README de la Practica 5

Escamilla Soto Cristopher Alejandro, 314309253
Montiel Manriquez Ricardo, 314332662

13 de Mayo del 2022

Ejercicio 1:

Definimos el proceso curry.

Definimos un nuevo lenguaje en el que se modificaran las definiciones de lambda.

Creamos el parser del nuevo lenguaje L9.

Ahora si creamos la función que hará el proceso de curry-let. Este proceso se encarga de currificar las expresiones lambda así como las aplicaciones de función.

Ejercicio2:

Definimos el proceso type-const.

Definimos un nuevo lenguaje en el que se modificaran las definiciones de quot

Definimos el parser del nuevo lenguaje L10

Función que nos regresa el tipo de una constante

Creamos la función que hará el proceso de type-const

Ejercicio 3:

Implementar la función J con base en las reglas definidas para el algoritmo J sobre el lenguaje L10.

Esta función recibe una expresión del lenguaje y un contexto inicial, y regresa el tipo correspondiente a la expresión.

J: (type L10) ambiente -¿boolean — Donnde t1 y t2 son del mismo tipo.

variables utilizamos la función auxiliar para buscar directamente en el ambiente

Cuando tenemos constantes pasamos el tipo que esta contenido en el const

Necesitamos encontrar la ultima expresión y obtener su tipo

En el caso del if conocemos su estructura y por lo tanto debemos verificar que la primera expresión (e0) sea de tipo Bool Las expresiones then(e1) else(e2) sean del mismo tipo

Cuando tenemos lambdas el tipo sera definido por el cuerpo

Cuando tenemos let nuevamente revisamos el body y añadimos (x t) al ambiente

Cuando tenemos letc nuevamente revisamos el body y añadimos (x t) al ambiente

Cuando tenemos letfun nuevamente revisamos el body y añadimos (x t) al ambiente

Cuando tenemos list. Si es vacia devolvemos List. Si no, regresamos los tipos de los elementos. Si todos son del mismo tipo unificamos y devolvemos el tipo de lo contrario un error.

Utilizamos and map para verificar que todos sean del mismo tipo

Definimos el proceso type-infer que se encarga de quitar la anotación de tipo Lambda y sustituirlas por el tipo '(T $\rightarrow$ T) que corresponda a la definición de la función.

Ejercicio 4:

Se encarga de quitar la anotación de tipo Lambda y sustituirlas por el tipo '(T $\rightarrow$ T) que corresponda a la definición de la función. Y sustituye las anotaciones de tipo List por el tipo (List of T) de ser necesario.

type-infer:: L10 $\rightarrow$ L10

Para let solo nos fijamos en el tipo de de t si es un List, lo denombra a List of. Para let solo nos fijamos en el tipo de de t si es List o Lambda, colocamos T$\rightarrow$T y List of. Para letfun siempre nos fijamos en el tipo de t, aplicamos ya sea lambda o list.

Funciones Auxiliares:

Función para buscar una variable dentro de un ambiente. search-type:: x env $\rightarrow$ boolean

Función que valida tipo para cada operación. check-types:: pr arg $\rightarrow$ boolean | pr = operador | arg = argumentos

Funcion que agrega la tupla (x,t) al ambiente ; add-env:: x t env -¿list — x = variable, t = tipo, env = ambiente

Comentarios:

Consideramos que la practica viene bien documentada por lo que en este README prácticamente copiamos y pegamos lo comentarios que pusimos en la practica.

Una disculpa por la entrega tardía, tuvimos unos contratiempos de fuerza mayor y no pudimos reunirnos en tiempo.