Teoría de la Computación
II Semestre 2024
Profesor: Rodrigo De Castro K.

Capítulo 4

Autómatas con pila

En el presente capítulo presentamos el modelo de autómata con pila; distinguiremos las versiones determinista y no-determinista, pero, a diferencia de lo que sucede con los modelos AFD y AFN, los autómatas con pila deterministas y no-deterministas no resultan ser computacionalmente equivalentes.

4.1. Definición de los autómatas con pila

4.1.1. Autómatas con Pila Deterministas (AFPD)

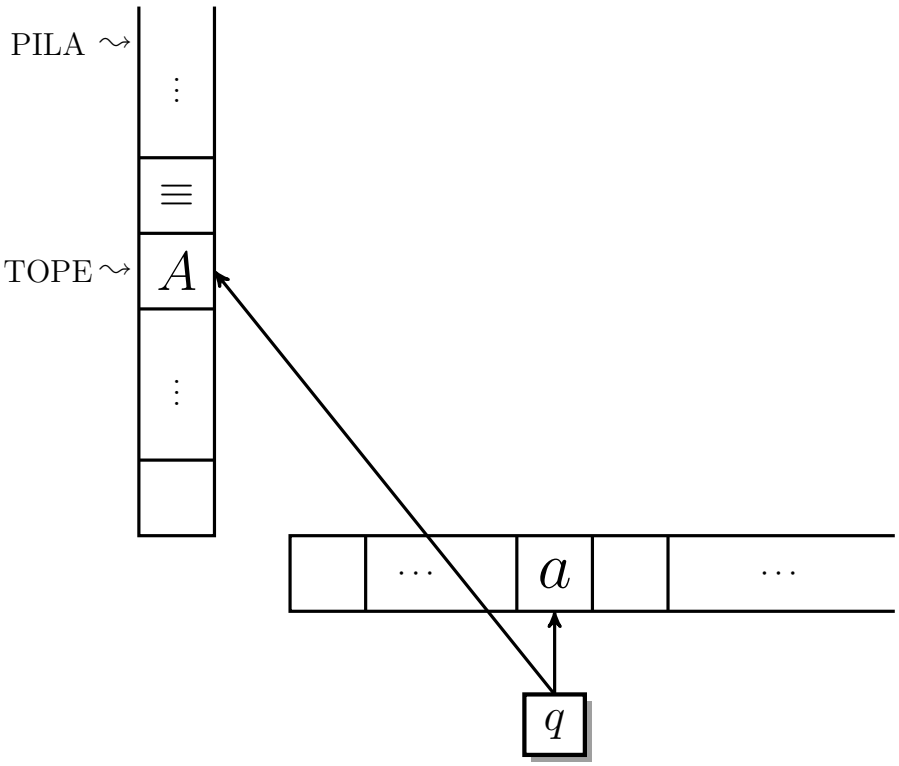
Un *Autómata Finito con Pila Determinista* (AFPD) es una séxtupla, $M = (Q, q_0, F, \Sigma, \Gamma, \Delta)$, con los siguientes componentes:

1. Q es el conjunto (finito) de estados internos de la unidad de control.
2. $q_0 \in Q$ es el estado inicial.
3. F es el conjunto de estados finales o de aceptación, $\varnothing \neq F \subseteq Q$.
4. Σ es el alfabeto de entrada, también llamado alfabeto de cinta.
5. Γ es el alfabeto de pila.
6. Δ es la función de transición del autómata:

$$\Delta : Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \longrightarrow (Q \times (\Gamma \cup \{\lambda\})).$$

Como en los modelos de autómatas básicos considerados en Capítulo 2, un autómata con pila lee cadenas sobre una cinta de entrada semi-infinita; estas entradas están escritas con el alfabeto Σ. Pero hay una cinta adicional, llamada pila, también semi-infinita, que es utilizada por el autómata como lugar de almacenamiento o memoria temporal. Los símbolos que el autómata puede colocar en la pila pertenecen al alfabeto Γ. Los alfabetos Σ y Γ pueden tener símbolos comunes, es decir, no son necesariamente disyuntos. Inicialmente, la pila está vacía y la unidad de control está escaneando el fondo de la pila.

En un momento determinado, la unidad de control del autómata está en el estado q escaneando un símbolo a sobre la cinta de entrada, y el símbolo A en el tope o cima de la pila, como lo muestra la siguiente gráfica:

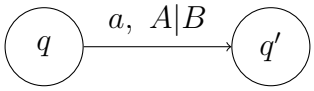


En todo momento la unidad de control solo tiene acceso al símbolo colocado en el tope de la pila (esa es la razón por la cual la pila se dibuja verticalmente). Al leer el símbolo a, la unidad de control se desplaza una casilla a la derecha, cambia al estado q' (que puede ser el mismo q) y realiza sobre la pila una de las siguientes cuatro acciones: o reemplaza el tope de la pila por otro símbolo, o añade un nuevo símbolo a la pila, o borra el tope de la pila, o no altera la pila. Las instrucciones permitidas están definidas en términos de la función de transición

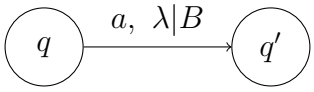
$$\Delta : Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \longrightarrow (Q \times (\Gamma \cup \{\lambda\}))$$
$$\Delta(q, a, A) = (q', B).$$

Hay que tener en cuenta los casos $a \in \Sigma$, $a = \lambda$, $A, B \in \Gamma$, $A = \lambda$ o $B = \lambda$, que se detallan a continuación.

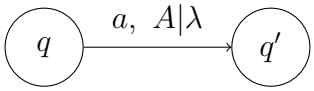
(1) Caso $a \in \Sigma$; $A, B \in \Gamma$. Instrucción $\Delta(q, a, A) = (q', B)$. El tope de la pila A se reemplaza por B. En otras palabras, el símbolo B sobre-escribe a A. En el grafo del autómata, esta instrucción la representamos como



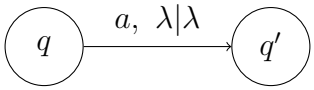
(2) Caso $a \in \Sigma$, $A = \lambda$, $B \in \Gamma$. Instrucción $\Delta(q, a, \lambda) = (q', B)$. Independientemente del tope actual de la pila, se inserta o añade un nuevo símbolo B a la pila. B pasa a ser el nuevo tope de la pila. En el grafo del autómata, esta instrucción la representamos como



(3) Caso $a \in \Sigma$, $A \in \Gamma$, $B = \lambda$. Instrucción $\Delta(q, a, A) = (q', \lambda)$. Se borra el tope de la pila A. La unidad de control pasa a escanear el símbolo ubicado en la casilla inmediatamente debajo, que es el nuevo tope de la pila. Si la A está inicialmente colocada en el fondo de la pila, entonces la pila se vacía y la unidad de control queda escaneando el fondo vacío. En el grafo del autómata, esta instrucción la representamos como



(4) Caso $a \in \Sigma$, $A = B = \lambda$. Instrucción $\Delta(q, a, \lambda) = (q', \lambda)$. El contenido de la pila no se altera. En el grafo del autómata, esta instrucción la representamos como



En las instrucciones (1) a (4) el símbolo a se consume, pero también se permiten transiciones λ o transiciones espontáneas, que se ejecutan independientemente del símbolo escaneado en la cinta de entrada. Con las transiciones λ, la unidad de control no se desplaza a la derecha. Las siguientes son las transiciones λ permitidas; las acciones realizadas por el autómata sobre la pila son las mismas que con las instrucciones homólogas (1) a (4):

(1′) $\Delta(q, \lambda, A) = (q', B)$.

(2′) $\Delta(q, \lambda, \lambda) = (q', B)$.

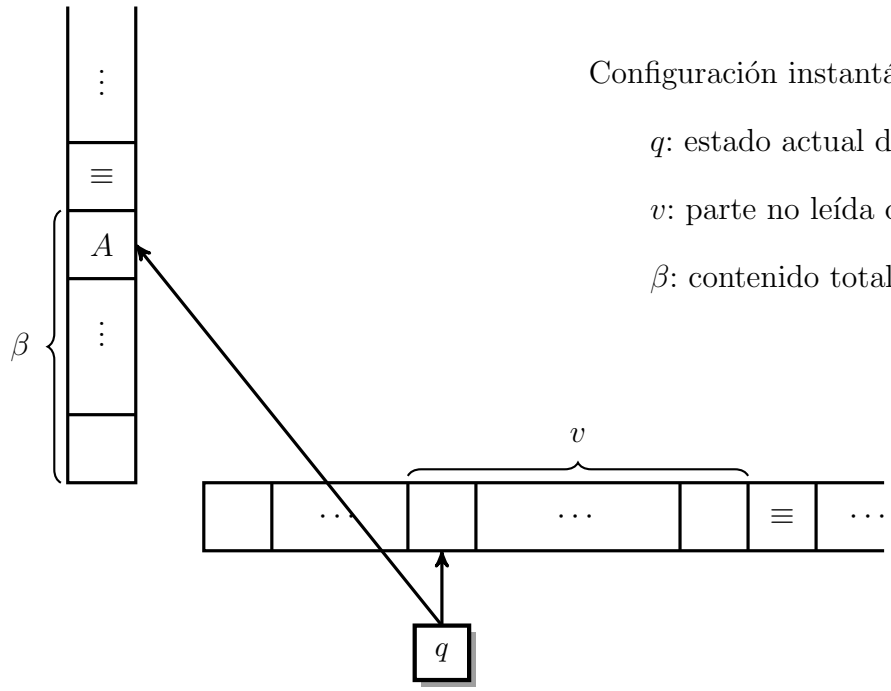
(3′) $\Delta(q, \lambda, A) = (q', \lambda)$.

(4′) $\Delta(q, \lambda, \lambda) = (q', \lambda)$.

Las transiciones λ o espontáneas permiten que el autómata cambie el contenido de la pila sin consumir símbolos sobre la cinta de entrada.

Es importante aclarar que la función Δ puede estar parcialmente definida; es decir, $\Delta(q, a, A)$ puede no estar definida, para algunos valores $q \in Q$, $a \in (\Sigma \cup \{\lambda\})$, $A \in (\Gamma \cup \{\lambda\})$. Se infiere que la lectura de algunas cadenas de entrada puede abortarse o detenerse sin que las entradas se lean completamente.

Configuración o descripción instantánea. Es una tripla $[q, v, \beta]$ que representa lo siguiente: la unidad de control está en el estado q, v es la parte no procesada de la cadena de entrada (la unidad de control está escaneando el primer símbolo de v), y la cadena β es el contenido total de la pila, tal como se exhibe en la siguiente gráfica:



Se adopta la convención de que la cadena β se lee de arriba hacia abajo. Así, en la gráfica anterior el primer símbolo de β es A, que es el tope de la pila.

La notación

$$[q, v, \beta] \vdash [q', w, \gamma].$$

representa un *paso computacional*, es decir, el autómata pasa de la configuración instantánea $[q, v, \beta]$ a la configuración $[q', w, \gamma]$ al ejecutar una de la instrucciones definidas por la función de transición Δ. Similarmente, la notación

$$[q, v, \beta] \vdash^* [p, w, \gamma]$$

significa que el autómata pasa de la configuración instantánea $[q, v, \beta]$ a la configuración instantánea $[p, w, \gamma]$ en uno o más pasos computacionales. También utilizaremos la notación $[q, v, \beta] \vdash^{k} [p, w, \gamma]$ para indicar que el autómata pasa de la configuración $[q, v, \beta]$ a la configuración $[p, w, \gamma]$ en k pasos.

Modelo Determinista (AFPD). En el modelo de autómata con pila determinista (AFPD) se debe cumplir la siguiente restricción adicional: dada cualquier configuración instantánea $[q, v, \beta]$ solamente hay una instrucción posible (a lo sumo) que se puede aplicar entre las transiciones disponibles desde el estado q. Así por ejemplo, para garantizar el determinismo, si la instrucción $\Delta(q, a, A)$ está definida (con $a \in \Sigma$ y $A \in \Gamma$), entonces ni $\Delta(q, \lambda, A)$ ni $\Delta(q, a, \lambda)$ pueden estar definidas simultáneamente; de lo contrario, en una configuración instantánea de la forma $[p, av, A\gamma]$ el autómata tendría varias opciones de proseguir el procesamiento de la entrada. En el modelo determinista se permiten transiciones λ siempre y cuando satisfagan la restricción anterior. Esto implica que un AFPD lee cada cadena de entrada $u \in \Sigma^*$ de manera única, aunque es posible que el procesamiento de u se aborte (o se detenga) sin consumir toda la entrada.

Configuración inicial y configuración de aceptación. Hay dos casos particulares importantes de configuraciones instantáneas. En primer lugar, la *configuración inicial* es $[q_0, u, \lambda]$ para una cadena de entrada $u \in \Sigma^*$. Al comenzar a leer una entrada, la pila está vacía y la unidad de control está escaneando el fondo de la pila.

La configuración $[p, \lambda, \lambda]$, siendo p un estado final o de aceptación, se llama *configuración de aceptación*. Esto significa que, para ser aceptada, una cadena de entrada debe ser leída completamente, la pila debe estar vacía y la unidad de control en un estado de aceptación.

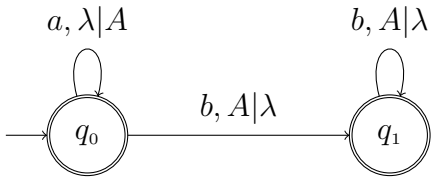
Lenguaje aceptado por un AFPD. El lenguaje aceptado por un AFPD M se define como

$$L(M) := \{u \in \Sigma^* : [q_0, u, \lambda] \vdash^{*} [p, \lambda, \lambda],\ p \in F\}.$$

O sea, una cadena u es aceptada si el único procesamiento posible de u desde la configuración inicial $[q_0, u, \lambda]$ termina en una configuración de aceptación. Como caso particular se deduce que si el estado inicial q_0 es un estado de aceptación, la cadena vacía λ es aceptada, ya que $[q_0, \lambda, \lambda]$ sería una configuración tanto inicial como de aceptación.

Ejemplo Diseñar un AFPD M que acepte el lenguaje $L = \{a^n b^n : n \geq 0\}$, sobre el alfabeto $\Sigma = \{a, b\}$. Recordemos que L no es regular y no puede ser aceptado por ningún autómata básico (sin pila).

Solución. La idea es almacenar las $a$es en la pila y borrar luego una a del tope de la pila por cada b que sea leída sobre la cinta. Una cadena será aceptada si es leída completamente y la pila se vacía. Para distinguir los símbolos de la cinta de entrada de los que hay en la pila, cada a se apilará como A; es decir, el alfabeto de pila es $\Gamma = \{A\}$. El grafo de M es:



Podemos ilustrar el procesamiento de varias cadenas de entrada $u \in \Sigma^*$. Sea, inicialmente, $u = aaabbb$.

$$[q_0, aaabbb, \lambda] \vdash [q_0, aabbb, A] \vdash [q_0, abbb, AA] \vdash [q_0, bbb, AAA] \vdash [q_1, bb, AA]$$
$$\vdash [q_1, b, A] \vdash [q_1, \lambda, \lambda].$$

La última es una configuración de aceptación; por lo tanto la cadena $u = aaabbb$ es aceptada.

Si una cadena tiene más $b$es que $a$es, el autómata no la lee completamente. Por ejemplo, para la cadena de entrada $u = aabbb$, se obtiene el siguiente procesamiento:

$$[q_0, aabbb, \lambda] \vdash [q_0, abbb, A] \vdash [q_0, bbb, AA] \vdash [q_1, bb, A]$$
$$\vdash [q_1, b, \lambda] \quad \text{(Procesamiento abortado)}.$$

Obsérvese que el autómata termina en el estado de aceptación q_1, con la pila vacía, pero la cadena de entrada no es aceptada debido a que no se ha leído completamente; $[q_1, b, \lambda]$ no es una configuración de aceptación.

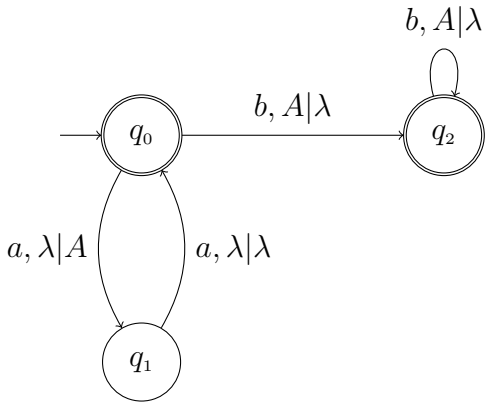
Si una cadena tiene más $a$es que $b$es, la pila no se vacía. Por ejemplo, para la cadena de entrada $u = aaabb$, se tiene:

$$[q_0, aaabb, \lambda] \vdash^3 [q_0, bb, AAA] \vdash [q_1, b, AA] \vdash [q_1, \lambda, A].$$

A pesar de que la cadena de entrada w ha sido leída completamente y el procesamiento termina en el estado de aceptación q_1, la configuración $[q_1, \lambda, A]$ no es de aceptación. Por lo tanto, $u = aaabb$ no es aceptada.

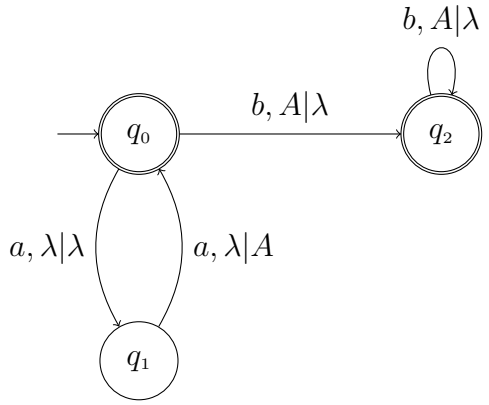
Ejemplo Diseñar un AFPD M que acepte el lenguaje $L = \{a^{2n}b^n : n \geq 0\}$, sobre el alfabeto $\Sigma = \{a, b\}$. L no es un regular y no puede ser aceptado por ningún autómata básico (sin pila).

Solución. La idea es apilar una A por cada dos $a$es leídas en la cinta y luego borrar una A de la pila por cada b que sea leída sobre la cinta. El alfabeto de pila es $\Gamma = \{A\}$ y el grafo de M es:



Obsérvese que las cadenas aceptadas por M tienen un número par de $a$es. Además, si una entrada solamente tiene $a$es, no es aceptada porque la pila no se vacía.

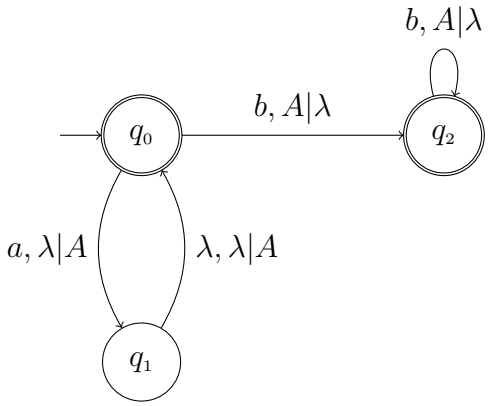
Las dos transiciones que unen a q_0 con q_1 se pueden invertir para aceptar el mismo lenguaje:



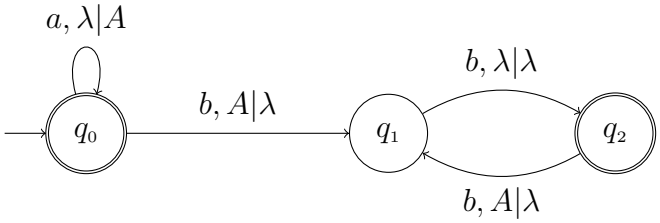
Ejemplo Diseñar un AFPD M que acepte el lenguaje $L = \{a^n b^{2n} : n \geq 0\}$, sobre el alfabeto $\Sigma = \{a, b\}$. L no es un regular y no puede ser aceptado por ningún autómata básico (sin pila).

Solución 1. Se apilan dos Aes por cada a leída en la cinta, y luego se borra una A de la pila por cada b que sea leída sobre la cinta. El alfabeto de pila es $\Gamma = \{A\}$ y el grafo de M se muestra en la página siguiente.

Se utiliza una transición λ de q_1 a q_0 para almacenar una A en la pila sin consumir ningún símbolo en la cinta de entrada. Nótese además que, cuando aparece la primera b el número de Aes en la pila es necesariamente par. Es importante resaltar que este autómata es determinista (AFPD), a pesar de la transición λ que hay entre q_1 y q_0.



Solución 2. Se almacenan todas las $a$es en la pila y luego se borra una A de la pila por cada dos $b$es leídas en la cinta de entrada. El alfabeto de pila es $\Gamma = \{A\}$ y el grafo de M es:



Obsérvese que una cadena aceptada por M necesariamente tiene un número par de $b$es, y tal número es exactamente el doble del número de $a$es.

Nota. En los tres ejemplos anteriores utilizamos $\Gamma = \{A\}$ como alfabeto de pila pero podemos perfectamante hacer $\Gamma = \{a\}$ y apilar $a$es en vez de Aes.

4.1.2. Autómatas con Pila No-Deterministas (AFPN)

Un *Autómata Finito con Pila No-Determinista* (AFPN) consta de los mismos seis componentes de un AFPD, $M = (Q, q_0, F, \Sigma, \Gamma, \Delta)$, pero la función de transición Δ está definida como:

$$\Delta : Q \times (\Sigma \cup \{\lambda\}) \times (\Gamma \cup \{\lambda\}) \longrightarrow \wp(Q \times (\Gamma \cup \{\lambda\})),$$

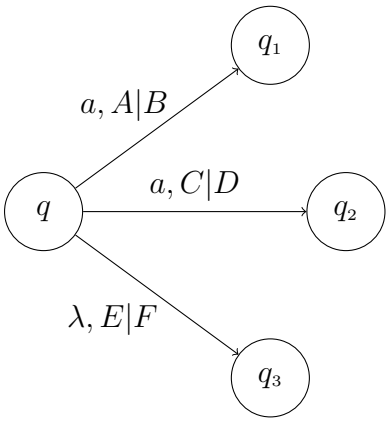
donde $\wp(Q \times (\Gamma \cup \{\lambda\}))$ es el conjunto de subconjuntos de $Q \times (\Gamma \cup \{\lambda\})$. Para $q \in Q$, $a \in \Sigma \cup \{\lambda\}$ y $A \in (\Gamma \cup \{\lambda\})$, $\Delta(q, a, A)$ es de la forma

$$\Delta(q, a, A) = \{(p_1, B_1), (p_2, B_2), \ldots, (p_k, B_k)\},$$

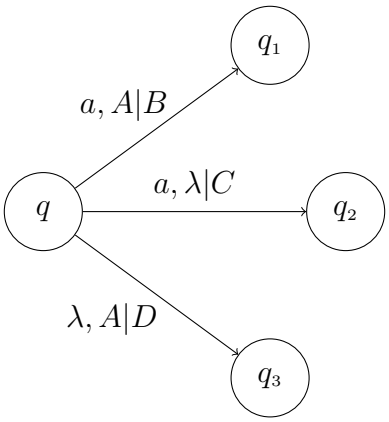
donde $p_1, p_2, \ldots, p_k \in Q$, y $B_1, B_2, \ldots, B_k \in (\Gamma \cup \{\lambda\})$. El significado de esta transición es: cuando la unidad de control escanea el símbolo a sobre la cinta de entrada y el símbolo A en el tope de la pila, la unidad de control puede ejecutar (aleatoriamente) una de las instrucciones (p_k, B_k) $(1 \leq i \leq k)$. A diferencia de lo que sucede con los AFPD, en el modelo

AFPN las instrucciones $\Delta(q, a, A)$ y $\Delta(q, a, \lambda)$ pueden estar definidas simultáneamente, y las transiciones λ, $\Delta(q, \lambda, A)$, no tienen restricción alguna. De esta manera, una cadena de entrada puede tener muchos procesamientos diferentes.

A modo de comparación, el siguiente diagrama muestra algunas transiciones permitidas en los casos determinista (izquierda) y no determinista (derecha).



Modelo determinista (AFPD)



Modelo no determinista (AFPN)

En el modelo AFPN también se permite que $\Delta(q, a, A) = \varnothing$, lo que da lugar a procesamientos abortados que no consumen completamente la entrada.

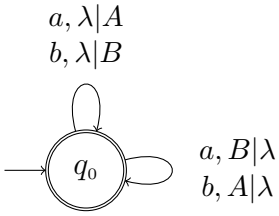
El lenguaje aceptado por un AFPN M se define como:

$$L(M) := \{u \in \Sigma^* : \text{existe un procesamiento } [q_0, u, \lambda] \stackrel{*}{\vdash} [p, \lambda, \lambda],\ p \in F\}.$$

O sea, una cadena u es aceptada si existe por lo menos un procesamiento de u desde la configuración inicial $[q_0, u, \lambda]$ hasta una configuración de aceptación. Para ser aceptada, la entrada u debe ser leída completamente, el procesamiento debe terminar con la pila vacía y la unidad de control en un estado de aceptación.

Ejemplo Diseñar un Autómata Finito con Pila No-Determinista (AFPN) que acepte el lenguaje L de todas las cadenas sobre el alfabeto $\Sigma = \{a, b\}$ que tienen igual número de $a$es que de $b$es. Es decir, $L = \{u \in \Sigma^* : \#_a(u) = \#_b(u)\}$.

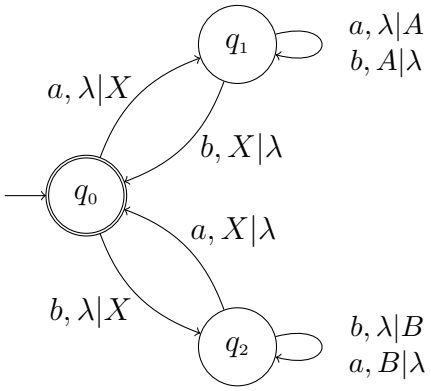
Solución. Para resolver este problema utilizamos el alfabeto de pila $\Gamma = \{A, B\}$. La idea es almacenar en la pila $a$es consecutivas o $b$es consecutivas. Si en el tope de la pila hay una A y el autómata lee una b, se borra la A; similarmente, si en el tope de la pila hay una B y el autómata lee una a, se borra la B. De esta manera, cada a de la entrada se empareja con una b, y viceversa. Para implementar esta idea solamente se requiere un estado:



El autómata M es no-determinista porque en un momento determinado las dos instrucciones $a, \lambda|A$ y $a, B|\lambda$ le otorgan a M dos opciones al leer una a en la cinta de entrada: almacenar A o borrar el tope de la pila B. Algo similar sucede con las dos instrucciones $b, \lambda|B$ y $b, A|\lambda$. De todas maneras, si una cadena tiene igual número de $a$es que de $b$es, existe un procesamiento que consume toda entrada y desocupa la pila. Por otro lado, si una entrada u tiene un número diferente de $a$es que de $b$es, entonces cualquier procesamiento completo de u, consume la entrada sin vaciar la pila.

Ejemplo Diseñar un Autómata Finito con Pila Determinista (AFPD) que acepte el lenguaje L de todas las cadenas sobre el alfabeto $\Sigma = \{a, b\}$ que tienen igual número de $a$es que de $b$es. Es decir, $L = \{u \in \Sigma^* : \#_a(u) = \#_b(u)\}$.

Solución. En el ejemplo anterior se presentó un AFPN que acepta a L, pero es también es posible diseñar un AFPD. Vamos a utilizar un recurso que es útil en muchas situaciones: colocar un marcador de fondo X en la pila al iniciar el procesamiento de una entrada; de esta forma el autómata sabrá, al vaciar la pila en pasos subsiguientes, cuándo se ha llegado al fondo. Para el presente problema el alfabeto de pila es $\Gamma = \{X, A, B\}$; al consumir el primer símbolo de la entrada, ya sea a o b, el autómata coloca X en el fondo.



A continuación ilustramos el funcionamiento de M procesando la entrada *aababbba*, la cual debe ser aceptada porque tiene cuatro $a$es y cuatro $b$es.

$$[q_0, aababbba, \lambda] \vdash [q_1, ababbba, X] \vdash [q_1, babbba, AX] \vdash [q_1, abbba, X] \vdash [q_1, bbba, AX]$$
$$\vdash [q_1, bba, X] \vdash [q_0, ba, \lambda] \vdash [q_2, a, X] \vdash [q_0, \lambda, \lambda].$$

Ejercicios de la sección 4.1

① Diseñar un AFPD que acepte el lenguaje $L = \{a^m b^n : n > m \geq 1\}$, sobre el alfabeto $\Sigma = \{a, b\}$. Utilizando la notación de configuración (o descripción) instantánea, procesar paso a paso las cadenas a^3b^4 (aceptada), ab^5 (aceptada), a^4b^3 (rechazada) y a^3b^3 (rechazada).

② Diseñar un AFPN que acepte el lenguaje $L = \{a^m b^n : m > n \geq 0\}$, sobre el alfabeto $\Sigma = \{a, b\}$. Utilizando la notación de configuración (o descripción) instantánea, presentar procesamientos de aceptación para las entradas a^4b^3 y a^5b. Explicar por qué las cadenas a^3b^4 y a^3b^3 son rechazadas.
NOTA: Con las técnicas de la sección 4.3 se puede demostrar que el lenguaje L no puede ser aceptado por ningún AFPD.

③ Diseñar AFPD que acepten los siguientes lenguajes sobre el alfabeto $\Sigma = \{a, b\}$.

(i) $\{a^n b^{n+1} a : n \geq 1\}$.

(ii) $\{a^n b^m a^{n+1} : n \geq 0, m \geq 1\}$.

(iii) $\{a^n b^k a^m : k \geq 1, m > n \geq 1\}$.

④ Diseñar autómatas con pila, deterministas o no-deterministas, que acepten los siguientes lenguajes sobre el alfabeto $\Sigma = \{a, b, c\}$.

(i) $\{a^k b^m c^n : k, m, n \geq 0, k + m = n\}$.

(ii) $\{a^k b^m c^n : k, m, n \geq 0, k + n = m\}$.

(iii) $\{a^k b^m c^n : k, m, n \geq 0, m + n = k\}$.

(iv) $\{a^k b^m c^n : k, m, n \geq 0, 2n = k + m\}$.

(v) $\{a^k b^m c^n : k, m \geq 0, n \geq 1, n > k + m\}$.

(vi) $\{a^k b^m c^n : k, m \geq 1, n \geq 0, n < k + m\}$.

⑤ Sea $\Sigma = \{a, b\}$. Diseñar autómatas con pila, deterministas o no-deterministas, que acepten los siguientes lenguajes sobre el alfabeto $\Sigma = \{a, b\}$.

(i) $\{a^m b^n : m, n \geq 0, m \neq n\}$.

(ii) $\{u \in \Sigma^* : \#_a(u) > \#_b(u)\}$.

(iii) $\{u \in \Sigma^* : \#_a(u) \geq \#_b(u)\}$.

(iv) $\{u \in \Sigma^* : \#_a(u) \neq \#_b(u)\}$.

(v) $\{u \in \Sigma^* : \#_a(u) = 2\#_b(u)\}$.

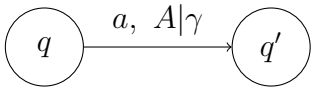
4.2. Inserción de cadenas en la pila

En los autómatas con pila la instrucción básica es $\Delta(q, a, A) = (p, B)$, que incluye varios casos: $a \in \Sigma$, $a = \lambda$, $A, B \in \Gamma$, $A = \lambda$ o $B = \lambda$. Al ejecutar una instrucción básica, el autómata solo puede añadir (a lo sumo) un símbolo a la pila. No obstante, las instrucciones por medio de las cuales se insertan en la pila cadenas de longitud arbitraria se pueden simular con las transiciones básicas y, por consiguiente, se pueden permitir en el diseño de autómatas. Esto se precisa en la siguiente proposición.

4.2.1 Proposición. En los autómatas con pila se permiten instrucciones de los siguientes tipos:

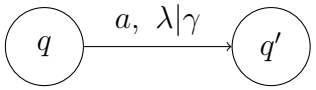
1. $\Delta(q, a, A) = (q', \gamma)$. Instrucción por medio de la cual el tope de la pila A se sustituye por la cadena $\gamma \in \Gamma^*$, o sea, γ sobre-escribe a A. El nuevo tope de la pila pasa a ser el primer símbolo de γ. Aquí se incluyen los casos $a \in \Sigma$ (se consume a en la cinta y) y $a = \lambda$ (transición λ).

$$\Delta(q, a, A) = (q', \gamma)$$



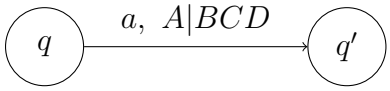
2. $\Delta(q, a, \lambda) = (q', \gamma)$. Instrucción por medio de la cual se inserta en pila la cadena $\gamma \in \Gamma^*$, independientemente del tope actual de la pila (este último se mantiene en la pila). El nuevo tope de la pila pasa a ser el primer símbolo de γ. Aquí se incluyen los casos $a \in \Sigma$ (se consume a en la cinta y) y $a = \lambda$ (transición λ).

$$\Delta(q, a, \lambda) = (q', \gamma)$$

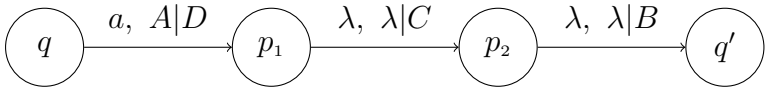


Bosquejo de la demostración. Cada una de las instrucciones $\Delta(q, a, A) = (q', \gamma)$ y $\Delta(q, a, \lambda) = (q', \gamma)$ se puede simular (añadiendo estados auxiliares) por medio de una secuencia de k transiciones básicas, donde k es la longitud de γ. □

Ejemplo La instrucción $\Delta(q, a, A) = (q', BCD)$ consume a en cinta y reemplaza (o sustituye) el tope de la pila A por la cadena de tres símbolos BCD; el nuevo tope de la pila pasa a ser B:



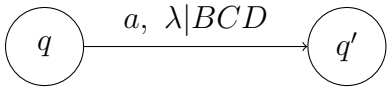
Esta instrucción se puede simular con una secuencia de tres transiciones básicas, como se muestra a continuación:



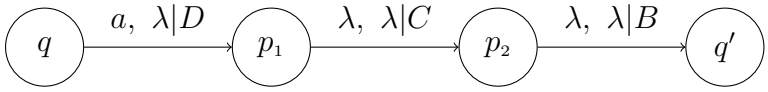
La secuencia es: primero sobre-escribir A por D, luego añadir C a la pila y luego añadir B. En la primera transición se consume la a, y las demás son transiciones λ. Los estados

p_1 y p_2 son estados auxiliares (nuevos) que se introducen con el único propósito de hacer la simulación anterior.

La instrucción $\Delta(q, a, \lambda) = (q', BCD)$ consume a en cinta e inserta la cadena de tres símbolos BCD encima del tope actual de la pila (este último se mantiene en la pila); el nuevo tope de la pila pasa a ser B:



Esta instrucción se puede simular con una secuencia de tres transiciones básicas, como se muestra a continuación:

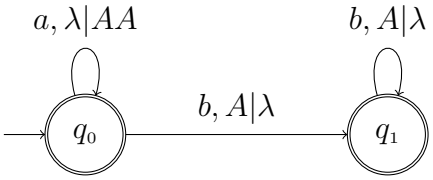


La secuencia es: primero añadir D encima del tope actual de la pila, luego añadir C y luego añadir B. En la primera transición se consume la a, y las demás son transiciones λ.

Las instrucciones de la Proposición 4.2.1 se pueden usar en ejercicios concretos de diseño de autómatas.

Ejemplo Diseñar un AFPD M que acepte el lenguaje $L = \{a^n b^{2n} : n \geq 0\}$, sobre el alfabeto $\Sigma = \{a, b\}$.

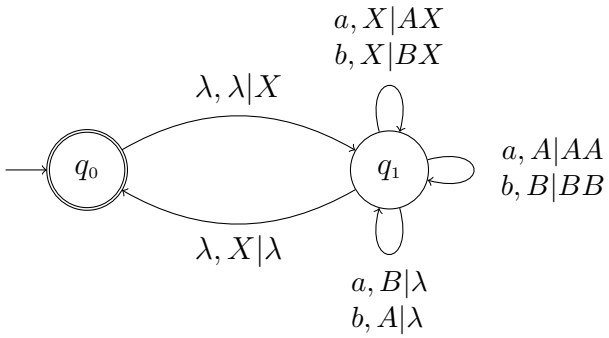
Solución. Se almacenan en la pila dos Aes por cada a leída en la cinta, y luego se borra una A de la pila por cada b que sea leída sobre la cinta.



En la sección 4.1 se utilizó esta misma idea para resolver este problema, pero añadir dos Aes a la pila cada a leída en la cinta, usando transiciones básicas, requiere un estado auxiliar adicional.

Ejemplo En la sección 4.1.2 presentamos dos autómatas con pila, uno determinista y otro no-determinista, para aceptar el lenguaje $L = \{u \in \Sigma^* : \#_a(u) = \#_b(u)\}$ formado por todas las cadenas que tienen igual número de $a$es que de $b$es. A continuación presentamos otro autómata para aceptar este mismo lenguaje. El alfabeto de pila es $\Gamma = \{A, B, X\}$; el símbolo X se utiliza como marcador de fondo.

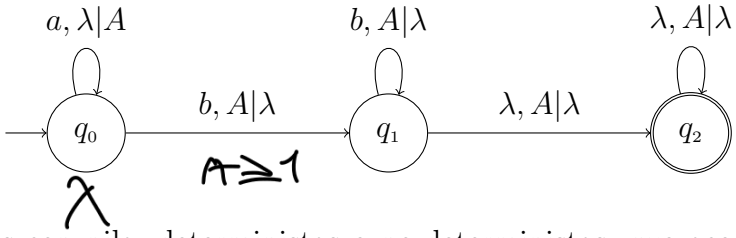
La instrucción $\Delta(q_1, \lambda, X) = (q_0, \lambda)$ es una transición λ que sirve para borrar el marcador de fondo y aceptar, pero introduce una opción no-determinista con respecto a los bucles $\Delta(q_1, a, X) = (q_1, AX)$ y $\Delta(q_1, b, X) = (q_1, BX)$. Por consiguiente, este es un AFPN.



Ejercicios de la sección 4.2

① Sea $\Sigma = \{a, b\}$. Describir explícitamente el lenguaje aceptado por el siguiente autómata con pila:

M_1:



② Diseñar autómatas con pila, deterministas o no-deterministas, que acepten los siguientes lenguajes.

(i) $\{a^{n+1}b^{2n+1} : n \geq 0\}$ sobre el alfabeto $\Sigma = \{a, b\}$.

(ii) $\{a^{2n}b^{3n} : n \geq 0\}$ sobre el alfabeto $\Sigma = \{a, b\}$.

(iii) $\{a^{3n}b^{2n} : n \geq 0\}$ sobre el alfabeto $\Sigma = \{a, b\}$.

(iv) $\{a^k b^m c^n : k, m, n \geq 0, n = 2k + m\}$ sobre el alfabeto $\Sigma = \{a, b, c\}$.

(v) $\{a^k b^m c^n : k, m, n \geq 0, m = k + 2n\}$ sobre el alfabeto $\Sigma = \{a, b, c\}$.

(vi) $\{a^k b^m c^n : k, m, n \geq 0, k = 2m + n\}$ sobre el alfabeto $\Sigma = \{a, b, c\}$.

(vii) $\{a^k b^m c^n : k, m, n \geq 0, 2k = m + n\}$ sobre el alfabeto $\Sigma = \{a, b, c\}$.

(viii) $\{a^m b^n : 0 \leq n \leq 2m\}$ sobre el alfabeto $\Sigma = \{a, b\}$.

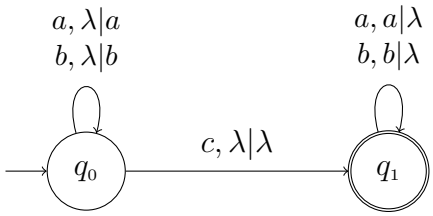
4.3. Los modelos AFPD y AFPN no son equivalentes

En contraste con lo que sucede con los modelos AFD y AFN, los modelos de autómata con pila determinista (AFPD) y no-determinista (AFPN) no resultan ser computacionalmente equivalentes: existen lenguajes aceptados por autómatas AFPN que no pueden ser aceptados por ningún AFPD. Un ejemplo concreto es el lenguaje $L = \{ww^R : w \in \Sigma^*\}$ sobre el alfabeto $\Sigma = \{a, b\}$. Las cadenas de la forma ww^R son palíndromes de longitud par; en efecto,

$$(ww^R)^R = (w^R)^R w^R = ww^R.$$

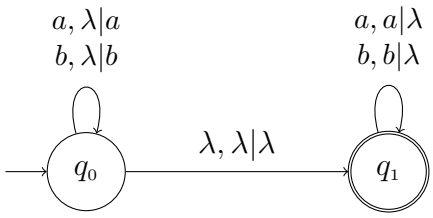
Como se mostrará en la presente sección, se puede construir un autómata con pila no-determinista para aceptar a L, pero no es posible diseñar ningún autómata con pila determinista que lo haga. La demostración de esta imposibilidad se presenta en el Teorema 4.3.1

Ejemplo Primero consideramos el lenguaje $L' = \{wcw^R : w \in \{a, b\}^*\}$ sobre el alfabeto $\Sigma = \{a, b, c\}$. Nótese que las cadenas w y w^R sólo poseen $a$es y/o $b$es. Es muy fácil diseñar un AFPD que acepte el lenguaje L' porque el símbolo c actúa como un separador entre las dos mitades w y w^R. Para aceptar una cadena de la forma wcw^R, la idea es almacenar en la pila los símbolos que preceden a la c central. El contenido de la pila, leído de arriba hacia abajo será precisamente w^R. El autómata consume la c (sin alterar la pila) y luego procesa el resto de la entrada borrando en cada paso el tope de la pila. El alfabeto de pila para este autómata es $\Gamma = \{a, b\}$ y su grafo se exhibe a continuación.



Ejemplo Diseñar un AFPN que acepte el lenguaje $L = \{ww^R : w \in \Sigma^*\}$, donde $\Sigma = \{a, b\}$. Como se señaló arriba, L es exactamente el lenguaje de los palíndromes de longitud par.

Solución. El lenguaje L del presente ejemplo es similar al lenguaje $\{wcw^R : w \in \{a, b\}^*\}$ del ejemplo anterior, excepto que ya no aparece el separador c entre w y w^R. El no-determinismo se utiliza para permitirle al autómata la opción de "adivinar" cuál es la mitad de la cadena de entrada. Si acierta, procederá a comparar el resto de la cadena de entrada con los símbolos almacenados en la pila. Si no acierta entonces o bien, no se consume toda la entrada, o bien, no se vacía la pila. Si la cadena de entrada tiene la forma deseada, entre todos los cómputos posibles estará aquél en el que el autómata adivina correctamente cuándo ha llegado a la mitad de la cadena.



La transición λ de q_0 a q_1 le permite al autómata una opción no-determinista: conjeturar que se ha llegado a la mitad de la cadena de entrada. En este último caso, la unidad de control pasa al estado q_1 y comienza a borrar los símbolos ya almacenados en la pila, que deben coincidir con los que se leen en la cinta.

4.3.1 Proposición. El lenguaje $L = \{ww^R : w \in \Sigma^*\}$, donde $\Sigma = \{a, b\}$, no puede ser aceptado por ningún AFPD (Autómata Finito con Pila Determinista).

Demostración. Razonamiento por contradicción. Se supone que existe un AFPD M tal que $L(M) = L$. Las potencias a^2, a^4, a^6, ... son palíndromes de longitud par y, por tanto, son aceptadas por M. Pero como hay infinitas cadenas de la forma a^{2n}, con $n \geq 0$, y solamente hay un número finito de configuraciones de aceptación (ya que M tiene un número finito de estados), por el Principio del Palomar existen dos cadenas a^{2i} y a^{2j}, con $i \neq j$, cuyo procesamiento termina en la misma configuración de aceptación $[p, \lambda, \lambda]$, con $p \in F$. Esto es, $[q_0, a^{2i}, \lambda] \vdash^* [p, \lambda, \lambda]$ y $[q_0, a^{2j}, \lambda] \vdash^* [p, \lambda, \lambda]$. Como M es determinista, se sigue entonces que, para toda cadena $x \in \Sigma^*$, M leerá las cadenas $a^{2i}x$ y $a^{2j}x$ exactamente de la misma forma ya que $[q_0, a^{2i}x, \lambda] \vdash^* [p, x, \lambda]$ y $[q_0, a^{2j}x, \lambda] \vdash^* [p, x, \lambda]$. Esto implica que, para toda cadena $x \in \Sigma^*$, $a^{2i}x$ y $a^{2j}x$ son ambas aceptadas o ambas rechazadas por M; en otras palabras,

$$(\forall x \in \Sigma^*)[(a^{2i}x \in L \text{ y } a^{2j}x \in L) \text{ ó } (a^{2i}x \notin L \text{ y } a^{2j}x \notin L)].$$

La contradicción surge porque existe una cadena $x \in \Sigma^*$ tal que $a^{2i}x$ es aceptada pero $a^{2j}x$ es rechazada. Por simple inspección observamos que tomando $x = bba^{2i}$ se tendrá $a^{2i}x = a^{2i}bba^{2i} \in L$ pero $a^{2j}x = a^{2j}bba^{2i} \notin L$ (porque $i \neq j$). De esta manera se llega a una contradicción. □

Utilizando la noción de distinguiblidad entre cadenas, el argumento de la Proposición 4.3.1 se puede convertir en una técnica práctica para demostrar que ciertos lenguajes no pueden ser aceptados por autómatas con pila deterministas. Eso se hace en la siguiente proposición, muy similar al criterio de no regularidad (Teorema 3.1.1).

4.3.2 Proposición. Sea Σ un alfabeto dado y L un lenguaje sobre Σ. Si L contiene infinitas cadenas L-distinguibles dos a dos, entonces L no puede ser aceptado por ningún AFPD (Autómata Finito con Pila Determinista).

Demostración. Razonamiento por contradicción. Se supone que existe un AFPD M tal que $L(M) = L$. Por hipótesis existen en L infinitas cadenas u_1, u_2, u_3, ... que son L-distinguibles dos a dos. Estas infinitas cadenas son aceptadas por M; puesto que solamente

hay un número finito de configuraciones de aceptación (ya que M tiene un número finito de estados), por el Principio del Palomar existen dos cadenas u_i y u_j, con $i \neq j$, cuyo procesamiento termina en la misma configuración de aceptación $[p, \lambda, \lambda]$, con $p \in F$. Esto es, $[q_0, u_i, \lambda] \vdash^* [p, \lambda, \lambda]$ y $[q_0, u_j, \lambda] \vdash^* [p, \lambda, \lambda]$. Como M es determinista, se sigue entonces que, para toda cadena $x \in \Sigma^*$, M leerá las cadenas $u_i x$ y $u_j x$ exactamente de la misma forma ya que $[q_0, u_i x, \lambda] \vdash^* [p, x, \lambda]$ y $[q_0, u_j x, \lambda] \vdash^* [p, x, \lambda]$. Esto implica que, para toda cadena $x \in \Sigma^*$, $u_i x$ y $u_j x$ son ambas aceptadas o ambas rechazadas por M; en otras palabras,

$$(\forall x \in \Sigma^*)[(u_i x \in L \text{ y } u_j x \in L) \text{ ó } (u_i x \notin L \text{ y } u_j x \notin L)].$$

Esto contradice que u_i y u_j son L-distinguibles. □

Ejemplo Utilizar la Proposición 4.3.2 para demostrar que el lenguaje L de todos los palíndromes de longitud par sobre $\Sigma = \{a, b\}$, o sea, $L = \{ww^R : w \in \Sigma^*\}$, no puede ser aceptado por ningún AFPD (Autómata Finito con Pila Determinista).

Solución. Las infinitas cadenas $a^2, a^4, a^6, \ldots$ son palíndromes de longitud par y, por tanto, pertenecen a L. Vamos a comprobar que son L-distinguibles dos a dos. Sean $i, j \geq 1$, con $i \neq j$. Queremos mostrar que a^{2i} y a^{2j} son L-distinguibles, para lo cual hay que encontrar una cadena x tal que $a^{2i}x \in L$ pero $a^{2j}x \notin L$ (o viceversa). La escogencia más sencilla de x es $x = bba^{2i}$. Se tiene que $a^{2i}x = a^{2i}bba^{2i} \in L$ pero $a^{2j}x = a^{2j}bba^{2i} \notin L$ (porque $i \neq j$). Esto muestra que a^{2i} y a^{2j} son L-distinguibles si $i \neq j$ y, por la Proposición 4.3.2, L no puede ser aceptado por ningún AFPD.

Ejercicios de la sección 4.3

① Sea $\Sigma = \{a, b\}$. Diseñar autómatas con pila no-deterministas que acepten los siguientes lenguajes.

(i) El lenguaje de todos los palíndromes.

(ii) El lenguaje de todos los palíndromes de longitud impar.

(iii) $\{a^m b^n : m > n \geq 0\}$.

(iv) $\{a^m b^n : m, n \geq 0, m \neq n\}$.

(v) $\{u \in \Sigma^* : \#_a(u) > \#_b(u) \geq 0\}$.

(vi) $a^* \cup \{a^n b^n : n \geq 0\}$.

(vii) $\{a^n b^n : n \geq 0\} \cup \{a^n b^{2n} : n \geq 0\}$.

(viii) $\{a^{2n} b^n : n \geq 0\} \cup \{a^n b^{2n} : n \geq 0\}$.

② Demostrar que ninguno de los lenguajes del problema ① puede ser aceptado por un AFPD (Autómata Finito con Pila Determinista).

③ Sea $\Sigma = \{a, b, c\}$. Diseñar un autómata con pila no-determinista que acepte el lenguaje $L = \{a^k b^m c^n : k, m, n \geq 0, k = m \text{ o } k = n\}$. Demostrar que L no puede ser aceptado por ningún AFPD (Autómata Finito con Pila Determinista).

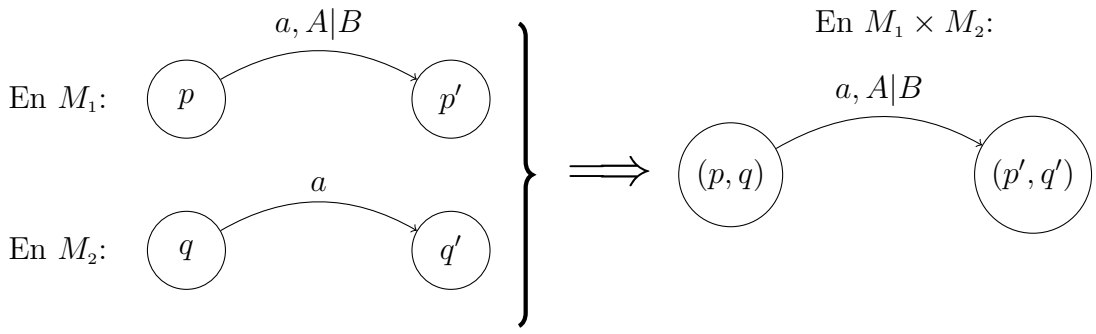
4.4. Producto cartesiano de un autómata con pila y y un AFD

Sea $M_1 = (Q_1, q_1, F_1, \Sigma, \Gamma, \Delta)$ un autómata con pila, ya sea AFPD o AFPN, y sea $M_2 = (\Sigma, Q_2, q_2, F_2, \delta)$ un AFD. Los dos autómatas tiene el mismo alfabeto de entrada Σ. El Producto Cartesiano $M_1 \times M_2$ es un autómata con pila definido como

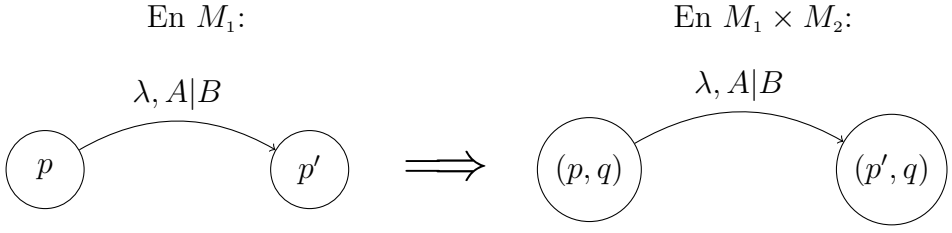
$$M_1 \times M_2 = (Q_1 \times Q_2, (q_1, q_2), F_1 \times F_2, \Sigma, \Gamma, \Delta')$$

donde la función de transición Δ' consta de las siguientes instrucciones:

(1) Si $(p', B) \in \Delta(p, a, A)$ y $\delta(q, a) = q'$, entonces $((p', q'), B) \in \Delta'((p, q), a, A)$; aquí se tienen en cuenta todas las transiciones de M_1, con $a \in \Sigma$, $A, B \in \Gamma \cup \{\lambda\}$. Considerando los grafos de los autómatas, las anteriores instrucciones se pueden presentar esquemáticamente de la siguiente forma:



(2) Una transición λ de p a p' en M_1 da lugar a la misma transición λ de (p, q) a (p', q) en $M_1 \times M_2$, para todos los estados q de M_2. De esta forma, una transición λ en M_1 origina k transiciones λ en $M_1 \times M_2$, donde k es el número de estados de M_2. Esquemáticamente,



En términos de las funciones de transición, se tiene que si $(p', B) \in \Delta(p, \lambda, A)$ entonces $((p', q), B) \in \Delta'((p, q), \lambda, A)$ para todos los estados q de M_2, y considerando $A, B \in \Gamma \cup \{\lambda\}$.

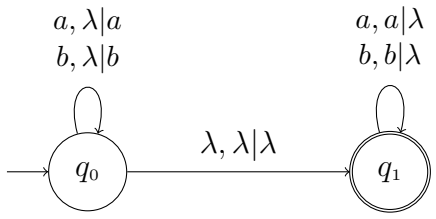
Al examinar detenidamente la anterior construcción, se observa que el producto cartesiano $M_1 \times M_2$ simula los autómatas M_1 y M_2 "en paralelo": con las primeras componentes de $M_1 \times M_2$ se simula el funcionamiento de M_1 mientras que con las segundas componentes se simula a M_2. Un símbolo de $a \in \Sigma$ se consume solamente si tanto M_1 como M_2 lo consumen simultáneamente, y un estado (p, q) es de aceptación si tanto p como q son estados de aceptación. Por este argumento, el siguiente teorema resulta intuitivamente claro.

4.4.1 Teorema. Sea $M_1 = (Q_1, q_1, F_1, \Sigma, \Gamma, \Delta)$ un autómata con pila (AFPD o AFPN) tal que $L(M_1) = L_1$ y sea $M_2 = (\Sigma, Q_2, q_2, F_2, \delta)$ un AFD tal que $L(M_2) = L_2$. El autómata con pila $M_1 \times M_2$ definido arriba satisface $L(M_1 \times M_2) = L_1 \cap L_2$.

Ejemplo Utilizar el Teorema 4.4.1 para construir un autómata con pila que acepte el lenguaje L de todos los palíndromes de longitud par que no tienen dos $b$es consecutivas, sobre el alfabeto $\Sigma = \{a, b\}$.

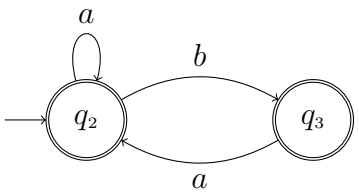
Solución. El lenguaje L se puede escribir como $L = L_1 \cap L_2$ donde L_1 es el lenguaje de todos los palíndromes de longitud par y L_2 es el lenguaje de las cadenas no tienen dos $b$es consecutivas. Para aceptar a L_1 tenemos el AFPN M_1 diseñado en la sección anterior:

M_1:

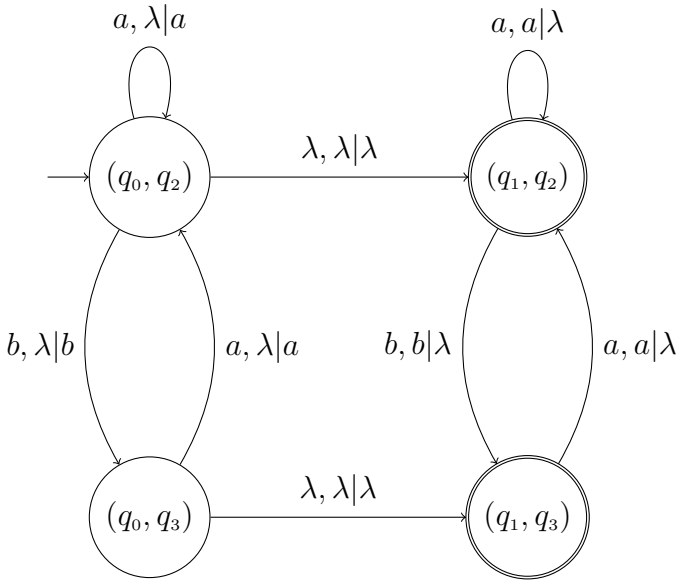


Para aceptar a L_2 tenemos el siguiente AFD M_2. Se ha omitido el estado limbo porque solamente se necesitan las trayectorias de aceptación si se desea aceptar $L_1 \cap L_2$.

M_2:



Al formar el producto cartesiano $M_1 \times M_2$ se obtiene el AFPN exhibido en la página siguiente. La transición λ en M_1 da lugar a dos transiciones λ en $M_1 \times M_2$, manteniendo fijas las segundas componentes. Al examinar el autómata $M_1 \times M_2$ se observa que los estados de la izquierda se utilizan para almacenar símbolos en la pila pero impidiendo que aparezcan dos $b$es consecutivas. Los estados de la derecha se utilizan para desocupar la pila impidiendo también dos $b$es consecutivas.



Ejercicios de la sección 4.4

Utilizando la técnica del producto cartesiano de un AFPN con un AFD construir autómatas con pila que acepten los siguientes lenguajes, definidos sobre el alfabeto $\Sigma = \{0, 1\}$.

① El lenguaje de todos los palíndromes que no terminan en 01.

② El lenguaje de todos los palíndromes que no contienen la subcadena 101.

③ El lenguaje de todas las cadenas de longitud impar que tienen más ceros que unos.

④ $L = \{u \in \Sigma^* : \#_0(u) = \#_1(u) = \text{impar}\}$.

⑤ $L = \{u \in \Sigma^* : \#_0(u) > \#_1(u) \text{ y } \#_0(u) \text{ es impar}\}$.

⑥ $L = \{u \in \Sigma^* : \#_0(u) > \#_1(u) \text{ y } \#_1(u) \text{ es impar}\}$.