Introduction à la calculabilité

Pierre Wolper

Email: pw@montefiore.ulg.ac.be

URL: http: //www.montefiore.ulg.ac.be/~pw/
http: //www.montefiore.ulg.ac.be/
~pw/cours/calc.html

Référence

Pierre Wolper, *Introduction à la calculabilité - 2ième édition*, Dunod, 2001.

Chapitre 1
Introduction

1.1 Motivation

- Comprendre les limites de l'informatique.
- Distinguer problèmes solubles et insolubles par des algorithmes.
- Obtenir des résultats indépendants de la technologie employée pour construire les ordinateurs.

1.2 Problèmes et Langages

- Quels problèmes sont solubles par un programme exécuté sur un ordinateur ?

  Il faut préciser :

  – la notion de problème,

  – la notion de programme exécuté sur un ordinateur.

La notion de problème

Problème : question *générique*.

Exemples :

- trier un tableau de nombres ;
- déterminer si un programme écrit en C s'arrête quelles que soient les valeurs des données qui lui sont fournies (problème de l'arrêt) ;
- déterminer si une équation à coefficients entiers a des solutions entières (10ième problème de Hilbert).

La notion de programme

Procédure effective : programme pouvant être exécuté sur un ordinateur.

Exemples :

- Procédure effective : programme écrit en JAVA ;

- Procédure non effective : “pour résoudre le problème de l’arrêt, il faut déterminer si le programme n’a pas de boucles ou de séquences d’appels récursifs infinies.”

Problème de l'arrêt

```
recursive function threen (n: integer):integer;
begin
if (n = 1) then 1
           else if even(n) then threen(n ÷ 2)
                           else threen(3 × n + 1);
end;
```

1.3 La formalisation des problèmes

Comment représente-t-on les instances de problèmes ?

Alphabets et mots

Alphabet : ensemble fini de symboles.

Exemples

- $\{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$
- $\{\alpha, \beta, \gamma\}$
- $\{1, 2, 3\}$
- $\{\clubsuit, \diamondsuit, \heartsuit\}$

Mot sur un alphabet : séquence *finie* d'éléments de cet alphabet.

Exemples

- *a, abs, zt, bbbssnbnzzyyyyyddtrra, grosseguindaille* sont des mots sur l'alphabet $\{a, \ldots, z\}$.

- $4\clubsuit 3\diamondsuit 5\heartsuit 2\spadesuit, 12765, \clubsuit\heartsuit$ sont des mots sur l'alphabet $\{0, \ldots, 8, \clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$.

Mot vide : désigné par e, ε ou encore λ.

Longueur du mot w : $|w|$

$w = aaabbaaaabb$
$w(1) = a$, $w(2) = a$,. . . , $w(11) = b$

Représentation des problèmes

Encodage d'un problème

Considérons un problème binaire dont les instances sont encodées par des mots définis sur un alphabet Σ. L'ensemble de tous les mots définis sur Σ peut être partitionné en 3 sous-ensembles :

- *instances positives : réponse* oui (*positive instances*) ;
- *instances négatives : réponse* non (*negative instances*) ;
- mots ne représentant pas des instances du problème.

Ou encore :

- les mots représentant des instances du problème pour lesquelles la réponse est *oui*, *instances positives* ;

- les mots ne représentant pas des instances du problème ou représentant des instances du problème pour lesquelles la réponse est *non*, *instances négatives*.

Langages

Langage (*language*) : ensemble de mots définis sur le même alphabet.

Exemples

- $\{aab, aaaa, \varepsilon, a, b, abababababbbbbbbbbbb\}$, $\{\varepsilon, aaaaaaa, a, bbbbbb\}$ et $\emptyset$ (l'ensemble vide) : langages sur l'alphabet $\{a, b\}$.
- pour l'alphabet $\{0, 1\}$,

  $\{0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100,$
  $101, 110, 111, \ldots\}$: langage contenant tous les mots.
- langage $\emptyset \neq$ langage $\{\varepsilon\}$.
- ensemble des mots représentant les programmes C qui s'arrêtent toujours.

1.4 La description des langages

Opérations sur les langages

Soit deux langages L_1 et L_2.

- $L_1 \cup L_2 = \{w | w \in L_1 \text{ ou } w \in L_2\}$;
- $L_1 \cdot L_2 = \{w | w = xy, x \in L_1 \text{ et } y \in L_2\}$;
- $L_1^* = \{w | \exists k \geq 0 \text{ et } w_1, \ldots, w_k \in L_1 \text{ tels que } w = w_1 w_2 \ldots w_k\}$;
- $\overline{L_1} = \{w | w \notin L_1\}$.

Langages réguliers

$\mathcal{R}$ (ensemble des langages réguliers sur un alphabet Σ) est le plus petit ensemble de langages tel que :

1. $\emptyset \in \mathcal{R}$ et $\{\varepsilon\} \in \mathcal{R}$,

2. $\{a\} \in \mathcal{R}$ pour tout $a \in \Sigma$ et

3. si $A, B \in \mathcal{R}$, alors $A \cup B$, $A \cdot B$ et $A^* \in \mathcal{R}$.

Les expressions régulières

Notation pour représenter les langages réguliers.

1. $\emptyset$, ε et les éléments de Σ sont des expressions régulières.

2. Si α et β sont des expressions régulières, alors $(\alpha\beta)$, $(\alpha \cup \beta)$, $(\alpha)*$ sont des expressions régulières.

Les expressions régulières constituent un langage sur l'alphabet $\Sigma' = \Sigma \cup \{), (, \emptyset, \cup, *, \varepsilon\}$.

Langage dénoté par une expression régulière

1. $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$,

2. $L(a) = \{a\}$ pour tout $a \in \Sigma$,

3. $L((\alpha \cup \beta)) = L(\alpha) \cup L(\beta)$,

4. $L((\alpha\beta)) = L(\alpha) \cdot L(\beta)$,

5. $L((\alpha)*) = L(\alpha)^*$.

Théorème

Un langage est régulier

si et seulement si

il est dénoté par une expression régulière.

Langages réguliers : exemples

- L'ensemble de tous les mots sur $\Sigma = \{a_1, \ldots, a_n\}$ est dénoté par $(a_1 \cup \ldots \cup a_n)^*$ (ou encore Σ^*).

- L'ensemble de tous les mots non vides sur $\Sigma = \{a_1, \ldots, a_n\}$ est dénoté par $(a_1 \cup \ldots \cup a_n)(a_1 \cup \ldots \cup a_n)^*$ (ou encore $\Sigma\Sigma^*$, ou Σ^+).

- l'expression $(a \cup b)^* a (a \cup b)^*$ dénote le langage des mots composés de "a" et "b" qui contiennent au moins un "a".

Langages réguliers : exemples (suite)

$$(a^*b)^* \cup (b^*a)^* = (a \cup b)^*$$

Démonstration

- $(a^*b)^* \cup (b^*a)^* \subset (a \cup b)^*$ car $(a \cup b)^*$ dénote l'ensemble de tous les mots composés des caractères "a" et "b".

- considérons un mot arbitraire

  $$w = w_1 w_2 \dots w_n \in (a \cup b)^*.$$

  On peut distinguer les 4 cas suivants...

1. $w = a^n$ et donc $w \subset (\varepsilon a)^* \subset (b^*a)^*$;
2. $w = b^n$ et donc $w \subset (\varepsilon b)^* \subset (a^*b)^*$;
3. w contient des a et des b et se termine par b

$$w = \underbrace{a \ldots ab}_{a^*b} \underbrace{\ldots b}_{(a^*b)^*} \underbrace{a \ldots ab}_{a^*b} \underbrace{\ldots b}_{(a^*b)^*}$$

   $\Rightarrow w \in (a^*b)^* \cup (b^*a)^*$;
4. w contient des a et des b et se termine par $a \Rightarrow$ décomposition similaire à celle du cas 3.

1.5 Les langages non réguliers

Fait

Il n'y a pas assez d'expressions régulières pour représenter tous les langages !

Définition

Cardinalité d'un ensemble...

Exemple

Les ensembles $\{0, 1, 2, 3\}$, $\{a, b, c, d\}$, $\{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$ ont tous la même taille.
Ils peuvent être mis en bijection, par exemple $\{(0, \clubsuit), (1, \diamondsuit), (2, \heartsuit), (3, \spadesuit)\}$.

Les ensembles dénombrables

Définition

Un ensemble infini est *dénombrable* (*denumerable*) si il existe une bijection entre cet ensemble et l'ensemble des nombres naturels.

Remarque

Au sens courant de "dénombrable", tout ensemble fini est aussi dénombrable.

Ensembles dénombrables : exemples

1. L'ensemble des nombres pairs est dénombrable :

$$\{(0,0),(2,1),(4,2),(6,3),\ldots\}.$$

2. L'ensemble des mots sur l'alphabet $\{a,b\}$ est dénombrable :
   $\{(\varepsilon,0),(a,1),(b,2),(aa,3),(ab,4),(ba,5),$
   $(bb,6),(aaa,7)\ldots\}$.

3. Les nombres rationnels sont dénombrables :
   $\{(0/1,0),(1/1,1),(1/2,2),(2/1,3),(1/3,4),$
   $(3/1,5),\ldots\}$.

4. Les expressions régulières sont dénombrables.

La technique de la diagonale

Théorème

L'ensemble des sous-ensembles d'un ensemble dénombrable n'est pas dénombrable.

Démonstration

	a_0	a_1	a_2	a_3	a_4	$\cdots$
s_0	$\times$	$\times$		$\times$		
s_1	$\times$	$\square$		$\times$		
s_2		$\times$	$\times$		$\times$	
s_3	$\times$		$\times$	$\square$		
s_4		$\times$		$\times$	$\square$	
$\vdots$						

$$D = \{a_i \mid a_i \notin s_i\}$$

Conclusion

- L'ensemble des langages n'est pas dénombrable.
- L'ensemble des langages réguliers est dénombrable.
- Il y a donc (beaucoup) plus de langages que de langages réguliers.

1.6 Un aperçu de la suite...

- Notion de procédure effective (automates).
- Problèmes non solubles algorithmiquement.
- Problèmes non solubles efficacement.