M1 Calculabilité avancée : théorèmes de Gödel et calculabilité

Kévin Perrot – Aix Marseille Université – printemps 2021

Table des matières

1 Théorèmes d'incomplétude de Gödel

Sources : surtout [1], et aussi [2, 3, 4, 5, 6, 7].

Mise en garde : les développements qui suivent sont plein de subtilités, et on leur fait rapidement « dire » plus de qui est démontré. Ce sont des mathématiques qui parlent des mathématiques : des « métamathématiques » (?).

1.1 Un peu d'histoire (début du XXe siècle)

Source : bande-dessinnée *Logicomix* [2], très chaleureusement recommandée !

En 1900 les mathématiciens se posent la question [2, page 105 case -1]

« qu'est-ce que les mathématiques ? »

— The map of mathematics (`https://njbiblio.com/tag/computer-science/`).
— Idée : fondements = axiomatisation. Russell-Whitehead [2, page 106 case 1].
— Hilbert est convaincu que « tout ce qui peut être énoncé rigoureusement, peut être répondu logiquement » [2, page 142].
— Exemple de difficulté : le paradoxe de Russell [2, page 152 et page 159 case 2].
— Les *Principia Matematica* : redéfinir toutes les mathématiques à partir de la théorie des ensembles (`https://fr.wikipedia.org/wiki/Principia_Mathematica#Preuve_de_1+1=2`).

Une nouvelle question émerge alors [2, page 258 case 1]

« Peut-on démontrer tous les théorèmes vrais ? »

— Hilbert est toujours convaincu que « oui » [2, page 267 case -1].
— Et Gödel démontre que « non » [2, de page 269 cases -4 à page 270 case 2].

1.2 Définitions et énoncés

Définition 1. *Un* **système formel** *est un donné par un* langage *(les énoncés que l'on peut formuler), un ensemble d'*axiomes *(énoncés de base qui sont admis) et un ensemble de* règles d'inférence ou règles de déduction *(qui nous permettent, à partir des axiomes, de démontrer de nouveaux énoncés).*

Définition 2. *Dans un système formel F la notion de* **vérité** *est donnée par une* sémantique*, qui associe une valeur booléenne à toute formule close (souvent en utilisant la théorie des ensembles : vrai si et seulement si son ensemble de modèles est non-vide). Nous nous en tiendrons à l'idée intuitive de « vérité », et au fait que tout énoncé est soit vrai soit faux.*

Définition 3. *Dans un système formel F, une* **preuve** *d'un énoncé peut être vue comme un* arbre *dont la* racine *est l'énoncé prouvé, les* feuilles *sont des axiomes, et les* noeuds internes *correspondent à des applications des règles de déduction.*

Exemples de systèmes formels :
— Géométrie Euclidienne (`https://en.wikipedia.org/wiki/Euclidean_geometry#Axioms`).
— Artithmétique de Peano (`https://fr.wikipedia.org/wiki/Axiomes_de_Peano`).
— ZF et ZFC (`https://en.wikipedia.org/wiki/Zermelo-Fraenkel_set_theory`).
— Quel système formel contient toutes « les mathématiques » ? Réponse courte : ZF. Réponse longue : (`https://en.wikipedia.org/wiki/Foundations_of_mathematics`).

Remarque 4. *Les ensembles d'axiomes et de règles de déduction peuvent être infinis, mais il doivent être semi-décidables (***récursivement énumérable***).*

Voici maintenant les propriétés des systèmes formels que nous allons étudier.

Définition 5. *Un système formel est* **cohérent** *si l'on ne peut pas démontrer un énoncé et sa négation (*càd non-contradictoire*) (sinon on peut démontrer* tous *les énoncés).*

Définition 6. *Un système formel est* **complet** *si l'on peut démontrer tous les énoncés qui sont vrais (*càd *pour tout énoncé on peut démontrer soit l'énoncé soit sa négation).*

On peut alors formuler les théorèmes d'incomplétude de Gödel (1931) [1].

Théorème 7 (incomplétude I)**.** *Aucun système formel contenant l'arithmétique élémentaire ne peut pas être à la fois cohérent et complet.*

Théorème 8 (incomplétude II)**.** *Tout système formel cohérent et contenant l'arithmétique élémentaire ne peut pas démontrer sa propre cohérence.*

Donc même si on ne peut pas le démontrer (theorème 8), on *admet* comme hypothèse implicite à notre utilisation des mathématiques, qu'elles sont cohérentes (non-contradictoires). Il s'ensuit (théorème 7) que c'est un système formel incomplet (*e.g.* l'**hypothèse du continu** est indépendante de ZFC).

1. La formulation originale requiert des définitions plus subtiles (ω-cohérence).

1.3 Premier théorème d'incomplétude

Sources : [1, 6]. On parle ici du théorème 7.

1.3.1 L'énoncé de Gödel

Pour démontrer le théoreme 7, Gödel utilise l'arithmétique du système formel F pour encoder la notion de preuve dans les énoncés. On appelle cela *l'arithmétisation des métamathématiques*. Le numéro de Gödel d'un énoncé $\phi = s_1 s_2 \dots s_n$ avec s_i des symboles est donné par $g(\phi) = \Pi_{i=1}^{n} p_i^{g(s_i)}$ où p_i est le i^{e} nombre premier (l'ensemble des nombres premiers est récursivement énumérable, on peut même les énumérer dans l'ordre) et $g(s_i)$ est le numéro du symbole s_i. On décode $g(\phi)$ en calculant sa décomposition en facteur premiers, dont les exposants nous donnent les symboles de l'énonce. Les preuves sont des suites de symboles, elle ont aussi un numéro de Gödel. C'est un encodage (plutôt « mathématique » que « informatique ») des énoncés et preuves en des nombres, comme peut l'être l'encodage en binaire (mais les notions d'ordinateur et d'information allaient révolutionner notre compréhension des mathématiques et du monde seulement quelques années plus tard...). Gödel parvient à construire l'énoncé $G(F)$ suivant :

$$G(F) = \text{« Cet énoncé n'est pas prouvable dans } F \text{ »}$$

Qui ressemble beaucoup au paradoxe du menteur qui affirme « cette phrase est fausse » : si la phrase est fausse c'est qu'elle est vraie, mais si elle vraie c'est qu'elle est fausse ! On remarquera également l'auto-référence, qui est un élément central dans ces énoncés.

Voici le raisonnement. Si F est complète, alors nous avons deux cas :
- si F prouve $G(F)$ alors F est incohérent (car cette preuve démontre $\neg G(F)$),
- si F prouve $\neg G(F)$ alors
  - soit il existe une preuve de $G(F)$ et alors F est incohérent,
  - soit il n'existe pas de preuve de $G(F)$, mais $\neg G(F)$ affirme qu'une telle preuve existe, et alors F prouve un théorème ($\neg G(F)$) faux, et n'est pas... correct.

Attention : le raisonnement précédent avec $G(F)$ montre un résultat légèrement moins fort que le théorème 7, dans lequel la notion de correction se substitue à la cohérence (à méditer : correction implique cohérence, mais pas réciproquement) : le théorème 10.

Définition 9. *Un système formel est* **correct** *si tous les énoncés que l'on peut prouver son vrais (*càd *les règles du système formel infèrent des raisonnements qui ont du sens).*

Théorème 10. *Aucun système formel contenant l'arithmétique élémentaire ne peut pas être à la fois correct et complet.*

On peut également démontrer le théorème 10 en se servant du théorème de l'arrêt.

Démonstration du théorème 10. Par l'absurde, supposons que l'on ait un système formel F correct et complet, qui soit suffisamment expressif pour raisonner sur les machines de Turing (c'est là que l'arithmétique élémentaire intervient). Alors on pourrait utiliser ce système formel pour résoudre le problème de l'arrêt : étant donnée $\langle M \rangle$ dont on se demande si elle s'arrête sur l'entrée vide, on a un algorithme qui consiste à énumérer toutes les preuves du système F, jusqu'à rencontrer :
- soit une preuve que la machine M s'arrête sur l'entrée vide,
- soit une preuve que la machine M ne s'arrête pas sur l'entrée vide.

Puisque F est complète, une de ces deux preuves existe et sera énumérée donc notre algorithme termine, et puisque F est correcte la conclusion de cette preuve sera vraie. Donc on pourrait décider si M s'arrête ou non, ce qui est en contradiction avec l'indécidabilité du problème de l'arrêt des machines de Turing sur l'entrée vide. □

Ici on voit que F correct plus complet implique F **décidable** : on peut construire un algorithme (c'est ce que fait la preuve) qui, étant donné un énoncé, **décide** s'il admet une preuve ou si sa négation admet une preuve. Alors si le système formel F est suffisamment expressif pour construire des énoncés qui parlent du comportement des machines de Turing (ce que permet tout système qui contient l'arithmétique élémentaire), on pourrait décider le problème de l'arrêt. Or on sait que ce n'est pas possible, contradiction.

Définition 11. *Un système formel est* **décidable** *si il existe un algorithme qui, étant donné un énoncé, décide s'il est prouvable non (ici vérité et prouvabilité coïncident).*

Corollaire 12. *Tout système formel F correct et complet, est* **décidable**.

1.3.2 L'énoncé de Rosser

Pour démontrer le théorème 7, on peut faire appel à l'idée de Rosser (1936) et construire l'énoncé $R(F)$ suivant (une *réfutation* d'un énoncé est une preuve de sa négation) :

« Pour toute preuve de cet énoncé dans F, il existe une réfutation plus courte. »

On a alors un raisonnement qui mène au théorème 7 :

- si F prouve $R(F)$, alors cela prouve qu'il existe une réfutation de $R(F)$ plus courte que cette preuve de $R(F)$, que l'on peut donc effectivement chercher (l'espace de recherche étant fini) et :
  - si on trouve une preuve de $\neg R(F)$ alors F est incohérent,
  - si on ne trouve pas de preuve de $\neg R(F)$, alors on vient de prouver $\neg R(F)$ (il n'existe pas de réfutation plus courte) et donc F est incohérent,
- si F prouve $\neg R(F)$ alors cela prouve qu'il existe une preuve de $R(F)$ plus courte que toute réfutation de $R(F)$, donc il existe une preuve de $R(F)$ plus courte que cette preuve de $\neg R(F)$, que l'on peut donc effectivement chercher (l'espace de recherche étant fini) et :
  - si on trouve une preuve de $R(F)$ alors F est incohérent,
  - si on ne trouve pas une preuve de $R(F)$, alors on vient de prouver $R(F)$ (il existe une réfutation plus courte que toute preuve) et donc F est incohérent.

On remarquera la symétrie de l'argumentation obtenue grâce à l'énoncé de Rosser.

Pour relier l'énoncé de Rosser aux machines de Turing, on peut définir le **problème de devinette cohérente** suivant : étant donné le code $\langle M \rangle$ d'une machine de Turing, on cherche un algorithme (une machine de Turing) qui :

- si M accepte ϵ alors accepte (en s'arrêtant),
- si M rejette ϵ en s'arrêtant alors rejette (en s'arrêtant),
- si M ne s'arrête pas sur ϵ alors accepte ou rejette, mais s'arrête.

On voit qu'il existe une symétrie dans ce problème entre l'arrêt acceptant et l'arrêt rejetant, avec le cas où M ne s'arrête pas qui est en quelque sorte ignoré.

Théorème 13. *Le problème de devinette cohérente n'est pas décidable.*

Démonstration. Supposons qu'il existe une machine P pour le résoudre, alors on peut construire la machine Q qui, sur l'entrée $\langle M \rangle$,

— rejette si $M(\langle M \rangle)$ accepte,
— accepte si $M(\langle M \rangle)$ rejette en s'arrêtant,
— s'arrête (et accepte ou rejette) si $M(\langle M \rangle)$ ne s'arrête pas.

Que vaut $Q(\langle Q \rangle)$? Que le calcul accepte, rejette en s'arrêtant, ou ne s'arrête pas, on obtient une contradiction. □

Démonstration du théorème 7. Par l'absurde, supposons que l'on ait un système formel F cohérent et complet (NB : mais pas nécessairement correct), qui soit suffisamment expressif pour raisonner sur les machines de Turing (c'est là que l'arithmétique élémentaire intervient). Alors on pourrait utiliser ce système formel pour résoudre le problème de devinette cohérente : étant donnée $\langle M \rangle$, on a un algorithme qui consiste à énumérer en parallèle toutes les possibles preuves et réfutations de l'énoncé « M accepte ϵ » dans le système F, jusqu'à rencontrer :

— une preuve de « M accepte ϵ », auquel cas l'algorithme accepte,
— une réfutation de « M accepte ϵ », auquel cas l'algorithme rejette.

Cet algorithme résout bien le problème de devinette cohérente. Tout d'abord, puisque F est complète, une de ces deux preuves existe et sera énumérée donc notre algorithme termine. Ensuite, puisque F est cohérente, cet algorithme ne peut pas faire d'erreur :

— si M rejette vraiment ϵ en s'arrêtant alors cela est démontrable (nombre fini d'étapes de calcul), et donc F serait incohérente,
— si M accepte vraiment ϵ alors cela est démontrable (nombre fini d'étapes de calcul), et donc F serait incohérente
— (et si M ne s'arrête vraiment pas sur ϵ alors notre algorithme... s'arrête).

Or ce problème est indécidable par le théorème 13, une contradiction. □

1.3.3 Chaitin et le paradoxe de Berry

Source : [5, page 2].

La difficulté principale dans le preuve originale de Gödel du théorème 7 réside dans l'autoréférence des énoncés tels de « cet énoncé n'est pas prouvable ». Voici un raisonnement de Chaitin TODO [Chaitin71] qui contourne cette difficulté conceptuelle, basé sur le paradoxe suivant.

Paradoxe de Berry
Soit l'expression « *le plus petit entier positif qui n'est pas définissable en moins de seize mots* ». Cette expression définit cet entier en moins de seize mots.

Définition 14. *La* **complexité de Kolmogorov** $K(x)$ *d'un entier x est définie comme la longueur (en bits) du plus petit programme qui calcule x en sortie (et s'arrête). Cette définition est basée sur le choix d'un langage de programmation.*

Démonstration du ~~théorème 7~~ théorème 10. Soit un système formel cohérent et capable d'exprimer le calcul de la complexité de Kolmogorov (avec l'arithmétique élémentaire), nous allons démontrer qu'il existe un entier L suffisamment grand tel que, pour tout entier x, l'énoncé « $K(x) > L$ » ne peut pas être prouvé.

Par l'absurde, supposons que pour un entier x il existe une preuve de l'énoncé « $K(x) > L$ ». Soit w la plus petite preuve (selon l'ordre lexicographique) d'un énoncé de la forme

« $K(x) > L$ », et soit z l'entier x tel que w prouve « $K(x) > L$ ». Il est facile de donner un programme qui calcule z en sortie : le programme énumère toutes les preuves p, une à une, et pour la première p qui prouve un énoncé de la forme « $K(x) > L$ », le programme donne en sortie la valeur de x et s'arrête. La longueur de ce programme est une constante $+ \log L$. Ainsi, si L est suffisamment grand, alors la complexité de Kolmogorov de z est plus petite que L. Puisque w est une preuve de « $K(z) > L$ » (qui est un énoncé faux), on conclut que le système formel est... ~~incohérent~~ [2] incorrect.

Remarquons que le nombre de programmes de longueur L bits est au plus 2^{L+1}. Alors, pour tout entier L, il existe un entier $0 \leq x \leq Z^{L+1}$ tel que $K(x) > L$. Donc, pour un entier x, l'énoncé « $K(x) > L$ » est vrai qui n'est pas prouvable. □

TODO Dans [5, page 2] une survey de telles preuves est référencée !

1.4 Second théorème d'incomplétude

Source : [5]. On parle ici du théorème 8.

Le second théorème d'incomplétude de Gödel peut être démontré en utilisant les idées de Chaitin (section 1.3.3), et le paradoxe suivant.

Paradoxe de l'interrogation surprise
L'enseignant annonce à la classe : « *la semaine prochaine vous aurez une interrogation, mais il vous sera impossible de savoir quel jour l'interrogation aura lieu, jusqu'au jour où elle aura lieu* ». Alors l'interrogation ne peut pas avoir lieu le vendredi, car sinon la nuit précédente les étudiants le sauront. Puisqu'elle n'aura pas lieu le vendredi, de la même façon l'interrogation ne pourra pas avoir lieu le jeudi, ni les autres jours.

TODO Donner l'idée de la démonstration.

1.5 Sur la longueur des preuves

Source : [7].

On peut également démontrer très simplement que la longueur des preuves croît (en fonction de la longueur des énoncés) plus rapidement que toute fonction calculable (Gödel avait déjà observé cela [3]).

Soit F un système formel **indécidable** et avec un ensemble d'axiomes et règles récursivement énumérables. Pour un énoncé ϕ, on notera $L(\phi)$ la longueur de la plus courte preuve de ϕ si une telle preuve existe, et $L(\phi) = 0$ sinon. Soit $L(n)$ la valeur maximale de $L(\phi)$ pour les énoncés ϕ de longueur au plus n.

Théorème 15. *L croît plus rapidement que toute fonction calculable.*

Démonstration. Par l'absurde, supposons qu'il existe une fontion calculable f qui borne supérieurement L. Alors on peut décider F : étant donnée une formule ϕ à décider, on a l'algorithme suivant :

1. calculer $f(|\phi|)$,
2. énumérer toutes les preuves de longueur au plus $f(|\phi|)$, et pour chacune d'elle vérifier si c'est une preuve de ϕ,

2. Dans [5, page 2] il est écrit *incohérent*, mais prouver un énoncé faux correspond plutôt à ne pas être *correct*.

3. si on trouve une preuve de ϕ alors accepter, sinon rejeter.

Cet algorithme termine puisqu'on ne vérifie qu'un ensemble fini de preuves (toutes celles de taille au plus $f(|\phi|)$), et est correct car si une preuve de ϕ existe, elle est par notre hypothèse de taille au plus $L(|\phi|) \leq f(|\phi|)$ donc on doit la rencontrer. Sinon c'est que $L(|\phi|) = 0$. Or notre système formel est indécidable, une contradiction. □

1.6 Correspondance de Curry-Howard

Pour aller plus loin, il existe des liens très forts entre les notions de :
— preuve dans un système formel,
— programme dans un modèle de calcul.

C'est un peu comme si quand vous écrivez des programmes, vous écrivez des preuves... en fait, c'est même exactement ce que dit la correspondance de Curry-Howard !

TODO En dire plus ? des références ?

Références

[1] S. Aaronson. Blog post : rosser's theorem via turing machines. *https://www.scottaaronson.com/blog/?p=710*, 2011 (consulté en mars 2021).

[2] A. K. Doxiàdis, C. Papadimitriou, A. Papadatos, and A. Di Donna. *Logicomix*. Vuibert (French edition), 2010.

[3] K. Gödel. On the length of proofs. *Traduction anglaise dans* The Undecidable : Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*, edité par M. Davis*, 2004.

[4] D. Hofstadter. *Gödel, Escher, Bach : Les Brins d'une Guirlande Éternelle*. Dunod (French edition), 1979 (1989).

[5] S. Kritchman and R. Raz. The surprise examination paradox and the second incompleteness theorem. *Notices of the AMS*, 57(11), 2010. arXiv:1011.4974.

[6] E. Nagel, J. .R. Newman, K. Gödel, and J.-Y. Girard. *Le théorème de Gödel*. Éditions du Seuil, 1989.

[7] A. E. Porreca. On the length of proofs (episode II). *https://aeporreca.org/blog/length-of-proofs-2*, 2010 (consulté en février 2019).