TD N°01

NOTION DE SYSTÈMES FORMELS

Introduction

Exercice n°01

Soit le système formel **SF = <A, γ, Ax, R>** :
A = {E, F, _}
γ : Les mots sont toutes les chaines de symboles ;
Ax = {xE_Ex} tel que x est une suite de '_'
R = {xEyFz → xzEy_Fz} tel que x,y,z sont des suites de '_'

Donnons les sens suivant aux symboles :
E : « divisé par »
F : « égale »
'_' : Représente l'unité (exp : _ _ _ _ veut dire le nombre 4)

Supposons que le mot aEbFc est un énoncé vrai. (veut dire « a divisé par b égale c »)
Soit les formules suivante :

a) accEb_ _Fc
b) aaE_ _EbFc
c) abEbFc_

1. La seule règle de R est-elle une règle de production ou de réécriture ?
2. Est-ce la suite de mots w_1w_2 est une preuve ? tel que :
   w_1=aEbFc
   w_2=acEb_Ec
3. Parmi les formules ci-dessus, lesquelles sont des théorèmes ?

Récursion et induction

Exercice n°02

La numérotation romaine basée sur le tableau suivant est normalisée comme suit :

Chiffre romain	Entier
I	1
V	5
X	10

L	50
C	100
D	500
M	1000

- Toute lettre 'b' placée à la droite d'une autre 'a' avec valeur(a) est **supérieure** ou **égale** à valeur(b) alors la valeur de 'b' **s'ajoute** à celle de 'a'.
- Toute lettre 'a' placée à la gauche d'une autre 'b' avec valeur(a) est **inférieur** à valeur(b) alors la valeur de 'a' se **soustraie** de celle de 'b'.
- La même lettre ne peut pas être employée **quatre fois** consécutivement sauf M.
- Lettres d'unités : I est une unité pour V et X, X est une unité pour L et C, C est une unité pour D et M.
  Exp : 49 = XLIX et non IL.
  On peut écrire MMMM mais au lieu d'écrire XXXX on écrit XL.
  1975 = MCMLXXV

1. Donnez un système formel qui formalise le problème de conversion de nombres romains en nombres entiers.
2. Ecrivez un algorithme récursif qui permet de convertir un nombre romain en un nombre entier.

Exercice n°03

Démontrer par récurrence que :
Pour tout $n \geq 1$, $1^3 + 2^3 + \ldots + n^3 = (n^2 (n + 1)^2) / 4 = (1 + 2 + \ldots + n)^2$.

Correction du TD n°01

Notion de systèmes formels

Exercice n°01

Soit le système formel **SF = <A, γ, Ax, R>** :
A = {E, F, _}
γ : Les mots sont toutes les chaines de symboles ;
Ax = {xE_Ex} tel que x est une suite de '_'
R = {xEyFz → xzEy_Fz} tel que x,y,z sont des suites de '_'

Donnons les sens suivant aux symboles :
E : « divisé par »
F : « égale »
'_' : Représente l'unité (exp : _ _ _ _ veut dire le nombre 4)

Supposons que le mot aEbFc est un énoncé vrai. (veut dire « a divisé par b égale c »)
Soit les formules suivante :

a) accEb_ _Fc
b) aaE_ _EbFc
c) abEbFc_

1. La seule règle de R est-elle une règle de production ou de réécriture ?

Correction :

La seule règle de R est une règle de production parce qu'elle produit un nouveau mot à partir d'un mot initial.

2. Est-ce la suite de mots w_1w_2 est une preuve ? tel que :
   w_1=aEbFc
   w_2=acEb_Ec

Correction :

La suite w_1w_2 n'est pas une preuve parce qu'on appliquant la règle de R à w_1 on n'arrive pas à w_2 :
aEbFc → acEb_Fc (tel que la variable x prend la valeur a, y prend la valeur b et z pend la valeur c)

3. Parmi les formules ci-dessus, lesquelles sont des théorèmes ?

La formule (a) est un théorème :

$aEbFc \rightarrow acEb\_Fc \rightarrow a) \rightarrow accEb\_ \_Fc$

La formule (b) n'est pas un théorème parce que la règle ne permet d'avoir un $2^{ème}$ E.

La formule (c) n'est pas un théorème parce que la règle ne permet d'avoir '_' après le 'c'.

Exercice n°02

La numérotation romaine basée sur le tableau suivant est normalisée comme suit :

Chiffre romain	Entier
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

- Toute lettre 'b' placée à la droite d'une autre 'a' avec valeur(a) est **supérieure** ou **égale** à valeur(b) alors la valeur de 'b' **s'ajoute** à celle de 'a'.
- Toute lettre 'a' placée à la gauche d'une autre 'b' avec valeur(a) est **inférieur** à valeur(b) alors la valeur de 'a' se **soustraie** de celle de 'b'.
- La même lettre ne peut pas être employée **quatre fois** consécutivement sauf M.
- Lettres d'unités : I est une unité pour V et X, X est une unité pour L et C, C est une unité pour D et M.
  Exp : 49 = XLIX et non IL.
  On peut écrire MMMM mais au lieu d'écrire XXXX on écrit XL.
  1975 = MCMLXXV

1. Donnez un système formel qui formalise le problème de conversion de nombres romains en nombres entiers.

Correction :

Soit le système formel formalisant le problème de conversion de nombre romain suivant:
SFRomain<A, γ, Ax, R>, avec:
Les symboles: A = {I, V, X, L, C, D, M, 1, 5, 10, 50, 100, 500, 1000}
Les mots: Toutes les chaines de symboles ;
L'axiome: Ax = {S} tel que S est une chaine de symboles de l'ensemble {I, V, X, L, C, D, M}.
Les règles: R = {

$I \rightarrow 1$;
$V \rightarrow 5$;
$X \rightarrow 10$;
$L \rightarrow 50$;
$C \rightarrow 100$;
$D \rightarrow 500$;

$M \rightarrow 1000$;
$S_1S_2 \rightarrow S_1 - S_2$; // Si valeur de S_1 > valeur de S_2
$S_1S_2 \rightarrow S_1 + S_2$; // Si valeur de $S_1 \leq$ valeur de S_2
} tel que S_1 et S_2 sont des symboles de l'ensemble {I, V, X, L, C, D, M}.

2. Ecrivez un algorithme récursif qui permet de convertir un nombre romain en un nombre entier.

Correction :

La fonction récursive pour ce problème :

```
Fonction decode(c) {
     si c = 'M' alors retourner 1000;
     si c = 'D' alors retourner 500;
     si c = 'C' alors retourner 100;
     si c = 'L' alors retourner 50;
     si c = 'X' alors retourner 10;
     si c = 'V' alors retourner 5;
     si c = 'I' alors retourner 1;
}

Fonction convertir(nbromain) {
     si taille(nbromain)=1 alors retourner decode(nbromain)
     sinon
          si decode(nbromain[0]) < decode(nbromain[1]) alors
               retourner convertir(nbromain[1 .. fin]) -
decode(nbromain[0])
          sinon
               retourner convertir(nbromain[1 .. fin]) +
decode(nbromain[0])
}
```

Exercice n°03

Démontrer par récurrence que :
Pour tout $n \geq 1$, $1^3 + 2^3 + \ldots + n^3 = (n^2 (n + 1)^2) / 4 = (1 + 2 + \ldots + n)^2$.

Correction :

On appel $P(n) = 1^3 + 2^3 + \ldots + n^3 = (n^2 (n + 1)^2) / 4$
Pour n = 1 on a $P(1) = 1 = (1^2 (1 + 1)^2) / 4$ la formule est vrai.
Supposant que P(n) est vrai et vérifiant que P(n+1) est vrai.
P(n) vrai $\Rightarrow 1^3 + 2^3 + \ldots + n^3 = (n^2 (n + 1)^2) / 4$

$$\Rightarrow 1^3 + 2^3 + \ldots + n^3 + (n+1)^3 = ((n^2 (n + 1)^2) / 4) + (n+1)^3$$
$$= (n^2 (n + 1)^2 + 4(n+1)^3) / 4$$
$$= (n + 1)^2 (n^2 + 4(n+1)) / 4$$
$$= (n + 1)^2 (n + 2)^2 / 4$$

$\Rightarrow$ P(n+1) est vrai aussi.
On conclu que pour tout $n \geq 1$, P(n) est vrai.