INSTITUTO TECNOLOGICO DE IZTAPALAPA

MATERIA: Lenguajes y Autómatas II

PROFESOR: Abiel Tomás Parra Hernández

ALUMNO:

*QUINTERO BOLIO ERIK EDUARDO 171080151

*COFRADIA RODRIGEZ RODRIGO B. 161080399

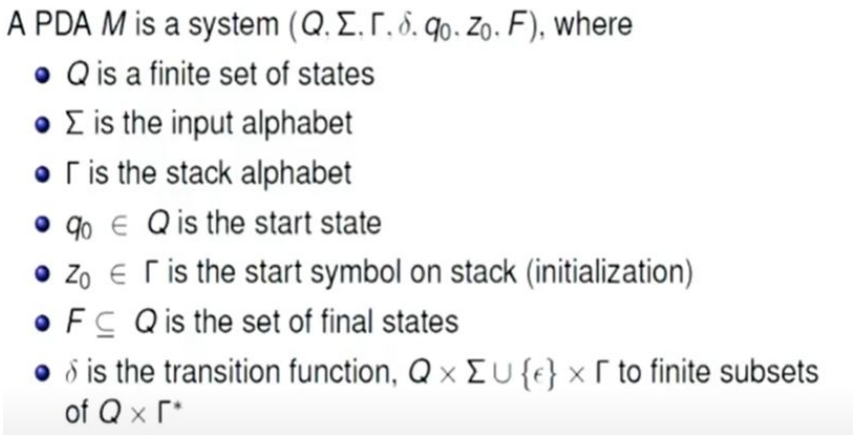
CARRERA: SISTEMAS COMPUTACIONALES

Cofradía Rodriguez Rodrigo B. 161080399 (videos)

En la actividad de esta semana se hicieron las anotaciones de los videos "Syntax Analysis: Context-free Grammars, Pushdown Automata and Parsing" y estos nos explican de la gramática libre de contextos.

Pushdown: Es un conjunto finito de estados Q, tiene un alfabeto de entrada tiene una coma de alfabeto de pila hay un estado de pila de inicio z y un conjunto de estados finales F.

En el conjunto Q de un estado q en un símbolo de entrada y los símbolos de pila agregan en la parte superior de stack pueden moverse al estado P1 o P2 o es p1,p3 y en ese proceso quita la parte superior del símbolo de la pila y lo reemplaza con gama1, gama2 cualquiera de estos dependiendo de en que estado se mueva este es el símbolo de entrada por uno.



Este tiene una forma gramatical que se utiliza para los idiomas libres de contexto, este pesa bastante entonces requiere 1 cuadro de tiempo para que así estén dos algoritmos muy conocidos.

Algoritmo de Earley

El es un algoritmo no determinista de análisis sintáctico para las gramáticas libres de contexto descrito originalmente por el informático estadounidense Jay Earley en 1970. Se ordena, a los lados de los algoritmos CYK y GLR, entre los algoritmos que usan la noción de reparto (de cálculos y de estructuras) y que construyen todos los análisis posibles de una frase. Es uno de los algoritmos no deterministas que usan ideas de la programación dinámica.

Algoritmo de Cocke-Younger-Kasami (CYK)

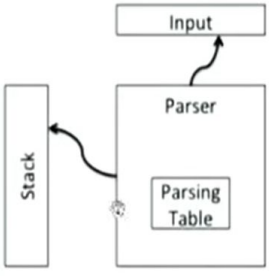
Este algoritmo es el que determina si una cadena puede ser generada por una gramática libre de contexto y, si es posible, cómo puede ser generada. Este proceso es conocido como análisis sintáctico de la cadena. El algoritmo es un ejemplo de programación dinámica.

Los subconjuntos de lenguajes libres de contexto suelen requerir 0(n) tiempo

* Análisis predictivo utilizando gramática LL(1)(análisis de método de arriba hacia abajo).

* Reduzca el análisis sintáctico utilizando la gramática LA(1) (método de análisis sintáctico de abajo hacia arriba).

El análisis de arriba hacia abajo usando pases predictivos eleva la derivación más a la izquierda de la cadena mientras se construyen las tres primeras, así que comenzamos desde el símbolo de inicio y predecimos la producción que se usa en la derivación, necesitamos más información y eso se conoce como tabla de análisis, que se construye sin conexión y se detiene, por lo que vamos a estudiar cómo la partícula es efectiva en la tabla de análisis que se construye fuera de línea y se almacena. Entonces vamos a estudiar cómo se construye la tabla de análisis sintáctico. Las próximas entrevistas de producción en la derivación se determinan observando el siguiente símbolo y también la tabla del analizador así que esta combinación te dice exactamente qué producción en particular se utilizará y el símbolo X que vemos se llama cuando miramos hacia adelante así que al imponer restricciones a la gramática, nos aseguramos de que no haya luego una reducción en cualquier tipo de tabla de personas, por lo que veremos que si hay más de una producción en cualquier trama de la tabla de cierre, entonces no podemos decidir qué producción usar, en el momento de la construcción de la tabla de análisis, hay dos producciones elegibles para colocarse en el mismo tipo de tabla de personas.




```
Initial configuration: Stack = S, Input = w$,
where, S = start symbol, $ = end of file marker
repeat {
   let X be the top stack symbol;
   let a be the next input symbol /*may be $*/;
   if X is a terminal symbol or $ then
       if X == a then {
           pop X from Stack;
           remove a from input;
       } else ERROR();
   else  /* X is a non-terminal symbol */
       if M[X,a] == X → Y1Y2... Yk then {
           pop X from Stack;
           push Yk, Yk-1, ..., Y1 onto Stack;
                (Y1 on top)
       }
} until Stack has emptied;
```

Condiciones comprobables para ll(1)

Puede ser un símbolo de terminal o el final de la tabla de símbolos de Entonces lo que tenemos que hacer es entender es que en primer lugar, por qué está determinado sólo por cómo sigue

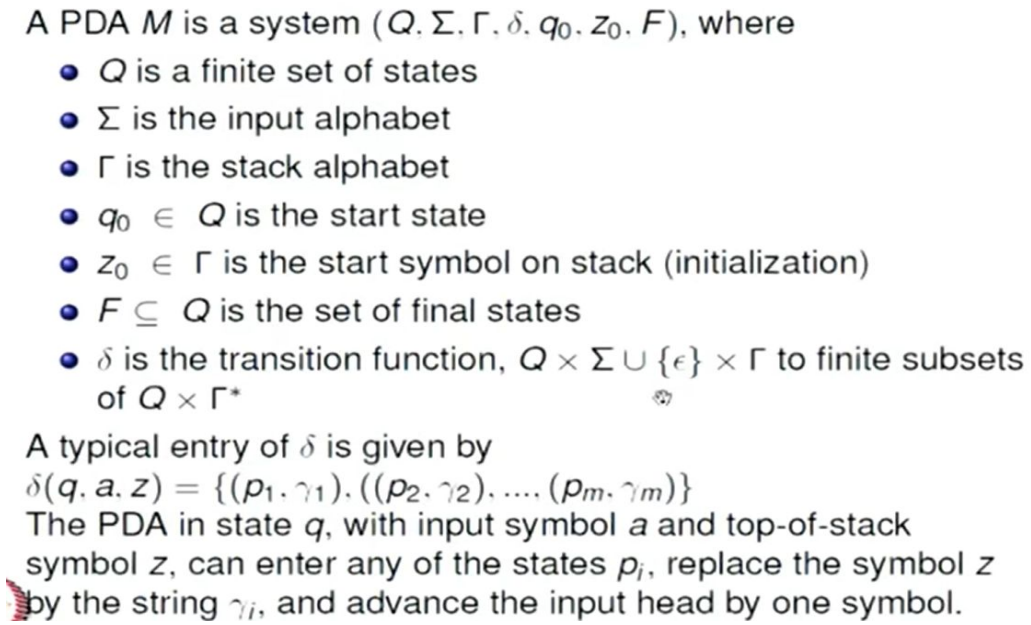
Se sabe muy simplemente con **First Step** como primero que no es más que el primero que son todas las cadenas de S Prime en realidad derivadas de S Y que si derivan todas las cuerdas que son, ya sabes derivadas por S comienzan con A ó C. Así que primero de s sería una C

Literalmente por el primero de ll y Tienes todas las cadenas derivables de un sabes que comienzan con la pequeña B. Y porque hay una S no termina aquí todas las cadenas que son derivables por S también están incluidas en la primera forma. y para B, entonces eso nos da a saber, el primero de a tiene a b c en este momento en lo que respecta a B, los símbolos son y el primero de los suyos están incluidos en el primero , así es como se completa todo esto.

Quintero Bolio Erik Eduardo 171080151 (videos)

Lo que pude entender de los videos es que nos explica de el Pushdown, este es un autómata tiene un conjunto finito de estados Q, este tendrá un alfabeto que el cual inicia con una z y termina con una f, a estos se les llama conjuntos de estados ya sea final o inicial.

Un conjunto Q de un estado “q” minúscula es un estado de entrada y los símbolos que le siguen consecuentemente llamados stack son llamados P1, o P2 y estos pueden variar como P1, P3 dependiendo del autómata.



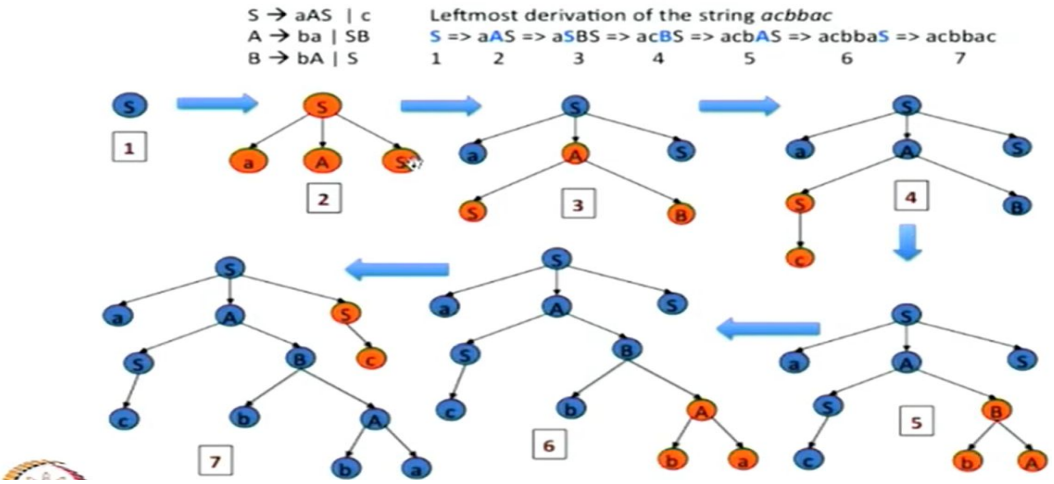
Pushdown parte del analizar el proceso de construcción de un árbol de análisis para esto usamos un autómata pushdown con algunas acciones adicionales para construir un árbol de análisis así que básicamente la máquina de análisis no es más que un papel de autómata de empuje determinista.

También en los videos nos habla de la gramática libre de contexto que está se divide en dos algoritmos: algoritmo de Earley y algoritmo de Cocke-Younger-Kasami (CYK).

El primero el algoritmo EARLEY es un algoritmo no determinista de análisis y el otro Cocke-Younger es un algoritmo determinista y este tiene dos subconjuntos de lenguaje que son:

Los subconjuntos de lenguajes libres de contexto suelen requerir 0(n) tiempo

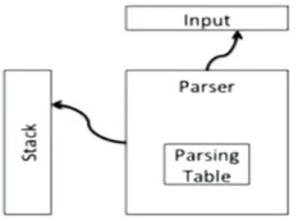
- Análisis predictivo utilizando gramáticas LL(1)(análisis de método de arriba hacia abajo)
- Reduzca el análisis sintáctico utilizando la gramática LA(1) (método de análisis sintáctico de abajo hacia arriba)



luego nos habla sobre la tabla de análisis que se construye fuera de línea y se almacena y en esta vamos a estudiar cómo se construye la tabla de análisis sintáctico. Las próximas entrevistas de producción en la derivación se determinan observando el siguiente símbolo y también la tabla del analizador así que esta combinación te dice exactamente qué producción en particular se utilizará y el símbolo X que vemos se llama cuando miramos hacia adelante así que al imponer restricciones a la gramática, nos aseguramos de que no haya luego una reducción en cualquier tipo de tabla de personas, por lo que veremos que si hay más de una producción en cualquier trama de la tabla de cierre, entonces no podemos decidir qué producción usar a Entonces, en el momento de la construcción de la tabla de análisis, hay dos Producciones elegibles para colocarse en el mismo tipo de tabla de personas.

Entonces se declara que la gramática no es apta para el derecho de así que sigamos adelante y veamos cómo funciona exactamente el algoritmo

Así que repite que X sea el símbolo de la pila superior para que en algún momento para empezar sea sí, y luego pueda convertirse en algún otro símbolo, deja que a sea el siguiente símbolo de podría ser el final del archivo en así que si la parte superior de la pila X es un símbolo de terminal o el símbolo de fin de archivo dólar y es igual al símbolo de entrada a, entonces obviamente es hora de parchear el conjunto que el repetidor puso en el autómata




```
Initial configuration: Stack = S, Input = w$,
where, S = start symbol, $ = end of file marker
repeat {
   let X be the top stack symbol;
   let a be the next input symbol /*may be $*/;
   if X is a terminal symbol or $ then
        if X == a then {
            pop X from Stack;
            remove a from input;
        } else ERROR();
   else  /* X is a non-terminal symbol */
        if M[X,a] == X → Y1Y2... Yk then {
              pop X from Stack;
              push Yk, Yk-1, ..., Y1 onto Stack;
                    (Y1 on top)
        }
} until Stack has emptied;
```

Luego puedes convertirte en otros símbolos, deja el siguiente símbolo de entrada de un Podría ser un dólar de Entonces la parte superior de la pila X es un símbolo de terminal o el símbolo de fin de archivo dólar y es igual al símbolo de entrada a, entonces obviamente diseñado para tocar el pero si nos pusieron en autómata también hace esto así que siempre que el símbolo de entrada del alfabeto de entrada coincide con los estallidos simbólicos de la pila, también elimina el punto de la esa es la entrada que se el punto se mueve un paso adelante si esto no es así, el reflejo no es igual a, eso significa que la pila y la entrada no lo Entonces, se debe informar un error. La siguiente posibilidad es que la parte superior de Stack sea un símbolo no terminal, aquí hay un símbolo terminal ahora, es un T no terminal.

Apuntes en equipo

Pushdown: Es un conjunto finito de estados Q, tiene un alfabeto de entrada tiene una coma de alfabeto de pila hay un estado de pila de inicio z y un conjunto de estados finales F.

En el conjunto Q de un estado q en un símbolo de entrada y los símbolos de pila agregan en la parte superior de stack pueden moverse al estado P1 o P2 o es p1,p3 y en ese proceso quita la parte superior del símbolo de la pila y lo reemplaza con gama1, gama2 cualquiera de estos dependiendo de en qué estado se mueva este es el símbolo de entrada por uno.

La gramática libre de contexto que está se divide en dos algoritmos: algoritmo de Earley y algoritmo de Cocke-Younger-Kasami (CYK).

El primero el algoritmo EARLEY es un algoritmo no determinista de análisis y el otro Cocke-Younger es un algoritmo determinista y este tiene dos subconjuntos de lenguaje que son:

Los subconjuntos de lenguajes libres de contexto suelen requerir 0(n) tiempo

- Análisis predictivo utilizando gramáticas LL(1)(análisis de método de arriba hacia abajo)
- Reduzca el análisis sintáctico utilizando la gramática LA(1) (método de análisis sintáctico de abajo hacia arriba)

La tabla de análisis que se construye fuera de línea y se almacena y en esta vamos a estudiar cómo se construye la tabla de análisis sintáctico. Las próximas entrevistas de producción en la derivación se determinan observando el siguiente símbolo y también la tabla del analizador así que esta combinación te dice exactamente qué producción en particular se utilizará y el símbolo X que vemos se llama cuando miramos hacia adelante así que al imponer restricciones a la gramática, nos aseguramos de que no haya luego una reducción en cualquier tipo de tabla de personas, por lo que veremos que si hay más de una producción en cualquier trama de la tabla de cierre, entonces no podemos decidir qué producción usar a Entonces, en el momento de la construcción de la tabla de análisis, hay dos Producciones elegibles para colocarse en el mismo tipo de tabla de personas.

Entonces se declara que la gramática no es apta para el derecho de así que sigamos adelante y veamos cómo funciona exactamente el algoritmo

Así que repite que X sea el símbolo de la pila superior para que en algún momento para empezar sea sí, y luego pueda convertirse en algún otro símbolo, deja que a sea el siguiente símbolo de podría ser el final del archivo en así que si la parte superior de la pila X es un símbolo de terminal o el símbolo de fin de archivo dólar y es igual al símbolo de entrada a, entonces obviamente es hora de parchear el conjunto que el repetidor puso en el autómata