Universidad de Valladolid
Departamento de Informática
Teoría de autómatas y lenguajes formales . 2º I.T.Informática. Gestión.

Examen ordinario. 9 de junio de 2011

Apellidos, Nombre.....

Notas para la resolución del examen:

- Entregue las respuestas en el orden del enunciado y el ejercicio 9 separadamente del resto
- El resultado de cada apartado puede ser utilizado aunque no se haya respondido

---

1 **(1 p.)** Dado el autómata siguiente, obtenga un R.F.D. mínimo equivalente, el lenguaje que reconoce, y una gramática de tipo 3 para tal lenguaje. Describa su complementario.

	a	b
$\to 0$	1	3
1	0	3
2	1	4
(3)	5	5
4	3	3
(5)	5	3

Solución: el algoritmo de minimización obtiene:

$$\begin{aligned}\mathcal{P}_0 &= \{\ \{0,1,2,4\}\,,\ \{3,5\}\ \}\\ \mathcal{P}_1 &= \{\ \{0,1\}\,,\{2\}\,,\{4\}\,,\ \{3,5\}\ \}\\ \mathcal{P}_2 &= \mathcal{P}_1\end{aligned}$$

Se ve sin dificultad que los estados 2 y 4 son inaccesibles desde el estado inicial, con lo que el autómata mínimo equivalente es

	a	b
$\to 1$	1	3
(3)	3	3

El lenguaje reconocido es $a^*b\,(a|b)^*$, (*cadenas que contienen al menos una* b) que puede generarse con la gramática de tipo 3 siguiente (se obtiene mediante el algoritmo correspondiente, y eliminando después reglas "épsilon" e inútiles, o bien directamente de la expresión regular)

$$A \to aA \mid bB \mid b \qquad\qquad B \to aB \mid bB \mid a \mid b$$

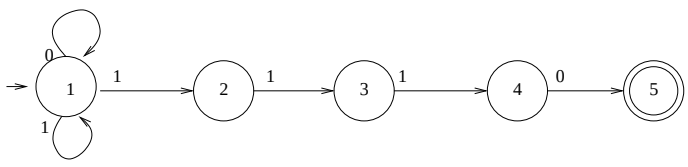
El complementario se caracteriza mediante el autómata

	a	b
$\to (1)$	1	3
3	3	3

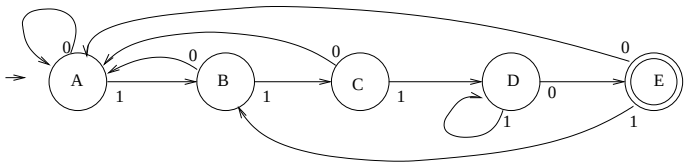
correspondiente a la expresión regular a^* (*cadenas que no contienen ninguna* b)

2 **(1 p.)** Construya un R.F.D. cuyo lenguaje reconocido sea el conjunto de cadenas binarias que terminen en 1110

Solución: se puede partir del RFN



y obtener el determinista mediante el algoritmo habitual, o bien obtenerlo directamente explicando el significado de cada estado. El resultado es



3 **(2 p.)** Para el lenguaje $\{a^{2n}b^nc \ / \ n \geq 0\}$

a) diseñe una gramática que lo genere
Solución: $S \to Rc \qquad R \to aaRb \mid \epsilon$

b) construya un autómata con pila que lo reconozca por vaciado de pila
Solución: Se puede aplicar el algoritmo propuesto en clase: S es el símbolo de inicio de pila.

$\to q$	a	b	c	ϵ
S				(q, Rc)
R				$(q, aaRb)$ (q, ϵ)
a	(q, ϵ)			
b		(q, ϵ)		
c			(q, ϵ)	

c) construya un autómata con pila que lo reconozca por estado final
Solución: También puede aplicarse el algoritmo visto en clase partiendo del anterior. Z es el símbolo de inicio de pila y r_f el (único) estado final:

$\to q_1$	ϵ
Z	(q, SZ)

q	a	b	c	ϵ
S				(q, Rc)
R				$(q, aaRb)$ (q, ϵ)
a	(q, ϵ)			
b		(q, ϵ)		
c			(q, ϵ)	
Z				(r_f, ϵ)

d) pruebe que no es regular
Solución: no cumple el lema de bombeo de los regulares: si lo cumpliese existiría una constante N para tal lema, pero la cadena $z = a^{2N}b^Nc$ (que pertenece al lenguaje y tiene longitud $3N+1 > N$) no admite ninguna subcadena bombeable entre los primeros N caracteres, dado que de ser así, debería estar compuesta por aes, de forma que la descomposición $z = uvw$ tendría que ser $u = a^p$, $v = a^q$ con $q > 0$ y $w = a^{2N-p-q}b^Nc$ y $uv^0w = a^{2N-q}b^Nc$ no pertenece al lenguaje.

4 **(1 p.)** Obtenga una gramática sin símbolos ni reglas inútiles equivalente a la siguiente

$$\begin{array}{lcl} S \to gAe \mid aYB \mid CY & B \to dd \mid D & D \to n \\ A \to bBY \mid ooC & C \to jVB \mid gl & U \to kV \end{array} \qquad \begin{array}{lcl} V \to baXXX \mid oV & X \to fV \\ W \to c & Y \to Yhm \end{array}$$

Solución: son no terminables U, V, X e Y. De la gramática resultante:

$$\begin{array}{lcl} S \to gAe & B \to dd \mid D \\ A \to ooC & C \to gl \end{array} \qquad D \to n \qquad W \to c$$

son inaccesibles B, D y W. Por lo tanto una gramática con las condiciones pedidas es

$$\begin{array}{lcl} S \to gAe \\ A \to ooC \end{array} \qquad C \to gl$$

5 **(1 p.)** Construya la tabla de análisis sintáctico predictivo para la gramática siguiente y escriba el pseudocódigo correspondiente a los procedimientos del analizador para C y U

$$\begin{array}{lcl} S \to AS \mid BAA & C \to cC \mid \epsilon \\ A \to CaA \mid b & B \to dU \end{array} \qquad U \to B \mid \epsilon$$

Solución:

Primeros	S	A	B	C	U
	a	c	d	c	d
	b	a		ϵ	ϵ
	c	b			
	d				

Siguientes	S	A	B	C	U
	$\$$	a	a	a	a
		b	b		b
		c	c		c
		\$			

	a	b	c	d	\$
S	$S \to AS$	$S \to AS$	$S \to AS$	$S \to BAA$	
A	$A \to CaA$	$A \to b$	$A \to CaA$		
B				$B \to dU$	
C	$C \to \epsilon$		$C \to cC$		
U	$U \to \epsilon$	$U \to \epsilon$	$U \to \epsilon$	$U \to B$	

```
procedure C
  begin
     if preanalisis = 'c'  then
        begin parea ('c'); C   end
     else if preanalisis = 'a' then
        begin end
     else
        error
  end
```

```
procedure U
  begin
     if preanalisis in {a, b, c} then
        begin end
     else if preanalisis = 'd'then
        begin B end
     else
        error
  end
```

6 **(1 p.)** a) Dado un lenguaje L_1 recursivamente numerable no recursivo, sobre el alfabeto $\{0, 1\}$, considere

$$L_1' = \{0w \ / \ w \in L_1\} \cup \{1w \ / \ w \notin L_1\}$$

Razone si L_1' podría ser recursivamente numerable.
(INDICACIÓN: construya un algoritmo que, leída una cadena w, construya $1w$ y haga trabajar a la potencial máquina reconocedora de L_1').
Solución: Si L_1' fuera recursivamente numerable, existiría para él una máquina reconocedora M_1'.
Se podría entonces construir una máquina M que lea una cadena binaria w, concatene un 1 por su izquierda para obtener $1w$ y ejecute M_1' sobre $1w$, dando la respuesta de ésta (si la hay).
Este proceso acabará aceptando si y solamente si $1w \in L_1'$, por lo tanto M acabará aceptando si y solamente si $w \notin L_1$, así que reconocerá el complementario de L_1.
Esto justificaría que el complementario de L_1 fuera recursivamente numerable, y como L_1 también lo es, ambos serían recursivos; en particular L_1 sería recursivo, y no es el caso.
Así que L_1' no puede ser recursivamente numerable.

b) Suponga ahora que se sabe de un lenguaje L_2, que el correspondiente

$$L_2' = \{0w \ / \ w \in L_2\} \cup \{1w \ / \ w \notin L_2\}$$

es recursivamente numerable. Razone entonces el tipo de L_2, entre recursivo, recursivamente numerable o no recursivamente numerable.
(PISTA: considere la intersección de L_2' con $0(0|1)^*$ y con $1(0|1)^*$)
Solución 1: $L_2' \cap 0(0|1)^* = 0L_2$.
Como tanto L_2' como $0(0|1)^*$ son recursivamente numerables, resulta que $0L_2$ es recursivamente numerable.
De ello puede deducirse que L_2 también es recursivamente numerable, considerando
o bien una máquina que leída una cadena w, concatene un 0 por su izquierda y lance la máquina reconocedora de $0L_2$, para determinar si $w \in L_2$,
o bien una máquina que genere $0L_2$ y al escribirla suprima el primer 0, para generar L_2
Como $L_2' \cap 1(0|1)^* = 1\overline{L_2}$, de forma similar se deduce que $\overline{L_2}$ es recursivamente numerable.
Por lo tanto, L_2 y $\overline{L_2}$ son recursivamente numerables, y de ahí que L_2 debe ser recursivo.
Solución 2: Se puede utilizar el resultado del apartado a), para deducir que, puesto que L_2' es recursivamente numerable, no es posible que L_2 sea *recursivamente numerable no recursivo*. Por lo tanto, L_2
o bien es recursivo
o bien no es recursivamente numerable.
Pero L_2 sí es recurivamente numerable: puede generarse mediante una máquina de Turing que utilice una generadora de L_2' y solamente escriba las cadenas que empiecen por 0, sin escribir este 0.
Por lo tanto, L_2 tiene que ser recursivo.

7 **(1 p.)** La gramática (ambigua)

$$S \to A;$$
$$A \to A = A \mid \textbf{id}$$

genera expresiones de asignación múltiple en C (por ejemplo "`x = t = z = r;`", que asignaría el valor de `r` a las variables `x, t` y `z`).

En C, `=` es un operador asociativo por la derecha (de forma que `z=r` es una expresión, cuya evaluación produce la asignación del valor de `r` a la variable `z`, y cuyo valor es el de `r`. Este valor se utiliza entonces para la siguiente asignación de la misma manera).

Se pide

a) obtener una gramática no ambigua equivalente que refleje la interpretación adecuada del operador

Solución: La ambigüedad puede eliminarse utilizando la misma técnica que se usó para las expresiones aritméticas, eligiendo en este caso asociatividad por la derecha:

$$S \to A;$$
$$A \to \mathbf{id} = A \mid \mathbf{id}$$

b) si se decide mantener la gramática ambigua y procesarla con YACC ¿cuál sería su respuesta? El analizador que de todas formas YACC generaría ¿daría la interpretación correcta a las cadenas aceptadas? De no ser así ¿sería necesario indicarle de algún modo a YACC cómo debe actuar? Explíquelo brevemente.

Solución: YACC indicaría un conflicto. Concretamente un conflicto desplazamiento reducción, que puede ejemplificarse al analizar la cadena $id = id = id$;

Las dos derivaciones más a la derecha posibles para esta cadena son (se subraya en cada caso el consecuente del pivote)

$$S \Rightarrow \underline{A;} \Rightarrow \underline{A = A;} \Rightarrow A = \underline{A = A;} \Rightarrow A = A = \underline{id;} \Rightarrow A = \underline{id} = id; \Rightarrow \underline{id} = id = id;$$
$$S \Rightarrow \underline{A;} \Rightarrow \underline{A = A;} \Rightarrow A = \underline{id;} \Rightarrow \underline{A = A} = id; \Rightarrow A = \underline{id} = id; \Rightarrow \underline{id} = id = id;$$

La primera es la que refleja asociación por la derecha. Cuando el analizador se encuentre en la situación de la antepenúltima forma sentencial:

pila	entrada	acción
\$	$id = id = id\$$	
...	...	
$\$A = A$	$= id\$$	?

aparece el conflicto: si se reduce se está eligiendo la segunda derivación, y si se desplaza se irá a la busca del pivote en id, para utilizar la primera.

A pesar de este aviso, YACC generará un analizador. Como su comportamiento por defecto en caso de desplazamiento-reducción es elegir desplazamiento (lo que da tendencia a elegir asociación por la derecha, como en este caso) no es necesario indicarle nada a YACC, porque su elección será la que se quiere. Es decir, el analizador generado dará la interpretación correcta a las cadenas aceptadas, y aceptará exactamente las cadenas correctas.

(De todos modos, si se le indica `%right =` el analizador será el mismo, pero YACC ya no encontrará conflictos).

8 **(1 p.)** Describa, en esta misma página, las organizaciones típicas posibles para una tabla de símbolos, señalando cuál es la mejor en términos generales y en qué casos es razonable no emplearla.

Solución: ver transparencias de la clase del profesor Kolár, números de página 222 y 223