Universidad Autónoma del Estado de Hidalgo

Práctica 2. AFD y AFND

OPCION 2

ALUMNO: ROBERTO MARTÍNEZ MAGAÑA
Dr. Eduardo Cornejo-Velázquez

Opción 2

INSTRUCCIONES. Resuelve el siguiente reto.

https://omegaup.com/arena/problem/Simulacion-de-AFD/

Sigue las instrucciones del reto y la condiciones que debe cumplir el programa. El entregable es el reporte de la práctica más el código del programa.

15320. Simulación de un Autómata Finito Determinista

Points	17.46	Memory limit	32 MiB
Time limit (case)	1s	Time limit (total)	1m0s
Input size limit (bytes)	10 KiB		

Figure 1: Ejercicio

Descripción

Un autómata finito determinista (AFD) es una 5-tupla $(Q, \sum, \delta, q_0, F)$, donde:

Qes un conjunto finito llamado estados,
$\sum$: es un conjunto finito llamado alfabeto,
$\delta : Qx \sum -> Q$ es la función de transición,
q_0 Q es el estado inicial,
F Q es el conjunto de estados de aceptación.

Sea M = $(Q, \sum, \delta, q_0, F)$ un AFD y sea $w = w_1 w_2 ... w_n$ una cadena de símbolos donde $w_i \sum$. Entonces M acpeta w si existe una secuencia de estados $w = w_0 w_1 ... r_n$ en Q con tres condiciones:

1. $r_0 = q_0$,
2.$\sum(r_i, w_i + 1)$, para $i = 0, ..., n-1, y$
3. $r_n F$.

La condición 1 establece que la máquina comienza en el estado inicial. La condición 2 establece que la máquina cambia desde un estado hacia otro estado de acuerdo a la función de transición. La condición 3 establece que la máquina acepta la cadena de entrada si termina en un estado de aceptación. Entonces, M reconoce el lenguaje A si A = w—M acepta w.

Escribir un programa para determinar si un conjunto de cadenas W pertenecen o no pertenecen al lenguaje A reconocido por un AFD M.

ENTRADA

La primer linea de entrada contiene seis enteros $N, S, D, q_0, TyC, (1 \leq N, S, C \leq 100, 1 \leq D \leq 10_4, 1 \leq q_0 \leq N, 0 \leq T \leq N)$, donde $N = |Q|, S = |\sum|, D = NxS$ es el número de transiciones en el automata, q_0 es el estado inicial, $T = |F|, yC$ es la cantidad de cadenas a verificar si son aceptadas o no por el autómata M. Cada estado qQ se identifica de manera implícita por un número entero con valor entre $1yN$.

La segunda línea contiene el alfabeto $\sum$,representado por una secuencia de S símbolos s_i separados por espacios, tal que cada símbolo s_i puede ser una letra, un dígito o cualquier carácter del código ASCII excepto por el espacio.

La tercer línea contiene el conjunto de estados de aceptación F, representado por una secuencia de T enteros $t_i (1 \leq t_i \leq N)$separados por espacios.

Las siguientes D líneas especifican las transiciones del autómata. Cada línea define una transición $\delta(I, X) = J$ por medio de un entero I, un carácter X y un entero $J (I, JQyX \sum)$ separados por espacios, representando la transición desde el estado I hacia el estado J cuando el símbolo de la entrada sea X.

Finalmente, cada una de las siguientes C líneas contienen una cadena W, compuesta por símbolos que pertenecen al alfabeto $\sum$. La longitud de la cadena W está entre 0 y 100 caracteres.

SALIDA

Para cada una de las C cadenas W se deberá imprimir el mensaje ACEPTADA si el autómata M acepta la cadena W, ó rechazada en caso contrario.

EJEMPLO

Ejemplo

Input	Output
3 2 6 1 1 3 0 1 2 1 0 1 1 1 2 2 0 3 2 1 2 3 0 2 3 1 2 100 0 11	ACEPTADA RECHAZADA ACEPTADA

Figure 2: Ejemplo

Descripción del ejemplo

En el ejemplo el autómata tiene $N = 3$ estados, un alfabeto con $S = 2$ símbolos símbolos (segunda línea $\sum = 0, 1$), se definen $D = 6$ transiciones (líneas $4 - 9$), el estado inicial es $q_0 = 1$,solamente hay $T = 1$ estado de aceptación (tercer línea $F = 2$) y se verifican $C = 3$ palabras W (líneas $10 - 12$). La palabra 100es aceptada, 0 es rechazada y 11 es aceptada. El autómata reconoce palabras que tienen una cantidad par de $0's$ después del último 1. La siguiente figura es una representación gráfica del autómata de ejemplo:

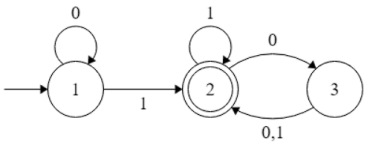

Figure 3: Descripción del ejemplo

Código del ejercicio

include ¡stdio.h¿
include ¡stdlib.h¿
include ¡string.h¿

define MAX_N 100
define MAX_S 100
define MAX_C 100

int transition[MAX_N + 1][256]; // Tabla de transiciones
int accepting[MAX_+ 1]; // Estados de aceptación

int main()

int N, S, D, q0, T, C;
char buffer[512];

// Leer la primera línea completa y parsearla
fgets(buffer, sizeof(buffer), stdin);

```
    sscanf(buffer, %d %d %d %d %d %d", &N, &S, &D, &q0, &T, &C;
```

char alphabet[S];
fgets(buffer, sizeof(buffer), stdin);
strcpy(alphabet, buffer);
alphabet[strcspn(alphabet, ””)] = 0; // Eliminar salto de línea

memset(accepting, 0, sizeof(accepting));
fgets(buffer, sizeof(buffer), stdin);
char *token = strtok(buffer, ” ”);
while (token != NULL)
int state = atoi(token);
accepting[state] = 1;
token = strtok(NULL, ” ”);

memset(transition, -1, sizeof(transition));
for (int i = 0; i ¡ D; i++)
int I, J;
char X;
fgets(buffer, sizeof(buffer), stdin);

```
        sscanf(buffer, "%d %c %d", &I, &X, &J);\\
```

transition[I][(int)X] = J;

for (int i = 0; i ¡ C; i++)
char input[101];
fgets(input, sizeof(input), stdin);
input[strcspn(input, ””)] = 0; // Eliminar salto de línea
int current$_s tate = q0$;
$for(intj = 0; input[j]! = ' '; j + +)$

current$_state = transition[current_state][(int)input[j]];$
$if(current_state == -1)break;$

if (current$_state != -1 accepting[current_state])printf("ACEPTADA");$
$elseprintf("RECHAZADA");$

$fflush(stdout);$

return 0;