

Sur la nième décimale des nombres
transcendants
ou
Le 400 milliardième chiffre
binaire de e est 1

Simon Plouffe
Centre for Experimental &
Constructive Mathematics
Wolfram Research Inc.
Lacim, UQAM

Avec la collaboration de Peter Borwein (SFU)
et David H. Bailey (Ames Research Center, NASA)

Résumé

- 0) Introduction aux nombres transcendants
- 1) Le calcul de $1/n$ est facile
- 2) On peut étendre aux polylogarithmes.
- 3) Il existe des identités pour $\text{Pi}...$
- 4) la complexité de l'algorithme est $n \cdot \log(n)$
- 5) Questions ouvertes

Que sait-on des nombres transcendants ?

C'est la classe des nombres NON algébriques.
(pas solution d'équations algébriques à coeff. rat.)
Presque tous les réels sont Tr.

Une brève histoire...

en 1844 Liouville découvrait qu'on peut construire des nombres non-algébriques...

Ce nombre est Tr. et il est aussi très près des rationnels.

$$\sum_{n=0}^{\infty} \frac{1}{10^{n!}} = 0.110001000000000000000001...$$

1873 Hermite : E est Tr.

1882 Lindemann : e est Tr.

1934 Gelfond :

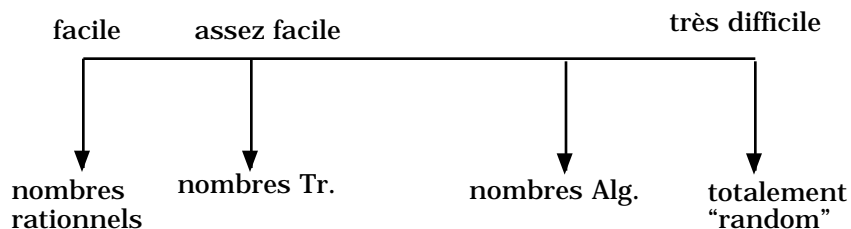
a^b est Tr. si a est algébrique $\neq 0,1$ et b algébrique irrationnel. -->
 $2^{\sqrt{2}}$ est Tr.

Depuis, quelques résultats éparses, $\sin(1)$, $J_0(1)$, $\log(\frac{1}{4})$, $(1/4)^{1/4}$:
Chudnovsky (récent).

La théorie des nombres Tr. : difficile et peu de résultats : $e^{\sqrt{2}}$ est
irrationnel, A. Froda. $e^{\sqrt{2}}$ est ?

On pourrait se faire le diagramme suivant...

Les Réels



Cette image pourrait cependant être fausse puisqu'il pourrait exister des exemples (très difficiles) qui seraient Tr.

De façon générale (mis à part les nombres artificiellement construits).

Irrationnels ---> travail (cpu + mémoire) = digits.

+ de travail = + de décimales...

Calcul de Pi à 51,539,600,000 de décimales (août 1997).
avec la 5eme plus grosse machine au monde.
Le Hitachi SR-2201 (1024 processeurs).

<http://www.cecm.sfu.ca/projects/ISC/records.html>

Donc,...

Il est donc normal de penser que la notion de distance (du point décimal) va avec la notion de travail.

Mais ceci est faux

On peut calculer la nième décimale (bin ou hexadécimale) de avec peu de travail, sans calculer les (n-1) précédentes et avec virtuellement aucune mémoire.

Comment ? d'abord 3 observations.

1) Le calcul de $1/n$ est très facile.

2) On peut étendre aux logarithmes, arctg et polylogarithmes.

3) Il existe de belles formules pour $\log(2)$, $\log(2)^2$, 2 , $\arctg(1/2)$...

La magie réside dans le calcul de 1/n.

Tout le monde connaît la longue division, un ordinateur effectue le calcul de la même façon.

$$1,00000000000000000000 \mid \underline{17} \\ 0,0588235294117647\dots$$

Ce calcul s'effectue avec 1 décimale à la fois.

Très très vieil algorithme... en se bornant à la base 10.
La k^{ième} décimale de 1/n est donnée par la solution de

$$10^k \equiv r \pmod{n}$$

les décimales sont avec le calcul de r/n à partir du rang (k+1).
Plus exactement la k^{ième} est $\lfloor 10^k r/n \rfloor$.

En d'autres mots,

$$\frac{10^k}{n} = \frac{r}{n}$$

EST la même chose. Donc en base b on a,

$$b^k \equiv r \pmod{n}$$

Mais où est le problème ?

Le problème est lorsque K est grand ou arbitraire.
En utilisant l'exponentiation rapide (Knuth, ACP #2),
vieux truc datant de 200 BC appelé aussi The Binary Method.
Par exemple on veut les décimales de 1/257 à partir du rang
1000. Alors il faut calculer,

$$10^{999} = ? \pmod{257}$$

Il suffit d'évaluer,

$10^0, 10^1, 10^2, 10^3, 10^6, 10^7, 10^{14}, 10^{15}, 10^{30}, 10^{31},$
 $10^{62}, 10^{124}, 10^{248}, 10^{249}, 10^{498}, 10^{499}, 10^{998}$ et enfin $10^{999} \pmod{257} = 96$.
Donc $96/257 = 0,373540$
sont les décimales à partir de 1000. La 1000^{ème} est 3.

En sachant que, si on a (analyse numérique I)

$$b^m \equiv r \pmod{n}$$

alors

$$(b^m)^2 \equiv r^2 \pmod{n}$$

Donc en multipliant par b et en mettant au carré successivement on arrive **en peu d'étapes** à 10^{999} .

En fait le développement binaire de 999 nous dit quand il faut effectuer l'une et/ou l'autre opération, d'où le nom de la méthode.

2 remarques importantes.

1) Pour un k donné on peut sauver des étapes, ici $(1.5 \cdot \log(k)) \rightarrow (1.29 \cdot \log(k))$ en choisissant de façon judicieuse les fois où il faut mettre au carré plutôt que multiplier par b. Voir Knuth, Brlek : Addition Chains.

2) En utilisant le petit théorème de Fermat, favorable si $k \gg n$ mais pas beaucoup si $k \sim n$. (th. des nombres).

Sauve 30 % du calcul en moyenne MAIS codage + compliqué.

Il faut se rendre compte que le calcul de cet algorithme est TRÈS serré, en BASIC le programme a 432 caractères de longueur (!). Donc toute addition de code ralentit la boucle principale.

La prochaine étape était de remarquer qu'il existe de belles séries qui sont simples et élégantes. Par exemple,

$$\sum_{n=1}^{\infty} \frac{1}{n2^n} = -\log(1/2) = \log(2).$$

Cette série se prête particulièrement bien au calcul puisque le terme 2^n n'est qu'un **DÉPLACEMENT** vers la droite du point 'décimal', il n'y a essentiellement que $1/n$ à calculer.

Si on veut la 10^6 position en base 2, on tasse de 1 position à chaque étape + le calcul de $1/n$. Il suffira de convertir n en binaire, on peut effectuer la conversion sans vraiment calculer ce développement.

La série converge si on pose $k=10^6$, oui, $1/(n \cdot 2^{**}n)$ est aussi petit qu'on veut.

Donc on peut calculer en base b, si on peut représenter le nombre par une série dans cette base (évident).

p.e. $\sum_{n=1} \frac{1}{n4^n} = -\log(3/4) = \log(3) - 2\log(2)$ permet de calculer $\log(3)$ en base 2, ... --> $\log(5), \log(7), \dots$

avec des variantes de séries connues pour le log.

Comme, $\log(1-x), \log \frac{1-x}{1+x}$ permet d'avoir la plupart des petits entiers. Mais pas TOUS, $\log(23)$ est impossible à obtenir de cette façon, puisque $2047=23*89$, on peut l'avoir avec 89 mais pas isolément.

Aussi, trivialement $\arctg(1/2)$ est calculable en base 2.

Alors ???!, ?, on a (Euler),

$$\frac{1}{4} = \arctg\left(\frac{1}{2}\right) + \arctg\left(\frac{1}{3}\right)$$

en base (2+3) ?, en base (2*3) ?, en base 2/3 ?...

Does not compute !...

Quelle est cette classe de nombres en fait ?

C'est la classe des nombres qui sont calculables en TEMPS : polynomial et en ESPACE : log-polynomial. On l'appelle la classe **SC** (Steven Class), de Steven Cook, mentionné dans Knuth. Cet 'espace' est limité, on ne peut pas utiliser les moyens techniques modernes de calcul comme la FFT (produit de grands nombres), on ne sait même pas si on peut convertir un nombre d'une base à l'autre. La conversion de base est un problème qui consomme du temps et de l'espace...

Il suffit d'avoir un mot de largeur et de descendre. la colonne + le dernier triangle pour avoir suffisamment de précision (largeur du mot). On arrête lorsque $m=k$.

Si on se restreint à n avec quelques mots mémoires de largeur (mise au carré) on peut donc calculer avec une grande précision (en base b), les nombres,

$$\sum_{n=1} \frac{1}{p(n)b^n}$$

et (conséquemment) on peut étendre à d'autres bases comme,

$$\sum_{n=1}^{\infty} \frac{1}{n10^n} = \log \frac{9}{10}$$

qui est $2\log(3) - \log(2) - \log(5)$.

Aussi lorsque $b=10^{96}$ à la 5,000,000,000 ième décimale (SFU) et $\log(9/10)$ à la 100,000,000 ième décimale.

On peut faire les $\text{arctg}(x)$, en base $b=x$, comme $\text{arctg}(1/2)$ en base 2. (ok, ok).

Finalement les polylogarithmes, en s'inspirant des belles identités du livre de Lewin. Oui mais pas directes. $\text{Li}_2(1/2)$ s'exprime avec $\log(2)^*$ et 2 .

Tout de même, avec arctg et un peu de finesse, on a,

$$\frac{\pi}{2} = 2\operatorname{arctg}\left(\frac{1}{\sqrt{2}}\right) + \operatorname{arctg}\left(\frac{1}{\sqrt{8}}\right)$$

en envoyant Mr. 2 de l'autre côté, on trouve facilement,

$$\sqrt{2} = 4f(1/2) + f(1/8)$$

avec

$$f(x) = \sum_{i=0}^{\infty} \frac{(-1)^i x^i}{2i+1}$$

D'autres comme ça ? oui avec 3 en base 2.

$$-9f(1/8) = \sqrt{3} + 3\log(3)$$

avec

$$f(x) = \sum_{i=0}^{\infty} \frac{(-1)^i x^i}{3i+1}$$

On revient encore avec le livre de Lewin, quelques autres avec des combinaisons de (3) , $2 \log(2)$ et $\log(2)^3$, mais aucune (3) avec pris isolément.

Il ne reste alors qu'à utiliser PSLQ de Bailey la version américaine de l'algorithme LLL.

Il est connu que les logiciels de calcul symbolique se plantent assez facilement avec les fonctions à valeurs complexes. Un bel exemple est le logarithme.

On se calme... ! Tout d'abord on a les séries du type,

les arctg avec $\sum_{n=0}^{\infty} \frac{(-1)^n x^n}{2n+1}$, les logs avec $\sum_{n=1}^{\infty} \frac{x^n}{n}$ ou des variantes de ces séries.

Une partie des difficultés tient au fait que Maple a beaucoup de difficultés à manipuler les logs avec des valeurs complexes.

On a ceci,

si c'est (n) -----> logarithmes.
si c'est (2n+1) -----> 2.
si c'est (+/-)(2n+1) -----> arctan.
si c'est 3n+1 -----> 3.
et pour 4n+1 -----> **surprise.**

Toutes ces formules sont en fait sont des formes variées de **LOGARITHMES** seulement. Après tout est un log !, c'est le $\log(-1)/i$. Arctg(x) est un log également,...

On a les identités suivantes en fouillant systématiquement le Dilogarithme,

$$2 = 36L_2\left(\frac{1}{2}\right) - 36L_2\left(\frac{1}{4}\right) - 12L_2\left(\frac{1}{8}\right) + 6L_2\left(\frac{1}{64}\right)$$

et aussi,

$$\log(2)^2 = 4L_2\left(\frac{1}{2}\right) - 6L_2\left(\frac{1}{4}\right) - 2L_2\left(\frac{1}{8}\right) + L_2\left(\frac{1}{64}\right)$$

Mais qui encore une fois n'apparaissent pas directement, seulement avec l'équation fonctionnelle une fois travaillée.

Ces mêmes formules se réécrivent,

$$\frac{2}{36} = \frac{a_i}{2^i i^2}, \text{ avec } a_i = [1, -3, -2, -3, 1, 0]$$

et aussi ,

$$\log(2)^2 = \frac{b_i}{2^i i^2}, \text{ avec } b_i = [-2, -10, -7, -10, 2, -1]$$

On peut remarquer que le calcul de $\log(2)$, $\log(3)$, $\log(5)$ mis sous cette forme permet (en prenant 15 mémoires) de calculer une centaine de logarithmes en MÊME temps.

Toutes les formules obtenues peuvent se mettre sous cette forme, plus pratique pour le calcul.

Finalement après 1 mois à tâtons, le programme PSLQ et 2 secondes de temps machine à 00h29 le 19 septembre 1995...

$$= 4 {}_2F_1 \left(\frac{1, 1/4}{5/4}; -\frac{1}{4} \right) + 2 \operatorname{arctg}(1/2) - \log(5)$$

En collectant les termes, le premier est une bissection de l'arctg, ... on a,

$$= \sum_{n=0}^{\infty} \frac{1}{16^n} \left(\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6} \right)$$

Avec 4 termes proportionnels à 1/n et un déplacement de 4 positions à chaque n (binaire).

On a donc lancé un SGI Power Challenge, R8000 pour 48 heures et obtenu la 40,000,000,000ième décimale (binaire) de π qui est 1, suivi de 0,0,1,0,0,1,0,... une trentaine de bits.

IL y a plusieurs moyens qui auraient pu arriver à la même formule, par exemple,

$$= \int_0^1 \frac{16x - 16}{x^4 - 2x^3 + 4x - 4} dx$$

Maple arrive à trouver avec cette dernière mais pas celle sous forme hypergéométrique, FAUX selon Maple. On a trouvé au moins une autre forme semblante et 2 autres pour π et $\log(2)^3$.

Tout ceci soulève d'intéressantes questions et remarques...

- 1) On peut rejoindre la 10^{15} position binaire de au moins avec cet algorithme, peut-être plus, un CRAY YMP effectue 29,5 milliards d'opérations/seconde sur 64 bits.
- 2) L'algorithme est parallélisable et c'en est même embarrassant (Rob Corless).
- 3) La distance du point 0. à la nième décimale n'est pas ce que l'on croyait, la 57ème position est plus loin que la 512ème.
- 4) est-il normal en base 2 ?, a-t-on les moyens de le prouver maintenant ?
- 5) Une formule existe-t-elle pour en base 10 ?
- 6) Y-a-t-il un motif dans le développement binaire de $\log(2)$?
- 7) Si (en plus de l'algorithme) on applique les outils modernes, FFT et calcul parallèle jusqu'où peut-on se rendre ?, 10^{20} ?,
- 8) 2 erreurs 'graves' ont été détectées dans les machines IBM 590 et R8000 en calculant ...
- 9) En ce moment même, Fabrice Bellard calcule la position 10^{12} en France, le calcul se terminera cette semaine!**

Daniel Shanks a dit en 1962 (à 100265 décimales),
“We will NEVER reach the billion'th digit of ”

Borwein & Borwein (1988) :

“We will never reach the 10^{1000} digit of ”

Spock , Star Trek, 1967 ...

“Computer !, compute to the LAST digit”.



Références

LLL et PSLQ

- 1) Faire ?lattice dans une session Maple
- 2) Chercher les noms Ferguson-Forcade, Helaman Ferguson et David H. Bailey.

Aussi : MPFUN, PSLQ dans LYCOS

<http://www.nas.nasa.gov/NAS/TechReports/RNRreports/dbailey/RNR-91-032/RNR-91-032.html>

article :

voir :

<http://www.lacim.uqam.ca/home/plouffe/WWW>

Peter B. Borwein :

<http://www.cecm.sfu.ca/personal/~pborwein>

David H. Bailey :

<http://www.nas.nasa.gov/>