



Trabajo Práctico 2

Dada el TAD para secuencias que se encuentra definido en el archivo `Seq.hs`:

- Implementar en un módulo `ListSeq.hs` una instancia para el tipo listas. La implementación debe ser tan eficiente como sea posible (la nota dependerá de la corrección y de la eficiencia de la implementación.) Puede usar el módulo `Par` para paralelizar operaciones.
- Dar la especificación de costos para esta implementación de las funciones `filterS`, `showtS`, `reduceS` y `scanS`. Justificar.
- El tipo `Arr` de arreglos paralelos está dado en el archivo `Arr.hs`. Tiene las siguientes operaciones:
 - $length :: Arr\ a \rightarrow Int$
 $length\ a$ devuelve el tamaño de el arreglo a .
 - $(!) :: Arr\ a \rightarrow Int \rightarrow a$
 $a\ !\ i$ es la proyección i -ésima del arreglo a .
 - $subArray :: Int \rightarrow Int \rightarrow Arr\ a \rightarrow Arr\ a$
 $subArray\ i\ n\ a$ extrae n elementos del arreglo a comenzando por el elemento de la posición i .
 - $tabulate :: (Int \rightarrow a) \rightarrow Int \rightarrow Arr\ a$
 $tabulate\ f\ n$ construye un arreglo a de tamaño n tal que $a\ !\ i \equiv f\ i$ para todo $0 \leq i < n$.
 - $fromList :: [a] \rightarrow Arr\ a$
 $fromList\ xs$ construye un arreglo a partir de la lista xs .
 - $flatten :: Arr\ (Arr\ a) \rightarrow Arr\ a$
 $flatten\ a$ aplana un arreglo de arreglos.

Las operaciones tienen la siguiente especificación de costos:

Operación	W	S
$length\ p$ $p\ !\ i$ $subarray\ i\ n\ a$	$O(1)$	$O(1)$
$tabulate\ f\ n$	$O\left(\sum_{i=0}^n W(f\ i)\right)$	$O\left(\max_{i=0}^n S(f\ i)\right)$
$fromList\ xs$	$O(xs)$	$O(xs)$
$flatten\ a$	$O(a) + \sum_{i=0}^{ a -1} O(a\ !\ i)$	$O(\lg a)$

En un módulo `ArrSeq.hs`, dar una instancia de secuencias para tipo `Arr` que cumpla con la especificación de costos dados en clase.

Para evitar conflictos de nombres importar el módulo `Arr` en forma calificada:

```
import qualified Arr as A
```

Luego se puede acceder a las operaciones como `A.length`, etc. Para tener acceso a la operación `!` directamente (sin escribir `A.!`) importarla de la siguiente manera:

```
import Arr (!)
```

- Justificar la especificación de costos de las operaciones `filterS`, `showtS`, `reduceS` y `scanS`.

En las dos implementaciones tener especial cuidado con el orden de reducción de `reduceS` y `scanS`.

Entrega

El trabajo se podrá realizar en forma individual o en grupos de dos personas y se deberá entregar, tanto por mail (a tps.edyaII@gmail.com) como en forma impresa:

- Un archivo Haskell con las implementaciones.
- Un documento en formato PDF con las especificaciones de costos y sus justificaciones.

Fecha de entrega : 4/6/15

Costos Esperados de la Implementación con Arreglos

Operación	W	S
<i>emptyS</i> <i>singletonS</i> <i>lengthS</i> <i>nthS</i> <i>takeS s n</i> <i>dropS s n</i> <i>showtS s</i> <i>showlS s</i>	$O(1)$	$O(1)$
<i>append s t</i>	$O(s + t)$	
<i>fromList xs</i>	$O(xs)$	$O(xs)$
<i>joinS s</i>	$O(s) + \sum_{i=0}^{ s -1} O(s_i)$	$O(\lg s)$
<i>tabulateS f n</i>	$O\left(\sum_{i=0}^{n-1} W(f\ i)\right)$	$O\left(\max_{i=0}^{n-1} S(f\ i)\right)$
<i>mapS f s</i>	$O\left(\sum_{i=0}^{ s -1} W(f\ s_i)\right)$	$O\left(\max_{i=0}^{ s -1} S(f\ s_i)\right)$
<i>filterS f s</i>	$O\left(\sum_{i=0}^{ s -1} W(f\ s_i)\right)$	$O\left(\lg s + \max_{i=0}^{ s -1} S(f\ s_i)\right)$
<i>reduceS</i> \oplus <i>b s</i>	$O\left(s + \sum_{(x\oplus y)\in\mathcal{O}_r(\oplus,b,s)} W(x\oplus y)\right)$	$O\left(\lg s \max_{(x\oplus y)\in\mathcal{O}_r(\oplus,b,s)} S(x\oplus y)\right)$
<i>scanS</i> \oplus <i>b s</i>	$O\left(s + \sum_{(x\oplus y)\in\mathcal{O}_s(\oplus,b,s)} W(x\oplus y)\right)$	$O\left(\lg s \max_{(x\oplus y)\in\mathcal{O}_s(\oplus,b,s)} S(x\oplus y)\right)$