

Documentación

Integrantes:

Juan Jose Rueda Mejia - jjruedam@unal.edu.co

Julio Javier Munoz Quinones - jjmunozq@unal.edu.co

Camilo Andres Carranza Carvajal - ccarranzac@unal.edu.co

AFD

Después de introducir los componentes del Autómata Finito determinista, Se establece una configuración instantánea inicial y se guarda en el procesamiento, antes de iniciar el procesamiento como tal. Se inicializan los valores de la configuración ingresada, generando los siguientes atributos:

1. Conjunto de estados
2. Estado inicial
3. Conjuntos de estado de aceptación
4. Alfabeto
5. Función de transición

Al ingresar una cadena , en primer lugar se verifica que esta pertenezca o no al alfabeto representado por el autómata finito determinista; la función `alfabetoCorrecto(cadena)` se encarga de esta tarea. Recorre cada carácter de la cadena para verificar que estos estén incluidos dentro del alfabeto especificado por el AFN. Posteriormente, se utilizar el método `esAceptada(cadena)`, la cual se encarga de realizar cada una de las transiciones indicadas en la cadena ingresada, con el fin de determinar si la cadena es o no aceptada por el AFN en cuestión.

Además, la clase también cuenta con la función `procesarListaCadenas(String cadenas[];String nombre, boolean imprimir)` la cual realiza en mismo procedimiento que la función anterior, pero aplicado a una lista de cadenas; además de permitirnos guardar el procesamiento generado en un archivo de texto.

```

*****
EL ARCHIVO CARGADO ES:
#alphabet
a
b
#states
q0
q1
q2
#initial
q0
#accepting
q0
q2
#transitions
q0:a>q0
q0:b>q1
q1:a>q1
q1:b>q2
q2:a>q1
q2:b>q1
*****

```

Run: Main

```

desea procesar las cadenas en un archivo adicional(1) o en pantalla(ingrese cualquier otro número): 1
Cuantas cadenas desea evaluar: 2
Ingrese la cadena 1: aababa
Ingrese la cadena 2: aabab
Ingrese el nombre del archivo que donde imprimir el procesamiento de las cadenas: salida
Desea que adicionalmente se imprima el procesamiento en consola (S/N): S
aababa
(q0,aababa)->
(q0,ababa)->
(q0,baba)->
(q1,aba)->
(q1,ba)->
(q2,a)->
(q1,$)>>
no
aabab
(q0,aabab)->
(q0,abab)->
(q0,bab)->
(q1,ab)->
(q1,b)->
(q2,$)>>
yes
ARCHIVO CREADO
Presione ENTER para salir...

```

salida.txt: Bloc de notas

Archivo Edición Formato Ver

aababa
(q0,aababa)->
(q0,ababa)->
(q0,baba)->
(q1,aba)->
(q1,ba)->
(q2,a)->
(q1,\$)>>
no
aabab
(q0,aabab)->
(q0,abab)->
(q0,bab)->
(q1,ab)->
(q1,b)->
(q2,\$)>>
yes

AFPD

Después de introducir los componentes del Autómata Finito con pila determinista, Se establece una configuración instantánea inicial y se guarda en el procesamiento, antes de iniciar el procesamiento como tal. Se inicializan los valores de la configuración ingresada, generando los siguientes atributos:

1. Conjunto de estados
2. Estado inicial
3. Conjuntos de estado de aceptación
4. Alfabeto de cinta
5. Alfabeto de pila
6. Función de transición

Al ingresar una cadena que se desea comprobar si pertenece o no al lenguaje representado por el autómata finito con pila determinista. Primero, el método (`alfabetoCorrecto(cadena)`), revisa cada elemento de la cadena estén en el alfabeto de cinta del autómata, el método termina retornando un booleano(`True`, para la cadena que pertenece al alfabeto de la cinta y `False` para la cadena que no pertenece al alfabeto).

Se sigue con el método `esAceptada(cadena)`. Para iniciar el procesamiento y para continuar realizando cada paso computacional, se verifica que el estado actual no sea el de aceptación, la cadena aún no ha sido procesada y la pila no tiene elementos. Una vez dentro de un paso computacional, se verifica si el estado actual, la letra en la posición actual coinciden con alguno de las exigidas por alguno de las transiciones en "delta" y se puede procesar la pila(`push`, `pop`), de ser así se reemplazan tanto el estado como la letra por las correspondientes de la transición y dependiendo del movimiento, se crea la nueva cadena que falta por procesar.

Si se procesa toda la cadena, la pila se encuentra vacía y se encuentra en un estado de aceptación, se establece el estatus en `Accepted` o `Rejected` si no se cumple las condiciones mencionadas. Por último, se genera un archivo e imprime el procesamiento y el estatus.

```
Run: Main x
Ingrese el nombre del archivo: ejemplo2.dpda
Extensión de archivo correcto
*****
EL ARCHIVO CARGADO ES:
#alphabet
0
1
#alphabetP
A
#states
q0
q1
#initial
q0
#accepting
q0
#transitions
q0:1|$>q1|A
q0:0|$>q1|A
q1:0|A>q0|$
q1:1|A>q0|$
*****
Presione ENTER para continuar...
```

```
Run: Main x
Ingrese la cadena 3: 111001
Ingrese el nombre del archivo que donde imprimir el procesamiento de las cadenas: resultado
Desea que adicionalmente se imprima el procesamiento en consola (S/N): N
4
3
2
1
0
(q0,10100,$)->(q1,0100,$A)->(q0,100,$)->(q1,00,$A)->(q0,0,$)->(q1,$,$A)>>no
3
2
1
0
(q0,1100,$)->(q1,100,$A)->(q0,00,$)->(q1,0,$A)->(q0,$,$)>>yes
5
4
3
2
1
0
(q0,111001,$)->(q1,11001,$A)->(q0,1001,$)->(q1,001,$A)->(q0,01,$)->(q1,1,$A)->(q0,$,$)>>yes
ARCHIVO CREADO
Presione ENTER para salir...
```

```
resultado.txt x
1 10100
2 (q0,10100,$)->(q1,0100,$A)->(q0,100,$)->(q1,00,$A)->(q0,0,$)->(q1,$,$A)>>no
3 1100
4 (q0,1100,$)->(q1,100,$A)->(q0,00,$)->(q1,0,$A)->(q0,$,$)>>yes
5 111001
6 (q0,111001,$)->(q1,11001,$A)->(q0,1001,$)->(q1,001,$A)->(q0,01,$)->(q1,1,$A)->(q0,$,$)>>yes
7
```

AFPN

Después de introducir los componentes del Autómata Finito con Pila no Determinista, Se establece una configuración instantánea inicial y se guarda en el procesamiento, antes de iniciar el procesamiento como tal. Se inicializan los valores de la configuración ingresada, generando los siguientes atributos:

1. Conjunto de estados
2. Estado inicial
3. Conjuntos de estado de aceptación
4. Alfabeto de la cinta
5. Alfabeto de la pila
6. Función de transición

Al ingresar una cadena , en primer lugar se verifica que esta pertenezca o no al alfabeto representado por el autómata finito con pila no determinista; la función `alfabetoCorrecto(cadena)` se encarga de esta tarea. Recorre cada carácter de la cadena para verificar que estos estén incluidos dentro del alfabeto especificado por el AFPN. Posteriormente, se utilizar el método `esAceptada(cadena)`, la cual se encarga de realizar cada una de las transiciones indicadas en la cadena ingresada, con el fin de determinar si la cadena es o no aceptada por el AFPN en cuestión. Para el desarrollo de esta funcionalidad se usó una estructura de datos denominada "Stack" la cual es perfecta para el desarrollo de de este tipo de autómatas. Para que una cadena de caracteres sea aceptada, debe cumplir tres requisitos: el primero, que la cadena se haya procesado completamente; el segundo, que la pila asociada al AFPN se encuentre vacía; y la tercera, que el procesamiento termine en un estado de aceptación.

Además, la clase también cuenta con la función `procesarListaCadenas(String cadenas[],String nombre, boolean imprimir)` la cual realiza en mismo procedimiento que la función anterior, pero aplicado a una lista de cadenas; además de permitirnos guardar el procesamiento generado en un archivo de texto.

```
*****
Ingrese el nombre del archivo:
ejemplo3.pda
Extensión de archivo correcto
*****
EL ARCHIVO CARGADO ES:
#states
q0
#initial
q0
#accepting
q0
#tapeAlphabet
a
b
#stackAlphabet
A
B
#transitions
q0:a:$>q0:A
q0:a:B>q0:$
q0:b:$>q0:B
q0:b:A>q0:$
*****
Presione ENTER para continuar...
|
```

```
Run: Main x
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[[]])>>
accepted
Procesamiento 2:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[A, B])>>
rejected
Procesamiento 3:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A, B, A])->
(q0,$,[A, B, A, B])>>
rejected
Procesamiento 4:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[[]])>>
accepted
Procesamiento 5:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,$)->
(q0,b,[A])->
(q0,$,[A, B])>>
rejected
ARCHIVO CREADO
-----
```

```
salida.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
abab
Procesamiento 1:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[[]])>>
yes
Procesamiento 2:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[A, B])>>
no
Procesamiento 3:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A, B, A])->
(q0,$,[A, B, A, B])>>
no
Procesamiento 4:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,[A, B])->
(q0,b,[A])->
(q0,$,[[]])>>
yes
Procesamiento 5:
(q0,abab,$)->
(q0,bab,[A])->
(q0,ab,$)->
(q0,b,[A])->
(q0,$,[A, B])>>
no
```

AF2P

Después de introducir los componentes del Autómata Finito con dos Pilas (AF2P), Se establece una configuración instantánea inicial y se guarda en el procesamiento, antes de iniciar el procesamiento como tal. Se inicializan los valores de la configuración ingresada, generando los siguientes atributos:

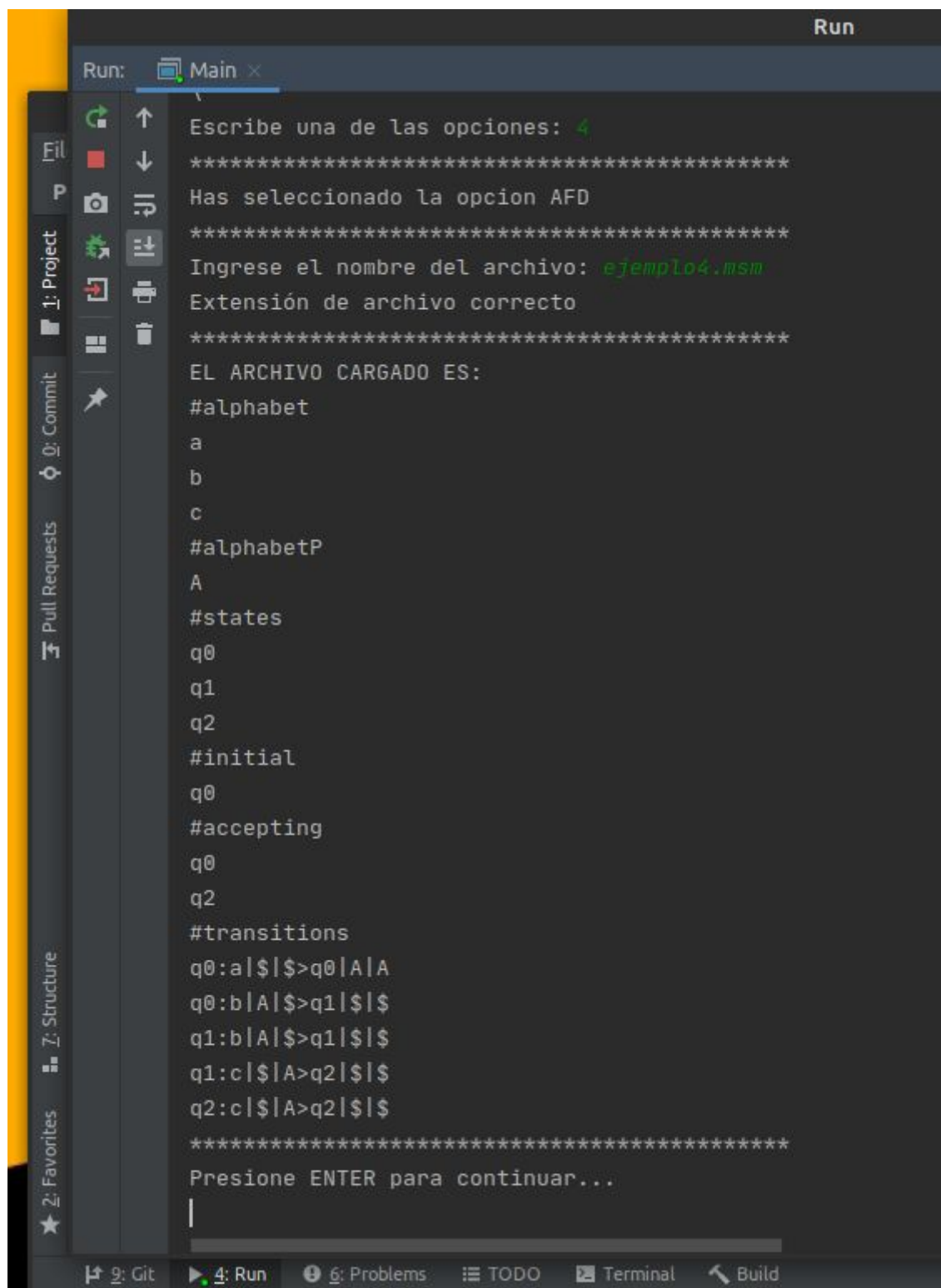
1. Conjunto de estados
2. Estado inicial
3. Conjuntos de estado de aceptación
4. Alfabeto de cinta
5. Alfabeto de pila
6. Función de transición

foto configuracion inicial

Al ingresar una cadena que se desea comprobar si pertenece o no al lenguaje representado por el autómata finito con dos pilas. Primero, el método (`alfabetoCorrecto(cadena)`), revisa cada elemento de la cadena estén en el alfabeto de cinta del autómata, el método termina retornando un booleano(`True`, para la cadena que pertenece al alfabeto de la cinta y `False` para la cadena que no pertenece al alfabeto).

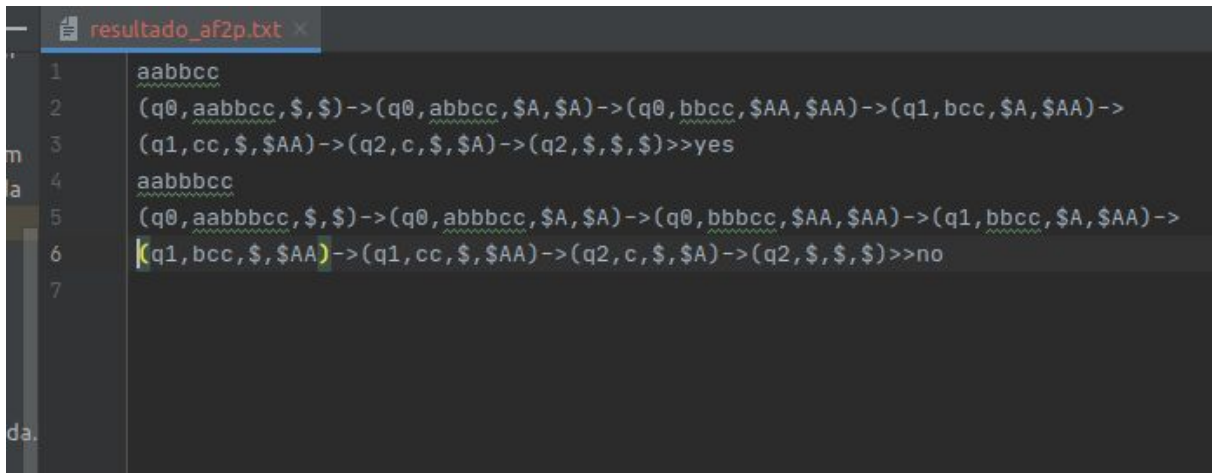
Para iniciar el procesamiento y para continuar realizando cada paso computacional, se verifica que el estado actual no sea el de aceptación, la cadena aún no ha sido procesada y las dos pilas no tiene elementos. Una vez dentro de un paso computacional, se verifica si el estado actual, el elemento de la cadena en la posición actual coinciden con alguno de los exigidos por alguno de las transiciones en "delta" y si se puede procesar con un añadir,eliminar o no hacer nada la pila 1 y/o pila 2 , de ser así se reemplazan tanto el estado como las pilas por las correspondientes de la transición y dependiendo del movimiento, se crea la nueva cadena que falta por procesar.

Si se procesa toda la cadena, las pilas se encuentran vacías y el paso computacional llegó a un estado de aceptación, se establece el estatus en `Accepted` o `Rejected` si no se cumple las condiciones mencionadas. Por último se imprime la cadena, el procesamiento y el estatus.



```
*****
Presione ENTER para continuar...

desea procesar las cadenas en un archivo adicional(1) o en pantalla(ingrese cualquier otro número): 1
Cuántas cadenas desea evaluar: 2
Ingrese la cadena 1: aabbcc
Ingrese la cadena 2: aabbbcc
Ingrese el nombre del archivo que donde imprimir el procesamiento de las cadenas: resultado_af2p
Desea que adicionalmente se imprima el procesamiento en consola (S/N): N
1
1
(q0,aabbcc,$,$)->(q0,abbcc,$A,$A)->(q0,bbcc,$AA,$AA)->(q1,bcc,$A,$AA)->(q1,cc,$,$AA)->(q2,c,$,$A)->(q2,$,$,$)->>yes
1
1
(q0,aabbbcc,$,$)->(q0,abbbcc,$A,$A)->(q0,bbbcc,$AA,$AA)->(q1,bbcc,$A,$AA)->(q1,bcc,$,$AA)->(q1,cc,$,$AA)->(q2,c,$,$A)->(q2,$,$,$)->>no
ARCHIVO CREADO
Presione ENTER para salir...
```



Maquina de Turing

Después de introducir las componentes de la máquina de Turing, se inicializan los valores del estado, posición(iniciaremos en la posición 1), status (Accepted or Rejected) y una línea de texto vacía para el procesamiento.

Al ingresar la palabra que se desea comprobar si pertenece o no al lenguaje representado por la MT, se actualiza la cinta introduciendo la palabra y dos signos "!", para expresar las posiciones vacías a los extremos de la palabra, por esto mismo la posición se inicializa en 1, para que señale la primera componente de la palabra.

Se establece una configuración instantánea inicial y se guarda en el procesamiento, antes de iniciar el procesamiento como tal.

Para iniciar el procesamiento y para continuar realizando cada paso computacional, se verifica que el estado actual no sea el de aceptación (El cual consideramos único, ya que las MT permiten esta exigencia). Una vez dentro de un paso computacional, se verifica si el estado actual y la letra en la posición actual coinciden con alguno de

las exigidas por alguno de las transiciones en “delta”, de ser así se reemplazan tanto el estado como la letra por las correspondientes de la transición y dependiendo del movimiento, se suma, resta una unidad o se mantiene igual.

Si una configuración pasa por cada uno de los procesamientos posibles, sin satisfacer las condiciones de ninguno de ellos, el estatus se establece en Rejected , se imprime imprime el procesamiento y el estatus. De lo contrario, llegara al estado de aceptación, el procesamiento se imprime y el estatus como Accepted.

```
*****
Has seleccionado la opcion Máquina de Turing - Modelo
*****
Ingrese el nombre del archivo: ejemplo5.tm
Extensión de archivo correcto
*****
EL ARCHIVO CARGADO ES:
#states
q0
q1
q2
q3
q4
q5
q6
q7
q8
#initial
q0
#accepting
q8
#inputAlphabet
a
b
c
#tapeAlphabet
a
b
c
A
B
C
#transitions
q0:a?q1:A:>
q1:a?q1:a:>
```

```

desea procesar las cadenas en un archivo adicional(1) o en pantalla(ingrese cualquier
Cuántas cadenas desea evaluar: 2
Ingrese la cadena 1: abbccc
Ingrese la cadena 2: abc
Ingrese el nombre del archivo que donde imprimir el procesamiento de las cadenas: salida.txt
Desea que adicionalmente se imprima el procesamiento en consola (S/N): S
abbccc
Procesamiento:
!q0abbccc!->!Aq1bbccc!->!ABq2bccc!->!ABbq2ccc!->!ABbq3Ccc!->!ABq3bCcc!->!Aq3BbCcc!->!
abc
!q0abc!->!Aq1bc!->!ABq2c!->!ABq3C!->!Aq3BC!->!q3ABC!->!Aq0BC!->!ABq4C! (Rejected)
ARCHIVO CREADO

Presione ENTER para salir...
|

```



```

salida.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
abbccc
Procesamiento:
!q0abbccc!->!Aq1bbccc!->!ABq2bccc!->!ABbq2ccc!->!ABbq3Ccc!->!ABq3bCcc!->!Aq3BbCcc!->!
abc
!q0abc!->!Aq1bc!->!ABq2c!->!ABq3C!->!Aq3BC!->!q3ABC!->!Aq0BC!->!ABq4C! (Rejected)

```

Máquina de Turing Modelo con una Cinta dividida en Pistas

Similar al caso de la MT, se introducen las componentes de la MTP, nuevamente asumiendo un único estado de aceptación. A partir de la expresión explícita de la función delta se deduce el número de pistas de la MTP, señalado como N_{Pistas} , un número que será de suma importancia en la implementación.

También se inicializan el estado, la posición (nuevamente en 1 por los mismos motivos que en la MT), el estatus y un vector de " N_{Pistas} " componentes, cada una una línea de texto vacía, la cual se actualizará cuando se introduzca la palabra que se va a procesar, siendo la $\text{Pista}[0] = "!" + \text{myObj.nextLine()} + "!"$ y las otras pistas, cadenas de "!", de la misma longitud de $\text{Pista}[0]$, para manejar fácilmente el posicionamiento (simultáneo) de todas las pistas.

Se establece la configuración instantánea inicial y se guarda, al iniciar el procesamiento, al igual que en la MT, se recorre cada posible transición y se comparan las condiciones de la misma.

En este caso, por el posible gran número de pistas y a favor del entendimiento de quien lo programó, aunque esta posiblemente no sea la mejor manera de hacerlo, se comparó pista por pista, la letra en la posición actual, con las correspondientes letras en las transiciones, guardando una "coincidencia" cada que esto ocurre, para luego comparar el número de coincidencias con el de pistas, si este coincide se procede a verificar que los estados correspondiera adecuadamente y de ser así a hacer los reemplazos de cada una de las letras de la posición actual de cada pista, del estado y finalmente de la posición.

Al igual que en la MT la intención es que si la configuración recorre cada posible transición, sin coincidencias, la cadena sea rechazada y de lo contrario llegaría inevitablemente a la aceptación.

```
*****
Has seleccionado la opcion Máquina de Turing Modelo con una Cinta dividida en Pistas
*****
Ingrese el nombre del archivo: ejemplo6.ttm
Extensión de archivo correcto
*****
EL ARCHIVO CARGADO ES:
#states
q0
q1
q2
q3
q4
q5
#initial
q0
#accepting
q5
#inputAlphabet
a
b
c
#tapeAlphabet
a
b
c
X
Y
Z
#transitions
q0:a:!?q1:a:X:>
q1:a:!?q1:a:!:>
q1:b:Y?q1:b:Y:>
```

```
desea procesar las cadenas en un archivo adicional(1) o en pantalla(ingrese cualquier otro número): 1
Cuantas cadenas desea evaluar: 2
Ingrese la cadena 1: aabbcc
Ingrese la cadena 2: abcc
Ingrese el nombre del archivo que donde imprimir el procesamiento de las cadenas: salida
Desea que adicionalmente se imprima el procesamiento en consola (S/N): S
aabbcc
Procesamiento:
(!,!) (q0)(a,!)(a,!)(b,!)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(q1)(a,!)(b,!)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(a,
abcc
(!,!) (q0)(a,!)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(q1)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(b,Y)(q2)(c,!)(c,!)(!,
ARCHIVO CREADO
Presione ENTER para salir...
|
```

```
salida.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
aabbcc
Procesamiento:
(!,!) (q0)(a,!)(a,!)(b,!)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(q1)(a,!)(b,!)(b,!)(c,!)(c,!),
X)(b,Y)(b,Y)(c,Z)(c,Z)(q4)(!,!)->(!,!) (a,X)(a,X)(b,Y)(b,Y)(c,Z)(c,Z)(q5)(!,!) (Accept
abcc
(!,!) (q0)(a,!)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(q1)(b,!)(c,!)(c,!)(!,!)->(!,!) (a,X)(b,
```

Máquina de Turing Modelo con Múltiples Cintas

Al igual que los dos casos anteriores, se inicializa cada componente de la MTMC, y varios atributos para el control de las cintas y transiciones.

La idea al implementar múltiples cintas es fundamentalmente la misma que en la división en pistas, se crea un vector, cuyo tamaño depende de cómo esté definida la función delta y cuyas componentes corresponderá a las múltiples cintas, con la diferencia fundamental de que en este caso se independiza el movimiento de cada cinta (componente del vector), mediante la creación de un vector, del mismo tamaño que el de cintas, pero en este caso cada componente es un número natural, que indica en qué posición de la cinta se encuentra, y de salirse de la misma agregar el símbolo "!", a manera de vacío.

Se conservará la manera de verificar las exigencias de cada transición, para efectuar o no, si se da el caso que recorre cada transición, sin cumplir las condiciones, se rechazará, si nunca es rechazada, inevitablemente llegará a la aceptación.