Hacia las Gramáticas Propias

Gramáticas sin Ciclos

Universidad de Cantabria

Esquema

1 Introducción

2 Gramáticas $\lambda-$libres

3 Gramáticas sin Producciones Simples

Introducción

Las gramáticas libres de contexto pueden presentar diferentes problemas. Ya hemos visto como eliminar los símbolos inútiles que tiene una gramática.
Veamos como transformar una gramática para que no tenga ciclos.

Cosas que Evitar

Las gramáticas presentan ciclos porque:

- Hay producciones de la forma $A \mapsto B$.
- Hay producciones de la forma $A \mapsto \lambda$.

Cosas que Evitar

Las gramáticas presentan ciclos porque:

- Hay producciones de la forma $A \mapsto B$.
- Hay producciones de la forma $A \mapsto \lambda$.

Cosas que Evitar

Claramente, no siempre es posible quitar todas las producciones del tipo $A \mapsto \lambda$, ya que siempre deberá haber al menos una si el lenguaje contiene a la palabra vacía.

Eliminación de $\lambda-$producciones

Definición

Sea $G = (V, \Sigma, Q_0, P)$ una gramática libre de contexto.

1. *Llamaremos $\lambda-$producciones en G a todas las producciones de la forma $X \mapsto \lambda$, donde $X \in V$ es un símbolo no terminal.*
2. *Diremos que la gramática G es $\lambda-$libre si verifica una de las dos propiedades siguientes:*
   - *O bien no posee $\lambda-$producciones,*
   - *o bien la única $\lambda-$producción es de la forma $Q_0 \mapsto \lambda$ y Q_0 no aparece en el lado derecho de ninguna otra producción de P (es decir, no existe ninguna producción de la forma $X \mapsto \alpha Q_0 \beta$, con $\alpha, \beta \in (V \cup \Sigma)^*$).*

Eliminación de $\lambda-$producciones

Definición

Sea $G = (V, \Sigma, Q_0, P)$ una gramática libre de contexto.

1. *Llamaremos $\lambda-$producciones en G a todas las producciones de la forma $X \mapsto \lambda$, donde $X \in V$ es un símbolo no terminal.*
2. *Diremos que la gramática G es $\lambda-$libre si verifica una de las dos propiedades siguientes:*
   - *O bien no posee $\lambda-$producciones,*
   - *o bien la única $\lambda-$producción es de la forma $Q_0 \mapsto \lambda$ y Q_0 no aparece en el lado derecho de ninguna otra producción de P (es decir, no existe ninguna producción de la forma $X \mapsto \alpha Q_0 \beta$, con $\alpha, \beta \in (V \cup \Sigma)^*$).*

Ejemplo de Gramática

Ejemplo

Consideremos la gramática cuyas producciones son:

$$Q_0 \mapsto aQ_0bQ_0 \mid bQ_0aQ_0 \mid \lambda.$$

No es una gramática $\lambda-$libre.

Pregunta sobre las Gramáticas

A la vista de este ejemplo y la definición, ¿se puede resolver con un autómata cuando una gramática es $\lambda-$libre?

- Las variables pueden definirse como q_N, por lo tanto se pueden representar como expresiones regulares.
- Los elementos del alfabeto son representables por expresiones regulares.
- Incluiremos $\mapsto$ como un símbolo.

Resultado

Teorema (Transformación a Gramática $\lambda-$libre)

Toda gramática libre de contexto es equivalente a una gramática $\lambda-$libre. Además, dicha equivalencia es calculable algorítmicamente.

El Grafo de las λ-producciones

Utilizaremos la idea que se utilizó para la eliminación de símbolos inútiles. Primero guardamos en un conjunto $V_\lambda := \{A \in V \ : \ A \vdash_G \lambda\}$.

Algoritmo

Calculamos ahora un nuevo sistema de producciones $\overline{P}$ por medio de una serie de transformaciones, que se basan solamente en sustituciones. Al principio, consideramos que $\overline{P}$ es el conjunto vacio.

Algoritmo

Consideremos todas las producciones de la forma siguiente:

$$A \mapsto \alpha_0 B_1 \alpha_1 \cdots B_k \alpha_k,$$

donde $\alpha_i \in ((V \setminus V_\lambda) \cup \Sigma)^*$, $B_i \in V_\lambda$ y no todos los α_i son iguales a λ.
Definamos

$$\overline{P} := \overline{P} \cup \{A \mapsto \alpha_0 X_1 \alpha_1 \cdots X_k \alpha_k \ : \ X_i \in \{B_i, \lambda\}\}.$$

Algoritmo

Consideremos todas las producciones de la forma siguiente:

$$A \mapsto B_1 \cdots B_k,$$

donde $B_i \in V_\lambda$.
Definamos:

$$\overline{P} := \overline{P} \cup (\{A \mapsto X_1 \cdots X_k \; : \; X_i \in \{B_i, \lambda\}\} \setminus \{A \mapsto \lambda\}).$$

Algoritmo

Añadimos todas las producciones que no sean del tipo de las dos mencionadas anteriormente. Esto es, aquellas en las que en la parte derecha no aparezca ninguna variable de V_λ y no sean $\lambda-$producciones.

Algoritmo

Notar que no hemos introducido ninguna $\lambda-$producción a $\overline{P}$.
Finalmente, si $Q_0 \in V_\lambda$ sea $\overline{V} := V \cup \{Q_0'\}$, con $Q_0' \notin V$. Y añadamos

$$\overline{P} = \overline{P} \cup \{Q_0' \mapsto Q_0 \mid \lambda\}.$$

En otro caso, $\overline{V} = V$.

Últimas Observaciones

Varios puntos para reflexionar:

- ¿Tiene la nueva gramática símbolos inútiles?
- ¿Siempre crece de tamaño? ¿puede ser más pequeña que la inicial?
- ¿Como cambian los árboles de derivación de una palabra?

Producciones Simples

Definición (Producciones Simples o Unarias)

Se llaman producciones simples (unarias) a las producciones de una gramática libre de contexto de la forma $A \mapsto B$, donde A y B son símbolos no terminales.

Resultado

Teorema (Eliminación de Producciones Simples)

Toda gramática $\lambda-$libre es equivalente a una gramática sin producciones simples. Esta equivalencia es calculable algorítmicamente.

Algoritmo

El algoritmo tiene dos partes. La primera parte sigue el mismo esquema algorítmico usado en resultados anteriores. La segunda parte se dedica a eliminar todas las producciones simples.

Clausura Transitiva

Clausura Transitiva de símbolos no terminales. Se trata de calcular, para cada $A \in V$, el conjunto siguiente:

$$V_A := \{B \in V \ : \ A \vdash_G B\} \cup \{A\}.$$

A partir de aquí, el algoritmo seguirá derroteros muy similares.

Algoritmo

Entrada: Una gramática libre de contexto $G := (V, \Sigma, Q_0, P)$

Para cada $A \in V$ calcular

$M_A := \emptyset$

$N_A := \{A\}$

mientras $N_A \neq M$ **hacer**

$M := N_A$

$N_A := \{C \in V \ : \ B \mapsto C \text{ está en P}, \ \text{y } B \in N_A\} \cup N_A$

fin hacer

Salida: Para cada $A \in V$, N_A.

Algoritmo

Eliminar las producciones simples. Para cada variable B tal que existe una producción simple $A \mapsto B$ en P, procederemos somo sigue:

- Hallar todos los X's tales que $B \in V_X$.
- Para cada producción $B \mapsto \alpha$ que no sea producción simple, añadir a P la producción $X \mapsto \alpha$.
- Eliminar toda producción del tipo $X \mapsto B$.

Últimas Observaciones

Varios puntos para reflexionar:

- ¿Tiene la nueva gramática símbolos inútiles?
- ¿Es necesario que la gramática sea $\lambda-$libre para que el algoritmo funcione?
- ¿Después de aplicar el algoritmo sigue siendo la gramática $\lambda-$libre?
- ¿Como cambian los árboles de derivación de una palabra?