UNIVERSITÉ TOULOUSE III PAUL SABATIER

Rédigé avec LaTeX
Version du 1er octobre 2014

Traduction de langages

TL

M1 Informatique – Développement Logiciel
Semestre 7

Cours donné par Christine Maurel
Rédigé par Antoine de Roquemaurel

2014

Table des matières

1 Analyse Lexicale – Système d'équations de langages

1.1 Analyse lexicale

$$C = \{0, 1, \cdots, 8, 9\}$$

1.1.1

$$\begin{aligned} L &= O^*2\$(0+1)^+ + 0^*3\$(0+1+2) + \cdots + 0^*10\$(0+1+\cdots+9)^+ + (0+\cdots+9)^+ \quad (1.1)\\ &= C^+\$c^+ + c^+ = cc^*\$cc^* + cc^* \quad (1.2)\end{aligned}$$

R
— Solution (1.1) est à déterminiser
— Solution (1.2) il faudra ajouter des contraintes

1.1.2

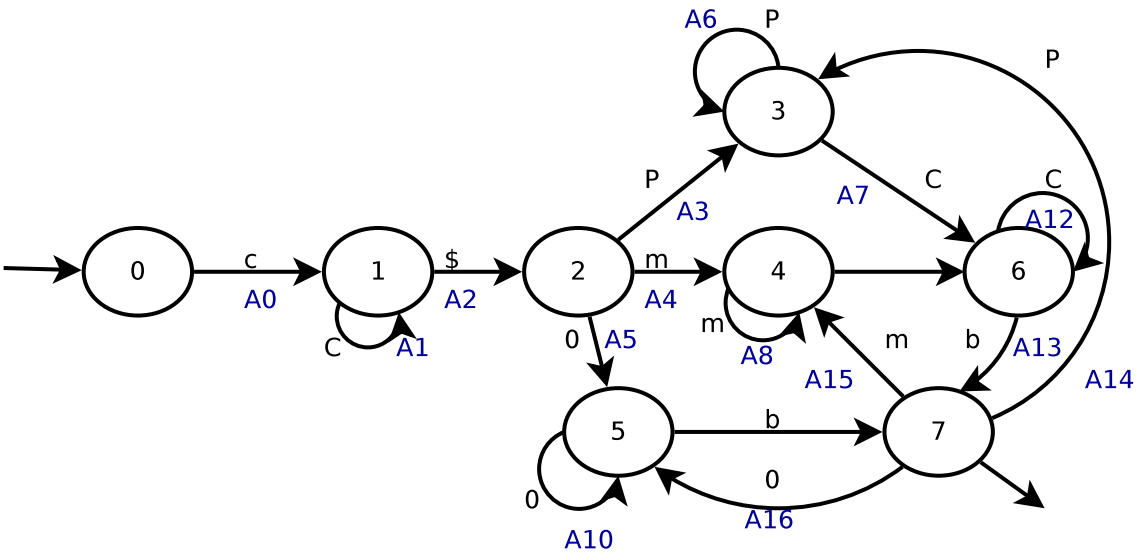

FIGURE 1.1 – Automate fini

```
  carrcour ;  val(carrcour)
variables:
  base; -- valeur décimale de la base avant '$'
  ecart; -- Valeur décimale d'un écart
  cpt; -- Compter les '+' ou '-' ou '0' successifs
  signe; -- Indique si on a des '+' ou '-' ou '0'
  --  +1 si on a '+'
  --  -1 si on a '-'
  --  0 si on a '0'

A1:  {base := base * 10 + val(valcarcour);}
A2:  {∅}
A3:  {cpt := 1; signe := 1;\}
A4:  {cpt := 1 ; signe := -1;\}
A5:  {cpt := 1; signe := 0; \}
A6:  {cpt := cpt + 1;\}
A7:  {ecart := val(carcours)\}
A8 = A10 = A6
A9 = A7
A12: \{ecart := ecart * 10 + val(carcours)}
A13: {
    tantque cpt != 0 faire
      printf(base + (signe * ecart));
      cpt := cpt - 1;
    fin tanque;
   }
 A11 = A13
 A14 = A3
 A15 = A4
 A16 = A5
```

1.1.3 Langage ALGOL60

R Une écriture en grammaire est également une écriture en langages, ainsi :

$$P = \begin{cases} A & \to & \alpha_1 \\ A & \to & \alpha_2 \\ & \cdots & \\ A & \to & \alpha_n \end{cases} \Leftrightarrow L(A) = \alpha_1 + \alpha_2 + \cdots + \alpha_n$$

1.1.3.1 Définition de la grammaire

$$\begin{aligned} X &= \{0, \cdots, 9, +, -, ., e\} \\ N &= \{I, J, K, L, M, N, O\} \\ S &= O \\ P &= I + \lambda = cP + \lambda \end{aligned}$$

1.1.3.2 Système d'équation de langage égaux à G

$$\begin{cases} I & = & c + cI \\ J & = & I + sI \\ K & = & .I \\ L & = & \underbrace{I + K + IK}_{Problèmederégularité?} \\ M & = & eJ \\ N & = & \text{Ł} + M + LM \\ O & = & N + sN \end{cases}$$

I Entier signé

c $c \in \{0 \cdots 9\}$

J Entier

s $s \in \{+, -\}$

K Partie fractionnaire

L Nombre décimal

M Partie exponentiel

N Nombre non signé

O Nombre

> **R** Un automate fini ne peut reconnaître que les langages réguliers, qui sont engendrés par des grammaires linéaires à droite.
> L'union de langages régulier engendre un langage régulier, de même le produit de deux langages réguliers donnent un langage régulier.

Dans notre cas, L ne pose donc aucun problème de régularité étant donné que I et K sont des langages réguliers ainsi, l'union et le produit de langages réguliers engendrant des langages réguliers, L sera régulier.

Le problème pourrait se poser pour L mais aussi pour N et M : la réponse étant la même.

O est régulier, on peut donc trouver un automate finis le reconnaissant.

$$
\begin{aligned}
I &= c(\lambda + I)cP \\
P &= I + \lambda = cP + \lambda \\
J &= I + sI = cP + sI \\
K &= .I \\
L &= I + K + IK = \underline{c}P + .I + \underline{c}PK = \underbrace{\underline{c}(P + PK)}_{Q} + .I \\
&= cQ + .I \\
Q &= P + PK = cP + \lambda + (cP + \lambda)K = cP + \lambda + cPK + K = c(\underbrace{\underline{P + PK}}_{Q}) + .I + \lambda \\
&= cQ + .I + \lambda \\
M &= eJ \\
N &= L + M + LM = cQ + .I + eJ + (cQ + .I)M = cQ + .I + eJ + cQM + .IM + .IM \\
&= c(\underbrace{\underline{Q + QM}}_{R}) + .(\underbrace{\underline{I + IM}}_{S}) + eJ \\
R &= Q + QM = cQ + .I + \lambda + cQM + .IM + eJM = c(Q + QM) + .(I + IM) + eJ + \lambda \\
S &= I + IM = cP + cPM = c(P + PM) \\
T &= P + PM = cP + \lambda + cPM + \overbrace{\underline{eJ}}^{M} = c(P + PM) = e + \lambda \\
O &= N + sN = cR + .S + eJJ + sN
\end{aligned}
$$

D'où le système d'équation de langage suivant :

$$
\begin{cases}
I &= cP \\
P &= cP + \lambda \\
J &= cP + sI \\
K &= .I \\
L &= cQ + .I \\
Q &= cQ + .I + \lambda \\
M &= eJ \\
N &= cR + .S + eJ \\
R &= cR + .s + eJ + \lambda \\
S &= cT \\
T &= cT + eJ + \lambda \\
O &= cR + .s + eJ + sN
\end{cases}
$$

L'axiome O est un état initial et {T,R,Q,P} sont des états finaux.

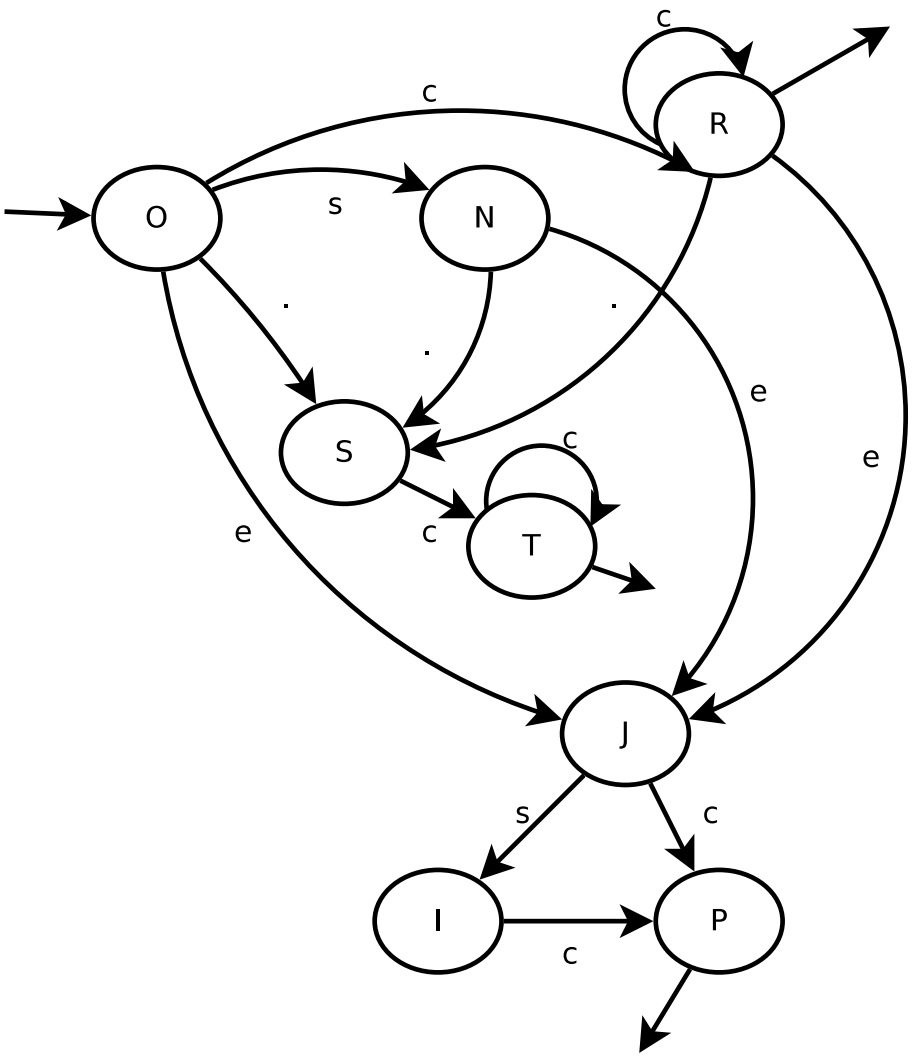

FIGURE 1.2 – Automate fini

1.2 Systèmes d'équations algébriques de langages

R 3 théorèmes :
Arden $X = r_1X + r_2 \Rightarrow X = r_1^*r_2$
Arden Bis $X = Xr_1 + r_2 \Rightarrow X = r_2r_1^*$
AnBn $X = Yr_1 + r_{\Rightarrow}X = r_2r_1^*$

1.2.1 G1

$$S = S\underbrace{a}_{r_1} + \underbrace{b}_{r_2} \Rightarrow S$$
$$\overset{Ardenbis}{=} ba* = ba^n n \geq 0$$

1.2.2 G2

$$\begin{cases} S &= aSb + T \overset{AnBn}{\Longrightarrow} S = a^n T b^n, n \geq 0 = a^n b^+ b^n n \geq 0 \\ T &= Tb + b \overset{Ardenbis}{\Longrightarrow} \end{cases}$$

2 Gramaires LL(1) – Descente récursive

2.1

2.1.1 Firsts

$$
\begin{aligned}
S' &\to S\$ && (2.1)\\
S &\to bRS && (2.2)\\
S &\to RcSa && (2.3)\\
S &\to \lambda && (2.4)\\
R &\to acR && (2.5)\\
R &\to b && (2.6)\\
& && (2.7)
\end{aligned}
$$

$$
\begin{aligned}
first_1(R) &= \{a, b\}\\
first_1(S &= \{a, b, \lambda\}\\
first_2(R) &= \{ac, b\}\\
first_2(S) &= \{\underbrace{\lambda}_{2.3}, \underbrace{bb}_{2.2+R}\}
\end{aligned}
$$

R Pour le $first_2(R)$, ac est le préfixe de longueur 2 des mots prouits par R, et b est le mot de longueur inférieur ou égale à 2 produit par R.

Nous avons chercher les $first$ intuitivement, cependant nous nous sommes servis de la formule suivante : $first_k(\alpha_1\alpha_2\cdots\alpha_n) = first_k(first_k(\alpha_a)first(\alpha_2)\cdots first_k(\alpha_n)$