

# Reducción- $\beta$ en el Cálculo- $\lambda$ Puro

**José de Jesús Lavalle Martínez**

Benemérita Universidad Autónoma de Puebla  
Facultad de Ciencias de la Computación  
Fundamentos de Lenguajes de Programación CCOS 255

1 Reducción- $\beta$

2 Ejercicios

## Definición 1

La relación reducción- $\beta$ , en símbolos  $\rightarrow_\beta$ , es inducida por el axioma para reducción- $\beta$ , también llamado regla de reducción- $\beta$ :

$$(\lambda x.M)N \rightarrow_\beta M[x := N]$$

## Definición 1

La relación reducción- $\beta$ , en símbolos  $\rightarrow_\beta$ , es inducida por el axioma para reducción- $\beta$ , también llamado regla de reducción- $\beta$ :

$$(\lambda x.M)N \rightarrow_\beta M[x := N]$$

y tres reglas adicionales para establecer que la regla de reducción- $\beta$  se puede aplicar en cualquier lugar de un término- $\lambda$ :

$$\frac{M \rightarrow_\beta M'}{(\lambda x.M) \rightarrow_\beta (\lambda x.M')}$$

$$\frac{M \rightarrow_\beta M'}{MN \rightarrow_\beta M'N}$$

$$\frac{N \rightarrow_\beta N'}{MN \rightarrow_\beta MN'}$$

- Como nuestra experiencia nos muestra, evaluar o reducir una expresión a su mínima expresión no siempre se logra en un solo paso de reducción.

- Como nuestra experiencia nos muestra, evaluar o reducir una expresión a su mínima expresión no siempre se logra en un solo paso de reducción.
- Como nuestro conocimiento nos demanda, debemos definir la cerradura reflexiva y transitiva de la relación reducción- $\beta$  para poder realizar una secuencia de reducciones- $\beta$ .

## Definición 2

La relación reflexiva y transitiva  $\rightarrow_{\beta}^*$  de la relación  $\rightarrow_{\beta}$  se define mediante:

$$\frac{}{M \rightarrow_{\beta}^* M}$$
$$\frac{M \rightarrow_{\beta} M'' \quad M'' \rightarrow_{\beta}^* M'}{M \rightarrow_{\beta}^* M'}$$

## Definición 3

Un *redex*- $\beta$  es un término- $\lambda$  de la forma  $(\lambda x.M)N$ , *redex* es un acrónimo de *reducible expression*.

- $(\lambda x.x)z \rightarrow_{\beta} z$ . La función identidad  $(\lambda x.x)$  regresa su argumento sin cambiarlo.



- $(\lambda x.yxx)z \rightarrow_{\beta} yzz$ . Se duplica el argumento  $z$ .

- $(\lambda x.y)z \rightarrow_{\beta} y$ . La función constante  $y$  borra el argumento  $z$  y siempre regresa  $y$ .



$$\begin{aligned} ((\lambda x. (\lambda y. zxy))z_1)z_2 &\rightarrow_{\beta} (\lambda y. z z_1 y)z_2 \\ &\rightarrow_{\beta} (z z_1)z_2. \end{aligned}$$

Podemos pensar a  $z$  como una función que necesita los argumentos  $z_1$  y  $z_2$  para poderse evaluar.

## Ejemplos de reducciones- $\beta$ II



$$\begin{aligned}(\lambda x.x)(\lambda y.y)z &\rightarrow_{\beta} (\lambda y.y)z \\ &\rightarrow_{\beta} z.\end{aligned}$$

El argumento a su vez puede ser una función, así que el cálculo- $\lambda$  es de **alto orden**.

- Al término- $\lambda$   $(\lambda z.(\lambda x.z(zx)))$  a veces se le llama duplicador, suponga que  $z_1$  es una función de un argumento y que  $z_2$  es su argumento, entonces tenemos:

$$\begin{aligned}((\lambda z.(\lambda x.z(zx)))z_1)z_2 &\rightarrow_{\beta} (\lambda x.z_1(z_1x)z_2) \\ &\rightarrow_{\beta} z_1(z_1z_2).\end{aligned}$$

- El término- $\lambda$   $(\lambda x.xx)((\lambda y.y)z)$  se puede reducir a  $(\lambda x.xx)z$ , pero también se puede reducir a  $((\lambda y.y)z)((\lambda y.y)z)$ . Así reducción- $\beta$  **no es determinista**. Sin embargo ambos términos se reducen a  $zz$ , así reducción- $\beta$  no es determinista pero **es confluente**. Es decir si  $M \rightarrow_{\beta}^* M'$  y  $M \rightarrow_{\beta}^* M''$ , entonces existe un  $N$  tal que  $M' \rightarrow_{\beta}^* N$  y  $M'' \rightarrow_{\beta}^* N$ .

- El término- $\lambda$   $(\lambda x.xx)(\lambda x.xx)$  se reduce a si mismo.

$$(\lambda x.xx)(\lambda x.xx) \rightarrow_{\beta} (\lambda x.xx)(\lambda x.xx).$$

Así reducción- $\beta$  **no siempre termina.**

## Definición 4

Un término- $\lambda$  que no contiene un redex- $\beta$  se dice que es una **forma normal- $\beta$**  o sólo **forma normal**.



## Definición 4

Un término- $\lambda$  que no contiene un redex- $\beta$  se dice que es una **forma normal- $\beta$**  o sólo **forma normal**.

Los siguientes son ejemplos de formas normales- $\beta$ :  $x$ ,  $(\lambda x.x)$ ,  $x(\lambda y.y)$ .

Intuitivamente, una secuencia de reducciones- $\beta$  modela un cálculo y una forma normal- $\beta$  es el resultado de un cálculo, si tal resultado existe.

A un término- $\lambda$  cerrado también se le llama combinador. Los siguientes son cuatro combinadores famosos:

- 1  $I = (\lambda x.x),$
- 2  $K = (\lambda x.(\lambda y.x)),$
- 3  $S = (\lambda x.(\lambda y.(\lambda z.xz(yz))))),$
- 4  $\Omega = (\lambda x.xx)(\lambda x.xx).$

- 1 Implemente las funciones:

```
> val beta = fn : t -> t
> val redex = fn : t -> bool
> val betaster = fn : t -> t
```

- 2 Realice a **mano** reducciones- $\beta$  de un paso y multi pasos para cada uno de los casos de prueba que están en el archivo `http://aleteya.cs.buap.mx/~jlavalle/flp/betabetasterCasosPrueba.txt`.
- 3 Pruebe su implementación con los casos de prueba que están en el archivo `http://aleteya.cs.buap.mx/~jlavalle/flp/betabetasterCasosPrueba.txt`.