Universidad de Talca
Facultad de Ingeniería
Departamento de Ciencias de la Computación

Modelos de Computabilidad

Tarea 2 - Informe

Erik Regla
eregla09@alumnos.utalca.cl

10 de Diciembre del 2014

1. Introducción

Dada una calculadora infija proporcionada como base, diseñar una calculadora prefija que soporte suma, resta, multiplicación, división, potencia, negación, asignación de variables y soporte para numeros de doble prescición.

2. Análisis del problema

2.1. Desambiguación del Lenguaje

2.1.1. Descripción del problema

Las operaciones solicitadas piden que se ejecuten las operaciones en un formato similar al de *Racket*, cada operación rodeada de paréntesis, el primer miembro siendo el operador en caso de estar disponible. El problema se presenta en que dado que el operador se presenta en un extremo de la *expresion*, entonces no tenemos forma de saber que operación le precede a cada elemento. Para lo cual se implementó una lista inherente a cada tipo de operación, de modo tal que permita

```
operacion   :   NUM                       { $$ = $1;      }
            |   '+' lista_suma expresion  { $$ = $2 + $3; }
;

lista_suma  :   lista_suma  expresion     { $$ = $1 + $2; }
            |   expresion                 { $$ = $1;      }
            ;
```

Figura 1: Ejemplo de desambiguación para la operación *suma*.

alojar expresiones de forma recursiva y siempre aplique la misma operación a los miembros de su nivel (ver ejemplo en la Figura 1). Esta misma desambiguación fue realizada para las operaciones *resta*, *multiplicación* y *división*.

3. Implementación

3.1. Parser Léxico

Algoritmo 1 BFS

Precondición: W es un puntero a un caracter del lenguaje

Postcondición: NUM si se encuentra un número, un caracter si se encuentra alguna otra cosa, error en caso contrario.

1: consumir espacios en blanco de W
2: **si** W es un dígito o un punto **entonces**
3: $NUM \leftarrow$leer numero en W
4: **devolver** NUM
5: **si no**, **si** W es un caracter **entonces**
6: **devolver** W
7: **fin si**
8: **devolver** error

3.2. Variables

Las variables son almacenadas en un arreglo unidimensional de tipo double, Todas las variables si no han sido inicializadas tienen como valor por defecto 0. El tratamiento de la asignación se hace en la definición de *variable*, de modo de mantener la coherencia del resto de la sintáxis.

4. Notas

- Las fuentes de *Blumenkranz*, el programa que implementa los algoritmos anteriormente descritos, está alojado en `https://bitbucket.org/eregla/ecd2014-1/` al igual que la prueba y tarea anterior.
- El programa está licenciado bajo una licencia abierta MIT.