Colloque en l'honneur de René Cori

A propos de la théorie des démonstrations

(du programme de Hilbert aux programmes tout court)

Jean-Louis Krivine

Paris, 12 septembre 2014

Comme vous le savez bien (on est ici entre logiciens), les autres mathématiciens n'aiment pas beaucoup entendre parler de logique, et c'est principalement à cause de la *théorie des démonstrations*. Ils pensent qu'on prétend leur apprendre leur métier, en les assommant de chinoiseries et de détails sans intérêt.
On aurait pu croire que cette aversion allait disparaître peu à peu, mais c'est l'inverse qui s'est produit : maintenant, y compris pour la plupart des logiciens (théoriciens des modèles ou des ensembles), la théorie des démonstrations relève du pinaillage et de la scolastique. Quelle drôle d'idée de vouloir interdire l'emploi de l'axiome du choix ou, pire encore, du raisonnement par l'absurde ?

Ce que je vais vous raconter maintenant, n'est pas l'histoire de la théorie des démonstrations, mais plutôt celle de mon itinéraire dans ce domaine. Comme tous les mathématiciens « normaux », j'ai commencé par être méfiant et plutôt hostile. Mais, à cause de mon intérêt pour l'informatique, j'ai fini par en comprendre l'utilité, au point d'en faire maintenant mon sujet principal de recherches. Et, comme tout théoricien de la démonstration qui se respecte, j'en suis venu à m'imposer une nouvelle nuisance, inconnue jusqu'alors : abandonner l'axiome d'extensionnalité ! Rassurez-vous, on n'en parlera pas ici.

La logique est née parce qu'on s'est posé des questions très naturelles : qu'est-ce que les mathématiques ? et tout ce qui s'ensuit : pourquoi faut-il des démonstrations ? pourquoi joue-t-on à ce jeu bizarre, et pour commencer, quelles sont ses règles ?
En tous cas, c'est pour cela que je m'y suis intéressé.

Mais les mathématiciens ont toujours été comme des drogués à un jeu en réseau (je fais partie des intoxiqués). Leur dépendance est telle que, pendant des siècles, ils y ont joué sans règles, par consensus. Il a fallu une catastrophe pour qu'ils commencent à se poser des questions. La catastrophe est apparue justement quand on a voulu formaliser les règles du jeu : ce sont les *paradoxes* qui amènent des contradictions, *qui empêchent de continuer à jouer*. Alors là, ça ne peut plus durer, l'état de manque commence à se faire sentir, vite un remède, qu'on oublie ce malheureux contretemps et qu'on reprenne le jeu !

Les solutions qu'on trouve sous l'empire de l'urgence sont souvent des aberrations : la *Théorie des ensembles* de Bourbaki en est un bon exemple.

Le programme de Hilbert

D. Hilbert a pris les choses plus sérieusement, et proposé un programme de travail qui aboutirait à *garantir* que de pareilles horreurs ne se produisent *plus jamais*. L'objectif était toujours le même : pouvoir nous livrer comme avant à notre jeu favori mais avec, en plus, l'esprit tranquille.
Mais, en admettant qu'on y arrive, c'en est quand même fini de la tranquillité d'esprit, car le choc a été trop grand et nos yeux se sont ouverts (au moins pour certains). On va continuer à jouer, bien sûr, on ne peut pas faire autrement, dépendance oblige. Mais on voudra maintenant comprendre la raison de ce jeu et de la force irrésistible qui nous pousse à y jouer.

Le programme de Hilbert consistait à :

i) Enoncer précisément les règles du jeu ; c'est ce qui a été fait, avec le *théorème de complétude* (Gödel, 1930) et *l'axiomatique de la théorie des ensembles* (Zermelo, 1908 ; Frænkel et Skolem, 1922). Le jeu consiste à appliquer les règles de déduction aux axiomes de l'arithmétique ou de la théorie des ensembles.
ii) Démontrer, de façon *finitiste*, qu'on n'arrive jamais à une contradiction. La notion de démonstration finitiste, ou combinatoire, est assez claire intuitivement ; ça veut dire qu'on n'a pas le droit d'utiliser des ensembles infinis.

Une fois ce programme rempli, on pourrait ensuite utiliser sans crainte, les axiomes et les divers objets mathématiques plus ou moins exotiques (axiome du choix, ensembles infinis plus ou moins grands, nombres réels, ...).
Il faut bien dire que le recours à l'« intuition » pour justifier toutes ces notions bizarres n'est vraiment pas convaincant ; par exemple, l'axiome du choix et ses conséquences invraisemblables, ou même l'existence de diverses tailles d'infini, n'ont absolument rien d'intuitif. Comme l'a dit Jerry Bona : *The axiom of choice is obviously true, the well-ordering principle obviously false, and who can tell about Zorn's lemma ?*
Pour Hilbert, toute ces notions abstraites ne sont que des représentations intuitives, permettant de manipuler commodément des structures finies, qui sont les preuves formelles.

Et c'est alors, qu'après le choc des paradoxes, le coup de grâce a été porté par le *théorème d'incomplétude* (Gödel, 1931), qui a anéanti tout espoir d'obtenir un quelconque résultat de non contradiction. C'en était trop pour la plupart des mathématiciens, qui se sont alors définitivement désintéressés de la logique, puisqu'elle est incapable de remplir son rôle d'assurance contre les paradoxes.

Mais pour nous, les logiciens, pas question de se laisser décourager si facilement ; il faut continuer à essayer de comprendre le jeu mathématique. Qu'est-ce donc qui ne va pas, dans le programme de Hilbert ?
En réalité, son gros défaut est le manque d'ambition : il ne cherche pas à *expliquer* le jeu mathématique, mais seulement à le *sécuriser*. Or maintenant, nous ne voulons plus jouer sans comprendre pourquoi, même si le jeu est toujours aussi addictif, et justement parce qu'il l'est.

Il reste, tout de même, l'idée très intéressante et juste, que derrière les notions et axiomes follement abstraits, se cache une manipulation concrète de structures finies. Mais quels sont, au juste, ces objets finis ?
Hilbert a cru qu'il s'agissait des formules mathématiques avec leur syntaxe et leurs règles de

démonstration. C'était naturel, mais c'est pourtant là-dessus qu'il s'est trompé ; on ne peut guère le lui reprocher car, à cette époque, ces mystérieux objets n'avaient pas encore été découverts.
Pourtant, sous ses yeux, à ce moment précis, en 1929, à Göttingen même, quelqu'un était en train de les découvrir. C'était un jeune américain, étudiant en thèse de D. Hilbert et de P. Bernays, qui s'appelait Haskell B. Curry. Mais on était encore loin de faire le rapprochement, qui n'est venu que trente ans après, sous le nom de *correspondance de Curry-Howard*.

L'intuitionnisme et la correspondance de Curry-Howard

Pour comprendre la découverte de cette correspondance, il nous faut revenir en arrière, en 1923, au moment où J. Brouwer publie un article où il met en doute la validité de l'axiome du *tiers exclu*. C'est le début de *l'intuitionnisme*, dont les règles de déduction seront ensuite formalisées par A. Heyting. Vu de l'extérieur, on a l'impression que, pour résoudre les problèmes de la logique, on jette le bébé avec l'eau du bain. Pour le mathématicien normal, il est absolument hors de question de se priver du raisonnement par l'absurde ! Une raison de plus pour rejeter la théorie des démonstrations.

Mais la logique intuitionniste va vraiment montrer son importance à partir des années 1950, dans un tout autre domaine, avec le développement exponentiel de l'informatique et l'apparition des *langages de programmation*.

En 1958, H. Curry découvre l'analogie formelle entre :
– d'une part, les preuves à partir des axiomes de la logique propositionnelle *intuitionniste* ;
– d'autre part, la *logique combinatoire* qu'il a développée et qui est, en fait, le premier prototype de langage de programmation, une sorte de *langage machine universel*.
Puis, en 1969, William Howard observe la même analogie entre :
– d'une part, les règles de la *déduction naturelle intuitionniste* de Gentzen ;
– d'autre part, le *lambda-calcul*, inventé en 1936 par A. Church, qui est un langage de programmation très voisin de la logique combinatoire, mais un peu plus évolué. Notons que le LISP, qui est l'un des tous premiers langages de programmation, dérive explicitement du lambda-calcul.
La *correspondance de Curry-Howard*, appelée aussi *correspondance preuves-programmes*, venait de faire son apparition.
Elle a eu une influence énorme sur la théorie de la démonstration et réciproquement, on peut en espérer d'importantes applications en informatique, dont je ne parlerai pas ici.

Ce sont des logiciens qui l'ont découverte, tout au moins en ce qui concerne la *logique intuitionniste*. Mais c'est un informaticien, T. Griffin, qui a opéré son extension à la *logique classique*, trente ans après, en 1990. C'est, de nouveau, une découverte capitale et tout à fait inattendue : en effet, le raisonnement par l'absurde est associé à une instruction très sophistiquée du langage SCHEME (une variante de LISP), qui a été inventée pour gérer, entre autres, les exceptions et le multi-tâches.
Il est tout à fait clair qu'aucun logicien ne pouvait avoir cette idée, qui nécessite la pratique approfondie d'un langage de programmation. Elle utilise la notion d'*environnement*, qui est essentielle en programmation, mais inconnue jusqu'ici en logique. L'instruction associée au tiers exclu est destinée à *sauvegarder, puis rétablir, l'environnement*. Son équivalent, dans le

langage C, est indispensable en programmation système et réseau.

Si l'on y réfléchit une minute, c'est proprement stupéfiant : le raisonnement par l'absurde est le pain quotidien du mathématicien depuis deux mille ans. Mais alors, Euler, Fermat, Euclide même, se préoccupaient donc d'exceptions, de système et de réseau ?
Il n'y a qu'une seule explication possible, et je reviendrai tout à l'heure là-dessus.

L'intuitionnisme a donc été un ingrédient essentiel dans l'élaboration de la correspondance preuves-programmes et on admire l'intuition de J. Brouwer qui a pressenti, en 1920, la spécificité extraordinaire du tiers exclu parmi tous les axiomes logiques. Il était absolument nécessaire de le mettre à part, ce que Brouwer a fait de façon peut-être un peu radicale, en prononçant son excommunication.

Toutefois, je comparerai le rôle de la logique intuitionniste à celui de l'échafaudage dans la construction d'une voûte ; une fois la clé de voûte en place (ici, c'est la contrepartie informatique du tiers exclu), il faudrait penser à retirer l'échafaudage. Mon avis est que la logique intuitionniste a bien rempli son rôle et que, maintenant, je peux la mettre de côté.

Les programmes

Comme je vous l'ai dit au début, il s'agit là de mon itinéraire personnel ; ce n'est pas du tout de ce qui se passe en réalité, il n'est pas question d'oublier la logique intuitionniste. En effet, c'est sur elle que sont centrées pratiquement toutes les recherches actuelles sur la correspondance de Curry-Howard : l'école suédoise, avec la théorie de Martin-Löf et les divers systèmes d'aide à la preuve qui lui sont associés ; l'école française avec la logique linéaire, le calcul des constructions et le système Coq ; les écoles anglaise, hollandaise et allemande, avec la théorie des topos ; et, bien sûr, plein d'interactions entre tout ce monde. L'intuitionnisme a donc de beaux jours devant lui.

Mais revenons à nos moutons, ou plutôt donc, à *mes* moutons : la transformations des preuves en programmes *pour les mathématiques classiques*. C'est maintenant acquis, pour la partie des mathématiques que l'on peut formaliser dans ZF + DC (théorie des ensembles avec axiome du choix dépendant), c'est-à-dire un immense domaine. L'axiome du choix complet fait encore un peu de résistance.

Derrière chaque théorème $\mathscr{T}$ se cache donc une classe de programmes, qui correspondent à ses diverses démonstrations. Ces programmes ont un comportement commun, qui est la *spécification* associée au théorème $\mathscr{T}$. Un exemple usuel est le théorème d'Euclide : *il existe une infinité de nombres premiers*. Les programmes associés calculent, pour chaque entier n, un nombre premier $> n$.

Mais, contrairement à ce qui se passe pour cet exemple, il est en général très difficile de trouver la spécification associée à un théorème donné.
Par exemple, la spécification associée au théorème $(\neg A \to A) \to A$ (le raisonnement par l'absurde) est de *sauvegarder l'environnement*. C'est la découverte de T. Griffin et c'est tout sauf évident.

A propos du programme de Hilbert, je vous ai dit que les notions abstraites et dangereuses, comme les ensembles infinis, l'axiome du bon ordre, les cardinaux inaccessibles ou mesurables, ... , servaient en fait, à manipuler certaines structures finies. Comme vous l'avez maintenant compris, ces structures finies, ce sont *les programmes* ; et nous connaissons le

langage machine dans lequel ces programmes sont écrits d'origine : c'est la logique combinatoire de Curry, revue et augmentée par Griffin.
Nous comprenons mieux, maintenant, la nature de notre jeu mathématique, si mystérieux et si addictif : nous manipulons des programmes. Mais, attention, ce n'est pas de la programmation, puisque nous écrivons des programmes *sans connaître leur spécification*, et que la partie la plus difficile (donc la plus plaisante) du jeu consiste justement à la trouver.
En informatique, cela s'appelle *désassemblage*, parce que le langage de plus bas niveau, le plus proche de la machine, s'appelle *assembleur* ou *langage d'assemblage*. Et il y a un assembleur différent pour chaque machine, c'est-à-dire chaque type de processeur : par exemple, les x86 d'Intel, utilisés dans les PC et les Mac, les PowerPC d'IBM, utilisés dans les anciens Mac, les divers ARM dans les téléphones mobiles, . . . , et le vénérable MOS 6502, présent dans beaucoup d'ordinateurs des années 80, pour lequel j'ai une affection particulière.

Quelle est donc la machine dont le langage d'assemblage est la logique combinatoire de Curry-Griffin ? Je vous laisse réfléchir à cette question, mais voici une remarque pour vous aider.

Pourquoi chercherait-on à désassembler des programmes ? Le but est de comprendre leur fonctionnement, pour les utiliser plus efficacement et éventuellement, arriver à les modifier et les mettre à jour. On peut aussi chercher des failles dans le programme, pour les exploiter dans un but plus ou moins honorable ; par exemple, le « jailbreak » des téléphones d'Apple.
Il serait évidemment beaucoup plus facile de s'adresser à l'auteur, pour avoir les *sources* (c'est-à-dire le texte du programme, mais *en langage évolué*). Malheureusement, bien souvent, il ne veut pas coopérer (c'est le cas d'Apple), ou alors il a tout simplement disparu.
Par exemple, vous pourriez vous proposer de désassembler le système d'exploitation de votre ordinateur, Unix ou Windows. Je vous le déconseille fortement (surtout que, au moins pour Unix, les sources sont à votre disposition) ; mais si vous insistez, sachez qu'il vous faudra constituer une équipe et travailler quelques dizaines, ou même quelques centaines d'années, en repassant donc la tâche aux générations suivantes.
Mais au fait, c'est exactement ce que font les mathématiciens ; de nouveau, je vous laisse deviner à quel ordinateur et à quel système d'exploitation ils se sont attaqués, les malheureux ! C'est autre chose que Windows et Unix ; et en l'occurrence, il n'y a vraiment personne à qui demander les sources.

Selon les divers domaines des mathématiques, il y a une sorte de distribution des rôles : par exemple, les géomètres cherchent sans doute à désassembler le pilote de la caméra embarquée. Et bien sûr, j'ai tendance à attribuer à la logique et à la théorie des ensembles le rôle central, à savoir ce qu'on appelle le *noyau* du système d'exploitation : ordonnancement des tâches, gestion de la mémoire, système de fichiers, . . . Et aussi, la *programmation réseau*, un domaine dont l'importance ne vous échappera pas.

Je pense qu'il y a là l'explication de notre addiction au jeu mathématique. Que peut-il, en effet, y avoir de plus amusant, que de décrypter les programmes qui nous font fonctionner ? et d'y trouver des ressemblances troublantes avec ceux au milieu desquels nous sommes plongés, depuis l'avènement de la *société de l'information*.

Par contre, j'ai toujours trouvé inintéressantes et même assez ridicules, les discussions philosophiques sur la vérité en mathématique, sur la nature et la réalité des objets mathématiques, la caverne de Platon, etc. La réponse est, en effet, toute simple : puisque ces objets

sont des programmes, ils sont bien réels, mais ils n'ont pas de support matériel, car un programme, c'est de l'information qui peut être recopiée n'importe où ; d'ailleurs, l'une des qualités essentielles d'un langage de programmation est sa *portabilité* ; ce qui veut dire que les programmes écrits dans ce langage peuvent être non seulement copiés, mais *exécutés* dans toutes sortes d'environnements.
Il n'y a rien de mystérieux dans tout cela. Quant à savoir si les démonstrations fournissent des énoncés « vrais », on voit bien que la question n'a aucun sens : on ne peut pas dire d'un programme qu'il est vrai ou faux, il s'agit seulement de savoir s'il fonctionne ou non. La seule raison d'être des démonstrations mathématiques est tout simplement de fournir des programmes corrects, et pas du tout d'établir je ne sais quelles « vérités universelles ».
Pourquoi $2+2=4$ est-il « vrai » ? parce que, si vous programmez en partant de $2+2=3$, l'exécution de vos programmes va tourner au désastre.
Or, les programmes à l'œuvre dans notre cerveau sont *corrects*, car l'évolution et la sélection naturelle y ont veillé jalousement. Elles ont donc veillé à ce que $2+2=4$ soit implicitement présent dans l'environnement de programmation. C'est pourquoi le mathématicien, qui ne fait que lire et désassembler ces programmes, va y découvrir, émerveillé, une démonstration de $2+2=4$.
On peut dire la même chose du théorème de Pythagore, mais aussi de la théorie des fonctions de variable réelle ou complexe, des probabilités, etc. Et également des grandes théories physiques, comme la loi de la gravitation de Newton, les équations de Maxwell pour l'électromagnétisme, la relativité générale, la mécanique quantique, ...

Malgré toutes ces découvertes dont nous sommes si fiers, il faut reconnaître que nous n'avons réussi à décoder qu'une infime partie de notre système. Pensez simplement, par exemple, à l'ensemble des programmes qui nous permettent de marcher en évitant les obstacles. Cela dépasse déjà très largement, en complexité, toutes les théories mathématiques et physiques dont je viens de parler. Il y a donc pas mal de travail en perspective.

Ma conclusion vient très naturellement, après tout ce que je vous ai raconté : je pense qu'avec le développement de la correspondance preuves-programmes, la théorie des démonstrations est devenue le domaine le plus intéressant de la logique ; que dis-je, de la logique, c'est le domaine le plus captivant des mathématiques. Mais non, ce n'est pas encore ça, je voulais dire : le domaine scientifique le plus passionnant, toutes disciplines confondues.