LOGIQUE & CALCUL

Des indécidables à portée de main

Les énoncés dont on ne peut prouver ni qu'ils sont vrais ni qu'ils sont faux semblent moins rares qu'on ne l'imaginait : de tels « indécidables », redoutés par les mathématiciens, ont été trouvés avec des problèmes portant sur de petites machines de Turing.

Jean-Paul DELAHAYE

Si un problème mathématique paraît assez simple, comme « Est-ce que, pour tout entier positif n, $4n^4 + 1$ est un nombre premier ? », le mathématicien ne doute pas qu'il saura le résoudre. Bien sûr, d'un mathématicien à l'autre, cette perception de ce qui est facile et de ce qui ne l'est pas change, et plus un mathématicien est compétent, plus nombreux sont les problèmes qu'il est capable de classer en « facile » ou « probablement difficile ».

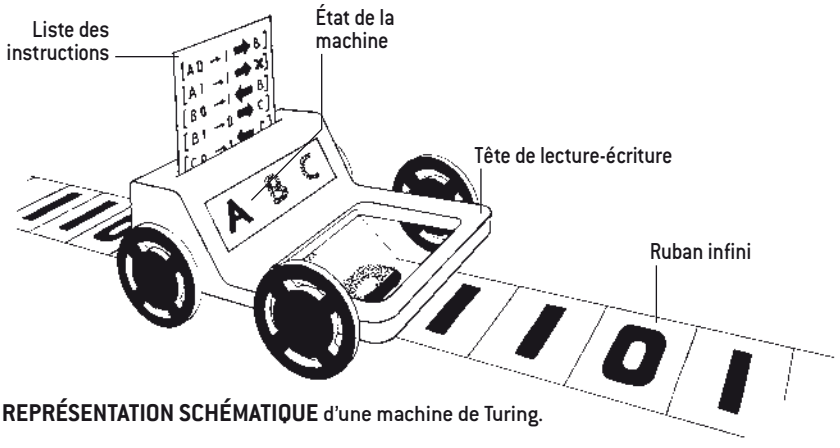
Il se trouve cependant que même les meilleurs des mathématiciens sont impuissants à évaluer la difficulté de questions à l'apparence élémentaire. Fermat n'avait certainement pas imaginé combien il était difficile de démontrer qu'il n'y a pas de solution à l'équation $n^q + m^q = p^q$, où n, m, p désignent des entiers positifs et q un entier supérieur à 2. De graves obstacles sont dissimulés dans des énoncés d'allure innocente. Les problèmes les plus difficiles sont ceux qualifiés d'« indécidables ». Un jeu utile et délicat est de repérer les indécidables les plus simples dans le but de mieux comprendre où cette apparition d'obstacles graves risque de se produire. Identifier où se niche leur absolue difficulté fait progresser les mathématiques aussi bien que la résolution de problèmes.

Qu'est-ce que l'indécidabilité ? Un énoncé est indécidable dans une théorie donnée si celle-ci ne peut ni démontrer l'énoncé, ni démontrer sa négation. Bien sûr, pour établir que des énoncés sont indécidables dans une théorie mathématique donnée T, il faut avoir très précisément défini cette théorie. Il faut aussi avoir acquis une maîtrise de ce qui s'y passe et, en particulier, il faut pouvoir prouver des théorèmes à propos des démonstrations !

C'est la tâche de la logique mathématique, parfois nommée métamathématique, puisqu'elle fournit une compréhension mathématique des mathématiques. Son développement aux XIX^e et XX^e siècles nous a permis de comprendre et de mesurer avec finesse le pouvoir déductif des théories.

Démontrer l'axiome des parallèles (par un point extérieur à une droite donnée D, il ne passe qu'une droite D' ne rencontrant pas D) ou sa négation à partir des autres axiomes de la géométrie du plan n'est pas possible. L'axiome des parallèles est un indécidable de la géométrie « de base », aussi dénommée géométrie absolue, ou géométrie neutre. De même, démontrer l'« axiome du choix » et l'« hypothèse du continu » ou leur négation à partir des axiomes habituels de la théorie des ensembles est impossible. C'est un résultat obtenu en deux temps par l'Autrichien Kurt Gödel en 1938 et l'Américain Paul Cohen en 1963.

Mieux et beaucoup plus général : selon le premier théorème d'incomplétude de Gödel démontré en 1931, dès qu'une théorie mathématique T non contradictoire est capable de mener suffisamment de raisonnements arithmétiques (et c'est le cas de toutes les théories intéressantes), alors il existe des énoncés E qui ont un sens dans cette théorie et qui sont indécidables. Autrement dit, ni l'énoncé E lui-même, ni sa négation (non-E) ne sont démontrables dans T : la théorie est lacunaire, incomplète.



REPRÉSENTATION SCHÉMATIQUE d'une machine de Turing.

© PLS/Cassandra Vion

Plus précis, le second théorème d'incomplétude de Gödel établit que l'énoncé affirmant que la théorie T est non contradictoire est l'un des indécidables de T : une théorie n'a pas le pouvoir de démontrer d'elle-même qu'elle est non contradictoire… sauf si elle est contradictoire, auquel cas elle n'a aucun intérêt.

Ces formidables résultats de Gödel établissent que l'indécidabilité est toute proche des questions que l'on considère comme fondamentales, car la non-contradiction d'une théorie T est bien sûr la première des questions qu'on doit se poser quand on envisage cette théorie.

Cependant, ces énoncés ne sont simples que par leur sens : la non-contradiction d'une théorie est toujours assez complexe. On l'exprime souvent sous la forme : « Jamais une suite de formules obtenues en suivant les règles de raisonnement de la théorie ne conduira à la formule 0 = 1. » Or traduire une telle affirmation en un énoncé arithmétique exige qu'on traduise d'abord les règles de raisonnement de T, ce qui demande un travail délicat et long dénommé arithmétisation de la syntaxe et dont l'invention est due à Gödel.

Indécidabilité avec des énoncés simples ?

Pour évaluer la proximité de l'indécidabilité, les logiciens ont donc recherché des énoncés simples dont on puisse prouver l'indécidabilité. Récemment, de nouveaux résultats permettent d'évaluer le risque d'une rencontre avec un indécidable que certains mathématiciens considéraient comme improbable. Nous allons voir plus loin que l'énoncé concernant la fonction $s(n)$ de Radó, qui permettrait d'en connaître la valeur pour l'entier 1 919, est indécidable dans la puissante théorie classique des ensembles : $s(1\,919)$ est définitivement inconnaissable pour le mathématicien qui voudrait mener tout son travail avec la théorie classique des ensembles.

Pour expliquer ces nouveaux résultats, nous utiliserons les machines introduites par Alan Turing en 1936. Une machine de Turing est un automate qui peut se trouver dans un nombre fini d'états, par exemple l'état A, B ou C si ce nombre est 3. L'automate dispose d'un ruban découpé en cases. Sur le ruban, la tête de lecture-écriture de la machine lit et écrit des symboles qui viennent éventuellement effacer des symboles déjà inscrits. La machine n'a accès qu'à une seule case à la fois et déplace sa tête de lecture-écriture d'une case dans un sens ou dans l'autre. Elle parcourt ainsi le ruban pour y opérer un calcul.

Le programme de la machine est une liste finie d'instructions. Chacune est du type [A 0→1 droite B] qui signifie : « Quand je suis dans l'état A, que je lis 0 dans la case que je suis en train d'examiner, alors j'écris 1 dans cette case, je déplace la tête de lecture-écriture d'une case vers la droite et je passe dans l'état B. »

Le castor affairé à 3 états

Un castor affairé à *n* états est une machine de Turing à *n* états qui fonctionne le plus longtemps possible (pour une machine à *n* états) et qui s'arrête. Le programme du castor affairé à 3 états comporte 6 instructions. Il a été trouvé par Tibor Radó. Il écrit cinq « 1 » successifs et fonctionne 21 fois avant de s'arrêter. Aucune machine de Turing à 3 états ne fonctionne plus de 21 fois avant de s'arrêter, sauf si elle ne s'arrête jamais.

Les 6 instructions du programme du castor affairé à 3 états sont les suivantes :

[A 0→1 droite B],
[A 1→1 droite ARRÊT],
[B 0→1 gauche B],
[B 1→0 droite C],
[C 0→1 gauche C]
[C 1→1 gauche A].

Voici comment il faut lire ce programme et le faire fonctionner. La machine *(comme celle schématisée page ci-contre)* est posée sur un ruban recouvert de 0 doublement infini (vers la droite et vers la gauche). Supposons-la dans l'état A, et que sa tête de lecture-écriture lit un 0. L'instruction à appliquer est donc la première instruction [A 0→1 droite B] de son programme, qui indique que le 0 lu doit être remplacé par un 1, que la tête de lecture-écriture doit se déplacer d'une case vers la droite, et que le nouvel état de la machine doit être B. On a ainsi, en supposant que la machine débute sa lecture à la quatrième case et en marquant en rouge la position de la tête de lecture-écriture :

État initial
A - 000000000000000000000000000…,
État atteint
B - 000100000000000000000000000…
La machine est maintenant dans l'état B et lit un 0. Elle applique donc la 3e instruction [B 0→1 gauche B], ce qui conduit à la situation :
B - 000110000000000000000000000…

Le calcul se poursuit alors. Voici l'ensemble du calcul jusqu'à l'arrêt :

A - 000000000000000000000000000…
B - 000100000000000000000000000…
B - 000110000000000000000000000…
C - 000010000000000000000000000…
A - 000010000000000000000000000…
B - 000110000000000000000000000…
C - 000100000000000000000000000…
C - 000101000000000000000000000…
C - 000111000000000000000000000…
A - 000111000000000000000000000…
B - 001111000000000000000000000…
C - 001011000000000000000000000…
A - 001011000000000000000000000…
B - 001111000000000000000000000…
C - 001101000000000000000000000…
A - 001101000000000000000000000…
B - 001111000000000000000000000…
C - 001110000000000000000000000…
C - 001110100000000000000000000…
C - 001111100000000000000000000…
A - 001111100000000000000000000…
ARRÊT - 001111100000000000000000…

Considérons la machine M à 2 états A et B, et dont les instructions sont [A 0→0 droite A], [A 1→0 droite A] et [A 2→0 gauche B]. Quand on place la tête de lecture-écriture de M au début d'un ruban comportant par exemple 0101101020010..., alors M parcourt le ruban de gauche à droite en remplaçant les 1 par des 0, puis, arrivée au 2, elle recule d'une case et s'arrête, car il n'y a aucune instruction lui indiquant ce qu'elle doit faire quand elle se trouve dans l'état B.

Si on place cette machine sur un ruban infini ne comportant que des 0 et des 1, elle le parcourt vers la droite sans jamais s'arrêter en remplaçant les 1 par des 0. La machine de Turing peut ainsi s'engager dans un calcul infini.

L'état A dans lequel on place la machine au démarrage est l'« état de départ ». Un état comme l'état B de notre exemple, auquel ne correspond aucune instruction, est l'« état d'arrêt ».

Quand on compte les états d'une machine de Turing, on convient de ne pas compter l'état d'arrêt. La machine que nous avons donnée en exemple est, selon cette convention, une machine à un état. Sauf exception, nous ne considérerons que des calculs de machines de Turing commençant sur un ruban (potentiellement infini) entièrement couvert de 0, et nous supposerons que les machines n'utilisent que les deux symboles 0 et 1.

Les machines de Turing qui calculent le plus longtemps

Pour chaque entier positif *n*, il existe une machine de Turing à *n* états record, c'est-à-dire telle qu'aucune autre machine à *n* états n'utilisant que les symboles 0 et 1 ne calcule plus longtemps, sauf celles qui ne s'arrêtent jamais. Le nombre de mouvements de ce calcul record est noté *s*(*n*).

Les résultats connus à ce jour sont :
$s(1) = 1$; $s(2) = 6$; $s(3) = 21$ (Lin et Radó, 1965) ;
$s(4) = 107$ (Brady, 1975) ; $s(5) \geq 47\,176\,870$ (Marxen et Buntrock, 1990) ;
$s(6) \geq 7{,}4 \times 10^{36\,534}$ (Kropitz, 2010) ;

$s(7) > 10^{10^{10^{10^{10^{7}}}}}$ (Wythagoras, 2014).

Nous avons indiqué ci-dessous les programmes de quelques machines record, que l'on dénomme « castors affairés ». En écrivant un programme d'exécution sur ordinateur qui simule une machine de Turing, on vérifie facilement le nombre d'étapes (égal au nombre de déplacements de la tête de lecture-écriture de la machine de Turing).

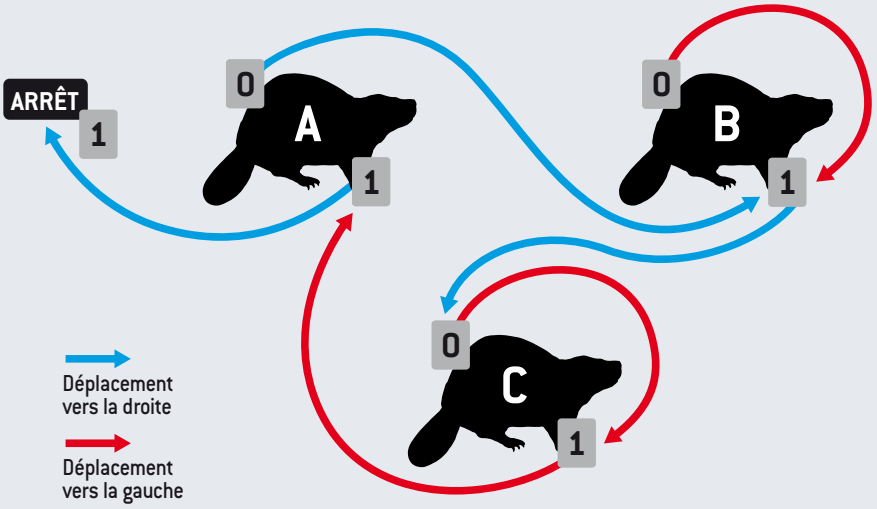
Le castor affairé à 2 états s'arrête en 6 étapes après avoir écrit quatre « 1 ». Son programme :
[A 0→1 droite B],
[A 1→1 gauche B],
[B 0→1 gauche A],
[B 1→1 droite ARRÊT].

Le castor affairé à 3 états s'arrête en 21 étapes après avoir écrit cinq « 1 ». Son programme :
[A 0→1 droite B],
[A 1→1 droite ARRÊT],
[B 0→1 gauche B],
[B 1→0 droite C],
[C 0→1 gauche C],
[C 1→1 gauche A].

Le castor affairé à 4 états s'arrête en 107 étapes après avoir écrit treize « 1 ». Son programme :
[A 0→1 droite B],
[A 1→1 gauche B],
[B 0→1 gauche A],
[B 1→0 gauche C],
[C 0→1 droite ARRÊT],
[C 1→1 gauche D],
[D 0→1 droite D],
[D 1→0 droite A].

Le castor affairé à 5 états, record actuel, s'arrête en 47 176 870 étapes après avoir écrit 4 098 « 1 ». Son programme :
[A 0→1 gauche B], [A 1→1 droite C], [B 0→1 gauche C],
[B 1→1 gauche B], [C 0→1 gauche D], [C 1→0 droite E],
[D 0→1 droite A], [D 1→1 droite D], [E 0→1 gauche ARRÊT],
[E 1→0 droite A].

En mai et juin 2016, une série de travaux ont conduit à la conclusion que le nombre *s*(1 919) était indécidable dans la théorie des ensembles. Cela signifie qu'aucun raisonnement mené dans cette théorie ne permet de démontrer la formule $s(1\,919) = k$ avec la bonne valeur de *k*, laquelle est pourtant parfaitement et clairement définie.



REPRÉSENTATION GRAPHIQUE de l'algorithme du castor affairé à 3 états.

© PLS/Cassandra Vion

On comprend sans mal qu'une machine simple, avec peu d'états et peu d'instructions, placée sur un ruban couvert uniquement de 0, va soit s'engager dans un calcul infini (par exemple parcourir le ruban en changeant les 0 en 1, ou osciller entre deux cases sans rien y faire), soit calculer un petit moment puis s'arrêter. Une machine simple produit soit un calcul infini, soit un calcul court, et on verra que c'est justement ce qui pose un défi aux mathématiciens et les expose aux formes les plus élémentaires d'indécidabilité.

En 1962, le mathématicien hongrois Tibor Radó s'est intéressé au nombre maximal, noté $s(n)$, de mouvements qu'exécute une machine de Turing à n états avant de s'arrêter (dans le cas où elle s'arrête).

Les castors affairés

Les machines M à n états qui atteignent ce maximum sont dénommées « castors affairés ». Le nombre $s(n)$ est un nombre parfaitement défini, puisqu'il n'y a qu'un nombre fini de machines à n états n'utilisant que les symboles 0 et 1. Le nombre de machines à n états a d'ailleurs été déterminé : il est égal à $(4n+4)^{2n}$.

Pour calculer $s(n)$ pour un n donné, il suffit de faire fonctionner toutes les machines à n états en repérant celles qui calculent le plus longtemps et s'arrêtent. Avec les incroyables ordinateurs dont nous disposons, il semblerait facile d'obtenir rapidement les valeurs de $s(n)$.

Grave erreur ! Aujourd'hui, malgré de considérables efforts, on ne connaît que $s(1) = 1$, $s(2) = 6$, $s(3) = 21$ et $s(4) = 107$. On sait aussi que $s(5) \geqslant 47\,176\,870$, et que $s(6) \geqslant 7{,}4 \times 10^{36\,534}$, mais personne n'est certain que ces valeurs sont définitives. Pourquoi est-il si difficile d'évaluer $s(n)$?

La raison est double. D'abord, le nombre de machines à n états augmente très rapidement quand n augmente et on ne sait guère faire mieux, pour calculer $s(n)$, que de les prendre une à une. Ensuite, et c'est encore plus ennuyeux, les machines qui ne s'arrêtent pas sont délicates à traiter. Il faut en effet raisonner sur leurs instructions ou leur comportement pour être sûr qu'elles ne s'arrêtent pas. Pour certaines, c'est facile, mais on tombe rapidement sur des machines dont le comportement est complexe et dont on n'arrive pas à déterminer si elles s'arrêteront et donc si elles interviennent pour la détermination de $s(n)$, ou si elles poursuivent indéfiniment leurs calculs – auquel cas elles sont sans importance pour $s(n)$. On sait par ailleurs qu'aucune méthode algorithmique générale ne permet de répondre à la question. C'est la fameuse indécidabilité de l'arrêt démontrée par Turing en 1936 : aucun algorithme ne peut déterminer, pour toute machine de ce type placée sur un ruban ne comportant que des 0, si oui ou non elle finira par s'arrêter.

Une conséquence de ce résultat, qui est la raison de l'introduction par Radó de sa fonction $s(n)$, est que $s(n)$ n'est pas calculable par algorithme. Plus intéressant

Les castors affairés calculent l'entropie

La connaissance des valeurs de la fonction $s(n)$ de Radó, dont il vient d'être démontré que $s(1\,919)$ est indécidable, est utile pour certains calculs fondamentaux de la théorie de la complexité. En utilisant la valeur conjecturée comme la bonne pour $n = 5$, $s(5) = 47\,176\,870$, une équipe réunie autour de Fernando Soler-Toscano a calculé l'entropie algorithmique, qui généralise l'entropie statistique de Shannon, des suites courtes de 0 et de 1.

Cette entropie est aussi nommée complexité de Kolmogorov : pour une suite donnée, c'est la taille du plus court programme qui produit la suite. Plus une suite est complexe, plus son entropie algorithmique est grande *(voir l'article indiqué en bibliographie de F. Soler-Toscano et al.)*. La méthode est la suivante. On fait calculer chacune des 26 559 922 791 424 (26 000 milliards) machines de Turing à 5 états jusqu'à ce qu'elle s'arrête où jusqu'à l'étape $s(5)$, ce qui signifie que la machine ne s'arrête jamais. Le nombre k de fois où est obtenue une suite donnée (par exemple la suite $t = 01010$) est directement lié à son entropie algorithmique $K(t)$ par une formule de Leonid Levin, $K(t) \approx -\log_2[k/N(5)]$, où $N(5)$ désigne le nombre de machines à 5 états, et donne donc une valeur approchée de l'entropie.

La formule de Leonid Levin est remarquable et très profonde. Elle signifie que les suites complexes (c'est-à-dire celles dont le plus court programme de création est long) sont aussi celles qu'une machine tirée au hasard produit avec la plus faible probabilité.

Quelques valeurs ainsi obtenues sont présentées ci-dessous.

SUITE	FRÉQUENCE	ENTROPIE
1	0,175036	2,51428
0	0,175036	2,51428
11	0,0996187	3,32744
10	0,0996187	3,32744
01	0,0996187	3,32744
00	0,0996187	3,32744
111	0,0237546	5,3962
000	0,0237546	5,3962
110	0,0229434	5,44578
100	0,0229434	5,44578
011	0,0229434	5,44578
001	0,0229434	5,44578
101	0,0220148	5,5038
010	0,0220148	5,5038
1111	0,0040981	7,03083
0000	0,0040981	7,03083
1110	0,00343136	8,187
1000	0,00343136	8,187
0111	0,00343136	8,187
0001	0,00343136	8,187

Des machines pour Goldbach et Riemann

Tout comme on peut envisager d'écrire explicitement une machine de Turing qui recherche une contradiction dans la théorie des ensembles ZFC et qui n'en trouvera pas, on peut expliciter une machine de Turing qui recherche un contre-exemple à la conjecture de Goldbach, c'est-à-dire un entier pair supérieur à 2 qui ne s'écrit pas comme somme de deux nombres premiers.

Une telle « machine de Goldbach » ne s'arrête que si la conjecture de Goldbach est fausse. Le nombre d'états de cette machine de Turing qu'on tente de rendre le plus petit possible est une mesure de la difficulté de la conjecture. Le record obtenu tout récemment est 47. Une machine à 27 états est en cours de vérification (www.scottaaronson.com/blog/?m=201605).

L'hypothèse de Riemann est une conjecture centrale de la théorie des nombres, formulée en 1859 par le mathématicien allemand Bernhard Riemann. Elle énonce que les zéros non triviaux de la fonction zêta de Riemann ont tous pour partie réelle 1/2 et donne des précisions sur la densité des nombres premiers.

On recherche des machines de Turing qui ne s'arrêtent que si la conjecture est fausse. La plus petite connue aujourd'hui est à 744 états. Si la conjecture de Riemann est indécidable dans ZFC (c'est une éventualité parfois envisagée), on pourra alors affirmer que $s(744)$ est indécidable dans ZFC, ce qui serait un progrès.

De même, si la conjecture de Goldbach se révélait indécidable dans ZFC, on saurait alors que $s(47)$ est indécidable dans ZFC.

encore, cette fonction croît plus rapidement que n'importe quelle fonction calculable par algorithme.

Voici le raisonnement de Radó. Soit $f(n)$ une fonction calculable par algorithme dont on suppose qu'elle majore $s(n)$, c'est-à-dire que pour tout entier n, $f(n) \geqslant s(n)$. En l'utilisant, on pourrait savoir par algorithme si une machine donnée M à n états s'arrête ou non. En effet, on calcule $f(n)$, puis on calcule $f(n)$ étapes du fonctionnement de M ; si M s'est arrêtée pendant ces $f(n)$ étapes, c'est qu'elle s'arrête, sinon c'est qu'elle ne s'arrête jamais, par définition de $s(n)$. Une fonction calculable par algorithme et qui majore $s(n)$ ne peut donc pas exister. Autrement dit : pour toute fonction $f(n)$ calculable par algorithme, il existe un n tel que $f(n) < s(n)$. Sans beaucoup d'effort, on montre même qu'il existe une infinité de n tels que $f(n) \leqslant s(n)$, et en outre que c'est le cas pour tous les entiers à partir d'une certaine valeur.

Voyons maintenant le rapport entre cette fonction de Radó et la théorie des ensembles, ce qui nous mènera aux nouveaux indécidables courts découverts il y a quelques mois.

Des indécidables grâce à la fonction de Radó

La théorie des ensembles permet de représenter pratiquement tous les objets mathématiques et tous les raisonnements qu'on fait avec eux. Sa version la plus utilisée (par exemple dans le traité *Éléments de mathématique,* de Nicolas Bourbaki) est notée ZFC, pour « Zermelo-Fraenkel avec axiome du choix » (Ernst Zermelo et Abraham Fraenkel sont deux mathématiciens qui ont contribué à sa définition). L'axiome du choix stipule que pour toute famille **F** d'ensembles non vides et disjoints deux à deux, par exemple $\{\{a\}, \{1, 2, 3\}, \{X, Y\}, \{p, q, r, s\}\}$, il existe un ensemble C de « choix » contenant un élément exactement de chaque ensemble de **F** ($C = \{a, 2, X, r\}$ conviendrait pour notre exemple). Depuis un siècle qu'on utilise ZFC, on n'y a jamais trouvé de contradiction et nous supposerons qu'il n'y en a pas.

On montre que lorsqu'une fonction $f(n)$ n'est pas calculable par algorithme, la théorie des ensembles ZFC ne peut pas en connaître certaines valeurs. Autrement dit il existe au moins un entier n tel que l'énoncé $f(n) = k$ qui en donne la valeur est un indécidable de ZFC. C'est vrai aussi pour n'importe quelle autre théorie non contradictoire utilisée à la place de ZFC, mais bien sûr pas nécessairement avec le même entier n.

Concernant la fonction $s(n)$ de Radó, on sait donc depuis 1962 qu'avec la théorie ZFC, il existe au moins un entier n_0 tel qu'on ne peut déterminer la valeur de $s(n_0)$: l'énoncé $s(n_0) = k$, pour la bonne valeur de k, n'est pas démontrable dans ZFC, et sa négation non plus. La théorie ZFC est impuissante pour $s(n_0)$, qui échappe à son pouvoir déductif.

La question que se sont récemment posée Scott Aaronson, professeur au MIT, et Adam Yedidia, l'un de ses étudiants, est la suivante : est-ce que ce n_0 qui marque l'impuissance de ZFC vaut 10, 1 000, 100 milliards ou autre chose ? Répondre en proposant une valeur aussi petite que possible de n_0 est une façon parfaitement concrète de mesurer la proximité des indécidables de ZFC.

Le résultat auquel ils sont parvenus est que $s(7\,918)$ échappe à ZFC : jamais un mathématicien travaillant avec la théorie classique des ensembles ne pourra connaître $s(7\,918)$. Dit autrement, parmi les machines de Turing à 7 918 états qui ne s'arrêtent pas, certaines le font pour des raisons que la théorie n'est pas capable d'analyser. C'est la première fois qu'on réussit à avoir une précision de ce type sur l'indécidabilité dans la théorie des ensembles.

Chose amusante, ce travail, rendu public en mai 2016, a aussitôt suscité d'autres travaux sur la même question, lesquels ont abouti à son amélioration. Stefan O'Rear a établi dans un premier temps que $s(5\,349)$ échappe à ZFC, résultat encore amélioré quelques jours plus tard pour arriver cette fois à la démonstration que $s(1\,919)$ est hors d'atteinte de ZFC.

Désormais, personne ne pourra plus dire que les indécidables de la théorie des

ensembles ne concernent que des problèmes alambiqués et si complexes qu'aucun mathématicien ne s'y intéressera jamais de lui-même : s'interroger sur $s(1\,919)$ est naturel, car cela concerne la partie la plus élémentaire de la théorie mathématique du calcul…

Notons que ce record $n_0 = 1\,919$ est sans doute améliorable, car, comme le savent bien ceux qui tentent de connaître les valeurs de $s(n)$, les difficultés concrètes commencent dès $s(5)$, qui reste inconnu à ce jour. Une bonne marge de travail est donc disponible entre 5 et 1 919, marge que l'on réduira soit en progressant dans le calcul des $s(n)$, soit en réussissant à prouver l'indécidabilité d'énoncés de la forme $s(n) = k$ pour des valeurs encore plus petites que $n_0 = 1\,919$.

La machine Z ne s'arrête jamais

La méthode utilisée par Adam Yedidia et Scott Aaronson est intéressante. L'idée la plus naturelle serait de mettre en œuvre les techniques d'arithmétisation de la syntaxe de Gödel et donc de construire une machine de Turing M qui recherche une contradiction dans ZFC en énumérant toutes les démonstrations de ZFC et en tentant d'y reconnaître une preuve de l'énoncé « 0 = 1 ». Une telle machine ne trouvera pas de contradiction, et pourtant ZFC ne pourra pas le démontrer (sinon, cela signifierait que ZFC démontre qu'elle n'est pas contradictoire, ce que le second théorème d'incomplétude de Gödel interdit). Cependant, cette méthode par l'arithmétisation de la syntaxe conduirait à une machine de Turing à plusieurs centaines de milliers d'états, voire des millions, et l'on ne démontrerait donc l'indécidabilité de $s(n) = k$ que pour un n très grand.

La méthode utilisée par Scott Aaronson et Adam Yedidia évite cette explosion du nombre d'états de la machine de Turing dont ZFC ne peut prouver le non-arrêt. Elle se fonde sur des travaux récents de Harvey Friedman, de l'université d'État de l'Ohio, qui élabore depuis des années des énoncés d'arithmétiques indécidables dans ZFC ou d'autres théories. L'un d'eux concerne une propriété des graphes et cette propriété, trop abstraite pour être décrite ici, est équivalente à celle exprimant la non-contradiction de ZFC. Cette propriété s'écrit assez facilement en termes arithmétiques et la machine, notée Z, qui cherche un contre-exemple à cet énoncé est donc relativement simple comparée à celle qu'on aurait obtenue par arithmétisation de la syntaxe : elle énumère des séries d'entiers et effectue des tests sur eux. Le fait que Z ne s'arrête pas dans cette recherche de contre-exemple est nécessairement un indécidable de ZFC, sinon, encore une fois, ZFC démontrerait d'elle-même qu'elle est non contradictoire.

Soit n_0 le nombre d'états de Z. Puisque pour connaître $s(n_0)$, ZFC doit savoir prouver le non-arrêt de chaque machine à n_0 états qui ne s'arrête pas, il en résulte que ZFC ne peut pas démontrer $s(n_0) = k$ pour la bonne valeur k. Précisons que la propriété en question des graphes est démontrable dans une théorie un peu plus forte que ZFC, et donc qu'on a toutes les raisons de la croire vraie. Notons aussi qu'une fois leur machine Z programmée, Scott Aaronson et Adam Yedidia l'ont effectivement lancée pour voir si elle s'arrêtait ; au bout de plusieurs heures, elle ne s'était pas arrêtée.

Sans effort particulier, utiliser ce raccourci déduit des travaux de Friedman n'aurait cependant pas permis à lui seul d'arriver aux petites valeurs de n_0 aujourd'hui connues. Scott Aaronson et Adam Yedidia ont, et c'est la seconde idée originale de leur travail, conçu un langage informatique nommé Laconic permettant d'exprimer la propriété de Friedman de manière concise. Ils ont aussi écrit et utilisé le traducteur des programmes écrit en Laconic qui en tire des machines de Turing du type considéré par Radó. C'est ainsi qu'ils ont obtenu la machine de Turing Z à 7 918 états qui ne s'arrête jamais, mais dont ZFC ne peut pas démontrer qu'elle ne s'arrête jamais. Le passage de 7 918 états à 1 919 états s'est fait en améliorant le langage et les opérations de transformation des programmes en machines de Turing.

Saura-t-on faire encore mieux et trouver des valeurs encore plus petites que le $n_0 = 1\,919$ d'aujourd'hui ? C'est probable, et peut-être qu'à l'heure où vous lirez ce texte, le record sera déjà battu. Pour le savoir, consultez le blog de Scott Aaronson ! ■

L'AUTEUR

J.-P. DELAHAYE est professeur émérite à l'université de Lille et chercheur au Centre de recherche en informatique, signal et automatique de Lille (CRISTAL).

BIBLIOGRAPHIE

S. Aaronson, **The 8000th Busy Beaver number eludes ZF set theory : New paper by Adam Yedidia and me**, blog de mai 2016, www.scottaaronson.com/blog/?m=201605

H. Marxen, **List of the known Busy Beaver values**, 2016, www.drb.insel.de/ffheiner/BB/

P. Michel, **The Busy Beaver competition : A historical survey**, 2016, http://export.arxiv.org/pdf/0906.3749v4

A. Yedidia et S. Aaronson, **A relatively small Turing machine whose behavior is independent of set theory**, 2016, https://arxiv.org/pdf/1605.04343.pdf

F. Soler-Toscano *et al.*, **Calculating Kolmogorov complexity from the output frequency distributions of small Turing machines**, *PLoS One*, vol. 9(5), e96223, 2014.

C. S. Calude et E. Calude, **Evaluating the complexity of mathematical problems : Part 1 & Part 2**, *Complex Systems*, vol. 18, pp. 267-285, 2009 et pp. 387-401, 2010.

Retrouvez la rubrique Logique & calcul sur www.pourlascience.fr