

# Langages rationnels

## Table des matières

<b>1 Généralités sur les langages</b>	<b>2</b>
1.1 Un peu de vocabulaire . . . . .	2
1.2 Exemples de langages . . . . .	2
1.3 Opérations sur les langages . . . . .	2
<b>2 Expressions rationnelles</b>	<b>3</b>
2.1 Définition des expressions rationnelles . . . . .	3
2.2 Un peu de Caml . . . . .	3
<b>3 Langages rationnels</b>	<b>4</b>
3.1 Langage associé à une expression rationnelle . . . . .	4
3.2 Un exercice Caml . . . . .	4
3.3 Retour sur les exemples . . . . .	5
3.4 Une caractérisation de la classe des langages rationnels . . . . .	6
<b>4 Questions pour la suite</b>	<b>6</b>



Staring : toto

# 1 Généralités sur les langages

## 1.1 Un peu de vocabulaire

### DEFINITION 1

- Un *alphabet* est un ensemble fini ; ces éléments sont des *lettres*.
- Un *mot* sur l'alphabet  $A$  est une suite **finie** de lettres de  $A$  :  $w = a_1 \cdots a_n$ .
- La *longueur* du mot précédent, notée  $|w|$  est  $n$  (nombre de lettres).
- On définit un *mot vide* noté généralement  $\varepsilon$ , constitué d'aucune lettre. Sa longueur est nulle.
- L'ensemble des mots de longueur  $n$  est noté  $A^n$  ; l'ensemble des mots est noté  $A^*$ . L'ensemble des mots non vides est noté  $A^+$ .
- On définit la *concaténation* de deux mots  $w_1 = a_1 \cdots a_n$  et  $w_2 = b_1 \cdots b_m$  par  $w_1.w_2 = a_1 \cdots a_n b_1 \cdots b_m$ , de sorte que  $|w_1.w_2| = |w_1| + |w_2|$ .
- $a^n$  désigne la concaténation de  $n$  lettres  $a$  (si  $n = 0$ , alors  $a^n = \varepsilon$ ).
- Par convention,  $w.\varepsilon = \varepsilon.w = w$ , de sorte que  $\cdot$  définit sur  $A^*$  une loi de composition interne associative, non commutative (sauf si  $A$  est un singleton), de neutre  $\varepsilon$ . Ainsi,  $(A^*, \cdot)$  constitue un *monoïde*. Une autre convention consiste à confondre une lettre et le mot constitué uniquement de cette lettre.
- Un *langage* sur  $A$  est une partie de  $A^*$ .
- Si  $w = a_1 \cdots a_n$ , les *préfixes* (resp. *suffixes*) de  $w$  sont les mots de la forme  $a_1 \cdots a_k$  (resp.  $a_k \cdots a_n$ ), avec  $1 \leq k \leq n$ . De plus, par convention,  $\varepsilon$  est préfixe et suffixe de tout mot. Un préfixe (resp. suffixe) *propre* de  $w$  est un préfixe (resp. suffixe) distinct de  $w$ . De plus, on convient parfois que le mot vide est préfixe et suffixe de tout mot.
- Un *facteur* de  $w = a_1 \cdots a_n$  est un mot de la forme  $a_i \cdots a_j$ , avec  $1 \leq i \leq j \leq n$ , alors qu'un *sous-mot* est un mot de la forme  $a_{i_1} \cdots a_{i_k}$ , avec  $1 \leq i_1 < \cdots < i_k \leq n$ . Ici encore,  $\varepsilon$  est un facteur et un sous-mot de tout mot.
- Le *miroir* de  $w = a_1 \cdots a_n$  est  $\tilde{w} = a_n \cdots a_1$ . Si  $w = \tilde{w}$ , on dit que  $w$  est un *palindrome*.

## 1.2 Exemples de langages

On peut décrire des langages de bien des façons différentes. Parmi les langages suivants, sauriez-vous écrire un programme (caml, pascal, maple, assembleur, logo. . .) déterminant si un mot donné est élément du langage?<sup>1</sup>

1.  $\{w \in A^*; 3 \mid |w|\}$ .
2. Ensemble des mots sur  $\{a, b\}$  contenant  $aabb$  comme sous-mot (resp. facteur, préfixe, suffixe. . .).
3. Ensemble des mots sur  $\{a', \dots, z'\}$  qui n'admettent pas  $agaga$  comme préfixe.
4. Ensemble des palindromes sur un alphabet donné.
5.  $\{a^n b^n \mid n \in \mathbb{N}\}$  ;  $\{a^n b^m \mid n, m \in \mathbb{N}\}$  ;  $\{a^p b^q c^{p+q} \mid p, q \in \mathbb{N}\}$ .
6. Ensemble des énoncés faux du cours d'info de cette année (resp. l'année dernière).
7. Ensemble des mots (sur un alphabet à préciser. . .) décrivant une fonction caml syntaxiquement correcte.
8. Ensemble des mots décrivant une fonction caml de type `int -> int` calculant  $n!$  pour toute entrée  $n$ .
9. Ensemble des mots décrivant une fonction caml dont le calcul termine en moins de  $10^5$  années sur une machine donnée, sur toute entrée correspondant à son typage.
10. Ensemble des mots décrivant une fonction caml dont le calcul termine sur toute entrée correspondant à son typage.

## 1.3 Opérations sur les langages

Les langages sur  $A$  étant des parties de  $A^*$ , on dispose naturellement des notions de réunion, intersection, et complémentaire de langages. On peut de plus définir la concaténation de deux langages  $L_1$  et

<sup>1</sup>on dit que le programme *reconnait* le langage, qui est alors déclaré *calculable*

$L_2$  par :

$$L_1.L_2 = \{w_1w_2 \mid w_1 \in L_1, w_2 \in L_2\}.$$

On définit alors de façon naturelle  $L^n$  comme l'ensemble des mots constitués de  $n$  mots de  $L$  concaténés. Par convention,  $L^0 = \{\varepsilon\}$ . Notons que cet ensemble n'est pas vide : il est constitué d'un mot (qui est le mot vide, certes...).

La *fermeture de Kleene* d'un langage  $L$  est la réunion des  $L^n$ , pour  $n \in \mathbb{N}$ . Elle est notée  $L^*$ , et contient toujours le mot vide.  $L^+$  désigne la réunion des  $L^n$  pour  $n \geq 1$  : ce langage contient le mot vide si et seulement si  $\varepsilon \in L$ .

## 2 Expressions rationnelles

### 2.1 Définition des expressions rationnelles

On définit les *expressions rationnelles* de façon inductive :

#### DEFINITION 2

L'ensemble  $\mathcal{E}$  des expressions rationnelles sur un alphabet  $A$  est le plus petit ensemble tel que :

- $\mathcal{E}$  contient les éléments de  $A$ , ainsi que deux éléments distincts des précédents  $\varepsilon$  et  $\emptyset$ .
- Si  $e_1$  et  $e_2$  sont deux éléments de  $\mathcal{E}$ , alors  $(e_1) + (e_2)$  et  $(e_1).(e_2)$  également.
- Si  $e \in \mathcal{E}$ , alors  $(e)^* \in \mathcal{E}$ .

#### REMARQUES 1

1. On note parfois  $(e_1)|(e_2)$  à la place de  $(e_1) + (e_2)$ .
2. On a donné une présentation "à l'informaticienne", un peu comme on "définit"  $\mathbb{C}$  en terminale : "un ensemble contenant  $\mathbb{R}$ , un certain élément  $i$ , avec les règles de calcul...". Pour une définition plus Bourbakeuse, on peut voir les expressions rationnelles comme des mots d'un certain type sur l'alphabet  $A \cup \{., (, ), *, +\}$ , ou encore comme des arbres.
3. Cette "définition" décrit bien un ensemble de façon univoque : il s'agit de la réunion des  $E_n$ , où  $E_0 = \{\emptyset, \varepsilon\} \cup A$ , et  $E_{n+1}$  est constitué des éléments de  $E_n$ , ainsi que les  $(e_1)^*$ ,  $(e_1) + (e_2)$  et  $(e_1).(e_2)$ , pour  $e_1, e_2 \in E_n$ .
4. Cette définition permet les preuves par induction : pour montrer une propriété  $\mathcal{P}(e)$  pour toute expression  $e$ , il suffit d'établir  $\mathcal{P}(\emptyset)$ ,  $\mathcal{P}(\varepsilon)$ ,  $\mathcal{P}(\alpha)$  pour tout  $\alpha \in A$ , puis de montrer que si  $\mathcal{P}(e_1)$  et  $\mathcal{P}(e_2)$  sont vérifiées, alors  $\mathcal{P}((e_1) + (e_2))$ ,  $\mathcal{P}((e_1).(e_2))$  et  $\mathcal{P}((e_1)^*)$  également.
5. Pour éviter toute confusion entre les expressions rationnelles et les mots/langages, il faudrait distinguer l'expression rationnelle  $\varepsilon$  du mot vide, ainsi que l'expression rationnelle  $\emptyset$  du langage vide...
6. Grâce au parenthésage, il y a unicité de la structure d'une expression rationnelle, au sens où deux expressions rationnelles sont égales si et seulement si ce sont deux "éléments atomiques" ( $\varepsilon, \emptyset, \alpha$ ) égaux, ou alors sont obtenus par deux constructions de même nature, à partir d'expressions égales. Sans le parenthésage, l'expression  $a.b + c$  pourrait être obtenue comme  $e_1 + e_2$  avec  $e_1 = a.b$  et  $e_2 = c$ , ou bien  $e_1.e_2$  avec  $e_1 = a$  et  $e_2 = b + c$ .

### 2.2 Un peu de Caml

Une définition "à la caml" des expressions rationnelles serait :

```
#type 'a exp_rat= vide
| epsilon
| Lettre of 'a
| Etoile of 'a exp_rat
| Plus of ('a exp_rat)*('a exp_rat)
| Concat of ('a exp_rat)*('a exp_rat);;
```

Remarquons que cette définition autorise la création d'expression auto-référente, du type :

```
let rec er=Plus(Lettre('a'),er);;
```

De telles déclarations ne correspondent pas à la définition des expressions rationnelles (en structure interne, on a des objets qui pointent sur d'autres, et on a créé ici un cycle...).

## 3 Langages rationnels

### 3.1 Langage associé à une expression rationnelle

#### DEFINITION 3

On définit de façon inductive le langage associé à une expression rationnelle  $e$ , noté  $\mathcal{L}(e)$  :

1.  $\mathcal{L}(\varepsilon) = \{\varepsilon\}$ ;  $\mathcal{L}(\emptyset) = \emptyset$ ;  $\mathcal{L}(a) = \{a\}$  pour tout  $a \in A$ ;
2.  $\mathcal{L}((e_1).(e_2)) = \mathcal{L}(e_1).\mathcal{L}(e_2)$ ;
3.  $\mathcal{L}((e_1) + (e_2)) = \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$ ;
4.  $\mathcal{L}(e^*) = (\mathcal{L}(e))^*$ .

#### REMARQUES 2

1. Grâce à la remarque 6 de 2.1, il n'y a pas d'ambiguïté dans la définition de  $\mathcal{L}(e)$ .
2. On prendra garde à bien distinguer le langage vide  $\mathcal{L}(\emptyset) = \emptyset$  du langage constitué de l'unique mot vide  $\mathcal{L}(\varepsilon) = \{\varepsilon\}$ .

#### DEFINITION 4

Un langage est dit *rationnel* si c'est le langage associé à une expression rationnelle. On note  $\text{Rat}(A)$  l'ensemble des langages rationnels :  $\text{Rat}(A) \subset \mathcal{P}(A^*)$ .

Dans ce chapitre et (surtout) le suivant, on cherchera à caractériser les langages rationnels parmi tous les langages. Notons déjà que l'on a le :

**FAIT 1** *Il existe une infinité dénombrable de langages rationnels sur tout alphabet  $A$ , alors qu'il existe une infinité non dénombrable de langages sur  $A$ .*

**PREUVE** : D'après leur définition (voir aussi la remarque 3 de 2.1), il existe une infinité dénombrable d'expressions rationnelles, puis **au plus** une infinité dénombrable de langages rationnelles. Mais comme les langages associés aux expressions  $a$ ,  $aa$ ,  $aaa$ , etc. . . sont distinctes, l'ensemble des langages rationnels est donc bien dénombrable (ce qui, *bien entendu* ne signifie pas que ces langages sont eux même infinis. . .).

Maintenant  $A^*$  est dénombrable (réunion dénombrable d'ensembles finis non vides disjoints), et on sait qu'il n'existe jamais de surjections<sup>2</sup> d'un ensemble sur l'ensemble de ses parties<sup>3</sup>. C'est gagné. ■

**COROLLAIRE 1** *Il existe des langages non rationnels.*

#### REMARQUES 3

1. Le résultat précédent (et son corollaire) s'appliquent à tous les modèles usuels de calculs, basés sur des constructions dénombrables à partir d'objets finis.
2. Les expressions  $((a + b) + (c)) + (d)$  et  $(a) + (((b) + (c)) + (d))$  sont associées au même langage. On préférera donc la notation plus légère  $a + b + c + d$ .
3. Même remarque pour la concaténation. On s'autorisera donc des expressions de la forme  $aba + bab$ , avec pour convention syntaxique la priorité de la concaténation par rapport à la somme. Le statut de l'étoile est plus problématique, et on paranthésiera donc systématiquement.
4. On s'autorise de même l'expression rationnelle  $A$ , qui signifie  $a + b + \dots + z$  si  $A = \{a, b, \dots, z\}$ .

### 3.2 Un exercice Caml

**EXERCICE 1** *Avec les types définis précédemment, écrire une fonction Caml de signature*

```
string -> char expr_rat -> bool
```

*déterminant si un mot est dans le langage associé à une expression rationnelle.*

<sup>2</sup>donc a fortiori de bijections

<sup>3</sup>on suppose l'existence d'une telle surjection  $s$ , et on considère l'ensemble  $X$  des  $x \in E$  tels que  $x \notin s(x)$ , etc. . .

SOLUTION : Le test utilisera bien entendu fortement la définition inductive des expressions rationnelles. Si l'expression est de la forme  $\varepsilon$ ,  $\emptyset$  ou  $(e_1) + (e_2)$ , cela ne pose pas de problème. Par contre, si  $e = (e_1)^*$  ou  $(e_1).(e_2)$ , on est obligé de tester les différents découpages possibles du mot :

```
#let rec test w e=let n=string_length w in match e with
| vide -> false
| epsilon -> n=0
| Lettre(l)->n=1 && w.[0]=l
| Plus(e1,e2)->(test w e1)|| (test w e2)
| Concat(e1,e2)->(let r=ref false and i=ref 0 in
  begin while (!i<=n)&& (not !r) do
    r:=(test (sub_string w 0 !i) e1)
      &&(test (sub_string w !i (n- !i)) e2);
    i:= !i+1 done; !r end)
| Etoile(e1)->(let r=ref (n=0) and i=ref 1 in
  begin while (!i<=n)&& (not !r) do
    r:=(test (sub_string w 0 !i) e1)
      &&(test (sub_string w !i (n- !i)) e);
    i:= !i+1 done; !r end);;
test : string -> char exp_rat -> bool = <fun>
```

Quelques essais (pertinents) :

```
let w1="" and w2="toto" and w3="totot"
and e1=Etoile(Plus(Lettre 't',Lettre 'o'))
and e2=Etoile(Concat(Lettre 't',Lettre 'o'));;
let e3=Concat(e2,Lettre 't');;

test w1 e1,test w1 e2,test w1 e3;;
- : bool * bool * bool = true, true, false
test w3 e1,test w2 e2,test w2 e3;;
- : bool * bool * bool = true, true, false
test w3 e1,test w3 e2,test w3 e3;;
- : bool * bool * bool = true, false, true
```

La preuve de la terminaison et de la correction de cette fonction se fait par induction : on utilise pour cela l'ordre lexicographique sur le couple d'entrée  $(|e|, |w|)$ , où  $|e|$  est la longueur de  $e$ , vue comme un mot. En effet,  $|e|$  diminue strictement sauf dans un cas (étoile), mais alors c'est  $|w|$  qui diminue strictement.

### 3.3 Retour sur les exemples

Une partie des arguments suivants seront justifiés dans le prochain chapitre sur les automates finis.

**EXERCICE 2** Reprendre les différents exemples de langages de la première partie, et déterminer lesquels sont rationnels.

SOLUTION : On sait déjà que tout langage rationnel peut être reconnu par un programme Caml (utiliser l'exercice 1). Pour ce qui est du choix du langage, il est sans importance, puisque tous les langages usuels de programmation reconnaissent les mêmes langages (on peut simuler les uns par les autres).

1. Il s'agit du langage associé à  $(A^3)^*$ .
2. Ensemble des mots admettant  $aabb$  comme préfixe :  $aabbA^*$  ; suffixe :  $A^*aabb$  ; facteur :  $A^*aabbA^*$  ; sous-mot :  $A^*aA^*aA^*bA^*bA^*$ .
3. Il s'agit du complémentaire d'un langage rationnel (voir la question précédente) : on sait alors qu'un tel langage est rationnel (mais ce n'est pas évident avec ce que vous savez, petits 3/2...). On peut donner une expression rationnelle :

$$\varepsilon + A + A^2 + A^3 + A^4 + \sum_{w \in E} wA^*,$$

où  $E$  est l'ensemble (fini) des mots de longueur 5 autres que *agaga*.

4. C'est un grand classique : l'ensemble des palindromes sur un alphabet non trivial est irrationnel (attendre quelques semaines). Il n'est pas difficile en revanche d'écrire un programme caml reconnaissant ce langage.
5. Ces langages sont facilement reconnaissables par un programme caml. Le premier et le troisième sont *notoirement* non rationnels (on le verra quelques fois cette année...), alors que le second est associé à l'expression  $a^*b^*$ .
6. Dans les deux cas, il s'agit d'ensembles finis (probablement non vides), donc rationnels. Pour ce qui est de les énumérer, c'est plus délicat...
7. Le compilateur caml (écrit en caml!) reconnaît les fonctions syntaxiquement correctes. Par contre, ce langage est trop compliqué pour être rationnel : y réfléchir, ou (pour les  $3/2$ ...) demander aux  $5/2$ .
8. Ce langage n'est pas caml-calculable, donc a fortiori il est irrationnel. Supposons en effet qu'une telle fonction `test` existe. Elle est de type `int -> int -> bool`. Si on considère la fonction :

```
let rec f n=if test f then -1 else facto n;;
```

avec `facto` une fonction calculant effectivement la fonction factorielle. Que `test f` rende vrai ou faux, il se trompe...

9. Il s'agit d'un langage fini (pourquoi, au fait?) donc rationnel.
10. Cette fonction n'est pas caml-calculable, donc est irrationnelle. Dans le même esprit que quelques lignes plus haut, on pourra considérer la fonction

```
let rec zblog ()=if arret zblorg then zblorg () else 1515;;
```

qui va poser quelques problèmes.

### 3.4 Une caractérisation de la classe des langages rationnels

D'après la définition des expressions et langages rationnels, il n'est pas difficile de montrer le :

**FAIT 2** *Rat(A) est la plus petite partie de  $\mathcal{P}(A^*)$  contenant  $\emptyset$ , les singletons, et stable par réunion, concaténation, et fermeture de Kleene.*

On pourrait d'ailleurs définir les langages rationnels de cette façon ; on montrerait ensuite que de tels langages correspondent aux langages associés aux expressions rationnelles...

## 4 Questions pour la suite

- Comment déterminer si un langage donné est rationnel ou non ?
- L'intersection de deux langages rationnels est-elle rationnelle ?
- Le complémentaire d'un langage rationnel est-il rationnel ? (relier cette question à la précédente...)
- Comment tester l'égalité des langages définis par deux expressions rationnelles ?
- Même chose pour l'inclusion ?

On termine avec un exercice :

**EXERCICE 3** *L'ensemble des expressions rationnelles sur un alphabet A donné est-il rationnel sur  $A \cup \{(), \cdot, +, *\}$  ? (heu, ça a un sens ça ? ben oui...)*