



**UNIVERSIDAD
DE LA LAGUNA**

GRADO EN INGENIERÍA INFORMÁTICA

Computabilidad y Algoritmia

PRÁCTICA 7. Diseño y simplificación de gramáticas en JFLAP

Presentado por:

Aarón Ramírez Valencia

alu0101438238@ull.edu.es

25 / 10 / 2024

Recordatorio:

Una gramática regular G es una 4-tupla $G = (\Sigma, N, S, P)$, donde:

- Σ es un alfabeto.
- N es un conjunto de símbolos no terminales.
- S es un no terminal llamado símbolo inicial (start) o de arranque.
- P es un conjunto de reglas de sustitución denominadas producciones, tales que (caben dos alternativas):
 - O bien todas las producciones son lineales por la derecha: $A \rightarrow uB|v$
($A, B \in N$ y $u, v \in \Sigma^*$)
 - O todas las producciones son lineales por la izquierda: $A \rightarrow Bu|v$
($A, B \in N$ y $u, v \in \Sigma^*$)

Ejercicios

1. Diseñar una gramática independiente del contexto que genere el lenguaje

$$L = \{a^n b^n \mid n \geq 0\}$$

$$L = \{a^n b^n \mid n \geq 0\}$$

$$S \rightarrow a S b \mid \epsilon$$

Eliminar prod vacías (que tienen ϵ ó pueden llegar a tenerlo)

$$\textcircled{1} V' = \emptyset$$

No hay prod. vacías $\Rightarrow S \rightarrow a S b \mid \epsilon \mid ab$
(sin ser S)

$$V' = \{S\}$$

$H = \{S\}$ S es generadora de cadenas

Eliminar prod. unitarias

$$\text{Unitarias} = \{S\}$$

No hay prod. unitarias $\Rightarrow S \rightarrow a S b \mid \epsilon \mid ab$

$X \rightarrow Y \rightarrow$ No tengo

Eliminar variables inútiles (símbolos y prod inútiles)

Generadoras = $\{S\}$ No tengo variables = $S \rightarrow a S b \mid ab \mid \epsilon$
Pues tengo en $S \rightarrow ab$.

File Input Test Convert Help

Editor **Lambda Removal**

Table Text Size

LHS		RHS
S	→	aSb
S	→	λ

Do Step Do All **Proceed** Export

Lambda removal complete.
"Proceed" or "Export" available.
Set that derives lambda: [S]

Delete Complete Selected

LHS		RHS
S	→	aSb
S	→	ab

File Input Test Convert Help

Editor **Chomsky Converter**

LHS		RHS
S	→	aSb
S	→	ab

Convert Selected Do All **What's Left?** Export

Welcome to the Chomsky converter.
2 production(s) must be converted.

LHS		RHS
S	→	aSb
S	→	ab

Table Text Size

Start **Pause** **Step** **Noninverted Tree**

Input **ab**

String accepted! 4 nodes generated.

LHS		RHS
S	→	aSb
S	→	λ

```

graph TD
    S1((S)) --- a((a))
    S1 --- S2((S))
    S1 --- b((b))
    S2 --- lambda((λ))
  
```

Derived λ from S. Derivations complete.

Table Text Size

Start Pause Step Noninverted Tree

Input: aabb
String accepted! 5 nodes generated.

LHS		RHS
S	→	aSb
S	→	λ

Derived λ from S. Derivations complete.

File Input Test Convert Help

Editor Brute Parser

Table Text Size

Start Pause Step Noninverted Tree

Input: aaabbb
String accepted! 6 nodes generated.

LHS		RHS
S	→	aSb
S	→	λ

Derived λ from S. Derivations complete.

2. Diseñar una gramática independiente del contexto que genere el lenguaje
 $L = \{a^n b^m \mid n, m \geq 0, n \neq m\}$

$$L = \{a^n b^m \mid n, m \geq 0, n \neq m\}$$

$$S \rightarrow A \mid B$$

$$A \rightarrow aA \mid aAb \mid a$$

$$B \rightarrow Bb \mid aBb \mid b$$

1º Eliminar prod. vacías

$$\text{Anulables} = \{ \}$$

Que tienen ϵ o pueden tenerlo

No hay
= prod. vacías

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow aA \mid aAb \mid a \\ B &\rightarrow Bb \mid aBb \mid b \end{aligned}$$

2º Eliminar producciones unitarias

$$\text{Unitarias} = \{(S, A), (S, B)\}$$

$S \rightarrow A$ sustituyo la prod.
 $S \rightarrow B$ por lo que podría ser.

$$S \rightarrow aA \mid aAb \mid a \mid Bb \mid aBb \mid b$$

$$A \rightarrow aA \mid aAb \mid a$$

$$B \rightarrow Bb \mid aBb \mid b$$

3º Eliminar variables inútiles

$$\text{Generadoras} = \{S, A, B\}$$

No hay variables inútiles

$$S \rightarrow aA \mid aAb \mid a \mid Bb \mid aBb \mid b$$

$$A \rightarrow aA \mid aAb \mid a$$

$$B \rightarrow Bb \mid aBb \mid b$$

File Input Test Convert Help

Editor Unit Removal

LHS	RHS
S	→ A
S	→ B
A	→ aA
A	→ aAb
A	→ a
B	→ Bb
B	→ aBb
B	→ b

Do Step Do All Proceed Export

Unit removal complete.
"Proceed" or "Export" available.

Automaton Size

Delete Complete Selected

LHS	RHS
A	→ aA
A	→ aAb
A	→ a
B	→ Bb
B	→ aBb
B	→ b
S	→ aAb
S	→ b
S	→ a

File Input Test Convert Help

Editor Unit Removal Chomsky Converter

LHS	RHS
S	→ aAb
S	→ b
S	→ a
S	→ aA
S	→ Bb
S	→ aBb
B	→ b
B	→ aBb
B	→ Bb
A	→ a
A	→ aAb
A	→ aA

Convert Selected Do All What's Left? Export

Welcome to the Chomsky converter.
8 production(s) must be converted.

LHS	RHS
S	→ aAb
S	→ b
S	→ a
S	→ aA
S	→ Bb
S	→ aBb
B	→ b
B	→ aBb
B	→ Bb
A	→ a
A	→ aAb
A	→ aA

File Input Test Convert Help

Editor Brute Parser

Table Text Size

Start Pause Step Noninverted Tree

Input: a

String accepted! 2 nodes generated.

LHS	RHS
S	→ aAb
S	→ b
S	→ a
S	→ aA
S	→ Bb
S	→ aBb
B	→ b
B	→ aBb

Derived a from S. Derivations complete.

File Input Test Convert Help

Editor Brute Parser

Table Text Size

Start Pause Step Noninverted Tree

Input: aab
String accepted! 7 nodes generated.

LHS		RHS
S	→	aAb
S	→	b
S	→	a
S	→	aA
S	→	Bb
S	→	aBb
B	→	b
B	→	aBb

Derived a from A. Derivations complete.

File Input Test Convert Help

Editor Brute Parser

Table Text Size

Start Pause Step Noninverted Tree

Input: aabbb
String accepted! 17 nodes generated.

LHS		RHS
S	→	aAb
S	→	b
S	→	a
S	→	aA
S	→	Bb
S	→	aBb
B	→	b
B	→	aBb

Derived b from B. Derivations complete.

3. Diseñar una gramática independiente del contexto para el lenguaje
 $L = \{ww^l \mid w \in \{a, b\}^*\}$

$$L = \{ ww^T \mid w \in \{a, b\}^* \}$$

$$S \rightarrow aSa \mid bSb \mid \epsilon$$

1º Eliminar prod vacías.

Anulables = $\{S\}$ No hay prod. vacías = $S \rightarrow aSa \mid bSb \mid \epsilon \mid aa \mid bb$
(sin ser S)

2º Eliminar prod unitarias

Unitarias $\{\emptyset\}$ No hay prod. unitarias = $S \rightarrow aSa \mid bSb \mid \epsilon \mid aa \mid bb$

3º Eliminar prod / variables inútiles

Generadoras = $\{S\}$ No hay prod. inútiles = $S \rightarrow aa \mid bb \mid aSa \mid bSb \mid \epsilon$

File Input Test Convert Help

Editor Lambda Removal

Table Text Size

LHS		RHS
S	→	aSa
S	→	bSb
S	→	λ

Do Step Do All Proceed Export

Lambda removal complete.
"Proceed" or "Export" available.
Set that derives lambda: [S]

Delete Complete Selected

LHS		RHS
S	→	aSa
S	→	bSb
S	→	aa
S	→	bb

File Input Test Convert Help

Editor Lambda Removal Chomsky Converter

LHS		RHS
S	→	aSa
S	→	bSb
S	→	aa
S	→	bb

Convert Selected Do All What's Left? Export

Welcome to the Chomsky converter.
4 production(s) must be converted.

LHS		RHS
S	→	aSa
S	→	bSb
S	→	aa
S	→	bb

Table Text Size

Start Pause Step Noninverted Tree

Input aa

String accepted! 4 nodes generated.

LHS		RHS
S	→	aSa
S	→	bSb
S	→	λ

```

graph TD
    S1((S)) --- a1((a))
    S1 --- S2((S))
    S1 --- a2((a))
    S2 --- lambda((λ))
  
```

Derived λ from S. Derivations complete.

Table text Size

Start Pause Step Noninverted Tree

Input: aaaa
String accepted! 6 nodes generated.

LHS		RHS
S	→	aSa
S	→	bSb
S	→	λ

Derived λ from S. Derivations complete.

Table text Size

Start Pause Step Noninverted Tree

Input: abbbba
String accepted! 6 nodes generated.

LHS		RHS
S	→	aSa
S	→	bSb
S	→	λ

Derived λ from S. Derivations complete.

4. Diseñar una gramática independiente del contexto que genere el lenguaje
 $L = \{a^n b^m c^n \mid n \geq 0, m \text{ impar}\}$

$$L = \{a^n b^m c^n \mid n \geq 0, m \text{ impar}\}$$

$$S \rightarrow aSc \mid A$$

$$A \rightarrow bAb \mid b$$

1º Eliminar prod vacías

$$\text{Anulables} = \{\epsilon\} \text{ No tengo prod vacías} = \begin{array}{l} S \rightarrow aSc \mid A \\ A \rightarrow bAb \mid b \end{array}$$

2º Eliminar prod unitarias

$$\text{Unitarias } \{(S, A)\} = \begin{array}{l} S \rightarrow aSc \mid bAb \mid b \\ S \rightarrow A \\ A \rightarrow bAb \mid b \end{array}$$

3º Eliminar prod/variables inútiles

$$\begin{array}{l} \text{Generadoras} = \{S, A\} \\ \text{No tengo prod/var inútiles} \end{array} = \begin{array}{l} S \rightarrow aSc \mid bAb \mid b \\ A \rightarrow bAb \mid b \end{array}$$

File Input Test Convert Help

Unit Removal

LHS	RHS
S	→ aSc
S	→ A
A	→ b
A	→ bAb

Do Step Do All Proceed Export

Unit removal complete.
"Proceed" or "Export" available.

Automaton Size

Delete Complete Selected

LHS	RHS
S	→ aSc
A	→ b
A	→ bAb
S	→ b
S	→ bAb

File Input Test Convert Help

Unit Removal Chomsky Converter

LHS	RHS
S	→ aSc
S	→ b
S	→ bAb
A	→ bAb
A	→ b

Convert Selected Do All What's Left? Export

Welcome to the Chomsky converter.
3 production(s) must be converted.

LHS	RHS
S	→ aSc
S	→ b
S	→ bAb
A	→ bAb
A	→ b

Start Pause Step Noninverted Tree

Input **b**

String accepted! 3 nodes generated.

LHS	RHS
S	→ aSc
S	→ A
A	→ b
A	→ bAb

Derived b from A. Derivations complete.

Start Pause Step Noninverted Tree

Input: `abbbbc`
String accepted! 6 nodes generated.

LHS		RHS
S	→	aSc
S	→	A
A	→	b
A	→	bAb

```

graph TD
    S1((S)) --- a1((a))
    S1 --- S2((S))
    S1 --- c1((c))
    S2 --- A1((A))
    A1 --- b1((b))
    A1 --- A2((A))
    A1 --- b2((b))
    A2 --- b3((b))
  
```

Derived b from A. Derivations complete.

Start Pause Step Noninverted Tree

Input: `aabbbcc`
String accepted! 8 nodes generated.

LHS		RHS
S	→	aSc
S	→	A
A	→	b
A	→	bAb

```

graph TD
    S1((S)) --- a1((a))
    S1 --- S2((S))
    S1 --- c1((c))
    S2 --- a2((a))
    S2 --- S3((S))
    S2 --- c2((c))
    S3 --- A1((A))
    A1 --- b1((b))
    A1 --- A2((A))
    A1 --- b2((b))
    A2 --- b3((b))
  
```

Derived b from A. Derivations complete.

5. Diseñar una gramática independiente del contexto que genere el lenguaje
 $L = \{a^n b^m c^m d^n \mid n, m \geq 1\}$

$$L = \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

S

$$S \rightarrow a S d \mid a X d$$

~~$$S \rightarrow a X d$$~~

$$X \rightarrow b X c \mid bc$$

1º Eliminar prod vacías

$$\text{Anulables} = \{ \} = S \rightarrow a S d \mid a X d$$

$$\text{No tengo prod vacías} \quad X \rightarrow b X c \mid bc$$

2º Eliminar prod unitarias

$$\text{Unitarias} = \{ \} = S \rightarrow a S d \mid a X d$$

$$\text{No tengo prod unitarias} \quad X \rightarrow b X c \mid bc$$

3º Eliminar prod/variables inútiles

$$\text{Generadoras } \{S, X\}$$

Se tiene la prod $X \rightarrow bc$ por lo que X es generadora y en $S \rightarrow a X b$ y S deriva a una generadora (X)

$$S \rightarrow a S d \mid a X d$$

$$X \rightarrow b X c \mid bc$$

E

Start Pause Step Noninverted Tree

Input `abbccd`
String accepted! 5 nodes generated.

LHS		RHS
S	→	aSd
S	→	aXd
X	→	bXc
X	→	bc

Derived bc from X. Derivations complete.

Start Pause Step Noninverted Tree

Input `aabbccdd`
String accepted! 7 nodes generated.

LHS		RHS
S	→	aSd
S	→	aXd
X	→	bXc
X	→	bc

Derived bc from X. Derivations complete.

6. Diseñar una gramática independiente del contexto que genere el lenguaje
 $L = \{a^n b^m \mid n > m \geq 0\}$

$$L = \{a^n b^m \mid n > m \geq 0\}$$

$$S \rightarrow aSb \mid aS \mid a$$

1º) Eliminar prod vacíos

Amables = $\{ \}$ = $S \rightarrow aSb \mid aS \mid a$
 No tengo prod vacíos

2º) Eliminar prod unitarios

Unitarios $\{ \}$ = $S \rightarrow aSb \mid aS \mid a$
 No tengo prod unitarios

3º) Eliminar prod / var inútiles

Generadoras = $\{S\}$ = $S \rightarrow aSb \mid aS \mid a$
 No tengo prod / var inútiles


LHS		RHS	
S	→	aSb	
S	→	aS	
S	→	a	

LHS		RHS	
S	→	aSb	
S	→	aS	
S	→	a	

Start Pause Step Noninverted Tree

Input: a
String accepted! 2 nodes generated.

LHS		RHS
S	→	aSb
S	→	aS
S	→	a

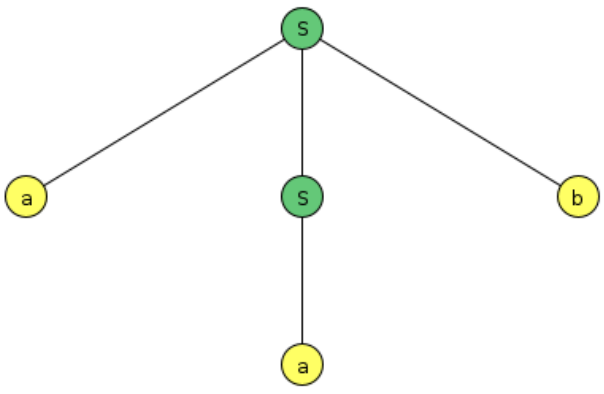


Derived a from S. Derivations complete.

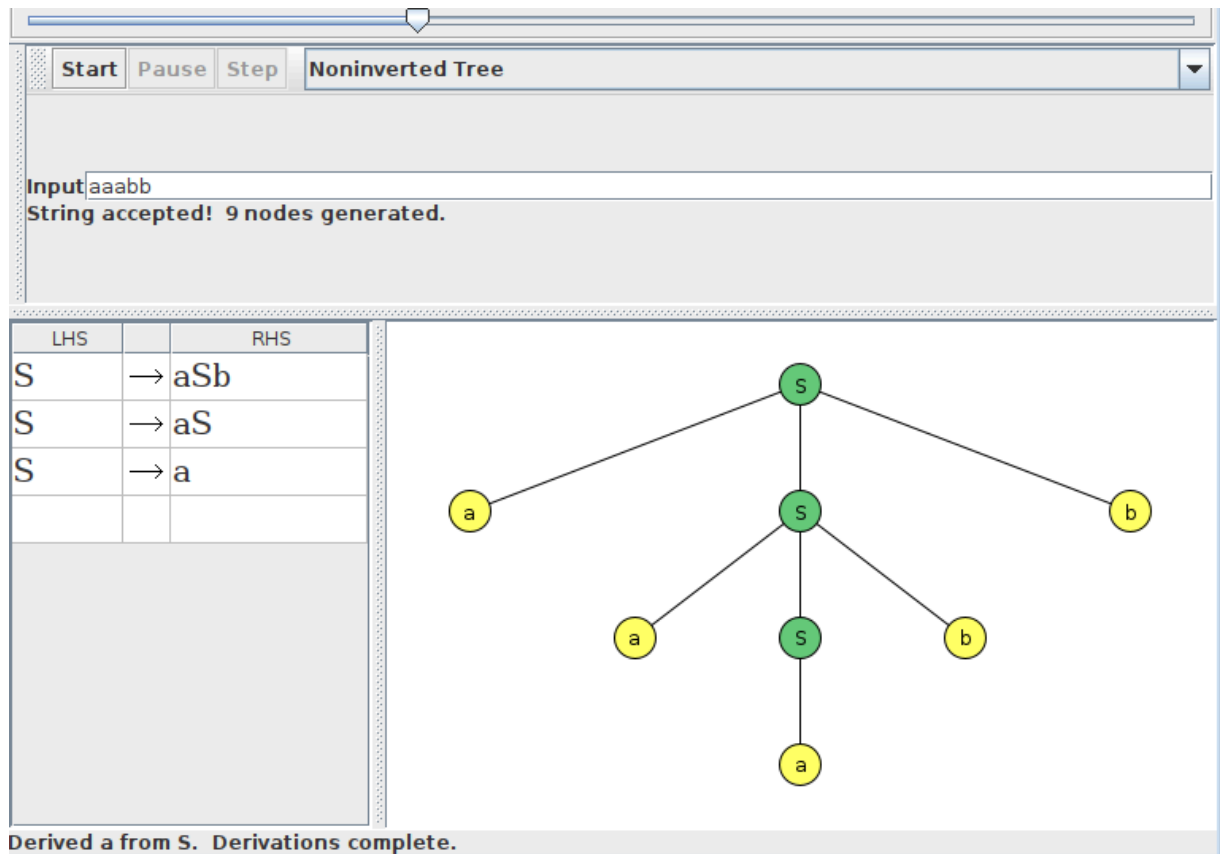
Start Pause Step Noninverted Tree

Input: aab
String accepted! 5 nodes generated.

LHS		RHS
S	→	aSb
S	→	aS
S	→	a



Derived a from S. Derivations complete.



7. Diseñar una gramática independiente del contexto que genere el lenguaje
 $L = \{a^i b^j c^{i+j} \mid i, j \geq 1, i + j \text{ par}\}$

$$L\{a^i b^j c^{i+j} \mid i, j \geq 1, i+j \text{ par}\}$$

$$S \rightarrow aAc$$

$$A \rightarrow aAc \mid bBc \mid bc$$

$$B \rightarrow bBc \mid bc$$

1º Eliminar prod. vacías

$$\text{Anulables} = \{ \}$$

No tengo prod vacías

$$S \rightarrow aAc$$

$$A \rightarrow aAc \mid bBc \mid bc$$

$$B \rightarrow bBc \mid bc$$

2º Eliminar prod. unitarias

$$\text{Unitarias} = \{ \}$$

No tengo producciones

$$S \rightarrow aAc$$

$$A \rightarrow aAc \mid bBc \mid bc$$

$$B \rightarrow bBc \mid bc$$

3º Eliminar prod. inútiles

$$\text{Generadoras} = \{S, A, B\}$$

No tengo prod/voz inútiles

$$S \rightarrow aAc$$

$$A \rightarrow aAc \mid bBc \mid bc$$

$$B \rightarrow bBc \mid bc$$

File Input Test Convert Help

Editor Chomsky Converter

LHS	RHS
S	→ aAc
A	→ aAc
A	→ bBc
A	→ bc
B	→ bBc
B	→ bc

Convert Selected Do All What's Left? Export

Welcome to the Chomsky converter.
6 production(s) must be converted.

LHS	RHS
S	→ aAc
A	→ aAc
A	→ bBc
A	→ bc
B	→ bBc
B	→ bc

Start Pause Step Noninverted Tree

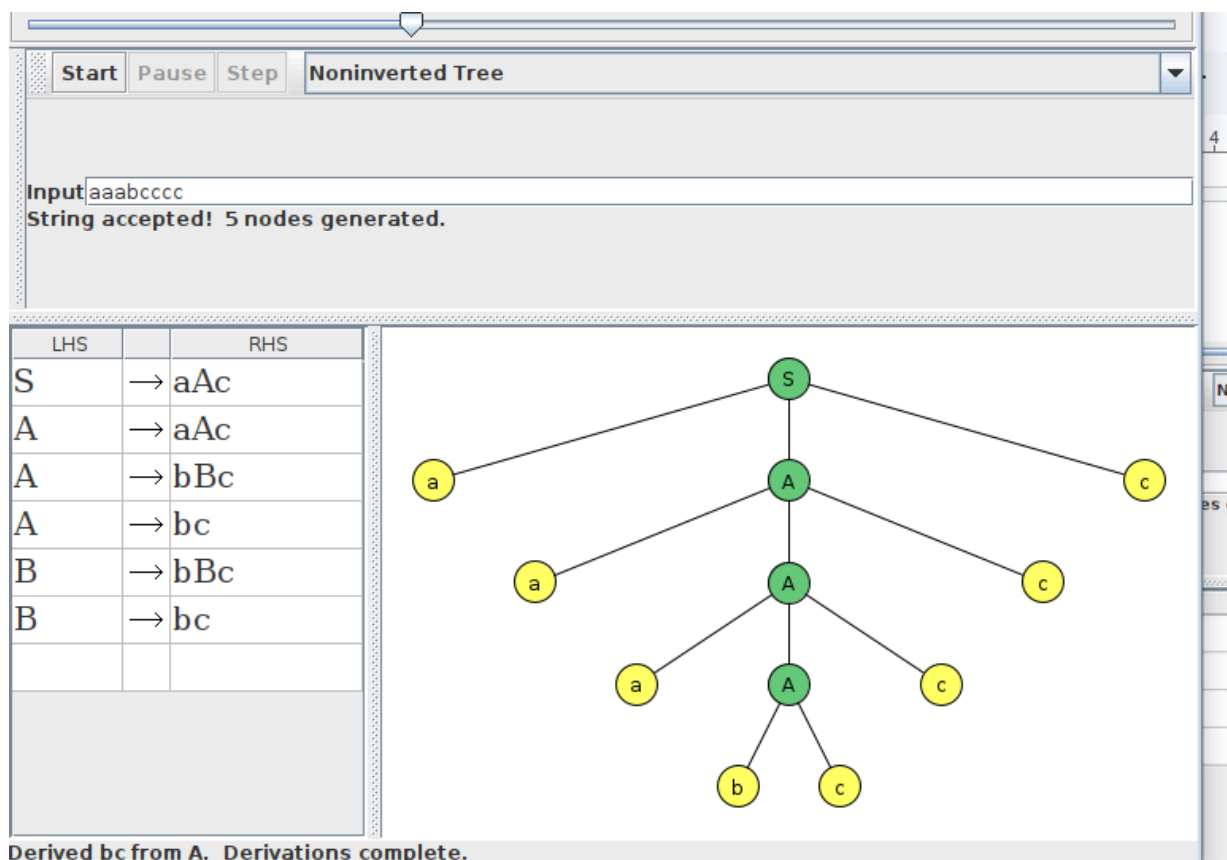
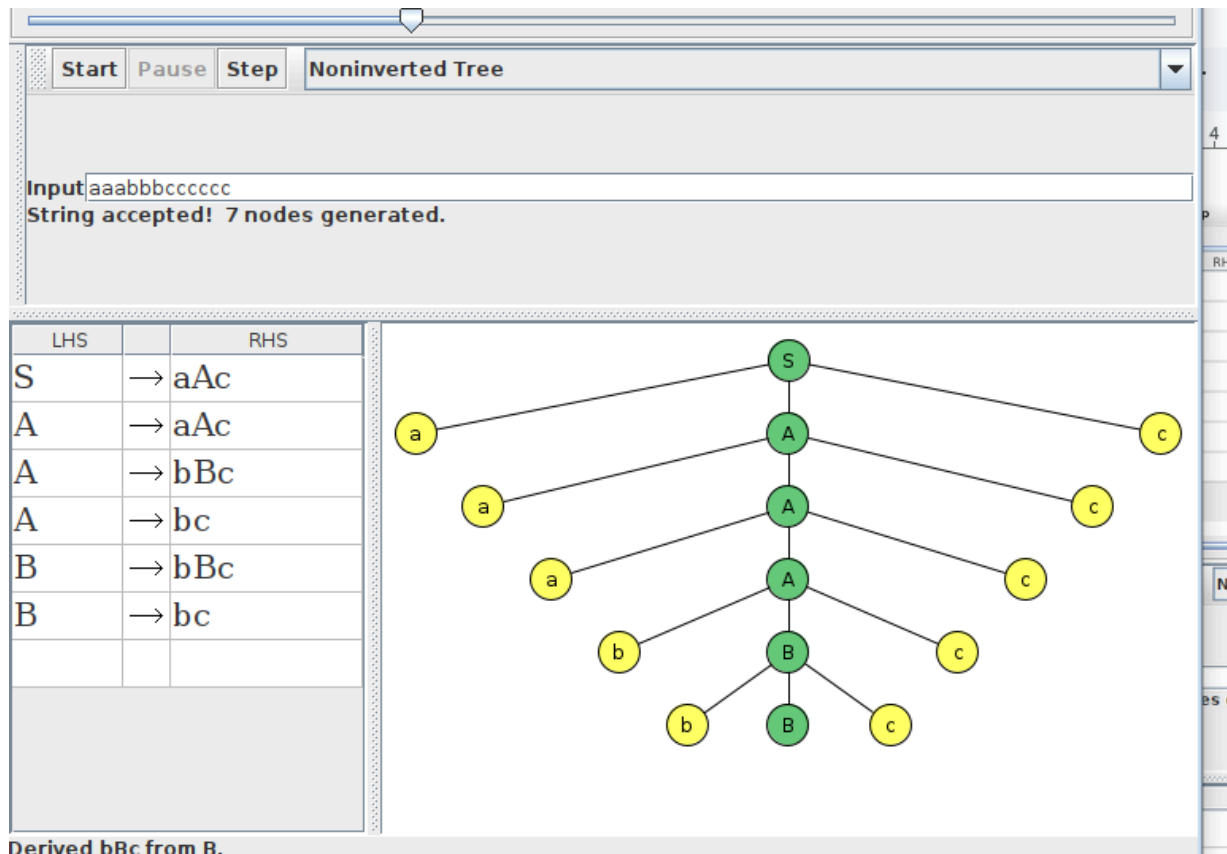
Input: abcc
String accepted! 3 nodes generated.

LHS	RHS
S	→ aAc
A	→ aAc
A	→ bBc
A	→ bc
B	→ bBc
B	→ bc

```

graph TD
    S((S)) --- a((a))
    S --- A((A))
    S --- c1((c))
    A --- b((b))
    A --- c2((c))
  
```

Derived bc from A. Derivations complete.



8. Diseñar una gramática independiente del contexto que genere el lenguaje de las expresiones booleanas con los operadores AND, OR, y NOT, usando paréntesis para agrupar. Ejemplos: “(true AND false)”, “NOT (true OR false)”, “true AND (false OR true)”.

8. Expresiones booleanas con AND, OR y NOT

$$S \rightarrow E$$

$$E \rightarrow E \text{ OR } E \mid E \text{ AND } E \mid \text{NOT } E \mid (E) \mid \text{true} \mid \text{false}$$

1º Eliminar prod. vacías

$$\text{Ambles} = \{ \}$$

No tengo prod vacías

$$S \rightarrow \bar{E}$$

$$\bar{E} \rightarrow E \text{ OR } E \mid E \text{ AND } E \mid \text{NOT } E \mid (E) \mid \text{true} \mid \text{false}$$

2º Eliminar prod unitarias

$$\text{Unitarias} = \{ S, E \}$$

$$S \rightarrow E$$

\rightarrow

$$S \rightarrow E \text{ OR } E \mid E \text{ AND } E \mid \text{NOT } E \mid (E) \mid \text{true} \mid \text{false}$$

$$E \rightarrow E \text{ OR } E \mid E \text{ AND } E \dots$$

3º Eliminar prod ^{no} inútiles

$$\text{Generadores } \{ S, E \}$$

No tengo prod/var inútiles

$$S \rightarrow E \text{ OR } E \mid E \text{ AND } E \mid \text{NOT } E \mid (E) \mid \text{true} \mid \text{false}$$

$$E \rightarrow E \text{ OR } E \mid E \text{ AND } E \mid \text{NOT } E \mid (E) \mid \text{true} \mid \text{false}$$

LHS	RHS
S	→ E
E	→ E or E
E	→ E and E
E	→ not E
E	→ (E)
E	→ t
E	→ f

Do Step Do All Proceed Export
Unit removal complete.
Proceed or Export available.

```

graph LR
    S((S)) --> E((E))
  
```

Automaton Size

Delete Complete Selected

LHS	RHS
E	→ E or E
E	→ E and E
E	→ not E
E	→ (E)
E	→ t
E	→ f
S	→ E and E
S	→ E or E
S	→ f
S	→ t
S	→ not F

LHS	RHS
S	→ E
E	→ E or E
E	→ E and E
E	→ not E
E	→ (E)
E	→ t
E	→ f

Do Step Do All Proceed Export
Unit removal complete.
Proceed or Export available.

```

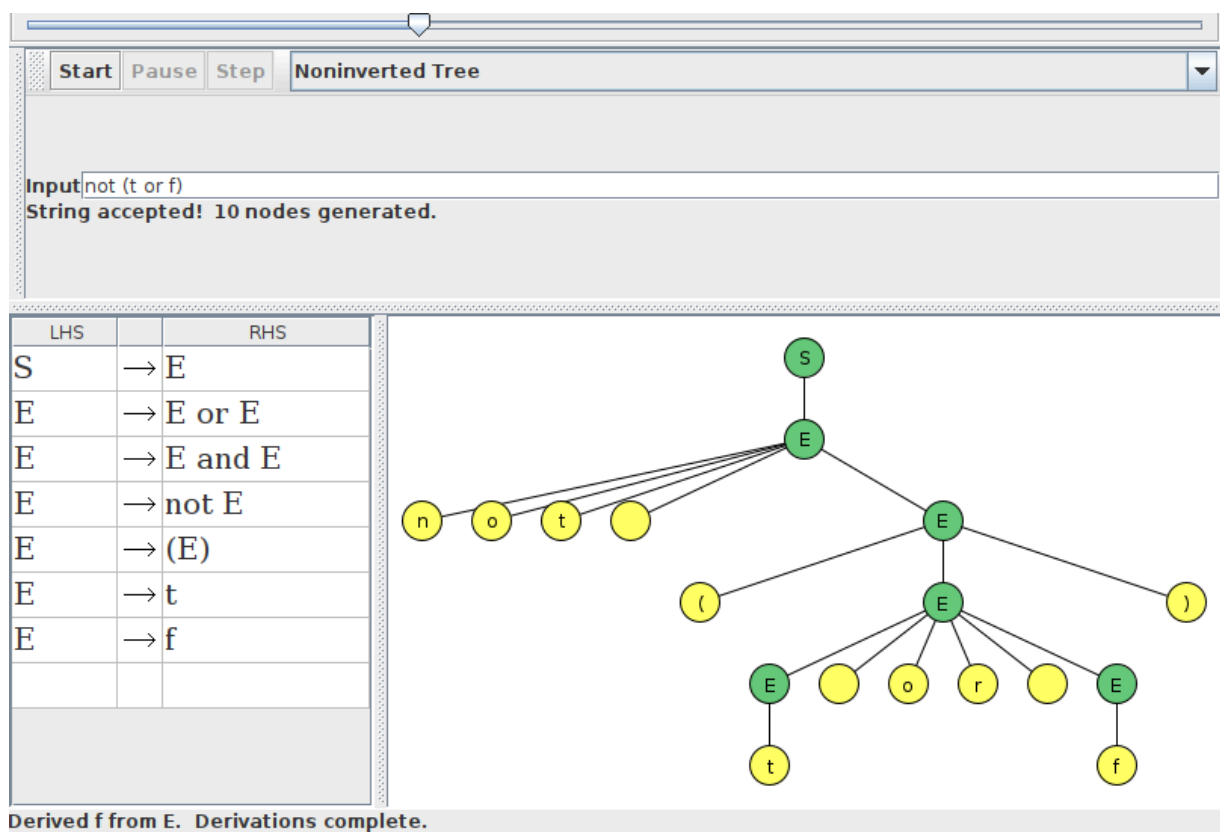
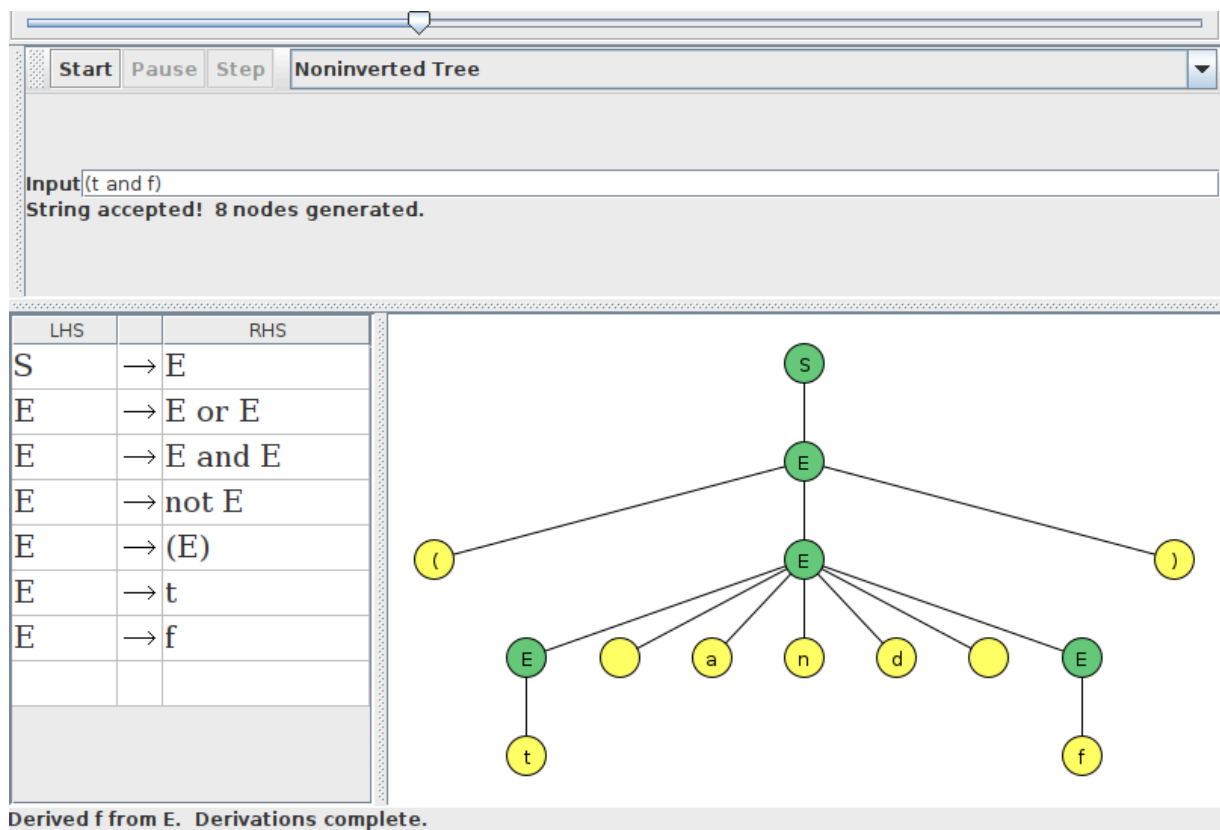
graph LR
    S((S)) --> E((E))
  
```

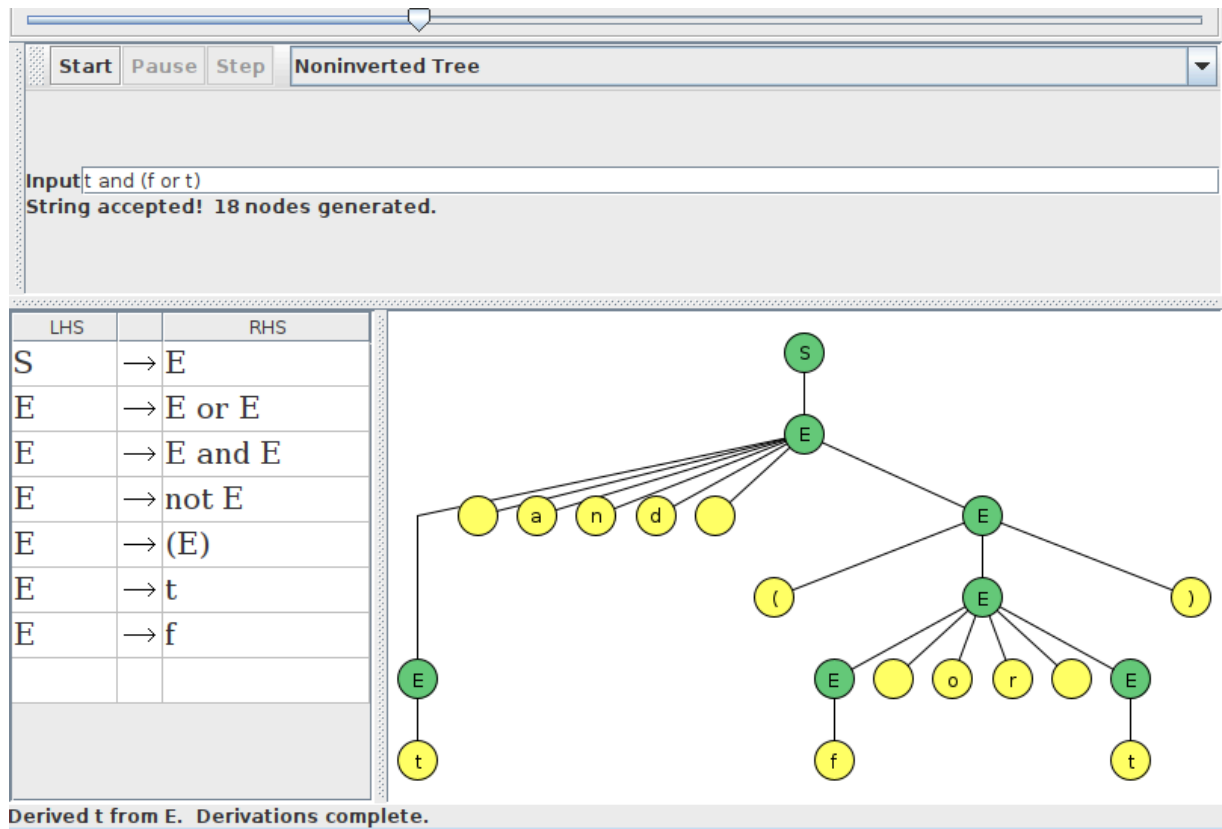
Automaton Size

Delete Complete Selected

LHS	RHS
E	→ E or E
E	→ E and E
E	→ not E
E	→ (E)
E	→ t
E	→ f
S	→ E and E
S	→ E or E
S	→ f
S	→ t
S	→ not F

Illegal Grammar
Grammar has the (character, which is reserved.
Aceptar





9. Diseñar una gramática independiente del contexto que genere expresiones aritméticas simples con suma y multiplicación, utilizando los símbolos +, *, (,) y los números 0, 1, etc. Ejemplos: “1+2”, “(1+2)*3”, “4*(5+6)”.

9) expresiones aritméticas con suma y multiplicación

$S \rightarrow NON$

$N \rightarrow NON \mid (NON) \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid 0$

$O \rightarrow + \mid *$

1º Eliminar prod vacías

Amables = $\{ \}$ =

No tengo prod vacías

$S \rightarrow NON$

$N \rightarrow NON \mid (NON) \mid 1 \mid 2 \mid 3 \dots \mid 0$

$O \rightarrow + \mid *$

2º Eliminar prod unitarias

Unitarias = $\{ S, N, O \}$

~~$S \rightarrow NON$~~
 ~~$N \rightarrow NON$~~

No tengo
prod unitarias =

$S \rightarrow NON$

~~$N \rightarrow NON$~~ $N \rightarrow (NON) \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid 0$

$O \rightarrow + \mid *$

3º Eliminar prod / var inútiles

Generadoras = $\{ S, N, O \}$ =

No tengo prod / var inútiles

= $S \rightarrow NON$

$N \rightarrow NON \mid (NON) \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid 0$

$O \rightarrow + \mid *$

Table Text Size

LHS		RHS
S	→	NON
N	→	(NON)
N	→	NON
N	→	1
N	→	2
N	→	3
N	→	4
N	→	5
N	→	6
N	→	7
N	→	8
N	→	9
N	→	0
O	→	+
O	→	*

Illegal Grammar ×

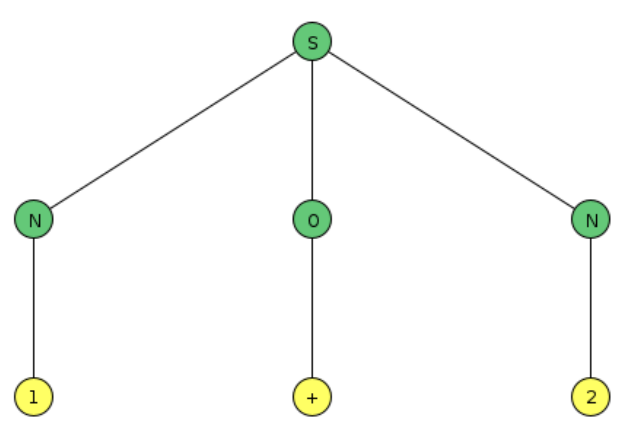
Grammar has the (character, which is reserved.

Table Text Size

Start Pause Step Noninverted Tree

Input: 1+2
String accepted! 9 nodes generated.

LHS		RHS
S	→	NON
N	→	(NON)
N	→	NON
N	→	1
N	→	2
N	→	3
N	→	4
N	→	5
N	→	6
N	→	7
N	→	8
N	→	9



```

graph TD
    S((S)) --- N1((N))
    S --- O((O))
    S --- N2((N))
    N1 --- 1((1))
    O --- plus((+))
    N2 --- 2((2))
  
```

Derived 2 from N. Derivations complete.

Table Text Size

Start Pause Step Noninverted Tree

Input: $(1+2)*3$
String accepted! 99 nodes generated.

LHS	RHS
S	→ NON
N	→ (NON)
N	→ NON
N	→ 1
N	→ 2
N	→ 3
N	→ 4
N	→ 5
N	→ 6
N	→ 7
N	→ 8
N	→ 0

```

graph TD
    S((S)) --- N1((N))
    S --- O1((O))
    S --- N2((N))
    N1 --- LParen("(")
    N1 --- N3((N))
    N1 --- O2((O))
    N1 --- N4((N))
    N1 --- RParen(")")
    N3 --- 1((1))
    O2 --- Plus("+")
    N4 --- 2((2))
    O1 --- Star("*")
    N2 --- 3((3))
  
```

Derived 3 from N. Derivations complete.

Table Text Size

Start Pause Step Noninverted Tree

Input: $4*(5+6)$
String accepted! 48 nodes generated.

LHS	RHS
S	→ NON
N	→ (NON)
N	→ NON
N	→ 1
N	→ 2
N	→ 3
N	→ 4
N	→ 5
N	→ 6
N	→ 7
N	→ 8
N	→ 0

```

graph TD
    S((S)) --- N1((N))
    S --- O1((O))
    S --- N2((N))
    N1 --- 4((4))
    O1 --- Star("*")
    N2 --- LParen("(")
    N2 --- N3((N))
    N2 --- O2((O))
    N2 --- N4((N))
    N2 --- RParen(")")
    N3 --- 5((5))
    O2 --- Plus("+")
    N4 --- 6((6))
  
```

Derived 6 from N. Derivations complete.

10. Diseñar una gramática independiente del contexto que genere el lenguaje de listas anidadas usando corchetes, como en los lenguajes de programación. Ejemplos: [], [], [], [[1,2],[3,4]].

10. Listas anidadas usando corchetes

$$S \rightarrow [L] \mid []$$

$$L \rightarrow E \mid E, L$$

$$E \rightarrow S \mid D$$

$$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

1º Eliminar prod vacías

Anulables = $\{ \}$ =
No tengo prod vacías

$$S \rightarrow [L] \mid []$$

$$L \rightarrow E \mid E, L$$

$$E \rightarrow S \mid D$$

$$D \rightarrow 0 \mid 1 \mid \dots \mid 9$$

2º Eliminar prod unitarias

$$\text{Unitarias} = \{ (L, E), (E, S), (E, D) \}$$

$$L \rightarrow E$$

$$E \rightarrow S$$

$$E \rightarrow D$$

$$S \rightarrow [L]$$

$$S \rightarrow [L] \mid []$$

$$L \rightarrow E, L \mid [L] \mid [] \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$E \rightarrow [L] \mid [] \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

3º Eliminar prod / Por inútiles

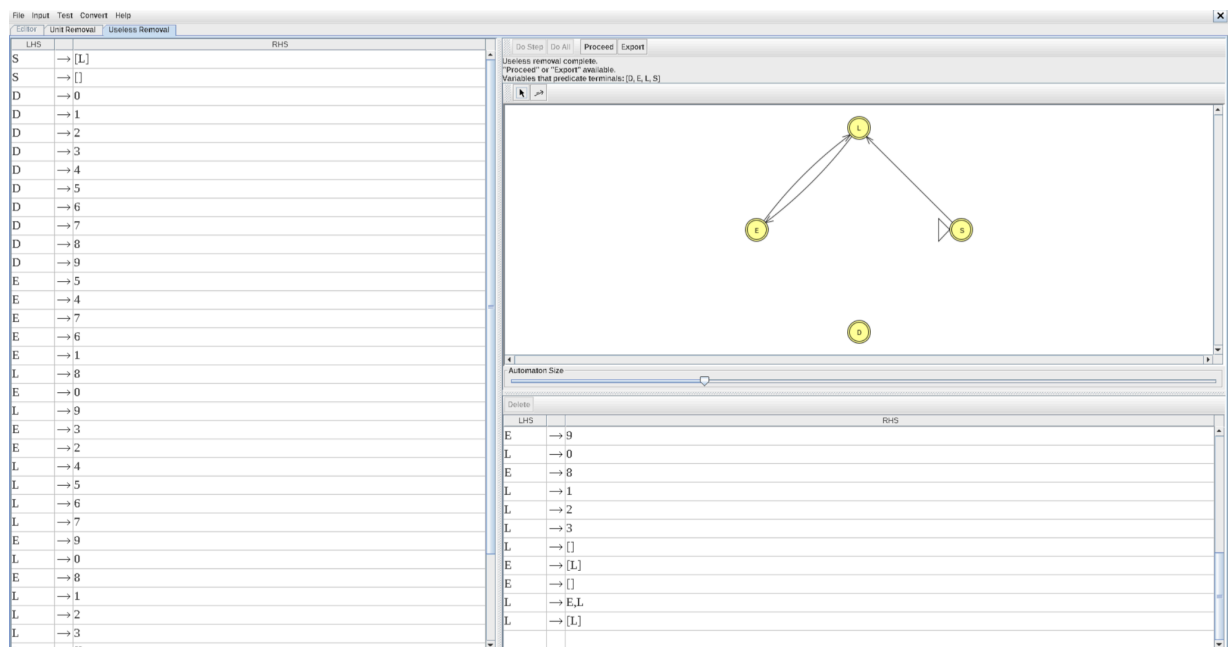
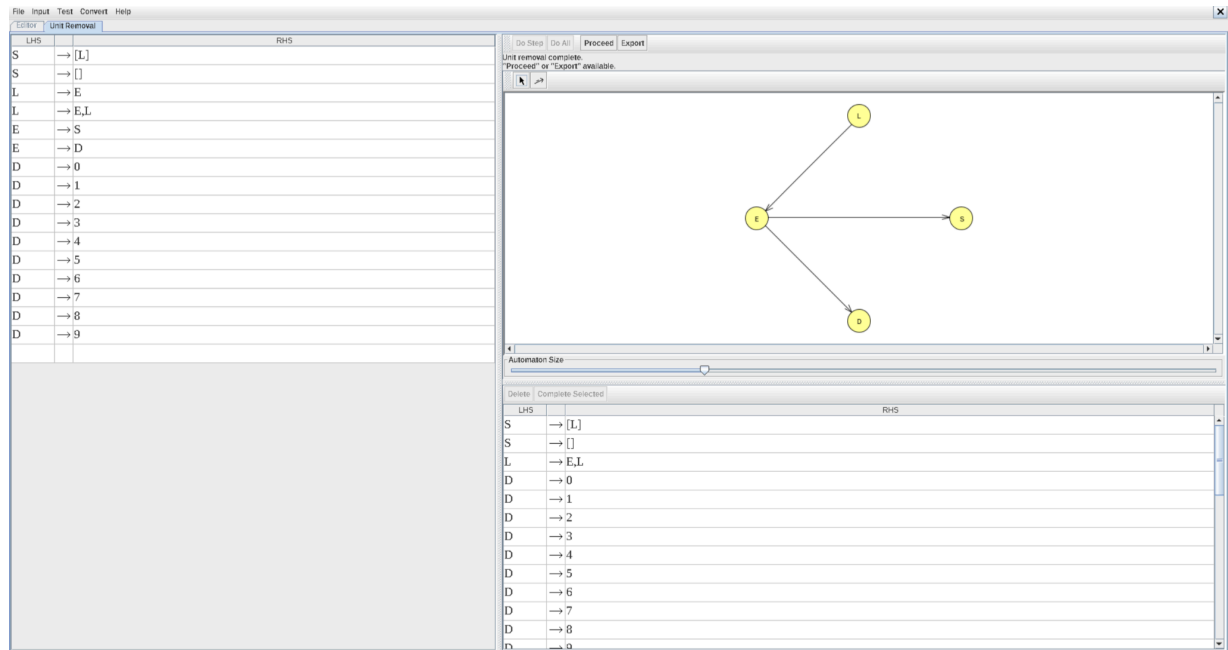
$$\text{Generadores } \{ S, L, E, D \}$$

Sin embargo D no es alcanzable,
por lo que se elimina.

$$S \rightarrow [L] \mid []$$

$$L \rightarrow E, L \mid [L] \mid [] \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$E \rightarrow [L] \mid [] \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$



File Input Test Convert Help		
Editor Unit Removal Useless Removal Chomsky Converter		
LHS		RHS
S	→	[L]
S	→	[]
E	→	5
E	→	4
E	→	7
E	→	6
E	→	1
L	→	8
E	→	0
L	→	9
E	→	3
E	→	2
L	→	4
L	→	5
L	→	6
L	→	7
E	→	9
L	→	0
E	→	8
L	→	1
L	→	2
L	→	3
L	→	E,L
L	→	[L]
E	→	[]
E	→	[L]
L	→	[]

File Input Test Convert Help

Editor

Brute Parser

Table Text Size

Start

Pause

Step

Noninverted Tree

Input []

String accepted! 2 nodes generated.

LHS		RHS
S	→	[L]
S	→	[]
L	→	E
L	→	E,L
E	→	S
E	→	D
D	→	0
D	→	1
D	→	2

S

L

I

I

S → [] is production 1

Derived [] from S. Derivations complete.

File Input Test Convert Help

Editor Brute Parser

Table Text Size

Start Pause Step Noninverted Tree

Input

String accepted! 6 nodes generated.

LHS		RHS
S	→	[L]
S	→	[]
L	→	E
L	→	E,L
E	→	S
E	→	D
D	→	0
D	→	1
D	→	2

Derived [] from S. Derivations complete.



Modificación

$$L = \{ a^n b^{2n} \mid n \text{ es impar} \}$$

$$L = \{a^n b^{2n} \mid n \text{ es impar}\}$$

$$S \rightarrow aAbb \mid abb$$

$$A \rightarrow aBb$$

$$B \rightarrow aAbb \mid abb$$

1º Eliminar prod vacías

$$A \text{ nulas} = \{ \}$$

No tengo prod. vacías

$$S \rightarrow aAbb \mid abb$$

$$A \rightarrow aBb$$

$$B \rightarrow aAbb \mid abb$$

2º Eliminar prod unitarios

$$\text{unitarios} \rightarrow \{ \}$$

No tengo prod. unitarios = igual

3º Eliminar prod / var inútiles

$$\text{Generadores} = \{S, A, B\}$$

No tengo prod / var inútiles =

$$S \rightarrow aAbb \mid abb$$

$$A \rightarrow aBb$$

$$B \rightarrow aAbb \mid abb$$

LHS		RHS
S	→	aAbb
S	→	abb
A	→	aBbb
B	→	aAbb
B	→	abb

Convert Selected
Do All
What's Left?
Export

Welcome to the Chomsky converter.
5 production(s) must be converted.

LHS		RHS
S	→	aAbb
S	→	abb
A	→	aBbb
B	→	aAbb
B	→	abb

Table Text Size

Start
Pause
Step
Noninverted Tree

Input: abb
String accepted! 2 nodes generated.

LHS		RHS
S	→	aAbb
S	→	abb
A	→	aBbb
B	→	aAbb
B	→	abb

```

graph TD
    S((S)) --- a((a))
    S --- b1((b))
    S --- b2((b))

```

Derived abb from S. Derivations complete.

Table Text Size

Start Pause Step Noninverted Tree

Input: aaabbbbbb
String accepted! 4 nodes generated.

LHS		RHS
S	→	aAbb
S	→	abb
A	→	aBbb
B	→	aAbb
B	→	abb

Derived abb from B. Derivations complete.

Table Text Size

Start Pause Step Noninverted Tree

Input: aaaaabbbbbbbbbb
String accepted! 6 nodes generated.

LHS		RHS
S	→	aAbb
S	→	abb
A	→	aBbb
B	→	aAbb
B	→	abb

Derived abb from B. Derivations complete.