, nos queda: $S\text{scanS}(n) = O(1) + S\text{scanS}(\lceil \frac{n}{2} \rceil) + O(1)$

Por suavidad y aplicando el Teorema Maestro, tenemos $a = 1$ y $b = 2$ y caemos en el 2^{do} caso del teorema.

$$\therefore S\text{sanS} \in O(\lg(n))$$