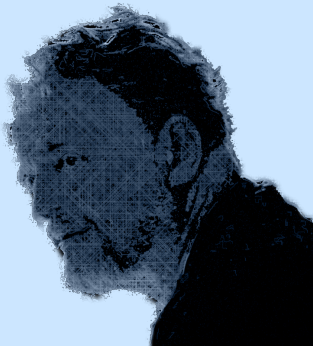
Le calcul numérique

Nicolas Ollinger (LIFO, Université d'Orléans)

EJCIM 2022, Nice — 10 juin 2022



Avant-propos

Cet exposé est principalement une introduction au cours qui suit sur **le calcul analogique**.

Nous présentons quelques éléments de **calculabilité** autour de **modèles de calcul classiques**.

Un bagage de M1 informatique...

Avant-propos

Cet exposé est principalement une introduction au cours qui suit sur **le calcul analogique**.

Nous présentons quelques éléments de **calculabilité** autour de **modèles de calcul classiques**.

Un bagage de M1 informatique. . . bientôt en lycée ?

Contenus	Capacités attendues	Commentaires
Notion de programme en tant que donnée. Calculabilité, décidabilité.	Comprendre que tout programme est aussi une donnée. Comprendre que la calculabilité ne dépend pas du langage de programmation utilisé. Montrer, sans formalisme théorique, que le problème de l'arrêt est indécidable.	L'utilisation d'un interpréteur ou d'un compilateur, le téléchargement de logiciel, le fonctionnement des systèmes d'exploitation permettent de comprendre un programme comme donnée d'un autre programme.

Thèse de Church, Turing, Kleene, Post *et al.*

Théorème (Gandy 1980) Toute fonction **discrète** calculable par un dispositif **mécanique déterministe**, régit par les règles de la physique classique et qui satisfait les **hypothèses** suivantes, est Turing-calculable : **(...)**

Thèse de Church, Turing, Kleene, Post *et al.*

Théorème (Gandy 1980) Toute fonction **discrète** calculable par un dispositif **mécanique déterministe**, régit par les règles de la physique classique et qui satisfait les **hypothèses** suivantes, est Turing-calculable : **(...)**

Thèse de Church-Turing Toute fonction calculable par une méthode effective est Turing-calculable.

Thèse de Church, Turing, Kleene, Post *et al.*

Théorème (Gandy 1980) Toute fonction **discrète** calculable par un dispositif **mécanique déterministe**, régit par les règles de la physique classique et qui satisfait les **hypothèses** suivantes, est Turing-calculable : **(...)**

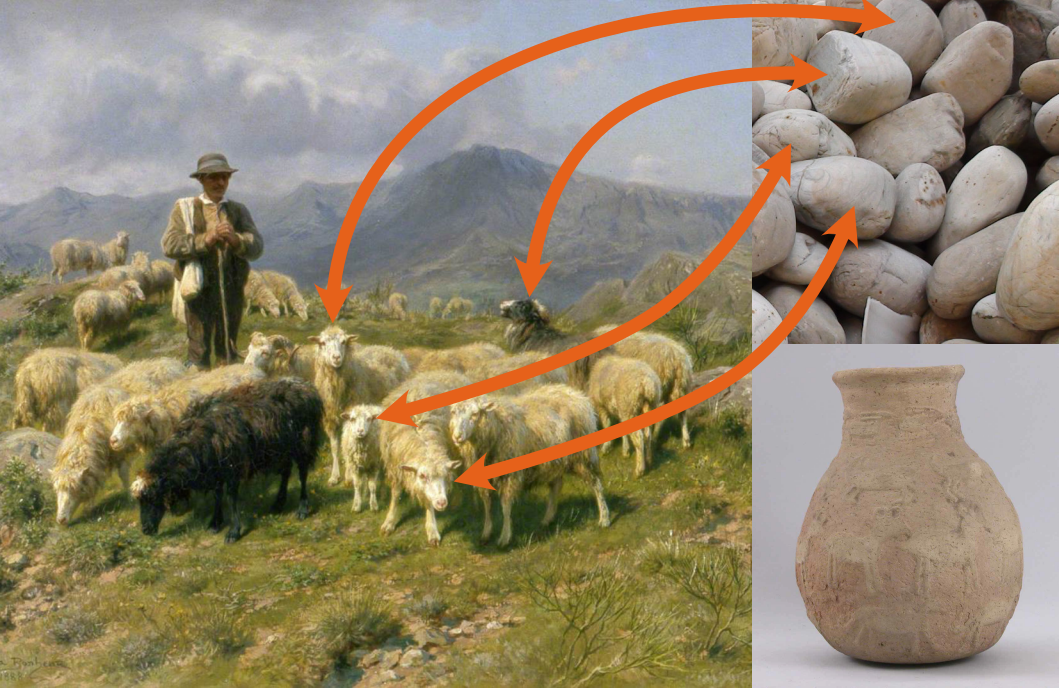
Thèse de Church-Turing Toute problème **décidable** par une méthode effective est Turing-**décidable**.

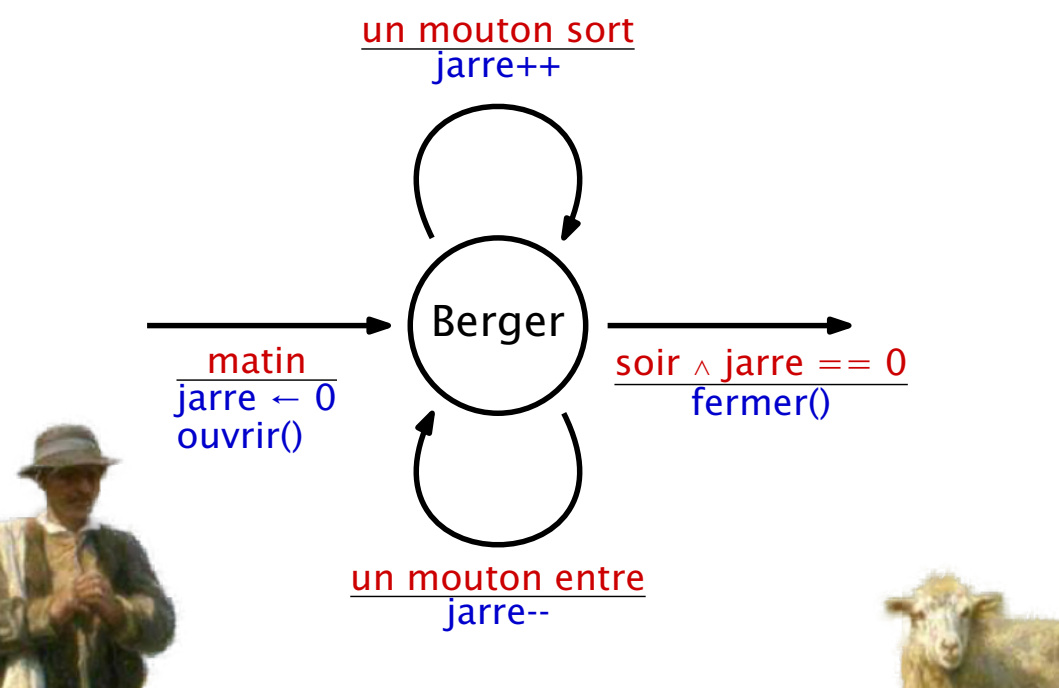
Thèse de Church, Turing, Kleene, Post *et al.*

Théorème (Gandy 1980) Toute fonction **discrète** calculable par un dispositif **mécanique déterministe**, régit par les règles de la physique classique et qui satisfait les **hypothèses** suivantes, est Turing-calculable : **(...)**

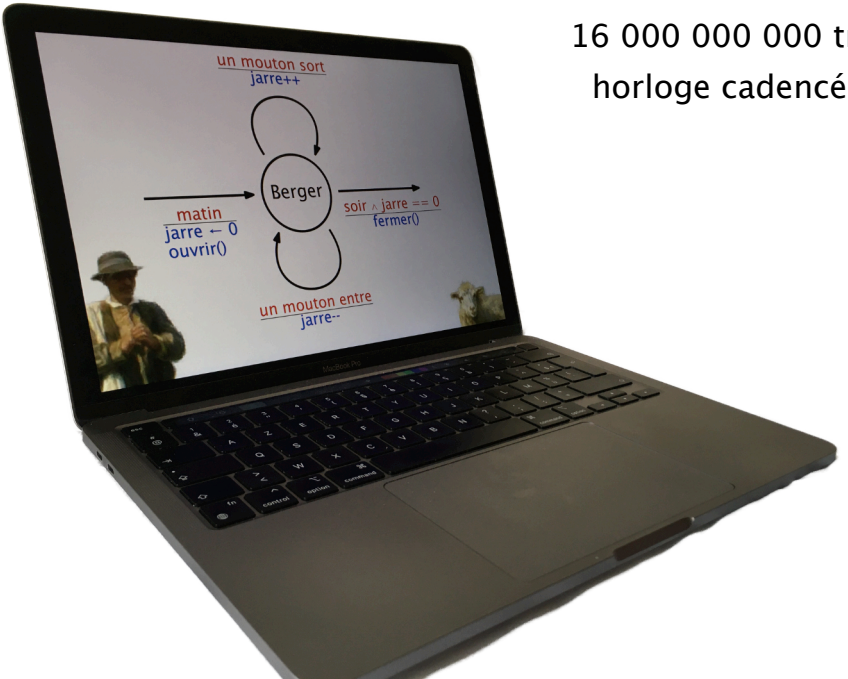
Thèse de Church-Turing Toute problème **décidable** par une méthode effective est Turing-**décidable**.

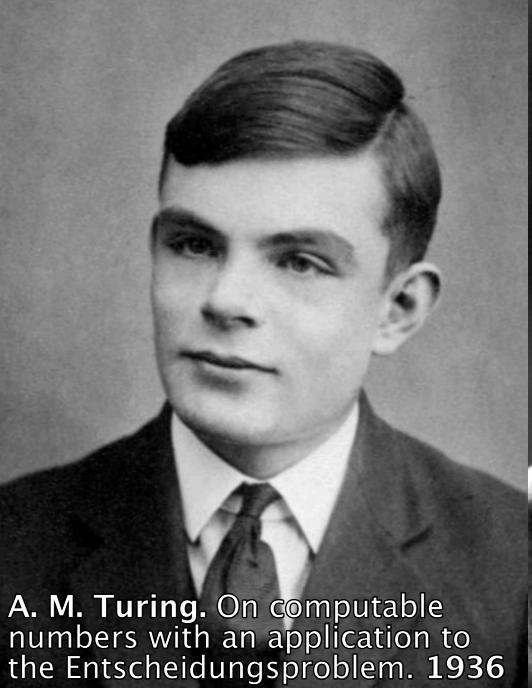
Corollaire Tout problème Turing-**indécidable** est **indécidable** par toute méthode effective.



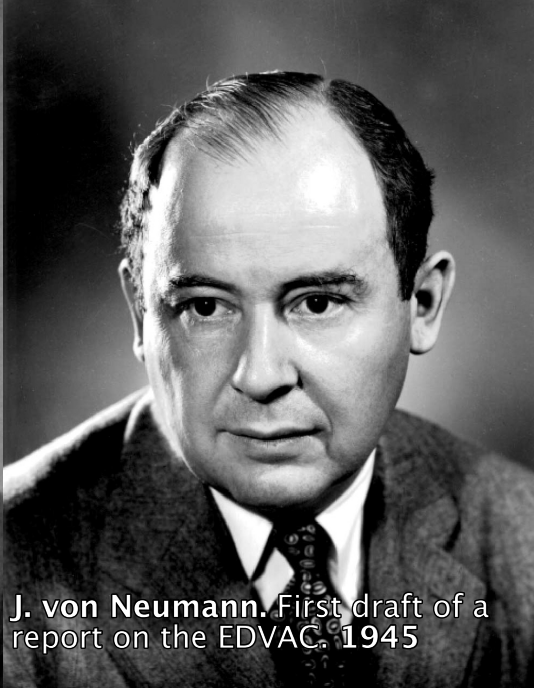

un mouton sort
jarre++
Berger
matin
jarre ← 0
ouvrir()
soir ∧ jarre == 0
fermer()
un mouton entre
jarre--

16 000 000 000 transistors
horloge cadencée à 3 GHz

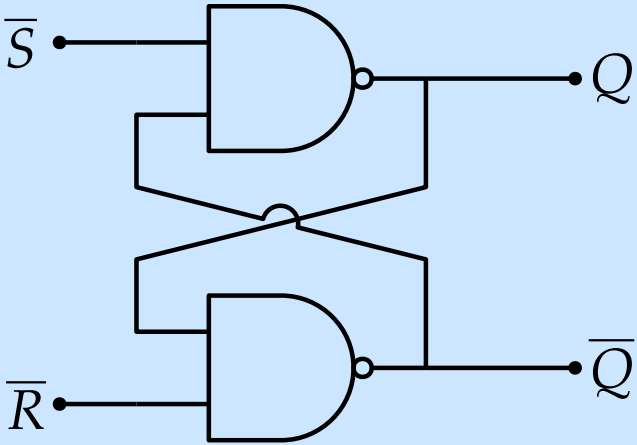




A. M. Turing. On computable numbers with an application to the Entscheidungsproblem. **1936**



J. von Neumann. First draft of a report on the EDVAC. **1945**



1. Machines finies et circuits booléens

Calcul de fonction

$$f : A \to B$$

Calcul de fonction

$$f : \{0,1\}^m \to \{0,1\}^n$$

Calcul de fonction

$$f : \{0,1\}^m \to \{0,1\}^n$$

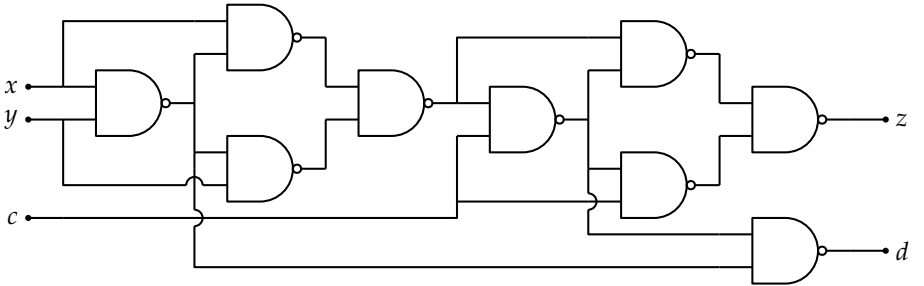
Théorème Toute fonction **binaire** est calculée par un **circuit combinatoire** comportant des portes **ET**, **OU** et **NON**.

Calcul de fonction

$$f : \{0,1\}^m \to \{0,1\}^n$$

Théorème Toute fonction **binaire** est calculée par un **circuit combinatoire** comportant des portes **ET**, **OU** et **NON**.

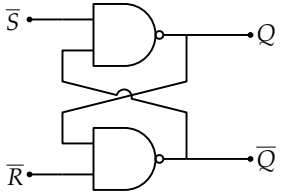
Théorème Toute fonction **binaire** est calculée par un **circuit combinatoire** comportant uniquement des portes **NON-ET**.



Addition de trois bits avec neuf portes NON-ET

Circuits séquentiels

Que se passe-t-il lorsque le **graphe du circuit** comporte des **cycles** ?

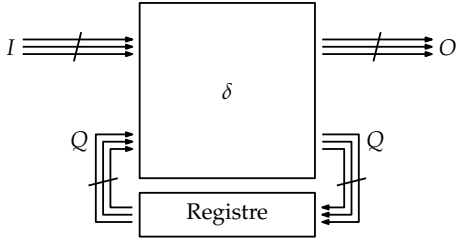


On obtient des **circuits séquentiels** dans lesquels les sorties dépendent des **états passés** :

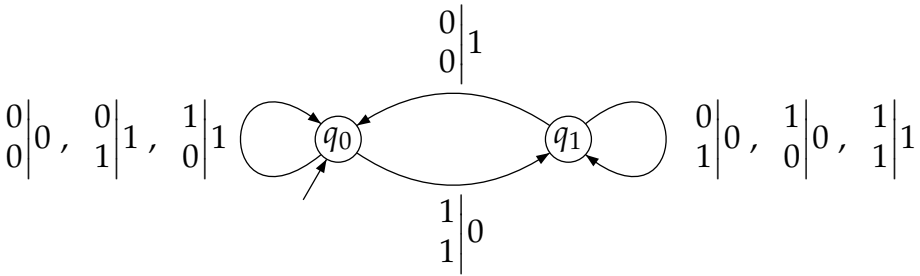
$$y(t+1) = F(x(t), y(t))$$

Automates de Mealy

Définition Un **automate de Mealy** est un tuple (Q, I, O, q_0, δ) où Q est l'ensemble fini des **états**, I et O sont les **alphabets** finis d'entrée et de sortie, $q_0 \in Q$ est l'**état initial** et $\delta : Q \times I \to Q \times O$ est la **fonction de transition** de l'automate.



Théorème Tout **automate de Mealy** peut être mis en œuvre par un **circuit séquentiel synchrone** composé de portes NON-ET et de registres mémoire.



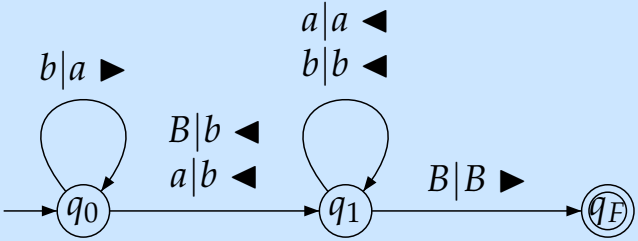
Addition de deux nombres binaires de longueur non bornée

Limites du calcul sans mémoire

Proposition Il n'est pas possible de **multiplier** deux entiers avec le même codage.

Il suffit de considérer la famille de produits $2^n \times 2^n = 2^{2n}$ et d'utiliser un argument de **pompage**.

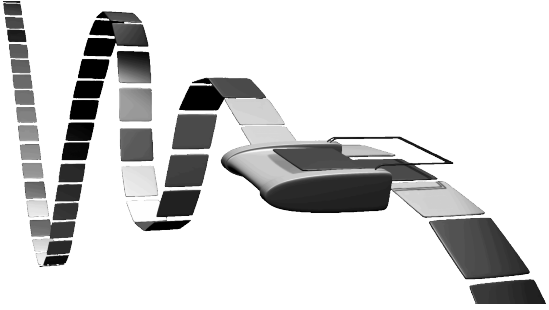
Il faut se laisser le **temps** et l'**espace** pour calculer.



2. Machines à mémoire non bornée

Machines de Turing

La **machine de Turing** classique : un **contrôle fini** couplé à un **ruban** biinfini muni d'une **tête** d'E/S mobile pointant sur une cellule.



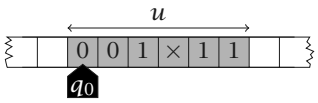
Formellement

Définition Une **MT** est un tuple $(Q, \Gamma, \Sigma, \delta, q_0, B, q_F)$ où

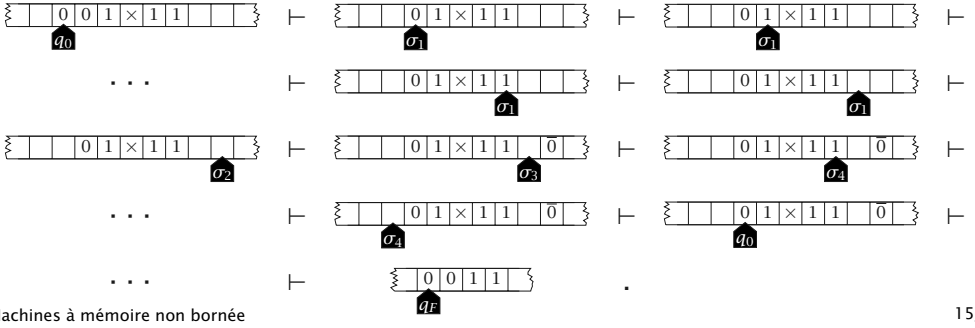
- Q est l'ensemble fini des **états** ;
- Γ est l'**alphabet** fini de **travail** ;
- $\Sigma \subseteq \Gamma$ est l'**alphabet** fini **d'entrée** ;
- $\delta : Q \times \Gamma \rightharpoonup Q \times \Gamma \times \{\blacktriangleleft, \blacktriangledown, \blacktriangleright\}$ est la **fonction de transition**, partielle ;
- $q_0 \in Q$ est l'**état initial** ;
- $B \in \Gamma \setminus \Sigma$ est le **symbole blanc** ;
- $q_F \in Q$ est l'**état d'acceptation** de la machine.

Une transition $\delta(q, a) = (q', b, \Delta)$ signifie :
« Dans l'état q, lorsque je lis le symbole a,
le remplacer par b, entrer dans l'état q' et se déplacer de Δ »

Configurations et étapes de calcul



Partant d'une **configuration initiale** où la tête est placée dans l'état q_0 au début du **mot d'entrée** u, la MT calcule en enchaînant les **transitions**.

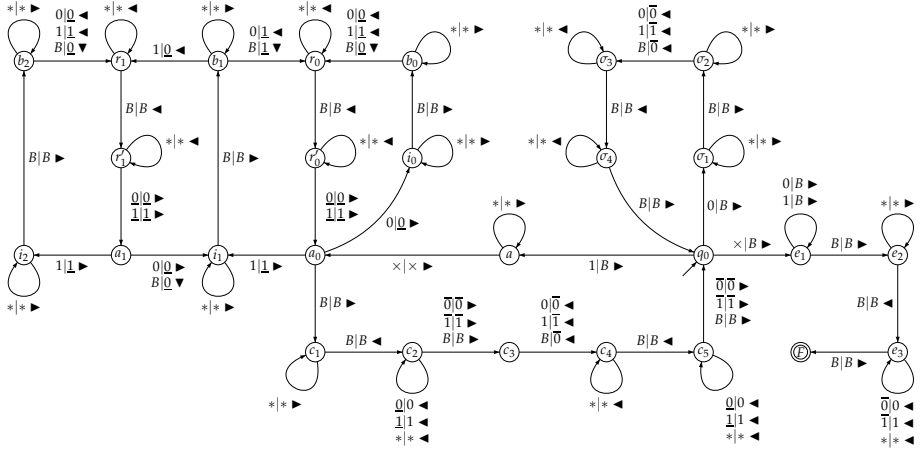


Fonctions Turing-calculables

Lorsque le calcul **termine**, si la machine est dans l'état acceptant q_F, la sortie est lue à droite de la tête jusqu'au premier symbole blanc.

Définition La **fonction partielle** $f_{\mathcal{M}} : \Sigma^* \rightharpoonup \Gamma^*$ calculée par une MT $\mathcal{M}$ est la fonction qui à une entrée associe sa sortie pour $\mathcal{M}$, lorsqu'elle est définie.

Définition Une **fonction partielle** $f : \Sigma^* \rightharpoonup \Gamma^*$ est **Turing-calculable** si elle est calculée par une machine de Turing.



Multiplication de deux nombres binaires de longueur non bornée codés bit de poids faible en tête

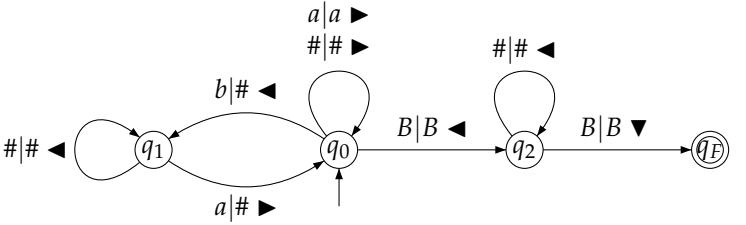
Langages récursifs

Pour raisonner sur le **calculable**, il est souvent pratique de se restreindre aux **prédicats calculables**, c'est-à-dire à une notion de **langage reconnaissable**.

Définition Le **langage $L_{\mathcal{M}}$ associé** à une MT $\mathcal{M}$ est l'ensemble des entrées pour lesquelles $\mathcal{M}$ termine dans l'**état acceptant**.

Définition Un langage est **recursivement énumérable** s'il est reconnu par une MT.

Définition Un langage est **récursif** s'il est reconnu par une MT **totale**.



MT totale reconnaissant les mots bien parenthésés sur $\{a, b\}$

Limites du calcul

Proposition Il existe des langages **non récursifs**.

Il suffit de compter :

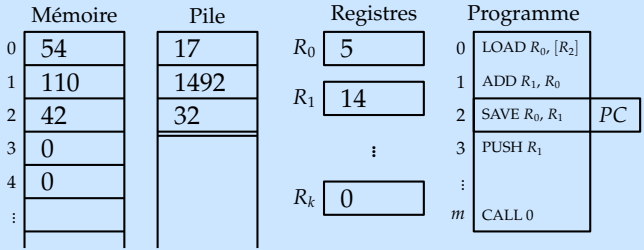
- les MT sont **dénombrables** ;
- les langages sur $\{a, b\}$ ne le sont **pas**.

Un modèle robuste

Le modèle des MT présenté semble très **ad hoc**.

Cependant, la classe des fonctions définies est **robuste** à tout un tas de **variations** :

- ruban mono-infini ;
- plusieurs rubans ;
- mémoire comme une file ;
- mémoire comme deux piles ;
- mémoire 2D ;
- mémoire collection de compteurs unaires ;
- ...



3. Calculer autrement

Thèse de Church, Turing, Kleene, Post *et al.*

Thèse de Church-Turing Toute fonction calculable par une méthode effective est Turing-calculable.

Théorème (Gandy 1980) Toute fonction **discrète** calculable par un dispositif **mécanique déterministe**, régit par les règles de la physique classique et qui satisfait les **hypothèses** suivantes, est Turing-calculable :

- homogénéité de l'espace ;
- homogénéité du temps ;
- densité bornée d'information dans l'espace ;
- vitesse bornée de propagation de l'information à travers l'espace ;
- quiescence initiale de l'espace de calcul sauf dans une zone bornée.

Turing-complétude

Un **modèle de calcul** décrit une **famille dénombrable** d'objets et la manière de les utiliser pour calculer.

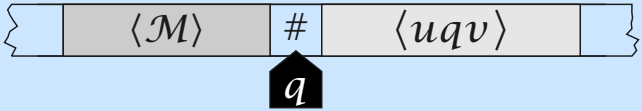
Une **fonction de codage acceptable** décrit une transformation raisonnable entre modèles pour coder les entrées et les sorties.

Définition Deux modèles de calcul se **simulent** entre eux s'ils calculent les même fonctions à un codage acceptable près.

Définition Un modèle de calcul est **Turing-complet** s'il est équivalent au modèle des machines de Turing.

Quelques exemples de modèles Turing-complets

- le λ-calcul de Church ;
- le modèle RAM ;
- votre langage de programmation favori ;
- les fonctions récursives de Herbrand/Gödel/Kleene ;
- les systèmes canoniques de Post ;
- les automates cellulaires ;
- . . .



4. Machines programmables

Changer de machine

L'**automate** au cœur de nos **ordinateurs** est fixé *in silico*.

Contrairement aux calculatrices de notre enfance à 4, 10, 100 fonctions, on ne change plus de machine pour avoir de nouvelles possibilités de calcul.

Les **programmes** sont des **données** comme les autres, **chargés** et **exécutés** et même **compilés** sur place.

Cette possibilité existe dans la plupart des modèles de calcul !

Machine de Turing universelle

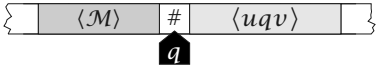
Théorème Il existe une machine de Turing **universelle** U qui **simule** toutes les machines de Turing.

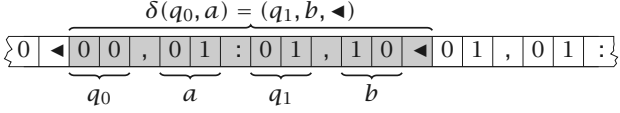
Pour toute machine de Turing $\mathcal{M}$ sur l'alphabet Σ et toute entrée $u \in \Sigma^*$, la machine U s'arrête sur l'entrée $\langle \mathcal{M}, u \rangle$ si et seulement si $\mathcal{M}$ s'arrête sur l'entrée u. Elle accepte si et seulement si $\mathcal{M}$ accepte et dans ce cas

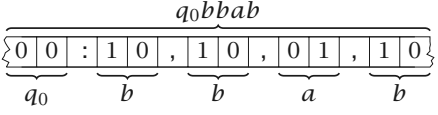
$$f_U\left(\langle \mathcal{M}, u \rangle\right) = \left\langle f_{\mathcal{M}}(u) \right\rangle .$$

Remarque Nécessite de préciser un **codage acceptable** des MT.

Principes de construction







Problème de l'arrêt

entrée : *une machine de Turing $\mathcal{M}$ et un mot u*
question : *est-ce que $\mathcal{M}$ s'arrête sur l'entrée u ?*

5. Indécidabilité et limites du calcul

Problème de décision

Définition Un **problème de décision** $\mathcal{P}$ est un prédicat sur un ensemble récursif d'entrées E, les instances du problème.

Le **langage associé** au problème est le langage des codages des **instances positives**, *i.e.* $L_{\mathcal{P}} = \{\langle x \rangle | x \in E \wedge \mathcal{P}(x)\}$.

Problème de l'arrêt

entrée : *une machine de Turing $\mathcal{M}$ et un mot u*
question : *est-ce que $\mathcal{M}$ s'arrête sur l'entrée u ?*

Définition Un problème de décision est **décidable** si le langage associé est récursif, **indécidable** sinon.

Un problème indécidable

Théorème Le **problème de l'arrêt** est **indécidable**.

Le langage associé est $K = \{\langle \mathcal{M}, u \rangle | \mathcal{M} \text{ s'arrête sur } u\}$.

Par l'absurbe, en supposant K reconnu par une MT totale $\mathcal{H}$, on construit une MT Δ qui sur l'entrée $\langle \mathcal{M} \rangle$:

1. exécute $\mathcal{H}$ sur l'entrée $\langle \mathcal{M}, \mathcal{M} \rangle$
2. si $\mathcal{H}$ accepte alors Δ **diverge** ;
3. si $\mathcal{H}$ rejette alors Δ **s'arrête**.

Par construction, la MT Δ possède un **codage** $\langle \Delta \rangle$. Étudions $\Delta(\langle \Delta \rangle)$!

Corollaire le langage K n'est **pas co-récursivement énumérable**.

Des problèmes indécidables

Une technique pour dériver de nombreux problèmes indécidables à partir d'un premier consiste à utiliser des **réductions** : des pré-ordres sur les langages qui **préservent la récursivité**.

Définition Soient $A \subseteq \Sigma^*$ et $B \subseteq \Gamma^*$ deux langages. Une **réduction** (many-one) de A à B est une fonction totale calculable $f : \Sigma^* \to \Gamma^*$ telle que

$$u \in A \Leftrightarrow f(u) \in B \qquad \forall u \in \Sigma^* \quad .$$

Le langage A **se réduit** au langage B, noté $A \leqslant_m B$ s'il existe une réduction de A à B.

Proposition La relation $\leqslant_m$ est une relation de pré-ordre.

Problème de l'arrêt sur l'entrée vide

entrée : *une machine de Turing* $\mathcal{M}$
question : *est-ce que* $\mathcal{M}$ *s'arrête sur l'entrée vide* ε *?*

Le langage associé est $K_0 = \{\langle \mathcal{M} \rangle | \mathcal{M}$ s'arrête sur $\varepsilon\}$.

Montrer que $K_0 \leqslant_m K$ et $K \leqslant_m K_0$

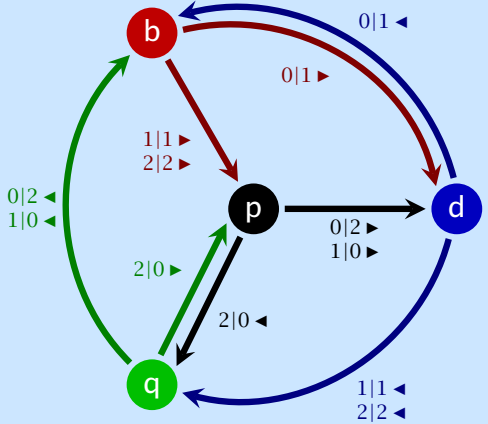
Théorème de Rice

Une **MT** est un objet **syntaxique** qui décrit la manière d'organiser un calcul.

Le **langage reconnu** par la machine est de nature **sémantique**, c'est le résultat du calcul.

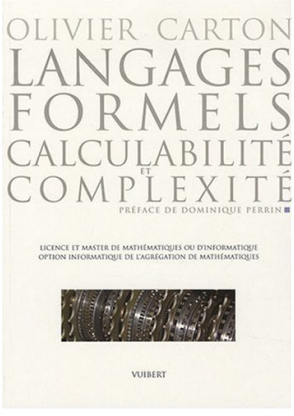
Le **théorème de Rice** montre que tout problème qui voudrait caractériser une propriété purement sémantique des machines est **indécidable** ou **trivial**.

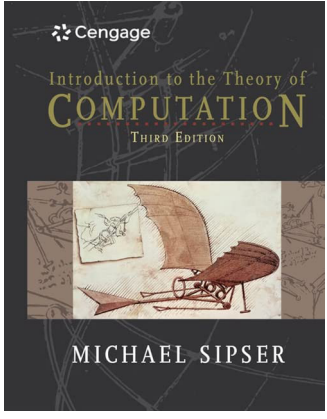
Remarque La démonstration se fait par **réduction**.



Pour aller plus loin

Un peu de lecture





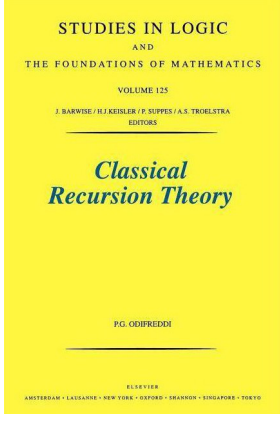


Table des matières