República Bolivariana de Venezuela
Ministerio del Poder Popular para la Educación Superior
Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
Traductores e Interpretadores
Profesor: MSc Ricardo Monascal

INFORME

ETAPA 1

Autores:
Br. Jean Alexander 12-10848
Br. Diego Pedroza 12-11281

Caracas, mayo de 2016

Índice de contenido

IMPLEMENTACIÓN

La segunda etapa del proyecto de Traductores, es realizar un analizador sintáctico para el lenguaje Neo, el cual tiene palabras reservadas, identificadores, comentarios, caracteres, números y operadores. La finalidad del analizador es revisar si existe algún error sintáctico, mostrando los errores léxicos.

Para esta etapa, se usó Haskell como lenguaje de programación como en la entrega anterior, con la herramienta Alex, un analizador lexicográfico. Además, la herramienta Happy, un parser de la plataforma de Haskell. Primero se realizó la gramática correspondiente al lenguaje Neo,y luego siguiendo ejemplos y recomendaciones se desarrolló un parser para Neo.

De esta forma, se procederá a responder las preguntas teórico-prácticas

Revisión Teórico-prácticas

Se utilizó Dia de Gnome para realizar los árboles de derivación. Se procederá a resolver las preguntas.

1 PREGUNTA 1

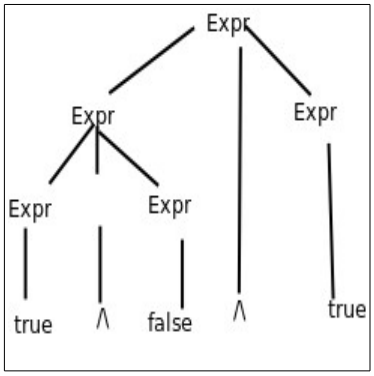
Sea G1 la gramática ({Expr}, {+,true,false}, P1, Expr) Y P1:

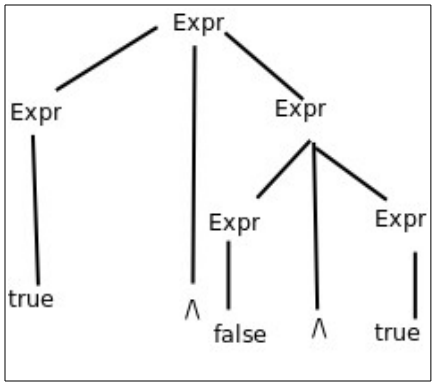
Expr -> Expr ∧ Expresiones

| true

| false

a) Muestre que la frase true ∧ false ∧ true de G1 es ambigua.





Como existe al menos dos árboles de derivación distintos leftmost para una frase, entonces la gramática es ambigua.

b) Dé una gramática no ambigua Izq(G1) que genere el mismo lenguaje que G1 y que a asocie las expresiones lógicas generadas hacia la izquierda. Dé también una Der (G1) con las mismas características pero que asocie hacia la derecha.

Izq(G1)

Expr-> Expr /\ A

| A

A-> true | false

Der(G1)

Expr->A /\ Expr

| A

A-> true | false

c) ¿Importa si se asocian las expresiones hacia la izquierda o hacia la derecha? Considere el caso de un operador de implicación ("⇒").

Sí, sí importan. Por ejemplo ((a => b) => c) no es equivalente a (a => (b=>c))

Si a=false, b=false, c=false

Entonces

La primera dará como resultado , false

y la segunda, true

2 PREGUNTA 2

En Neo la secuenciación, o composición secuencial, es un operador binario infijo sobre instrucciones. Este operador es en realidad virtual, ya que no es representado por ninguna secuencia de símbolos en particular. Sin embargo, a efectos de esta pregunta, denotaremos la secuenciación por el símbolo " ; ". Suponga que para el manejo de este operador se decide utilizar la gramática G2 definida como ({ Instr } , { ; , IS } , P2 , Instr) , con P2

Instr - > Instr ; Instr

| IS

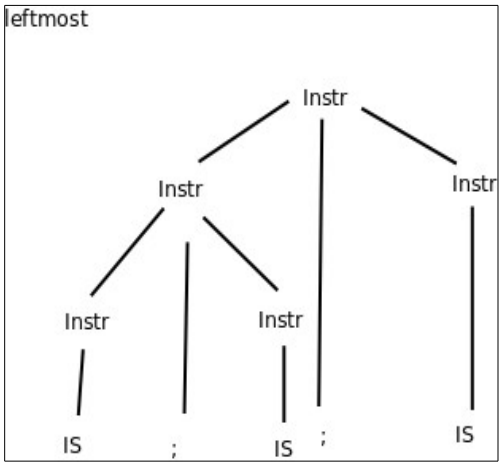
a) ¿Presenta G2 los mismos problemas de ambigüedad que G1 ? ¿Cuáles son las únicas frases no ambiguas de G2 ?

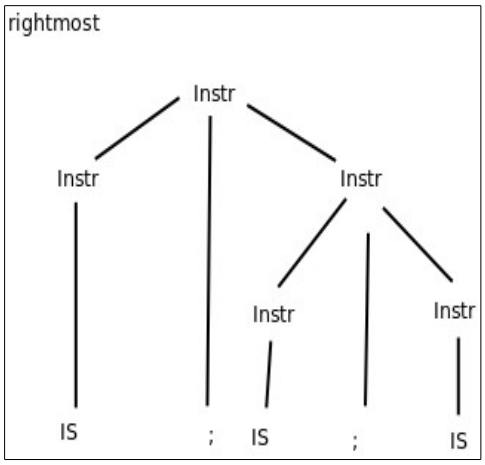
Sí presentan los mismos problemas. Las únicas dos frases son: IS y IS;IS

b) ¿Importa si la ambigüedad se resuelve con asociación hacia la izquierda o hacia la derecha?

No, no afectaría la ejecución del programa, porque el orden del mismo se seguiría manteniendo

c)Dé una derivación “más a la izquierda” (leftmost) y una derivación “más a la derecha” (rightmost) de G2 para la frase IS ; IS ; IS .





3 PREGUNTA 3

Consideremos ahora los constructores de instrucciones condicionales de Neo, pero ignorando momentáneamente el uso que la sintaxis de éstos hace de los “end” y manteniendo el operador “ ; ” propuesto en la pregunta anterior.

Sea G3 la gramática ({ Instr } , { ; , IF , : , OTHERWISE , Bool , IS } , P3 , Instr) , con a P3 compuesto por:

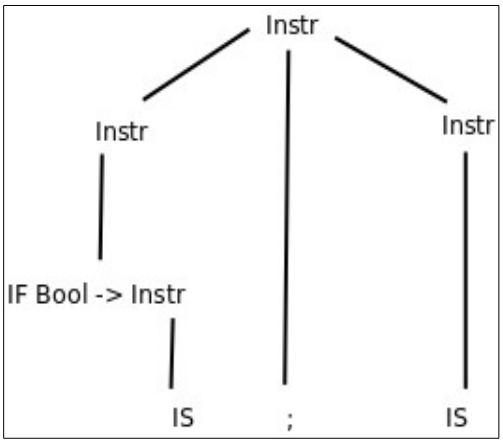
Instr - >Instr ; Instr

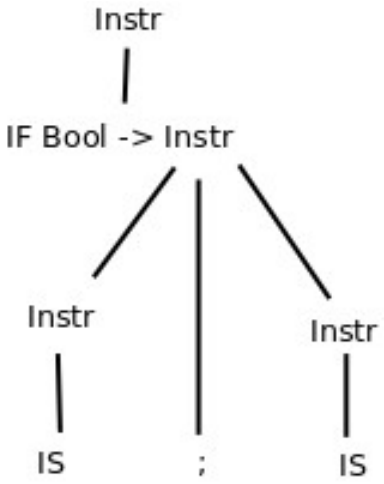
| IF Bool − > Instr

| IF Bool − > Instr OTHERWISE − > Instr

| IS

a) Note que G3 mantiene el mismo problema de G2 , i.e. frases con más de una ocurrencia de " ; " son ambiguas. Mas, G3 tiene otros problemas pues frases con ninguna ocurrencia de " ; " también pueden ser ambiguas. Sea f la frase "IF Bool − > IS ; IS ". Muestre que f es una frase ambigua de G3 .

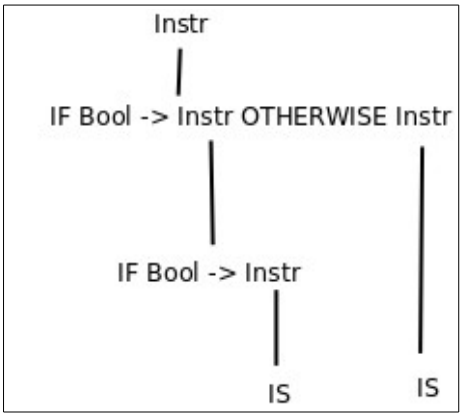


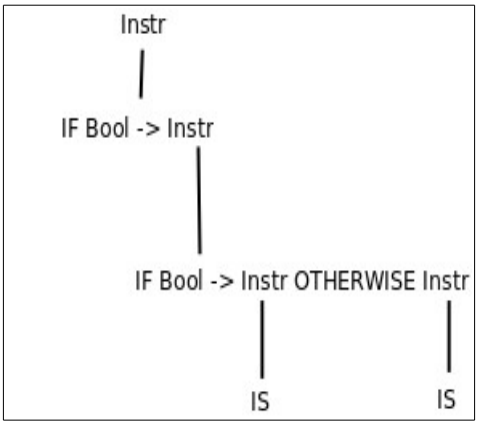


Como existe al menos dos árboles de derivación distintos leftmost para una frase, entonces la gramática es ambigua.

b) Dé una frase g de G3 sin ocurrencias de " ; " que sea ambigua, y muestre que lo es.

g: IF Bool -> IF Bool -> IS OTHERWISE IS





Como existe al menos dos árboles de derivación distintos leftmost para una frase, entonces la gramática es ambigua.

c) En lenguajes como Java, C y C++ se hace uso de los símbolos "{" y "}" ("begin" y "end " en Pascal) para diferenciar las dos posibles interpretaciones de la frase f . Lo mismo ocurre con la frase g . ¿Cómo se escribir las dos interpretaciones de f y las dos interpretaciones de g usando las llaves como separadores?

f: IF Bool -> {IS}; IS
IF Bool -> {IS;IS}

g: IF Bool -> {IF Bool -> {IS} OTHERWISE-> {IS}}
IF Bool -> {IF Bool -> {IS} OTHERWISE -> {IS}

d) En Neo, que utiliza los terminadores "end" en la sintaxis de sus condicionales, ¿cómo se escribir las dos interpretaciones de f y las dos interpretaciones de g ?

f: IF Bool -> IS end; IS
IF Bool -> IS; IS end

g: IF Bool -> IF Bool -> IS OTHERWISE -> IS end end
IF Bool -> IF Bool -> IS end OTHERWISE -> IS end