



Calculabilité, logique et complexité

Chapitre 8

Analyse et perspectives

Acquis d'apprentissage

A l'issue de ce chapitre, les étudiants seront capables de

- Expliquer, justifier et analyser la thèse de Church -Turing
- Décrire et expliquer les propriétés caractéristiques attendues d'un formalisme de calculabilité ainsi que les relations entre ces propriétés
- Comprendre et appliquer les techniques pour analyser et aborder un problème potentiellement non calculable ou potentiellement intrinsèquement complexe
- Analyser et argumenter les relations entre calculabilité et intelligence humaine

Chapitre 8 : Analyse et perspectives

1. Fondements de la thèse de Church-Turing
2. Formalismes de calculabilité
3. Techniques de raisonnement
4. Aspects non couverts par la calculabilité
5. Au delà de la calculabilité ?
6. Au delà des modèles de calculs ?

1. Fondements de la thèse de C-T

Thèse de Church Turing (forme originale)

- **Première partie** : Toute fonction calculable par une machine de Turing est effectivement calculable
- **Seconde partie** : Toute fonction effectivement calculable est calculable par une machine de Turing

La première partie est évidente et peut être prouvée

La seconde partie ne peut être prouvée car la notion de “effectivement calculable” n’est pas formellement définie

Si définition formelle, cela ne fait que reporter le problème ...

Existence d’évidences qui supportent la seconde partie

- Evidences heuristiques
- Equivalences des divers formalismes

1. Fondements de la thèse de C-T

Evidences heuristiques

- La définition des machines de Turing s'approche de la notion intuitive de *procédé effectif* (d'un point de vue humain)
- Toutes fonctions particulières s'avérant effectivement calculables (peu importe le formalisme utilisé) ont été montrées être calculables par une machine de Turing
- Beaucoup d'essais ont été réalisés, couvrant beaucoup de classes de problèmes
- Des méthodes ont été développées pour montrer qu'une fonction effectivement calculable peut être calculée par une machine de Turing
- *Difficile d'imaginer une fonction effectivement calculable qui ne soit pas calculable par une machine de Turing*
- Beaucoup de tentatives de trouver une fonction effectivement calculable qui ne soit pas calculable par une machine de Turing
- Echec de toutes ces tentatives

1. Fondements de la thèse de C-T

Equivalence des formalismes

- Différentes définitions des mêmes formalismes de calculabilité ont conduit à la définition de la même classe de fonctions calculables
- Différents formalismes (très différents; basés sur des principes distincts) ont été proposés pour définir la calculabilité. Ceux-ci possèdent les mêmes évidences heuristiques que pour les machines de Turing
 - logique mathématique (Gödel)
 - lambda calcul (Church)
 - fonctions récursives (Kleene)
 - algorithme de Markov
 - Système de Post
 - langages de programmation
 - ...

The Most Amazing Fact : *Tous ces formalismes recouvrent la même classe de fonctions calculables*

1. Fondements de la thèse de C-T

Mécanisme de calcul

Argument de R. Gandy

On a montré que *les mécanismes de toutes machines constructibles par la mécanique newtonienne ne peuvent calculer que des fonctions programmables*

2. Formalismes de calculabilité

Question: Qu'est-ce qu'un *bon* formalisme pour la calculabilité ?

Réponse : Un formalisme qui vérifie les fondements de la thèse ...

Caractérisation des propriétés nécessaires et suffisantes des formalismes

Soit D , un nouveau formalisme de calculabilité

Caractéristiques

- SD : Soundness des descriptions
- CD : Complétude des descriptions
- SA : Soundness algorithmique
- CA : Complétude algorithmique
- U : description universelle
- S : propriété S-m-n affaiblie

2. Formalismes de calculabilité



SD : Soundness des définitions

Toute fonction D -calculable est calculable

Si il existe une description $d \in D$ calculant une fonction f ,
alors cette fonction est calculable

Première partie de la thèse de Church-Turing

CD : Complétude des définitions

Toute fonction calculable est D -calculable

Si une fonction f est calculable,
alors il existe une description $d \in D$ calculant cette fonction f

Seconde partie de la thèse de Church-Turing

2. Formalismes de calculabilité

SA : Soundness algorithmique



L'interpréteur de D est calculable

Toute description $d \in D$ peut être effectivement exécutée sur des données

Le formalisme D est *algorithmique*

Cette caractéristique assure que D décrit bien des procédés effectifs

CA : Complétude algorithmique

Si P est SA, alors il existe un compilateur (calculable) tel que étant donné $p \in P$, ce compilateur appliqué à P produit une description $d \in D$, avec p équivalent à d (i.e. calcule la même fonction)



Cette caractéristique assure l'équivalence des formalismes

2. Formalismes de calculabilité

U : description universelle

L'interpréteur de D est D-calculable 

Le formalisme doit permettre de décrire son propre interpréteur

S : propriété S-m-n affaiblie

$$\forall d \in D \quad \exists S : d(x,y) = [S(x)](y)$$

Il existe un **transformateur de programme** (total calculable), qui recevant comme données :



- un programme d à 2 arguments
- 1 valeur x

fournit comme résultat :

- un programme d' à 1 argument tel que $d'(y)$ calcule la même fonction que $d(x,y)$

2. Formalismes de calculabilité

Propriétés

- $SA \Rightarrow SD$ ✓
- $CA \Rightarrow CD$
- $SD \text{ et } U \Rightarrow SA$ 
- $CD \text{ et } S \Rightarrow CA$
- $SA \text{ et } CD \Rightarrow U$
- $CA \text{ et } SD \Rightarrow S$
- $S \text{ et } U \Rightarrow S\text{-}m\text{-}n$
- $SA \text{ et } CA \Leftrightarrow SD \text{ et } CD \text{ et } U \text{ et } S$ 
- $SA \text{ et } CD \text{ et } S \Leftrightarrow CA \text{ et } SD \text{ et } U$

\exists interpréteur de D calculable (SA) \rightarrow Toute fct calculable en D peut être calculée grâce à cet interpréteur (SD)

2. Formalismes de calculabilité

Formalisme complet

Un **bon** formalisme de calculabilité doit posséder toutes ces caractéristiques

Exemples de “mauvais” formalismes

Fonctions primitives récursives, langage BLOOP

- SD, SA, S
- non CD, non CA, non U

Autres “mauvais” formalismes possibles

- SD, CD, CA, S, non SA, non U
- SD, CD, SA, U, non CA, non S
- CD, CA, U, non SD, non SA, non S

3. Techniques de raisonnement

Comment démontrer la non calculabilité d'une fonction, d'un problème ?

Essentiel d'un point de vue programmation

Schéma de raisonnement

1. Essayer de trouver un algorithme qui résout le problème
2. Si (1) ne marche pas (après avoir essayer un certains temps), se demander si suspicion possible de non calculabilité du problème (i.e. un algorithme n'existe pas)
3. Dans ce cas, essayer de prouver la non calculabilité
4. Si problème non calculable, essayer de définir une problème approché qui est calculable

3. Techniques de raisonnement

Techniques de preuve

- Utiliser le théorème de Rice
- Démonstration directe de la non calculabilité : **diagonalisation** et preuve par l'absurde
- Méthode de réduction
 - réduire la solution d'un problème A à celle d'un autre problème A' (A est un cas particulier de A')
 - à partir d'un algorithme pour A' construire un algorithme pour A
 - conclusion :
 - si A' calculable
alors A calculable
 - si A non calculable
alors A' non calculable

3. Techniques de raisonnement

Comment montrer qu'un problème est intrinsèquement complexe ?

- Attention que l'existence d'un algorithme exponentiel pour résoudre un problème ne rend pas nécessairement ce problème exponentiel ! Il peut exister un autre algorithme polynomial pour ce même problème
- Si suspicion que le problème est exponentiel, regarder si un problème NP-complet connu peut être réduit polynomialement à ce problème
- Existe-t-il un algorithme non-déterministe polynomial pour ce problème ? Si oui, le problème est dans NP. Si non, ce sera encore plus difficile...

3. Techniques de raisonnement

Comment aborder un problème intrinsèquement complexe ?

- Utiliser un algorithme exponentiel si la plupart des instances à résoudre sont de complexité polynomiale
- Changer le problème en un problème plus simple (polynomial) dont la pertinence est proche de celle du problème initial
- Utiliser une technique d'exploration, mais en se limitant à un nombre polynomial de cas (heuristique). L'algorithme est incomplet; il peut donner une réponse erronée ou approximative. Mais c'est parfois mieux que pas de réponse du tout
- Si problème d'optimisation, calculer une solution approximative.

4. Aspects non couverts par la calculabilité

La calculabilité considère le calcul de *fonctions*

- Certains programmes informatiques ne peuvent être assimilés au calcul d'une fonction
 - système d'exploitation
 - système de réservations aérienne
 - App hautement interactive
 - IoT
 - ...
- Certains programmes informatiques ne peuvent être simulés par une machine de Turing car ils exploitent certaines caractéristiques de leur environnement
 - programmes utilisant comme données des valeurs en provenance de sondes extérieures
 - ...

4. Aspects non couverts par la calculabilité

Définition calculabilité

Fonction calculable

- *il existe* un programme qui calcule cette fonction
- Il est possible que personne ne puisse déterminer quel algorithme calcule cette fonction
- Perd-on la notion intuitive de calculabilité ?

Objectif de la définition de calculabilité : accent sur les conséquences de la non calculabilité

- Si une fonction est non calculable, inutile d'espérer de pouvoir la calculer
- Si une fonction est calculable, on peut alors essayer de trouver un algorithme (efficace) pour la calculer

4. Aspects non couverts par la calculabilité

Ressources

Une fonction calculable (en théorie) peut être non calculable en pratique

- Nécessite trop de ressources (temps ou espace)

Théorie de la complexité

Déterminer si une fonction (théoriquement) calculable peut effectivement être calculée sur un ordinateur : problèmes pratiquement faisables

5. Au delà de la calculabilité ?

- Pourquoi ne peut-on pas imaginer un modèle de calcul qui soit plus puissant que les machines de Turing ou que les langages de programmation ?
- Nos modèles de calculabilité sont-ils limités par les limites de l'intelligence humaine ?
- Une fonction non calculable peut-elle être “calculée” par un être humain ?
- Quelles différences existent-ils entre une machine de Turing et un être humain ?
- Le processus mental d'un être humain peut-il être décrit par une machine de Turing ?
- ...

5. Au delà de la calculabilité ?

Calculabilité et intelligence humaine

Si un être humain particulier est capable de calculer une fonction particulière (en un temps fini, même résultat pour une même donnée)

nous dirons que cette fonction est *H-calculable*

H-calculable = T-calculable ?

Thèse CT: version “procédés publics”

Si une fonction est H-calculable et que l'être humain calculant cette fonction est capable de décrire (à l'aide du langage) à un autre être humain sa méthode de calcul de telle sorte que cet autre personne soit capable de calculer cette fonction,
alors la fonction est T-calculable

Cette version n'exclut pas l'existence de fonctions H-calculables, mais non calculables

5. Au delà de la calculabilité ?

Thèse CT: version réductionniste

Si une fonction est H-calculable, alors elle est T-calculable

- Le comportement des éléments constitutifs d'un être vivant peut être simulé par une machine de Turing
- Tous les processus cérébraux dérivent d'un substrat calculable

Thèse CT: version anti-réductionniste

Certains types d'opérations effectuées par le cerveau, mais pas la majorité d'entre elles, et certainement pas celles qui sont intéressantes, peuvent être exécutées de façon approximative par les ordinateurs

Certains aspects seront cependant toujours hors de portée des ordinateurs

H-calculable \neq T-calculables

5. Au delà de la calculabilité ?

H-calculable = T-calculable ?

Pas de réponse purement scientifique à cette question

Différentes positions possibles

“Croyance”

- H-calculable = T-calculable
- Exemple: réductionniste

“Athée”

- H-calculable \neq T-calculable
- Exemple: anti-réductionniste

“Agnostique”

- Pas intéressés par cette question

“Iconoclaste”

- La question est mal posée, n’a pas de sens

5. Au delà de la calculabilité ?

Calculabilité et intelligence artificielle

Deux approches possibles pour l'intelligence artificielle

- (IA forte) Simulation du cerveau humain à l'aide d'un ordinateur
En copiant le fonctionnement, on espère ainsi obtenir des résultats similaires
- (IA faible) Programmation de méthodes et techniques de raisonnement en exploitant les caractéristiques propres des ordinateurs

La première approche (IA forte) s'apparente à la version “réductionniste” de la thèse de Church- Turing

5. Au delà de la calculabilité ?

Puissance cerveau humain

- Puissance cerveau humain estimée à 10^{18} instructions par seconde
- Ordinateurs les plus puissants (11/2020)
 - Fukagu (Japan), 442 PFLOPS (10^{15})
 - Summit (IBM, Tennessee), 148.8 PFLOPS (10^{15})
 - Sierra (Californie), 94.6 PFLOPS (10^{15})
 - SunwayTaihuLight (Chine), 93 PFLOPS (10^{15})
 - 1 FLO est environ 2 à 10 instructions
 - Ordinateur le plus puissant : environ 10^{18} instructions par seconde
- Les ordinateurs les plus puissants approchent la puissance estimée du cerveau humain
- Mais faut-il encore un programme adapté...

5. Au delà de la calculabilité ?

Les machines peuvent-elles penser ?

Test de Turing (1950)

Principes

- 3 acteurs : un ordinateur (A), un être humain (B), un interrogateur (humain)
- communication exclusivement via message écrit
- l'interrogateur ne sait qui parmi (A) et (B) est l'ordinateur, ni qui est l'être humain
- l'interrogateur peut poser des questions à (A) et (B) qui doivent répondre
- objectif de l'interrogateur : trouver qui est l'ordinateur et qui est l'être humain
- objectif de l'ordinateur : essayer de faire perdre l'interrogateur
- objectif de l'être humain: essayer de faire gagner l'interrogateur

Si l'interrogateur se trompe en moyenne une fois sur deux,
alors *les machines peuvent penser...*

5. Au delà de la calculabilité ?

Les machines pensent-elles ?

Principe de raisonnement (Turing 1950):

- Hypothèse: les machines pensent
- Si on ne peut réfuter cette hypothèse, c'est que celle-ci est vraie
- Envisager toutes les objections possibles et ensuite réfuter chacune d'elle

Objections possibles

- objection théologique
- objection de l'autruche
- objection mathématiques (calculabilité)
- objection issue de la conscience
- objection par les émotions
- objection par la naissance
- objection par aspects discrets des machines
- objection issue du manque de créativité
- objection de la non formalisation du comportement humain

5. Au delà de la calculabilité ?

Questions ouvertes

- L'intelligence peut-elle s'incarner dans un quelconque substrat (organique, électronique, ...) ?
- L'Esprit est-il plus qu'une "configuration" de neurones ?
- Comment distinguer un esprit "véritable" d'un habile déguisement ?
- La créativité obéit-elle à des règles ? Si oui, n'est-ce pas contradictoire ?
- L'intellect et les émotions sont-ils dans des "compartiments" distincts ?
- Une machine peut-elle être sujette à des émotions ?
- ...

5. Au delà de la calculabilité ?

Science et philosophie

Exemples de champs philosophiques ayant été bouleversés par les progrès scientifiques

- Conception de la vie et de l'homme
 - Théorie Darwinienne
- Conception du temps et de l'espace
 - Théorie de la relativité
- Conception du déterminisme et des objets physiques
 - Mécanique quantique
- Conception du raisonnement et de ses limitations
 - Calculabilité et logique mathématique

6. Au delà des modèles de calculs ?

De nouveaux modèles de calcul, basés sur de nouvelles technologies peuvent-ils modifier les frontières de la calculabilité et de la complexité ?

- Frontière de la non calculabilité est indépendante des progrès technologiques
- Les problèmes NP-complets pourraient-ils être résolus en temps polynomial grâce à de nouvelles technologies ?

Nouveaux modèles de calcul

- Ordinateurs basés sur l'ADN
- Ordinateurs quantiques

6. Au delà des modèles de calculs ?

Ordinateurs quantiques

- Le résultat est une superposition de réponses, avec chacune une probabilité
- Un algorithme quantique ne calcule pas de réponse exacte
- Comparaison de puissance difficile avec une machine de Turing
- Bounded-error Quantum Computing (BQP) : $A \in \text{BQP}$ ssi il existe un algorithme quantique polynomial qui donne la bonne réponse au moins 2 fois sur 3
- Pas de réponse exacte, mais précision aussi précise que voulue en un temps polynomial
- Relation supposée entre BQP et NP

