

## Tarea IV: Prolog

(5 pts)

### Implementación:

(1.5 pts) – Considere la siguiente definición de los números de *Church*:

“El cero (0) es representado por la constante universal  $\underline{*}$ , luego cada número natural se representa como la aplicación repetida de una función  $\underline{up}$  sobre el cero. El número es entonces igual a la cantidad de aplicaciones de dicha función.”

Por ejemplo:

$\ast$	$\mapsto$	0
$\text{up}(\ast)$	$\mapsto$	1
$\text{up}(\text{up}(\ast))$	$\mapsto$	2
$\text{up}(\text{up}(\text{up}(\ast)))$	$\mapsto$	3

Tomando en cuenta la definición anterior, se desea que implemente en Prolog una calculadora para números de Church. De los tres argumentos, los primeros dos deben estar siempre instanciados. El tercero puede o no estarlo (*Nota: No debe transformar los números de Church en números enteros*).

- a) (0.5 pts) – Debe implantar el predicados `suma/3`, que tomar los primeros dos argumentos y almacenar, en el tercer argumento, el resultado de evaluar la suma de los mismos.

**Por ejemplo:**

```
suma(up(up(*)), up(*), X).  
X = up(up(up(*)))
```

- b) (0.5 pts) – Debe implantar el predicados `resta/3`, que tomar los primeros dos argumentos y almacenar, en el tercer argumento, el resultado de evaluar la resta de los mismos.

**Por ejemplo:**

```
resta(up(up(*)), up(*), X).  
X = up(*)
```

- c) (0.5 pts) – Debe implantar el predicados `producto/3`, que tomar los primeros dos argumentos y almacenar, en el tercer argumento, el resultado de evaluar el producto de los mismos.

Por ejemplo:

**Por ejemplo:**

```
producto(up(up(up(*))), up(up(*)), X).
X = up(up(up(up(up(up(*))))))
```

## Investigación:

- a) (0.25 pts) – Considere el dominio de valores  $\{a, b, c\}$ . ¿Cuántas posibles fórmulas *ground* (en el dominio sugerido) se pueden construir a partir de la fórmula  $P(x) \wedge Q(y)$ ?  
*Nota: Recuerde que una fórmula es ground cuando todas sus variables están completamente instanciadas con átomos en el dominio de valores. Por ejemplo:  $P(a) \vee Q(b)$ .*
- b) (0.25 pts) – Aumente ahora el dominio, con dos funcionales  $f$  (de aridad 1) y  $g$  (de aridad 2). ¿Cuántas posibles fórmulas *ground* (en el dominio sugerido) se pueden construir ahora a partir de la fórmula  $P(x) \wedge Q(y)$ ?
- c) (0.5 pts) – Plantee el conjunto de posibles valores que pueden reemplazar bien sea a  $x$  o a  $y$  en la fórmula de la pregunta anterior (*Nota: Puede utilizar las operaciones básicas de conjuntos, así como definiciones inductivas*).
- d) (0.5 pts) – Considere el siguiente programa:

```
p(f(X)) :- q(X, Y), r(Y).
```

```
q(g(X, Y), Z) :- r(X), r(Z), q(f(Z), a).
q(X, a).
```

```
r(f(f(b))).
r(b).
```

Diga un modelo para el programa (un conjunto que hace cierta la conjunción de todas las fórmulas).

- e) (0.25 pts) – Considerando el mismo programa, diga cuales predicados *ground* son ciertos sin ejecutar el programa (solo hechos).
- f) (0.25 pts) – Considerando el mismo programa, diga cuales predicados *ground* son ciertos ejecutando a lo sumo una sola vez las reglas.

- g) (0.5 pts) – Plantee una ecuación general recursiva  $H_k$ , que dado el conjunto de predicados *ground* que son ciertos ejecutando a lo sumo  $k - 1$  veces las reglas ( $H_{k-1}$ ), calcule dicho conjunto ejecutando las reglas a lo sumo  $k$  veces.
- h) (0.25 pts) – Sea  $k^*$  el valor más bajo de  $k$  tal que  $H_k = H_{k+1}$  (mínimo punto fijo de  $H$ ). Diga si el modelo que encontró en la parte (d) corresponde a un subconjunto de  $H_{k^*}$ . ¿Su respuesta sería igual para cualquier otro modelo del programa?

- i) (0.25 pts) – Considere el siguiente programa:

$p(X) :- q(X, Y), \text{not}(q(X, X)), \text{not}(r(Y)).$

$q(X, Y) :- r(X), p(Y).$   
 $q(X, a).$

$r(X) :- \text{not}(q(X, b)).$   
 $r(c).$

Se dice que un predicado  $p$  es de nivel  $k$  si todos los predicados negados que aparezcan en cada cuerpo donde  $p$  esté en la cabeza, están a lo sumo e nivel  $k - 1$ . El nivel de un predicado que no depende de otros que estén negados es 0.

Diga el nivel correspondiente a los predicados  $p$ ,  $q$  y  $r$

- j) (0.25 pts) – Calcule  $H_{k^*}$  para el programa anterior: Tomando en cuenta primero para los predicados de nivel 0 (de haberlos), luego para los predicados de nivel 1 (de haberlos) y así en adelante.
- k) (0.25 pts) – Si existen ciclos de predicados negados en un programa, los niveles antes mencionados no están bien definidos. Discuta como afecta esto el cálculo de  $H_{k^*}$ .

## Detalles de la Entrega:

La entrega de la tarea consistirá de un único archivo **t4\_<carné1>&<carné2>.pdf**, donde <carné1> y <carné2> son los números de carné de los integrantes de su equipo. Tal archivo debe ser un documento PDF con su implementación para las funciones pedidas y respuestas para las preguntas planteadas. Por ejemplo, si el equipo está conformado por 00-00000 y 11-11111, entonces su entrega debe llamarse: **t4\_00-00000&11-11111.pdf**.

La tarea deberá ser entregada a ambos profesores encargados del curso (Carlos Infante y Alexander Romero), únicamente a sus direcciones de correo electrónico institucionales: ( 13-10681@usb.ve y 13-11274@usb.ve ) a más tardar el Viernes 20 de Marzo, a las 11:59pm. VET.