Théorie des langages

Anca Muscholl

LaBRI Bordeaux

Cours licence 2006/07

Organisation

1. Cours : 6 séances (fin mi-mars) ; TD : 11 séances
2. Contrôle continu : 4 interrogations et/ou DM (3 meilleures notes comptent)
3. Note finale $0,75$ exam + $0,25$ max (exam,CC)
4. Notes de cours, feuilles de TD et références : voir
   `http ://www.labri.fr/perso/anca/Langages.html`

Plan

1. Automates finis et mots
2. Automates finis et arbres
3. Automates à pile

Motivations

- Description et analyse de langages (traitement du texte, codes, langages de programmation, langages naturels, . . .)
- Modèles de calcul, conception d'algorithmes

Bibliographie

- J.E. Hopcroft, J.D. Ullman. Introduction to automata theory, languages and computation.
  Addison-Wesley, 1979.
- M. Sipser. Introduction to the theory of computation.
  PWS Publishing Company, 1996.
- J. Berstel. Automates et grammaires (notes de cours,
  Univ. Marne-la-Vallée, 2005).

Définitions

Mots

- Alphabet A, B, Σ, Γ : ensemble fini de lettres (symboles) $a, b, \alpha, \beta, \ldots$
- Mot : suite finie de lettres $w = a_1\, a_2 \cdots a_n$ ($a_i \in \Sigma$) ; longueur n
- Mot vide ϵ (longueur 0)
- Concaténation (produit) de 2 mots : juxtaposition (associative)
- Σ^* : ensemble des mots sur l'alphabet Σ
  $(\Sigma^*, \cdot, \epsilon)$ monoïde libre (généré par Σ)
  $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$ ensemble des mots non-vides
- Longueur $|w|$ de w
- Langage (de mots) $L \subseteq \Sigma^*$ (notation $L, K, X, Y, \ldots$)
- Opérations sur les langages ($L, K \subseteq \Sigma^*$) :
  - opérations booléennes : union $L \cup K$, intersection $L \cap K$, complémentation L^c, différence $L \setminus K = L \cap K^c$
  - produit $LK = \{uv \mid u \in L, v \in K\}$, puissances $L^n = \{u_1 \cdots u_n \mid u_k \in L\}$
  - itération (étoile) $L^* = \bigcup_{n \geq 0} L^n$, itération stricte $L^+ = \bigcup_{n > 0} L^n$
  - quotients $K^{-1}L = \{v \in \Sigma^* \mid \exists u \in K :\ uv \in L\}$ et LK^{-1}

Automates finis sur les mots

Applications : Modélisation des neurones, de circuits logiques (années '50-'60), analyse lexicale en compilation, traitement du texte (ex. reconnaissance de motifs), modélisation de mécanismes de contrôle, ...

Definition

Un automate fini $\mathcal{A} = \langle Q, \Sigma, \delta, I, F \rangle$ sur l'alphabet Σ est composé d'un ensemble fini d'états Q, d'une relation de transitions $\delta \subseteq Q \times \Sigma \times Q$, d'un ensemble $I \subseteq Q$ d'états initiaux, et d'un ensemble $F \subseteq Q$ d'états finaux (ou terminaux).

- Représentation par graphes étiquetés (Σ-étiquettes sur les arcs)
- Calcul dans $\mathcal{A}$ (étiqueté par $a_0 \cdots a_n$) : $q_0 \xrightarrow{a_0 \cdots a_n} q_{n+1}$

$$(q_0, a_0, q_1)(q_1, a_1, q_2) \cdots (q_n, a_n, q_{n+1}), \qquad (q_k, a_k, q_{k+1}) \in \delta, \forall k$$

  Calcul acceptant : $q_0 \in I$, $q_{n+1} \in F$
- Langage reconnu (ou accepté) $\mathcal{L}(\mathcal{A})$: ensemble des mots ayant un calcul acceptant.

Langages reconnaissables

Definition

Un langage $L \subseteq \Sigma^*$ est reconnaissable (ou rationnel, ou régulier) s'il est reconnu par un automate fini.

$\mathrm{Rec}(\Sigma^*)$: ensemble des langages reconnaissables sur l'alphabet Σ

Questions

- Différents types d'automates finis : déterministes non-déterministes sans transitions-ϵ (Définition), avec transitions-ϵ, . . . (alternants, boustrophédons, transducteurs, etc.)
- Autres descriptions des reconnaissables (ex. méchanisme de génération : expressions rationnelles, grammaires, logique) ? Traductions effectives ?
- Propriétés de fermeture de $\mathrm{Rec}(\Sigma^*)$ sous certaines opérations.

Langages reconnaissables

DFA automates déterministes (deterministic finite-state automata),
NFA automates non-déterministes (nondeterministic finite-state automata),

Déterminisation (DFA = NFA)

Tout NFA $\mathcal{A}$ peut être transformé en un DFA équivalent $\mathcal{B}$: $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$.

$\mathcal{B}$: automate des parties (taille exponentielle)

Definition

L'ensemble Rat(Σ^*) des expressions rationnelles sur Σ est défini par :

- $\emptyset$ et a ($a \in \Sigma$) appartiennent à Rat(Σ^*)
- Si E et F sont dans Rat(Σ^*), alors $(E + F)$, $(E \cdot F)$ et E^* aussi.

On identifie une expression rationnelle E et le langage $\mathcal{L}(E)$ décrit par E.

Théorème de Kleene

Rat$(\Sigma^*) =$ Rec(Σ^*), pour tout alphabet Σ.

Kleene

Théorème de Kleene

Rat$(\Sigma^*) =$ Rec(Σ^*), pour tout alphabet Σ.

- Rec$(\Sigma^*) \longrightarrow$ Rat(Σ^*) : algorithme de McNaughton/Yamada, système d'équations linéaires (FOI), ...
- Rat$(\Sigma^*) \longrightarrow$ Rec(Σ^*) : opérations de fermeture de Rec(Σ^*) (FOI), construction de Thomson, Glushkov, ...

McNaughton/Yamada : Soit $\mathcal{A} = \langle Q = \{1, \ldots, n\}, \Sigma, \delta, I, F \rangle$ un NFA.

L'algorithme construit les expressions rationnelles $R_{i,j}^{(k)}$, avec $i, j \in Q$, $k \in Q \cup \{0\}$:
$R_{i,j}^{(k)}$ est l'ensemble des mots qui étiquetent un chemin de l'état i vers l'état j, et qui n'utilise que les états $\{1, \ldots, k\}$ comme états intermédiaires.

- (Base) $R_{i,j}^{(0)} = A$, où $A = \{a \in \Sigma \cup \{\epsilon\} \mid (i, a, j) \in \delta\}$
- (Induction) $R_{i,j}^{(k)} = R_{i,j}^{(k-1)} + R_{i,k}^{(k-1)} (R_{k,k}^{(k-1)})^* R_{k,j}^{(k-1)}$

Kleene

$\mathrm{Rat}(\Sigma^*) \longrightarrow \mathrm{Rec}(\Sigma^*)$

Construction de Glushkov : NFA sans transitions ϵ

- Soit E une expression rationnelle (= mot sur $\Sigma \cup \{\epsilon, +, \cdot, *, (,)\}$).
- $\mathsf{Pos}(E)$ = ensemble des positions de E correspondant aux lettres de Σ
- On suppose que toutes les lettres de Σ dans E sont distinctes (sinon, renommage).
- On calcule (inductivement) les ensembles suivants :
  $\mathsf{first}(E) = \{a \in \mathsf{Pos}(E) \mid \mathcal{L}(E) \cap a\Sigma^* \neq \emptyset\}$,
  $\mathsf{last}(E) = \{a \in \mathsf{Pos}(E) \mid \mathcal{L}(E) \cap \Sigma^* a \neq \emptyset\}$
  $\mathsf{follow}(E) = \{(a, b) \mid \mathcal{L}(E) \cap \Sigma^* ab\Sigma^* \neq \emptyset\}$
- On construit le DFA $\mathcal{A} = \langle \mathsf{Pos}(E) \cup \{q_0\}, \Sigma, \delta, q_0, F\rangle$, où $q_0 \xrightarrow{a} a$ si $a \in \mathsf{first}(E)$, et $a \xrightarrow{b} b$ si $(a, b) \in \mathsf{follow}(E)$; $F = \mathsf{last}(E)$ is $\epsilon \notin \mathcal{L}(E)$, et $F = \mathsf{last}(E) \cup \{q_0\}$ sinon.

Logique

Syntaxe

Soit $\text{Var}_1 = \{x, y, z, \ldots\}$, $\text{Var}_2 = \{X, Y, Z, \ldots\}$ deux ensembles disjoints de variables (on appelera les éléments de Var_1 variables du premier ordre, et ceux de Var_2 variables du second ordre).

Les formules de MSOL (logique monadique du second ordre) sur un alphabet Σ sont définies inductivement :

- (Base) $a(x)$, $y = x + 1$, $x \leq y$, $x \in X$ sont des formules
- (Ind.) Si $\varphi_1, \varphi_2, \varphi$ sont des formules, alors

$$\varphi_1 \wedge \varphi_2, \quad \neg\varphi, \quad \exists x.\, \varphi, \quad \exists X.\, \varphi$$

sont des formules aussi.

Formules dérivées :

$$\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2),\ \varphi_1 \to \varphi_2 = \neg\varphi_1 \vee \varphi_2,$$
$$\forall x.\, \varphi = \neg(\exists x.\, \neg\varphi),\ \forall X.\, \varphi = \neg(\exists X.\, \neg\varphi)$$

Logique

Sémantique (sur les mots)

On fixe un mot w sur l'alphabet Σ (modèle). Soit $n = |w|$.

- Les variables de Var_1 sont interprétées par des positions $k \in \{1, \ldots, n\}$ de w. Les variables de Var_2 sont interprétées par des ensembles de positions $K \subseteq \{1, \ldots, n\}$ de w.
- La formule $a(x)$ est vraie pour $x = k$ ssi $w_k = a$. La formule $y = x + 1$ ($x \leq y$, respectivement) est vraie pour $x = i, y = j$ ssi$j = i + 1$ (ssi $i \leq j$, respectivement). La formule $x \in X$ est vraie pour $x = k, X = K$ ssi $k \in K$.
- Les connecteurs booléens $\wedge, \neg$ ainsi que les quantificateurs $\exists, \forall$ ont l'interprétation usuelle (voir FOI).

Exemples et notation

$$\forall x \forall y.\, (y = x + 1 \rightarrow ((a(x) \wedge b(y)) \vee (b(x) \wedge a(y))$$

$$\exists X.(\forall x \forall y.\, (y = x + 1 \rightarrow (x \in X \leftrightarrow y \notin X)) \wedge 1 \in X \wedge \max \in X)$$

On note par $w \vdash \varphi$ le fait que $w \in \Sigma^*$ satisfait une formule φ (sans variable libre) et par $\mathcal{L}(\varphi) = \{w \in \Sigma^* \mid w \vdash \varphi\}$ le langage décrit par φ.

Logique

Théorème de Buechi

Pour tout NFA $\mathcal{A}$ il existe une formule MSOL $\varphi_{\mathcal{A}}$ t.q. $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\varphi_{\mathcal{A}})$, et réciproquement.

NFA $\rightarrow$ MSOL :

Soit $\mathcal{A} = \langle Q, \Sigma, \delta, I, F \rangle$ un NFA, $Q = \{1, \ldots, n\}$. On construit une formule MSOL

$$\varphi_{\mathcal{A}} = \exists X_1 \exists X_2 \ldots \exists X_n (\psi_1 \wedge \psi_2 \wedge p_3)$$

- ψ_1 dit que $(X_1, \ldots, X_n)$ est une partition de l'ensemble des positions : X_k correspond aux positions étiquetées par l'état k dans un calcul acceptant fixé (état après la transition)
- $\psi_2 = \forall x. \, (x \neq \max \rightarrow \bigvee_{(i,a,j) \in \delta} (x \in X_i \wedge x + 1 \in X_j \wedge a(x+1)))$ ("l'étiquetage par états représente un calcul")
- $\psi_3 = \bigvee_{i \in I, (i,a,k) \in \delta} (1 \in X_k \wedge a(1)) \wedge \bigvee_{j \in F} \max \in X_j$ ("le calcul est acceptant")

Propriétés de fermeture

Proposition

La classe Rec(Σ^*) est fermée par

- les opérations booléennes,
- le produit et l'itération,
- préfixe, suffixe et facteur,
- quotient gauche/droit : si L est reconnaissable et K quelconque, alors $K^{-1}L$ et LK^{-1} le sont aussi.
- morphisme, substitution rationnelle et morphisme inverse :
  - Soit $h : \Sigma^* \to \Gamma^*$ un morphisme. Si $L \subseteq \Sigma^*$ est reconnaissable, alors $h(L) = \{h(w) \mid w \in L\}$ l'est aussi. Si $K \subseteq \Gamma^*$ est reconnaissable, alors $h^{-1}(K) = \{w \in \Sigma^* \mid h(w) \in K\}$ l'est aussi.
  - Une substitution $f : \Sigma \to \mathcal{P}(\Gamma^*)$ associe à chaque lettre de Σ un langage $f(a) \subseteq \Gamma^*$. On l'étend en un morphisme $f : \Sigma^* \to \mathcal{P}(\Gamma^*)$ défini par $f(\epsilon) = \{\epsilon\}$, $f(ua) = f(u)f(a)$ ($u \in \Sigma^*$, $a \in \Sigma$). Pour $L \subseteq \Sigma^*$ on note $f(L) = \{f(w) \mid w \in L\}$. La substitution est rationnelle, si $f(a) \in$ Rec(Γ^*) pour tout $a \in \Sigma$.
    Si $L \subseteq \Sigma^*$ est reconnaissable, alors $f(L)$ l'est aussi.

Sans-étoile

Expressions sans-étoile (angl. star-free)

L'ensemble SF(Σ^*) des expressions sans-étoile sur Σ est défini par :

- $\emptyset$ et a ($a \in \Sigma$) appartiennent à SF(Σ^*)
- Si E et F sont dans SF(Σ^*), alors $(E + F)$, $(E \cdot F)$ et E^c aussi.

Exemples : $\Sigma^* = \emptyset^c$, $(ab)^*$, $\{w \in \{a, b\}^* \mid aab$ n'est pas facteur de $w\}$

FOL

La logique du premier ordre FOL (angl. first-order logic) est le fragment de MSOL qui n'utilise que des variables de premier ordre ($\text{Var}_2 = \emptyset$). On note par FOL(Σ^*) l'ensemble des langages décrits par des formules de FOL.

Théorème (McNaughton/Papert)

SF$(\Sigma^*) =$ FOL(Σ^*), pour tout alphabet Σ.