Escuela Politécnica Superior

Universidad de Alcalá

CRA

Temas 4 y 5

Grado en Ingeniería Informática – Curso 2018/2019

PREGUNTAS TEÓRICAS

PREGUNTAS TEÓRICAS DEL EXAMEN – APARTADO 1

MAYO 2012

¿Es el número de combinadores de punto fijo finito?

No, no lo es.

Para demostrar que el número de combinadores de punto fijo es infinito, consideremos el siguiente λ-término:

$$\Gamma = \gamma\gamma \ldots \gamma \text{ (n veces con n} \geq 2)$$

Considerando también el siguiente alfabeto:

$$\{a_1, a_2, \ldots, a_{n-1}\} \text{ con n - 1 letras distintas}$$

Sea w una palabra de longitud n en dicho alfabeto, y γ sea:

$$\gamma \equiv \lambda a_1 a_2 \ldots a_{n-1} f.f(wf)$$

Entonces Γ es un combinador de punto fijo:

$$\Gamma\ F \equiv \gamma\gamma \ldots \gamma\ F \twoheadrightarrow (\lambda f.f(\gamma\gamma \ldots \gamma f))F \rightarrow F(\gamma\gamma \ldots \gamma\ F) \equiv F(\Gamma\ F)$$

Dado que, en el proceso anterior se puede elegir cualquier n ≥ 2, y hay infinitos números naturales, se concluye que se pueden construir infinitos combinadores de punto fijo.

MAYO 2012 (II)

Suponemos que dados dos λ-términos M y N son iguales, ¿Es cierto que $M \twoheadrightarrow N$ ó $N \twoheadrightarrow M$? Dar un ejemplo que lo demuestre.
No, cuando dos λ-términos son iguales no implica que uno reduzca al otro.

Sabemos que dos λ-términos son iguales si de uno podemos llegar a otro aplicando β−reducciones y expansiones.

Si consideramos como contraejemplo la aplicación del combinador paradójico:

$$\begin{gathered} Y \equiv \lambda f.(\lambda x.f(xx))\ (\lambda x.f(xx)) \\ \rightarrow (\lambda x.F(xx))\ (\lambda x.F(xx)) \\ \rightarrow F(\ (\lambda x.F(xx))\ (\lambda x.F(xx))\ ) \\ \leftarrow_\beta F(Y\ F) \end{gathered}$$

Tenemos que $YF \equiv F(YF)$, pero ni $YF \twoheadrightarrow F(YF)$ ni $F(YF) \twoheadrightarrow YF$.

MAYO 2014

¿Por qué se exige que, para la sustitución $\mathbf{M[N/x]}$ sea correcta, $\mathbf{BV\ (M) \backslash\ F\ V\ (N)} = \emptyset$?

Se exige que se cumpla esa condición porque, de no cumplirse, la sustitución interferiría en la ligadura de las variables de M, causando que una variable libre en la abstracción sea reemplazada por una o más variables ligadas dentro de la misma:

$$(\lambda x.M)\ N \equiv (\lambda x.xy)\ (\lambda x.y)$$
$$BV(M) = \{x\} \text{ asociado a } (\lambda x.xy)$$
$$FV(N) = \{y\} \text{ asociado a } (\lambda x.y)$$
$$BV(M) \cap FV(N)? \Rightarrow \emptyset$$

Si no se cumpliera la condición en la sustitución, una variable libre pasaría a ser ligada sin tener que serlo. Este problema se soluciona renombrando las variables de una de las expresiones en conflicto.

MAYO 2014 (II)

¿Todo λ-término admite forma normal?

No, algunos λ-términos no admiten forma normal.

Recordemos que un λ-término β-conversiones está en forma normal si no admite más aplicaciones de β-reducciones ni η−reducciones.

Como contraejemplo, consideremos al λ-término:

$$\Omega \equiv (\lambda x.xx)\ (\lambda x.xx)$$

Si intentamos llegar mediante reducciones a su forma normal, sucede esto:

$$(\lambda x.xx)\ (\lambda x.xx) \to (xx.xx)\ (\lambda x.xx) \to (\lambda x.xx)\ (\lambda x.xx) \to \dots$$

Encontramos que en Ω siempre se podrá aplicar reducción, por lo que nunca estará en forma normal.

MAYO 2014 (III)

Enunciar el teorema de Church-Russel. ¿Qué consecuencias se pueden extraer del mismo con respecto a la igualdad λ-términos y formas normales?

El teorema de Church-Russel enuncia que:

$$\text{Si } M = N\text{; existe } L \text{ tal que } M \twoheadrightarrow L \text{ y } N \twoheadrightarrow L$$

De este teorema se extraen las siguientes conclusiones:

- Si M = N y uno de ellos se encuentra en forma normal, entonces el otro reduce a esa forma normal.
$$\text{Si } N = M \text{ y } N \text{ está en forma normal, entonces } M \twoheadrightarrow N$$
- Si M = N y ambos se encuentran en forma normal, entonces M ≡ N
$$\text{Si } N = M\text{, } M \text{ y } N \text{ están en forma normal, entonces } M \equiv N$$

De esta última conclusión podemos deducir por ejemplo que xy ≠ xx o que λxy.x ≠ xy.y .

MAYO 2018

Definir la precondición más débil para una expresión si y un predicado q. Explicar el por qué de la definición.

Para una expresión SI y un predicado q, se define la precondición más débil y se denota como $wp(S,q)$ como la fórmula p de entre las que verifiquen $\models \{p\}\ S\ \{q\}$.

La precondición más débil de la expresión y de la postcondición. Formalizando un programa como transformación de predicados tal y como lo hace wp permite comenzar la especificación del resultado del programa completo y trabajar “hacia atrás”, cosa útil a la hora de verificar un programa.

EJERCICIOS DE CODIFICADORES

EJERCICIOS DE CODIFICADORES – APARTADO 2

NOR – MAYO 2012

Dados los λ-términos y el conectivo lógico ↓

P	Q	P ↓ Q
0	0	1
0	1	0
1	0	0
1	1	0

true ≡ λxy.x
false ≡ λxy.y
if ≡ λpxy.pxy

escribir un λ-término "nor" que codifique el conectivo lógico ↓

nor ≡ λpq.p false (q false true)

Verificar la corrección de la citada codificación.

CASO 1

nor false false ≡ (λ pq.p false (q false true)) false false
≡ ((λp. (λq.p false (q false true))) false) false
$_\beta$→ (λq. false false (q false true)) false $_\beta$→ false false (false false true)
≡ (λq. false false (q false true)) false $_\beta$→ false false (false false true)
$_\beta$→ (λy.y) (false false true) $_\beta$→ false false true
≡ (λxy.y) false true ≡ ((λx.(λy.y)) false) true
$_\beta$→(λy.y) true $_\beta$→ **true**

CASO 2

nor false true ≡ (λpq.p false (q false true)) false true
≡ ((λp.(λq.p false (q false true))) false) true
$_\beta$→(λq.false false (q false true)) true $_\beta$→ false false (true false true)
≡ (λxy.y) false (true false true) ≡ ((λx.(λy.y)) false) (true false true)
β→(λy.y) (true false true) $_\beta$→ true false true ≡ (λxy.x) false true
≡ ((λx.(λy.x)) false) true $_\beta$→ (λy.false) true $_\beta$→ **false**

CASO 3

nor true false ≡ (λpq.p false (q false true)) true false
≡ ((λp.(λq.p false (q false true))) true) false
$_\beta$→(λq.true false (q false true)) false $_\beta$→ true false (false false true)
≡ (λxy.x) false (false false true) ≡ ((λx.(λy.x)) false) (false false true)
β→(λy.false) (false false true) $_\beta$→ **false**

CASO 4

nor true true ≡ (λpq.p false (q false true)) true true
≡ ((λp.(λq.p false (q false true))) true) true
$_\beta$→(λq.true false (q false true)) true $_\beta$→ true false (true false true)
≡ (λxy.x) false (true false true) ≡ ((λx.(λy.x)) false) (true false true)
β→(λy.false) (true false true) $_\beta$→ **false**

NAND – MAYO 2012

Dados los λ-términos y el conectivo lógico |

P	Q	P | Q
0	0	1
0	1	1
1	0	1
1	1	0

true ≡ λxy.x
false ≡ λxy.y
if ≡ λpxy.pxy

escribir un λ-término "nor" que codifique el conectivo lógico |

nand ≡ λpq.p (q false true) true

Verificar la corrección de la citada codificación.

Nota: a partir de aquí se escribe true como T y false como F para mayor legibilidad.

CASO 1

nand F F ≡ (λpq.p(q F T) T)F F
≡ ((λp.(λq.p(q F T) T)) F) F
$_\beta$→(λq.F (q F T) T) F $_\beta$→ F (F F T) T
≡ (λxy.y) (F F T) T ≡ ((λx.(λy.y))(F F T)) T
$_\beta$→(λy.y) true $_\beta$→ **true**

CASO 2

nand F T ≡ (λpq.p(q F T) T)F T
≡ ((λp.(λq.p(q F T) T)) F) T
$_\beta$→(λq.F (q F T) T) T $_\beta$→ F (T F T) T
≡ (λxy.y) (T F T) T ≡ ((λx.(λy.y))(T F T)) T
$_\beta$→(λy.y) T $_\beta$→ **true**

CASO 3

nand T F ≡ (λpq.p(q F T) T)T F
≡ ((λp.(λq.p(q F T) T)) T) F
$_\beta$→(λq.T (q F T) T) F $_\beta$→ T (F F T) T
≡ (λxy.x) (F F T) T ≡ ((λx.(λy.x))(F F T)) T
$_\beta$→(λy.(F F T)) T $_\beta$→ F F T ≡ (λxy.y) F T
≡ ((λx.(λy.y))F)T $_\beta$→ (λy.y) T $_\beta$→ **true**

CASO 4

nand T T ≡ (λpq.p(q F T) T)T T
≡ ((λp.(λq.p(q F T) T)) T) T
$_\beta$→(λq.T (q F T) T) T $_\beta$→ T (T F T) T
≡ (λxy.x) (T F T) T ≡ ((λx.(λy.x))(T F T)) T
$_\beta$→(λy.(t F T)) T $_\beta$→ T F T ≡ (λxy.x) F T
≡ ((λx.(λy.x))F)T $_\beta$→ (λy.F) T $_\beta$→ **false**

IMPLICA – MAYO 2016

Dados los λ-términos y el conectivo lógico →

P	Q	P → Q
0	0	1
0	1	0
1	0	1
1	1	1

$$\textbf{true} \equiv \lambda xy.x$$
$$\textbf{false} \equiv \lambda xy.y$$
$$\textbf{if} \equiv \lambda pxy.pxy$$

escribir un λ-término "nor" que codifique el conectivo lógico →

$$\textbf{implica} \equiv \lambda pq.p\ (q\ \text{false true})\ \text{true}$$

Verificar la corrección de la citada codificación.

CASO 1

$$\text{implica F F} \equiv (\lambda pq.p\ T(q\ F\ T))F\ F \equiv ((\lambda p.(\ \lambda q.p\ T(q\ F\ T)))\ F)\ F$$
$$_{\beta}\!\!\to (\lambda q.F\ T(q\ F\ T))\ F)\ F\ _{\beta}\!\!\to F\ T\ (F\ F\ T) \equiv (\lambda xy.y)\ T\ (F\ F\ T)$$
$$\equiv ((\lambda x.(\lambda y.y))T)\ (F\ F\ T)\ _{\beta}\!\!\to (\lambda y.y)\ (F\ F\ T)\ _{\beta}\!\!\to F\ F\ T$$
$$\equiv (\lambda xy.y)\ F\ T \equiv ((\lambda x.(\ \lambda y.y))F)T$$
$$_{\beta}\!\!\to (\ \lambda y.y)\ T\ _{\beta}\!\!\to \textbf{true}$$

CASO 2

$$\text{implica F T} \equiv (\lambda pq.p\ T(q\ F\ T))F\ T \equiv ((\lambda p.(\ \lambda q.p\ T(q\ F\ T)))\ F)\ T$$
$$_{\beta}\!\!\to (\lambda q.F\ T(q\ F\ T))\ T\ _{\beta}\!\!\to F\ T\ (T\ F\ T) \equiv (\lambda xy.y)\ T\ (T\ F\ T)$$
$$\equiv ((\lambda x.(\lambda y.y))T)\ (T\ F\ T)\ _{\beta}\!\!\to (\lambda y.y)\ (T\ F\ T)\ _{\beta}\!\!\to T\ F\ T$$
$$\equiv (\lambda xy.x)\ F\ T \equiv ((\lambda x.(\ \lambda y.x))F)T$$
$$_{\beta}\!\!\to (\ \lambda y.F)\ T\ _{\beta}\!\!\to \textbf{false}$$

CASO 3

$$\text{implica T F} \equiv (\lambda pq.p\ T(q\ F\ T))T\ F \equiv ((\lambda p.(\ \lambda q.p\ T(q\ F\ T)))\ {\color{red}T})\ F$$
$$_{\beta}\!\!\to (\lambda q.F\ T(q\ F\ T))\ {\color{red}T})\ F\ _{\beta}\!\!\to T\ T\ (F\ F\ T) \equiv (\lambda xy.x)\ T\ (F\ F\ T)$$
$$\equiv ((\lambda x.(\lambda y.x))T)\ (F\ F\ T)\ _{\beta}\!\!\to (\lambda y.T)\ (F\ F\ T)\ _{\beta}\!\!\to \textbf{true}$$

CASO 4

$$\text{implica T T} \equiv (\lambda pq.p\ T(q\ F\ T))T\ T \equiv ((\lambda p.(\ \lambda q.p\ T(q\ F\ T)))\ T)\ T$$
$$_{\beta}\!\!\to (\lambda q.T\ T(q\ F\ T))\ {\color{red}T})\ F\ _{\beta}\!\!\to T\ T\ (T\ F\ T) \equiv (\lambda xy.x)\ T\ (T\ F\ T)$$
$$\equiv ((\lambda x.(\lambda y.x))T)\ (T\ F\ T)\ _{\beta}\!\!\to (\lambda y.T)\ (T\ F\ T)\ _{\beta}\!\!\to \textbf{true}$$

EJERCICIOS DE PROBAR COMBINADORES DE PUNTO FIJO

EJERCICIOS DE PUNTO FIJO – APARTADO 2

Probar que el λ-término Θ definido por:

$$A \equiv \lambda xy.y(xxy)$$
$$\Theta \equiv AA$$

es un combinador de punto fijo.

Si es un combinador de punto fijo, debe de cumplir la siguiente propiedad:

- Que sus variables sean ligadas
- $\Theta F \equiv F(\Theta F)$

Si lo expandimos:
$\Theta F \equiv F(\Theta F) \equiv (\lambda xy.y(xxy))AF \equiv ((xx.(\lambda y.y(xxy)))A)F$
${}_{\beta}\!\rightarrow (\lambda y.y(AAy))F \;{}_{\beta}\!\rightarrow (f(AAF) \equiv F(\Theta F)$

Vemos que, además de cumplir la propiedad del punto fijo, todas sus variables son ligadas. Por tanto, concluimos que Θ es un combinador, por ser un término cerrado, de punto fijo, es decir, que satisface $\Theta F \equiv F(\Theta F)$.

RESTO DE DIVISÓN ENTERA – JUNIO 2012

Usando Θ, definir un λ-término que calcule el resto de la división de dos números naturales positivos codificados *a la Church*.

$$resto(a,b) = \begin{cases} n & si \quad a < b \\ resto(a-b,b) & en \quad cualquier\ otro\ caso \end{cases}$$

Utilizaremos dos λ-términos:

- $\mathbf{esMenorQue} \equiv \lambda nm.iszero(sub\ (suc\ n)\ m)$
- $\mathbf{resto} \equiv \Theta\ (\lambda fnm.(esMenorQue\ n\ m)\ n\ (f\ (sub\ n\ m)\ m)$

Ejemplo: 7 % 3

$(esMenorQue\ 7\ 3)\ False \rightarrow (f\ (sub\ 7\ 3)\ 3) \Rightarrow n = 4$

$(esMenorQue\ 4\ 3)\ False \rightarrow (f\ (sub\ 4\ 3)\ 3) \Rightarrow n = 1$

$(esMenorQue\ 4\ 3)\ True \rightarrow n = 1$

Ejemplo: 4 % 6

$(esMenorQue\ 4\ 6)\ True \rightarrow (f\ (sub\ 4\ 6)\ 6) \rightarrow n = 4$

Nota: $(esMenorQue\ 4\ 6)$; $(suc\ 4) = 5$; $(sb\ 5\ 6) = -1$; $iszero\ (-1) = true$

SUMA DE ELEMENTOS DE UNA LISTA – MAYO 2014

Usando Θ, definir un λ-término que calcule la suma de todos los elementos de una lista de números enteros.

$$sumaLista(l) = \begin{cases} 0 & si \quad [\ ] \\ (l) + sumaLista(tl(l)) & en \quad cualquier\ otro\ caso \end{cases}$$

sumaLista ≡ Θ (λ fl. (null l) 0 (add (hd l) (f (tl l)))

COCIENTE DE DOS NÚMEROS POSITIVOS

Usando Θ, definir un λ-término que calcule el cociente de la división de dos números naturales positivos codificados *a la Church*.

$$cociente(a,b) = \begin{cases} 0 & si \quad a < b \\ 1 + cociente(a-b,b) & en \quad cualquier\ otro\ caso \end{cases}$$

Utilizaremos dos λ-términos:

- **esMenorQue** ≡ λ.nm.iszero(sub (suc n) m)
- **cociente** ≡ Θ (λfnm.(esMenorQue n m) 0 (suc(f(sub n m) m))

FACTORIAL DE UN NÚMERO

Usando Θ, definir un λ-término que calcule el factorial de un número natural codificado *a la Church*. Después, calcular el factorial de dos.

$$factorial(a,b) = \begin{cases} 1 & si \quad n = 0 \\ n \cdot factorial(n-1) & en \quad cualquier\ otro\ caso \end{cases}$$

fact ≡ Θ (λfn.(iscero n) 1 (mult n (f (pred n)))

Para calcular el factorial de 2:

Fact 2 ≡ (ΘF)2 ≡ (AAF) 2 ≡ ((λxy.y(xxy))AF)2 ↠ (F (AAF))2 ≡ (F (ΘF))2 ≡ (F fact) 2

≡ ((λfn.(iszero n) 1 (mult n(y(pre n)))) fact) 2 ↠ (iszero 2) 7 (mult 2(fact (pre 2)))
↠ mult 2(fact 1) ≡ mult 2((ΘF)7) ….. y pun se convierte en mult 2(1) ↠ 2

MÁXIMO COMÚN DIVISOR

Usando Θ, que satisface definir un λ-término que calcule el máximo común divisor de dos números naturales codificados *a la Church*.

$$mcd(a,b) = \begin{cases} a & si \quad b = 0 \\ mcd(b, resto(a,b)) & en \quad cualquier\ otro\ caso \end{cases}$$

Utilizaremos 3 λ-términos:

- **esMenorQue** ≡ λ.nm.iszero(sub (suc n) m)
- **resto** ≡ Θ (λfnm.(esMenorQue n m) n (f (sub n m) m)
- **mcd** ≡ Θ (λfnm.(iszero n) m (mcd (resto m n)n)

Ejemplo:

mcd(36, 27) → mcd(36%27, 27) → mcd(27%9, 9) → mcd(0, 9) = 9

EJERCICIOS DE PUNTO FIJO – APARTADO 3

En λ-cálculo, a partir de la codificación de pares ordenados, se pueden codificar listas como sigue.

$$\begin{array}{lcl} \textbf{nil} & \equiv & \lambda z.z \\ \textbf{cons} & \equiv & \lambda xy.\ \textbf{pair}\ \textbf{false}(\ \textbf{pair}\ xy) \\ \textbf{null} & \equiv & \textbf{fst} \\ \textbf{hd} & \equiv & \lambda z.\ \textbf{fst}\ (\ \textbf{snd}\ z) \\ \textbf{tl} & \equiv & \lambda z.\ \textbf{snd}\ (\ \textbf{snd}\ z). \end{array}$$

Teniendo en cuenta lo anterior, probar que

Y ≡ λf.(λx.f(xx))(λx.f(xx))

Es un combinador de punto fijo.

Y es un combinador debido a que las variables están ligadas, por tanto, es un término cerrado.

Para probar que es un punto fijo, debe cumplirse que ΘF ≡ F(ΘF)

Si lo expandimos:
YF ≡ F(YF) ≡ (λf(λxf(xx))(λx.f(xx)))F $_\beta$→ (λx.F(xx))(λx.F(xx))
$_\beta$→ F((λx.F(xx))(λx.F(xx))) → F((λf.(λx.f(xx))(λx.f(xx)))F) ≡ F(YF)

Vemos que se cumple, y es, por tanto, un punto fijo.

PERTENECE A LISTA

Usando Y, definir un λ-término que, aplicando un número natural codificado *a la Church* y una lista de naturales codificados en λ cálculo, determine si el número natural pertenece a la lista dada o no.

$$pertenece(a,l) = \begin{cases} false & si \quad [\ ] \\ true & si \quad esigual(a, hd(l)) \\ pertenece(a, tl(l)) & en \quad cualquier\ otro\ caso \end{cases}$$

Utilizaremos 2 λ-términos:

- **esigual** ≡ λab.iszero(sub a b)
- **pertenece** ≡ Y(λfal.(null l) false ((esigual a hd(l)) true (f a (tl(l)))

LONGITUD

Usando Y, definir un λ-término que, calcule la longitud de una lista codificada en λ-cálculo. Además, aplicarlo a la lista L = [1]

$$long(l) = \begin{cases} 0 & si \quad [\ ] \\ 1 + long(tl(l)) & en \quad cualquier\ otro\ caso \end{cases}$$

long ≡ λfl.(null l) 0 (add 1 (g (tl(l)))

INVERTIR

Usando Y, definir un λ-término que, de una lista codificada en λ-cálculo, devuelva su inversa.

$$invertir(L) = \{\ invertiraux(L, nil)$$

$$invertiraux(L1, L2) = \begin{cases} L2 & si \quad L1 = [\ ] \\ invertiraux(tl(L1), cons(hd(L2), L2)) & en \quad cualquier\ otro\ caso \end{cases}$$

ENLACES ÚTILES

https://jwodder.freeshell.org/lambda.html