

# Grammaires formelles : Automates

Karën Fort

karen.fort@sorbonne-universite.fr / <https://members.loria.fr/KFort/>

# Quelques sources d'inspiration

par ordre d'importance décroissant

- ▶ *Introduction à la calculabilité* (Pierre Wolper) – InterEditions, 1991
- ▶ cours de G. Perrier (Université de Lorraine)
- ▶ cours de F. Yvon (LIMSI / CNRS) sur les transducteurs
- ▶ cours en ligne de J-F. Perrot (Paris 6), avec son accord :  
<http://pagesperso-systeme.lip6.fr/Jean-Francois.Perrot/inalco/Automates/Cours5.html>
- ▶ Wikipédia sur les transducteurs (consultée le 23 oct. 2014)

Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Types d'automates

Transducteurs finis

Conclusion sur les automates

Pour finir

# D'une expression régulière à un automate

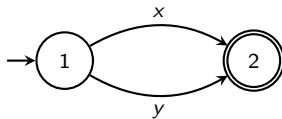
## Les trois opérations de base

- ▶ union :  $x|y$
- ▶ concaténation :  $xy$
- ▶ étoile de Kleene :  $x^*$

# D'une expression régulière à un automate

Les trois opérations de base

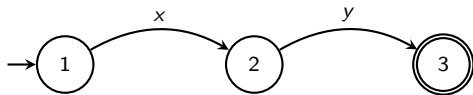
► union :  $x|y$



# D'une expression régulière à un automate

Les trois opérations de base

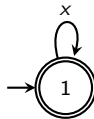
► concaténation :  $xy$



# D'une expression régulière à un automate

Les trois opérations de base

► étoile de Kleene :  $x^*$



# D'une expression régulière à un automate

Décomposons et testons, au tableau

- ▶  $ab^*a$
- ▶  $(a|b)c$
- ▶  $a(bc)^*$



# Expressions régulières

http:

[//www.cheatography.com/davechild/cheat-sheets/regular-expressions/](http://www.cheatography.com/davechild/cheat-sheets/regular-expressions/)

Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Exercice 1

Exercice 2

Exercice 4

Types d'automates

Transducteurs finis

Conclusion sur les automates

Pour finir

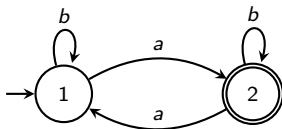
# Exercice 1

Automate qui reconnaît :

1. les mots (sur  $L = \{a, b\}$ ) contenant un nombre impair de lettres « a »
2. (sur  $L = \{a, b\}$ )  $a * b^*$
3. les mots (sur  $L = \{a, b\}$ ) commençant par  $ab$
4. les mots (sur  $L = \{a, b\}$ ) finissant par  $ab$
5. le digicode (imaginaire) : 0278B

## Exercice 1-1

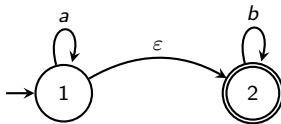
mots (sur  $L = \{a, b\}$ ) contenant un nombre impair de lettres « a »



## Exercise 1-2

(sur  $L = \{a, b\}$ )  $a^* b^*$

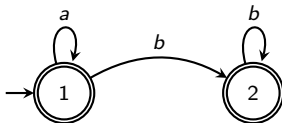
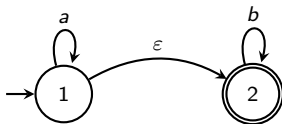
2 solutions :



## Exercice 1-2

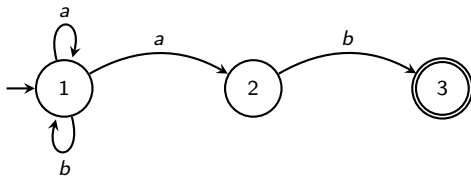
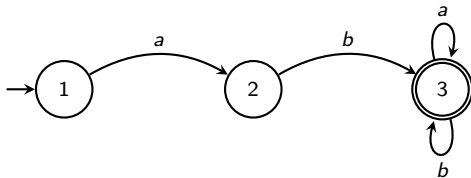
(sur  $L = \{a, b\}$ )  $a^* b^*$

2 solutions :



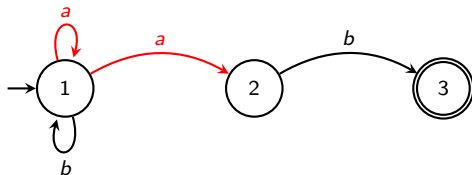
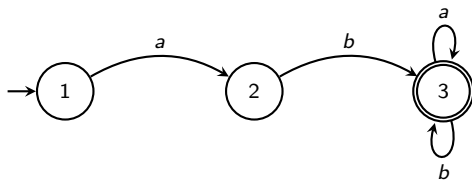
## Exercice 1-3 et 1-4

mots (sur  $L = \{a, b\}$ ) commençant|finissant par  $ab$



## Exercice 1-3 et 1-4

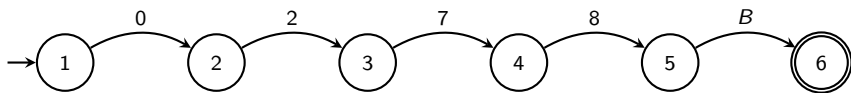
mots (sur  $L = \{a, b\}$ ) commençant|finissant par  $ab$





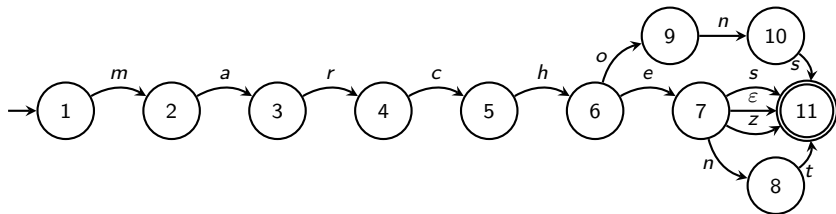
## Exercice 1-5

digicode (imaginaire) : 0278B



## Exercice 2-2

formes fléchies du verbe « marcher » à l'indicatif présent



Expression régulière possible (à tester sur <https://www.debuggex.com/>) :  
*march(e|s|z|nt)?|ons)*

## Exercice 3

l'automate qui reconnaît un numéro de sécurité sociale de français nés en métropole

Correction sur :

<http://automatonsimulator.com/>

Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Types d'automates

- Automate asynchrone/synchrone

- Automate non déterministe/déterministe

- Automate minimal

Transducteurs finis

Conclusion sur les automates

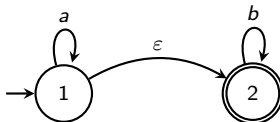
Pour finir

# Automate asynchrone (epsilon)

## Définition

Un automate asynchrone est un automate fini qui possède des transitions sur le mot vide ( $\epsilon$ ).

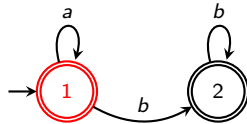
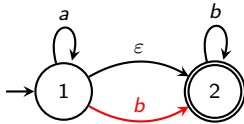
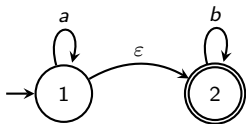
Exemple :



# Rendre synchrone un automate asynchrone

## algorithme de fermeture transitive

1. Pour chaque chemin d'un état  $x$  à un état  $y$  formé de  $\varepsilon$ -transitions, et pour chaque transition de  $y$  à un état  $z$  portant une lettre  $\alpha$ , ajouter une transition de  $x$  à  $z$  d'étiquette  $\alpha$
2. Pour chaque chemin d'un état  $x$  à un état  $y$  terminal formé de  $\varepsilon$ -transitions, ajouter  $x$  à l'ensemble des états terminaux
3. Supprimer les  $\varepsilon$ -transitions



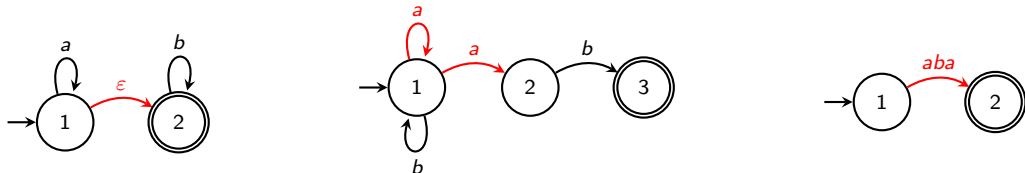
# Automate non déterministe

## Définition

Un automate non déterministe est un automate fini où l'on permet :


- ▶ plusieurs transitions correspondant à la même lettre dans chaque état (transitions non exclusives)
- ▶ des transitions sur le mot vide
- ▶ des transitions sur des mots de longueur supérieure à 1

Exemples :



## Non déterminisme : conséquences

- ▶ plusieurs exécutions sur un même mot (il suffit qu'une de ces exécutions mène à un état final pour que le mot soit accepté)  
⇒ on laisse à l'automate le soin de choisir
- ▶ permet de représenter très simplement la disjonction
- ▶ mais on perd du temps dans le parcours

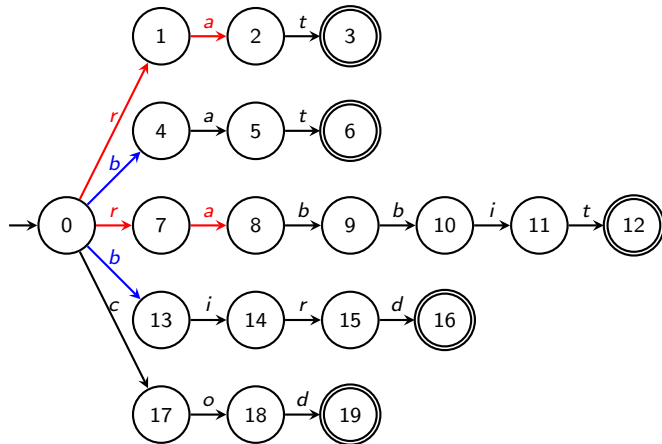
 on peut **toujours** remplacer un automate fini non déterministe par un automate fini déterministe équivalent



# D'une regexp à un automate non déterministe

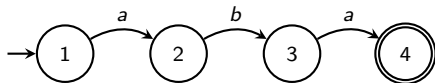
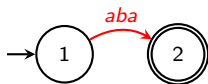
Quelques animaux anglais

*animals = rat|bat|rabbit|bird|cod;*



# Rendre déterministe un automate non déterministe

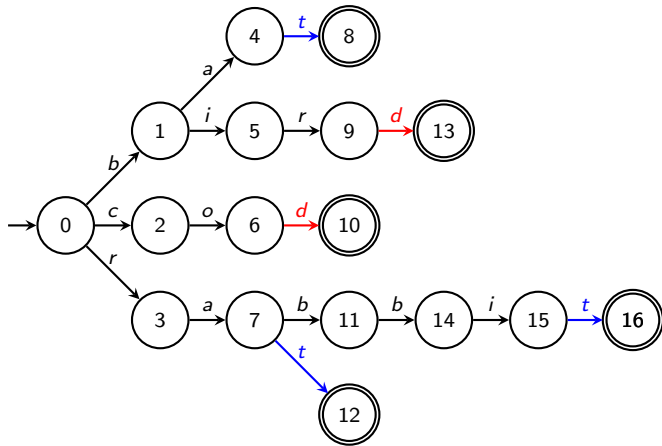
Cas des transitions sur des mots de longueur supérieure à 1



# Rendre déterministe un automate non déterministe

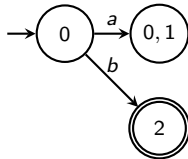
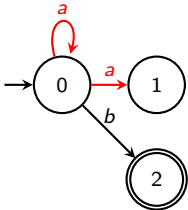
Cas (simple) de transitions non exclusives

*animals = rat|bat|rabbit|bird|cod;*



# Rendre déterministe un automate non déterministe

Cas des transitions non exclusives



# Unicité de l'automate fini minimal

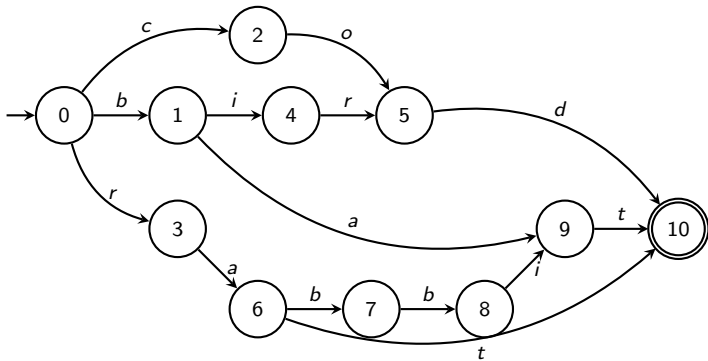
## Définition

« Il existe, pour tout automate fini, un **seul** automate fini déterministe minimal (c'est-à-dire ayant un nombre minimal d'état) qui est équivalent à l'automate donné. »  
[Wikipédia, Automate fini, consultée le 01/11/14]

# Minimiser un automate

Quelques animaux anglais

*animals = rat|bat|rabbit|bird|cod;*



Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Types d'automates

**Transducteurs finis**

- Automates et transducteurs (finis)

- Transducteurs et TAL

- Composer des transducteurs

Conclusion sur les automates

Pour finir

# Transducteur fini

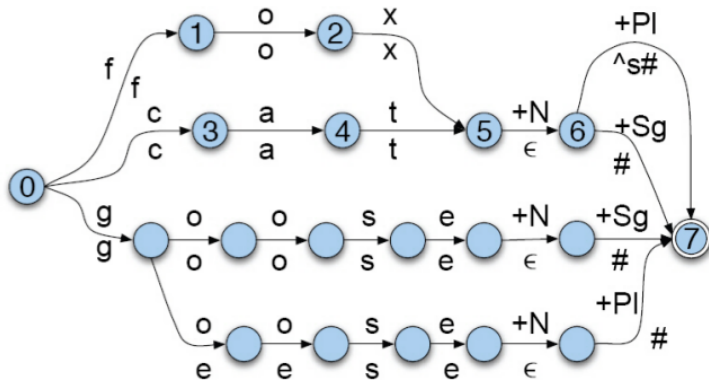
## Définition

Un transducteur fini est un automate fini avec **sorties** :

- ▶ extension des automates finis
- ▶ au lieu de simplement accepter ou refuser le mot, il le transforme en un ou plusieurs mots sur un alphabet de sortie



## Exemple de transducteur



? Que fait ce transducteur ?

# Applications

en Traitement Automatique des Langues (TAL) :

- ▶ morphologie
- ▶ phonologie
- ▶ correction orthographique
- ▶ reconnaissance de la parole (TAP)
- ▶ etc

## Morpho-syntaxe : solutions en extension

```
<lexicalEntry id="championne_1">
  <feminineVariantOf target="champion_1">champion</feminineVariantOf>
  <formSet>
    <lemmatizedForm>
      <orthography>championne</orthography>
      <grammaticalCategory>commonNoun</grammaticalCategory>
      <grammaticalGender>feminine</grammaticalGender>
    </lemmatizedForm>
    <inflectedForm>
      <orthography>championne</orthography>
      <grammaticalNumber>singular</grammaticalNumber>
    </inflectedForm>
    <inflectedForm>
      <orthography>championnes</orthography>
      <grammaticalNumber>plural</grammaticalNumber>
    </inflectedForm>
  </formSet>
  <originatingEntry target="TLF">CHAMPION, ONNE, subst.</originatingEntry>
</lexicalEntry>
```

Morphalou 2 = **160 Mo**

Temps d'accès ?

# Morpho-syntaxe : transducteurs finis

Lexique du français :

- ▶ taille une fois compilé : **800 Ko** (français), 500 Ko (anglais)
- ▶ accès immédiat
- ▶ gestion des langues agglutinantes, des cycles, ...

## Morpho-syntaxe : transducteurs finis en Kleene

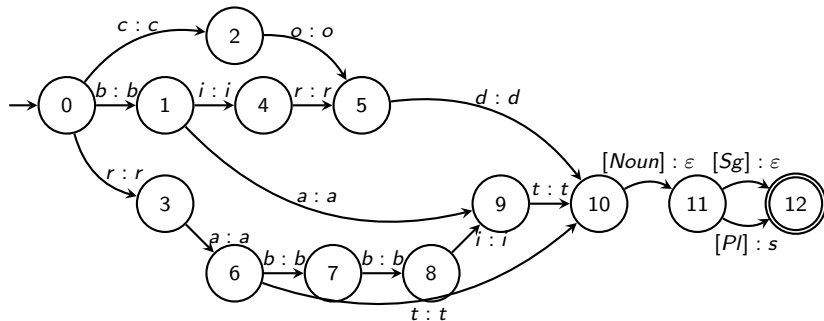
```
// a set of noun roots
$nroot = rat|bat|rabbit|bird|cod ;
$ncat = '[Noun]':"" ; // [Noun] tag on the upper side,
// the empty string on the lower side

$num = '[Sg]':"" | '[Pl]':s;
// [Sg] on the upper side,
// and empty string on the lower.
// [Pl] on the upper side,
// and s on the lower

// now just concatenate the three FSMs
$nouns = $nroot $ncat $num ;
test $nouns ; // and launch the test window
```

# Morpho-syntaxe : transducteurs finis en Kleene

## Visualisation



# Bidirectionnalité des transducteurs

Les transducteurs peuvent être appliqués dans un sens ou dans l'autre :

- ▶ analyse : birds  $\rightarrow$  bird [Noun] [PI]
- ▶ génération : bird [Noun] [PI]  $\rightarrow$  birds



## Morpho-syntaxe : transducteurs finis en XFST

```
[ [l e a v e %+VBZ .x. l e a v e s] |  
[l e a v e %+VB .x. l e a v e] |  
[l e a v e %+VBG .x. l e a v i n g] |  
[l e a v e %+VBD .x. l e f t] |  
[l e a v e %+NN .x. l e a v e] |  
[l e a v e %+NNS .x. l e a v e s] |  
[l e a f %+NNS .x. l e a v e s ] |  
[l e f t %+JJ .x. l e f t] ]
```

APPLY DOWN> leave+VBD

left

APPLY UP> leaves

leave+NNS

leave+VBZ

leaf+NNS

# Composition de transducteurs

## Définition

Soient deux transducteurs finis  $T_1$  et  $T_2$  tels que l'alphabet de sortie de  $T_1$  coïncide avec l'alphabet d'entrée de  $T_2$ .

Le composé  $T_3 = T_2 \circ T_1$  est défini par la relation  $[T_3]$  :

$$x[T_3]y \Leftrightarrow \exists z \text{ tel que } x[T_1]z \text{ et } z[T_2]y$$



la composition de deux transducteurs finis est encore un transducteur fini.



À quoi ça sert ?

## Exemple (factice) d'utilisation de la composition

- ▶ Une phrase = un automate de lettres
  - ▶ Un segmenteur = remplace ponctuation par CR
  - ▶ Un lexique = relation entre mots et catégories morpho-syntaxique
- Composition des trois : une relation entre séquences de lettres et séquences de catégories

## Exemple (réel) d'utilisation de la composition

1 transducteur pour le portugais  $\circ$  1 transducteur filtre  
= 1 transducteur pour le brésilien

Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Types d'automates

Transducteurs finis

Conclusion sur les automates

- Motivation pour leur utilisation

- Automates et langages

Pour finir

# Intérêt des automates

Minimisation :

- ▶ du temps de parcours
- ▶ de l'espace de stockage

# Automates vs expressions rationnelles

Deux formalismes permettent de décrire des langages :

1. les expressions rationnelles :

- ▶ basées sur la description d'ensembles de mots
- ▶ ensembles élémentaires et opérations sur ces ensembles

2. les automates finis :

- ▶ définissent les langages de manière opérationnelle
- ▶ donnent un mécanisme permettant de les reconnaître

1 langage = 1 automate minimal = une infinité d'expressions rationnelles

# Langage fini et automate

- ▶ tout langage fini est reconnaissable par un automate
- ▶ les langages reconnus par des automates sont des langages réguliers

⇒ tout langage fini est régulier



# Langage infini et automate

Lemme de l'étoile (*pumping lemma*)

## Définition

Si un langage  $L$  reconnu par un automate fini  $A$  est infini, alors tout mot suffisamment long de  $L$  contient un facteur non vide qui peut être répété un nombre arbitraire de fois sans sortir du langage.

Si l'automate  $A$  a  $n$  états, suffisamment long signifie de longueur supérieure à  $n$ .

Autrement dit :

« tout mot suffisamment long d'un langage rationnel peut être pompé, au sens qu'une partie centrale du mot peut être répétée un nombre quelconque de fois, et que chacun des mots produits est encore dans le langage. »

[Wikipédia, Lemme de l'étoile, consultée le 01/11/14]

# Langages non réguliers (non rationnels)

Exemples :

- ▶ langage dont les mots sont de la forme « un certain nombre de  $x$ , suivis du même nombre de  $y$  » :  $L = \{x^n y^n | n \geq 0\}$
- ▶ langage des systèmes de parenthèses

🗨️ Donner un autre exemple de langage non rationnel (en langue naturelle, par exemple)

☀️ Le lemme est une condition nécessaire mais non suffisante de rationalité

Sources

Pratique des automates

Correction (de certains exercices) du TD précédent

Types d'automates

Transducteurs finis

Conclusion sur les automates

Pour finir

CQFR : Ce Qu'il Faut Retenir

TD



- ▶ types d'automates :
  - ▶ synchrones / asynchrones
  - ▶ déterministes / non déterministes
  - ▶ minimal ou non
- ▶ lien avec les langages et les expressions rationnelles
- ▶ transducteurs :
  - ▶ définition
  - ▶ composition

## Exercice 1

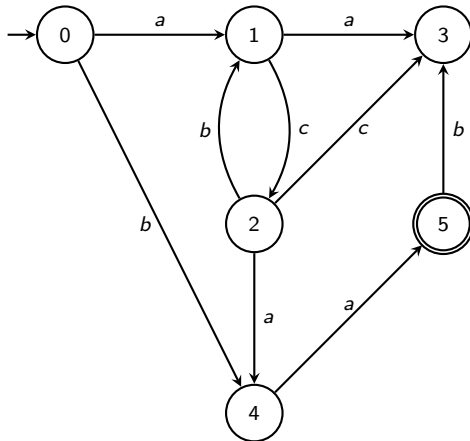
On considère l'alphabet usuel des caractères utilisés pour écrire un texte en français. Dans chacun des cas suivants, construire (à la main) un automate qui reconnaît le langage défini par l'expression régulière :

1.  $ab^*|a^*b$
2.  $[ab]?([\text{^}a]|b)^*$
3.  $[abc]\{1,3\}e^+$
4.  $a^*b^+(c|d|e|f)g?$

Note :  $[ab]$  correspond à "a ou b", ? signifie "éventuellement" et  $[\text{^}a]$  signifie "non a"

## Exercice 3

Déterminer **une** expression régulière représentant le langage reconnu par l'automate ci-dessous :



## Exercice 4

Construire un transducteur représentant la morphologie des mots ayant la racine « jet »  
(pour des raisons de lisibilité, on se limitera au(x) noms, adjectifs et verbes au présent de l'indicatif)