Langages rationnels et automates finis. Exemples et applications. [CAR]

Motivations Premier contact avec les langages formels, prépare à l'introduction d'un cadre formel pour l'étude des langages algébriques et plus généralement des concepts liés aux MT. Enjeu sur l'outil de modélisation que sont les automates. L'occasion de consolider les acquis du formalisme des structures inductives.

I. Langages réguliers et langages reconnaissables

A. Mot et Langage

Définition 1 **Lettre et alphabet.** Un alphabet est un ensemble fini non vide de symboles appelés lettres. Par la suite on notera les alphabets sur Σ.

Définition 2 **Un mot** w sur un Σ est une suite finie de lettre $w_1, ..., w_n$ avec $n \in \mathbb{N}$. Si w est composé de n lettres on dit qu'il est de taille n et on note $|w| = n$. Le mot de taille 0 sera noté ε. On définit la concaténation de $u = u_1...u_n$ et $v = v_1...v_m$ comme le mot $u.v = u_1...u_n v_1...v_m$ de taille $|u| + |v|$.

Définition 3 **Un langage** sur Σ est un ensemble fini ou infini de mots sur Σ.

Exemple 4 $L_1 = \{\varepsilon, aa, aab\}$ est le langage fini sur $\Sigma\{a,b\}$ composé du mot vide, du mot aa de taille 2 et du mot aab de taille 3. $L_2 = \{w$ avec autant de a que de $b\}$ est un langage infini.

Transition 5 Briques de base pour la suite de la leçon. On va miantenant pouvoir manipuler des languages et notamment définir des opérations sur ces derniers.

B. Langage réguliers

Définition 6 **Opérations régulières**

- Union $(+)$: $L + L' = L \cup L' = \{w \mid w \in L$ ou $w \in L'\}$
- Concaténation $(.)$: $L.L' = \{u.v \mid u \in L, v \in L'\}$
- Etoile de Kleene $(*)$: $L^0 = \{\varepsilon\}$, $\forall i \in \mathbb{N}, L^{i+1} = L.L^i$, $L^* = \cup_{i \in \mathbb{N}} L^i$

Définition 7 **Langage régulier** L'ensemble $\mathrm{Rat}(\Sigma)$ des langages réguliers (ou rationnels) sur Σ est le plus petit ensemble stable par opérations régulières contenant $\emptyset$ et $\{a\}$ pour $a \in \Sigma$.

Exemple 8

- $\{a \mid a \in \Sigma^*\}^* = \{w \mid w$ est un mot de $\Sigma\}$, par abus de notation on note ce langage Σ^*.
- $\{a\}^*.\{b\}^* = \{a^n b^m \mid n, m \in \mathbb{N}\}$

Définition 9 **Expressions régulières** L'ensemble des expressions régulières $\mathcal{E}(\Sigma)$ sur Σ est un langage sur l'alphabet $\{0, 1, |, ., *, (,)\} \cup \Sigma$ défini inductivement par :

- $0, 1 \in \mathcal{E}(\Sigma)$
- $\forall a \in \Sigma, a \in \mathcal{E}(\Sigma)$
- Si $e_0, e_1, e_2 \in \mathcal{E}(\Sigma)$ alors $(e_0^*), (e_1 \mid e_2), (e_1.e_2) \in \mathcal{E}(\Sigma)$

Remarque 10 On peut voir les symboles $+, |$ et $*$ comme des opérateurs sur les expressions régulières et poser des règles de priorités pour d'affranchir des parenthèses : $*$ plus prioritaire que . plus prioritaire que $|$: $((a.(b^*) \mid (a^*))$ se réecrit $a.b^* \mid a^*$. De même, on peut omettre d'écrire . pour alléger les notation : on obtient $ab^* \mid a^*$.

Définition 11 **Langage dénoté** On appelle $\mathcal{L}(E)$ le language dénoté par l'expression régulière e, défini inductivement comme :

- $\mathcal{L}(\emptyset) = \emptyset$
- $\mathcal{L}(e^*) = \mathcal{L}(e)$
- $\mathcal{L}(e_1.e_2) = \mathcal{L}(e_1).\mathcal{L}(e_2)$
- $\mathcal{L}(\varepsilon) = \{\varepsilon\}$
- $\forall a \in \Sigma,\ \mathcal{L}(a) = \{a\}$
- $\mathcal{L}(e_1|e_2) = \mathcal{L}(e_1)|\mathcal{L}(e_2)$

Propriété 12 $\mathcal{L}(\mathcal{E}(\Sigma)) = \mathrm{Rat}(\Sigma)$

Exemple 13 $a^*.b^*$ dénote $\{a\}^*.\{b\}^*$ et $ab^* \mid a^*$ dénote $(\{a\}.\{b\}^*) + \{a\}^*$

Application 14 Les expressions régulières sont un moyen très succint de représenter des motifs simples(réguliers). Le standard **POSIX** est une extension des expressions régulières servant à la recherche de motif.

Transition 15 Modèle expressif mais peut algorithmique. Comment faire de la reconnaissance de motif avec des expressions régulières, par exemple?

C. Langages reconnaissables

Définition 16 **Un automate fini** $A = (Q, \Sigma, \delta, I, F)$ est la donnée :
- d'un ensemble d'états Q
- d'un alphabet Sigma
- d'une fonction de transition $\delta : Q \times \Sigma \longrightarrow \mathcal{P}(Q)$
- d'ensembles $I, F \subseteq Q$ d'états initiaux et finaux.

Définition 17 **Mot et langage reconnu.** Soit un automate $A = (Q, \Sigma, \delta, I, F)$. Un chemin dans A est une suite de transitions entre les états de $A : q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} q_n$ telle que $\forall i \in [\![1, n]\!], q_i \in \delta(q_{i-1} a_i)$. Un tel chemin est dit étiqueté par le mot $a_1 \dots a_n$. On dit que w est reconnu par A, s'il existe un chemin de q_0 à q_n étiqueté par w tel que $q_0 \in I$ et $q_n \in F$.
Le language reconnu par A, noté $\mathcal{L}(A)$, est l'ensemble des mots reconnus par A.

Définition 18 **Langage reconnaissable** Un langage est reconnaissable s'il existe un automate qui le reconnait. L'ensemble des langages sur Σ qui sont reconnaissables est noté $\mathrm{Rec}(\Sigma)$.

Activité 19 **Modélisation par automate fini** Machine à café, ascenceur, etc...

Applications 20
- Machines à états (architecture, robotique, ...)
- Model-Checking [hors-programme] (automates de Büchi, automates temporisés, ...)

Exemple 21
- A_1 reconnait le langage dénoté par $a.(b|c)$.
- A_2 reconnait le langage dénoté par $a^*.a.b^*$

Schémas de deux automates

Définition 22 **Une ε-transition** (ou transition spontanée) est une transition étiquetée par ε. Formellement, un automate qui autorise les ε-transition est tel que $\delta \in Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$.

Propriété 23 Les languages reconnu par automates avec ε-transition sur Σ sont exactement $\mathrm{Rec}(\Sigma)$.

DEV

Théorème 24 **Théorème de Kleene** $\mathrm{Rat}(\Sigma) = \mathrm{Rec}(\Sigma)$

Démonstration 25 $\subseteq$: Méthode de Thompson $\mathcal{O}(|e|)$.
$\supseteq$: Algorithme de McNaughton-Yamada $\mathcal{O}(|Q^4|)$.

Méthode 26 Construction de Glushkov: comme alternative à Thompson.
Algorithme de Brzozowski et McCluskey : comme alternative McNaughton-Yamada.

DEV

Remarque 27 **Calcul *Premier Nullable Dernier*.** La construction de Glushkov utilise les fonction *Premier*, *Nullable*, *Dernier* pour calculer un automate qui reconnait tous les mots représentés par une expression régulière.

Propriété 28 Les languages reconnaissables sont stables par opérations régulières.

Lemme 29 **Lemme de l'étoile** Pour tout langages L reconnu par un automate avec au plus N états et tout $w \in L$, alors il existe u_1, v, u_2 tel que :
- $w = u_1 v u_2$
- $|v| > 0$
- $\forall i \in \mathbb{N}, u_1 v^n u_2 \in L$
- $|u_1 v| \leq N$

Application 30 **Des languages non-reconnaissables:** existent $\{a^n b^n \mid n \in \mathbb{N}\}$.

Transition 31 On dispose de l'outil pour réfléchir algorithmiquement sur les languages réguliers/reconnaissables. Quelles opérations peut-on réalilser sur les automates finis?

II. Opérations sur les automates finis

Rappel 32 **Compositions régulières.** Les automates finis peuvent être composés par opérations régulières.

A. Opérations de base

Définition 33 **Un automate est complet** si $\forall (q,a), \delta(q,a) \geq 1$.

Définition 34 **Automate émondé** Un automate est acccessible s'il existe un chemin d'un état initial vers n'importe quel autre état.

- un automate est co-accessible s'il existe un chemin de tout état vers un état final.
- un automate est émondé lorsqu'il est accessible et co-accessible.

Définition 35 **Automate déterministe** Un automate fini $A = (Q, \Sigma, \delta, I, F)$ est déterministe lorsque : $\forall (q,a), |\delta(q,a)| < 1$ et $|$ I$| = 1$.

Propriété 36 Soit A un automate fini, il existe un automate déterministe qui reconnait $\mathcal{L}(A)$.

Algorithme 37 **Automate des parties** L'automate des parties est calculé en considérant toutes les parties de l'ensemble des états d'un automate non-déterministe. $\mathcal{O}(2^{|Q|})$.

Propriété 38 **Complémentaire et intersection** On peut calculer le complémentaire d'un automate déterministe complet.
On peut calculer l'intersection de deux automates.

B. Minimisation

Définition 39 **Automate minimal** Un automate deterministe est minimal s'il n'existe pas d'automate déterministe avec moins d'état reconnaissant le même langage.

Définition 40 **Résidu** Pour tout mot u et tout langage L, on définit le résidu de L par u comme $u^-1 = \{w \in \Sigma^* | \ uw \in L\}$.

Théorème 41 **de Myhill-Nerode.** Un langage est régulier si et seulement s'il admet un nombre fini de résidus.

Définition 42 **Automate des résidus** L'automate des résidus d'un langage régulier sur Σ est $\mathcal{A}_R = (Q_R, \Sigma, \delta_R, I_R, F_R)$ avec :

- $Q_R = \{u^{-1}L \mid u \in \Sigma^*\}$
- $F_R = \{u^{-1}L \mid u \in L\}$
- $I_R = \{\varepsilon^{-1}L\} = \{L\}$
- $\delta_{R(u^{-1}L,a)} = \{ua^{-1}L\}$

Propriété 43 **L'automate des résidus est minimal**

Définition 44 **Congruence de Nérode** Etant donné un automate déterministe A, on définit la congruence de Nerode sur les états de A par $p \sim q$ si et seulement si pour tout mot w, les existences d'un chemin étiqueté par w depuis p et q vers un état de F sont équivalentes.

Définition 45 **Automate de Nérode** L'automate de Nerode de A est défini à l'aide de la congruence de Nerode : $A_\sim = \left(Q_\sim, \Sigma, \delta_\sim, \{\bar{i} \mid I = \{i\}\}, \left\{\bar{f} \mid f \in F\right\}\right)$ où $Q_\sim$ est le quotient de Q par $\sim$ et $\delta_\sim(\bar{q}, a) = \overline{\delta(q,a)}$ où $\bar{q}$ est la classe d'équivalence de q selon $\sim$.

Propriété 46 L'automate de Nérode est minimal.

Algorithme 47 **L'algorithme de Moore** calcule l'automate de Nerode d'un automate déterministe complet (en temps $\mathcal{O}(|Q|^2)$) en procédant par raffinements successifs d'une approximation de $\sim$:

- $p \underset{0}{\sim} q \iff p, q \in F$
- $p \underset{k+1}{\sim} q \iff \left[p \underset{k}{\sim} q \text{ et } \forall a \in \Sigma, \delta(p,a) \underset{k}{\sim} \delta(q,a)\right]$

III. Applications

A. Recherche de motif

Problème 48 **Recherche de motif.** Entrée: un texte T et un motif m. Sortie: l'indice de la première occurence de m dans T si elle existe, -1 sinon.

Algorithme 49 **Test exhaustif Naïf** : complexité en $\mathcal{O}(|m|.|T|)$

Algorithme 50 **Knuth-Morris-Pratt** Le calcul des bords permet d'avoir l'automate des préfixes et accélérer la recherche de motif: complexité en $\mathcal{O}(|m| + |T|)$.

B. Compilation

Définition 51 **Analyse Lexicale**

DEV

Idée 52 On utilise des expressions régulières pour représenter les motifs auxquels doivent correspondre les mots associés à un lexeme donné.

Algorithme 53 **REGEXP2DFA** [DRAG]

Langages rationnels et automates finis. Exemples et applications. [CAR]

I. Langages réguliers et langages reconnaissables

A. Mot et Langage

1 Def Lettre et alphabet.
2 Def Un mot
3 Def Un langage
4 Ex Exemple de language
5 Transition

B. Langage réguliers

6 Def Opérations régulières
7 Def Langage régulier
8 Ex L'ensemble des mots est régulier
9 Def Expressions régulières
10 Rem Priorité des ops sur les regexps
11 Def Langage dénoté
12 Prop RAT = REG
13 Ex Language dénoté par une regexp
14 App Standard POSIX
15 Transition

C. Langages reconnaissables

16 Def Un automate fini
17 Def Mot et langage reconnu.
18 Def Langage reconnaissable
19 Activité Modélisation par automate fini
20 Applications Machines à états et model-checking
21 Ex Deux exemples d'automates
22 Def Une ε-transition
23 Prop ε-transitions ne changent pas l'expressivité
24 Thm Théorème de Kleene
25 Démonstration
26 Métho
27 Rem Calcul *Premier Nullable Dernier.*
28 Prop Stabilité par opérations régulières
29 Lemme Lemme de l'étoile
30 App Des languages non-reconnaissables:
31 Transition

II. Opérations sur les automates finis

32 Rappel Compositions régulières.

A. Opérations de base

33 Def Un automate est complet
34 Def Automate émondé
35 Def Automate déterministe
36 Prop Déterminisation
37 Algo Automate des parties
38 Prop Complémentaire et intersection

B. Minimisation

39 Def Automate minimal
40 Def Résidu
41 Thm de Myhill-Nerode.
42 Def Automate des résidus
43 Prop L'automate des résidus est minimal
44 Def Congruence de Nérode
45 Def Automate de Nérode
46 Prop Automate Nérode minimal.
47 Algo L'algorithme de Moore

III. Applications

A. Recherche de motif

48 Prob Recherche de motif.
49 Algo Test exhaustif Naïf
50 Algo Knuth-Morris-Pratt

B. Compilation

51 Def Analyse Lexicale
52 Idée
53 Algo REGEXP2DFA[DRAG]

Remarques

Les livres possibles pour écrire cette leçon son multiple :

- [TOR Chap. ...]
- [VERT Chap. ...]
- [CAR] particulièrement détaillé pour cette leçon.

Fil directeur Différents modèles sont équivalents → opération possibles → applications.

Notes Lien avec les machines à états en archi / Thomson au lieu de Glushkov / pas de morphismes / pas de logique du premier ordre en application

Bibliographie

[CAR] O. Carton & G. G. Gagnees, *Langages formels: Calculabilité et complexité.*

[DRAG] A. Aho & M. Lam & R. Sethi & J. Ullman, *Compilateurs, Principes, techniques et outils.*

[TOR] T. Balabonski & S. Conchon & J. Filliâtre & K. Nguyen & L. Sartre, *MP2I MPI, Informatique Cours et exercices corrigés.*

[VERT] F. Becker & O. Bournez & J. Carré & M. Liedloff & J. Reichert & G. Rozsavolgyi, *Informatique MP2I-MPI.*