Algoritmos y Estructuras de Datos II
Primer parcial – 27 de septiembre de 2014

Aclaraciones

- El parcial es **a libro abierto**.
- Cada ejercicio debe entregarse **en hojas separadas**.
- Incluir en cada hoja el número de orden asignado, número de hoja, apellido y nombre.
- Al entregar el parcial, completar el resto de las columnas en la planilla.
- Cada ejercicio se calificará con **MB**, **B**, **R** o **M**, y podrá recuperarse independientemente de los demás. Para aprobar el parcial se puede tener hasta 1 (un) ejercicio **R** (regular), siempre que no se trate del ejercicio 1. Para más detalles, ver "Información sobre la cursada" en el sitio Web.

Ej. 1. Especificación

Un grupo de biólogos requiere un sistema que permita administrar los experimentos. Se dispone de n laboratorios. En cada momento puede haber hasta n experimentos en curso, no más de uno por laboratorio. Cuando se quiere iniciar un experimento, el sistema selecciona algún laboratorio libre donde llevarlo a cabo.

A cada experimento se le van registrando observaciones. Una observación puede tener tres posibles resultados: (1) observación anómala, (2) observación legítima que confirma la hipótesis, (3) observación legítima que contradice la hipótesis. Si en cualquier momento el número de observaciones anómalas supera el número de observaciones legítimas, el experimento se cancela al instante.

Un experimento está *avanzado* si cuenta con más de treinta observaciones legítimas. Si se quiere iniciar un nuevo experimento y no hay laboratorios libres, se desaloja alguno de los experimentos avanzados, de tal manera que el laboratorio que ocupaba se destine al nuevo experimento. En caso de que no haya laboratorios libres ni experimentos avanzados, no es posible iniciar un nuevo experimento.

Además, se quiere conocer en todo momento qué cantidad de los experimentos avanzados (dentro del conjunto de experimentos actualmente vigentes) son consistentes con la hipótesis. Un experimento es consistente con la hipótesis si la mayoría de las observaciones legítimas correspondientes a ese experimento confirman la hipótesis.

Especificar el sistema de administración de experimentos utilizando TADs.

Ej. 2. Inducción estructural

Este TAD modela árboles binarios sin información en los nodos.

TAD AB

generadores

nil	:		$\longrightarrow$	AB
bin	:	AB $\times$ AB	$\longrightarrow$	AB

observadores básicos

esNil?	:	AB	$\longrightarrow$	bool	
izq	:	AB a	$\longrightarrow$	AB	$\{\neg$esNil?(a)$\}$
der	:	AB a	$\longrightarrow$	AB	$\{\neg$esNil?(a)$\}$

otras operaciones

$\bullet \sqsubseteq \bullet$	:	AB $\times$ AB	$\longrightarrow$	bool
h	:	AB	$\longrightarrow$	nat

axiomas $\forall b, i, d$: AB.

$S0)$	nil $\sqsubseteq b$	$\equiv$	true
$S1)$	bin$(i,\ d) \sqsubseteq b$	$\equiv$	$\neg$esNil?$(b) \wedge_{\mathrm{L}} (i \sqsubseteq \mathrm{izq}(b) \wedge d \sqsubseteq \mathrm{der}(b))$
$H0)$	h(nil)	$\equiv$	0
$H1)$	h(bin$(i,\ d)$)	$\equiv$	1 + máx(h(i), h(d))
$N0)$	esNil?(nil)	$\equiv$	true
$N1)$	esNil?(bin$(i,\ d)$	$\equiv$	false
$I0)$	izq(bin$(i,\ d)$)	$\equiv$	i
$D0)$	der(bin$(i,\ d)$)	$\equiv$	d

Fin TAD

Interesa demostrar por inducción estructural la siguiente propiedad:

$$(\forall a : \mathrm{AB})(\forall b : \mathrm{AB})(a \sqsubseteq b \Rightarrow \mathrm{h}(a) \leq \mathrm{h}(b))$$

a) Escribir el predicado unario. Luego escribir, completo, **el esquema de inducción** a utilizar.
En el esquema, marcar **claramente** CB(s), PI(s), HI(s), TI(s) y el alcance de cada cuantificador.

b) Plantear el/los caso(s) base y resolverlo(s), justificando cada paso de la demostración.

c) Plantear el/los paso(s) inductivo(s) y resolverlo(s), justificando cada paso de la demostración. **Nota:** se pueden usar sin demostrar propiedades aritméticas y lógicas que no involucren árboles binarios. Aclarar explícitamente en qué pasos de la demostración se recurre a ellas.

Ej. 3. Rep y Abs

En este ejercicio se modela un editor de textos. La operación insertarLetra(a, e) inserta la letra a en la posición donde se encuentra actualmente el cursor. La operación moverCursor(b, e) mueve el cursor una letra hacia la izquierda (si b es false) o una letra hacia la derecha (si b es true).

TAD Letra **es** Nat

TAD Editor

generadores

abrir	:		$\longrightarrow$	editor
insertarLetra	:	letra × editor	$\longrightarrow$	editor
moverCursor	:	bool b × editor e	$\longrightarrow$	editor

$\{0 \leq \Delta(b, \text{posCursor}(e)) \wedge \Delta(b, \text{posCursor}(e)) < \text{long}(\text{texto}(e))\}$

observadores básicos

texto	:	editor	$\longrightarrow$	secu(letra)
posCursor	:	editor	$\longrightarrow$	nat

axiomas

$\forall$ a : letra, $\forall$ e : editor, $\forall$ b : bool

texto(abrir) $\equiv$ <>

texto(insertarLetra(a, e)) $\equiv$ subsecu(texto(e), 0, posCursor(e)) & $a \bullet$ <> & subsecu(texto(e), posCursor(e) + 1, long(texto(e)))

texto(moverCursor(b, e)) $\equiv$ texto(e)

posCursor(abrir) $\equiv$ 0

posCursor(insertarLetra(a, e)) $\equiv$ posCursor(e) + 1

posCursor(moverCursor(b, e)) $\equiv$ Δ(b, posCursor(e))

Fin TAD

Se agregan además las correspondientes operaciones y axiomas en otros TADs:

Δ : bool × nat $\longrightarrow$ nat

Δ(true, n) $\equiv$ $n + 1$

Δ(false, n) $\equiv$ $n - 1$

subsecu : secu(letra) s × nat i × nat j $\longrightarrow$ secu(letra) $\{0 \leq i \wedge i \leq j \wedge j < \text{long}(s)\}$

subsecu(s, i, j) $\equiv$ **if** $i = 0$ **then**
  if $j = 0$ **then** <> **else** prim(s) $\bullet$ subsecu(fin(s), 0, $j - 1$) **fi**
else
  subsecu(fin(s), $i - 1$, $j - 1$)
fi

Se cuenta con la siguiente estructura de representación. En el diseño, además de escribir el documento, se quiere poder determinar eficientemente la posición de ciertas cadenas de texto (*targets*):

string es secu(letra)

editor **se representa con** estr

estr es tupla ⟨
anteriores : string
siguientes : string
targets : conj(string)
indice : dicc(string, conj(nat))
indice_reverso : dicc(nat, string)
⟩

En la estructura estr:

- *anteriores*: letras desde el inicio del texto hasta la posición actual del cursor.
- *siguientes*: letras desde la posición actual del cursor hasta el final del texto.
- *targets*: *strings* que nos interesa buscar; puede ser cualquier conjunto de *strings*.
- *indice*: para cada *target*, el conjunto de posiciones del texto en los que hay una ocurrencia de dicha cadena. Decimos que hay una ocurrencia de la cadena s en la posición p del texto t si $s = \text{subsecu}(t, p, p + \text{long}(s))$.
- *indice_reverso*: para cada índice del texto en el que haya un target, le asocia la cadena correspondiente a dicho target.

Se pide:

a) Escribir el invariante de representación en castellano.

b,c) Escribir formalmente *b)* el invariante de representación y *c)* la función de abstracción.