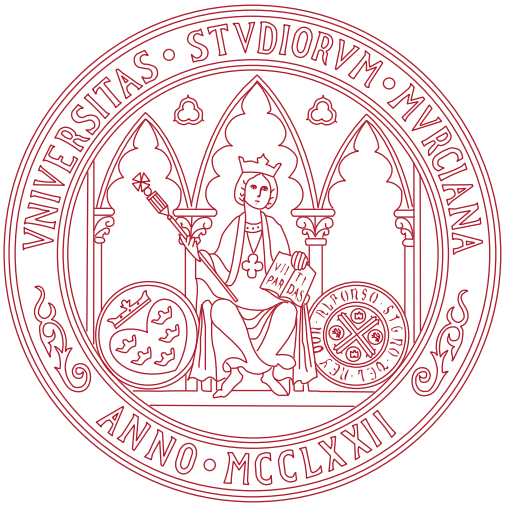


UNIVERSIDAD DE MURCIA

GRADO EN INGENIERÍA INFORMÁTICA

Conflictos de análisis LR debidos a la precedencia y asociatividad de operadores

Autores:
Eduardo MARTÍNEZ GRACIÁ
María Antonia CÁRDENAS VIEDMA
María Antonia MARTÍNEZ CARRERAS

26 de abril de 2021

Este documento pretende aclarar una de las dudas más frecuentes que los alumnos de la asignatura de Compiladores tienen en relación con el análisis ascendente LR: *los conflictos* provocados por la precedencia y asociatividad de los operadores en las gramáticas.

1. La gramática de siempre, con operadores aritméticos

Muchos conflictos tiene su origen en el uso de gramáticas ambiguas. Como es normal, conviene partir de una gramática de ejemplo para poder concretar el problema. La siguiente gramática permite expresar operaciones aritméticas de variables (identificadores) con sumas y multiplicaciones:

$$E \to E + E$$
$$E \to E * E$$
$$E \to \mathbf{id}$$

Esta gramática provoca conflictos independientemente del método que use, SLR, LALR o LR(1).

2. Aplicación inocente del método SLR

Imaginemos que no sospechamos, a priori, los problemas que puede generar la gramática. Vamos a intentar hacer un analizador lo más sencillo posible, es decir, un analizador SLR. Empezamos a construir el conjunto de ítems:

$$I_0 = \text{CLAUSURA}(\{E' \to \cdot E\}) = \{E' \to \cdot E \\ E \to \cdot E + E \\ E \to \cdot E * E \\ E \to \cdot \mathbf{id}\ \}$$

$$\text{GOTO}(I_0, E) = I_1 = \{\ E' \to E\cdot \\ E \to E \cdot + E \\ E \to E \cdot * E\ \}$$

$$\text{GOTO}(I_0, id) = I_2 = \{\ E \to \mathbf{id}\cdot\ \}$$

$$\text{GOTO}(I_1, +) = I_3 = \{\ E \to E + \cdot E \\ E \to \cdot E + E \\ E \to \cdot E * E \\ E \to \cdot \mathbf{id}\ \}$$

$$\text{GOTO}(I_1, *) = I_4 = \{\ E \to E * \cdot E \\ E \to \cdot E + E \\ E \to \cdot E * E \\ E \to \cdot \mathbf{id}\ \}$$

$$\text{GOTO}(I_3, E) = I_5 = \{\ E \to E + E\cdot \\ E \to E \cdot + E \\ E \to E \cdot * E\ \}$$

$$\text{GOTO}(I_3, \mathbf{id}) = I_2$$

$$\text{GOTO}(I_4, E) = I_6 = \{\ E \to E * E\cdot \\ E \to E \cdot + E \\ E \to E \cdot * E\ \}$$

$$\text{GOTO}(I_4, \mathbf{id}) = I_2$$

$$\text{GOTO}(I_5, +) = I_3$$

$$\text{GOTO}(I_5, *) = I_4$$

$$\text{GOTO}(I_6, +) = I_3$$

$$\text{GOTO}(I_6, *) = I_4$$

Hasta aquí, aparentemente, todo bien. Partiendo de la colección de ítems obtenida, calculamos la tabla de análisis SLR de la gramática. Necesitamos los conjuntos PRIMERO y SIGUIENTE, cosa fácil:

$$\text{PRIMERO}(E) = \{id\}$$
$$\text{SIGUIENTE}(E) = \{\$, +, *\}$$

La tabla de análisis que se obtiene es:

ESTADO	ACCIÓN				IR-A
	id	$+$	$*$	\$	E
0	d2				1
1		d3	d4	acc	
2		r3	r3	r3	
3	d2				5
4	d2				6
5		d3/r1	d4/r1	r1	
6		d3/r2	d4/r2	r2	

Como se puede observar, la tabla tiene conflictos, luego no es SLR. Se puede intentar aplicar el método LR(1), más potente, pero también produce estos conflictos. Llegados a una situación así, es lógico pensar que la gramática puede tener algún problema de diseño que conlleva los conflictos. En este caso es la ambigüedad de las precedencias y asociatividades de los operadores. Vamos a pensar en el modo de solucionar los conflictos de alguna manera lógica. Para ello, simularemos el análisis de distintas cadenas de entrada, y comprobaremos cuál es la consecuencia de optar por la acción de desplazar o la acción de reducir en cada una de las casillas en las que hay conflicto.

3. Conflicto del estado 5 con el operador +

Vamos a concentrarnos en la casilla $[5, +]$ de la tabla de análisis. Para poder concentrarnos en este sitio, y conseguir que no nos estorben los otros conflictos que hay en la tabla, tenemos que pensar en una cadena de entrada que no necesite a las otras casillas de la tabla que son problemáticas. En el estado 5 tenemos los ítems:

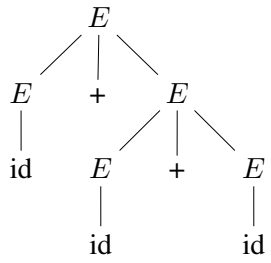
$$I_5 = \{\ E \to E + E\cdot\ ,\ E \to E \cdot + E\ ,\ E \to E \cdot * E\ \}$$

La casilla en la que tenemos puesta nuestra atención, $[5, +]$, corresponde al análisis de un token $+$ en la entrada, es decir, estando en el estado 5, el analizador se encuentra un token $+$. Recordemos que el conjunto de ítems de un estado, el 5 en este caso, contiene ítems con los que se deducen las acciones de las distintas casillas de dicho estado. Esto significa que no todos los ítems del estado 5 nos interesan para estudiar la casilla $[5, +]$. De hecho, podemos prescindir del ítem $E \to E \cdot * E$ del estado 5, puesto que únicamente refleja la situación en la que se encuentra un $*$ en la entrada, es decir, aporta información para la casilla $[5, *]$, y eso es algo que, por ahora, no nos interesa.

Vamos a suponer que tenemos que analizar algo que nos obliga a optar por la acción de desplazar o reducir en la casilla $[5, +]$. ¿Qué cadena de entrada nos lleva a la casilla $[5, +]$? Los dos primeros ítems del estado 5 nos dan la pista. El primer ítem es $E \to E + E\cdot$, y representa la situación en la que se han visto dos expresiones sumadas y están a punto de ser reducirlas. Se debe recordar que, en el método SLR, la reducción se aplica cuando en la entrada se encuentra cualquiera de los símbolos que pertenecen a SIGUIENTE(E), conjunto en el que está el $+$. El segundo ítem es $E \to E \cdot + E$, y representa la situación en la que se ha visto una expresión y ahora se encuentra un operador $+$ en la entrada. Si combinamos las dos situaciones en una cadena de entrada completa, parece que algo del tipo $id + id + id$ nos puede valer. Cuando el analizador haya visto los dos primeros id sumados, y se encuentre con el segundo operador $+$ en la entrada, se llegará a la situación en la que los dos ítems mencionados tienen que competir. Vamos a ver qué pasa si optamos, en ese momento, por desplazar o por reducir.

Empezamos simulando la situación en la que desplazamos, es decir, optamos por la acción d3 al llegar a la casilla $[5, +]$. A la izquierda se encuentra la simulación, y a la derecha el árbol de análisis que se obtiene:

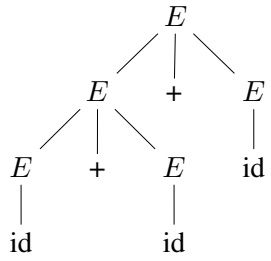
PILA	ENTRADA	ACCIÓN
0	$id\ +\ id\ +\ id\ \$$	d2
$0\ id\ 2$	$+\ id\ +\ id\ \$$	r3 $E \to id$
$0\ E\ 1$	$+\ id\ +\ id\ \$$	d3
$0\ E\ 1\ +\ 3$	$id\ +\ id\ \$$	d2
$0\ E\ 1\ +\ 3\ id\ 2$	$+\ id\ \$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5$	$+\ id\ \$$	d3 ¡Optamos por desplazar!
$0\ E\ 1\ +\ 3\ E\ 5\ +\ 3$	$id\ \$$	d2
$0\ E\ 1\ +\ 3\ E\ 5\ +\ 3\ id\ 2$	$\$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5\ +\ 3\ E\ 5$	$\$$	r1 $E \to E + E$
$0\ E\ 1\ +\ 3\ E\ 5$	$\$$	r1 $E \to E + E$
$0\ E\ 1$	$\$$	acc



Un árbol sintáctico asocia un significado a la cadena de entrada, es decir, a partir de la configuración que el árbol establece se va a generar un código en el proceso de traducción. La rama de la derecha nos indica que primero se va a generar el código para sumar los dos últimos identificadores. Ese código luego se concatenará con el código que suma el resultado de ese cálculo con el primer identificador. Es decir, la cadena de entrada $id + id + id$ se ha analizado como si fuese $id + (id + id)$. A esa forma de ordenar las operaciones de suma cuando tienes varias seguidas se le denomina *asociatividad por la derecha*. Por tanto, la consecuencia de elegir la acción de desplazamiento en la casilla $[5, +]$ es imponer una asociatividad por la derecha en la suma.

En general, salvo que se pida lo contrario, será más normal buscar una *asociatividad por la izquierda*. Parece lógico que, si se opta por la acción de reducción en la casilla $[5, +]$, se consiga esta asociatividad por la izquierda en la suma. Vamos a comprobarlo:

PILA	ENTRADA	ACCIÓN
0	$id\ +\ id\ +\ id\ \$$	d2
$0\ id\ 2$	$+\ id\ +\ id\ \$$	r3 $E \to id$
$0\ E\ 1$	$+\ id\ +\ id\ \$$	d3
$0\ E\ 1\ +\ 3$	$id\ +\ id\ \$$	d2
$0\ E\ 1\ +\ 3\ id\ 2$	$+\ id\ \$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5$	$+\ id\ \$$	r1 $E \to E + E$ ¡Optamos por reducir!
$0\ E\ 1$	$+\ id\ \$$	d3
$0\ E\ 1\ +\ 3$	$id\ \$$	d2
$0\ E\ 1\ +\ 3\ id\ 2$	$\$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5$	$\$$	r1 $E \to E + E$
$0\ E\ 1$	$\$$	acc



Así es, hay que elegir la acción **r1** en la casilla $[5, +]$ para tener la asociatividad por la izquierda en la suma. Es interesante observar que la pila del analizador crece menos si la asociatividad es por la izquierda. Esa es una buena razón para usar ésta y no la otra. Ya tenemos un conflicto resuelto.

4. Conflicto del estado 6 con el operador *

¿Se puede extrapolar la solución del conflicto de la casilla $[5, +]$ a alguno de los otros conflictos. La respuesta es sí. La casilla $[6, *]$ corresponde a estar en el estado 6 y ver un $*$ en la entrada. Observando los ítems del estado 6:

$$I_6 = \{\ E \to E * E\cdot\ ,\ E \to E \cdot + E\ , E \to E \cdot * E\ \}$$

llegamos a la conclusión de que son exactamente iguales a los del estado 5, pero cambiando los papeles de $+$ y $*$. Los ítems para resolver el problema de la casilla $[6, *]$ son los que implican al operador $*$, es decir, nos podemos despreocupar, por ahora, del segundo ítem. Para centrarse en el problema de la casilla $[6, *]$ valdría simular la cadena $id * id * id$, es decir, igual que en la sección anterior, pero cambiando $+$ por $*$.

La acción que habría que seleccionar para conseguir asociatividad por la izquierda en el operador de multiplicación es la reducción **r2**, igual que se hizo en el apartado anterior. Si hay alguna duda, siempre se puede hacer la simulación eligiendo una u otra opción.

5. Conflicto del estado 5 con el operador *

Ahora llegamos a otro tipo de conflicto, que se refleja en la casilla $[5, *]$ de la tabla de análisis. A estas alturas, si has entendido lo anterior, deberías estar capacitado para deducir qué tipo de cadena de entrada va a conducir a la casilla con conflicto $[5, *]$. Vamos a pensar en ello siguiendo exactamente el mismo procedimiento que en los apartados anteriores, es decir, mirando los ítems del estado 5 que determinan acciones con relación al operador $*$. Echamos, de nuevo, un vistazo sobre esos ítems:

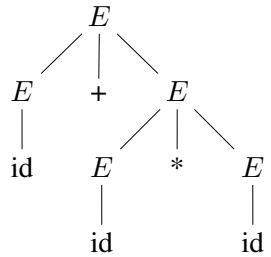
$$I_5 = \{\ E \to E + E\cdot\ ,\ E \to E \cdot + E\ ,\ E \to E \cdot * E\ \}$$

Está claro que el último tiene algo que decirnos ahora. Corresponde con la situación en la que se ha visto una expresión y ahora se encuentra un operador de multiplicación, estando en el estado 5. ¿Y el segundo, nos dice algo relativo a la entrada actual? La respuesta es no. ¿Y el primer ítem? Sí, ése también me dice algo respecto al operador $*$, aunque no salga por ningún sitio. Ese ítem me está diciendo que, una vez vista una suma de dos expresiones, puedo reducir esa suma a una nueva expresión en el caso de encontrar en la entrada a un token perteneciente a SIGUIENTE(E), conjunto en el que se encuentra el operador $*$.

El siguiente paso consiste en combinar las situaciones reflejadas por el primer y último ítem del conjunto 5 para dar lugar al uso de la casilla $[5, *]$. ¿Qué cadena de entrada puede reflejar esto? A estas alturas parece fácil: $id + id * id$, por ejemplo. Cuando se haya analizado la suma y se vea en la entrada el operador $*$ nos encontraremos, precisamente, en la necesidad de usar la casilla que ahora nos ocupa.

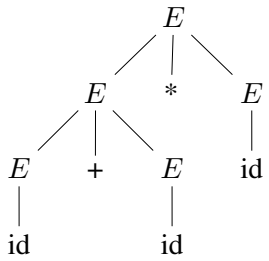
Para tantear la solución a este conflicto optamos, primero, por la acción de desplazamiento en la casilla $[5, *]$. Realizamos ahora el análisis de la cadena $id + id * id$ usando esta acción en la simulación:

PILA	ENTRADA	ACCIÓN
0	$id\ +\ id\ *\ id\ \$$	d2
$0\ id\ 2$	$+\ id\ *\ id\ \$$	r3 $E \to id$
$0\ E\ 1$	$+\ id\ *\ id\ \$$	d3
$0\ E\ 1\ +\ 3$	$id\ *\ id\ \$$	d2
$0\ E\ 1\ +\ 3\ id\ 2$	$*\ id\ \$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5$	$*\ id\ \$$	d4 ¡Optamos por desplazar!
$0\ E\ 1\ +\ 3\ E\ 5\ *\ 4$	$id\ \$$	d2
$0\ E\ 1\ +\ 3\ E\ 5\ *\ 4\ id\ 2$	$\$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5\ *\ 4\ E\ 6$	$\$$	r2 $E \to E * E$
$0\ E\ 1\ +\ 3\ E\ 5$	$\$$	r1 $E \to E + E$
$0\ E\ 1$	$\$$	acc



El resultado es un árbol en el que la rama de la derecha agrupa la multiplicación de los dos últimos ídentificadores, y al resultado de esta operación se le añade el primer identificador. Por tanto, la operación que se está ejecutando es $id + (id * id)$, es decir, se da precedencia (o prioridad) a la multiplicación sobre la suma. Esto está bien. Vamos a ver qué pasaría si la acción elegida fuese la contraria.

PILA	ENTRADA	ACCIÓN
0	$id\ +\ id\ *\ id\ \$$	d2
$0\ id\ 2$	$+\ id\ *\ id\ \$$	r3 $E \to id$
$0\ E\ 1$	$+\ id\ *\ id\ \$$	d3
$0\ E\ 1\ +\ 3$	$id\ *\ id\ \$$	d2
$0\ E\ 1\ +\ 3\ id\ 2$	$*\ id\ \$$	r3 $E \to id$
$0\ E\ 1\ +\ 3\ E\ 5$	$*\ id\ \$$	r1 $E \to E + E$ ¡Optamos por reducir!
$0\ E\ 1$	$*\ id\ \$$	d4
$0\ E\ 1\ *\ 4$	$id\ \$$	d2
$0\ E\ 1\ *\ 4\ id\ 2$	$\$$	r3 $E \to id$
$0\ E\ 1\ *\ 4\ E\ 6$	$\$$	r2 $E \to E * E$
$0\ E\ 1$	$\$$	acc



Este árbol de análisis no es adecuado, porque la interpretación que representa es la correspondiente a $(id + id) * id$, es decir, se da precedencia a la suma sobre la multiplicación, y eso no es una cosa muy lógica, teniendo el cuenta la semántica habitual de esos operadores aritméticos.

Resumiendo, el conflicto de la entrada $[5, *]$ se resuelve optando por el desplazamiento $\mathbf{d4}$.

6. Conflicto del estado 6 con el operador +

Éste queda como ejercicio. Si has entendido todo lo anterior, no te resultará complicado. Si no lo has entendido, es mejor que visites a algún profesor en horario de tutorías, ¡y no lo dejes para el final!