Gramáticas Libres de Contexto

Matemáticas Computacionales
(TC2020)

M.C. Xavier Sánchez Díaz
sax@itesm.mx

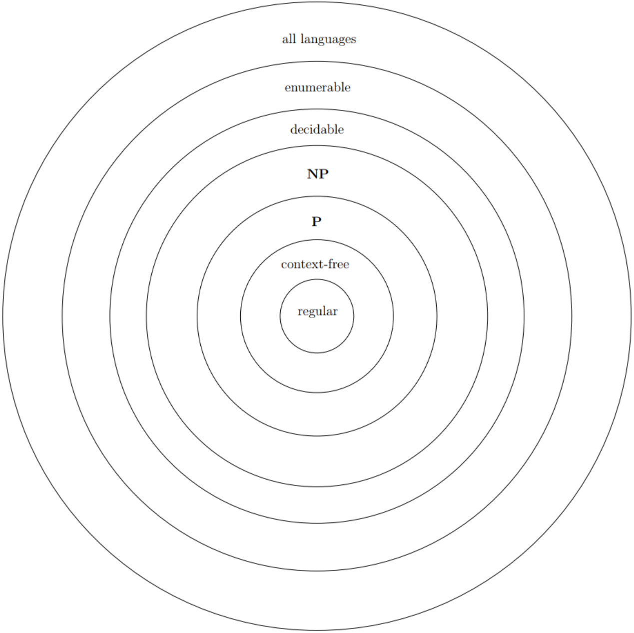

Image from Maheshwari and Smid's *Theory of Computation*, 2017

El caso $a^n b^n$

Lenguajes libres de contexto

Tesis

Sean $L = \{a^n b^n, n \in \mathbb{N}\}$ y LR el conjunto de los lenguajes regulares. Entonces, $L \notin LR$.

Como un autómata debe tener un número *finito* de estados, no hay manera de hacer que recuerde cuántas as van para saber cuántas bs deberíamos tener.

El caso $a^n b^n$

Tesis

Sean $L = \{a^n b^n, n \in \mathbb{N}\}$ y LR el conjunto de los lenguajes regulares. Entonces, $L \notin LR$.

Como un autómata debe tener un número *finito* de estados, no hay manera de hacer que recuerde cuántas as van para saber cuántas bs deberíamos tener.

Las reglas en una **GR** son del modo $A \rightarrow bC$ o $A \rightarrow b$.

En una **Gramática Libre de Contexto** (GLC) son del modo $A \rightarrow bCd$ o $A \rightarrow b$.

Además, incluimos nuevas reglas: $A \rightarrow \varepsilon$

Las reglas en una **GR** son del modo $A \to bC$ o $A \to b$.

En una **Gramática Libre de Contexto** (GLC) son del modo $A \to bCd$ o $A \to b$.

Además, incluimos nuevas reglas: $A \to \varepsilon$

Las reglas en una **GR** son del modo $A \rightarrow bC$ o $A \rightarrow b$.

En una **Gramática Libre de Contexto** (GLC) son del modo $A \rightarrow bCd$ o $A \rightarrow b$.

Además, incluimos nuevas reglas: $A \rightarrow \varepsilon$

GRs y GLCs

Diferencias entre GRs y GLCs

Las reglas en una **GR** son del modo $A \to bC$ o $A \to b$.

En una **Gramática Libre de Contexto** (GLC) son del modo $A \to bCd$ o $A \to b$.

Además, incluimos nuevas reglas: $A \to \varepsilon$

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

¿Qué patrón podemos identificar en los ejemplos?

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Estructura: Grupos anidados de una a al inicio y una b al final.

Volvemos al caso $a^n b^n$

GLCs bien formadas

Palabras aceptadas: $\varepsilon, ab, aabb, aaabbb, \ldots$

Estructura: Grupos anidados de una a al inicio y una b al final.

GLC:

1. $S \to aSb$
2. $S \to \varepsilon$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \rightarrow SS$

Reglas:

1. $S \rightarrow (S)$
2. $S \rightarrow ()$
3. $S \rightarrow SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \rightarrow SS$

Reglas:

1. $S \rightarrow (S)$
2. $S \rightarrow ()$
3. $S \rightarrow SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \to SS$

Reglas:

1. $S \to (S)$
2. $S \to ()$
3. $S \to SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \dots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\dots)))$
- Cada grupo de $(((\dots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \to SS$

Reglas:

1. $S \to (S)$
2. $S \to ()$
3. $S \to SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \to SS$

Reglas:

1. $S \to (S)$
2. $S \to ()$
3. $S \to SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \to SS$

¿Cómo concatenamos dos grupos?

Reglas:

1. $S \to (S)$
2. $S \to ()$
3. $S \to SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \to SS$

Reglas:

1. $S \to (S)$
2. $S \to ()$
3. $S \to SS$

Ejemplo: *Matching Parenthesis*

GLCs bien formadas

Ejemplos: $(), (()), ()(), (())(), \ldots$

Estructura:

- Grupos concatenados de paréntesis anidados $(((\ldots)))$
- Cada grupo de $(((\ldots)))$ es similar a $\{a^n b^n\}$
- Concatenamos usando una regla de tipo $S \rightarrow SS$

Reglas:

1. $S \rightarrow (S)$
2. $S \rightarrow ()$
3. $S \rightarrow SS$

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas
GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \to E + T$
2. $E \to T$
3. $T \to T * F$
4. $T \to F$
5. $F \to CF$
6. $F \to C$
7. $C \to 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \to E + T$
2. $E \to T$
3. $T \to T * F$
4. $T \to F$
5. $F \to CF$
6. $F \to C$
7. $C \to 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \to E + T$
2. $E \to T$
3. $T \to T * F$
4. $T \to F$
5. $F \to CF$
6. $F \to C$
7. $C \to 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \to E + T$
2. $E \to T$
3. $T \to T * F$
4. $T \to F$
5. $F \to CF$
6. $F \to C$
7. $C \to 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: expresiones aritméticas

GLCs bien formadas

Podemos formar expresiones aritméticas también, como $25 + 3 * 12$, donde la multiplicación debe tener precedencia sobre la suma:

1. $E \rightarrow E + T$
2. $E \rightarrow T$
3. $T \rightarrow T * F$
4. $T \rightarrow F$
5. $F \rightarrow CF$
6. $F \rightarrow C$
7. $C \rightarrow 0|1|2|3|4|5|6|7|8|9$

¿Cómo derivamos $25 + 3 * 12$? ¿Y $5 * 2 + 26$?

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a$, $S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

¿Cómo hacemos si es par, e.g. $abba$?

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a, S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a, S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

¿Cómo hacemos si es impar, e.g. $abbba$?

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a, S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a, S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Ejemplo: palíndromos en $\{a, b\}$

GLCs bien formadas

Ejemplos: $\varepsilon, a, aa, aaa, aba, abba, \ldots$

Caso Par: parecido a $\{a^n b^n\}$ pero con lo mismo en cada lado: $S \to aSa$ y $S \to bSb$, y usando $S \to \varepsilon$ para salir del *loop*.

Caso impar: idéntico al caso par, sólo que el *loop* se termina con un terminal no vacío: $S \to a, S \to b$.

1. $S \to aSa$
2. $S \to bSb$
3. $S \to a$
4. $S \to b$
5. $S \to \varepsilon$

Combinación de GLCs

Operaciones en GLCs

¿Cómo unimos dos GLCs?

Ejemplo: GLC para las palabras de forma $a^n b^m$ tal que $n \neq m$

Solución:

$$\{a^n b^m, n \neq m\} = \{a^n b^m, n > m\} \cup \{a^n b^m, n < m\}$$

Usamos un **símbolo inicial nuevo** para hacer dos nuevas reglas: $S_0 \rightarrow S_1$ y $S_0 \rightarrow S_2$, donde S_1 y S_2 son los símbolos iniciales de las GLCs 1 y 2, respectivamente.

La concatenación es similar. Agregamos una nueva regla: $S_0 \rightarrow S_1 S_2$.

Combinación de GLCs

Operaciones en GLCs

¿Cómo unimos dos GLCs?

Ejemplo: GLC para las palabras de forma $a^n b^m$ tal que $n \neq m$

Solución:

$$\{a^n b^m, n \neq m\} = \{a^n b^m, n > m\} \cup \{a^n b^m, n < m\}$$

Usamos un **símbolo inicial nuevo** para hacer dos nuevas reglas: $S_0 \to S_1$ y $S_0 \to S_2$, donde S_1 y S_2 son los símbolos iniciales de las GLCs 1 y 2, respectivamente.

La concatenación es similar. Agregamos una nueva regla: $S_0 \to S_1 S_2$.

Combinación de GLCs

Operaciones en GLCs

¿Cómo unimos dos GLCs?

Ejemplo: GLC para las palabras de forma $a^n b^m$ tal que $n \neq m$

Solución:

$$\{a^n b^m, n \neq m\} = \{a^n b^m, n > m\} \cup \{a^n b^m, n < m\}$$

Usamos un **símbolo inicial nuevo** para hacer dos nuevas reglas: $S_0 \rightarrow S_1$ y $S_0 \rightarrow S_2$, donde S_1 y S_2 son los símbolos iniciales de las GLCs 1 y 2, respectivamente.

La concatenación es similar. Agregamos una nueva regla: $S_0 \rightarrow S_1 S_2$.

¿Cómo unimos dos GLCs?

Ejemplo: GLC para las palabras de forma $a^n b^m$ tal que $n \neq m$

Solución:

$$\{a^n b^m, n \neq m\} = \{a^n b^m, n > m\} \cup \{a^n b^m, n < m\}$$

Usamos un **símbolo inicial nuevo** para hacer dos nuevas reglas: $S_0 \to S_1$ y $S_0 \to S_2$, donde S_1 y S_2 son los símbolos iniciales de las GLCs 1 y 2, respectivamente.

La concatenación es similar. Agregamos una nueva regla: $S_0 \to S_1 S_2$.

¿Cómo unimos dos GLCs?

Ejemplo: GLC para las palabras de forma $a^n b^m$ tal que $n \neq m$

Solución:

$$\{a^n b^m, n \neq m\} = \{a^n b^m, n > m\} \cup \{a^n b^m, n < m\}$$

Usamos un **símbolo inicial nuevo** para hacer dos nuevas reglas: $S_0 \rightarrow S_1$ y $S_0 \rightarrow S_2$, donde S_1 y S_2 son los símbolos iniciales de las GLCs 1 y 2, respectivamente.

La concatenación es similar. Agregamos una nueva regla: $S_0 \rightarrow S_1 S_2$.

Ejemplo de ambigüedad: $\#a = \#b$

Estructura: grupos de $a \dots b$ o $b \dots a$.

Reglas:

1. $S \to aSb$
2. $S \to bSa$
3. $S \to SS$
4. $S \to \varepsilon$

Probar **derivaciones** usando *DFS* para $abab = \langle 3, 1, 4, 1, 4 \rangle$ y $abab = \langle 1, 2, 4 \rangle$.

Estructura: grupos de $a \ldots b$ o $b \ldots a$.

Reglas:

1. $S \rightarrow aSb$
2. $S \rightarrow bSa$
3. $S \rightarrow SS$
4. $S \rightarrow \varepsilon$

Probar **derivaciones** usando *DFS* para $abab = \langle 3, 1, 4, 1, 4 \rangle$ y $abab = \langle 1, 2, 4 \rangle$.

Ejemplo de ambigüedad: $\#a = \#b$

Ambigüedad y Refinamiento de GLCs

Estructura: grupos de $a \dots b$ o $b \dots a$.

Reglas:

1. $S \rightarrow aSb$
2. $S \rightarrow bSa$
3. $S \rightarrow SS$
4. $S \rightarrow \varepsilon$

Probar **derivaciones** usando *DFS* para $abab = \langle 3, 1, 4, 1, 4 \rangle$ y $abab = \langle 1, 2, 4 \rangle$.

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Es posible reemplazar reglas en una GLC sin alterar el conjunto de palabras que se pueden formar con ellas.

Por ejemplo, aquellas que generen producciones vacías: $A \to \varepsilon$

Producciones 'inútiles' para reducir el tamaño de las derivaciones: $A \to B$ y $B \to a$.

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Es posible reemplazar reglas en una GLC sin alterar el conjunto de palabras que se pueden formar con ellas.

Por ejemplo, aquellas que generen producciones vacías: $A \to \varepsilon$

Producciones 'inútiles' para reducir el tamaño de las derivaciones: $A \to B$ y $B \to a$.

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Es posible reemplazar reglas en una GLC sin alterar el conjunto de palabras que se pueden formar con ellas.

Por ejemplo, aquellas que generen producciones vacías: $A \to \varepsilon$

Producciones 'inútiles' para reducir el tamaño de las derivaciones: $A \to B$ y $B \to a$.

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Existen algunos problemas con las reglas vacías:

Son **ineficientes**, pues crean un símbolo para después destruirlo.

Las reglas vacías pueden **crecer o decrecer** las derivaciones, e.g. derivaciones de ab

$$S \to SS \to SSS \to SS \to SSS \to SS \to S \to aSb \to ab$$

Existen algunos problemas con las reglas vacías:

Son **ineficientes**, pues crean un símbolo para después destruirlo.

Las reglas vacías pueden **crecer o decrecer** las derivaciones, e.g. derivaciones de ab

$$S \to SS \to SSS \to SS \to SSS \to SS \to S \to aSb \to ab$$

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Existen algunos problemas con las reglas vacías:

Son **ineficientes**, pues crean un símbolo para después destruirlo.

Las reglas vacías pueden **crecer o decrecer** las derivaciones, e.g. derivaciones de ab

$$S \to SS \to SSS \to SS \to SSS \to SS \to S \to aSb \to ab$$

Eliminación de reglas

Ambigüedad y Refinamiento de GLCs

Existen algunos problemas con las reglas vacías:

Son **ineficientes**, pues crean un símbolo para después destruirlo.

Las reglas vacías pueden **crecer o decrecer** las derivaciones, e.g. derivaciones de ab

$$S \to SS \to SSS \to SS \to SSS \to SS \to S \to aSb \to ab$$

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \rightarrow \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \rightarrow aSb$
2. $S \rightarrow \varepsilon$

$aabb = \langle 1, 1, 2 \rangle$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$

$aabb = \langle 1, 2 \rangle$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$aabb = \langle 1, 1, 2 \rangle$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$aabb = \langle 1, 2 \rangle$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$aabb = \langle 1, 1, 2 \rangle$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$aabb = \langle 1, 2 \rangle$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$aabb = \langle 1, 1, 2 \rangle$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$aabb = \langle 1, 2 \rangle$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$$aabb = \langle 1, 1, 2 \rangle$$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$$aabb = \langle 1, 2 \rangle$$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$$aabb = \langle 1, 1, 2 \rangle$$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$$aabb = \langle 1, 2 \rangle$$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas vacías

Ambigüedad y Refinamiento de GLCs

Para eliminar reglas de producciones vacías $A \to \varepsilon$, se reemplaza el lado derecho de cada ocurrencia de una regla:

1. $S \to aSb$
2. $S \to \varepsilon$

$$aabb = \langle 1, 1, 2 \rangle$$

donde podemos reemplazar cada S por ε, para generar una nueva regla:

1. $S \to aSb$
2. $S \to ab$

$$aabb = \langle 1, 2 \rangle$$

¿Qué pasa si el lenguaje $L(G)$ sí contiene a la palabra vacía ε?

Eliminación de reglas 'inútiles'

Ambigüedad y Refinamiento de GLCs

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow V$
4. $V \rightarrow a$

En lugar de tener dos reglas $S \rightarrow V$ y $V \rightarrow a$ podemos tener directamente una sola, $S \rightarrow a$:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow a$

¿Qué pasa si hay ciclos en plan $S \rightarrow T$ y $T \rightarrow S$?

Eliminación de reglas 'inútiles'

Ambigüedad y Refinamiento de GLCs

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow V$
4. $V \rightarrow a$

En lugar de tener dos reglas $S \rightarrow V$ y $V \rightarrow a$ podemos tener directamente una sola, $S \rightarrow a$:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow a$

¿Qué pasa si hay ciclos en plan $S \rightarrow T$ y $T \rightarrow S$?

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \to aSb$
2. $S \to ab$
3. $S \to V$
4. $V \to a$

En lugar de tener dos reglas $S \to V$ y $V \to a$ podemos tener directamente una sola, $S \to a$:

1. $S \to aSb$
2. $S \to ab$
3. $S \to a$

¿Qué pasa si hay ciclos en plan $S \to T$ y $T \to S$?

Eliminación de reglas 'inútiles'

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow V$
4. $V \rightarrow a$

En lugar de tener dos reglas $S \rightarrow V$ y $V \rightarrow a$ podemos tener directamente una sola, $S \rightarrow a$:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow a$

¿Qué pasa si hay ciclos en plan $S \rightarrow T$ y $T \rightarrow S$?

Eliminación de reglas 'inútiles'

Ambigüedad y Refinamiento de GLCs

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow V$
4. $V \rightarrow a$

En lugar de tener dos reglas $S \rightarrow V$ y $V \rightarrow a$ podemos tener directamente una sola, $S \rightarrow a$:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow a$

¿Qué pasa si hay ciclos en plan $S \rightarrow T$ y $T \rightarrow S$?

Eliminación de reglas 'inútiles'

Ambigüedad y Refinamiento de GLCs

Para eliminar las reglas 'inútiles' (o pasos intermedios) en una GLC hay que ir **conectando** los extremos:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow V$
4. $V \rightarrow a$

En lugar de tener dos reglas $S \rightarrow V$ y $V \rightarrow a$ podemos tener directamente una sola, $S \rightarrow a$:

1. $S \rightarrow aSb$
2. $S \rightarrow ab$
3. $S \rightarrow a$

¿Qué pasa si hay ciclos en plan $S \rightarrow T$ y $T \rightarrow S$?