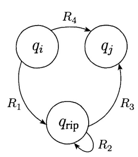
Master I d'informatique
INFO0809 : info. théorique
Pascal Mignot

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Aide-mémoire (DS du 13 février 2014)

Langages réguliers

- Un **alphabet** Σ est un ensemble **fini** de lettres. Σ^* est l'ensemble de toutes les chaines possibles pouvant être générées par l'alphabet Σ.
- Un **langage** L sur l'alphabet Σ est un sous-ensemble de Σ^* (i.e. $L \subseteq \Sigma^*$).
- Un automate déterministe fini (ADF) est un 5-uple $(Q, \Sigma, \delta, q_0, F)$, où Q est un ensemble fini d'états(ensemble des états), Σ est un ensemble fini de symboles (alphabet), $\delta : Q \times \Sigma \to Q$ est la fonction de transition, $q_0 \in Q$ est l'état de départ, $F \subset Q$ est l'ensemble des états acceptants.
- **Langage accepté par un ADF** M est l'ensemble des chaines L acceptées par M.
- **Langage régulier** est un langage accepté par un ADF M.
- **Opérations régulières** : les opérations régulières sont l'**Union** $A \cup B = \{x \mid x \in A \text{ ou } x \in B\}$, la **Concaténation** $A \circ B = \{xy \mid x \in A \text{ et } y \in B\}$, et l'**Etoile** $A^* = \{x_1 x_2 ... x_k \mid k \geq 0 \text{ et } x_i \in A \text{ pour tout } i\}$ où A et B sont deux langages.
- **Théorème** : L'ensemble des langages réguliers est fermé par toute opération régulière.
- **L'union de deux ADFs** $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ et $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ donne un ADF $M = (Q, \Sigma, \delta, q_0, F)$ avec pour états $Q = Q_1 \times Q_2$, comme fonction de transition $\forall (r_1, r_2) \in Q, a \in \Sigma, \delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$, comme état initial $q_0 = (q_1, q_2)$ et comme états finaux $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$.
- **Théorème** : l'ensemble des langages libres de contexte est fermé par les opérateurs d'intersection et de complémentarité (intersection : union avec $F = F_1 \times F_2$, complément : $\bar{F} = Q \backslash F$).
- Un automate non-déterministe fini (ANF) est similaire à un ADF, excepté que sa fonction de transition est définie comme $\delta : Q \times \Sigma_\epsilon \to \mathcal{P}(Q)$.
- **Théorème** : un langage est régulier si et seulement si il est reconnu par un ANF.
- Une **expression régulière** R (notée ER) sur un alphabet σ si R est un symbole $a \in \Sigma$, la chaîne vide ϵ, l'ensemble vide ϵ, une union d'expression régulière, une concaténation d'expression régulière, ou une opération étoile sur une expression régulières.
- **Théorème** : un langage est régulier si et seulement si il est reconnu par un ER.
- Transformation d'un ANF en ER :
  AGNF = ANF dont les transtions sont des ERs.
  1. partir d'un ANF modifié (AGNF) tel que ① l'état de départ a des transitions sortantes vers tous les autres états, ② l'état acceptant est unique.
  2. pour réduire l'état q_{rip}, réduire tous les cycles q_i, q_j, q_{rip} (voir ci-contre), en transition $R_{ij} = R_1 R_2{}^* R_4 \cup R_4$ entre q_i et q_j.
  3. répéter l'étape 2 jusqu'à ce qu'il n'y ait plus que 2 états. L'expression régulière est la transition entre les 2 états restants.



- **Théorème** : si un langage est fini ($\#L$ est fini), alors il est régulier.
- Deux mots u et v sont **L-équivalents** si $\forall z \in \Sigma^*, uz \in L \Leftrightarrow vz \in L$.
- **Théorème** (Myhill-Nerode) : un langage L est régulier **si et seulement si** il existe un nombre fini de L-classes distinctes.
- **Lemme de l'étoile** : soit un AFD $M = (Q, \Sigma, \delta, q_0, F)$ tel que $\mathcal{L}(M)$ soit infini. Pour tout mot w dans $\mathcal{L}(M)$ tel que $|w| \geq |Q|$, il existe une décomposition $w = xyz$ telle que $xy^i z \in \mathcal{L}(M)$ pour tout $i \geq 0$, $|y| > 0$ et $|xy| < |Q|$.

Grammaire libre de contexte

- Une **grammaire libre de contexte** (GLC) est un triplet (V, Σ, R, S) constitué d'un ensemble de variables V, d'un ensemble de symboles (terminaux) Σ, d'un ensemble de règle de réécriture sous la forme $u \to U$ où U est une combinaison de variables de V ou de symboles de Σ, S est la variable de départ.
- **Dérivations** : on écrit $u \Rightarrow v$ s'il existe une règle de réécriture $r \in R$ qui transforme u en v.
  on écrit $u \overset{*}{\Rightarrow} v$ s'il existe une suite de règles de réécriture $r_1 r_2 \dots r_k \in R^k$ qui transforme u en v.
- Un mot w est généré par une GLC si $S \overset{*}{\Rightarrow} w$.
- Un **langage libre de contexte** (LLC) est l'ensemble des mots de Σ^* générés par une GLC.
- L'**union de 2 GLCs** $G_1 = (V_1, \Sigma, R_1, S_1)$ et $G_2 = (V_2, \Sigma, R_2, S_2)$ (où V_1 et V_2 sont disjoints) est une GLC $G = (V, \Sigma, R, S)$ telle que $V = V_1 \cup V_2$ et $R = R_1 \cup R_2 \cup S \to S_1 | S_2$.

- Un grammaire linéaire (V, Σ, R, S) est une grammaire construite à partir d'un ADF $(Q, \Sigma, \delta, q_0, F)$, où l'on associe ① à chaque symbole $q_i \in Q$, une variable $v_i \in V$ ② à chaque transition $\delta(q_i, a) = q_j$, une règle $V_i \Rightarrow aV_j \in R$ ③ à chaque état final $q_i \in F$, une règle $V_i \rightarrow \epsilon$, ④ $S = V_0$ à l'état de départ q_0.
- L'ensemble des langages libres de contexte est l'ensemble des langages acceptés par les GLCs.
- **Théorème** : L'ensemble des langages libres de contexte est fermé par toute opération régulière.
- **Théorème** : Si un langage est régulier, alors il est accepté par une GLC.
- La **dérivation la plus à gauche** d'une chaîne consiste à chaque étape à remplacer la variable la plus à gauche.
- Une chaîne générée par une GLC est **ambigüe** si cette GLC permet de dériver cette chaine de plus d'une façon en n'utilisant que des dérivations les plus à gauche.
- Une grammaire est ambigüe si elle génère au moins une chaine ambigüe. Sinon, elle est non ambigüe.
- Une GLC est sous **forme normale de Chomsky** si chaque règle est sous la forme $A \rightarrow BC$ ou $A \rightarrow a$ ou $S \rightarrow \epsilon$ où ① a est n'importe quelle terminal, ② A, B, C, S sont des variables telles que S est la variable de départ, A est n'importe quelle variable (y compris S) et B,C sont des variables différentes de S.
- **Théorème** : tout LLC peut être généré par une GLC sous FNC en appliquant les étapes suivantes : ① ajouter une nouvelle règle $S_0 \rightarrow S$ ② supprimer les règles de la forme $A \rightarrow \epsilon$ (partout où A apparaît dans la RHS, ajouter une nouvelle règle sans A) ③ supprimer les règles unitaires $A \rightarrow B$ (pour toute règle $B \rightarrow U$, ajouter une règle $A \rightarrow U$) ④ supprimer les chaines $A \rightarrow u_1 u_2 \dots u_k$ (remplacer par les règles $A \rightarrow u_1 A_1$, $A_1 \rightarrow u_2 A_2$, $\dots A_{k-2} \rightarrow u_{k-1} u_k$ ⑤ supprimer les règles $A \rightarrow u_1 u_2$ où u_1 est un terminal (remplacer par $A \rightarrow U_1 u_2$ et $U_1 \rightarrow u_1$). Idem si u_2 est un terminal.
- Un **automate à pile** (AP) est un 6-uple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ où Q un ensemble fini d'états, Σ un alphabet, Γ les symboles de la pile, δ une fonction de transition $Q \times \Sigma' \times \Gamma' \rightarrow \mathcal{P}(Q \times \Gamma')$ avec $\Sigma' = \Sigma \cup \{\epsilon\}$ et $\Gamma' = \Gamma \cup \{\epsilon\}$, q_0 est l'état de départ, F est l'ensemble des états acceptants.
- **Théorème** : un langage est libre de contexte si et seulement si il est généré par un AP.
- **Lemme de l'étoile** (pour une GLC) : soit un langage L généré par une GLC. Pour tous les mots w de L de longueur au moins p (dépendante du langage L), alors on peut trouver u, v, x, y, z tel que $w = uvxyz$, $|vy| > 0$, $|vxy| \leq p$, et tel que $\forall i \geq 0, uv^i xy^i z \in L$.
- **Théorème** : l'ensemble des langages libres de contexte n'est pas fermé par les opérateurs d'intersection et de complémentarité.

Machine de Turing

- Une **machine de Turing** (MT) est un 7-uple $(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ où Q est l'ensemble fini des états, Σ est l'alphabet d'entrée, Γ est l'alphabet de la bande, $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ est la fonction de transition, $q_0 \in Q$ est l'état de départ, $q_a \in Q$ est l'état acceptant, $q_r \in Q$ est l'état rejetant.
- L'**exécution d'une MT** commence avec un mot w écrit sur sa bande et dans l'état q_0. La MT accepte (resp. rejette) w si elle atteint l'état q_a (resp. q_s), sinon elle ne s'arrête jamais.
- Une MT est dans l'état uq_iv si la bande contient la chaîne uv et le pointeur de lecture placé sur le premier caractères de la chaîne u.
- Un langage L est accepté par une MT M si pour tout $w \in L$, M s'arrête sur un état acceptant.
- Un langage est (récursivement) **énumérable** s'il existe une MT qui l'accepte.
- Une MT décide un langage L si MT accepte w pour tout $w \in L$ et rejette w pour tout $w \notin L$.
- **Théorème** : tous les modèles de MT (transition S, multibandes, ...) sont équivalents.
- **Complétude de Turing** : un formalisme de machine est Turing-complet s'il permet de simuler une MT.
- une MT est dite **non déterministe** (MTND) s'il existe plus d'une transition possible à partir d'un même état et symbole d'entrée.
- L'**exécution d'une MTND** conduit alors à l'évaluation de toutes les transitions possibles en même temps. La MNTD accepte l'entrée si au moins une branche l'accepte, rejette l'entrée si toutes les branches la rejettent ou boucle à l'infini.
- **Théorème** : toute MTND a un MT équivalente.
- **classes de langages** : $\mathcal{RE}$ = classe des langages énumérables.
  $co\mathcal{RE}$ = classe des langages dont le complément est énumérable (=co-énumérable).
  $\mathcal{R}$ = classe des langages décidables.
- **Théorème** : si un langage L est décidable, alors $\bar{L}$ est énumérable.
- **Théorème** : $\mathcal{R} = \mathcal{RE} \cap co\mathcal{RE}$ (un langage est décidable si et seulement si il est énumérable et co-énumérable).
- un énumérateur est une MT avec une bande de travail qui envoie sur sa sortie (éventuellement avec répétition) l'ensemble des mots reconnus par un langage.
- **Thèse de Church-Turing :** tout calcul informatique est équivalent à un algorithme s'exécutant sur une MT.
- **Encodage :** on note $< O >$ l'encodage de l'objet O sur la bande d'entrée d'une MT.
- une **MT universelle** est une MT qui peut simuler une MT M arbitraire sur une entrée arbitraire w.