Calculabilité et Complexité
(11X008)

Arnaud Casteigts

Bachelor en sciences informatiques,
Université de Genève

(Survol du cours)

Traitement de l'information

Tri de données :

Calcul d'itinéraire :

Reconnaissance d'image :

Mathématiques :

Plus généralement

Un algorithme prend en entrée une suite de symboles (un **mot**) et produit en sortie une autre suite de symboles (un autre **mot**). Il réalise un **traitement**.

Problème de "décision"

Cas particulier où la sortie est **oui** ou **non** (1 ou 0).

Résoudre un problème de décision $\equiv$ déterminer si le mot en entrée fait partie d'un ensemble de mots souhaités.

$\rightarrow$ langages formels.

Langages formels

Langage : ensemble de mots sur un alphabet Σ donné.

Exemples sur $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$:

$L_1 = \{\mathtt{a}, \mathtt{ab}, \mathtt{ac}, \mathtt{abc}, \dots\}$ // mots commençant par a (langage infini)

$L_2 = \{\mathtt{aa}, \mathtt{ab}, \mathtt{ac}, \mathtt{ba}, \mathtt{bb}, \mathtt{bc}, \mathtt{ca}, \mathtt{cb}, \mathtt{cc}\}$ // mots de longueur 2 (langage fini)

Cadre élégant pour étudier la difficulté d'un traitement informatique :

résoudre un problème de décision $\equiv$ déterminer si un mot appartient à un langage donné.

Exemples :

- Déterminer si un nombre k est premier revient à décider le langage :

$$L_{\text{PRIME}} = \{\langle k \rangle \mid k \text{ est premier}\}$$

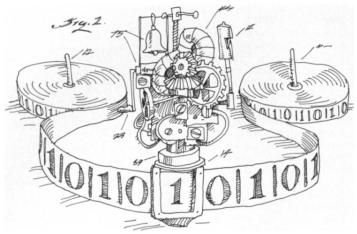
- Déterminer si une photo représente un chat revient à décider le langage :

$$L_{\text{PHOTO-CHAT}} = \{\langle k \rangle \mid k \text{ est une photo de chat}\}$$

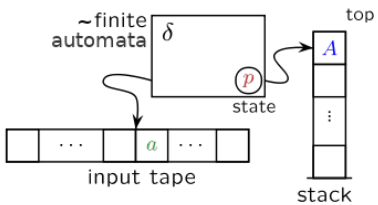
Ici, $\langle x \rangle$ désigne la représentation (ou l'encodage) de x dans l'alphabet considéré. Par exemple, cela pourrait être la représentation binaire d'un nombre ou les valeurs RGB des pixels d'une photo.

Modèles de calcul et expressivité

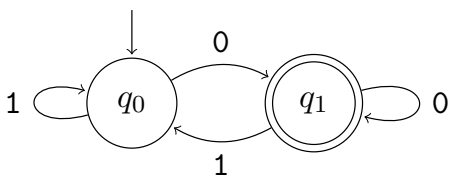
Machine de turing



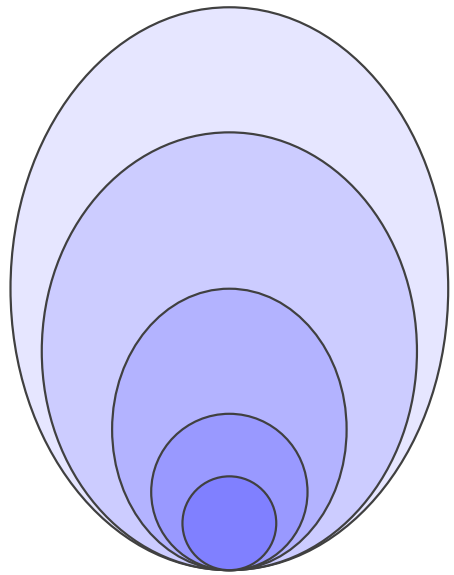
Automate à pile



Automate fini



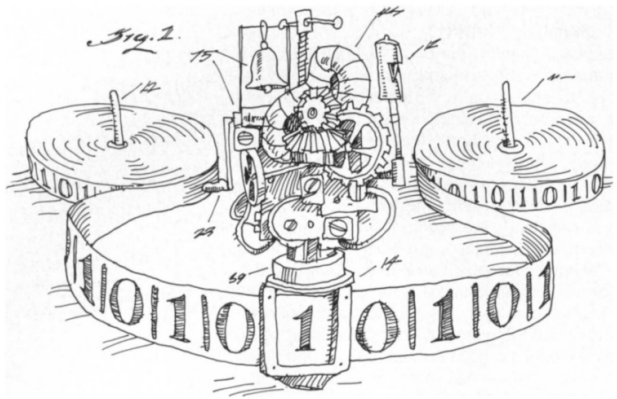
Familles de langages



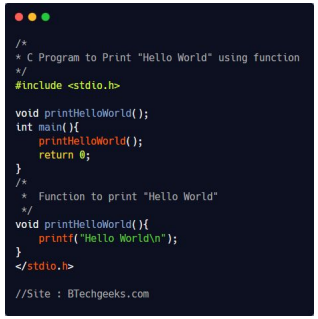
Plus le modèle de calcul est puissant, plus la famille des langages qu'il peut décider est grande.

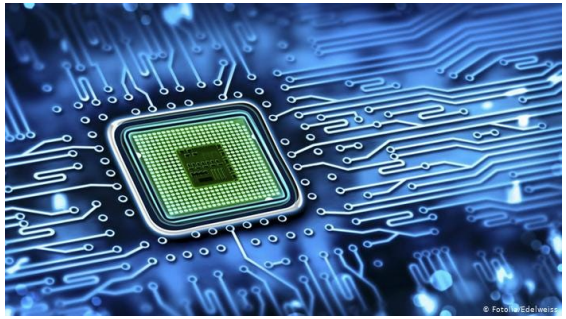
(Partie I)

Calculabilité

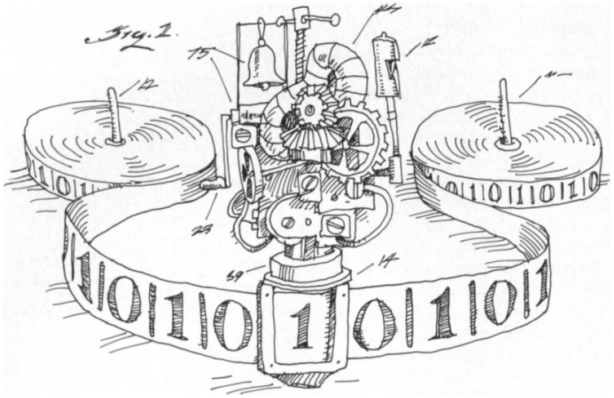


Quelle est la bonne abstraction ?





Machine de Turing (1936)



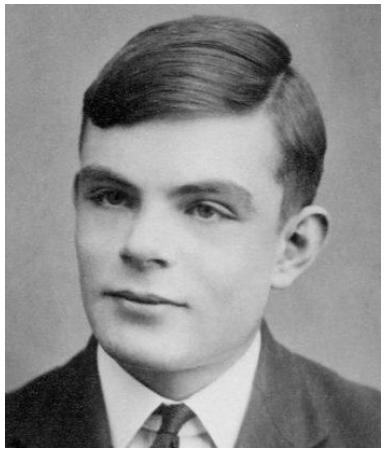
230 A. M. TURING [Nov. 12,

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

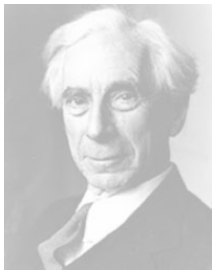
The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*. it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are,



→ Modèle **mathématique** d'algorithme, présumément capable de représenter **n'importe quel** traitement d'information physiquement réalisable (Conjecture de Church-Turing).

La crise des fondements

"Wir müssen wissen, wir werden wissen"
(1930)



Pas avec la théorie des ensembles ! (1901)

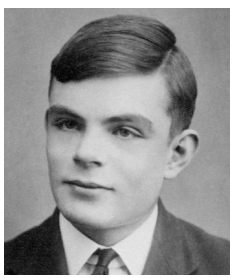
Soit $R = \{x \mid x \notin x\}$, alors $R \in R \iff R \notin R$

Bertrand Russell (1872-1970)

Quel que soit le système, en fait ! (1930)

Tout système axiomatique est soit incomplet, soit incohérent.

Kurt Gödel (1906-1978)



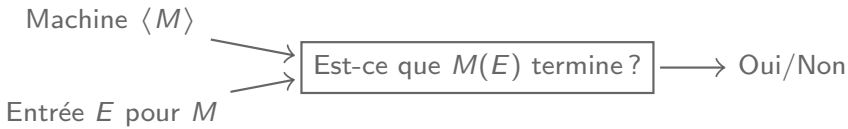
Certaines questions naturelles sont indécidables ! (1936)

Comme savoir si un algorithme donné s'arrête.

Alan Turing (1912-1954)

Le problème de l'arrêt

Problème de l'arrêt (Halting problem) :



Version universelle (pour info) :

Existe-t-il une machine $M_H(\langle M \rangle, E)$ capable de répondre à cette question dans tous les cas ?

Raisonnement par l'absurde

Supposons que M_H existe. On peut alors facilement créer une machine M_P qui résoud le cas particulier suivant :

$M_P(\langle M \rangle)$:
  Si $M_H(\langle M \rangle, \langle M \rangle)$ dit oui :
    Dire oui
  Sinon
    Dire non

On peut aussi (soyons tordus) utiliser M_H pour créer une machine M_T qui prend en entrée une machine $\langle M \rangle$ et adopte le comportement opposé de $M(\langle M \rangle)$, à savoir boucler si $M(\langle M \rangle)$ termine et terminer si $M(\langle M \rangle)$ boucle.

$M_T(\langle M \rangle)$:
  Si $M_H(\langle M \rangle, \langle M \rangle)$ dit oui :
    Boucler à l'infini
  Sinon
    Terminer

Que se passe-t-il si l'on exécute $M_T(\langle M_T \rangle)$?

- Si M_H nous dit que $M_T(\langle M_T \rangle)$ termine, alors $M_T(\langle M_T \rangle)$ boucle
- Si M_H nous dit que $M_T(\langle M_T \rangle)$ boucle, alors $M_T(\langle M_T \rangle)$ termine

$\rightarrow$ M_H ne peut pas exister.

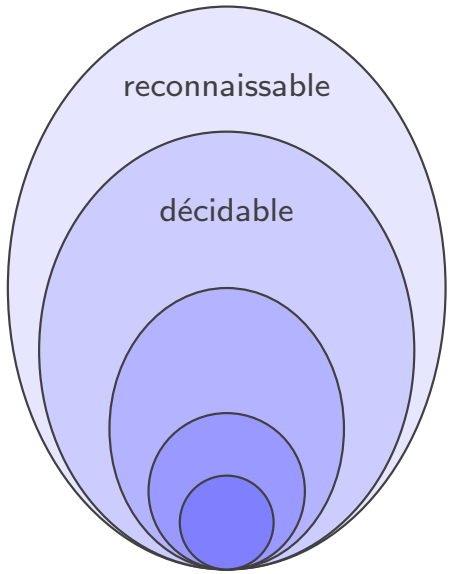
(et toc !)

Décidable vs reconnaissable

On peut définir une notion plus faible que la décidabilité : la **reconnaissabilité**.

Un langage L est **reconnaissable** s'il existe une machine M telle que :

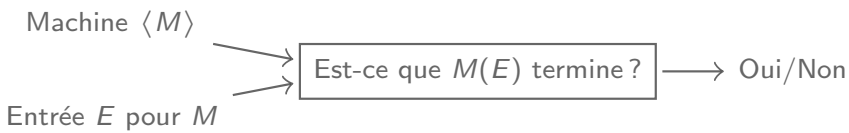
- $w \in L \implies M$ accepte w.
- $w \notin L \implies M$ rejette w ou boucle à l'infini.



Un langage L est **décidable** s'il existe une machine M telle que :

- $w \in L \implies M$ accepte w.
- $w \notin L \implies M$ rejette w.

Problème de l'arrêt (Halting problem) :



Bien que non décidable, le langage L_H associé au problème de l'arrêt est reconnaissable !

Quelques problèmes indécidables

Les problèmes suivants sont indécidables **dans le cas général**. Cela n'empêche pas des cas particuliers d'être décidable, bien sûr.

- Décider si une équation diophantienne (équation polynomiale à coefficients entiers) admet des solutions entières
  P. ex. : $x^2 + y^2 = 3z^2$.

- Problème de correspondance de Post

$$\frac{ca}{a} \quad \frac{a}{ab} \quad \frac{b}{ca} \quad \frac{abc}{c}$$

- Décider si un algorithme répond toujours OUI.
- Nombreuses propriétés de programme... (Théorème de Rice)

En fait :

- Le nombre d'**algorithmes** a la cardinalité des entiers ($\aleph_0$)
- Le nombre de **langages** a la cardinalité des réels ($2^{\aleph_0}$)

$\rightarrow$ La majorité des langages n'est même pas reconnaissable.

(fort heureusement, beaucoup n'ont aucun intérêt)

Natural	Real
0	0.236436775676...
1	0.098473294543...
2	0.193214042202...
3	0.843279242093...
4	0.012934812343...
5	0.639423412934...
6	0.017773923845...
7	0.238920090909...
8	0.123984732999...
9	0.646329878122...
10	0.000123943437...
11	0.981298312892...
⋮	⋮
	0.293233992132...
	0.746894310875...

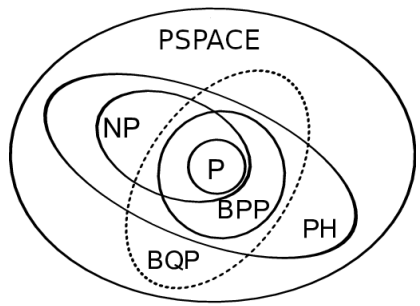
Argument diagonal (Cantor)

Au menu de la partie **calculabilité**

- Différents types de problèmes
- Encodage et simulation d'une machine de Turing par une autre
- Comment montrer "facilement" qu'un langage est indécidable (réduction)
- Problème de correspondance de Post
- Hiérarchie d'indécidabilité (degrés de Turing)
- Preuve de la cardinalité de l'ensemble des langages (diagonalisation)
- Théorème de Rice : toute propriété non-triviale d'un programme est indécidable

(Partie 2)

Complexité algorithmique



(désormais, on reste dans le monde décidable)

Gödel's letter to Von Neumann (1956)

Princeton, 20./II. 1956

Lieber Herr v. Neumann!

Ich habe mit größtem Bedauern von Ihrer Erkrankung gehört. Die Nachricht kam mir ganz unerwartet. Morgenstern hatte mir zwar schon im Sommer von einem Schwächeanfall erzählt, den Sie einmal hatten, aber er meinte damals, dass dem keine größere Bedeutung beizumessen sei. Wie ich höre, haben Sie sich in den letzten Monaten einer radikalen Behandlung unterzogen und ich freue mich, dass diese den gewünschten Erfolg hatte. Ihnen jetzt beste gelt. Ich hoffe u. wünsche Ihnen, dass Ihr Zustand sich bald noch weiter bessert u. dass die neuesten Errungenschaften der Medizin, wenn möglich, zu einer vollständigen Heilung führen mögen.

Da Sie sich, wie ich höre, jetzt kräftig fühlen, möchte ich mir erlauben, Ihnen über ein mathematisches Problem zu schreiben, über das mich Ihre Ansicht sehr interessieren würde: Man kann offenbar leicht eine Turingmaschine konstruieren, welche von jeder Formel F des engeren Funktionalkalküls u. jeder natürl. Zahl n zu entscheiden gestattet, ob F einen Beweis der Länge n hat [Länge = Anzahl der Symbole]. Sei $\psi(F,n)$ die Anzahl der Schritte, die die Maschine dazu benötigt u. sei $\varphi(n) = \max_F \psi(F,n)$. Die Frage ist, wie rasch $\varphi(n)$ für eine optimale Maschine wächst. Man kann zeigen $\varphi(n) \geq Kn$. Wenn es wirklich eine Maschine mit ~~$\varphi(n) = Kn$~~ $\varphi(n) \sim Kn$ (oder auch nur $\sim Kn^2$) gäbe, hätte das Folgerungen von der größten Tragweite. Es würde nämlich offenbar bedeuten, dass man trotz der Unlösbarkeit des Entscheidungsproblems die Denkarbeit der Mathematiker bei ja-oder-nein Fragen vollständig durch Maschinen ersetzen könnte. ~~(Jetzt)~~

Gödel's letter to Von Neumann (1956)

I would like to allow myself to write you about a mathematical problem, of which your opinion would very much interest me. One can obviously construct a Turing machine, which for every formula F in first order predicate logic and every natural number n, allows one to decide if there is a proof of F of length n (length = number of symbols). Let $\Psi(F, n)$ be the number of steps the machine requires for this and let $\varphi(n) = \max_F \Psi(F, n)$.

The question is how fast $\varphi(n)$ grows for an optimal machine. (...) If there really were a machine with $\varphi(n) \sim n$ (or even $\sim n^2$), this would have consequences of the greatest importance. Namely, it would mean that (...) the mental work of a mathematician concerning Yes-or-No questions could be completely replaced by a machine.

(...) It would be interesting to know, for instance, the situation concerning the determination of primality of a number and how strongly in general the number of steps in finite combinatorial problems can be reduced with respect to simple exhaustive search.

Et voilà lancé le domaine de la complexité algorithmique !

Complexité algorithmique ?

Ressources nécessaires pour résoudre un problème (décidable).

Quel type de ressources ?

- Temps (nombre d'opérations)
- Espace (quantité de mémoire)
- Impact du non-déterminisme ?
- Impact de l'aléa ?
- ...

Point de vue **asymptotique**

- **Évolution** de ces quantités en fonction de la **taille** n de l'entrée, quand $n \to \infty$
- Notations $O(\cdot), \Omega(\cdot), \Theta(\cdot)$ (ignore les facteurs constants et les termes dominés)

  Intuition $\leq \quad \geq \quad =$ Ex : $3n^2 + 5n + 4 = \Theta(n^2)$
- Quelques adjectifs :

Constant	$\Theta(1)$	Quadratique	$\Theta(n^2)$
Logarithmique	$\Theta(\log n)$	Exponentiel	$\Theta(2^n)$
Linéaire	$\Theta(n)$	Factoriel	$\Theta(n!)$
Quasi-linéaire	$\Theta(n \log n)$	Polynomial	$O(n^c) = n^{O(1)}$

En général, on s'intéresse au **pire cas** (maximum sur toutes les instances possibles du problème).

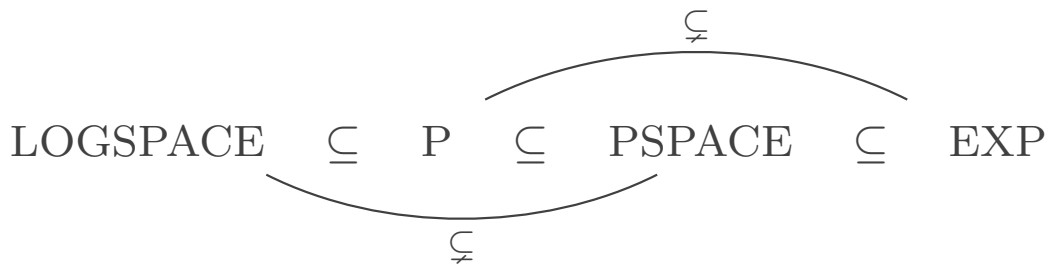
L'espace et le temps

Classes génériques de problèmes

- $\mathrm{TIME}(f(n))$: Problèmes que l'on peut résoudre en temps $O(f(n))$.
- $\mathrm{SPACE}(f(n))$: Problèmes que l'on peut résoudre en espace $O(f(n))$.

Cas particuliers les plus connus

Nom commun	Problèmes résolubles en...	Définition
LOGSPACE	espace logarithmique	$\mathrm{SPACE}(\log n)$
P	temps polynomial	$\mathrm{TIME}(n^{O(1)})$
PSPACE	espace polynomial	$\mathrm{SPACE}(n^{O(1)})$
EXP	temps exponentiel	$\mathrm{TIME}(2^n)$



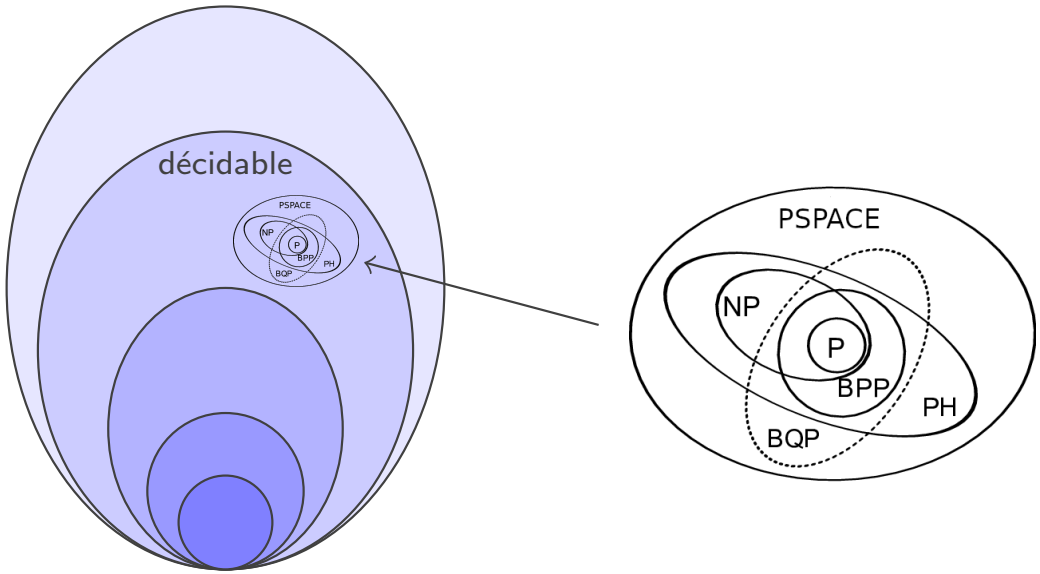
La plus importante est la classe P

Problèmes que l'on peut résoudre "rapidement" (en temps $n^{O(1)}$). (robuste / composable / réaliste)

Un paysage riche de complexité

À l'intérieur de ce qui est décidable, il y a beaucoup de diversité.

Certains problèmes restent inaccessibles, car leur complexité est trop élevée.



Au menu de la partie **complexité**

- Principales classes de complexité
- Impact du non-déterminisme
- Théorème de Savitch
- Classe NP et NP-complétude
- Réductions polynomiales entre problèmes
- Théorème de Cook-Levin
- Ouverture thématique