Nombre: Dana Carolina Ramírez Velázquez

Código: 220286547

Materia: Traductores de Lenguaje 2

Actividad: Tarea 13

Fecha: 26/10 /22

Árbol de Sintaxis Abstracta (ASA) Como nos indica Aho el al (1986), el uso de árboles sintácticos como representación intermedia permite que la traducción se separe del análisis sintáctico. Es importante indicar que los tipos de árboles que nos interesan son los árboles de sintaxis abstracta (ASA) que son árboles de análisis sintácticos a los que se les han eliminado los símbolos superfluos y es muy útil para representar las construcciones del lenguaje. Veamos los siguientes ejemplos: Como podemos ver, en un ASA nos interesan las operaciones y los operadores y es por eso que eliminamos los símbolos superfluos (terminales de la gramática), generándose un árbol condensado.

Tipos de expresiones

En la mayoría de los lenguajes encontramos las siguientes (en este caso utilizamos C como

ejemplo):

Declaraciones: int a.

Constantes: #define PI 3,141592.

Asignaciones: a := b.

Bloques.

Operaciones unarias: a := op b (donde op, por ejemplo, puede ser +, -, ! -negación-).

Operaciones binarías: a := b op c (dentro de las operaciones binarias, además de las aritméticas, entran las operaciones lógicas y las relacionales).

Bifurcaciones: goto A1.

Bifurcaciones condicionales: if (a == 5) goto A1 (que surgen cuando se desglosan los bucles).

Bucles. Por ejemplo, while y for.

while (expr_condicional) {Sentencias }.

for (a:=0; a < b, a++) {Sentencias }.

Control de flujo: if condición then S1, if condición then S1 else S2, switch.

Operaciones con arrays: asignación y carga (a := b[i] o b[i] = a).

Operaciones con punteros: a := , a := *b, *a:=b.

Llamadas a funciones: x:= f(arg1, arg2,..argN).

En este tema hemos comprendido porqué es necesario realizar la generación de código intermedio y qué ventajas tiene, como el facilitar la redestinación o poder aplicar la optimización de forma independiente de la máquina. También se ha visto que hay distintos tipos de máquinas para las que generar código, en función de las direcciones que podemos utilizar a la vez en una operación (0, 2 o 3 direcciones). Aunque hay varias modalidades de generación de código intermedio, las más ampliamente usadas son los ASA (cuando no se va a realizar la fase de optimización) o su variante mejorada los grafos dirigidos acíclicos (GDA) y el código de tres direcciones. Se ha podido ver con varios ejemplos, primero con operaciones aritméticas y posteriormente con uno más completo, como se construye el ASA y que tipos de nodos podemos encontrarnos en un lenguaje genérico. Asimismo se han comentado otras implementaciones que pueden resultar útiles a la hora de diseñar y construir un compilador.