Théorie des langages : THL

CM 6

Uli Fahrenberg

EPITA Rennes

S5 2021

Flex & Bison

Retour TP 1

`tp1_3.l`

Parsage

Programme du cours

1. Langages rationnels
2. Automates finis
3. Langages algébriques, grammaires hors-contexte, automates à pile
4. Parsage LL
5. TP 1 : `flex`

QCM 2

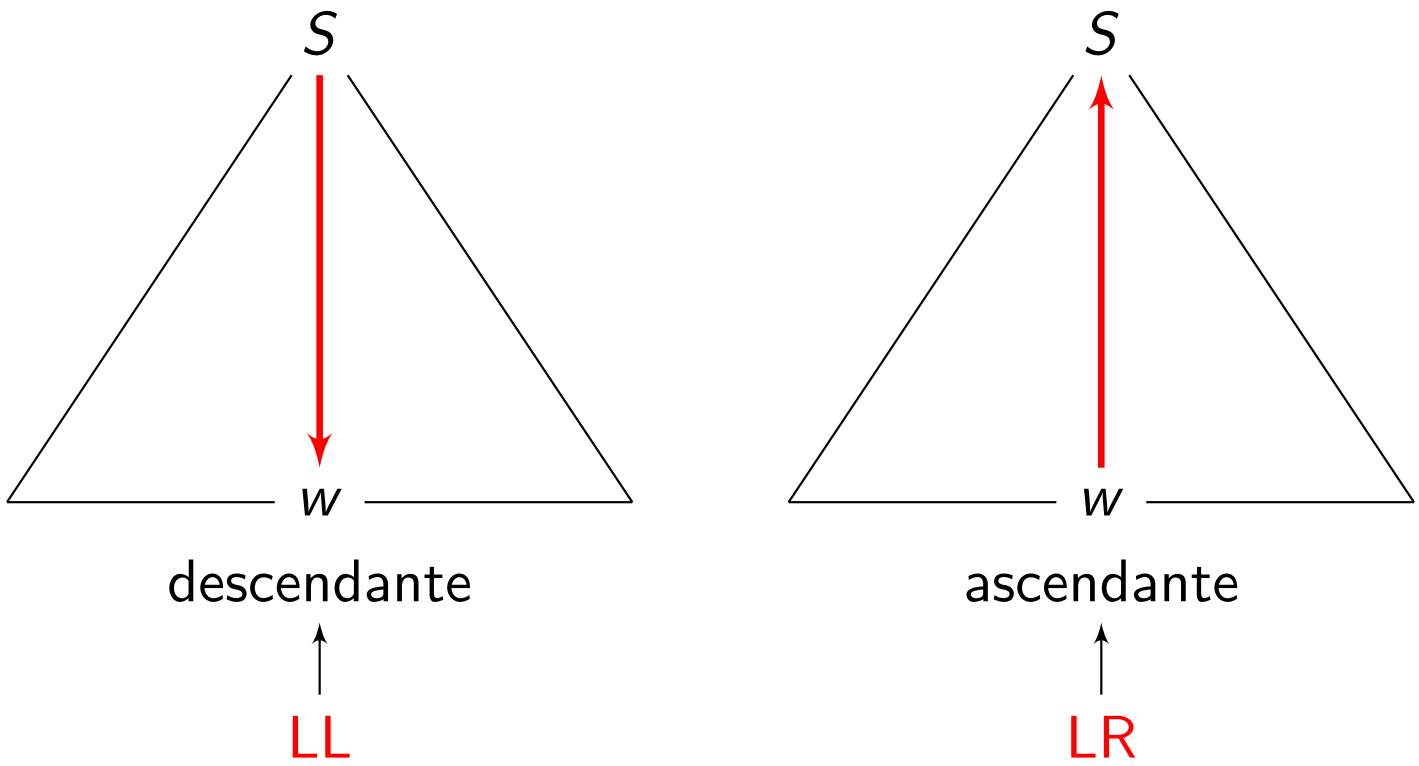
6. Parsage LR, partie 1
7. Parsage LR, partie 2
8. TP 2, 3 : `flex` & `bison`

Re : parsing LL(1)

0. entrée : une grammaire hors contexte $G = (N, \Sigma, P, S)$
   - si-dessous, $V = N \cup \Sigma$
   - éliminer récursion à gauche dans G ; factoriser G à gauche
1. calculer NULL
   - $\text{NULL} = \{A \in N \mid A \Rightarrow^* \varepsilon\}$
2. construire la table FIRST
   - $\text{FIRST}(A) = \{a \in \Sigma \mid \exists w \in V^* : A \Rightarrow^* aw\}$
3. construire la table FOLLOW
   - $\text{FOLLOW}(A) = \{a \in \Sigma \mid \exists B \in N, \alpha, \beta \in V^* : B \Rightarrow^* \alpha A a \beta\}$

4. construire la TABLE de parsage :
   1. pour chaque production $X \to w \quad (n)$:
      1. pour chaque $a \in \text{FIRST}(w)$: $\text{TABLE}(X, a)$ += $\{n\}$
      2. si $w \in \text{NULL}$ ou $w = \varepsilon$:
         – pour chaque $a \in \text{FOLLOW}(X)$: $\text{TABLE}(X, a)$ += $\{n\}$

Re : approches parsage

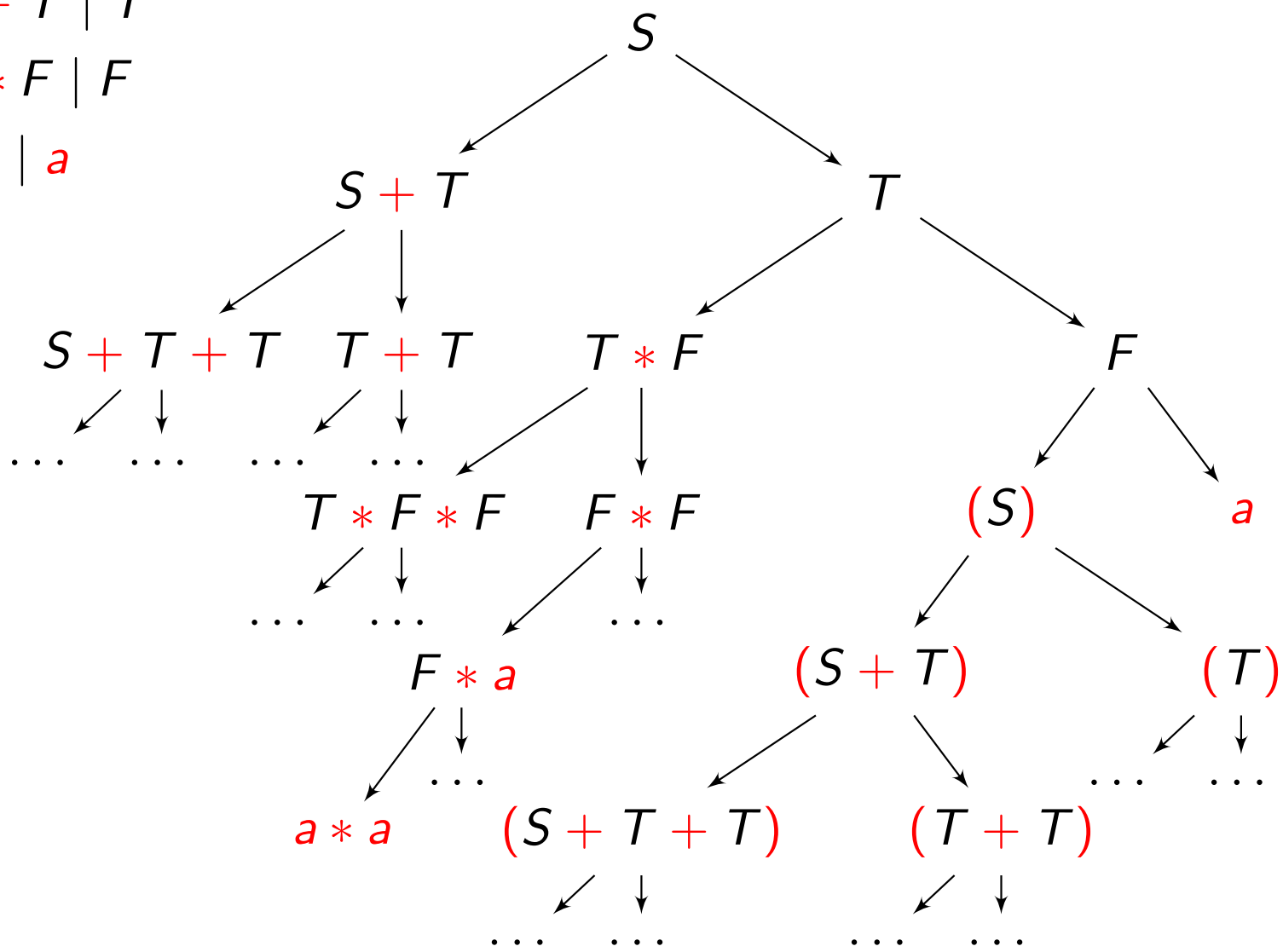


Plus précisement

$S \to S + T \mid T$

$T \to T * F \mid F$

$F \to (S) \mid a$

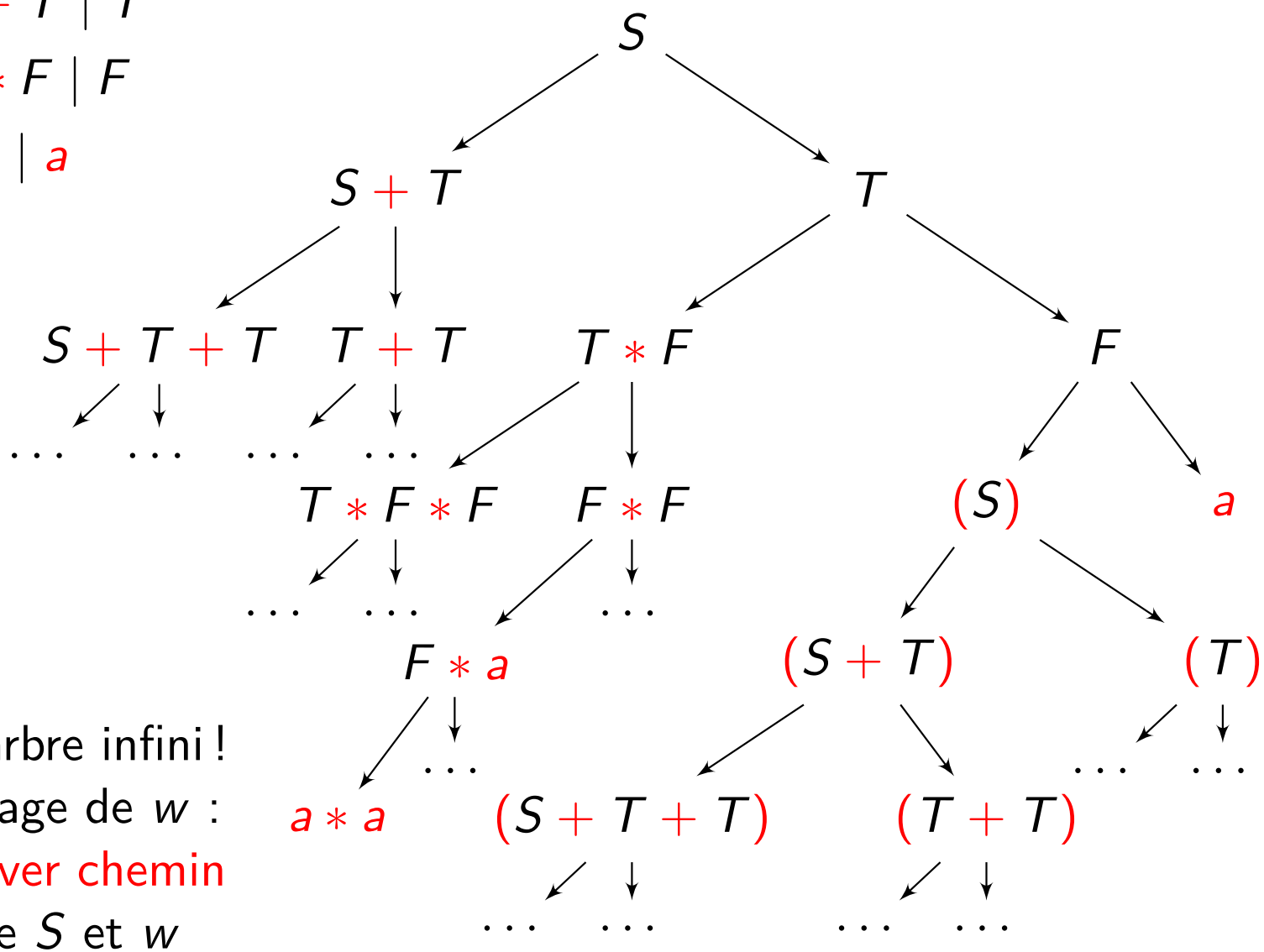


Plus précisement

$S \to S + T \mid T$

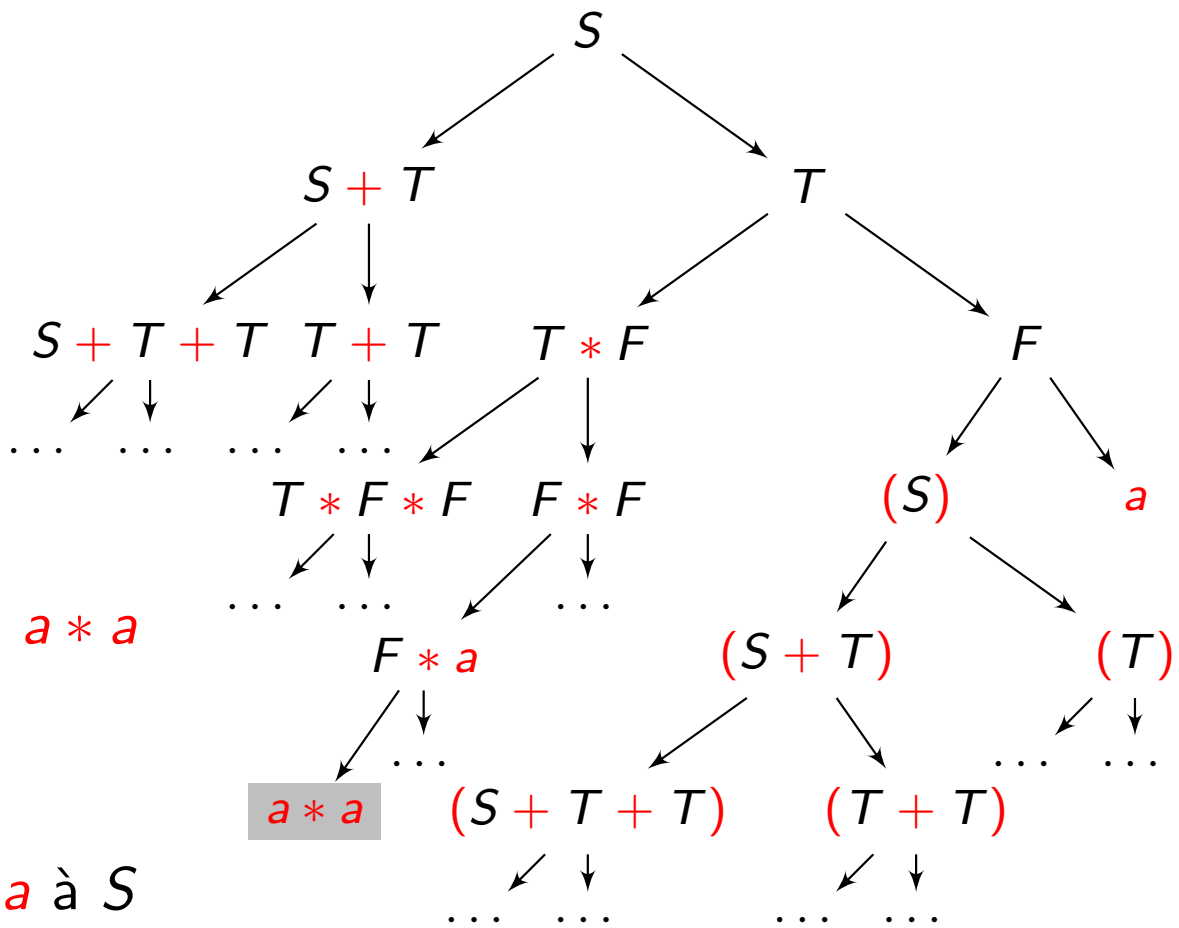
$T \to T * F \mid F$

$F \to (S) \mid a$



- un arbre infini !
- parsage de w : trouver chemin entre S et w

Approches parsage



Approche descendante :

- trouver chemin de S à $a * a$
- difficile ici !

Approche ascendante :

- trouver chemin de $a * a$ à S
- ici, facile !
  - l'inverse de l'arbre est (presque) déterministe
  - (pas toujours le cas . . .)

Parsage ascendant : the basics

function BULRP(α)
  if $\alpha = S$ **then**
    return True
  for $i \leftarrow 1$ **to** $|\alpha|$ **do**
    for $j \leftarrow i$ **to** $|\alpha|$ **do**
      for $A \in N$ **do**
        if $A \rightarrow \alpha_i \ldots \alpha_j$ **then**
          if BULRP($\alpha_1 \ldots \alpha_{i-1} A \alpha_{j+1} \ldots \alpha_n$) **then**
            return True
  return False

- très simple ¨‿
- plein de retours en arrière
- inutilisable ?

Parsage ascendant : the basics

function BULRP(α)
  if $\alpha = S$ **then**
    return True
  for $i \leftarrow 1$ **to** $|\alpha|$ **do**
    for $j \leftarrow i$ **to** $|\alpha|$ **do** ▷ décalage / SHIFT
      for $A \in N$ **do**
        if $A \rightarrow \alpha_i \ldots \alpha_j$ **then** ▷ réduction / REDUCE
          if BULRP($\alpha_1 \ldots \alpha_{i-1} A \alpha_{j+1} \ldots \alpha_n$) **then**
            return True
  return False

- très simple ¨‿
- plein de retours en arrière
- inutilisable ?

Exemple

$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$

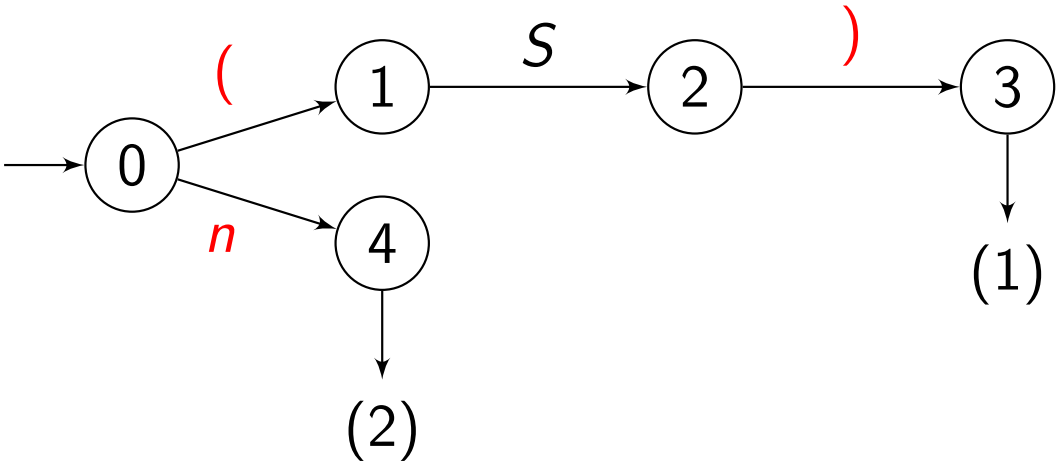
- utiliser une pile pour stocker les entrées décalés

entrée	pile	action
$((n))\$$	$\bot$	décaler
$(n))\$$	$\bot($	décaler
$n))\$$	$\bot(($	décaler
$))\$$	$\bot((n$	réduire 2
$))\$$	$\bot((S$	décaler
$)\$$	$\bot((S)$	réduire 1
$)\$$	$\bot(S$	décaler
$\$$	$\bot(S)$	réduire 1
$\$$	$\bot$	PROFIT

Automate de parsage

$$S \to (S) \qquad (1)$$
$$\mid n \qquad (2)$$

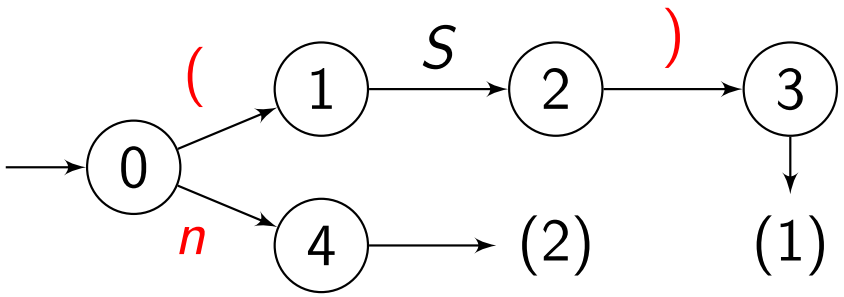
- reconnaissance de motifs par automate fini déterministe



- (en fait un transducteur fini)

Optimisations

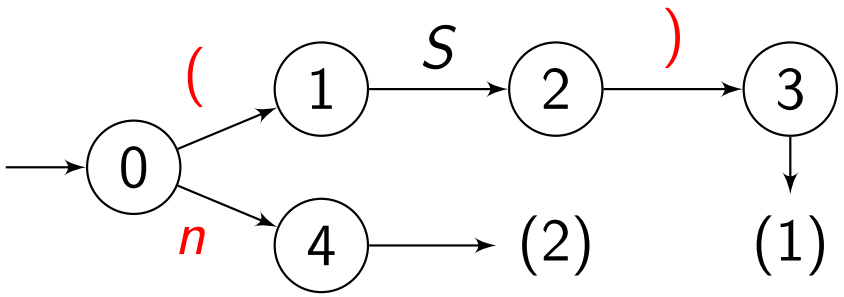
$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$



entrée	pile	action
$((n))\$$	$\bot$	décaler
$(n))\$$	$\bot($	décaler
$n))\$$	$\bot(($	décaler
$))\$$	$\bot((n$	réduire 2
$))\$$	$\bot((S$	décaler
$)\$$	$\bot((S)$	réduire 1
$)\$$	$\bot(S$	décaler
$\$$	$\bot(S)$	réduire 1
$\$$	$\bot$	PROFIT

Optimisations

$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$

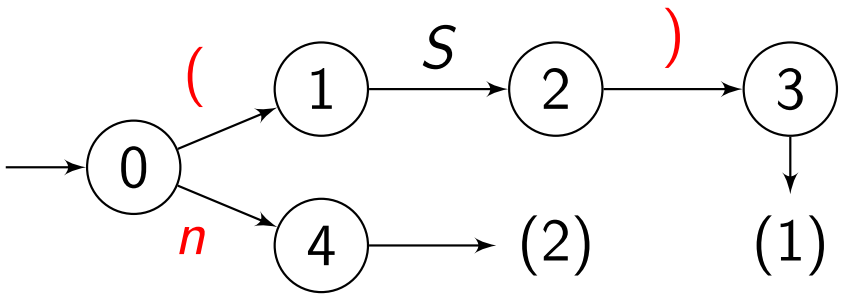


entrée	pile	action
$((n))\$$	$\bot$	décaler
$(n))\$$	$\bot(_1$	décaler
$n))\$$	$\bot(_1(_1$	décaler
$))\$$	$\bot(_1(_1 n_4$	réduire 2
$))\$$	$\bot(_1(_1 S_2$	décaler
$)\$$	$\bot(_1(_1 S_2)_3$	réduire 1
$)\$$	$\bot(_1 S_2$	décaler
$\$$	$\bot(_1 S_2)_3$	réduire 1
$\$$	$\bot$	PROFIT

- stocker l'état actuel dans la pile

Optimisations

$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$



entrée	pile	action
$((n))\$$	$\perp$	décaler
$(n))\$$	$\perp 1$	décaler
$n))\$$	$\perp 11$	décaler
$))\$$	$\perp 114$	réduire 2
$))\$$	$\perp 112$	décaler
$)\$$	$\perp 1123$	réduire 1
$)\$$	$\perp 12$	décaler
$\$$	$\perp 123$	réduire 1
$\$$	$\perp$	PROFIT

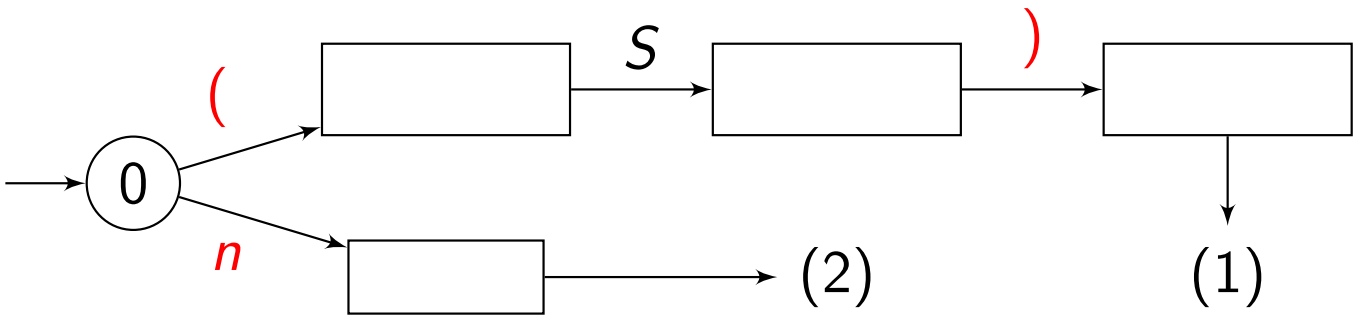
- stocker l'état actuel dans la pile
- en fait, plus besoin d'empiler des symboles !

Parsage LR

Construire l'automate de parsage

$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$

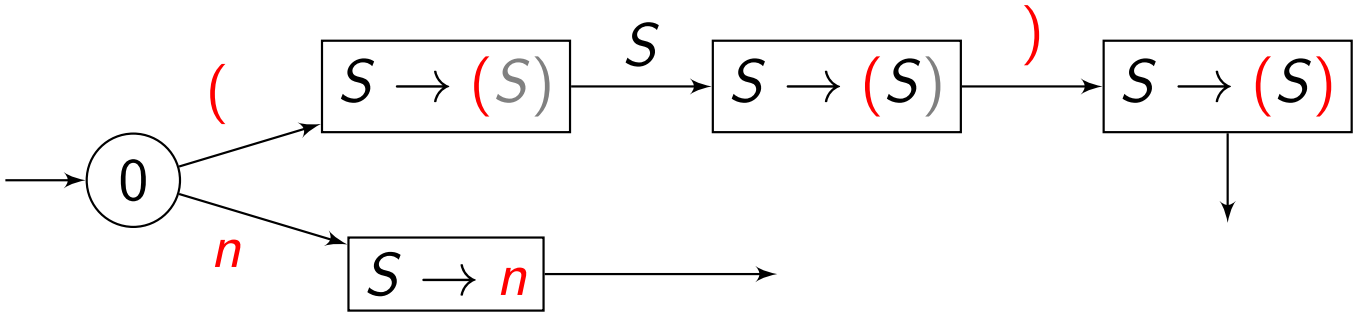
- états : productions partiellement achevées



Construire l'automate de parsage

$$S \to (S) \quad (1)$$
$$\mid n \quad (2)$$

- états : productions partiellement achevées



Définition (8.8)

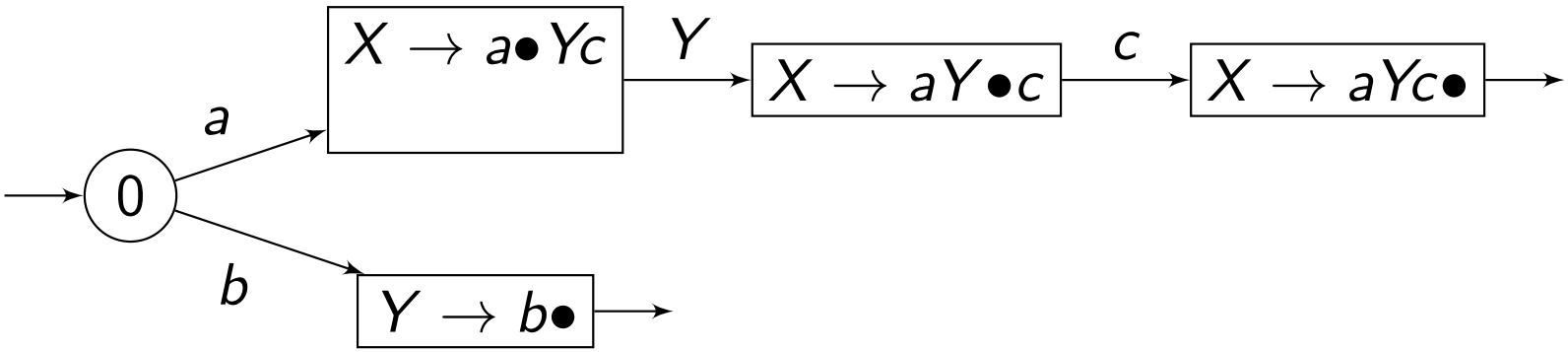
Soit G une grammaire hors-contexte. Une production pointée de G est une paire $(A, \alpha\bullet\beta)$ telle que $A \to \alpha\beta$ est une production de G.

- α : partie achevée, β : ce qui reste à trouver
- états : productions pointées
- transitions étiquetées dans V
- états finaux : productions de type $A \to w\bullet$

But Wait !

- plus compliqué que ça :

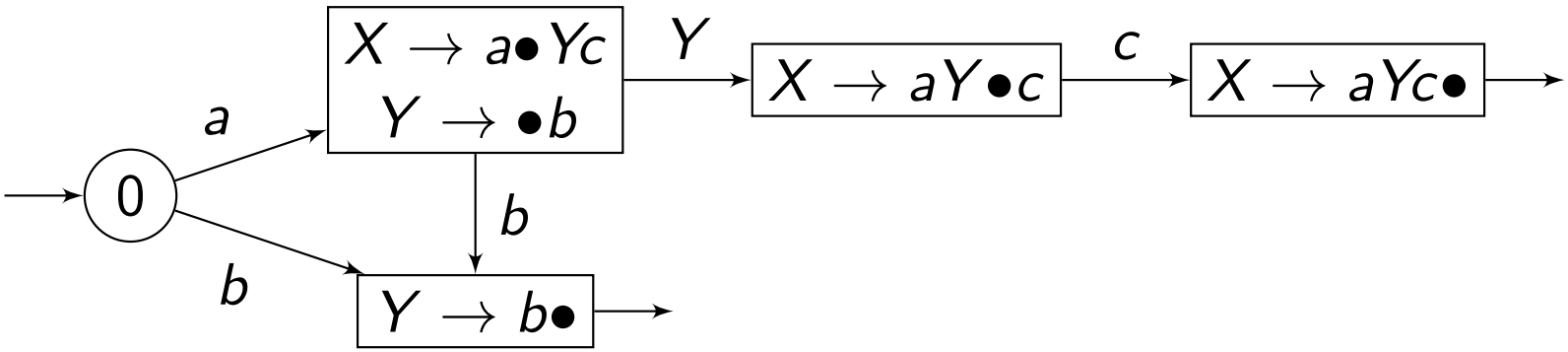
$$X \to aYc \qquad (1)$$
$$Y \to b \qquad (2)$$



But Wait !

- plus compliqué que ça :

$$X \to aYc \qquad (1)$$
$$Y \to b \qquad (2)$$



Clôture

Définition (8.10)

Soit G une grammaire hc et $\mathcal{I}$ un ensemble de productions pointées de G. La clôture de $\mathcal{I}$ est le plus petit ensemble $\mathrm{cl}(\mathcal{I})$ t.q. $\mathcal{I} \subseteq \mathrm{cl}(\mathcal{I})$ et

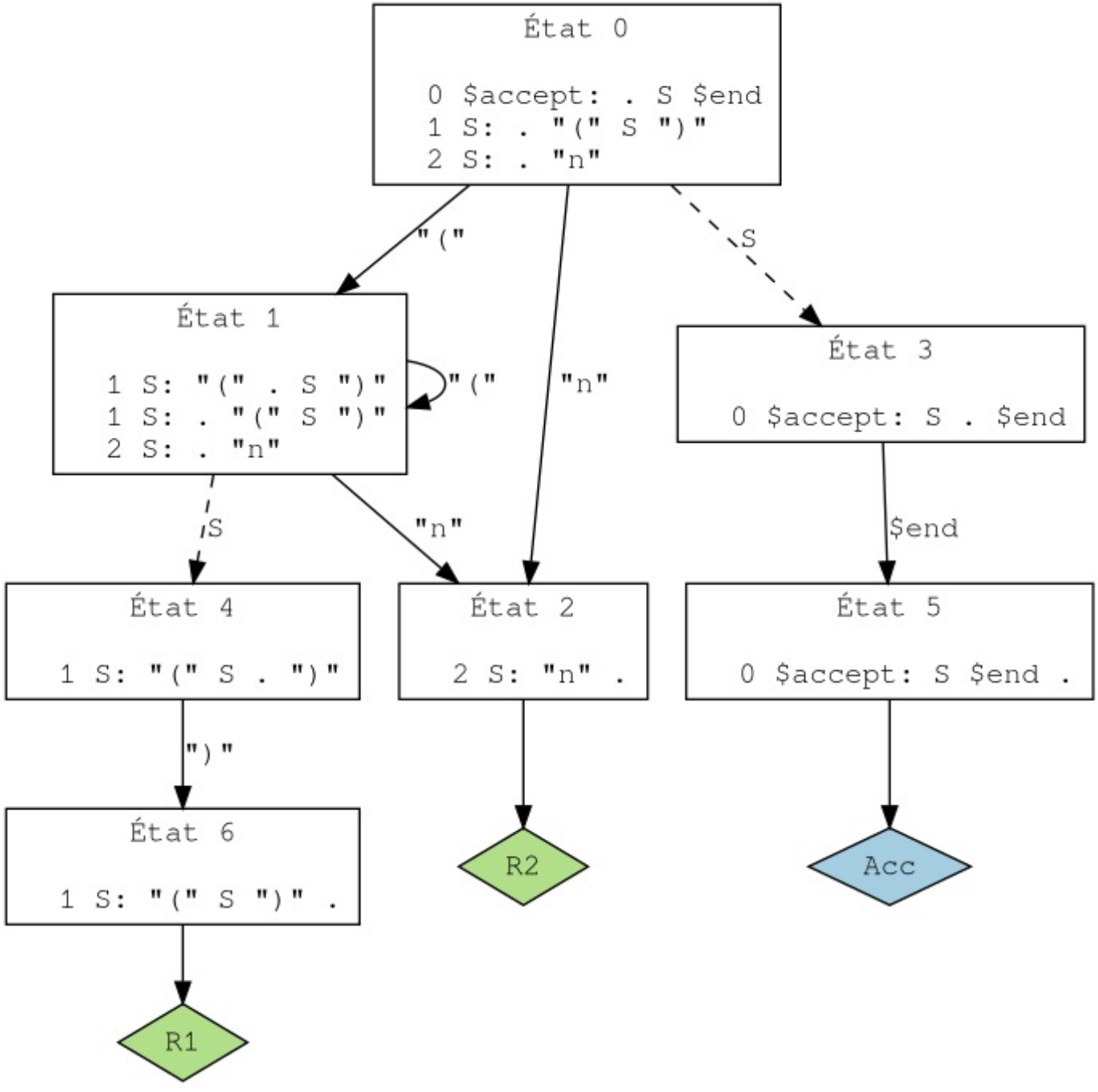
- si $(A, \alpha \bullet B \beta) \in \mathrm{cl}(\mathcal{I})$ et $B \to \gamma$ est une production de G, alors $(B, \bullet\gamma) \in \mathrm{cl}(\mathcal{I})$.

Définition

L'automate de parsage LR(0) d'une grammaire hors-contexte G est l'automate fini déterministe (Q, q_0, F, δ) avec

- $Q = \{\mathrm{cl}(\mathcal{I}) \mid \mathcal{I}$ ensemble de productions pointées de $G\}$;
- $q_0 = \mathrm{cl}(\{(Z, \bullet S\$)\})$;
- $F = \{q \in Q \mid \exists$ production $X \to w$ de G t.q. $(X, w\bullet) \in q\}$
- et $\delta : Q \times V \to Q$ donnée par

$$\delta(q, \beta) = \mathrm{cl}(\{(X, \alpha\beta \bullet \gamma) \mid (X, \alpha \bullet \beta\gamma) \in q\}).$$


État 0
0 $accept: . S $end
1 S: . "(" S ")"
2 S: . "n"
"("
S
État 1
1 S: "(" . S ")"
1 S: . "(" S ")"
2 S: . "n"
"("
"n"
État 3
0 $accept: S . $end
S
"n"
$end
État 4
1 S: "(" S . ")"
État 2
2 S: "n" .
État 5
0 $accept: S $end .
")"
État 6
1 S: "(" S ")" .
R2
Acc
R1


That's all Folks!