Langages formels

Montaigne 2023-2024

– mpi23@arrtes.net –

Introduction

Quels sont les problèmes susceptibles de recevoir une traitement automatique ?

ou de manière équivalente

Quel est l'objet de l'informatique ?

- D'un point de vue pratique, cette question équivaut à l'écriture d'un **programme** capable de résoudre un **problème** par un traitement automatique.
- Mais avant d'écrire un tel programme, l'existence même d'un traitement automatique du **problème** est-elle garantie ?

- Pour répondre à ces questions, il convient d'abord de définir **rigoureusement** les deux concepts centraux de **problème** et de **programme**.
- Cette définition rigoureuse exige la construction d'**outils théoriques**.

Qu'est-ce qu'un problème ?

On distingue plusieurs **familles de problèmes** :

- les problèmes d'**évaluation** ;
- les problèmes d'**optimisation** ;
- les problèmes d'**approximation** ;
- les problèmes d'**énumération** ;
- les problèmes de **comptage** ;
- les problèmes de **décision**.

Qu'est-ce qu'un problème ?

Les familles les plus importantes sont celles :

- des problèmes d'**évaluation**, pour lesquels on cherche à calculer la valeur d'une fonction sur différentes entrées ;
- des problèmes de **décision**, où l'on veut déterminer si les entrées vérifient une propriété donnée.

Dans un contexte de formalisation, on montre qu'on ne perd pas de généralité à se restreindre aux **problèmes de décision**. En définissant rigoureusement cette famille de problèmes, la formalisation des autres familles de problèmes est immédiate.

Qu'est-ce qu'un problème ?

Ainsi, **tout problème est de la forme suivante**.

Entrée :	$x \in I$
Question :	x satisfait-il la propriété P ?

où I est un **ensemble** dont les éléments sont appelés **instances** (ou **entrées**) du problème et P est une **propriété** des éléments de I.

Nature des objets manipulés

- La définition du concept de problème n'est complète que si on précise la **nature des objets manipulés**.
- Un programme informatique ne manipule pas des **objets abstraits** mais seulement des **représentations de ces objets**.
- Et la manière dont le programme opère dépend étroitement de cette représentation !

Nature des objets manipulés

- Les **instances** d'un programme sont donc les **codages d'objets abstraits** et non les objets eux-mêmes.
- Un **codage** est une succession de symboles qui forme ce qu'on appelle un **mot**.
- L'**ensemble des symboles** qui permettent la définition d'un mot est appelé un **alphabet**.
- Enfin, l'**ensemble des mots** sur un alphabet donné forme un **langage formel**.

Finalement, qu'est-ce qu'un problème ?

En informatique, résoudre un problème, c'est reconnaître un langage.

Entrée :	un mot x
Question :	x appartient-il au langage L ?

Il convient à présent de définir rigoureusement le concept de **langage formel**.

Qu'est-ce qu'un algorithme ?

- Les termes **algorithmes** et **programmes** se réfèrent à l'idée d'une succession d'opérations atomiques, selon un ordre qui dépend des objets pris en entrée.
- La mise en œuvre d'un algorithme dépend de la représentation des instances.
- De fait, la définition même du mot algorithme mène à la notion de **modèle de calcul**.

Qu'est-ce qu'un algorithme ?

- Un **modèle de calcul** est un objet mathématique, formé d'ensembles, de fonctions, de relations, qui abstrait les propriétés en définissant une notion d'**opération atomique** et en précisant scrupuleusement comment ces opérations se succèdent sur une entrée donnée pour former un calcul.
- Les **automates** forment un premier exemple, simple, de **modèle de calcul**. Ils feront l'objet de prochains chapitres.

Alphabets et mots

Alphabet

Définition 1 (alphabet)

Un **alphabet** Σ est un ensemble fini non vide de symboles a_i, $i = 1, \dots, k$, appelés lettres ou caractères.

$$\Sigma = \{a_1, a_2, \dots, a_k\}$$

Exemples

- $\Sigma = \{a, b, c, \dots, z\}$
- $\Sigma = \{a, b, c, \dots, z, let, rec, for, =, +, *, \dots\}$
- $\Sigma = \{00, 01, 10, 11\}$

Mot

Définition 2 (mot)

Un **mot** m est une suite finie de symboles sur un alphabet Σ.

- Soit $\Sigma = \{a_1, a_2, \ldots, a_k\}$ un alphabet.
  La suite $(a_{i_1}, a_{i_2}, \ldots, a_{i_n}) \in \Sigma^n$ est un **mot**.
- Traditionnellement, le mot est écrit sans les parenthèses et les virgules.

$$m = a_{i_1} a_{i_2} \ldots a_{i_n}$$

Exemple

Un mot formé sur l'alphabet $\Sigma = \{a, b, c\}$ est :

$$a, b, c, ab, ac, bc, aab, abc, acc, acb, aabbcabcab, \ldots$$

Longueur d'un mot

- Les symboles de l'alphabet Σ sont des mots de longueur 1.
- La **longueur d'un mot** $m = a_{i_1} a_{i_2} \dots a_{i_n}$ est : $|m| = n$.
- Le **mot vide** noté ε est l'unique mot de longueur 0.
- Si $a \in \Sigma$, $|m|_a$ désigne le nombre d'occurences du symbole a dans le mot m.
- De manière évidente, on a :

$$|mm'| = |m| + |m'| \qquad |mm'|_a = |m|_a + |m'|_a$$

- Si $\Sigma = \{a, b, c\}$ et $m = aaababba$, on a $|m| = 8$, $|m|_a = 5$, $|m|_b = 3$, $|m|_c = 0$.

Ensemble de mots

Soit Σ un alphabet.

- L'ensemble des **mots de longueur n sur Σ** est noté Σ^n.
- L'ensemble de **tous les mots sur Σ** est noté Σ^+ avec :

$$\Sigma^+ = \bigcup_{i=1}^{+\infty} \Sigma^i$$

- L'**ensemble étoilé Σ^*** est l'ensemble Σ^+ complété par le mot vide.

$$\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$$

Concaténation

Définition 3 (concaténation)

Soit Σ un alphabet. La **concaténation** est une loi de composition interne, notée $\cdot$, sur Σ^*.
Si $m = a_{i_1} a_{i_2} \dots a_{i_p}$ et $m' = a'_{i_1} a'_{i_2} \dots a'_{i_q}$ sont des mots construits sur Σ, la concaténation $m \cdot m'$ (ou mm') de m et m' est définie par :

$$m \cdot m' = a_{i_1} a_{i_2} \dots a_{i_p} a'_{i_1} a'_{i_2} \dots a'_{i_q}$$

- On adopte les notations suivantes.

$$a^n = \underbrace{aa \dots a}_{n \text{ fois}} \qquad a^m b^n = \underbrace{aa \dots a}_{m \text{ fois}} \underbrace{bb \dots b}_{n \text{ fois}} \qquad (ab)^n = \underbrace{abab \dots ab}_{n \text{ fois } ab}$$

- Par convention, pour tout élément a de Σ, on pose : $a^0 = \varepsilon$.
- Cette loi est associative et admet ε comme élément neutre. Σ^* muni de cette loi est appelé **monoïde** engendré par Σ.

Facteurs

Définition 4 (préfixe)

$p \in \Sigma^*$ est un **préfixe** de $m \in \Sigma^*$ si et seulement si il existe $v \in \Sigma^*$ vérifiant $m = pv$. On note parfois $p \sqsubseteq m$.
Si $p \neq \varepsilon$, p est appelé **préfixe propre** de m. On note $p \sqsubset m$.

Définition 5 (sufffixe)

$s \in \Sigma^*$ est un **suffixe** de $m \in \Sigma^*$ si et seulement si il existe $u \in \Sigma^*$ vérifiant $m = us$. On note $s \sqsupseteq m$.
Si $s \neq \varepsilon$, s est appelé **suffixe propre** de m. On note $s \sqsupset m$.

Définition 6 (facteur)

$f \in \Sigma^*$ est un **facteur** de $m \in \Sigma^*$ si et seulement si il existe $u \in \Sigma^*$ et $v \in \Sigma^*$ vérifiant $m = ufv$.
Si u et v ne sont pas simultanément égaux à ε, f est appelé **facteur propre** de m.

Facteurs

Exemple

Soit $\Sigma = \{a, b, c\}$ et $m = aaababba$.

- $aaab$ est un préfixe propre ainsi qu'un facteur propre de $m = aaababba$.
- aba est un facteur propre de $m = aaababba$ qui n'est ni un préfixe, ni un suffixe.
- Les préfixes de m sont :

$$\varepsilon, a, aa, aaa, aaab, aaaba, aaabab, aaababb, aaababba$$

- Les suffixes de m sont :

$$\varepsilon, a, ba, bba, abba, babba, ababba, aababba, aaababba$$

Langages formels

Langage

Définition 7 (langage)

Un **langage** sur un alphabet Σ (ou langage de Σ^*) est un ensemble de mots de Σ^*.

- Un langage sur Σ est un élément de $\mathcal{P}(\Sigma^*)$.
- Le langage qui ne contient aucun mot est appelé langage vide, noté $\varnothing$.
- Le langage vide $\varnothing$ et Σ^* sont des langages sur Σ.
- Ne pas confondre $\{\varepsilon\}$ et $\varnothing$. Le premier est un ensemble qui contient le mot vide. Le second ne contient aucun mot.

Langage

Exemples

Soit $\Sigma = \{a, b\}$ un alphabet. Les ensembles suivants sont des langages sur Σ.

- $L_1 = \{a^m b^n \mid (m, n) \in \mathbb{N}^2\}$ est l'ensemble des mots de Σ^* de la forme $a \dots ab \dots b$.
- $L_2 = \Sigma^n$ est l'ensemble des mots de Σ^* de longueur n.
- $L_3 = \{m \in \Sigma^* \mid |m|_a = |m|_b\}$ est l'ensemble des mots de Σ^* qui comportent autant de a que de b.
- $L_4 = \{m \in \Sigma^* \mid |m|_a = 1 \mod 2\}$ est l'ensemble des mots de Σ^* qui contiennent un nombre impair de a.

Opérations sur les langages

Définition 8 (Union de langages)

L'**union** (ou **somme**) des langages L et L', notée $L \cup L'$ ou $L + L'$ est définie par :

$$L \cup L' = \{m \mid m \in L \text{ ou } m \in L'\}$$

Exemples

Soit l'alphabet $\Sigma = \{0, 1\}$.
Soient les deux langages définis sur Σ :

$$L = \{\varepsilon, 0, 1, 10, 11\} \qquad L' = \{\varepsilon, 1, 0110, 11010\}$$

Alors :

$$L \cup L' = \{\varepsilon, 0, 1, 10, 11, 0110, 11010\}$$

Opérations sur les langages

Définition 9 (intersection)

L'**intersection** des langages L et L', notée $L \cap L'$, est définie par :

$$L \cap L' = \{m \mid m \in L \text{ et } m \in L'\}$$

Exemples

Soit l'alphabet $\Sigma = \{0, 1\}$.
Soient les deux langages définis sur Σ :

$$L = \{\varepsilon, 0, 1, 10, 11\} \qquad L' = \{\varepsilon, 1, 0110, 11010\}$$

Alors :

$$L \cap L' = \{\varepsilon, 1\}$$

Opérations sur les langages

Définition 10 (concaténation)

La **concaténation** (ou **produit**) des langages L et L', notée $L \cdot L'$ ou LL', est définie par :

$$LL' = \{mm' \mid m \in L \text{ et } m' \in L'\}$$

Exemples

Avec les langages $L = \{act\}$ et $L' = \{\varepsilon, eur, rice\}$, on a :

$$LL' = \{act, acteur, actrice\} \qquad L'L = \{act, euract, riceact\}$$

Noter que les langages L et L' peuvent être définis sur des alphabets différents. Si L est un langage sur un alphabet Σ, L' un langage sur un alphabet Σ', $L \cup L'$ est un langage sur $\Sigma \cup \Sigma'$.

Opérations sur les langages

Pour tout entier naturel n, l'**exponentation** d'un langage L est définie comme suit.

$$L^n = \begin{cases} L^0 = \{\varepsilon\} & \text{si } n = 0 \\ L^{n+1} = L \cdot L^n = L^n \cdot L & \text{si } n > 0 \end{cases}$$

Exemples

Si $L = \{ab\}$ est un langage sur l'alphabet $\Sigma = \{a, b\}$, alors :

$$\begin{aligned} L^0 &= \{\varepsilon\} \\ L^1 &= \{ab\} \\ L^2 &= L \cdot L^1 = \{abab\} \\ L^3 &= L \cdot L^2 = \{ababab\} \end{aligned}$$

Opérations sur les langages

Définition 11 (fermeture étoilée)

La **fermeture étoilée** (ou itération) du langage L, noté L^*, est le langage :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

Exemples

Si $L = \{a\}$ est un langage, alors :

$$L^* = \{\varepsilon, a, a^2, a^3, a^4, \dots\}$$

Opérations sur les langages

On note :

$$L^+ = \bigcup_{k \in \mathbb{N}^*} L^k$$

de sorte que :

$$L^* = L^+ \cup \{\varepsilon\} \qquad L^+ = LL^* = L^*L$$

Exemples

Si $L = \{a\}$ est un langage, alors :

$$L^+ = \{a, a^2, a^3, a^4, \dots\}$$

Opérations sur les langages

On a les propriétés suivantes.

- Si $L = \{a\}$, par **abus de notation**, L^* est noté a^*. De même, si $L = \{a\}$ et $L' = \{b\}$, $L^* L'^*$ est noté $a^* b^*$.
- De la définition, il vient également :

$$\begin{aligned} L = \{\varepsilon\} &\implies L^* = \{\varepsilon\} \\ L = \varnothing &\implies L^* = \{\varepsilon\} \end{aligned}$$

- Enfin, on établit que :

$$(L^*)^* = L^*$$

Ce résultat indique qu'aucun mot n'est ajouté en « étoilant » L^*. Cette propriété éclaire le choix du vocable **fermeture** étoilée pour désigner l'opération.