



## INF 302 : LANGAGES & AUTOMATES

### Chapitre 9 : Expressions régulières

Yliès Falcone

[ylies.falcone@univ-grenoble-alpes.fr](mailto:ylies.falcone@univ-grenoble-alpes.fr) — [www.ylies.fr](http://www.ylies.fr)

Univ. Grenoble-Alpes, Inria

Laboratoire d'Informatique de Grenoble - [www.liglab.fr](http://www.liglab.fr)  
Équipe de recherche LIG-Inria, CORSE - [team.inria.fr/corse/](http://team.inria.fr/corse/)

## Plan Chap. 9 - Expressions Régulières

- 1 Motivations
- 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- 3 Applications en informatique : commandes UNIX
- 4 Résumé

## Plan Chap. 9 - Expressions Régulières

- 1 Motivations
- 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- 3 Applications en informatique : commandes UNIX
- 4 Résumé

## Motivations

On cherche une notation plus **concise** que les automates pour décrire des langages à états.

### Exemple (Unix - grep)

Écrire un automate pour faire un **grep** sur Unix ou Linux est inconcevable.

### Exemple (Logiciels d'analyse lexicale)

Pour utiliser des logiciels d'analyse lexicale comme Lex ou Flex on doit spécifier les lexemes (token).

### Exemple (Vérification de chaînes de caractères)

Vérifier les adresses emails, dates de naissance, etc dans les formulaires.

## Motivations (suite)

### Exemple (Expression régulière décrivant un email valide)

Selon la RFC 5322 <sup>a</sup>

```
(?:[a-z0-9!#$%&'*/+=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*/+=?^_`{|}~-]+)*)
| "(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]
  \\\[\x01-\x09\x0b\x0c\x0e-\x7f])*")
@ (?: (?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?
| \[(?: (?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}
  (?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)|[a-z0-9-]*[a-z0-9] :
  (?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]
  | \\\[\x01-\x09\x0b\x0c\x0e-\x7f]))+)
```

a. [www.regular-expressions.info/email.html](http://www.regular-expressions.info/email.html)

- Les automates offrent la possibilité de décrire des langages de manière *opérationnelle* : par une sorte de machine (l'automate).
- Les expressions régulières permettent de le faire de manière *déclarative/algébrique*.

## Plan Chap. 9 - Expressions Régulières

### 1 Motivations

- ### 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- Syntaxe
  - Sémantique
  - Quelques propriétés : équivalence et simplification

### 3 Applications en informatique : commandes UNIX

### 4 Résumé

## Plan Chap. 9 - Expressions Régulières

### 1 Motivations

### 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés

- Syntaxe
- Sémantique
- Quelques propriétés : équivalence et simplification

### 3 Applications en informatique : commandes UNIX

### 4 Résumé

## Expressions régulières : syntaxe

Soit  $\Sigma$  un alphabet.

### Définition (Expressions régulières : syntaxe)

Les *expressions régulières* sur  $\Sigma$  sont définies inductivement (par les règles suivantes) :

- $\epsilon$  et  $\emptyset$  sont des expressions régulières sur  $\Sigma$ .
- Si  $a \in \Sigma$  alors  $a$  est une expression régulière sur  $\Sigma$ .
- Si  $e$  et  $e'$  sont des expressions régulières sur  $\Sigma$  alors  $e + e'$  est une expression régulière sur  $\Sigma$ .
- Si  $e$  et  $e'$  sont des expressions régulières sur  $\Sigma$  alors  $e \cdot e'$  est une expression régulière sur  $\Sigma$ .
- Si  $e$  est une expression régulière sur  $\Sigma$  alors  $e^*$  est une expression régulière sur  $\Sigma$ .

### Notation

L'ensemble des expressions régulières est dénoté par  $ER$ .

### Exemple (Expressions régulières sur $\Sigma = \{a, b\}$ )

- |               |                       |                   |                       |                                   |
|---------------|-----------------------|-------------------|-----------------------|-----------------------------------|
| • $\emptyset$ | • $b \cdot a$         | • $(\emptyset)^*$ | • $a + b$             | • $a + b$                         |
| • $b$         | • $a \cdot \emptyset$ | • $a^*$           | • $a \cdot (b + a)^*$ | • $(b \cdot a \cdot b)^* \cdot b$ |

## Plan Chap. 9 - Expressions Régulières

- 1 Motivations
- 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
  - Syntaxe
  - Sémantique
  - Quelques propriétés : équivalence et simplification
- 3 Applications en informatique : commandes UNIX
- 4 Résumé

## Expressions régulières : sémantique

Les expressions régulières décrivent des *langages*.

### Définition (Expressions régulières : sémantique)

- La sémantique est donnée par l'application  $L : ER \rightarrow \mathcal{P}(\Sigma^*)$  qui associe un langage (unique)  $L(e)$  à (toute expression régulière)  $e$ .
- L'application  $L$  est définie *inductivement* :
  - $L(\epsilon) = \{\epsilon\}$ ,
  - $L(\emptyset) = \emptyset$ ,
  - $L(a) = \{a\}$ ,
  - $L(e + e') = L(e) \cup L(e')$ ,
  - $L(e \cdot e') = L(e) \cdot L(e')$ ,
  - $L(e^*) = L(e)^*$ .

### Vocabulaire

Un langage  $L$  est **régulier** ssi il existe une expression régulière  $e$  telle que  $L(e) = L$ .

### Exemple (Langage régulier)

Les langages sur  $\{a, b\}$  dénotés par les expressions régulières suivantes sont réguliers

- $(a)^*$  - langage des mots contenant que des  $a$
- $(a \cdot b)^*$  - langage des mots formés par une répétition finie du facteur  $a \cdot b$ .

## Convention et notation

- Nous ne ferons plus la distinction explicitement entre
  - $\dot{\phantom{x}}$  et  $\cdot$ , d'une part ;
  - $\ast$  et  $*$ , d'autre part.
- Nous voulons aussi pouvoir écrire des expressions comme
  - $a + b + c$  à la place de  $(a + b) + c$ , et
  - $a + b^*$  à la place de  $(a + (b^*))$ .
- Pour éviter les ambiguïtés nous permettons l'utilisation des parenthèses et admettons les priorités suivantes dans un ordre décroissant :
  - 1  $\ast$
  - 2  $\cdot$
  - 3  $+$
- Nous écrivons aussi  $ee'$  à la place de  $e \cdot e'$ .

### Exemple (Convention et notation)

Les expressions

- $e_1 + e_2^*$  et  $e_1 + (e_2)^*$ , d'une part,
- $e_1 + e_2 \cdot e_3$  et  $e_1 + (e_2 \cdot e_3)$ , d'autre part,

dénotent les mêmes ensembles.

## Exemples d'expressions régulières

Soit  $\Sigma = \{a, b\}$ .

Exemple (Mots ne contenant que des  $a$ )

Exemple (Mots constitués de répétitions du facteur  $ab$ )

Exemple (Mots avec nombre pair de  $a$ )

Exemple (Mots avec nombre impair de  $b$ )

Exemple (Mots avec nombre pair de  $a$  ou nombre impair de  $b$ )

## Notation - Opérateur $^+$ (en exposant)

### Opérateur $^+$ (en exposant)

Soit  $e$  une expression régulière, nous notons  $e^+$  pour  $e \cdot e^*$ .

*L'expression régulière  $e^+$  dénote le langage des mots qui sont formés par la concaténation d'au moins un mot dans le langage dénoté par l'expression régulière  $e$ .*

### Exemple (Expression régulière avec opérateur $^+$ (en exposant))

Soit  $\Sigma = \{a, b, c, d\}$ , considérons l'expression régulière  $e = ab + cd$ .  
Alors l'expression régulière  $e^+$  dénote le langage

$$\{ab, cd, abab, abcd, cdab, cdcd, \dots\}$$

### Propriété

Soit  $e$  une expression régulière telle que  $\epsilon \in L(e)$ , alors  $L(e^+) = L(e^*)$ .

## Plan Chap. 9 - Expressions Régulières

### 1 Motivations

- ### 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- Syntaxe
  - Sémantique
  - Quelques propriétés : équivalence et simplification

### 3 Applications en informatique : commandes UNIX

### 4 Résumé

## Équivalence entre expressions régulières

### Définition (Équivalence entre deux expressions régulières)

Les expressions régulières  $e_1$  et  $e_2$  sont dites *équivalentes* lorsque :

$$L(e_1) = L(e_2).$$

(C'est-à-dire lorsque ces expressions régulières dénotent les mêmes langages.)

### Notation

Lorsque  $e_1$  et  $e_2$  sont équivalentes, nous le notons  $e_1 \equiv e_2$ .

**Remarque** La relation  $\equiv$  entre expressions régulières est effectivement une relation d'équivalence car la relation d'égalité est une relation d'équivalence sur les langages.  $\square$

## Équivalence entre expressions régulières : identités classiques

### Identités classiques

Expression régulière	Expression régulière équivalente	Remarque
$e + \emptyset$	$e$	trivial
$e \cdot \epsilon$	$e$	trivial
$e \cdot \emptyset$	$\emptyset$	trivial
$(e + f) + g$	$e + (f + g)$	associativité
$(e \cdot f) \cdot g$	$e \cdot (f \cdot g)$	associativité
$e \cdot (f + g)$	$(e \cdot f) + (e \cdot g)$	distributivité
$(e + f) \cdot g$	$(e \cdot g) + (f \cdot g)$	distributivité
$e + f$	$f + e$	commutativité



## Équivalence entre expressions régulières : identités classiques

### Identités classiques

Expression régulière	Expression régulière équivalente	Remarque
$e^*$	$\epsilon + e \cdot e^*$	apériodicité
$e^*$	$\epsilon + e^* \cdot e$	apériodicité
$(\emptyset)^*$	$\epsilon$	définition de l'opérateur de Kleene
$e + e$	$e$	idempotence
$(e^*)^*$	$e^*$	idempotence

## Équivalence entre expressions régulières

Soit  $\Sigma$  un alphabet tel que  $a \in \Sigma$  et  $e$  une expression régulière sur  $\Sigma$ .

### Exemple (Expressions régulières équivalentes)

- Les expressions  $(a + \epsilon)^*$  et  $a^*$  sont équivalentes.
  - $L((a + \epsilon)^*) \subseteq L(a^*)$ . Soit  $w \in L((a + \epsilon)^*)$ , d'après la sémantique des expressions régulières, soit i)  $w$  est  $\epsilon$  soit ii) s'écrit  $w_1 \cdot w_2 \cdots w_n$  avec  $w_i \in L(a + \epsilon)$ . Premier cas :  $w = \epsilon$  et dans ce cas  $w \in L(a^*)$  d'après la sémantique de  $a^*$  (fermeture de Kleene de  $L(a)$ ). Deuxième cas :  $w$  est formé par la concaténation des mots  $a$  et  $\epsilon$  et peut donc s'écrire  $w = w'_1 \cdots w'_m$  avec  $m \leq n$  et  $w_i = w'_i$  pour. Donc  $w \in \{a\}^* = L(a^*)$ .
  - $L((a + \epsilon)^*) \supseteq L(a^*)$ . On a  $L(a^*) = L(a)^* = (\{a\})^* \subseteq (\{a\} \cup X)^*$ , pour n'importe quel langage  $X$  et en particulier lorsque  $X = \{\epsilon\}$ .
- Les expressions  $(e + \epsilon)^*$  et  $e^*$  sont équivalentes. La preuve suit un principe similaire au précédent en raisonnant sur  $L(e)$  au lieu de  $L(a) = \{a\}$ .
- Les expressions  $\epsilon + e + ee^*e$  et  $e^*$  sont équivalentes, pour n'importe quelle expression régulière  $e$ . La preuve suit un principe similaire au précédent.

Est-ce que l'équivalence entre expressions régulières est décidable ?

## Simplification d'expressions régulières

### Principe de simplification d'expressions régulières

Si  $e$  et  $e'$  sont deux expressions régulières équivalentes (cad  $e \equiv e'$ ,  $L(e) = L(e')$ ), alors on peut substituer  $e$  par  $e'$  dans une expression régulière  $r$  sans changer le langage que  $r$  dénote.

### Exemple (Simplification d'expressions régulières)

Considérons l'expression régulière  $r = (a + \epsilon)^* + b^* + c \cdot d^*$ .

Comme  $L((a + \epsilon)^*) = L(a^*)$ ,  $r$  peut se simplifier en  $a^* + b^* + c \cdot d^*$ .

### Quelques faits utiles pour la simplification

Soient  $e_1$  et  $e_2$  deux expressions régulières.

- Si  $L(e_1) \subseteq L(e_2)$ , alors  $L(e_1 + e_2) = L(e_2)$ . Donc  $e_1 + e_2$  peut être remplacée par  $e_2$  dans une expression régulière sans changer le langage qu'elle dénote.
- $L(e \cdot \epsilon) = L(\epsilon \cdot e) = L(e)$ .

Est-ce qu'on sait déterminer automatiquement si  $e_1 + e_2$  peut être remplacée par  $e_2$  ?  
C'est-à-dire, est-ce que  $L(e_1) \subseteq L(e_2)$  est décidable ?

## Simplification d'expressions régulières : exemples

### Exemple (Simplification d'expressions régulières)

Expression régulière	Expression régulière simplifiée
$e^* + e$	$e^*$
$e^+ + e$	$e^+$
$e^+ + \epsilon$	$e^*$
$(e + \epsilon)^*$	$e^*$
$a + ab^*$	$ab^*$
$e + ee^*e$	$e^+$
$\epsilon + e + ee^*e$	$e^*$

**Remarque** Voir TD pour plus d'exemples de simplification, et les preuves d'équivalence entre ces expressions régulières. □

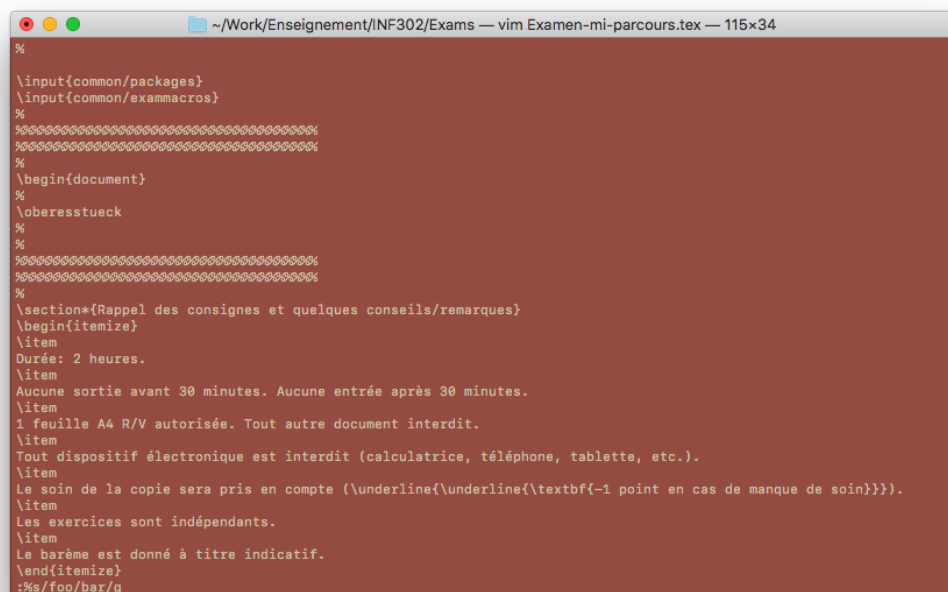
## Plan Chap. 9 - Expressions Régulières

- 1 Motivations
- 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- 3 Applications en informatique : commandes UNIX
- 4 Résumé

## Commandes UNIX

Beaucoup de commandes UNIX permettent de spécifier les chaînes de caractères à utiliser avec des expressions régulières.

- Editeurs de textes : `vi(m)`, `emacs`, `nano`



```

%
\input{common/packages}
\input{common/exammacros}
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
\begin{document}
%
\oberesstueck
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
\section*{Rappel des consignes et quelques conseils/remarques}
\begin{itemize}
\item
Durée: 2 heures.
\item
Aucune sortie avant 30 minutes. Aucune entrée après 30 minutes.
\item
1 feuille A4 R/V autorisée. Tout autre document interdit.
\item
Tout dispositif électronique est interdit (calculatrice, téléphone, tablette, etc.).
\item
Le soin de la copie sera pris en compte (\underline{\underline{\textbf{-1 point en cas de manque de soin}}}).
\item
Les exercices sont indépendants.
\item
Le barème est donné à titre indicatif.
\end{itemize}
:%%/foo/bar/g

```

## Commandes UNIX

- Recherche d'une chaîne dans un texte : `grep`

```
grep *exam* /Users/prof/teaching/*.*
```

Affiche les lignes contenant `*exam*` dans tous les fichiers (avec extension) du répertoire `/Users/prof/teaching/`.

- Transformation de chaîne dans un texte/fichier : `sed`

```
sed s/2017/2018/ <old.tex >new.tex
```

Remplace les occurrences de 2017 dans `old.tex` par 2018 et met le résultat dans `new.tex`.

- Recherche de fichiers : `find`

```
find . -name *examen* -print
```

Recherche dans le répertoire courant et dans les sous répertoire (.) les fichiers compatibles avec l'expression `*examen*`.

## Commandes UNIX (suite)

- Evaluation d'expressions : `expr`

```
$chaine : expression_reguliere
```

Comparaison de `$chaine` avec `expression_reguliere`

- Filtre et traitement de données en ligne : `awk`

```
pattern { action }
```

```
BEGIN { print "START" }
      { print          }
END   { print "STOP" }
```

Ajoute 1 ligne avec START au début et 1 ligne avec END à la fin d'un fichier.

```
BEGIN { print "File\tOwner" }
      { print $8, "\t", $3 }
END   { print " - DONE -" }
```

Script `fileOwner` qui affiche le propriétaire d'un fichier.

```
ls -l | FileOwner
```

Utilisation du script `FileOwner` en ligne de commande.

## Plan Chap. 9 - Expressions Régulières

- 1 Motivations
- 2 Expressions régulières : définition (syntaxe et sémantique) et quelques propriétés
- 3 Applications en informatique : commandes UNIX
- 4 Résumé

## Résumé

### Résumé

- Définition des **expressions régulières** :
  - syntaxe,
  - sémantique.
- Équivalence entre expressions régulières.
- Simplification d'expressions régulières
- Commandes UNIX utilisant les expressions régulières.