Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI3725 - Traductores e Interpretadores
Septiembre-Diciembre 2006

Examen Parcial III

(25 puntos)

Carnet:

Nombre:

1. Sea la Gramática G

$$\begin{aligned} B &\rightarrow B\mathbf{o}T|T \\ T &\rightarrow T\mathbf{y}F|F \\ F &\rightarrow \mathbf{n}F|(B)|\mathbf{c}|\mathbf{f} \end{aligned}$$

a) **(6 puntos)** Calcule el Autómata de Prefijos Viables para la gramática extendida y determine si la gramática es LR(0).

Aumentamos la Gramática con un nuevo símbolo inicial S y denotamos cada producción con un número para futura referencia

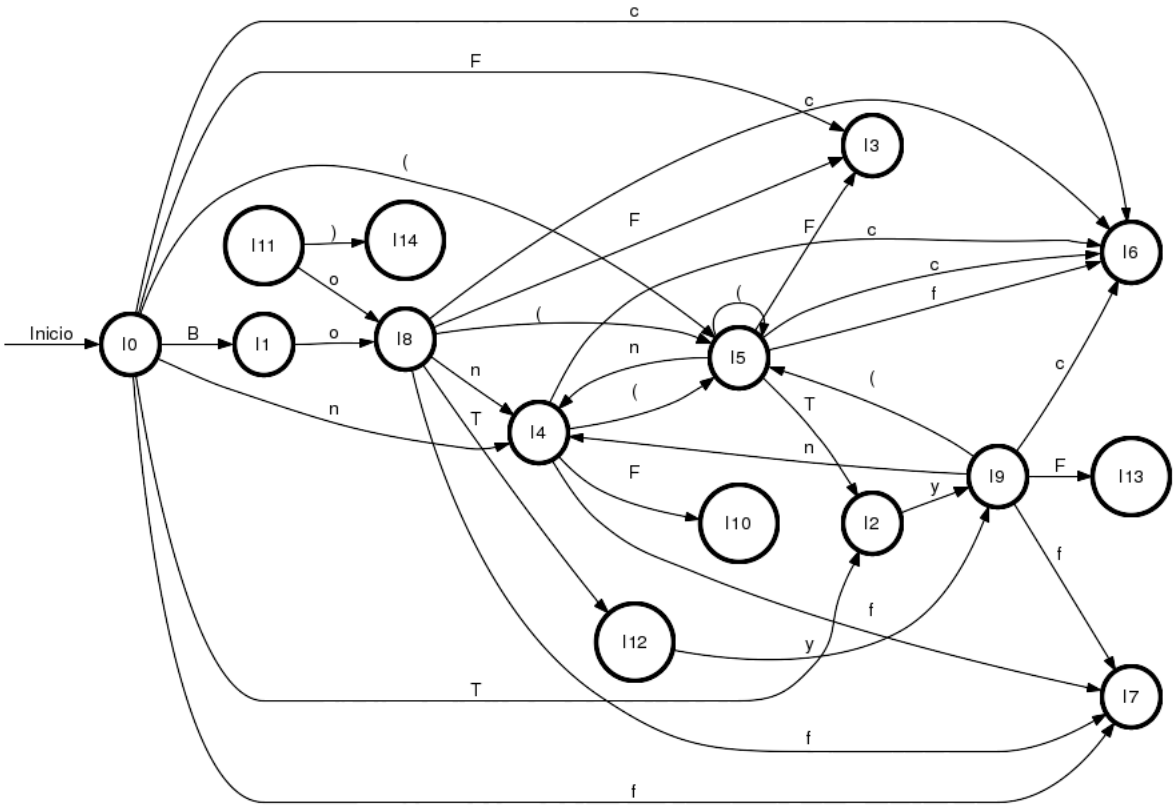
$$\begin{array}{lrcl} \text{Regla 0:} & S & \rightarrow & B \\ \text{Regla 1:} & B & \rightarrow & B\mathbf{o}T \\ \text{Regla 2:} & B & \rightarrow & T \\ \text{Regla 3:} & T & \rightarrow & T\mathbf{y}F \\ \text{Regla 4:} & T & \rightarrow & F \\ \text{Regla 5:} & F & \rightarrow & \mathbf{n}F \\ \text{Regla 6:} & F & \rightarrow & (B) \\ \text{Regla 7:} & F & \rightarrow & \mathbf{c} \\ \text{Regla 8:} & F & \rightarrow & \mathbf{f} \end{array}$$

Calculamos los Conjuntos de Items y su Clausura partiendo de I_0 como sigue:

$$\begin{array}{lcl} I_0 & : & S \rightarrow \cdot B \\ & & B \rightarrow \cdot B\mathbf{o}T \\ & & B \rightarrow \cdot T \\ & & T \rightarrow \cdot T\mathbf{y}F \\ & & T \rightarrow \cdot F \\ & & F \rightarrow \cdot \mathbf{n}F \\ & & F \rightarrow \cdot (B) \\ & & F \rightarrow \cdot \mathbf{c} \\ & & F \rightarrow \cdot \mathbf{f} \\ I_1 & : & S \rightarrow B \cdot \\ & & B \rightarrow B \cdot \mathbf{o}T \\ I_2 & : & B \rightarrow T \cdot \\ & & T \rightarrow T \cdot \mathbf{y}F \end{array}$$

$$
\begin{array}{rcl}
I_3 & : & T \to F \cdot \\
I_4 & : & F \to \mathbf{n} \cdot F \\
 & & F \to \cdot(B) \\
 & & F \to \cdot\mathbf{c} \\
 & & F \to \cdot\mathbf{f} \\
I_5 & : & F \to (\cdot B) \\
 & & B \to \cdot B\mathbf{o}T \\
 & & B \to \cdot T \\
 & & T \to \cdot T\mathbf{y}F \\
 & & T \to \cdot F \\
 & & F \to \cdot\mathbf{n}F \\
 & & F \to \cdot(B) \\
 & & F \to \cdot\mathbf{c} \\
 & & F \to \cdot\mathbf{f} \\
I_6 & : & F \to \mathbf{c} \cdot \\
I_7 & : & F \to \mathbf{f} \cdot \\
I_8 & : & B \to B\mathbf{o} \cdot T \\
 & & T \to \cdot T\mathbf{y}F \\
 & & T \to \cdot F \\
 & & F \to \cdot\mathbf{n}F \\
 & & F \to \cdot(B) \\
 & & F \to \cdot\mathbf{c} \\
 & & F \to \cdot\mathbf{f} \\
I_9 & : & T \to T\mathbf{y} \cdot F \\
 & & F \to \cdot\mathbf{n}F \\
 & & F \to \cdot(B) \\
 & & F \to \cdot\mathbf{c} \\
 & & F \to \cdot\mathbf{f} \\
I_{10} & : & F \to \mathbf{n}F \cdot \\
I_{11} & : & F \to (B \cdot) \\
 & & B \to B \cdot \mathbf{o}T \\
I_{12} & : & B \to B\mathbf{o}T \cdot \\
 & & T \to T \cdot \mathbf{y}F \\
I_{13} & : & T \to T\mathbf{y}F \cdot \\
I_{14} & : & F \to (B)\cdot
\end{array}
$$

El Autómata de Prefijos Viables nos queda como



La Gramática **no** es LR(0) pues presenta conflictos *shift-reduce* en los conjuntos I_1, I_2 e I_{12}.

b) **(2 puntos)** Construya la Tabla de Parsing SLR usando el algoritmo descrito en clase.

Para construir la Tabla de Parsing SLR es necesario calcular el $FOLLOW$ de todos los símbolos no terminales. Así tenemos

$$\begin{aligned} FIRST(S) = FIRST(B) = FIRST(T) = FIRST(F) &= \{\mathbf{n}, (, \mathbf{c}, \mathbf{f}\} \\ FOLLOW(S) &= \{\$\} \\ FOLLOW(B) &= \{\mathbf{o},), \$\} \\ FOLLOW(T) = FOLLOW(F) &= \{\mathbf{y}, \mathbf{o},), \$\} \end{aligned}$$

y la Tabla de Parsing nos queda

Estado	**o**	**y**	**n**	(	)	**c**	**f**	$	B	T	F
0			s4	s5		s6	s7		1	2	3
1	s8							accept			
2	r2	s9			r2			r2			
3	r4	r4			r4			r4			
4				s5		s6	s7				10
5			s4	s5		s6	s7		11	2	3
6	r7	r7			r7			r7			
7	r8	r8			r8			r8			
8			s4	s5		s6	s7			12	3
9			s4	s5		s6	s7				13
10	r5	r5			r5			r5			
11	s8				s14						
12	r1	s9			r1			r1			
13	r3	r3			r3			r3			
14	r6	r6			r6			r6			

c) **(3 puntos)** Utilice el parser SLR construido para obtener la derivación más derecha para la palabra **n(nconf)yc**

Entrada	Pila	Acción
$n(nconf)yc\$$	0 $	shift 4
$(nconf)yc\$$	4 0 $	shift 5
$nconf)yc\$$	5 4 0 $	shift 4
$conf)yc\$$	4 5 4 0 $	shift 6
$onf)yc\$$	6 4 5 4 0 $	reduce 7; pop 1; goto(4,F)
$onf)yc\$$	10 4 5 4 0 $	reduce 5; pop 2; goto(5,F)
$onf)yc\$$	3 5 4 0 $	reduce 4; pop 1; goto(5,T)
$onf)yc\$$	2 5 4 0 $	reduce 2; pop 1; goto(5,B)
$onf)yc\$$	11 5 4 0 $	shift 8
$nf)yc\$$	8 11 5 4 0 $	shift 4
$f)yc\$$	4 8 11 5 4 0 $	shift 7
$)yc\$$	7 4 8 11 5 4 0 $	reduce 8; pop 1; goto(4,F)
$)yc\$$	10 4 8 11 5 4 0 $	reduce 5; pop 2; goto(8,F)
$)yc\$$	3 8 11 5 4 0 $	reduce 4; pop 1; goto(8,T)
$)yc\$$	12 8 11 5 4 0 $	reduce 1; pop 3; goto(5,B)
$)yc\$$	11 5 4 0 $	shift 14
$yc\$$	14 11 5 4 0 $	reduce 6; pop 3; goto(4,F)
$yc\$$	10 4 0 $	reduce 5; pop 2; goto(0,F)
$yc\$$	3 0 $	reduce 4; pop 1; goto(0,T)
$yc\$$	2 0 $	shift 9
$c\$$	9 2 0 $	shift 6
$	6 9 2 0 $	reduce 7; pop 1; goto(9,F)
$	13 9 2 0 $	reduce 3; pop 3; goto(0,T)
$	2 0 $	reduce 2; pop 1; goto(0,B)
$	1 0 $	accept

y la derivación más derecha para la palabra $n(nconf)yc$ se construye usando en orden inverso las reglas indicadas por las reducciones, por tanto

$$\begin{aligned} B &\Rightarrow^2 T \\ &\Rightarrow^3 T\mathbf{y}F \end{aligned}$$

$$
\begin{aligned}
&\Rightarrow^{7} && T\mathbf{yc}\\
&\Rightarrow^{4} && F\mathbf{yc}\\
&\Rightarrow^{5} && \mathbf{n}F\mathbf{yc}\\
&\Rightarrow^{6} && \mathbf{n}(B)\mathbf{yc}\\
&\Rightarrow^{1} && \mathbf{n}(B\mathbf{o}T)\mathbf{yc}\\
&\Rightarrow^{4} && \mathbf{n}(B\mathbf{o}F)\mathbf{yc}\\
&\Rightarrow^{5} && \mathbf{n}(B\mathbf{on}F)\mathbf{yc}\\
&\Rightarrow^{8} && \mathbf{n}(B\mathbf{onf})\mathbf{yc}\\
&\Rightarrow^{2} && \mathbf{n}(T\mathbf{onf})\mathbf{yc}\\
&\Rightarrow^{4} && \mathbf{n}(F\mathbf{onf})\mathbf{yc}\\
&\Rightarrow^{5} && \mathbf{n}(\mathbf{n}F\mathbf{onf})\mathbf{yc}\\
&\Rightarrow^{7} && \mathbf{n}(\mathbf{nconf})\mathbf{yc}
\end{aligned}
$$

2. Sea la Gramática G que describe el lenguaje de expresiones sobre la variable x definida com

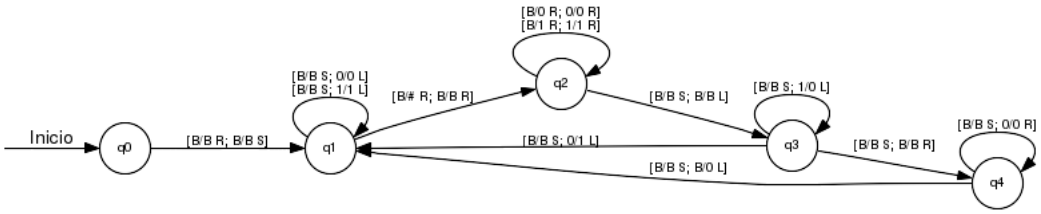
$$
\begin{aligned}
E &\rightarrow E{+}E|E{*}E|(E)\\
E &\rightarrow \mathbf{x}|\mathbf{0}|\mathbf{1}|\mathbf{2}|\mathbf{3}|\mathbf{4}|\mathbf{5}|\mathbf{6}|\mathbf{7}|\mathbf{8}|\mathbf{9}
\end{aligned}
$$

   a) **(4 puntos)** Defina formalmente una Gramática de Atributos para calcular la derivada simbólica de expresiones sobre la variable x **simplificando** el valor final de la expresión, e.g. para la expresión $\mathbf{x} * (\mathbf{x} + \mathbf{1})$ la derivada simbólica resultante debe ser la **cadena** $(\mathbf{x} + \mathbf{1}) + \mathbf{x} * \mathbf{1}$. Asuma la existencia de las funciones $ston(s)$ que recibe una cadena y devuelve su valor numérico asociado y $ntos(n)$ que recibe un número y devuelve la cadena que lo representa. Utilice el operador ++ para denotar la concatenación de cadenas entre comillas. Puede usar condicionales $if - then - else$.

   b) **(2 puntos)** Presente el árbol de derivación decorado con el cálculo de atributos para $(\mathbf{2} * \mathbf{x}) * (\mathbf{5} * (\mathbf{x} + \mathbf{1}))$

3. **(5 puntos)** Construya una Máquina de Turing E_{Σ^*} **determinística multicinta** con **exactamente** dos (2) cintas, tal que la primera cinta sea usada como "salida" para enumerar Σ^* en **orden canónico**, siendo $\Sigma = \{0, 1\}$. Represente la máquina gráficamente utilizando la notación $[L_1/E_1\, D_1; L_2/E_2\, D_2]$ para indicar el símbolo leído, el símbolo escrito y la dirección del movimiento en las cintas uno y dos respectivamente.

   Nota descriptiva: Una máquina enumeradora **debe** utilizar una cinta **solamente** para emitir salida. Esa cinta solamente puede moverse hacia la derecha (R) o permanecer estacionaria (S). Además, **debe** emitir las palabras de Σ^* separadas por un símbolo extra, típicamente #, y en orden, primero de longitud y luego en orden posicional numérico. Tal como comenté en clase, enumerar $\Sigma = \{0, 1\}$ es exactamente igual a construir un **contador binario**, de modo que en la cinta dos tendré siempre un número en binario, sólo que visto desde el bit menos significativo hasta el más significativo, e.g. si en la segunda cinta está el número 13, en binario sería 1101, pero en la cinta estaría $B1011B$. Fíjense que con ésta representación contar en binario es trivial: simplemente me muevo hacia la derecha hasta encontrar un blanco, luego hacia la izquierda: si hay un 0, cambio por 1 y regreso al principio de la cinta; si hay un 1 y mientras haya 1, lo cambio por 0 y sigo hacia la izquierda. Una vez incrementado, es trivial copiar ese número hacia la primera cinta y ponerle un # al final. *No es necesario colocar ésta explicación en la respuesta, pero la he colocado en caso de que haya alguna duda acerca de como funciona mi solución; por cierto, este fue uno de los ejercicios que propuse en clase para que resolvieran.*



4. **(3 puntos)** Demuestre que no existe un **algoritmo** tal que reciba la codificación de una Máquina de Turing M, una cadena $w \in \Sigma^*$y la codificación de $q_i \in Q$, y que pueda **decidir** si el cómputo de M con entrada w pasa por el estado q_i.

   Asumamos que existe la Máquina de Turing S definiendo un **algoritmo** que recibe como entrada la representación de una Máquina de Turing M, una cadena $w \in \Sigma^*$ y la codificación de $q_i \in Q$. La máquina S acepta la entrada si el cómputo de M con entrada w pasa por el estado q_i y no acepta en caso contrario, deteniéndose siempre.

   Consideremos ahora la Máquina Pre-procesadora P definida de la siguiente forma:

   a) La entrada de P es la Representación de una Máquina de Turing M y una cadena $w \in \Sigma^*$. Sea $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$.

   b) La máquina P verifica que la entrada corresponda a una representación válida para una máquina. En caso contrario, se detiene rechazando la entrada.

   c) La máquina P construye una nueva máquina $M' = (Q \cup q_f, \Sigma, \Gamma, \delta', q_0, \{q_f\})$ definida de la siguiente forma:

      1) $\delta'(q_i, x) = \delta(q_i, x)$ siempre que $\delta(q_i, x)$ esté definida. Esto es, se copian todas las transiciones de M a M'.
      2) $\delta'(q_i, x) = [q_f, x, R]$ siempre que $\delta(q_i, x)$ no esté definida. Esto es, se agregan transiciones para que cuando M se detenga, M'pase al estado final q_f.

   Ahora, podemos construir la Máquina de Turing H que recibe como entrada la Representación de una Máquina de Turing M y una cadena $w \in \Sigma^*$. Esta máquina:

   a) Recibe la entrada $R(M)w$ y la pasada al pre-procesador P que produce la máquina intermedia M' antes descrita y escribe en la cinta de entrada $R(M')w$ y luego agrega la representación de q_f al final.

   b) El producto de P en la cinta, es decir $R(M')wq_f$ es pasado a la máquina S

   Es claro que H aceptará $R(M)w$ si y sólo si, M' pasa por el estado q_f cuando calcula sobre w. Pero ésto solamente ocurrirá si M se detiene con la entrada w. Esto quiere decir que H es equivalente al Problema de la Parada para Máquinas de Turing, y lo hemos reducido a S utilizando P como preprocesador. Sabemos que H no es decidible, por tanto S no puede ser decidible como habíamos supuesto; en consecuencia **no existe** un **algoritmo** que decida si el cómputo de una Máquina de Turing M con entrada $w \in \Sigma^*$ pasa por el estado $q_i \in Q$.

   Nota: para que una reducción tenga sentido, la respuesta del problema a reducir tiene que ser la misma que daría el problema original, esto es, el problema que sale del reductor **debe** tener la misma respuesta que tendría el original. *Por cierto, este fue uno de los ejercicios que propuse en clase para que resolvieran.*