Qu’est ce que l’**INFORMATIQUE**?
D’où vient-elle?
Et l’IA, c’est quoi?

Christian Retoré **(prof. Informatique, Université de Montpellier)**

Responsable master Intelligence Artificielle et science des données 2021-2024

11 février 2025
Lycée Louis Feuillade
Lunel

Remarques maths/info

- Très liées
- Les étudiants qui réussissent en info ont besoin de compétences en maths, d'un niveau « convenable » surtout pour études longues en informatique (master ou ingénieur).
- Maths inutiles pour le simple utilisateur.

Mathématiques et informatique

- Maths & info ≠
  physique, chimie biologie, économie, linguistique,…
- Maths & info: PAS des sciences empiriques.
- Etudes de constructions abstraites,
  inventées par l'esprit humain.
- Implémentées sur des machines inventées
  et construites par l'être humain.
- Pas une science expérimentale,
  même si ça le devient un peu
  (par ex. tester une suite Python
  pour une tâche de classification de document)

Comparaison avec la mécanique:

- *On **observe** qu'en jetant un caillou il suit une parabole…*
- *Puis Newton, la gravité, la mécanique classique, propose un modèle qui corrobore les observations.*

- *En informatique?*
- *Gödel **invente** la notion de calculabilité et de codage (~1930)*
- *15 ans après Turing (ingénieur) et Von Neumann créent un ordinateur avec l'électronique existante (~1945)*

Informatique: omniprésente!

- Pas seulement comme
  « faire tourner un programme sur un ordinateur »
  (l'enseignement de l'info doit s'améliorer)
- Portables, circuits électroniques,…. ascenseur?
- Systèmes cyber-physiques…
  passage à niveau, machine à café, ascenceur, voiture, avion
- Parallélisme: par ex. gestion d'un site web,
  système de gestion de base de données
- Programmes utiles qui bouclent…. Windows, Androïd, iOS, etc.

Informatique:
dans tous les secteurs d'activités!

- Plus de la moitié des informaticiens
  ne travaillent PAS dans une société du secteur informatique.
- Santé
- Industrie automobile
- Agriculture
- Aéronautique
- Banque, Finances, Assurance
- …

Informatique:
lien profond avec les autres disciplines
Souvent dans les deux sens

A. Dans les deux sens
   - Maths
   - Electronique
   - Linguistique
   - Biologie
B. Utilisation de l'informatique
   - Physique
   - Chimie
   - Lettres
   - Astronomie
   - ….

Qu’est-ce que l’informatique?

- Science?
- Technologie?

Applications
« visibles » (analyse traitement d’image)
« impressionnantes », (chatGPT, DeepL),
« plaisantes » (jeux)
mais disent-elles la nature de l’informatique?

- Citation de Fellows Parberry (souvent attribuée à Dijkstra)
- **« Computer science is no more about computers than astronomy is about telescopes »**
- **« L’informatique ne s’occupe pas plus des ordinateurs que l’astronomie ne s’occupe des télescopes. »**

PAS de démo tout de suite… car c’est trompeur.

- Par exemple démo en biologie de traitement contre le covid par un antiviral
  1. Photo d’un patient mal en point
  2. Photo du patient prenant un comprimé d’antiviral
  3. Photo du patient guéri.
- C’est impressionnant….
- Mais ça ne m’apprend rien sur
  1. comment le virus infecte les cellules, ni sur
  2. comment fonctionne le traitement….

- Les **démos** en info c’est pareil!
- C’est impressionnant…
- Mais **ça n’explique rien** de ce qu’est l’informatique mise en œuvre.

Computer science vs informatique

- *Calcul (~petits cailloux avec lesquels on calculait)*
- *Information (représentation des données et connaissances).*

- *Dès le départ, il y a les 2 aspects:*
  - *Codage sur des entiers / suites finies de 0 et de 1*
  - *Calcul sur des nombres entiers*
- *Représentation des données par des entiers sur lesquels on peut calculer. (entier ~ suite de 0 et 1, parfait pour l'électricité)*

Qu’est ce que l’informatique?
D’où vient elle, historiquement?

- *Il y a des machines à calculer mécaniques:*
  - *Abaques boulier chinois grec abyssinien indien… antiquité*
  - *Blaise Pascal ~ 1650*
  - *Charles Babbage ~1850*
- *Mais ce n’est pas de là que proviennent:*
  - *L’idée de calcul et résolution de problèmes mécanisables*
  - *Ni les premières machines*
- *Origine de l’info:*
  - *Maths*
  - *Et surtout logique mathématique.*

Codage
Et
Représentation des données
Code/Crypto

Représentation des données

- *Une mémoire ne contient que des 0 et des 1*
- *1 donnée = 1 suite de 0 et de 1*

Représentation des données

- *Texte: tableau unidimensionnel de caractères*
- *Image: tableau bidimensionnel de pixels (R,G,B) valeur Red,Green,Blue entre 0 et 255*
- *Vidéo: suite temporelle d'images*
- *Son: suite temporelle de combinaisons de fréquences/puissances*

1 entier <—> 1 entier en binaire

- *problème: 13: 1101 = 01101 =001101 =0001101 = …*
- *Entier <-> suite finie de 0 et de 1 commençant par un 1*
- *Et 0 ???*
- *Codage:*
  Entier n —> n+1 en binaire (suite finie de 0 et de 1, un 1 en 1er)
- *Décodage:*
  suite finie de 0 et de 1, un 1 en 1er —> entier, on retranche 1

1 entiers <—> 2 entiers

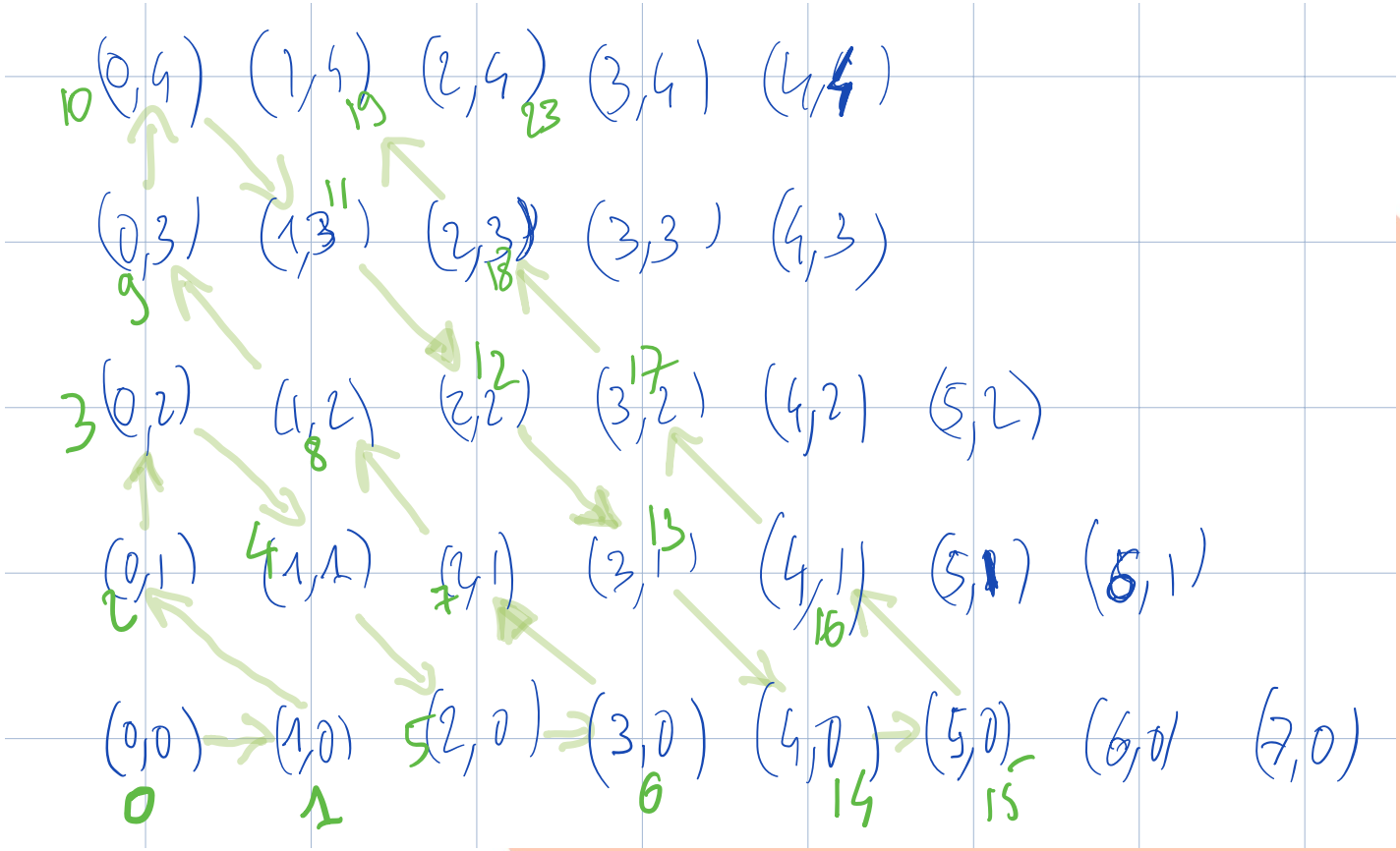
Codage:
2 entiers (n,p) —> 1 entier

Décodage
1 entier —> 2 entiers

Codage / Décodage
Formule simples (cf Wikipédia)

Codage <n,p>=z

Décodage
premier(z)=n
deuxième(z)=p



N entiers <—> 1 entier

- *On va représenter 7 11 7 13 avec le codage des couples d'entiers*
- *On note <n,p> l'entier qui code les 2 entiers n et p.*
- ***<4,<<<7,11>,7>,13>> = z = 1 entier!***
- *Pour décoder premier(z)=4*
- *Donc deuxième(z) = 4 entiers.*
- *deuxième(premier(premier(deuxième(z))=**11***
- ***On peut récupérer chacun des quatre nombres***

1 texte <—> 1 entier

- *1 caractère <—> 1 entier*
  codage ASCII
- *Avec la diapo précédente:*
  1 texte: 1 suite de N caractères = 1 suite de N entiers: —> 1 entier
- *Un texte (un programme, un message….) —> 1 entier*

1 musique <—> 1 entier

- *1 musique = 1 suite temporelle d informations sonores*
- *1 information = 1 nombre fini de couples fréquence intensité*
- *1 information —> 1 entier*
- *1 musique = 1 suite finie d'informations sonores =*
  = suite fini d'entiers = 1 entier!

1 image <—> 1 entier

- *2 entiers: taille de l'image en nombre de pixels L H*
- *Pixel = triplet RGB -> 1 entier*
- *Tableau suite de H lignes de pixels*
- *Lignes = suite de L pixels*
- *1 pixel —> 1 entier*
- *Suite de L pixels —> 1 entier*
- *Suite de H lignes -> 1 entier*

1 video <—> 1 entier

- *1 image —> 1 entier*
- *1 video = 1 suite finie d'image —> 1 entier*
- *Vidéo + son = suite <image, info sonore>*

- *ON A TOUJOURS 1 donnée= 1 entier = suite finie de 0 et de 1*

1 nombre réel, 1 suite infinie de 0 et de 1,
1 flux de 0 et de 1
PAS toujours représentable par une suite finie de 0 et de 1!

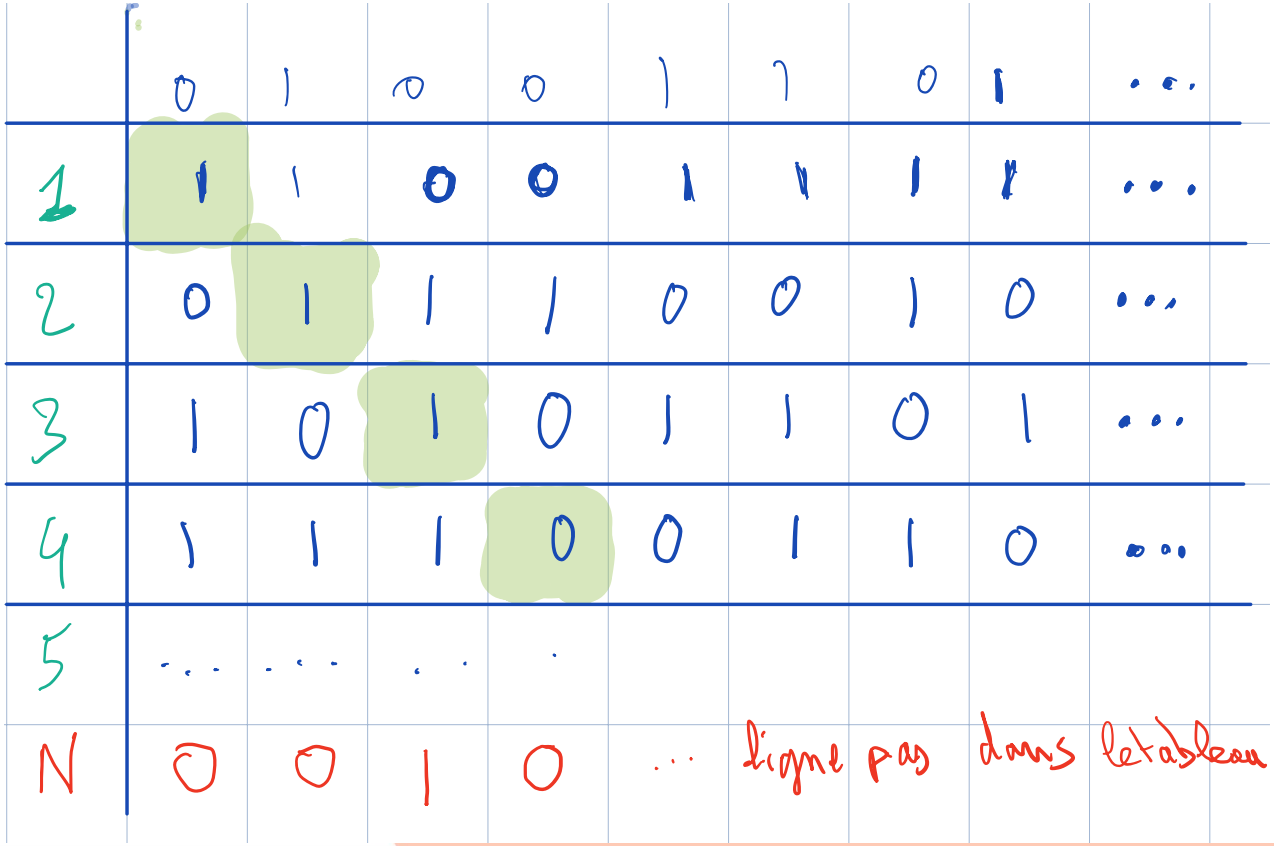
Si on avait une numérotation
des suites infinies de 0 et de 1:
Tableau de toutes les suites.

La ligne en rouge,
Le contraire de la diagonale

Ne peut PAS être dans le tableau:

Si c'est la Nième ligne:
Problème à la case N

Donc pas de de numérotation
des suites infinies de 0 et de 1



1 nombre réel, 1 suite infinie de 0 et de 1,
1 flux de 0 et de 1
PARFOIS représentable par une suite finie de 0 et de 1!

Suite infinie de 0 et de 1:

Si n est premier (seulement divisible par 1 et lui-même)
Alors le n-ième chiffre de la suite est 1
sinon le n-ième chiffre de la suite est 0.

Description finie d’une suite infinie de 0 et de 1

De la logique
Aux premiers ordinateurs

Quelles tâches sont mécanisables, automatisables, programmables?

Première tentative: démonstration automatique de théorèmes de l'arithmétique

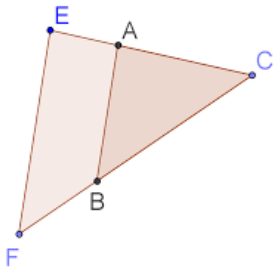
Aux origines de l'informatique: la logique

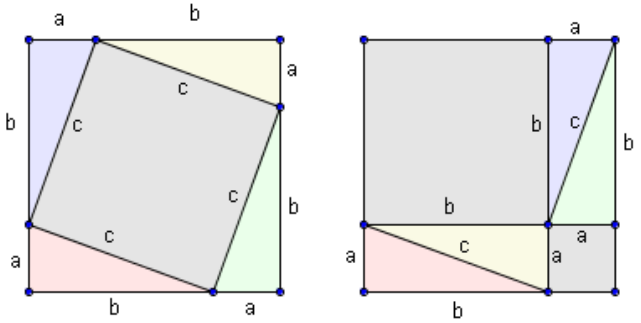
- *En France, la logique est assez peu enseignée:*
  - *Philosophie?*
    (un peu en terminale, et la logique est au programme)
  - *Mathématiques? (rare)*
  - ***Informatique? (dans toutes les licences et bcp de masters)***
  - *Linguistique (sémantique) et philosophie du langage?*

Logique —> démonstration automatique —> calculabilité —> premier ordinateur

- La première tentative d'automatisation de tâche complexe:
- Démontrer automatiquement (et formellement) des théorèmes de l'arithmétique
- Surprenant! D'autant que c'est IMPOSSIBLE!
- Mais
  - Codage
  - Calculabilité (ce qui est programmable)

La logique

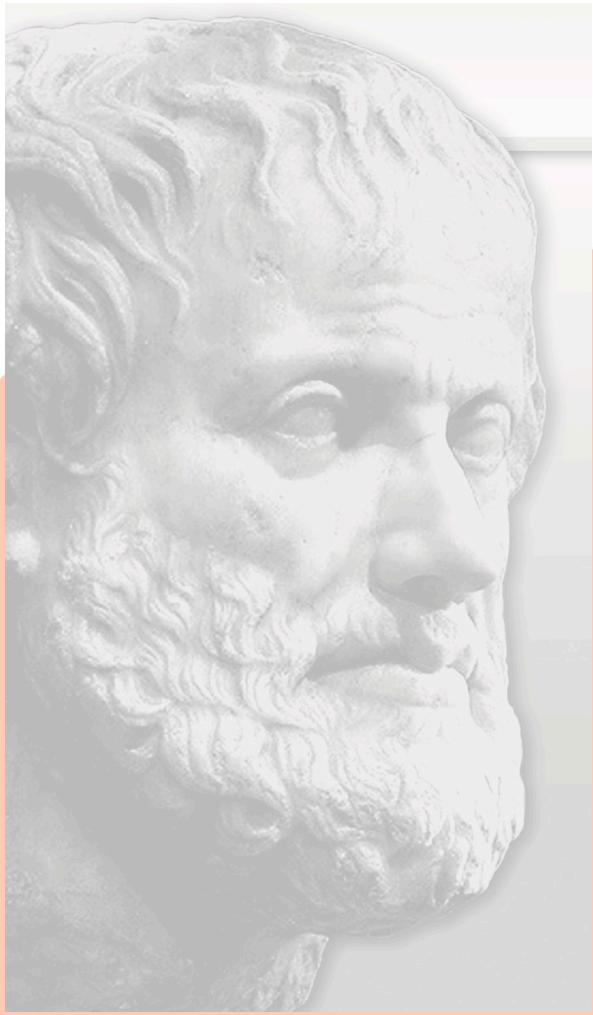




- Art de raisonner correctement
- Avec la rigueur des raisonnements mathématiques (Thalès, Pythagore,… VIIe siècle av. J.C)
- Dériver correctement des énoncés …mais à partir de quels axiomes?
- Etude de la vérité dans une situation particulière, mais cela est plus récent.

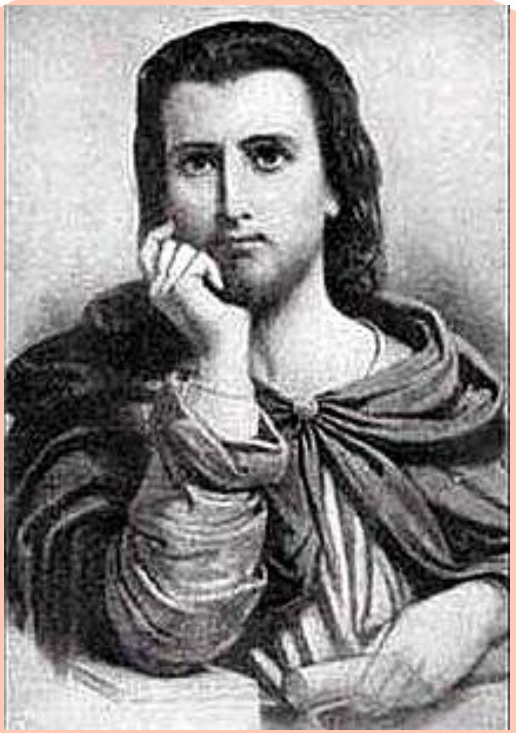
Aristote (III av JC) l'antiquité & la scolastique (moyen âge)

- Certains types d'énoncés:
  - A Tout A est B
  - E Certains A sont B
  - I Aucun A est B
  - O Tous les A ne sont pas B. (ou Certains A ne sont pas B, mais le **thème** est différent)



La scolastique (Antiquité et Moyen-Âge)

- Les fameux syllogismes (règles de déduction)
- Barbara :
  - *tout M est P,*
  - *or tout S est M,*
  - *donc tout S est P;*
- Baroco :
  - *tout P est M,*
  - *or quelque S n'est pas M,*
  - *donc quelque S n'est pas P*



Pierre Abélard (1079-1142)

Déductions formelles

- Des axiomes, par ex.
  - Si A alors A
- Des règles par ex.
  - SI A et (si A alors B) alors B
  - Si on a B en supposant A, alors on a A=>B
  - Si A et Si B alors A et B
  - Si pour tout x A(x) alors A(y) pour un y particulier.
  - Si A(y) pour une variable qui n'a rien de spécial alors « pour tout x A(x) »
  - Si A(t) est vrai pour un t particulier, alors il existe x tel qu A(x)
  - … que des évidences!

Giuseppe Peano (1858-1932)
Axiomatisation de l'arithmétique

- Esperanto mathématique …
- Arithmétique: symboles:
  - 0 (zéro)
  - fonction S: successeur (l'entier suivant)
  - + (addition)
  - x (multiplication)

Axiomes de l'arithmétique

- S(n) successeur de n, c'est-à-dire n+1 (l'entier qui suit n).

- ∀n S(n)≠0
- ∀n si n≠0 alors il existe p tel que S(p)=n
- ∀n ∀p si S(n)=S(p) alors n=p
- ∀n n + 0 = n
- ∀n ∀p n+S(p) = S(n+p)
- ∀n n * 0 = 0
- ∀n ∀p n * S(p) = (n * p) + n

… ainsi que le schéma d'axiomes de récurrence!

- Récurrence:
  pour établir que
  « la propriété P(…) est vraie de tout entier »
- il suffit de
  - Montrer qu'elle est vraie de 0
  - Montrer qu'elle passe au successeur:
    si P(n) alors P(S(n)) — avec S(n)=n+1.
- C'est un **schéma** d'axiome:
  il faut UN axiome
  pour CHAQUE propriété P(…) des nombres.

- SI une propriété est vraie en 0 et passe à l'entière suivant,
  alors elle est vraie de tout entier.

Programme de Terminale

David Hilbert (1862-1943) et son programme: les preuves formelles à l'honneur

Montrer en raisonnant sur les preuves formelles que les axiomes

ne conduisent jamais
à une proposition fausse

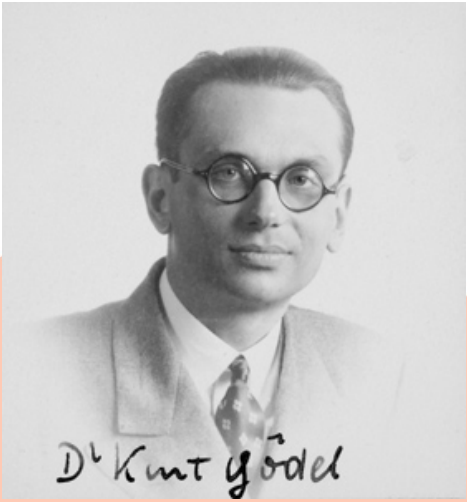
(par ex. 0=1)
par les règles

de déduction formelle

Mécanisation du raisonnement, notamment arithmétique.

C'est dans ce cadre qu'apparaît l'**informatique** !!!

Kurt Gödel (1906-1978)



- Montre que les mathématiciens de son époque se trompent: il n'y a pas de procédé mécanique pour démontrer toutes les propriétés de l'arithmétique.
- Définit la première notion de calculabilité:
  - que veut dire précisément résoudre mécaniquement?
  - Quelles fonctions de N dans N sont calculables? Pas toutes!
  - Définit le codage des suites finies de caractères par des entiers: un tableau d'entiers, une formule, une preuve, une image pixélisée -> un entier!

Résultat (difficiles)

- CODAGE: 1 formule → 1 entier
- CODAGE: 1 démonstration formelle = 1 suite finie de formules → 1 entier.
- Calculabilité …. Pour les théorèmes de l'arithmétique, car formules et démonstrations —> entiers.
- Il existe un entier d qui code une démonstration, qui démontre dans PA la formule codée par l'entier f. Existe d, Pr(d,f).
- Gödel: il existe des formules G telles que
  - PA ne démontre pas G
  - PA ne démontre pas « non G »
  - G est démontrable dans PA est indécidable (la vérité des théorèmes de l'arithmétique n'est **PAS CALCULABLE!)**
- **Vous trouvez ça très compliqué… vous avez raison!**
- ***Les plus grands mathématiciens de l'époque ont mis des années à comprendre la signification de ces résultats.***

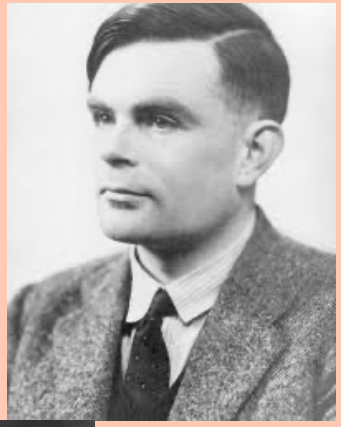
C'est la première notion de CALCULABILITE!

Et de suite on affirme que tout n'est pas calculable!

Et en particulier, on ne pourra jamais remplacer les mathématiciens par des ordinateurs!

Et l'informatique dans tout ça?
Princeton 1929-1945

- Church notion de calculabilité en composant des fonctions (cf. langages fonctionnels: Lisp, CaML, Haskell, Scala,…)
- Gödel en visites régulières à Princeton 1933-1935, fait part de ses travaux notamment à Church dans une série d'exposés en 1934
- Turing ingénieur anglais, vient faire sa thèse avec Church à Princeton 1936-1938
- Machine de Turing
- Machine de Von Neumann… CPU, bus mémoire vive, mémoire pérenne, -> Ordinateurs actuels (1943: colossus mark I, UK).



Logique et Sécurité

Applications en Génie Logiciel:

Spécification, conception et vérification de programmes et de systèmes cyber-physiques

Programmes certifiés:
les preuves vues comme des programmes

- Santé, aéronautique, finances, protocoles de connexion…
  il faut des programme sûrs:
  - Vérification
    - Model checking
    - Preuve de programme
  - Extraction de programmes depuis les preuves

Model Checking / vérification

- On spécifie le le système informatique POSSIBLEMENT parallèle (à la différence d'un programme)
- On utilise une logique qui permet de dire:
  - Dans toutes les situations à partir de maintenant on a …
  - Dans au moins une situation à partir de maintenant on a …
- On cherche à montrer que les bonnes propriétés sont toujours vérifiées:
  - A tout instant Si l'ascenseur monte ou descend, alors les portes sont fermées
  - A tout instant Si les portes sont ouvertes alors l'ascenseur est en face d'un étage.
  - A tout instant Il y aura plus tard un instant ou l'ascenseur sera au rez-de-chaussée.

Preuves de programmes

- On spécifie le programme:
- Par ex si l'entrée est une liste d'entiers,
  alors la sortie est la liste des mêmes entiers, mais triée par ordre croissant.
- On prouve formellement que le programme s'arrête et que la sortie est bien la liste des même entiers ordonnée par ordre croissant.

Extraction d'un programme certifié à partir d'une preuve formelle

- **Procédure:**
  1. On écrit la spécification du programme: (équations)
  2. On démontre formellement que
     pour tout x de type A
     il existe un y de type B tel que P(x,y)
     en utilisant les règles de la logique et les équations.
  3. Cette preuve peut être vu comme un programme qui sera totalement correct:
     qui satisfait exactement la spécification.

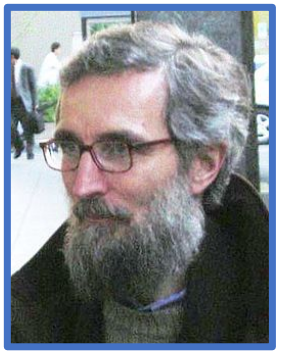
Le programme obtenu (lent mais 100% sûr) fait exactement ce qu'il est supposé faire.

Par exemple, à partir d'une preuve formelle que
pour tout entier n il existe 4 entiers a b c d, tels que n=a^2 + b^2 + c^2 + d^2
on extrait automatiquement
un programme qui prend n en entrée et rend 4 entiers a b c d tels que n=a^2 + b^2 + c^2 + d^2

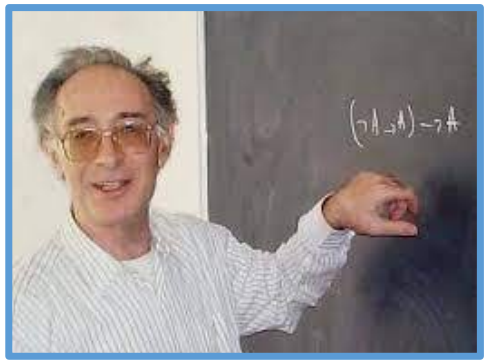
Preuves: des fondements des maths à la logique du calcul

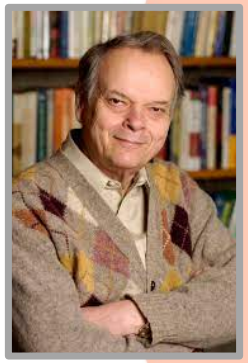
- Initialement: pour la cohérence des mathématiques
- Contenu calculatoire (dans une logique particulière)
- Programmes fonctionnels typés
  - Temps de calcul élevé
  - Mais totalement sûrs (important par ex. circuits aéronautiques)
- Ce sont des langages où tout programme termine.
  C'est lié aux propriétés des preuves de la logique utilisée.

Preuves



- Fondements des maths / informatique dès 1970
  - Jean-Yves Girard (1947-…) *mon directeur de thèse*
  - John Reynolds (1935-2013)
  - Per Martin-Löf (1942-…) Vladimir Voevodski (1966-2017)
  - Jean-Louis Krivine (1939-…)
  - Gérard Huet (1947-…)





Quelques références sur la logique

- **BD LOGICOMIX** de Apostolos Doxiadis, Christos H. Papadimitriou, Alecos Papadatos, and Annie di Donna Vuibert 2010
  Les principaux logiciens du début du XXe sont mis en scène.
  Pas technique, mais des annexes permettent d'aller plus loin.
- ***Video*** *YouTube* **Les théorèmes de Gödel (***David Louapre, directeur scientifique UbiSoft). Science étonnante #34*
- **Roman La déesse des petites victoires** de Yannick Grannec Editions Anne Carrière. 2012 Prix des libraires 2013. *Gödel vu par sa femme. Très bonne reconstitution de la vie scientifique à Vienne puis à Princeton, avec des personnages comme Einstein (l'unique ami de Gödel), von Neuman, Morgenstern,…Très bon roman* per se.
- **Livre** Lewis Carroll **Logique sans peine.** Illustrations de Max Ernst.
- **Livre** Raymond Smullyan **Le livre qui rend fou.**
- **Livre**: Raymond Smullyan **Quel est le titre de ce livre?**

AI — IA
l’intelligence artificielle:
Hier, Aujourd’hui et Demain

Attention aux idée reçues

Intelligence Artificielle

- Langue (langage naturel) écrit/oral:
  - Analyser -> représentations utilisables sur machine
  - Engendrer -> dialoguer, raisonner, traduire…
- Vue image/vidéo
  - Analyser
  - Produire
- Raisonner
  - Un robot transporte une poutre dans un couloir avec un angle
  - A un moment d'une partie d'échecs/de go quel est le meilleur coup?

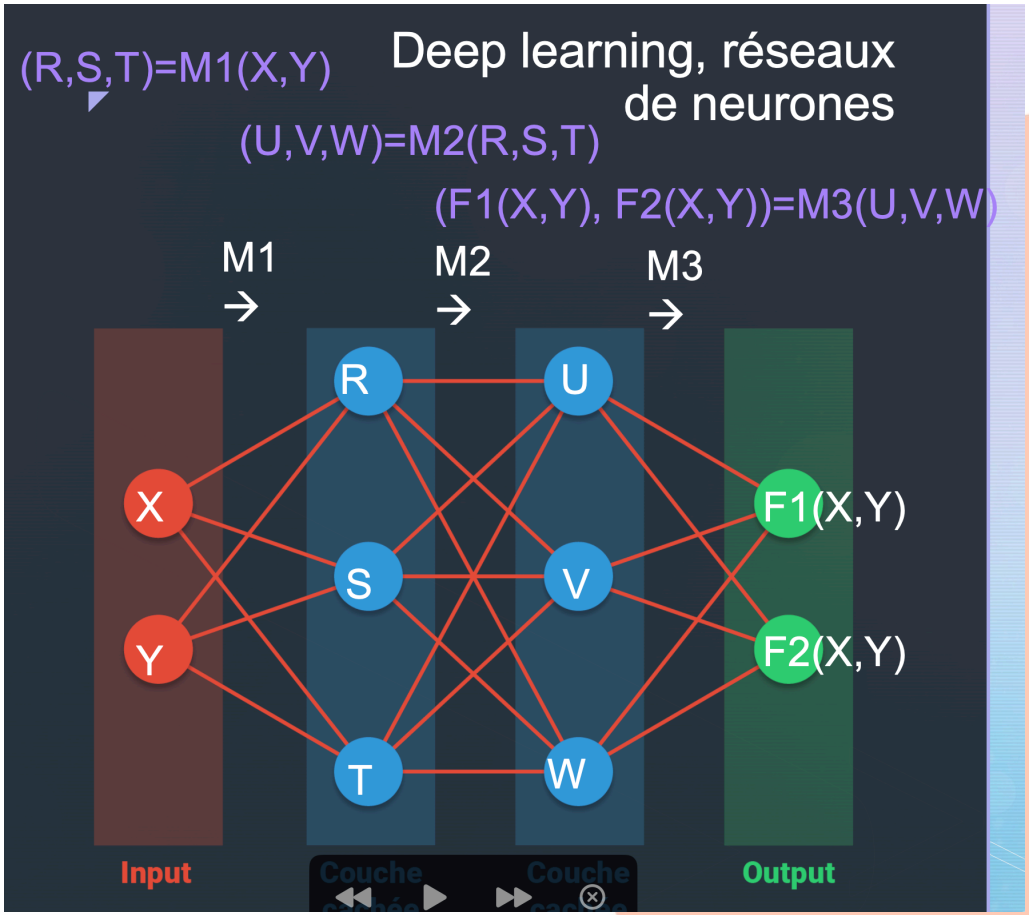
Intelligence Artificielle

- I Méthodes formelles (symboliques)
  - Grammaires formelles
  - Géométrie algorithmique
  - Graphes
  - Logique
- II Méthodes probabilistes et statistiques
  - Modèles de Markov Cachés
  - Réseaux de Neurones

- Analyse syntaxique
- Reconnaissance de formes
- Systèmes experts

- Classification
- Etiquetage
- Séquençage

Ajout de fonctions simples,
mais non linéaires entre les couches.



Deep Learning: résultats bluffants, mais…

- DeepL: traduction, mise dans un style particulier,…
- Création d'images surprenantes

- Parfois, des bugs: images de mammifères à 5 ou 6 pattes

- Ne fonctionne pas si peu de données ou données de mauvaises qualité

- Des difficultés en maths.

- A tendance à confondre « fréquent » et « vrai ».

Une informatique TRÈS différente

- Beaucoup de données bien annotées
  (sinon statistiques classiques) qui détient ces masses de données?
- Pas/peu de conception modification de méthodes
  développées par « une armée » d'ingénieurs
  Qui détient les données? Qui dispose de tous ces ingénieurs?
- Pas/peu de programmation (Python)
- Utilisation de suites Python (PyTorch)
- Définition d'un bon corpus bien annoté puis Fine tuning
  très couteux écologiquement (réchauffement planétaire, énergie)

IA générative (chat GPT)

- Technique: prédiction du mot suivant!
  Avec corpus gigantesque, et en prenant en compte TOUT le contexte!
- Bluffant… surtout la première réponse
- Rédaction à partir de connaissances: fantastique!
- Moins performant dans une discussion serrée
- Pas très fort pour démontrer un résultat mathématique
  sauf s'il s'agit d'un exercice ou d'un résultat connu.

- Problème éthiques:
  - Données (données personnelles, mais aussi données issues de notre production intellectuelle couverte par le droit d'auteur)
  - Ecologie (dépense énergétique souvent injustifiée, réchauffement climatique)
  - Société: Kényans payés 2$ pour rectifier les erreurs et les propos jugés politiquement incorrects — puis remettre leurs corrections dans le corpus d'apprentissage.

IA, études, emploi,...

- Toutes les entreprises veulent des étudiants maitrisant:
  IA, machine learning, apprentissage, analyse de données
  mode, subventions de l'Etat,..
- Souvent des statistiques classiques suffisent en l'absence de masses de données correctement annotées
- Il s'agit surtout d'utiliser et d'adapter les programmes existants
  (construction d'un corpus d'apprentissage spécialisé, fine tuning)
  MAITRISER LE DOMAINE D'APPLICATION EST TRÈS IMPORTANT
  (SAVOIR POSET LES BONNES QUESTIONS ET INTERPRETER LES RÉSULTATS)
- ChatGPT rend obsolète la programmation standard:
  ChatGPT peut programmer presque tous les programmes jusqu'à bac+3 / bac+4 !
- IBM Allemagne a recruté quelques linguistes
  ….plutôt que beaucoup de programmeurs…

Recherche en IA, recherche sur l'IA

- Définition de corpus adaptés, fine tuning
- Fiabilité, mesure de marge d'erreurs (méthodes statistiques)
- Intégration de différents aspects:
  - Image
  - Langage
  - Raisonnement
  - Déplacement…
- Applications: liens avec la robotique, médecine
- Problème éthiques, sociaux et politiques posés par l'IA

Conseils pour vos futures études

Un seul conseil: faites ce qui vous plait.

Qui suis-je? Comment en suis-je arrivé là?

- Etudes de maths à l'université Paris 6 puis Paris 7 (1982-1988)
  pion et remplaçant de prof de maths dans le 94 (1983-1987)
- Thèse de maths à Paris (1987-1992)
  dont 2 ans service national à Londres (Imperial College) au milieu
- Chercheur à l'Institut National de Recherche en Informatique et Automatique 1993-2003: Nice, Nancy, Rennes
- Professeur d'informatique, univ. Bordeaux 2003-2014
- Professeur d'informatique, univ. Montpellier depuis 2014

Université master bac+5 (niveau ingénieur)

- ≠IUT (bac +2/3) Technicien passerelle compliquée aujourd'hui.
- En informatique réussite des étudiants:
  souvent bloquée un faible niveau **maths**
  (le niveau en NSI, peut être rattrapé)
- Pour parcours sup niveau en maths.
- Pour mon master niveau en info théorique.
- 2 sortes de maths:
  - Calcul (image, apprentissage)
    - Algèbre linéaire matrices, vecteurs
    - Probabilité et statistiques
    - Pas beaucoup d'analyse (études de fonction)
  - Logique (arithmétique, raisonnement mathématique)

Choix difficiles pour un ado...
Quelques principes.

- Aimer jouer à des jeux vidéo
  ≠ aimer programmer des jeux vidéo.
  Aimer regarder un math de foot
  ≠ aimer courir après un ballon.
- IUT différent: formation pro courte
- Classes prépa: ça dépend de votre personnalité.
- Pour réussir faites ce que vous aimez faire.
- Qu'est ce que vous aimez faire?
  Question plus difficile que vous croyez.
- N'écouter pas vos amis, ni votre famille
  (pas objectifs, juge et partie, trop d'affect)
- Vous pouvez écouter vos profs qui connaissent les filières et vos aptitudes.

- Quelle que soit la filière, ceux qui réussissent bien trouvent un travail intéressant et bien rémunéré.
- La solution est en vous!
- En informatique ceux que l'informatique intéresse vraiment (pas comme simple utilisateur) réussissent bien le master.
- et trouvent tous un bon travail!
- **Mais si vous n'aimez pas l'informatique,**
  ni les maths
  surtout ne faites pas d'études d'informatique,
  il y a beaucoup d'autres choses passionnantes dans les études possibles
  mais aussi en dehors des études.