Universidad de los Andes

ISIS-1106 Lenguajes y máquinas
TP 2
Fecha: October 13, 2022

Dentro de la teoría de tipos existe una categoría de tipos que se llaman tipos lineales (linear type systems). Este sistema de tipos tiene la particularidad que únicamente permite que las variables deban ser utilizadas una única vez; si se intenta acceder a una variable por segunda vez, o las variables no se utilizan, el programa debe lanzar un error.

La utilidad de este sistema de tipos esta en el manejo de memoria, asegurando propiedades de seguridad (safety) sobre la memoria.

Para este taller vamos a desarrollar un verificador de tipos lineales para un pequeño lenguaje de programación consistente de definición de variables, asignaciones y operaciones aritméticas.

Las variables son definidas por medio del comando `var(a)`. En nuestro lenguaje exigiremos que todas las variables estén definidas como una única letra minúscula.

La asignación de variable se define por medio del comando `assign(VAR, X)`, donde `VAR` es el nombre de una variable y `X` es el valor numérico dado a la variable.

Las operaciones aritméticas están definidas como:

- `plus(X,Y)`, donde `X` y `Y` pueden ser variables o números
- `mult(X,Y)`, donde `X` y `Y` pueden ser variables o números
- `subs(X,Y)`, donde `X` y `Y` pueden ser variables o números
- `div(X,Y)`, donde `X` y `Y` pueden ser variables o números

De esta forma, un programa escrito dentro del lenguaje se puede definir como a continuación.

```
1 var(a)
2 var(b)
3 var(c)
4 assign(c, plus(a,b))

6 var(d)
7 mult(4, assign(d, 2))
```

Task 1. Su tarea es definir un autómata que sea capaz de leer las instrucciones del programa, y verifique que el programa cumpla la propiedad que las variables sean utilizadas una única vez.

La entrada del autómata será un conjunto de caracteres conteniendo el token (en mayúscula) que representa cada una de las instrucciones, `V, A, P, M, S, D`, definiendo respectivamente los comandos descritos anteriormente. Los números serán representados

como su propio token (*e.g.,* 42). La variables también se representan como su propio token (*e.g.,* `a`). Por último, los parámetros entre las funciones se definen entre los tokens de paréntesis `(, )` y separan por una token de coma `,`.

la entrada del programa anterior se define como sigue:

```
1 VaVbVcA(c,P(a,b))VdM(4,A(d,2))
```