Calcul formel (avec Maple)

François Bergeron, UQAM

30 août 2014

Introduction

Les systèmes de calcul formel permettent de manipuler concrètement des objets mathématiques abstraits de façon rigoureuse. Cela va des nombres entiers, rationnels, réels ou complexes (et des calculs sur ceux-ci) ; à des manipulations d'opérateurs sur des espaces de fonctions ; en passant par un vaste spectre de notions mathématiques de l'algèbre, du calcul, de l'analyse complexe, de la théorie des nombres, etc.

Dans les systèmes de calcul formel, une session de travail est habituellement un processus interactif qui consiste à donner au système une instruction de calcul (apparaissant en rouge dans ce qui suit), et on obtient alors comme résultat la valeur du calcul demandé (apparaissant en bleu). Dans notre cas, le système affiche automatiquement le symbole « > » chaque fois qu'il est prêt à effectuer une prochaine instruction. Le « ; » signifie la fin de l'instruction donnée, et la touche « return » (ou « enter ») déclenche le calcul. Ainsi, on peut demander de calculer

> $\mathbf{gcd}(x^{36}-1, x^{24}-1)$;

$$x^{12}-1$$

Ici, la fonction Maple « gcd » trouve que $x^{12}-1$ est un [1] plus grand commun diviseur des polynômes $x^{36}-1$ et $x^{24}-1$. En plus d'effectuer des calculs explicites, il est possible de donner des noms à certains objets au moyen l'assignation « := ». On peut donc poser

> $x := 100$;

$$x := 100$$

Dorénavant, x aura la valeur 100, et l'expression 2^x prendra donc la valeur :

> 2^x ;

$$1267650600228229401496703205376$$

Il faut bien distinguer cette assignation de la relation mathématique usuelle « = » d'égalité.

Aux fins d'une utilisation vraiment efficace des systèmes de calculs formels, la capacité qui est de loin la plus importante est la possibilité de définir de nouvelles fonctions (ou procédures) de calcul, à partir de celles déjà connues. On a ainsi un riche environnement de programmation spécialisée pour les mathématiques. Ces nouvelles fonctions peuvent se définir de nombreuses façons, mais celle qui est la plus naturelle est probablement via la récursivité. Ainsi on peut introduire la fonction :

```
> F := n->
      if n ≤ 1 then 1
           else F(n − 1) + F(n − 2)
      fi :
```

Dorénavant, « F » est la fonction d'une variable qui calcule (récursivement) les nombres de Fibonacci. On aura donc :

> $F(20)$;

$$10946$$

1. Puisque défini à un multiple scalaire prêt.

Les changements de lignes et les espaces supplémentaires n'ont ici aucun effet sur le calcul. Ils ne servent qu'à disposer la description de la fonction « F » de manière plus agréable. Normalement, le résultat d'une telle instruction est d'afficher le texte de la fonction ainsi définie. Le fait d'utiliser le « : », plutôt que le « ; » comme indication de fin de l'instruction, évite cet affichage assez inutile. Nous allons illustrer dans cette annexe comment il est facile d'utiliser de tels systèmes pour manipuler les objets combinatoires que nous avons rencontrés dans ce texte.

Aspects numériques

Nous allons voir dans cette première section pourquoi il est important de travailler avec des entiers de taille arbitraires et des nombres réels de précisions assez grandes lorsque l'on cherche à comprendre les propriétés de certains objets mathématiques. C'est d'abord le cas en théorie des nombres lorsque l'on étudie des familles de nombres comme les nombres de Mersenne (de la forme $2^p - 1$, pour p premier), ou encore lorsque l'on étudie les développements en fractions continues de nombres réels. Ainsi, tout système de calcul formel permet de manipuler des nombres de tailles arbitraires. Par exemple :

$> \mathbf{100!;}$

$$933262154439441526816992388562667004907159682643816214$$
$$685929638952175999932299156089414639761565182862536979$$
$$208272237582511852109168640000000000000000000000$$

D'autre part, les manipulations traditionnelles des entiers sont aussi disponibles. Ceci permet donc diverses explorations propres à la théorie des nombres comme, par exemple, la factorisation des entiers de la forme 2^p-1 (pour un premier p), pour déterminer ceux d'entre eux qui sont premiers. La *fonction* du système qui permet de factoriser des entiers se nomme **ifactor**. Par exemple,

$> \mathbf{ifactor}(2^3 - 1)\mathbf{;}$

$$7$$

$> \mathbf{ifactor}(2^5 - 1)\mathbf{;}$

$$31$$

$> \mathbf{ifactor}(2^7 - 1)\mathbf{;}$

$$127$$

$> \mathbf{ifactor}(2^{11} - 1)\mathbf{;}$

$$(23)\quad(89)$$

On dispose aussi d'outils algorithmiques permettant d'itérer ou de choisir des processus. Dans notre cas il s'agit de la commande d'itération

for ... **to** ... **do** ... **od**

ainsi que d'une commande permettant le choix du traitement des divers cas d'un problème

if ... **then** ... **else** ... **fi**

On pourra par exemple demander
> **for** i **to 30 do if isprime**(i) **then print**$(i, 2^i - 1,$ **ifactor**$(2^i - 1))$ **fi od;**

$$
\begin{array}{rrr}
2, & 3, & 3 \\
3, & 7, & 7 \\
5, & 31, & 31 \\
7, & 127, & 127 \\
11, & 2047, & (23)(89) \\
13, & 8191, & 8191 \\
17, & 131071, & 131071 \\
19, & 524287, & 524287 \\
23, & 8388607, & (47)(178481) \\
29, & 536870911, & (233)(1103)(2089)
\end{array}
$$

Mais, comme nous allons le constater, il faut aussi pouvoir manipuler des nombres réels avec une précision arbitraire. Ainsi le nombre π représenté formellement par Pi admet l'approximation rationnelle :

$$3.141592654$$

Si l'on utilise cette approximation pour développer π en fraction continue, on obtient le résultat suivant

$$
3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \cfrac{1}{293 + \cfrac{1}{11 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{7 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{3 + \frac{1}{2}}}}}}}}}}}}
$$

Comme cette fraction continue est caractérisée par la liste des *quotients partiels*, $3, 7, 15, 1, 293, \ldots, 2$, on la désignera dorénavant par

$$[3, 7, 15, 1, 293, 11, 1, 1, 7, 2, 1, 3, 3, 2] \tag{1}$$

Or l'étude du développement en fraction continue de π, nécessite la connaissance d'un grand nombre de chiffres significatifs de π. En effet, une meilleure approximation de π

$$3.14159265358979323846264338327950288419 7$$

donnera plutôt le développement

$$\begin{gathered}[3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, \\ 84, 2, 1, 1, 15, 3, 13, 1, 4, 2, 6, 6, 99, 1, 12, 2, 1, 4, 1, 1, \\ 1, 1, 16, 11, 1, 2, 1, 8, 5, 3, 4, 1, 14, 1, 6, 2, 3, 1, 1, 1, 2, 1, \\ 3, 5, 1, 3, 1, 2, 1, 1, 2, 1, 2, 3, 1, 2, 6, 3]\end{gathered} \tag{2}$$

ce qui montre que le développement (1) n'est valable que pour ses quatre premiers quotients partiels. Une question naturelle à ce stade de la discussion serait de chercher à savoir jusqu'à quel point le développement (2) est juste, mais ce genre d'exploration est laissé au lecteur intéressé.

D'autre part, un calcul usuel (avec des approximations quand même assez bonnes) avec des nombres réels peut mener à des conclusions fausses. Ainsi, il y a l'exemple célèbre suivant (calculé ici à 25 chiffres significatifs près)

$$e^{\pi\sqrt{163}} \simeq 262537412640768744.0000000$$

Ceci suggère que ce nombre est un entier, mais un calcul à 50 chiffres significatifs près montre que

$$e^{\pi\sqrt{163}} \simeq 262537412640768743.99999999999925007259719818568865 \tag{3}$$

Une des explications de ce phénomène se trouve dans le fait que le troisième nombre du développement en fraction continue de (3) est très grand

$$\begin{gathered}[262537412640768743, 1, 1333462407511, 1, 8, 1, 1, 5, 1, 5, 10, 5, 4, \\ 2, 6, 2, 1, 1, 1, 2, 2, 13, 1, 7, 1, 1, 1, 3, 2, 2, 2, 8, 8, 1, 6, \\ 1, 1, 16, 2, 1, 1, 1, 1, 6, 4]\end{gathered}$$

Bien que cela ne soit pas la fin de cette histoire, passons à un autre exemple.

Considérons la matrice (de Hilbert) suivante.

$$V = \begin{pmatrix} \frac{1}{83} & \frac{1}{84} & \frac{1}{85} & \frac{1}{86} & \frac{1}{87} & \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} \\ \frac{1}{84} & \frac{1}{85} & \frac{1}{86} & \frac{1}{87} & \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} \\ \frac{1}{85} & \frac{1}{86} & \frac{1}{87} & \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} \\ \frac{1}{86} & \frac{1}{87} & \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} \\ \frac{1}{87} & \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} & \frac{1}{95} \\ \frac{1}{88} & \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} & \frac{1}{95} & \frac{1}{96} \\ \frac{1}{89} & \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} & \frac{1}{95} & \frac{1}{96} & \frac{1}{97} \\ \frac{1}{90} & \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} & \frac{1}{95} & \frac{1}{96} & \frac{1}{97} & \frac{1}{98} \\ \frac{1}{91} & \frac{1}{92} & \frac{1}{93} & \frac{1}{94} & \frac{1}{95} & \frac{1}{96} & \frac{1}{97} & \frac{1}{98} & \frac{1}{99} \end{pmatrix}$$

Un calcul précis avec des (grands) rationnels du déterminant de la matrice V donne approximativement le résultat

$$\det(V) = .5672003590\ 10^{-127}$$

Or si l'on effectue le même calcul en substituant aux entrées de la matrice V des approximations décimales (à 10 chiffres), on obtient plutôt le résultat

$$.5087318076\ 10^{-79}$$

qui est 10^{48} fois trop grand.

Il y a en fait une grande diversité d'exemples qui montrent l'importance de savoir manipuler des nombres de précision arbitrairement grande lorsque l'on cherche à explorer des propriétés liées à ces nombres. Voilà pourquoi dans notre système de calcul formel les entiers, les rationnels, les réels et les complexes sont représentés de façon exacte ($100\,!$, $\frac{1}{3}$, $\sqrt{2}$, π, ou $e^{i\pi}$), et les simplifications usuelles de ces expressions sont disponibles ($3\,\frac{1}{3} = 1$, ou $e^{i\pi} = -1$).

Calcul abstrait

La véritable particularité du calcul formel est cependant le fait qu'il rend possible la manipulation d'expressions mathématiques abstraites. Ainsi la commande suivante provoque l'évaluation formelle de $\sum_{i=1}^{n-1} i$.

> **sum**$(i,\ i = 1..n-1)$ **;** (**sum** est une fonction du système)

$$\frac{n^2}{2} - \frac{n}{2}$$

On peut alors demander (avec la commande **factor**) à notre système de factoriser cette expression pour obtenir :
> **factor(%) ;** (pour Maple, le symbole % désigne l'expression qui vient d'être calculée)

$$\frac{n(n-1)}{2}$$

De même, on effectue facilement les calculs suivants, ce qui permet de développer une idée de la formule générale pour la somme :

$$\sum_{i=1}^{n-1} i^k$$

On a les cas particuliers

$$\begin{aligned}\sum_{i=1}^{n-1} i^2 &= \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6} \\ &= \frac{n(2n-1)(n-1)}{6}\end{aligned}$$

$$\begin{aligned}\sum_{i=1}^{n-1} i^3 &= \frac{n^4}{4} - \frac{n^3}{2} + \frac{n^2}{4} \\ &= \frac{n^2(n-1)^2}{4}\end{aligned}$$

$$\begin{aligned}\sum_{i=1}^{n-1} i^4 &= \frac{n^5}{5} - \frac{n^4}{2} + \frac{n^3}{3} - \frac{n}{30} \\ &= \frac{n(2n-1)(n-1)(3n^2-3n-1)}{30}\end{aligned}$$

Ce genre de possibilités n'est évidemment pas limité aux sommations finies, ainsi on obtient du système des valeurs de la fonction *Zeta* de Riemann

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$$

en demandant de calculer

> **sum**$(1/i^2,\ i$**=1..infinity**$)$ **;**

$$\frac{\pi^2}{6}$$

ou encore des expressions qui font indirectement appel à cette fonction

> **sum**$(1/(i^2+i^3),\ i$**=1..infinity**$)$ **;**

$$\frac{\pi^2}{6}-1$$

Ce qui implique que notre système de calcul formel doit connaître plusieurs fonctions mathématiques de base ainsi que leurs manipulations.

A la base d'un système de calcul formel, on a bien entendu à sa disposition les opérations sur les polynômes comme le calcul du plus grand commun diviseur. Ceci permet d'effectuer des expériences comme celle qui suit

> **gcd**$(x^{369}-1,x^{82}-1)$ **;**

$$x^{41}-1$$

Ce qu'il est intéressant de confronter au résultat suivant

> **igcd**$(369,82)$ **;** (ici **igcd** désigne le plus grand commun diviseur d'entiers)

$$41$$

Ceci suggère une exploration de la factorisation des polynômes de la forme x^n-1.

> **for** k **to 9 do print**$(k$**, ifactor**(k)**, factor**(x^k-1)**) od ;**

$$\begin{array}{rrl}
1, & 1, & (x-1)\\
2, & 2, & (x-1)(x+1)\\
3, & 3, & (x-1)(x^2+x+1)\\
4, & 2^2, & (x-1)(x+1)(x^2+1)\\
5, & (5), & (x-1)(x^4+x^3+x^2+x+1)\\
6, & (2)(3), & (x-1)(x+1)(x^2+x+1)(x^2-x+1)\\
7, & 7, & (x-1)(x^6+x^5+x^4+x^3+x^2+x+1)\\
8, & 2^3, & (x-1)(x+1)(x^2+1)(x^4+1)\\
9, & 3^2, & (x-1)(x^2+x+1)(x^6+x^3+1)
\end{array}$$

On peut alors exercer son sens de l'observation pour mettre en évidence les polynômes dits *cycloto-*

miques

$$\varphi_1(x) = x - 1$$
$$\varphi_2(x) = x + 1$$
$$\varphi_3(x) = x^2 + x + 1$$
$$\varphi_4(x) = x^2 + 1$$
$$\varphi_5(x) = x^4 + x^3 + x^2 + x + 1$$
$$\varphi_6(x) = x^2 - x + 1$$
$$\varphi_7(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$
$$\varphi_8(x) = x^4 + 1$$
$$\varphi_9(x) = x^6 + x^3 + 1$$

et observer encore plus précisément, par exemple, que

$$x^9 - 1 = \varphi_9(x)\varphi_3(x)\varphi_1(x) = (x^6 + x^3 + 1)(x^2 + x + 1)(x - 1).$$

ce qui suggère la formule générale

$$x^n - 1 = \prod_{d \,\text{divise}\, n} \varphi_d(x)$$

D'autre part il est bien connu que les racines complexes (dites *racines de l'unité*) des polynômes $x^n - 1$ sont de la forme

$$e^{\frac{2ki\pi}{n}}$$

Pour trouver parmi ces racines, celles qui sont racines des polynômes $\varphi_n(x)$, on pourra facilement évaluer ces polynômes en les diverses racines nièmes de l'unité, pour obtenir

$$\varphi_{12}(e^{\frac{i\pi}{6}}) = 0$$
$$\varphi_{12}(e^{\frac{2i\pi}{6}}) = 1 - \sqrt{3}i$$
$$\varphi_{12}(e^{\frac{3i\pi}{6}}) = 3$$
$$\varphi_{12}(e^{\frac{4i\pi}{6}}) = 1 + \sqrt{3}i$$
$$\varphi_{12}(e^{\frac{5i\pi}{6}}) = 0$$
$$\varphi_{12}(e^{\frac{6i\pi}{6}}) = 1$$
$$\varphi_{12}(e^{\frac{7i\pi}{6}}) = 0$$
$$\varphi_{12}(e^{\frac{8i\pi}{6}}) = 1 - \sqrt{3}i$$
$$\varphi_{12}(e^{\frac{9i\pi}{6}}) = 3$$
$$\varphi_{12}(e^{\frac{10i\pi}{6}}) = 1 + \sqrt{3}i$$
$$\varphi_{12}(e^{\frac{11i\pi}{6}}) = 0$$
$$\varphi_{12}(e^{\frac{12i\pi}{6}}) = 1$$

ce qui suggère qu'en général les valeurs de k pour lesquelles $\varphi_n(e^{\frac{2ki\pi}{n}}) = 0$ sont les k relativement premiers à n. En conséquence on aura

$$\varphi_n(x) = \prod_{gcd(k,n)=1} \left(x - e^{\frac{2ki\pi}{n}}\right)$$

Résolution d'équations

La recherche des racines n'est qu'un cas très spécial de résolution d'équations. Plus généralement on veut pouvoir obtenir la (ou les) solution(s) d'équations avec paramˆètres

> **solve**$(a\,x^2 + b\,x + c,\ x)$ **;**

$$\frac{\sqrt{b^2-4ac}-b}{2a}$$
$$\frac{-b-\sqrt{b^2-4ac}}{2a}$$

> **solve**$(x^3 + x - 1,\ x)$ **;**

$$(1/2 + \frac{\sqrt{31}}{6\sqrt{3}})^{1/3} + (1/2-\frac{\sqrt{31}}{6\sqrt{3}})^{1/3}$$
$$-\frac{(1/2 + \frac{\sqrt{31}}{6\sqrt{3}})^{1/3}}{2} - \frac{(1/2-\frac{\sqrt{31}}{6\sqrt{3}})^{1/3}}{2} + \frac{\sqrt{3}((1/2 + \frac{\sqrt{31}}{6\sqrt{3}})^{1/3} - (1/2-\frac{\sqrt{31}}{6\sqrt{3}})^{1/3})I}{2}$$
$$-\frac{(1/2 + \frac{\sqrt{31}}{6\sqrt{3}})^{1/3}}{2} - \frac{(1/2-\frac{\sqrt{31}}{6\sqrt{3}})^{1/3}}{2} - \frac{\sqrt{3}((1/2 + \frac{\sqrt{31}}{6\sqrt{3}})^{1/3} - (1/2-\frac{\sqrt{31}}{6\sqrt{3}})^{1/3})I}{2}$$

ainsi que pour des systèmes d'équations

> **solve**$(\{ax + by = s,\ cx + dy = t\},\{x, y\})$ **;**

$$\left\{y = \frac{cs-at}{cb-ad}, x = -\frac{ds-tb}{cb-ad}\right\}$$

Bien évidemment il n'est pas toujours possible de donner une formule close pour la ou les solutions d'un système d'équations, comme par exemple

> **solve**$(\{x^2 + y^2 = 1,\ x^3 + 2x = y\},\ \{x, y\})$ **;**

$$x = RootOf(Z^6 + 4Z^4 + 5Z^2 - 1)$$
$$y = x^3 + 2x$$

Voilà pourquoi le système de calcul symbolique donne ici un résultat qui fait appel à la racine d'un polynôme de degré 6. On peut cependant ensuite manipuler cette racine comme objet formel.

Définitions récursives

Les définitions récursives de fonctions sont très courantes. Bien que notre système de calcul formel contienne en fait tous les outils de programmation qui permettraient de définir une nouvelle fonction comme suit

> **fibo :=proc**(n) **option remember ;**
if $n \leq 2$ **then 1**
else fibo$(n-1)$ + **fibo**$(n-2)$
fi
end :

pour ensuite calculer par exemple

> **fibo(200) ;**

$$280571172992510140037611932413038677189525$$

l'approche traditionnelle des mathématiques pour résoudre cette récurrence, définissant les nombres de Fibonnacci, donnera plutôt

> **rsolve(**$\{f(n) = f(n-1) + f(n-2),\ f(1) = a,\ f(2) = b\}$**,** f**) ;**

$$\frac{(2b-a-\sqrt{5}a)(1/2-\frac{\sqrt{5}}{2})^n}{(\sqrt{5}-1)\sqrt{5}} + \frac{(2b-a+\sqrt{5}a)(1/2+\frac{\sqrt{5}}{2})^n}{\sqrt{5}(1+\sqrt{5})}$$

Nous avons donné ici des conditions initiales générales pour mieux illustrer la capacité de résolution du système. En donnant à a et b la valeur 1 dans cette expression, le système nous donnera après simplification l'expression suivante

$$f(n) = \frac{(1/2+\frac{\sqrt{5}}{2})^n}{\sqrt{5}} - \frac{(1/2-\frac{\sqrt{5}}{2})^n}{\sqrt{5}}$$

On pourra alors demander la valeur de cette expression pour un n donné, et le système proc`édera aux simplifications qui s'imposent. Une autre façon (parmi tant d'autres) d'obtenir ces mêmes nombres de Fibonnacci serait de procéder par développement en série de Taylor de leur fonction génératrice

$$\frac{1}{1-x-x^2}$$

ce qui donnerait

> **taylor(**$1/(1-x-x^2)$**,** x**)) ;**

$$1 + x + 2x^2 + 3x^3 + 5x^4 + 8x^5 + O(x^6)$$

On peut bien sur obtenir plus de termes de ce développement en demandant par exemple :

> **taylor(**$1/(1-x-x^2)$**,**x**,** 20**)) ;**

ce qui donnera 20 termes du dit développement.

Calcul différentiel

Un autre domaine de prédilection pour le calcul formel est bien entendu le calcul différentiel et intégral. En plus des possibilités de dérivation formelle, comme par exemple

> **diff**$(e^{\sin(x)}\log x,\ x)$ **;**

$$\cos(x)e^{\sin(x)}\log x + \frac{e^{\sin(x)}}{x},$$

on a la possibilité d'explorer les propriétés des primitives de toute sorte.

> **int**$(\log(x),\ x)$ **;**

$$x\log x - x$$

> **int**$(\log^2(x),\ x)$ **;**

$$x\log^2(x) - 2x\log(x) + 2x$$

> **int**$(\log^3(x),\ x)$ **;**

$$x\log^3(x) - 3x\log^2(x) + 6x\log(x) - 6x$$

> **int**$(\log^4(x),\ x)$ **;**

$$x\log^4(x) - 4x\log^3(x) + 12x\log^2(x) - 24x\log(x) + 24x$$

Ce qui suggère une formule générale simple pour

$$\int \log^k(x)dx$$

que le lecteur assidu s'empressera de trouver.

On a aussi la possibilité de manipuler abstraitement les résultats obtenus. Ainsi, bien que le système ne trouve pas de forme close pour

$$\int \frac{e^{x^2}\sin(x)}{x}\mathrm{d}x,$$

on peut quand même lui demander quel est le développement en série de taylor de cette fonction

> **taylor(%,**x**)**

$$x + \frac{5x^3}{18} + \frac{41x^5}{600} + O(x^6)$$

D'autre part l'étude des résultats suivants peut suggérer toute sorte d'expérimentations intéressantes. Ainsi, la commande

> **convert(taylor(**$\exp(x/t)$**,** x**,** 10**), confrac) ;**

correspond à demander le développement en fraction continue de la série de taylor de $e^{x/t}$. Ce qui donne

$$1+\cfrac{x}{t+\cfrac{x}{-2+\cfrac{x}{-3t+\cfrac{x}{2+\cfrac{x}{5t+\cfrac{x}{-2+\cfrac{x}{-7t+\cfrac{x}{2+\cfrac{x}{9t}}}}}}}}}$$

Un autre type de calcul amusant fera apparaître les nombres de Stirling de première ou deuxième espèce.

> **taylor**$(\exp(-t\log(1-x)),\ x)$**;**

$$1+tx+(t+t^2)\frac{x^2}{2!}+(2t+3t^2+t^3)\frac{x^3}{3!}+(6t+11t^2+6t^3+t^4)\frac{x^4}{4!}+$$
$$(24t+50t^2+35t^3+10t^4+t^5)\frac{x^5}{5!}+O(x^6)$$

Le coefficient de $x^n/n!$ dans cette série est un polynôme dont les coefficients sont les valeurs absolues des nombres de Stirling de première espèce. Un autre exemple du même genre fait apparaître les polynômes de Stirling de seconde espèce

> **taylor**$(e^{t(e^x-1)},\ x)$**;**

$$1+tx+(t+t^2)\frac{x^2}{2!}+(t+3t^2+t^3)\frac{x^3}{3!}+(t+7t^2+6t^3+t^4)\frac{x^4}{4!}+$$
$$(t+15t^2+25t^3+10t^4+t^5)\frac{x^5}{5!}+O(x^6)$$

Mentionnons aussi que notre système de calcul formel contient les connaissances appropriées concernant les limites, lui permettant de calculer par exemple

$$\lim_{n\to\infty}\left(1+\frac{x}{n}\right)^n=e^x$$

Simplifications

Comme on a déjà eu l'occasion de le constater, l'un des aspect les plus sophistiqués d'un système de calcul formel est certainement sa capacité à simplifier des expressions complexes. En effet, c'est

cette simplification qui permet de mettre en évidence des résultats qui sinon resteraient cachés sous une pléthore de symboles. Nous allons maintenant illustrer comment on peut mettre à profit cette capacité de simplification. Considérons

> **expand**$(\cos(x+y))$; (**expand** développe l'expression donnée)

$$\cos(x)\cos(y)-\sin(x)\sin(y)$$

On peut substituer (avec **subs**) à x et y la valeur t

> **subs**$(x=t,\ y=t,\ \%)$;

$$\cos(t)^2-\sin(t)^2$$

et, après simplification de ce dernier résultat, on obtient

> **simplify**$(\%)$;

$$2\cos(t)^2-1$$

en posant $\cos(t)=u$ et en renormalisant, ceci donne le polynôme de Tchebicheff

$$T_2(u)=u^2-1/2$$

Un procédé analogue permet de construire le polynôme $T_3(u)$ à partir de $\cos(x+y+z)$. Les étapes de ce dernier calcul seront alors

$$\begin{array}{rc} \cos(x) & \cos(y)\cos(z)-\cos(x)\sin(y)\sin(z) \\ & -\sin(x)\sin(y)\cos(z)-\sin(x)\cos(y)\sin(z) \end{array}$$

> **subs**$(x=t,\ y=t,\ z=t,\ \%)$;

$$\cos(t)^3-3\cos(t)\sin(t)^2$$

> **simplify**$(\%)$;

$$4\cos(t)^3-3\cos(t)$$

Pour obtenir enfin

$$T_3(u)=u^3-\frac{3u}{4}$$

On pourra alors étudier les propriétés de ces polynômes, comme par exemple

$$\int_{-1}^{1}\frac{(u^2-1/2)(u^3-\frac{3u}{4})}{\sqrt{1-u^2}}du=0$$

ou encore, mettre en évidence la récurrence qui caractérise les polynômes en question

$$T_n(u)=uT_{n-1}(u)-T_{n-2}(u)/4$$

pour ensuite la résoudre en demandant

> **rsolve**$(T(n) = uT(n-1) - T(n-2)/4,\ T(1) = u,\ T(2) = u^2 - 1/2,\ T)$;

$$\frac{(u\sqrt{u^2-1}-u^2+1)(\frac{u}{2}-\frac{\sqrt{u^2-1}}{2})^n}{(u-\sqrt{u^2-1})\sqrt{u^2-1}} + \frac{(u\sqrt{u^2-1}+u^2-1)(\frac{u}{2}+\frac{\sqrt{u^2-1}}{2})^n}{\sqrt{u^2-1}(u+\sqrt{u^2-1})}$$

Simplifions un peu, pour obtenir enfin

$$T_n(u) = (\frac{u}{2}-\frac{\sqrt{u^2-1}}{2})^n + (\frac{u}{2}+\frac{\sqrt{u^2-1}}{2})^n$$

Théorie des ensembles

Tout système de calcul formel (Sage, Maple, etc.) permet, entre autres, de manipuler des ensembles, des listes, et diverses constructions les concernant (avec les adaptations nécessaires). Ainsi, on peut donner le nom A à l'ensemble $\{a, b, c\}$ en posant

> $A := \{a, b, a, c, b, b\}$;

$$A := \{a, b, c\}$$

Observons ici (comme le veut la théorie) que la répétition d'un élément n'a aucun effet. Bien entendu, on a aussi les opérations usuelles sur les ensembles

> $\{a, b, c\}$ **intersect** $\{b, c, d\}$;

$$\{b, c\}$$

> $\{a, b, c\}$ **union** $\{b, c, d\}$;

$$\{a, b, c, d\}$$

> $\{a, b, c\}$ **minus** $\{b, c, d\}$;

$$\{a\}$$

En passant, rien n'empêche d'utiliser abstraitement ces opérations sur des ensembles B et C non spécifiées. Ainsi, on obtient

> $(B$ **union** $D)$ **intersect** $(C$ **union** $D)$;

$$(B \cup D) \cap (C \cup D)$$

> B **minus** B ;

$$\emptyset$$

On peut tester l'égalité d'ensembles, et l'appartenance à un ensemble, en exploitant la fonction « evalb » qui calcule la valeur logique d'une expression. Ainsi, on a

> **evalb**$(\{a, b, c\} = \{b, c, d\})$;

false

> **evalb**$(\{a, b, c\} = \{b, c, a\})$;

true

> b **in** $\{b, c, a\}$ **;**

$$b \in \{a, b, c\}$$

> **evalb**(b **in** $\{b, c, a\}$) **;**

true

La fonction « nops » est très générale. Elle permet de compter le nombre d'opérandes dans une expression. En particulier elle donne le nombre d'éléments d'un ensemble. Cependant, pour faciliter la compréhension, il est possible de lui donner un synonyme en posant :

> **card := nops :**

Par la même occasion, on se permet de mettre en place les autres synonymes :

> **‘&+‘ := ‘union‘ :**
> **‘&-‘ := ‘minus‘ :**

L'utilisation du caractère « & », dans « &+ », est nécessaire en Maple lorsqu'on désire considérer de nouveaux opérateurs binaires avec une notation « infixe ». C'est aussi une particularité de la syntaxe de Maple qui forcent l'utilisation des « ‘ » au moment de l'établissement de ces synonymes.

Pour décrire un ensemble de la forme $\{g(x) \mid x \in A\}$, on peut utiliser la fonction Maple « **seq** » qui permet de construire des séquences de valeurs $g(x)$ pour x variant dans A. Puisque l'ensemble A a déjà été défini (mais pas g), on obtient :

> $\{$**seq**($g(x), x$ **in** A)$\}$) **;**

$$\{g(a), g(b), g(c)\}$$

ou encore, avec la fonction de Fibonacci F :

> $\{$**seq**($F(x), x$ **in** $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$)$\}$) **;**

$$\{1, 2, 3, 5, 8, 13, 21, 34, 55\}$$

Autres opérations. On peut définir d'autres opérations usuelles sur les ensembles comme ci-dessous. Si $A = \{x\} + B$, le calcul de l'ensemble $\mathcal{P}[A]$ des parties de A est basé sur la récurrence :

$$\mathcal{P}[A] = \mathcal{P}[B] \ + \ \{C + \{x\} \mid C \in \mathcal{P}[B]\}. \tag{3}$$

Avec une syntaxe légèrement différente de celle déjà utilisée, on exploite cette récurrence pour obtenir la nouvelle fonction

```
> P := proc(A) local x, B :
    if A = {} then {{}}
        else x := op(1, A) :                    (x est le premier élément dans A)
            B := A &- {x} :
            P(B) &+ {seq( C &+ {x}, C in P(B)}
    fi
end :
```

Après avoir considéré le cas spécial $A = \emptyset$, on choisit x comme étant le premier élément de A, et la cinquième ligne reproduit (presque fidèlement) le membre de droite de (3). On obtient ainsi une fonction calculant l'ensemble $P(A)$ des parties de A :

```
> P({a, b, c}) ;
```

$$\{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$$

Avant de construire le produit cartésien comme opération binaire, on rappelle qu'en Maple un couple, normalement dénoté (x, y) dans des contextes mathématiques, est plutôt dénoté $[x, y]$. Ceci est dû au fait que les parenthèses usuelles sont réservées à d'autres fins syntaxiques. On pose donc

```
> `&X` := proc(A, B) local x, y :
    {seq(seq([x, y] , x in A), y in B)}
end :
```

Alors, on obtient

```
> {a, b, c} &X {a, b} ;
```

$$\{[1, a], [1, b], [2, a], [2, b], [3, a], [3, b]\}$$

et aussi

```
> {a, b, c} &X {} ;
```

$$\emptyset$$

comme il se doit. On peut maintenant obtenir la fonction qui calcule l'ensemble des relations entre A et B, simplement en posant :

```
> Relations := (A, B)-> P(A &X B) :
```

et donc on peut ensuite calculer que

```
> Relations({a, b}, {x, y}) ;
```

$$\{\emptyset, \{[a, x]\} , \{[a, y]\} , \{[b, x]\} , \{[b, y]\} , \{[a, x], [a, y]\} , \{[a, x], [b, x]\} , \{[a, x], [b, y]\} , \{[a, y], [b, x]\} ,$$
$$\{[a, y], [b, y]\} , \{[b, x], [b, y]\} , \{[a, x], [a, y], [b, x]\} , \{[a, x], [a, y], [b, y]\} , \{[a, x], [b, x], [b, y]\} ,$$
$$\{[a, y], [b, x], [b, y]\} , \{[a, x], [a, y], [b, x], [b, y]\} \}$$

Pour calculer récursivement l'ensemble des fonctions entre A et B, il suffit de poser

```
> Fonct := (A, B)->
    if A = {} then {{}}
        else x := A[1] :
            {seq(seq(f &+ {x, y} , y in B), f in Fonct(A &- {x}, B))}
    fi :
```

et alors

```
> Fonct({a, b}, {0, 1}) ;
```

$$\{\{[a, 0], [b, 0]\} , \{[a, 0], [b, 1]\} , \{[a, 1], [b, 0]\} , \{[a, 1], [b, 1]\}\}$$

Une construction similaire permet d'obtenir l'ensemble des mots de longueur k, sur un alphabet A, au moyen de la fonction Maple « cat » qui concatène deux mots.

```
> Mots := (A, k)->
      if k = 1 then A
           else {seq(seq(cat(w, x) , x in A), w in Mots(A, k - 1))}
      fi :
   > Mots({a, b, c}, 3) ;
```

$$\{aaa, aab, aac, aba, abb, abc, aca, acb, acc,$$
$$baa, bab, bac, bba, bbb, bbc, bca, bcb, bcc,$$
$$caa, cab, cac, cba, cbb, cbc, cca, ccb, ccc\}$$