Capítulo 5

Autómatas a pila

Los autómatas a pila son máquinas abstractas que reconocen exactamente la clase de los lenguajes independientes del contexto. En este capítulo introducimos este tipo de autómatas, que servirán de base a los modelos de autómata para lenguajes de adjunción de árboles que serán presentados en los capítulos siguientes.

5.1. Definición

Presentamos dos definiciones diferentes pero equivalentes de los autómatas a pila (*Push-Down Automata*, PDA) [85]. En primer lugar presentaremos la definición clásica, que considera un autómata a pila como una máquina abstracta que consta de tres componentes: una cadena de entrada, un control finito y una pila. Seguidamente presentaremos una definición más moderna en la cual se suprimen las referencias al control finito para centrase en el componente fundamental de este tipo de autómatas: la pila.

5.1.1. Definición con estados

En la definición clásica [85], los autómatas a pila son considerados tuplas $(Q, V_T, V_S, \delta, q_0, \$_0, Q_F)$, donde:

- Q es un conjunto finito de estados.
- V_T es un conjunto finito de símbolos terminales.
- V_S es un conjunto finito de símbolos de pila.
- $q_0 \in Q$ es el estado inicial.
- $\$_0 \in V_S$ es el símbolo inicial de la pila.
- $Q_F \subseteq Q$ es el conjunto de estados finales.
- δ es una relación de $Q \times (V_T \cup \{\epsilon\}) \times V_S$ en subconjuntos finitos de $Q \times V_S^*$ que define los movimientos o transiciones válidos del autómata. Una transición

  $$(q', \beta) \in \delta(q, a, Z)$$

  donde $q, q' \in Q$, $a \in V_T \cup \{\epsilon\}$, $Z \in V_S \cup \{\epsilon\}$, $\beta \in V_S^*$, se interpreta como sigue: si el autómata se encuentra en el estado q, el siguiente terminal en la cadena de entrada es a y el símbolo en la cima de la pila es Z, entonces puede pasar al estado q' y reemplazar la cima de la pila por β.

La *configuración* de un autómata a pila en un momento dado viene definida por el triple (q, α, w), donde q indica el estado en el que se encuentra el autómata, α el contenido de la pila y w la parte de la cadena de entrada que resta por leer. El cambio de una configuración a otra viene determinado por la aplicación de una transición, de tal modo que si $(q, \alpha Z, aw)$ es una configuración y $(q', \beta) \in \delta(q, a, Z)$, entonces el autómata pasará a la nueva configuración $(q', \alpha\beta, w)$, hecho que denotamos mediante $(q, \alpha, aw) \vdash (q', \alpha\beta, w)$. Denotamos por $\overset{*}{\vdash}$ el cierre reflexivo y transitivo de $\vdash$.

El *lenguaje aceptado por estado final* por un autómata a pila viene determinado por el conjunto de cadenas $w \in V_T^*$ tal que $(q_0, \$_0, w) \overset{*}{\vdash} (p, \alpha, \epsilon)$, donde $p \in Q_F$ y $\alpha \in V_S^*$.

El *lenguaje aceptado por pila vacía* por un autómata a pila viene determinado por el conjunto de cadenas $w \in V_T^*$ tal que $(q_0, \$_0, w) \overset{*}{\vdash} (p, \epsilon, \epsilon)$ para cualquier $p \in Q$.

Dado un autómata a pila que reconoce un determinado lenguaje por estado final, es posible construir otro autómata a pila que reconoce el mismo lenguaje por pila vacía y viceversa [85].

5.1.2. Definición sin estados

El control finito de un autómata a pila es un elemento prescindible, puesto que el estado de una configuración dada puede almacenarse en el elemento situado en la cima de la pila [24]. Como consecuencia obtenemos una definición alternativa equivalente [107, 24, 52], que juzgamos más simple y homogénea, según la cual un autómata a pila es una tupla $(V_T, V_S, \Theta, \$_0, \$_f)$, donde

- V_T es un conjunto finito de símbolos terminales.
- V_S es un conjunto finito de símbolos de pila.
- $\$_0 \in V_S$ es el símbolo inicial de la pila.
- $\$_f \in V_S$ es el símbolo final de la pila.
- Θ es un conjunto de transiciones, cada una de las cuales pertenece a uno de los tres tipos siguientes, donde $C, F, G \in V_S$, $\xi \in V_S^*$ y $a \in V_T \cup \{\epsilon\}$:

  SWAP: Transiciones de la forma $C \overset{a}{\longmapsto} F$ que reemplazan el elemento C de la cima de la pila por el elemento F mientras se lee a de la cadena de entrada. El resultado de aplicar una transición de este tipo a una pila ξC es una pila ξF.

  PUSH: Transiciones de la forma $C \overset{a}{\longmapsto} CF$ que apilan un nuevo elemento F en la pila mientras se lee a de la cadena de entrada. El resultado de aplicar una transición de este tipo a una pila ξC es una pila ξCF.

  POP: Transiciones de la forma $CF \overset{a}{\longmapsto} G$ que eliminan los dos elementos C y F de la cima de la pila y los sustituyen por G mientras se lee a de la cadena de entrada. El resultado de aplicar una transición de este tipo a una pila ξCF es una pila ξG, con lo cual el tamaño de la pila decrece en una unidad.

Según la nueva definición, la *configuración* de un autómata a pila en un momento dado viene determinada por el par (ξ, w), donde ξ es el contenido de la pila y w es la parte de la cadena de entrada que resta por leer. Una configuración (ξ, aw) deriva una configuración (ξ', w), hecho que denotamos mediante $(\xi, aw) \vdash (\xi', w)$, si y sólo si existe una transición que aplicada a ξ devuelve ξ' y consume a de la cadena de entrada. En caso de ser necesario identificar una derivación d concreta, utilizaremos la notación $\overset{*}{\vdash}_d$. Denotamos por $\overset{*}{\vdash}$ el cierre reflexivo y transitivo de $\vdash$.

Decimos que una cadena de entrada w es aceptada por un autómata a pila si $(\$_0, w) \overset{*}{\vdash} (\$_0\$_f, \epsilon)$. El *lenguaje aceptado* por un autómata a pila viene determinado por el conjunto de cadenas $w \in V_T^*$ tal que $(\$_0, w) \overset{*}{\vdash} (\$_0\$_f, \epsilon)$.

5.2. Esquemas de compilación

Un esquema de compilación es un conjunto de reglas que permite, a partir de una gramática independiente del contexto y de una estrategia de análisis sintáctico, construir un autómata a pila que describa los cálculos que se pueden realizar con dicha gramática utilizando la estrategia de análisis elegida.

Los esquemas de compilación que se van a mostrar se basan en el paradigma de llamada/retorno [55], utilizando para ello los seis tipos de reglas mostrados en la tabla 5.1. A toda regla [**CALL**] le corresponde una regla [**RET**] y viceversa. Las reglas [**INIT**], [**CALL**] y [**SEL**] definen las transiciones del autómata encargadas de la fase predictiva del algoritmo de análisis mientras que las reglas [**RET**] y [**PUB**] definen las transiciones encargadas de propagar la información en la fase ascendente. Por este motivo la fase descendente o predictiva de una estrategia de análisis cuando es implantada en un autómata a pila recibe el nombre de *fase de llamada*, mientras que la fase ascendente recibe el nombre de *fase de retorno*.

Regla	Tarea
[**INIT**]	inicia los cálculos a partir de la pila inicial.
[**CALL**]	requiere el análisis de un no-terminal de una producción.
[**SEL**]	selecciona una producción.
[**PUB**]	determina que una producción ha sido completamente analizada.
[**RET**]	continua el proceso de análisis después de terminar una producción.
[**SCAN**]	reconoce los terminales que componen la cadena de entrada.

Tabla 5.1: Reglas de los esquemas de compilación de gramáticas independientes del contexto

En primer lugar definiremos el esquema de compilación correspondiente a una estrategia genérica basada en el paradigma llamada/retorno, parametrizada con respecto a la información que se predice y propaga en las fases de llamada y de retorno, respectivamente. Utilizaremos la siguiente notación:

- $A_{r,s}$ para referirnos al elemento de la producción r que ocupa la posición s, de modo que para una producción r tenemos que $A_{r,0} \rightarrow A_{r,1} \ldots A_{r,n_r}$.
- $\nabla_{r,s}$ para indicar el reconocimiento parcial de una producción, notacionalmente equivalente a una producción con punto $A_{r,0} \rightarrow A_{r,1} \ldots A_{r,s} \bullet A_{r,s+1} \ldots A_{r,n_r}$.
- $\overrightarrow{A_{r,s}}$ para referirnos a la predicción de información con respecto a $A_{r,s}$.
- $\overleftarrow{A_{r,s}}$ para representar la información propagada ascendentemente con respecto a $A_{r,s}$.

Con el fin de simplificar al máximo la definición de los esquemas de compilación y sin pérdida de generalidad, supondremos que se cumplen las siguientes condiciones sobre la gramática independiente del contexto:

- El axioma sólo aparece en el lado izquierdo de la producción unitaria 0, que tiene la forma $S \to X$, con $X \in V_N \cup V_T$
- Los terminales sólo aparecen en producciones unitarias de la forma $A_{r,0} \to a$, donde $a \in V_T \cup \{\epsilon\}$.

Esquema de compilación 5.1 El esquema de compilación genérico de una gramática independiente del contexto en un autómata a pila queda definido por el conjunto de reglas mostrado a continuación y por los elementos inicial $\$_0$ y final $\overleftarrow{S}$. La primera columna indica el nombre de la regla, la segunda las transiciones generadas por la misma y la tercera las condiciones, generalmente referidas a la forma de las producciones, que se deben cumplir para que la regla sea aplicable.

[INIT]	$\$_0 \longmapsto \$_0 \; \nabla_{0,0}$	
[CALL]	$\nabla_{r,s} \longmapsto \nabla_{r,s} \; \overrightarrow{A_{r,s+1}}$	
[SEL]	$\overrightarrow{A_{r,0}} \longmapsto \nabla_{r,0}$	$r \neq 0$
[PUB]	$\nabla_{r,n_r} \longmapsto \overleftarrow{A_{r,0}}$	
[RET]	$\nabla_{r,s} \; \overleftarrow{A_{r,s+1}} \longmapsto \nabla_{r,s+1}$	
[SCAN]	$\overrightarrow{A_{r,0}} \overset{a}{\longmapsto} \overleftarrow{A_{r,0}}$	$A_{r,0} \to a$

§

A continuación presentamos tres versiones concretas del esquema de compilación genérico. La primera se corresponde con una estrategia de análisis descendente en la cual los no-terminales se predicen en la fase de llamada pero no se propagan en la fase ascendente. La segunda se corresponde con una estrategia mixta de tipo Earley [69] en la que los no-terminales se predicen en la fase de llamada y se propagan en la fase de retorno. La tercera y última se corresponde con una estrategia ascendente en la cual no hay ningún tipo de predicción en la fase de llamada mientras que en la fase de retorno se propagan los no-terminales analizados.

En la tabla 5.2 se muestran los valores que deben tomar los parámetros de predicción y propagación de información para transformar el esquema de compilación genérico en esquemas correspondientes a las tres estrategias de análisis citadas, donde $\square$ representa un símbolo de pila nuevo. En el caso de estrategias Earley es necesario distinguir la llamada de un no-terminal $A_{r,s+1}$ de su retorno, para lo cual utilizamos los símbolos $\overline{A_{r,s+1}}$ y $\overline{\overline{A_{r,s+1}}}$, respectivamente.

5.2.1. Estrategia descendente

Esquema de compilación 5.2 El esquema de compilación descendente de una gramática independiente del contexto en un autómata a pila queda definido por el conjunto de reglas mostrado en la tabla 5.3 y por los elementos inicial $\$_0$ y final $\square$. §

Estrategia	$\overrightarrow{A_{r,s+1}}$	$\overleftarrow{A_{r,s+1}}$
Ascendente	$\square$	$A_{r,s+1}$
Earley	$\overline{A_{r,s+1}}$	$\overline{\overline{A_{r,s+1}}}$
Descendente	$A_{r,s+1}$	$\square$

Tabla 5.2: Parámetros del esquema de compilación genérico de CFG en PDA

Ejemplo 5.1 Consideremos la gramática independiente del contexto definida por las producciones:

$$\begin{array}{ll} (0) & S \rightarrow X \\ (1) & X \rightarrow AXB \\ (2) & X \rightarrow \epsilon \\ (3) & A \rightarrow a \\ (4) & B \rightarrow b \end{array}$$

que genera el lenguaje $\{a^n b^n \mid n \geq 0\}$. En la tabla 5.4 se muestra el conjunto de transiciones del autómata a pila que se obtiene al aplicar el esquema de compilación descendente.

En la tabla 5.5 se muestra la secuencia de configuraciones que sigue el autómata para analizar correctamente la cadena de entrada *aabb*. La primera columna muestra la transición aplicada, la segunda el tipo de la regla que generó dicha transición, la tercera el contenido de la pila y la cuarta la parte que resta por leer de la cadena de entrada. Las configuraciones marcadas con $*$ son aquellas en las que hay más de una opción para proseguir el análisis. Se trata de configuraciones resultado de aplicar una transición de tipo **[CALL]**. Si el elemento apilado coincide con el lado izquierdo de varias producciones, debe proseguirse el análisis por cada una de ellas. En la tabla 5.5 sólo se muestran las configuraciones que llevan al reconocimiento de la cadena de entrada. El resto de las posibles configuraciones llevan al autómata a detenerse en configuraciones que no son finales. ¶

5.2.2. Estrategia Earley

Esquema de compilación 5.3 El esquema de compilación para una estrategia de tipo Earley de una gramática independiente del contexto en un autómata a pila queda definido por el conjunto de reglas mostrado en la tabla 5.6 y por los elementos inicial $\$_0$ y final $\overline{\overline{S}}$. §

5.2.3. Estrategia ascendente

Esquema de compilación 5.4 El esquema de compilación ascendente de una gramática independiente del contexto en un autómata a pila queda definido por el conjunto de reglas mostrado en la tabla 5.7 y por los elementos inicial $\$_0$ y final S. §

[INIT]	$\$_0 \longmapsto \$_0 \ \nabla_{0,0}$	
[CALL]	$\nabla_{r,s} \longmapsto \nabla_{r,s} \ \ A_{r,s+1}$	
[SEL]	$A_{r,0} \longmapsto \nabla_{r,0}$	$r \neq 0$
[PUB]	$\nabla_{r,n_r} \longmapsto \square$	
[RET]	$\nabla_{r,s} \ \square \ \longmapsto \nabla_{r,s+1}$	
[SCAN]	$A_{r,0} \overset{a}{\longmapsto} \square$	$A_{r,0} \to a$

Tabla 5.3: Reglas del esquema de compilación descendente de CFG en PDA

(a)	**[INIT]**	$\$_0 \longmapsto \$_0 \ \nabla_{0,0}$
(b)	**[CALL]**	$\nabla_{0,0} \longmapsto \nabla_{0,0} \ \ X$
(c)	**[RET]**	$\nabla_{0,0} \ \square \ \longmapsto \nabla_{0,1}$
(d)	**[PUB]**	$\nabla_{0,1} \longmapsto \square$
(e)	**[SEL]**	$X \longmapsto \nabla_{1,0}$
(f)	**[CALL]**	$\nabla_{1,0} \longmapsto \nabla_{1,0} \ A$
(g)	**[RET]**	$\nabla_{1,0} \ \square \ \longmapsto \nabla_{1,1}$
(h)	**[CALL]**	$\nabla_{1,1} \longmapsto \nabla_{1,1} \ X$
(i)	**[RET]**	$\nabla_{1,1} \ \square \ \longmapsto \nabla_{1,2}$
(j)	**[CALL]**	$\nabla_{1,2} \longmapsto \nabla_{1,2} \ B$
(k)	**[RET]**	$\nabla_{1,2} \ \square \ \longmapsto \nabla_{1,3}$
(l)	**[PUB]**	$\nabla_{1,3} \longmapsto \square$
(m)	**[SCAN]**	$X \overset{\epsilon}{\longmapsto} \square$
(n)	**[SCAN]**	$A \overset{a}{\longmapsto} \square$
(p)	**[SCAN]**	$B \overset{b}{\longmapsto} \square$

Tabla 5.4: Transiciones del autómata a pila con estrategia descendente

		$\$_0$					$aabb$	
(a)	**[INIT]**	$\$_0$	$\nabla_{0,0}$				$aabb$	
(b)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	X			$aabb$	$*$
(e)	**[SEL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,0}$			$aabb$	
(f)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,0}$	A		$aabb$	
(n)	**[SCAN]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,0}$	$\square$		abb	
(g)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$			abb	
(h)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	X		abb	$*$
(e)	**[SEL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,0}$		abb	
(f)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,0}$	A	abb	
(n)	**[SCAN]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,0}$	$\square$	bb	
(g)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,1}$		bb	
(h)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,1}$	X	bb	$*$
(m)	**[SCAN]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,1}$	$\square$	bb	
(i)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,2}$		bb	
(j)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,2}$	B	bb	
(p)	**[SCAN]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,2}$	$\square$	b	
(k)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\nabla_{1,3}$		b	
(l)	**[PUB]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,1}$	$\square$		b	
(i)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,2}$			b	
(j)	**[CALL]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,2}$	B		b	
(p)	**[SCAN]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,2}$	$\square$			
(k)	**[RET]**	$\$_0$	$\nabla_{0,0}$	$\nabla_{1,3}$				
(l)	**[PUB]**	$\$_0$	$\nabla_{0,0}$	$\square$				
(c)	**[RET]**	$\$_0$	$\nabla_{0,1}$					
(d)	**[PUB]**	$\$_0$	$\square$					

Tabla 5.5: Configuraciones del autómata a pila descendente durante el análisis de *aabb*

[INIT]	$\$_0 \longmapsto \$_0 \ \nabla_{0,0}$	
[CALL]	$\nabla_{r,s} \longmapsto \nabla_{r,s} \ \overline{A_{r,s+1}}$	
[SEL]	$\overline{A_{r,0}} \longmapsto \nabla_{r,0}$	$r \neq 0$
[PUB]	$\nabla_{r,n_r} \longmapsto \overline{\overline{A_{r,0}}}$	
[RET]	$\nabla_{r,s} \ \overline{\overline{A_{r,s+1}}} \longmapsto \nabla_{r,s+1}$	
[SCAN]	$\overline{A_{r,0}} \overset{a}{\longmapsto} \overline{\overline{A_{r,0}}}$	$A_{r,0} \to a$

Tabla 5.6: Reglas del esquema de compilación Earley de CFG en PDA

[INIT]	$\$_0 \longmapsto \$_0 \ \nabla_{0,0}$	
[CALL]	$\nabla_{r,s} \longmapsto \nabla_{r,s} \ \square$	
[SEL]	$\square \longmapsto \nabla_{r,0}$	$r \neq 0$
[PUB]	$\nabla_{r,n_r} \longmapsto A_{r,0}$	
[RET]	$\nabla_{r,s} \ A_{r,s+1} \longmapsto \nabla_{r,s+1}$	
[SCAN]	$\square \overset{a}{\longmapsto} A_{r,0}$	$A_{r,0} \to a$

Tabla 5.7: Reglas del esquema de compilación ascendente de CFG en PDA

5.3. Tabulación

La independencia del contexto de las transiciones de los autómatas a pila permite tabular la ejecución de los mismos. En esta sección presentamos dos técnicas diferentes para la tabulación de los autómatas a pila, una propuesta por Lang [104, 107] y la otra propuesta por Nederhof [126].

5.3.1. La técnica de Lang

En una gramática independiente del contexto, si $B \overset{*}{\Rightarrow} \delta$ entonces $\alpha B \beta \overset{*}{\Rightarrow} \alpha \delta \beta$ para todo $\alpha, \beta \in (V_N \cup V_T)^*$. Esta misma independencia del contexto se traslada a los autómatas a pila, de tal modo que si

$$(B, a_{i+1} \dots a_j \dots a_n) \overset{*}{\vdash} (BC, a_{j+1} \dots a_n)$$

entonces también se cumple que

$$(\xi B, a_{i+1} \dots a_j \dots a_n) \overset{*}{\vdash} (\xi BC, a_{j+1} \dots a_n)$$

para todo $\xi \in V_S^*$. Denominaremos *derivaciones independientes del contexto* a este tipo de derivaciones. Observamos que este tipo de derivaciones presenta gran semejanza con las transiciones PUSH.

Para representar una derivación independiente del contexto sólo precisamos almacenar los símbolos de pila B y C más las posiciones de la cadena de entrada i y j, puesto que la derivación es independiente de ξ. La técnica de tabulación de autómatas a pila propuesta por Lang [104, 107]

se basa precisamente en reemplazar la manipulación de pilas por la manipulación de ítems de la forma

$$[B, i, C, j]$$

que representan de forma compacta el conjunto de derivaciones independientes del contexto que comparten los elementos de la cima de la pila. Los ítems se combinan mediante *reglas de combinación* de la forma $\frac{ants}{cons}\ trans$, donde *cons* es el ítem consecuente que se obtiene al aplicar la transición *trans* sobre los ítems antecedentes *ants*. A continuación mostramos las reglas de combinación de ítems para los tres tipos de transiciones:

Transiciones SWAP: la aplicación de una transición $C \overset{a}{\longmapsto} F$ produce la derivación $(\xi C, a_j \dots a_n) \vdash (\xi F, a_k \dots a_n)$ en un autómata a pila, donde $k = j$ si $a = \epsilon$ y $k = j+1$ si $a = a_{j+1}$. Dada la derivación independiente del contexto $(\xi' B, a_{i+1} \dots a_n) \overset{*}{\vdash} (\xi' BC, a_{j+1} \dots a_n)$ que da lugar a la configuración $(\xi' BC, a_{j+1} \dots a_n)$, tras la aplicación de la transición obtendremos la derivación independiente del contexto $(\xi' B, a_{i+1} \dots a_n) \overset{*}{\vdash} (\xi' BF, a_{k+1} \dots a_n)$. La correspondiente regla de combinación de ítems es

$$\frac{[B, i, C, j]}{[B, i, F, k]}\ C \overset{a}{\longmapsto} F$$

Transiciones PUSH: la aplicación de una transición $C \overset{a}{\longmapsto} CF$ produce la derivación $(\xi C, a_{j+1} \dots a_n) \vdash (\xi CF, a_{k+1} \dots a_n)$, donde $k = j$ si $a = \epsilon$ y $k = j + 1$ si $a = a_{j+1}$. Esta derivación es por sí misma una derivación independiente del contexto, por lo que la correspondiente regla de combinación de ítems es

$$\frac{[B, i, C, j]}{[C, j, F, k]}\ C \overset{a}{\longmapsto} CF$$

Transiciones POP: la aplicación de una transición $CF \overset{a}{\longmapsto} G$ produce la derivación $(\xi CF, a_{k+1} \dots a_n) \vdash (\xi G, a_{l+1} \dots a_n)$. La configuración $(\xi CF, a_{k+1} \dots a_n)$ refleja una derivación independiente del contexto $(\xi C, a_{j+1} \dots a_n) \overset{*}{\vdash} (\xi CF, a_{k+1} \dots a_n)$, pero además necesitamos la derivación independiente del contexto $(\xi' B, a_{i+1} \dots a_n) \overset{*}{\vdash} (\xi' BC, a_{j+1} \dots a_n)$ que colocó C en la cima de la pila para obtener la derivación independiente del contexto $(\xi' B, a_{i+1} \dots a_n) \overset{*}{\vdash} (\xi' BG, a_{l+1} \dots a_n)$ resultante de la aplicación de la transición, donde $l = k$ si $a = \epsilon$ y $l = k+1$ si $a = a_{k+1}$. La correspondiente regla de combinación de ítems es

$$\frac{\begin{array}{c}[C, j, F, k] \\ [B, i, C, j]\end{array}}{[B, i, G, l]}\ CF \overset{a}{\longmapsto} G$$

La manipulación de configuraciones mediante la aplicación de transiciones es equivalente a la manipulación de ítems mediante las reglas de combinación correspondientes a cada transición [104, 107].

5.3.2. La técnica de Nederhof

Aunque Nederhof no ha presentado una técnica específica para la tabulación de autómatas a pila, la misma se obtiene como un caso especial de la técnica de tabulación propuesta por dicho autor para los autómatas lineales de índices [126].

A diferencia de la técnica de Lang, que traslada la propiedad de independencia del contexto de las gramáticas independientes del contexto a derivaciones con la forma de transiciones de PUSH, la técnica de tabulación de Nederhof traslada dicha propiedad a derivaciones con la forma de transiciones SWAP, puesto que si tenemos una derivación

$$(B, a_i \dots a_j \dots a_n) \overset{*}{\vdash} (C, a_j \dots a_n)$$

entonces también se cumple que

$$(\xi B, a_i \dots a_j \dots a_n) \overset{*}{\vdash} (\xi C, a_j \dots a_n)$$

para todo $\xi \in V_S^*$. Al igual que en la técnica de Lang, estas derivaciones también se denominan *derivaciones independientes del contexto*.

Para representar una derivación independiente del contexto sólo precisamos almacenar B, C y las posiciones i y j, por lo que se utilizarán ítems de la forma

$$[B, i, C, j]$$

Estos ítems se manipulan mediante reglas de combinación, una para cada tipo de transición:

Transiciones SWAP: la aplicación de una transición $C \overset{a}{\longmapsto} F$ produce una derivación $(\xi C, a_j \dots a_n) \vdash (\xi F, a_k \dots a_n)$ en un autómata a pila, donde $k = j$ si $a = \epsilon$ y $k = j + 1$ si $a = a_{j+1}$. Dada la derivación independiente del contexto $(\xi B, a_i \dots a_n) \overset{*}{\vdash} (\xi C, a_j \dots a_n)$, tras la aplicación de la transición obtendremos la derivación independiente del contexto $(\xi B, a_i \dots a_n) \overset{*}{\vdash} (\xi F, a_k \dots a_n)$. La correspondiente regla de combinación de ítems es

$$\frac{[B, i, C, j]}{[B, i, F, k]} \quad C \overset{a}{\longmapsto} F$$

Transiciones PUSH: la aplicación de una transición $C \overset{a}{\longmapsto} CF$ produce la derivación $(\xi C, a_{j+1} \dots a_n) \vdash (\xi CF, a_{k+1} \dots a_n)$, donde $k = j$ si $a = \epsilon$ y $k = j + 1$ si $a = a_{j+1}$. Dada la derivación independiente del contexto $(\xi B, a_i \dots a_n) \overset{*}{\vdash} (\xi C, a_j \dots a_n)$, tras la aplicación de la transición obtendremos la derivación independiente del contexto $(\xi F, a_{k+1} \dots a_n) \overset{0}{\vdash} (\xi F, a_{k+1} \dots a_n)$. La correspondiente regla de combinación de ítems es

$$\frac{[B, i, C, j]}{[F, k, F, k]} \quad C \overset{a}{\longmapsto} CF$$

Transiciones POP: la aplicación de una transición $CF \overset{a}{\longmapsto} G$ produce una derivación $(\xi CF, a_{k+1} \dots a_n) \vdash (\xi G, a_{l+1} \dots a_n)$, donde $l = k$ si $a = \epsilon$ y $l = k + 1$ si $a = a_{k+1}$. La configuración $(\xi CF, a_{k+1} \dots a_n)$ refleja una derivación constituida por

1. una derivación independiente del contexto $(\xi B, a_{i+1} \dots a_n) \overset{*}{\vdash} (\xi C, a_{j+1} \dots a_n)$;
2. un paso de derivación $(\xi C, a_{j+1} \dots a_n) \vdash (\xi CF', a_{k'+1} \dots a_n)$ resultado de la aplicación de una transición $C \overset{b}{\longmapsto} CF'$, donde $k' = j$ si $b = \epsilon$ y $k' = j + 1$ si $b = a_{j+1}$;
3. una derivación independiente del contexto $(\xi CF', a_{k'+1} \dots a_n) \overset{*}{\vdash} (\xi CF, a_{k+1} \dots a_n)$.

En consecuencia, la regla de combinación tendrá la forma:

$$\frac{\begin{array}{l}[F', k', F, k] \\ [B, i, C, j]\end{array}}{[B, i, G, l]} \quad \begin{array}{l} C \overset{b}{\longmapsto} CF' \\ CF \overset{a}{\longmapsto} G \end{array}$$