Simplificación de CFG

Alan Reyes-Figueroa
Teoría de la Computación **(Aula 15) 25.septiembre.2023**

Eliminar variables sobrantes
Remover producciones-ϵ
Remover producciones unarias

Variables que no derivan nada

- Consideremos:

  $S \rightarrow AB, \quad A \rightarrow aA \mid a, \quad B \rightarrow AB$
- Observe que A deriva todas las cadenas conteniendo a's,
  pero B no deriva ningún símbolo terminal.
- En este caso, S deriva nada, y el lenguaje generado es vacío.

Testando si una variable deriva alguna cadena terminal

- Base: Si existe una producción $A \rightarrow w$, donde w no tiene variables, entonces A deriva una cadena terminal.
- Inducción: Si existe una producción $A \rightarrow \alpha$, donde α consiste sólo de terminales y variables que derivan una cadena terminal, entonces A deriva una cadena terminal.

Testando si una variable deriva alguna cadena terminal

- Eventualmente, llegamos a no encontrar más variables.
- Haciendo una inducción sobre el orden en que las variables "aparecen" muestra que cada una deriva una cadena terminal.
- Recíprocamente, cualquier variable que deriva una cadena terminal siempre se puede encontrar mediante este algoritmo.

Algoritmo para eliminar variables que no derivan nada

1. Descubrir todas las variables que derivan cadenas terminales.
2. Para todas las demás variables, remover todas las producciones en donde dichas variables aparecen (ya sea en la izquierda o en la derecha).

Ejemplo: eliminar variables

$S \rightarrow AB \mid C$, $B \rightarrow bB$,
$A \rightarrow aA \mid a$, $C \rightarrow c$

- Base: A y C se marcan, ya que $A \rightarrow a$ y $C \rightarrow c$ (ambas derivan terminales).
- Inducción: S se marca, ya que $S \rightarrow C$ (deriva símbolo que deriva terminales).
- Nada más se marca.
- Resultado: $S \rightarrow C$, $A \rightarrow aA \mid a$, $C \rightarrow c$

Símbolos inalcanzables

- Otra forma en que un terminal o variable merece ser eliminada es si no puede aparecer en ninguna derivación desde el símbolo de inicio.
- Base: S es alcanzable (S símbolo inicial).
- Inducción: Si podemos alcanzar A desde S, y existe una producción $A \rightarrow \alpha$, entonces α es alcanzable desde S.

Símbolos inalcanzables

- Se puede mostrar (vía inducción) que cuando no podemos descubrir más símbolos alcanzables, tenemos todos y sólo los símbolos que aparecen en las derivaciones de S.
- Algoritmo: Remover de la gramática todos los símbolos no alcanzables desde S y todas las producciones que involucran a dichos símbolos.

Eliminar símbolos sin uso

- Un símbolo es *útil* (*useful*) si este aparece en alguna derivación de alguna cadena terminal desde el símbolo inicial S. En otro caso, es *sin uso* (*useless*).
- Eliminamos todos los símbolos sin uso:
  1. Eliminar símbolos que derivan cadenas no terminales.
  2. Eliminar símbolos no alcanzables.

Ejemplo: Símbolos sin uso

$S \rightarrow AB, \ A \rightarrow C, \ C \rightarrow c, \ B \rightarrow bB$

- Si eliminamos símbolos no alcanzables primero, encontraríamos que todo es alcanzable.
- Luego,
  A, C, y c nunca serían eliminados.

¿Por qué funciona?

- Luego del paso (1), todo símbolo remanente deriva alguna cadena terminal.
- Luego del paso (2), los únicos símbolos remanentes son aquellos derivables de S.
- Adicionalmente, estos símbolos aún derivan una cadena terminal, ya que tal derivación sólo envuelve símbolos alcanzables desde S.

Produccciones Épsilon

- Casi podemos evitar usar producciones del tipo $A \longrightarrow \varepsilon$ (llamadas producciones$-\epsilon$).
  - El problema es que ε no puede pertenecer al lenguaje generado por una gramática que no posee producciones-ϵ.

- Teorema: Si L es una gramática CFG, entonces L-{ϵ} posee una CFG sin producciones-ϵ.

Símbolos Anulables

- Para eliminar producciones-ϵ, primero debemos detectar las *variables anulables* =
  variables A tales que $A \Rightarrow^* \epsilon$.
- Base: Si hay alguna producción $A \rightarrow \epsilon$, entonces A es anulable.
- Inducción: Si existe una producción $A \rightarrow \alpha$, y todos los símbolos de α son anulables, entonces A es anulable.

Ejemplo: Símbolos Anulables

$$S \rightarrow AB,\ \ A \rightarrow aA \mid \epsilon,\ \ B \rightarrow bB \mid A$$

- **Base**: A es anulable ya que $A \rightarrow \epsilon$.
- **Inducción**: B es anulable ya que $B \rightarrow A$.

- En nuestro ejemplo:
  Entonces, S es anulable, ya que $S \rightarrow AB$.

Eliminar Producciones-ϵ

- Idea Clave: Convertir cada producción de la forma $A \rightarrow X_1 \dots X_n$
  en una familia de producciones.
- Para cada subconjunto de X's anulables, existe una producción con aquellos eliminados del lado derecho "in advance."
  - Excepto, si todos los X's son anulables, no crear una producción con ϵ en el lado derecho.

Ejemplo: Eliminar producciones-ϵ

$S \to ABC,\qquad B \to bB \mid \epsilon,$

$A \to aA \mid \epsilon,\qquad C \to \epsilon$

- A, B, C, y S son todos anulables.
- En la nueva gramática:

$S \to \cancel{ABC} \mid AB \mid \cancel{AC} \mid \cancel{BC} \mid A \mid B \mid \cancel{C}$

$A \to aA \mid a$

$B \to bB \mid b$

Note: C es ahora sin uso.
Eliminar sus producciones.

Producciones unarias

- Una *producción unaria* (*unit production*) es aquella cuyo lado derecho consiste únicamente de una variable.

  $X \rightarrow A$, A variable no-terminal

- Estas producciones pueden eliminarse.
- Idea: Si A =>* B es una serie de producciones unarias, y $B \rightarrow \alpha$ es una producción no-unaria, entonces añadimos $A \rightarrow \alpha$ a la gramática.
- Removemos todas las producciones unarias.

Producciones unarias

- Hallar todos los pares (A, B) tales que A =>* B mediante una secuencia de producciones unarias.
- Base: (A, A) siempre, para todo A.
- Inducción: Si ya hallamos (A, B), y $B \rightarrow C$ es una producción unaria, entonces añadimos el (A, C).

Reduciendo una Gramática

- Teorema: si L es una gramática CFL, entonces existe una CFG para L – {ϵ} que satisface:
  1. No posee símbolos sin uso (*useless*).
  2. No posee producciones-ϵ.
  3. No posee producciones unarias.
- *i.e.*, todo lado derecho o es un terminal o tiene longitud $\geq$ 2.

Reduciendo una Gramática

- Algoritmo: (de reducción de CFGs)
  Input: L, una CFG.
- Hacer los siguientes pasos, en orden:
  1. Eliminar producciones-ϵ.
  2. Eliminar producciones unarias.
  3. Eliminar variables que no derivan símbolos cadenas terminales.
  4. Eliminar variables no alcanzables.

Debe hacerse primero. Puede crear producciones unarias o variables sin uso.