CI3725 - Traductores e Interpretadores
Abril-Julio 2006
Prof. Ernesto Hernández-Novich

Laboratorio Semana VIII

Lema de Bombeo para Lenguajes Libres de Contexto (Pumping Lemma)

Sea L un lenguaje libre de contexto. Entonces existe una constante n tal que si z es cualquier palabra en L y $|z| \geq n$ entonces puede escribirse $z = uvwxy$ tal que $|vx| \geq 1$, $|vwx| \leq n$ y $\forall i \geq 0, uv^i wx^i y \in L$.

En muchos casos el Lema de Bombeo es útil para demostrar que un lenguaje **no** es libre de contexto. La estrategia es asumir que el lenguaje es libre de contexto y aprovechar la necesidad del Lema de Bombeo para encontrar una contradicción.

Ejemplo 1

Sea $L = \left\{ a^{k^2} | k \geq 1 \right\}$, el lenguaje de cadenas de a tales que su longitud es un cuadrado perfecto. Queremos demostrar que L **no** es libre de contexto usando el Lema de Bombeo.

Supongamos que L es en efecto libre de contexto, y para aplicar el Lema de Bombeo fijamos un n cualquiera. Esto quiere decir que si $z = a^{n^2}$ está en L y de acuerdo con el Lema de Bombeo puedo escribir $z = uvwxy$ con $|vx| \geq 1$ y $|vwx| \leq n$ y además $\forall i \geq 0, uv^i wx^i y \in L$.

Observemos lo que ocurre cuando $i = 2$. En primer lugar sabemos que $z = a^{n^2} \wedge z = uvwxy \Rightarrow |uvwxy| = n^2$, y por otro lado $|uvwxy| < \left|uv^2wx^2y\right|$, es decir $n^2 < \left|uv^2wx^2y\right|$. Pero al mismo tiempo observamos que $\left|uv^2wx^2y\right| = |uvvwxxy| = |uvwxy| + |v| + |x| = n^2 + |v| + |x| \leq n^2 + n$, pues $1 \leq |vx| \leq n$ según el Lema de Bombeo. Y también es cierto que $n^2 + n < (n+1)^2$. De manera que nos queda

$$n^2 < \left|uv^2wx^2y\right| \leq n^2 + n < (n+1)^2$$

esto es, la longitud de $\left|uv^2wx^2y\right|$ es estrictamente mayor que un cuadrado perfecto y estrictamente menor que el siguiente cuadrado perfecto, por lo tanto no es un cuadrado perfecto, i.e. $\left|uv^2wx^2y\right| \notin L$ que contradice el Lema de Bombeo, por lo tanto L no puede ser libre de contexto como habíamos asumido.

Ejemplo 2

Sea L el lenguaje de prefijos de la cadena $abaabaaabaaaab \ldots ba^n ba^{n+1} b \ldots$ que terminan en b. Queremos demostrar que L **no** es libre de contexto usando el Lema de Bombeo.

Supongamos que L en efecto es libre de contexto, y para aplicar el Lema de Bombeo fijamos un n cualquiera. Escojamos $z = abaab \ldots ba^n b$ que está en L y de acuerdo con el Lema de Bombeo puedo escribir $z = uvwxy$ con $|vx| \geq 1$ y $|vwx| \leq n$ y además $\forall i \geq 0, uv^i wx^i y \in L$.

La primera cláusula del Lema de Bombeo, nos dice que v o x no pueden ser vacíos simultáneamente. Supongamos que $v \neq \lambda$ y observemos las descomposiciones de z[1]:

- **Caso 1**: Es posible que v no tiene ninguna ocurrencia de b. En ese caso v sólo tiene a y está entre dos b consecutivas dentro de la cadena, esto es

  $$\ldots ba^k ba^i va^j ba^{k+2} b \ldots$$

  y tenemos que $i + |v| + j = n + 1$. Si se bombea v, entonces el número de a en el segmento será diferente a $n + 1$ y por tanto $uv^i wx^i y \notin L$, contradiciendo el lema de bombeo.

- **Caso 2**: Es posible que v tenga dos o más ocurrencias de b. En ese caso v tiene una subcadena de la forma

  $$ba^k b$$

  y si se bombea v, aparecerán dos o más subcadenas de esa forma y por tanto $uv^i wx^i y \notin L$, contradiciendo el lema de bombeo.

[1] El caso $x \neq \lambda$ se demuestra de la misma manera.

- **Caso 3**: Es posible que v tenga exactamente una ocurrencia de b. En ese caso $v = a^i b a^j$ y z es de la forma

$$\ldots ba^{k-1}ba^{k-i}va^{k+1-j}ba^{k+2}\ldots$$

Si se bombea v se produce

$$\ldots ba^{k-1}ba^{k-i}a^iba^ja^iba^ja^{k+1-j}ba^{k+2}\ldots$$

$$\ldots ba^{k-1}ba^kba^{j+i}ba^{k+1}ba^{k+2}\ldots$$

y por tanto $uv^iwx^iy \notin L$, contradiciendo el lema de bombeo.

En todos los casos, se contradice el lema de bombeo, por lo que L no puede ser libre de contexto.

Ejemplo 3

Sea $L = \{a^ib^ic^i | i \geq 1\}$. Queremos demostrar que L **no** es libre de contexto usando el Lema de Bombeo.

Supongamos que L en efecto es libre de contexto, y para aplicar el Lema de Bombeo fijamos un n cualquiera. Escojamos $z = a^nb^nc^n$ que está en L y de acuerdo con el Lema de Bombeo puedo escribir $z = uvwxy$ con $|vx| \geq 1$ y $|vwx| \leq n$ y además $\forall i \geq 0, uv^iwx^iy \in L$.

Como $|vwx| \leq n$, entonces vx no puede tener a y c al mismo tiempo, de modo que podemos considerar los siguientes casos:

- **Caso 1**: Es posible que vx sólo esté compuesto por a. Esto quiere decir que $uvwx = a^k$ y $y = a^{n-k}b^nc^n$. Si observamos $uv^0wx^0y = uwy$, resulta obvio que contiene b^n y c^n, pero necesariamente tiene a^j con $j < n$, por lo que no está en L.
- **Caso 2**: Es posible que vx sólo esté compuesto por b. Esto quiere decir que $u = a^n$, $vwx = b^k$ y $y = b^{n-k}c^n$. Si observamos $uv^0wx^0y = uwy$, resulta obvio que contiene a^n y c^n, pero necesariamente tiene b^j con $j < n$, por lo que no está en L.
- **Caso 3**: Es posible que vx sólo esté compuesto por c. Esto quiere decir que $u = a^nb^nc^k$ y $vwxy = c^{n-k}$. Si observamos $uv^0wx^0y = uwy$, resulta obvio que contiene a^nb^n, pero necesariamente tiene c^j con $j < n$, por lo que no está en L.

En todos los casos, se contradice el lema de bombeo, por lo que L no puede ser libre de contexto.

Propiedades de Clausura de los Lenguajes Libres de Contexto

Sean L_1 y L_2 dos lenguajes libres de contexto, generados por las gramáticas libres de contexto $G_1 =< N_1, \Sigma_1, P_1, S_1 >$ y $G_2 =< N_2, \Sigma_2, P_2, S_2 >$. Podemos asumir que $N_1 \cap N_2 = \emptyset$ pues siempre podemos renombrar símbolos según sea necesario.

- Los Lenguajes Libres de Contexto son cerrados sobre la unión de lenguajes. Consideremos la gramática

$$\begin{aligned} G_3 &= < N_1 \cup N_2 \cup \{S_3\}, \Sigma_1 \cup \Sigma_2, P_3, S_3 > \\ P_3 &= P_1 \cup P_2 \cup \{S_3 \to S_1, S_3 \to S_2\} \end{aligned}$$

y veamos que el lenguaje $L_3 = L(G_3)$ corresponde a $L_1 \cup L_2$.

Primero demostraremos que si $w \in L_1 \cup L_2$, entonces puede derivarse usando G_3. Si $w \in L_1$ entonces

$$S_3 \Rightarrow_{G_3} S_1 \Rightarrow^*_{G_1} w$$

es una derivación en G_3 para w; si $w \in L_2$ entonces

$$S_3 \Rightarrow_{G_3} S_2 \Rightarrow^*_{G_2} \Rightarrow w$$

es una derivación en G_3para w. Por tanto $L_1 \cup L_2 \subseteq L(G_3)$.

Ahora demostraremos que si $w \in L(G_3)$, entonces $w \in L_1 \cup L_2$. Si $w \in L(G_3)$ quiere decir que existe $S_3 \Rightarrow^*_{G_3} w$. Hemos de considerar dos casos:

- Si la derivación es de la forma
$$S_3 \Rightarrow_{G_3} S_1 \Rightarrow^*_{G_3} w$$
entonces sólo aparecen símbolos de G_1 puesto que $N_1 \cap N_2 = \emptyset$. Pero en P_3 las únicas producciones que contienen símbolos de G_1 son aquellas producciones que originalmente estaban en P_1. Esto quiere decir que $S_1 \Rightarrow^*_{G_1} w$ y entonces $w \in L(G_1) = L_1$.
- Si la derivación es de la forma
$$S_3 \Rightarrow_{G_3} S_2 \Rightarrow^*_{G_3} w$$
entonces sólo aparecen símbolos de G_2 puesto que $N_1 \cap N_2 = \emptyset$. Pero en P_3 las únicas producciones que contienen símbolos de G_2 son aquellas producciones que originalmente estaban en P_2. Esto quiere decir que $S_2 \Rightarrow^*_{G_2} w$ y entonces $w \in L(G_2) = L_2$.

Por tanto $L(G_3) \subseteq L_1 \cup L_2$.

- Los Lenguajes Libres de Contexto son cerrados sobre concatenación. Consideremos la Gramática
$$\begin{aligned} G_4 &= < N_1 \cup N_2 \cup \{S_4\}, \Sigma_1 \cup \Sigma_2, P_4, S_4 > \\ P_4 &= P_1 \cup P_2 \cup \{S_4 \rightarrow S_1 S_2\} \end{aligned}$$
cuya demostración tiene una estructura similar a aquella de la unión, quedando como ejercicio para el lector.

- Los Lenguajes Libres de Contexto son cerrados sobre clausura transitiva. Consideremos la Gramática
$$\begin{aligned} G_5 &= < N_1 \cup \{S_5\}, \Sigma_1, P_5, S_5 > \\ P_5 &= P_1 \cup \{S_5 \rightarrow S_1 S_5, S_5 \rightarrow \lambda\} \end{aligned}$$
cuya demostración tiene una estructura similar a aquella de la unión, quedando como ejercicio para el lector.

- Los Lenguajes Libres de Contexto **no** son cerrados obre la intersección. Consideremos el lenguaje $L_2 = \{a^i b^i c^j | i \geq 1, j \geq 1\}$ que es generado por la gramática
$$\begin{aligned} S_1 &\rightarrow AB \\ A &\rightarrow aAb \\ &\rightarrow ab \\ B &\rightarrow cB \\ &\rightarrow c \end{aligned}$$
Consideremos el lenguaje $L_3 = \{a^i b^j c^j | i \geq 1, j \geq 1\}$ que es generado por la gramática
$$\begin{aligned} S_2 &\rightarrow CD \\ C &\rightarrow aC \\ &\rightarrow a \\ D &\rightarrow bDc \\ &\rightarrow bc \end{aligned}$$
Ahora observemos que $L_2 \cap L_3 = \{a^i b^i c^i | i \geq 1\}$, y usnado el Lema de Bombeo ya hemos demostrado que este lenguaje no es libre de contexto, por lo que los lenguajes libres de contexto no son cerrados sobre la intersección.

- Los Lenguajes Libres de Contexto **no** son cerrados sobre el complemento. Supongamos que si lo fueran, esto es dados L_1 y L_2 lenguajes libres de contexto, diremos que $\overline{L_1}$ y $\overline{L_2}$ también son libres de contexto.

Como los lenguajes libres de contexto son cerrados sobre la unión, consideremos $L = \overline{L_1} \cup \overline{L_2}$, entonces L tiene que ser libre de contexto y según nuestra hipótesis $\overline{L}$ también. Sin embargo

$$\begin{aligned} \overline{L} &= \overline{\overline{L_1} \cup \overline{L_2}} \\ &\quad \{ \text{ De Morgan } \} \\ &= L_1 \cap L_2 \end{aligned}$$

Donde L_1 y L_2 son libres de contexto, sin embargo sabemos que los lenguajes libres de contexto **no** son cerrados sobre la intersección por lo cual hemos llegado a una contradicción resultante de haber asumido que los lenguajes libres de contexto son cerrados sobre el complemento.

Gramática de Atributos

Las gramáticas de atributos definen la semántica asociando valores a las frases del lenguaje. Esto ayuda a automatizar la construcción de reconocedores "decorados" con valores.

Los atributos pueden ser *sintetizados* o *heredados*. Los *sintetizados* son aquellos construidos para los no-terminales del lado **izquierdo** de cada producción, mientras que los *heredados* son los construidos para los terminales del lado **derecho** de cada producción. En cualquier caso, los atributos son construidos utilizando los valores asociados a los *tokens* del lenguaje y a los atributos de los no-terminales, usando operaciones convenientes para la semántica del lenguaje (aritmética, manipulación de cadenas, construcción de estructuras de datos, etc.).

Siempre es posible reescribir una gramática de atributos para que solamente emplee atributos sintetizados, pues la mayoría de las herramientas generadoras de analizadores sintácticos solamente operan con este tipo de atributos[2].

Ejemplo 1

Sintetizar el valor decimal de un número flotante escrito en binario según la gramática siguiente, e.g. 101,101 = 5,625

$$\begin{aligned} S &\to L \,.\, L \\ &\to L \\ L &\to LB \\ &\to B \\ B &\to \mathbf{0} \\ &\to \mathbf{1} \end{aligned}$$

La solución que presentamos está basada en utilizar atributos heredados y sintetizados. En la literatura suele decirse que utilizar atributos heredados mejora la claridad de la definición; en mi opinión (quizás viciada por la práctica) es más claro utilizar atributos sintetizados aunque puede ser un poco más difícil encontrar la solución "a simple vista".

La solución gira alrededor de los siguientes principios:

- Asociaremos un atributo sintetizado de nombre *valor* con tipo decimal asociado a los no-terminales. En este atributo calcularemos el valor de la expresión. En cada producción asumiremos que el *valor* ha sido calculado por separado para la parte entera y la parte real de la secuencia. El valor de un *bit* particular dependerá de su posición.
- Como la cadena se procesa de izquierda a derecha, la posición de los bits en la parte entera debe incrementarse a medida que son procesados. Asociaremos un atributo heredado de nombre *posicion* con tipo decimal asociado a los no-terminales. En este atributo calcularemos la posición de cada *bit* según vaya siendo procesado.

[2]Como el caso de Happy para Haskell. Bison para C/C++ soporta atributos heredados y sintetizados.

- Por el mismo razonamiento, al procesar la parte decimal, nos interesa calcular

Entonces nos queda la siguiente gramática de atributos

$$
\begin{array}{rcl}
S & \to & L \left\{ \begin{array}{lcl} S.val & \leftarrow & L.val \\ L.pos & \leftarrow & 0 \end{array} \right. \\
S & \to & L_1 \,.\, L_2 \left\{ \begin{array}{lcl} S.lval & \leftarrow & L_1.val + L_2.val \\ L_1.pos & \leftarrow & 0 \\ L_2.pos & \leftarrow & -L_2.lon \end{array} \right. \\
L & \to & B \left\{ \begin{array}{lcl} L.val & \leftarrow & B.val \\ B.pos & \leftarrow & L.pos \\ L.lon & \leftarrow & 1 \end{array} \right. \\
L_0 & \to & L_1 B \left\{ \begin{array}{lcl} L_0.val & \leftarrow & L_1.val + B.val \\ L_1.pos & \leftarrow & L_0.pos + 1 \\ B.pos & \leftarrow & L_0.pos \\ L_0.lon & \leftarrow & L_1.lon + 1 \end{array} \right. \\
B & \to & \mathbf{0} \left\{ \begin{array}{lcl} B.val & \leftarrow & 0 \end{array} \right. \\
B & \to & \mathbf{1} \left\{ \begin{array}{lcl} B.val & \leftarrow & 2^{B.pos} \end{array} \right.
\end{array}
$$

El mismo problema puede resolverse utilizando solamente atributos sintetizados.

Ejemplo 2

Definir una gramática de atributos para calcular la derivada simbólica de expresiones sobre la variable x compuestas de sumas, productos, constantes y expresiones parentizadas. No es necesario simplificar las expresiones resultantes, por ejemplo, la derivada calculada para la expresión

$$3 * (x + 1)$$

sería

$$0 * (x + 1) + 3 * (1 * 1 + x * 0)$$

Proponemos la gramática

$$
\begin{array}{rcl}
E & \to & E + E \\
 & \to & E * E \\
 & \to & \mathbf{digito} \\
 & \to & \mathbf{x} \\
 & \to & (\, E \,)
\end{array}
$$

y utilizaremos solamente atributos sintetizados. Para cada no-terminal nos interesa saber tanto el valor de la expresión simbólica como su derivada, de manera que puedan ser combinados en el cálculo de derivadas. Ambos atributos serán de tipo cadena alfanumérica y utilizaremos el símbolo $\cdot$ para denotar la concatenación de cadenas. Nos queda

$$
\begin{array}{rcl}
E_0 & \to & E_1 + E_2 \left\{ \begin{array}{lcl} E_0.d & \leftarrow & E_1.d \cdot {}' + {}' \cdot E_2.d \\ E_0.v & \leftarrow & E_1.v \cdot {}' + {}' \cdot E_2.v \end{array} \right. \\
E_0 & \to & E_1 * E_2 \left\{ \begin{array}{lcl} E_0.d & \leftarrow & E_1.d \cdot {}' * {}' \cdot E_2.v \cdot {}' + {}' \cdot E_1.v \cdot {}' * {}' \cdot E_2.d \\ E_0.v & \leftarrow & E_1.v \cdot {}' * {}' \cdot E_2.v \end{array} \right. \\
E & \to & \mathbf{digito} \left\{ \begin{array}{lcl} E.d & \leftarrow & {}'0' \\ E.v & \leftarrow & \mathbf{digito}.v \end{array} \right. \\
E & \to & \mathbf{x} \left\{ \begin{array}{lcl} E.d & \leftarrow & {}'1' \\ E.v & \leftarrow & {}'x' \end{array} \right. \\
E_0 & \to & E_1 \left\{ \begin{array}{lcl} E_0.d & \leftarrow & {}'('\cdot E_1.d \cdot {}')' \\ E_0.v & \leftarrow & {}'('\cdot E_1.v \cdot {}')' \end{array} \right.
\end{array}
$$