SOLUCIÓN TALLER CÁLCULO LAMBDA

Octubre 31 de 2023

Presentado por:
Ivonne Castaño Osorio
Juan Andrés García Moreno

1. ¿Se puede simular una Máquina de Turing usando el Cálculo λ? Explique su respuesta.

La relación entre la máquina de Turing y el cálculo lambda es fundamental para la teoría de la computación y se conoce como la tesis de Church-Turing. Esta tesis sostiene que cualquier función que sea computable puede ser calculada por una máquina de Turing o expresada en el cálculo lambda.

Para simular una máquina de Turing con el cálculo lambda, uno podría representar el estado de la máquina de Turing y su cinta como datos en el cálculo lambda, y luego definir funciones lambda para las operaciones básicas de la máquina de Turing. Estas funciones se pueden componer para simular cualquier programa de máquina de Turing.

Una ***Máquina de Turing*** puede realizar seis tipos de operaciones fundamentales:

1. **Leer:** El cabezal de la máquina lee el símbolo en el cuadrado actual de la cinta.
2. **Escribir:** El cabezal de la máquina puede cambiar el símbolo en el cuadrado actual a un nuevo valor.
3. **Mover hacia la izquierda:** El cabezal de la máquina se mueve un cuadrado hacia la izquierda en la cinta.
4. **Mover hacia la derecha:** El cabezal de la máquina se mueve un cuadrado hacia la derecha en la cinta.
5. **Cambiar de estado:** La máquina puede cambiar su estado interno, lo cual puede afectar su comportamiento en los siguientes pasos.
6. **Detenerse:** La máquina puede detener su ejecución.

Estas operaciones se pueden combinar para realizar cálculos complejos.

En el ***Cálculo Lambda***, la cinta de una Máquina de Turing puede ser representada como una lista de símbolos. Cada símbolo en la cinta puede ser representado como un átomo, y la cinta entera puede ser representada como una lista de estos átomos.

Esta es una posible representación de la cinta en términos de cálculo lambda:

λc.λn.λf.λx. f (c (λg.λh.h (g f)) (λu.x) (λu.u) n)

En esta representación, c es la cinta, n es un índice en la cinta, f es una función que se aplica a cada símbolo en la cinta, y x es un valor por defecto que se devuelve si el índice n está fuera de los límites de la cinta.

Las operaciones básicas de una Máquina de Turing pueden ser representadas en el ***Cálculo Lambda*** de la siguiente manera:

1. **Leer:** La operación de lectura puede ser representada como una función lambda que toma como entrada el estado actual de la cinta y devuelve el símbolo en la posición actual del cabezal. En notación lambda, esto podría ser algo como $\lambda t.t[n]$, donde t es la cinta y n es la posición actual del cabezal.
2. **Escribir:** La operación de escritura puede ser representada como una función lambda que toma como entrada el estado actual de la cinta y un nuevo símbolo, y devuelve una nueva cinta donde el símbolo en la posición actual del cabezal ha sido reemplazado por el nuevo símbolo. En notación lambda, esto podría ser algo como $\lambda t.\lambda s.t[n:=s]$, donde t es la cinta, s es el nuevo símbolo y n es la posición actual del cabezal.
3. **Mover hacia la izquierda:** La operación de mover hacia la izquierda puede ser representada como una función lambda que toma como entrada la posición actual del cabezal y devuelve una nueva posición que es una menos que la entrada. En notación lambda, esto podría ser algo como $\lambda n.n-1$.
4. **Mover hacia la derecha:** Similarmente, la operación de mover hacia la derecha puede ser representada como una función lambda que toma como entrada la posición actual del cabezal y devuelve una nueva posición que es una más que la entrada. En notación lambda, esto podría ser algo como $\lambda n.n+1$.
5. **Cambiar de estado:** La operación de cambio de estado puede ser representada como una función lambda que toma como entrada el estado actual de la máquina y devuelve un nuevo estado. El nuevo estado podría depender del símbolo actualmente bajo el cabezal y del estado actual de la máquina.
6. **Detenerse:** La operación de detenerse puede ser representada simplemente como una función lambda que no hace nada, o posiblemente que devuelve algún valor especial para indicar que la máquina se ha detenido.

2. Define las funciones “Mayor que” (<) y “Menor que” (>):

Menor que: $\lambda mn.((m(\lambda mnsz.n))(\lambda sz.z)n)$

Esta función devuelve cero si el primer argumento es menor que el segundo, y un número distinto de cero en caso contrario.

Esta es una función de dos argumentos m y n que representan dos números naturales en notación de Church.

$m(\lambda mnsz.n)$: Esta parte de la expresión aplica la función $\lambda mnsz.n$ a m. En el cálculo lambda, los números naturales se representan como el número de aplicaciones de una función a un argumento. Por lo tanto, esta expresión efectivamente reduce m en uno cada vez que se aplica, hasta que m llega a cero.

$(\lambda sz.z)n$: Esta es una función que siempre devuelve 0, en notación de Church, sin importar los argumentos que se le pasen. Se aplica a n.

((m(λmnsz.n))(λsz.z)n): La expresión completa toma el resultado de m(λmnsz.n) y lo aplica a (λsz.z)n. Si m es menor que n, entonces eventualmente reducirá m a cero antes de que se hayan agotado todas las aplicaciones de la función a n. En este caso, el resultado será cero. Si m no es menor que n, entonces no se agotarán todas las aplicaciones de la función, y el resultado será un número distinto de cero.

Mayor que: λmn.((n(λmnsz.n))(λsz.z)m)

Esta función devuelve cero si el primer argumento es mayor que el segundo, y un número distinto de cero en caso contrario.

Al igual que la anterior, esta es una función de dos argumentos m y n que representan dos números naturales en notación de Church.

n(λmnsz.n): Esta parte de la expresión aplica la función λmnsz.n a n. En el cálculo lambda, los números naturales se representan como el número de aplicaciones de una función a un argumento. Por lo tanto, esta expresión efectivamente reduce n en uno cada vez que se aplica, hasta que n llega a cero.

(λsz.z)m: Esta es una función que siempre devuelve 0, en notación de Church, sin importar los argumentos que se le pasen. Se aplica a m.

((n(λmnsz.n))(λsz.z)m): La expresión completa toma el resultado de n(λmnsz.n) y lo aplica a (λsz.z)m. Si n es menor que m, entonces eventualmente reducirá n a cero antes de que se hayan agotado todas las aplicaciones de la función a m. En este caso, el resultado será cero. Si n no es menor que m, entonces no se agotarán todas las aplicaciones de la función, y el resultado será un número distinto de cero.

3. Evalúe las siguientes sustituciones:

a. [(uv)/x] **(λy.x(λw.vwx))**
No es posible ya que x no es una variable ligada.

b. [(λy.xy)/x]**(λy.x(λx.x))**
Hay ambigüedad en la variable a sustituir, por lo que no es posible a menos que se asuman combinaciones específicas.

c. [(λy.vy)/x] **(y (λv.xv))**
No es posible realizar las sustituciones debido a que x no esta ligada.

d. [(uv)/x] **(λx.zy)**
Aunque x está ligada, la misma no está en el cuerpo por lo tanto si se sustituye y reduce el resultado sería zy.

4. Reduzca los siguientes términos a la forma β-Normal:

a. (λx.xy)(λu.vuu)

→β (λu.vuu)y

→β **vyy ≡ β-Normal**

b. (λxy.yx)uv

→β (λy.yu)v

→β **vu ≡ β-Normal**

c. (λx.x(x(yz))x)(λu.uv)

→β (λu.uv)((λu.uv)(yz))(λu.uv)

→β (λu.uv)((yv)(z))(λu.uv)

→β (λu.uv)(yvz)(λu.uv)

→β **(yv)(vz)(λu.uv) ≡ β-Normal**

d. (λx.xxy)(λy.yz)

→α (λx.xxy)(λp.pq)

→β (λp.pq)(λp.pq)y

→β (λp.pq)yq

→β **yqq ≡ β-Normal**

e. (λxy.xyy)(λu.uyx)

→α (λxy.xyy)(λp.pqr)

→β λy.(λp.pqr)yy

→β **λy.yqry ≡ β-Normal**

f. (λxyz.xz(yz))((λxy.yx)u)((λxy.yx)v)w

→α (λxyz.xz(yz))((λab.ba)u)((λmn.nm)v)w

→β (λxyz.xz(yz))(λb.bu)((λn.nv)w)

→β (λxyz.xz(yz))(λb.bu)(wv)

→β (λyz.(λb.bu)z(yz))(wv)

→β (λyz.(zu)(yz))(wv)

→β (λz.(zu)(wz))(v)

→β **(vu)(wz) ≡ β-normal**

5. Pruebe que **$(\lambda xyz.xzy)(\lambda xy.x)$** $=_\beta$ **$(\lambda xy.x)(\lambda x.x)$**

	$(\lambda xyz.xzy)(\lambda xy.x)$	$\equiv$	$(\lambda xy.x)(\lambda x.x)$
$\rightarrow_\alpha$	$(\lambda xyz.xzy)(\lambda mn.m)$	$\equiv$	$\rightarrow_\alpha$ $(\lambda xy.x)(\lambda m.m)$
$\rightarrow_\beta$	$\lambda yz.(\lambda mn.m)zy$	$\equiv$	$\rightarrow_\beta$ $\lambda y.(\lambda m.m)$
$\rightarrow_\beta$	$\lambda yz.(\lambda n.z)y$	$\equiv$	$\rightarrow_\beta$ $\lambda ym.m$
$\rightarrow_\beta$	**$\lambda yz.z$**	$\equiv$	**$\lambda ym.m$**