UNIVERSIDAD
DE SANTIAGO
DE CHILE

Departamento de Matemática y Ciencia de la Computación

Tarea 1

Generador de tablas de verdad de una proposición

Primer Semestre 2018

Lógica Computacional
Licenciatura en Ciencia de la Computación

Nicolás Honorato Droguett, Eduardo Baltra Rojas
nicolas.honorato@usach.cl, eduardo.baltra@usach.cl

1 Introducción

En este informe se da a conocer el desarrollo, análisis e implentación de un algoritmo que, dada una o varias proposiciones lógicas, pueda generar la tabla de verdad respectiva y poder dar una conclusión sobre ella.

2 Objetivo

El trabajo en cuestión tiene como objetivo crear e implementar un algoritmo que pueda ser capaz de transformar, dar formato y generar la tabla de verdad para una proposición lógica escrita en formato LaTeX.

3 Procedimiento

Dada una proposición escrita en LaTeX, esta se debe transformar a un formato más simple. Luego, con la proposición transformada, se le añaden los paréntesis respectivos para respetar la prioridad de los operadores y dejar las expresiones de cada paréntesis de la forma operando-operador-operando.

Finalmente, se genera la tabla de verdad y se da una conclusión respecto a ésta.

Si los valores de la tabla son todos 0, se concluye que la proposición es una contradicción. Por el contrario, si la tabla se compone de solo valores 1, se concluye que la proposición es una tautología. En otro caso, se concluye que la proposición es una contingencia.

3.1 Consideraciones

- Para la correcta ejecución del programa, los caracteres son case-sensitive, es decir, difieren estando en minúscula o mayúscula.
- En el caso de las proposiciones que tengan un número de átomos mayor a 6, la tabla no podrá imprimirse, por lo que el programa solo mostrará la conclusión generada respecto a ésta. Esto se debe a que la tabla es de tamaño muy grande para ser mostrada por pantalla.

3.2 Restricciones

- Se asume que la proposición viene escrita de una forma válida para LaTeXsin los signos $ al principio y al final.
- Los caracteres que representen a los átomos proposicionales deben ser de tamaño unitario y deben pertenecer al alfabeto inglés.
- El programa no permite expresiones escritas en LaTeXque contengas dos o ms negaciones consecutivas.

4 Estructuras de Datos Utilizadas

La estructura utilizada corresponde a un árbol en el cual se guarda la proposición, su tabla de valores de verdad y donde sus dos hijos son las sub-proposiciones que estan unidas al conectivo principal de la proposición.

```
struct Proposicion{
    char *proposicion;
    unsigned int *valores;
    struct Proposicion *izq,*der;
};
```

5 Algoritmo

El algoritmo se encarga de transformar la proposición escrita en LaTeXa una conveniente para el programa(se convierten los operadores escritos en LaTeXa operadores unitarios). Luego, la cadena transformada, se pasa desde formato infix a postfix para luego transformarla nuevamente a infix pero con los paréntesis correspondientes respetando la prioridad de los operadores, y de modo que en cada paréntesis solo haya una proposición de dos operandos y un operador. Ejemplo: $(p \wedge (q \vee r))$.

Finalmente, teniendo la cadena transformada, se descompone recursivamente en un árbol semántico en donde las hojas del árbol son los átomos proposicionales.

5.1 Análisis de Complejidad

La ejecución de cada instancia de la función que construye el árbol semántico tiene un coste de ejecución de O(n), donde n es la cantidad de combinaciones de la tabla de verdad.

6 Implementación

El algoritmo está implementado en lenguaje C, específicamente con el formato ANSI C.

6.1 Plataforma Computacional

El programa fue ejecutado en un computador con las siguientes características:

- **Procesador:** Procesador Intel(R) Core(TM) i5-6500 CPU 3200MHz - 3600MHz.
- **Memoria RAM:** 8192000 kiB
- **Sistema Operativo:** Linux - Ubuntu 16.04.2 LTS

7 Resultados

La ejecución del programa, muestra que el algoritmo es capaz de procesar una proposición con un máximo de 26 átomos diferentes. Esto, visto de otra forma, significa que el algoritmo es capaz de procesar una tabla de verdad de tamaño 2^{26} como máximo.

En la siguiente figura se muestra el tiempo de ejecución de una proposición con 26 sentencias diferentes.

```
#1--------------------------
Proposicion: ((((((((((((((((((((a*b)*c)*d)*e)*f)*g)*h)*i)*j)*k)*l)*m)*n)*o)*p)*q)*r)*s)*((((((t*u)*v)*w)*x)*y)*z))
Sentencias: zyxwvutsrqponmlkjihgfedcba
Combinaciones totales: 67108864
Es una contingencia!


real    0m21.628s
user    0m7.172s
sys     0m12.969s
```

Figure 1: Ejemplo proposición 26 sentencias diferentes.

8 Conclusión

Cómo síntesis, se puede concluir que el trabajo ha aportado a la habilidad de análisis de problemas de los integrantes y al conocimiento de transformación de expresiones matemáticas a formatos infix y postfix. Además, el desarrollo de éste algoritmo deja en claro que cada proposición lógica se puede tratar o expresar cómo un árbol en dónde cada nodo es una sub-proposición de la proposición principal.