Lenguajes y Máquinas

Silvia Takahashi

8 de octubre de 2012

Capítulo 1

Máquinas de Turing

En este capítulo se presentan las máquinas de Turing. Estas máquinas son más poderosas computacionalmente que los autómatas de pila. A primera vista una máquina de Turing se parece a los autómatas vistos en los capítulos anteriores. Al igual que los autómatas de estados finitos, y los autómatas de Pila, las máquinas de Turing tienen un número finito de estados y una cinta de donde pueden leer. No tienen una pila, pero pueden escribir sobre la misma cinta que leen. En cada instante, el autómata, dependiendo del estado en el que está y del símbolo que está leyendo, pasa a otro estado y puede escribir un símbolo en la cinta y moverse a la izquierda o a la derecha en la cinta. El poder escribir y devolverse es lo que le agrega poder computacional a las máquinas de Turing.

1.1. Defincione̊s

La descripción formal de una Máquina de Turing se enuncia a continucación:

Definición 1. *Una máquina de Turing es una cuádrupla:* (Q, Σ, q_I, δ) *donde:*

- Q *es un conjunto finito de estados,*
- Σ *es un alfabeto finito que incluye el símbolo* # *que representa el blanco.*
- $q_I \in Q$ *es el estado inicial*
- $\delta : (Q \times \Sigma) \to (Q \cup \{\bigcirc\} \times (\Sigma \times (\rightarrow, \leftarrow))$ *es una función transición.*

La función de transición de estados sirve para indicar el comportamiento de la máquina al estar en un estado dado y leyendo un símbolo en la cinta de entrada. Esta es una función parcial. Esto quiere decir, que puede haber parejas (estado, símbolo) para las cuales la función no está definida.

El comportamiento de la máquina está dado por la función de transición:

- Si $\delta(q, \sigma) = (q_1, \rho, \rightarrow)$, esto quiere decir que que si está en un estado q, leyendo σ, escriba ρ, mueva la cabeza lectora a la derecha y pase al estado q_1.
- Si $\delta(q, \sigma) = (q_1, \rho, \leftarrow)$, esto quiere decir que que si está en un estado q, leyendo σ, escriba ρ, mueva la cabeza lectora a la izquierda y pase al estado q_1.
- Si $\delta(q, \sigma) = (\bigcirc, \rho, \rightarrow)$, esto quiere decir que que si está en un estado q, leyendo σ, escriba ρ, mueva la cabeza lectora a la derecha y detenga la ejecución.
- Si $\delta(q, \sigma) = (\bigcirc, \rho, \leftarrow)$, esto quiere decir que que si está en un estado q, leyendo σ, escriba ρ, mueva la cabeza lectora a la izquierda y detenga la ejecución.

La ejecución termina normalmente, si se ejecutan cualquiera de las dos últimas instrucciones descritas arriba. También puede detenerse si para algún estado y algún símbolo, no está especificado qué acción se debe tomar. La ejecución también termina si la cabeza lectora se sale por el extremo izquierdo de la cinta, y en este caso terminaría en error.

La configuración de una máquina de Turing está dada por el estado en que se ecuentra, el contenido de la cinta de entrada, y la posición de la cabeza lectora sobre la cinta. Así la cinta de entrada sea infinita, nos interesa sólo la parte de la cinta que no está compuesta sólo de blancos. Para los problemas que vamos a ver, la porción de la cinta que no está en blanco es finita. Por lo tanto, podemos decir que la cadena sobre la cinta es de la forma: $\omega\#^+$.

Podemos definir formalmente la configuración de una máquina de Turing así:

Definición 2. *Dada una Máquina de Turing $M = (Q, \Sigma, q_I, \delta)$, una* **configuración** *de M es una tripla : $\langle q, \omega, p\rangle \in (Q \cup \{\bigcirc\} \times \Sigma * \times \mathbb{N}$. Esto indica que la M está en el estado q. que la cinta lectora tiene una cadena de la forma $\omega\#^+$ y que la cabeza lectora está en la posición p de la cinta.*

Por ejemplo: suponga que: $\omega = \sigma_1 \ldots \sigma_{n-1}\sigma_n$, donde $\sigma_n = \#$; estamos leyendo σ_i; y estamos en el estado q. Podemos representar la configuración así: $(q, \sigma_1 \ldots \underline{\sigma_i} \ldots \sigma_{n-1}\sigma_n)$, subrayando el símbolo que se está leyendo. Si se está en una configuración en que el estado es $\bigcirc$, se sabrá que el cálculo ha terminado sin problemas. Si estamos en una configuración: $(q, \sigma_1 \ldots \underline{\sigma_i} \ldots \sigma_{n-1}\sigma_n)$, y $\delta(q, \sigma_i)$ no está definido, la ejecución está detenida.

Definición 3. *Decimos que una configuración K es alcanzable en un paso de una configuración C, y escribimos $C \Rightarrow K$ si:*

- $C = (q, \sigma_1 \ldots \underline{\sigma_i} \ldots \sigma_{n-1}\sigma_n)$ *(con $i < n$), $\delta(q, \sigma i) = (q_1, (\rho, \rightarrow)$ y $K = (q, \sigma_1 \ldots \rho\underline{\sigma_{i+1}} \ldots \sigma_{n-1}\sigma_n)$.*
- *Si la máquina de Turing está en una configuración: $(q, \sigma_1 \ldots \underline{\sigma_i} \ldots \sigma_{n-1}\sigma_n)$, (con $1 < i$), si $\delta(q, \sigma i) = (q_1, (\rho, \leftarrow)$ entonces pasaría a la configuración: $(q, \sigma_1 \ldots \underline{\sigma_{i-1}}\rho \ldots \sigma_{n-1}\sigma_n)$.*
- *Si la máquina de Turing está en una configuración: $(q, \sigma_1 \ldots \sigma_i \ldots \sigma_{n-1}\underline{\sigma_n})$, y si $\delta(q, \sigma i) = (q_1, (\rho, \rightarrow)$ entonces pasaría a la configuración: $(q, \sigma_1 \ldots \sigma_i \ldots \rho\underline{\#})$.*
- *Si la máquina de Turing está en una configuración: $(q, \underline{\sigma_1} \ldots \sigma_n)$, si $\delta(q, \sigma i) = (q_1, (\rho, \leftarrow)$ entonces la ejecución terminaría abruptamente pues la cabeza lectora no puede moverse a lizquierda a partir de esa posición.*

Si podemos pasar de una configuración C_1 a otraC_n en un paso, ecribimos $C_1 \Rightarrow C_n$. Si podemos pasar de de una configuración C_1 a otraC_2 en cero o más pasos escribimos: $C_1 \Rightarrow^* C_n$

Definición 4. *Decimos que una configuración C_n es alcanzable en cero o más pasos a partir de C_1 si:*

- $C_1 = C_n$
- $C_1 \Rightarrow C_n$
- *Existe una configuración C tal que $C_1 \Rightarrow C$ y $C \Rightarrow^* C_n$.*

1.2. Extensiones a la definición

Es útil extender la definición de máquinas de Turing para que pueda mover la cabeza sin leer y que pueda leer sin avanzar o retroceder. También, sería útil poder hacer que ignore la cinta al leer o escribir. Esto no le agrega poder computacional pero puede hacer que sea más fácil describir ciertos cálculos.

1.2.1. Operación Nula

Comenzamos extendiendo las Máquinas de Turing, agregando la acción de no mover la cabeza lectora. Si hacemos esto, la función de transición tendría la siguiente forma:

$$\delta : (Q \times \Sigma) \to (Q \cup \{\bigcirc\} \times (\Sigma \times (\rightarrow, \leftarrow, \leftrightarrow))$$

Esto no agrega poder computacional. Podemos ver que si tenemos un máquina de Turing con la operación $\leftrightarrow$, es fácil construir una máquina de Turing con el mismo comporamiento, que no tiene esta operación.

Teorema 5. *Toda máquina de Turing extendida con la operación $\leftrightarrow$ puede convertirse a una máquina de Turing equivalente que no tiene esta operación.*

Considere una máquina de Turing con la operación $\leftrightarrow$

$$T = (Q, \Sigma, q_I, \delta)$$

Podemos construir una máquina de Turing equivalente que no contiene esta operación así:

$$T' = (Q', \Sigma, q_I, \delta')$$

Sea $Q' = Q \cup q_p : p \in Q$
Suponemos que $Q \cap Q' = \emptyset$
Definimos δ' así:

- *Las transiciones que no tienen $\leftrightarrow$ no cambian: $\delta'(q, \sigma) = (p, \rho, op)$ si $\delta(q, \sigma) = (p, \rho, op)$ y $op \neq \leftrightarrow$*
- *Las transiciones tienen NOP, se cambian por una transición al estado q_p moviéndose a la derecha: $\delta'(q, \sigma) = (q_p, \rho, \rightarrow)$ si $\delta(q, \sigma) = (p, \rho, \leftrightarrow)$.*
- *Agregamos transiciones para cada q_p para que pase al estado p moviendose a la izquierda con cualquier símbolo y volviendo a escribir el mismo, para así no cambiar la cinta. $\delta'(q_p, \rho) = (p, \rho, \leftarrow)$ para cada $p \in Q$ y cada $\rho in \Sigma$*

1.2.2. Ignorar la cinta

Podemos también extender la máquina para ignorar la cinta. Es decir, para que ejecute la acción independientemente de lo que hay en la cinta y para que no escriba nada en la cinta.

LA función de transición entonces quedaría así:

$$\delta : (Q \times (\Sigma \cup \{\lambda\}) \to (Q \cup \{\bigcirc\} \times (((\Sigma \cup \{\lambda\}) \times (\rightarrow, \leftarrow, \leftrightarrow))$$

Donde
$\delta(q, \lambda) = \ldots$ indica que para cualquier símbolo del alfabeto se ejecute la acción.
$\delta(q, c) = (\lambda \ldots)$ que no se escribe nada y se deja tal como está.
Es fácil ver que estas transiciones.λ pyuden reemplazarse por una transición por cada símbolo del alfabeto para quedar con una máquina sin transiciones $|lambda$.

Sin embargo estas transiciones-λ podrían agregar no determinismo si tenemos una mezcla de transiciones-λ con transiciones normales saliendo de un mismo estado. Lo cual en sí no es problema, ya que las Máquinas de Turing no-determinísticas tienen el mismo poder computacional que las determinísticas, Sin ambargo, podría ser problemático para su simulación. Si queremos evitar esto se debe garantizar que:

- Si existe una transición que salga de un estado q, que se active al leer λ, no debe existir ninguna otra que se active con algun $\sigma \in \Sigma$
- Si existe una transición que salga de un estado q, y que escriba λ, no debe existir una que escriba otra cosa.

Teorema 6. *Si tenemos una Máquina de Turing con transiciones que se ejecutan al leer λ (es decir, ignorando la cinta) y que permiten escribir λ (es decir, no escribir nada), podemos construir una máquina de Turing equivalente que no lee λ ni escribe λ.*

Lo que queremos hacer, entonces, es cambiar las transiciones que tienen λ por transiciones equivalentes que no tienen λ.

- *Cambiamos las transiciones que leen σ (un símbolo distinto de λ) y no escriben nada (transiciones de la forma: $\delta(q, \sigma) = (p, \lambda, op)$) por la transición: $\delta(q, \sigma) = (q, \sigma)$, una transición en la que se escribe lo mismo que se lee.*

- *Cambiamos las transiciones que ignora la cinta y escribe ρ (un símbolo distinto de λ) - transiciones de la forma: $\delta(q, \lambda) = (p, \rho, op)$ por transiciones: $\delta(q, \sigma) = (p, \rho, op)$ (una por cada $\sigma \in \Sigma$).*

- *Cambiamos las transiciones que ni leen y escriben (transiciones de la forma: $\delta(q, \lambda) = (p, \lambda, op)$) por transiciones: $\delta(q, \sigma) = (p, \sigma, op)$, una por cada $\sigma en \Sigma$*

Formalmente: $\delta' = \delta$
$\{((q, \lambda), (p, \lambda, op))\} \cup \{((q, \sigma), (p, \sigma, op)) : \sigma \in \Sigma, \}$

1.3. Representación Gráfica

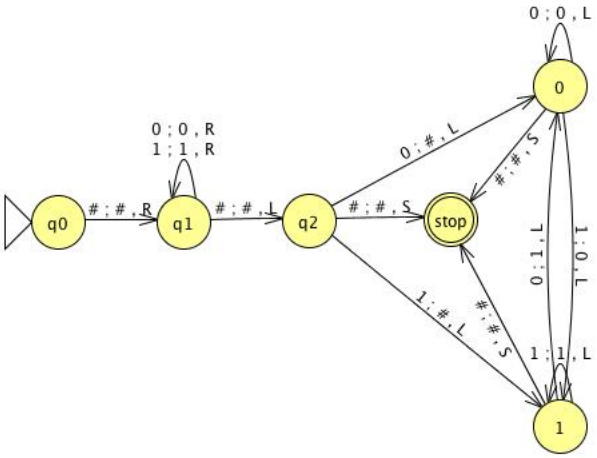
Graficamente, podemos describir una máquina de Turing con un multigrafo donde los nodos representan estados y los arcos las transiciones. Los arcos se etiquetan con el símbolo que se lee seguido del símbolo / seguido de una pareja $Instruccion(\sigma)$, donde instrucción puede ser $\Rightarrow$, $\Leftarrow$ o $\leftrightarrow$.

Si tenemos que $\delta(q, \sigma) = (p, \rho, act)$, esto se dibujaría así:

1.4. Ejemplos

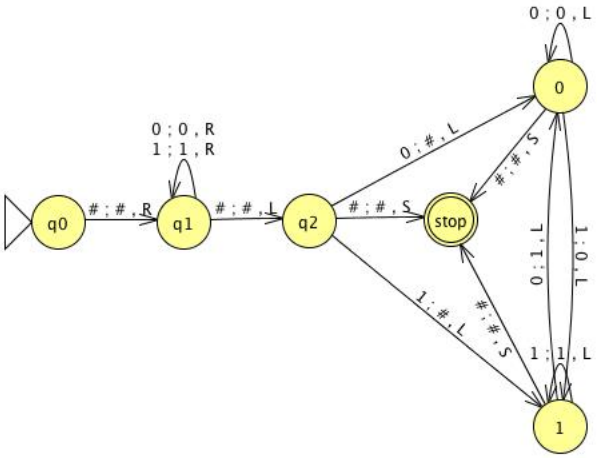
1.4.1. Shift Left

Problema: La máquina comienza en una configuración de la forma: $\underline{\#}\sigma_1\sigma_2 \ldots \sigma_n$ y termina en una configuaración: $\underline{\#}\sigma_2 \ldots \sigma_n$. Para facilidad suponga que $\sigma_i \in \{0, 1\}$ para $0 \leq i \leq n$. Tenemos entonces:



1.4.2. Invertir una cadena

Problema: La máquina comienza en una configuración de la forma: $\underline{\#}\sigma_1\sigma_2\ldots\sigma_n$ y termina en una configuaración: $\underline{\#}\sigma_n\ldots\sigma_1$. Donde tanto al principio como al final, la cabeza se encuentra al principio de la cinta. Para facilidad suponga que $\sigma_i \in \{0,1\}$ para $0 \leq i \leq n$. Tenemos entonces:



1.4.3. $a^n b^n c^n$

En el Capítulo 2, vimos que este lenguaje no es independiente del contexto. No podemos construir un autómata de pila que lo reconoza.

Vamos a suponer que comenzamos en una configuración $\underline{\#}\omega$ y terminamos en una configuración en la que hemos escrito Y si la cadena está en el lenguaje y N de lo contrario.

beginexample Considere el lenguaje $\{a^n b^n c^n : n \leq 0\}$. Este es un lenguaje que no es independiente del contexto (ni regular). Para reconocer este lenguaje debemos leer una a y aseguranos de que haya tanto una b como una c por esa a.

Para facilidad, vamos a suponer que comenzamos en una configuración que tiene la siguiente forma:

$$\underline{\#}\omega\#^+$$

Queremos una Máquina de Turing que comience en esta configuración y si

$$\omega \in \{a, b, c\}*$$

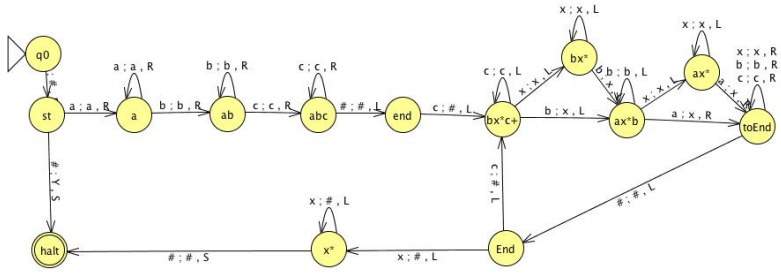
y

$$\omega = a^n b^n c^n$$

entonces la Máquina debe terminar en una configuración:

$\underline{Y}$

sin importar qué está a la derecha o a la izquierda de Y.



A continuación explicamos el comportamiento de la máquina.

1.4.4. Dividir por 4

Al igual que en los autómatas podemos definir las máquinas usando fórmulas en lugar de dibujos.

Suponga que queremos definir una máquina de Turing que comienza en el extremo izquierdo de la cinta donde aparece una cadena de dígitos y queremos escribir el resultado de dividir el número que aparece en la cinta por cuatro (ver el ejemplo del trasductor del capítulo 1), esta máquina se describe así:

- $Q = \{0, 1, 2, 3, R_1, R_2, R_3\}$
- $\Sigma = \{0, 1, 2, 3, R, \#\}$
- $q_I = 0$
- La función de transición la escribimos así:
  - $\delta(s, d) = ((10 \cdot s) + d) mod 4, (((10 \cdot s) + d) \div 4), \rightarrow)$, para $d \in 1, 2, 3, 4$
  - $\delta(0, \#) = (\bigcirc, (\#, \leftrightarrow))$
  - $\delta(d, \#) = (R_d, (R, \Rightarrow))$ para $d \neq 0$
  - $\delta(R_d, \#) = (\bigcirc, (d, \Rightarrow))$

Comenzamos leyendo el número de derecha a izquierda. Tenemos 4 estados para esta fase que indican cuanto se lleva de la división anterior.

Para la fase final tenemos 3 estados para escribir el residuo.

Esta sería la traza de la ejecución de la máquina con el número: 12345

1.4.5. Otras extensiones

Existen otras extensiones de Máquinas de Turing; ninguna de las cuales agrega poder computacional. Simplemente, agregan poder de expresión.

- Máquinas con una cinta infinita en ambas direcciones
- Máquinas no determinísiticas
- Máquinas con dos cintas
- Máquinas con más de dos cintas (pero el número de cintas debe ser predeterminado)

1.5. Mejorando la notación

Buscamos mejorar la notación para que la descripción de las máquinas no sea tan engorrosa. Para esto necesitamos dos conceptos: máquinas atómicas sencillas que ejecutan procesos simples y combinación de máquinas.

1.5.1. Máquinas Sencillas

En esta sección se definen unas máquinas de Turing que ejecutan acciones simples. Estas luego pueden combinarse para realizar tareas más complejas.

Moverse a la derecha

$$R = (\{q\}, \Sigma, q, \delta_R)$$

Donde

$$\delta_R(q, \lambda) = (\bigcirc, \lambda, \Rightarrow)$$

1.5.2. Moverse a la izquierda

$$L = (\{q\}, \Sigma, q, \delta_L)$$

Donde

$$\delta_L(q, \lambda) = (\bigcirc, \lambda, \Leftarrow)$$

1.5.3. Escribir un símbolo: σ

$$W_\sigma = (\{q\}, \Sigma, q, \delta_W)$$

Donde

$$\delta_{W_\sigma}(q, \lambda) = (\bigcirc, \sigma, \Leftrightarrow)$$

1.5.4. Moverse a la derecha hasta encontrar un símbolo dado, σ

Esta máquina hace las siguiente secuencia de operaciones:

1. Se mueve a la derecha
2. Si está leyendo el símbolo σ, se detiene; de lo contrario, vuelve al paso 1

$$R_{=\sigma} = (\{q, p\}, \Sigma, q, \delta_R)$$

Donde

$$\delta_{R_{=\sigma}}(q, \lambda) = (p, \lambda, R)$$

$$\delta_{R_{=\sigma}}(p, \sigma) = (\bigcirc, \sigma, NOP)$$

Para $\rho \neq \sigma$:

$$\delta_{R_{=\sigma}}(p, \rho) = (p, \rho, \Rightarrow)$$

1.5.5. Moverse a la izquierda hasta encontrar un símbolo, σ

Esta máquina hace las siguiente secuencia de operaciones:

1. Se mueve a la izquierda
2. Si está leyendo el símbolo σ, para de lo contrario vuelve al paso 1

$$L_{=\sigma} = (\{q, p\}, \Sigma, q, \delta_L)$$

Donde

$$\delta_{L_{=\sigma}}(q, \lambda) = (p, \lambda, R)$$
$$\delta_{L_{=\sigma}}(p, \sigma) = (\bigcirc, \sigma, NOP)$$

Para $\rho \neq \sigma$:

$$\delta_{L_{=\sigma}}(p, \rho) = (p, \rho, \Leftarrow)$$

1.5.6. Moverse a la derecha hasta no estar leyendo un símbolo dado, σ

Esta máquina hace las siguiente secuencia de operaciones:

1. Se mueve a la derecha
2. Si está no está leyendo el símbolo σ, para de lo contrario vuelve al paso 1

$$R_{\neq\sigma} = (\{q, p\}, \Sigma, q, \delta_{R_{\neq\sigma}})$$

Donde

$$\delta_{R_{\neq\sigma}}(q, \lambda) = (p, \lambda, R)$$
$$\delta_{R_{\neq\sigma}}(p, \sigma) = (p, \sigma, NOP)$$

Para $\rho \neq \sigma$:

$$\delta_{R_{\neq\sigma}}(p, \rho) = (\bigcirc, \rho, \Rightarrow)$$

1.5.7. Moverse a la derecha hasta no estar leyendo un símbolo dado, σ

Esta máquina hace las siguiente secuencia de operaciones:

1. Se mueve a la derecha
2. Si está no está leyendo el símbolo σ, para de lo contrario vuelve al paso 1

$$R_{\neq\sigma} = (\{q, p\}, \Sigma, q, \delta_{R_{\neq\sigma}})$$

Donde

$$\delta_{R_{\neq\sigma}}(q, \lambda) = (p, \lambda, R)$$
$$\delta_{R_{\neq\sigma}}(p, \sigma) = (p, \sigma, NOP)$$

Para $\rho \neq \sigma$:

$$\delta_{R_{\neq\sigma}}(p, \rho) = (\bigcirc, \rho, \Rightarrow)$$

1.5.8. Combinación de máquinas

Las máquinas de Turing pueden combinarse agegando una operación que es una llamda a otra máquina.

1.5.9. Una nueva notación

La notación para describir máquinas de Turing puede resultar un poco engorrosa. En los ejemplos mostrados arriba, veíamos que acciones como buscar un símbolo, requieren muchos estados. Muchos autores, introducen una notación en la que priman las operaciones. Se tienen algunas operaciones básicas, e inclusive se pueden definir máquinas más complejas y conectarlas. En este caso, los nodos son operaciones y los arcos se etiquetan con símbolos que permiten el paso de una operación a otra. Esto en realidad, lo que está haciendo mostrar en un diagrama la forma en que se combinan las máquinas simples que se describieron arriba.

Estas máqunas representan las sigientes operaciones básicas.

- Moverse a la derecha sin escribir. Es lo mismo que tener la siguiente transición.
- Moverse a la izquierda sin escribir
- Moverse a la derecha hasta encontrar un símbolo dado. Note que se mueve a la derecha. Si es el símbolo dado entonces ya termina. se vuelve a mover a la derecha.
- Moverse a la derecha hasta que no esté leyendo un símbolo dado
- Moverse a la izquierda hasta encontrar un símbolo dado
- Moverse a la izquierda hasta que no esté leyendo un símbolo dado
- Escribir un símbolo

Ejemplo 1. *Digamos que queremos verificar si se está leyendo una cadena de la forma : $\#a^{2n}$. paran > 0*

Primero vamos a verificar que la cadena esté formada sólo con a's, antes de comenzar el proceso. Luego volvemos al principio de la cinta.

De haber al menos 1 a. Entonces comenzamos marcando esa a, cambiándola por X. Estamos entonces en esta situacion: Xa^m donde m+1 es de la forma $2n$.

Lo que hacemos es ver si se pueden multplicar las X por 2. Es decir, pasar de esta configuarción a XXa^m y luego $XXXXa^m$, y así sucesivamente.

Estamos entonces replicando la secuencia de X's siempre y cuando haya a´s.

Para esto hacemos lo siguiente: marcamos la X con una M y avanzamos buscando una a. Si la encontramos la marcamos con una H. Si encontramos blanco, hay un error. Al buscar la a, pueden también aparecer a´s marcadas con H. Sabemos que hemos terminado de duplicar las X cuando encontramos una H en lugar de una X. En ese caso tenemos que cambiar las H, por X. Si después de la última H, hay una z es por que debemos seguir multiplicando por dos.

Ejemplo 2. *Este se habría podido hacer más fácilmente combinando máquinas así:*

1.6. Equivalencias

En esta sección veremos cómo simular los autómatas vistos en capítulos anteriores con máquinas de Turning.

1.6.1. Autómata Finito

Digamos que tenemos un autómata finito determinístico definido así:

$$M = (Q, \Sigma, q_I, F, \delta_M)$$

donde $\{\#, \boxtimes, \boxtimes\} \not\subset \Sigma$ y $\bigcirc \notin Q$. Entonces la máquina de Turing defininida así:

$$T = (Q \cup \{\bigcirc\}, \Sigma \cup \{\#, \boxtimes, \boxtimes\}, q_I, \delta_T)$$

donde:

- $\delta_T(q, \sigma) = (\delta(q, \sigma), (\sigma, \rightarrow))$ para $\sigma \in \Sigma$
- $\delta_T(q, \#) = (\bigcirc, \boxtimes)$ para $q \in F$
- $\delta_T(q, \#) = (\bigcirc, \boxtimes)$ para $q \notin F$

simula el comportamiento del autómata M. Terminando correctamente escribiendo ☑ si la cadena es aceptada, y escribiendo ⊠ de lo contrario.

Sea $\omega = \sigma_1 \dots \sigma_n \boxtimes$, Si T comienza en una configuración $(q_I, \underline{\sigma_1} \dots \sigma_n)$:

- Si $\omega \in L(M)$, entonces la máquina de Turing terminaría en la siguiente configuración: $(\bigcirc, \sigma_1 \dots \sigma_n \underline{\boxtimes}$
- Si $\omega \in L(M)$, entonces la máquina de Turing terminaría en la siguiente configuración: $(\bigcirc, \sigma_1 \dots \sigma_n \underline{\boxtimes}$

1.6.2. Autómata de Pila

Haremos la equivalencia para un autómata de pila simplificado.

Un autómata de pila simplificado es de la forma:

$$M = (Q, \Sigma, \Gamma, s, f, \Delta)$$

Donde:

$$\Delta \subseteq (Q \times (\Sigma \cup \lambda) \times (\Gamma \cup \lambda)) \times (Q \times (\Gamma \cup \lambda))$$

Y:

$$((q, a, b), (p, c)) \in \Delta \Rightarrow (b = \lambda \Leftrightarrow c \neq \lambda)$$

Recuerde que lo que indica esta regla es simplemente que las reglas sólo pueden empilar o desempliar un único símbolo. Sólo pueden ser *push* o *pop*. No puede ser *Skip*, *pushOn*, *changeTop*.

La máqina de Turning extendida correspondiente se construiría así:

Vamos a construir una máquina de Turing tal que para todo $\omega \in L(M)$ si la máquina comienza en una configuración $\underline{\#}\omega$ en el estado inicial terminará en un estado: $\beta \boxtimes \phi$ y para cada $\omega \notin L(M)$ si la máquina comienza en una configuración $\bigcirc, \underline{\#}\omega$ en el estado inicial terminará en un estado: $\bigcirc, \beta \boxtimes \phi$ o trancada por no poder seguir.

Comenzamos:

$$T = (Q_T, \Sigma_T, q_{IT}, \delta_T$$

Inicialmente:

- $Q_T = Q \cup \{q_{IT}\}$
- $\Sigma_T = \Sigma \cup \Gamma\{\$\}$ donde \$ es un símbolo que no está ni en Σ ni en *Gamma*.

Vamos a agregar unas reglas que le permiten a la máquina pasar de la configuración:

$(q_{IT}, \underline{\#}\omega)$ a la configuración: $(q_I, \omega\$)$

donde está parado en el primer símbolo de ω o en el blanco siguiente si ω es vacío.

Estas serían las reglas $\delta(q_{IT}, \#) = (q_{mark}, (\#, \Rightarrow))$ y para cada $\sigma \in \Sigma$ agregamos la regla $\delta(q_{mark}, \sigma) = (q_{mark}, (\sigma, \Rightarrow))$ y además la siguiente: $\delta(q_{mark}, \#) = (q_{markR}, (\$, \leftarrow))$ El estado $markR$ sirve para devolvernos a principio: $\delta(q_{markR}, \sigma) = (q_{markR}, (\sigma, \leftarrow))$ $\delta(q_{markR}, \#) = (q_I, (\#, \rightarrow))$

Vamos a agregar estados que nos servirán para empilar un valor en la pila y volver a donde íbamos leyendo.

$delta(q_{push_\rho}, \sigma) = (delta(q_{push_\rho}, (\sigma, \rightarrow)))$

$delta(q_{push_\rho}, \$) = ((p_{push_\rho}, \mu, \rightarrow)))$

$delta(q_{push_\rho}, \$) = ((p_{push_\rho}, \mu, \rightarrow)))$