Chapitre 5 : Machines de Turing

Vincent Guisse
vincent.guisse@esisar.grenoble-inp.fr

Grenoble INP - ESISAR
3e année IR & C

5 avril 2022

Limites de nos précédents modèles de calcul

- Les automates finis déterministes reconnaissent les langages réguliers, mais pas $\{a^n b^n,\ n \in \mathbb{N}\}$ par exemple car leur mémoire est finie.
- Avec les automates finis non déterministes, on ne gagne rien.
- Les automates à pile ont une mémoire illimitée et reconnaissent les langages hors-contexte comme $\{a^n b^n,\ n \in \mathbb{N}\}$ (et les automates à pile déterministe reconnaissent les langages hors-contexte déterministe), mais pas $\{a^n b^n c^n,\ n \in \mathbb{N}\}$ par exemple, à cause de la limitation d'accès LIFO de la mémoire.

Le dernier modèle de calcul du cours, les machines de Turing déterministes, va bénéficier d'une mémoire sans limite comme les automates à pile, mais cette mémoire pourra être parcourue de manière plus souple.

Principe

Un concept abstrait d'ordinateur proposé par Alan Turin en 1936 :

- qui peut être dans un **nombre fini d'états**,
- qui dispose d'**une bande infinie** sur laquelle est écrit le mot en entrée, et qui sert ensuite de mémoire (lecture et écriture possibles sur la bande).
- dont l'**accès** à cette **mémoire** est **séquentiel** : seul le caractère qui vient d'être lu peut-être modifié, et la tête de lecture / écriture ne se déplace que d'un case vers la droite ou la gauche à chaque étape de calcul.

Machine de Turing déterministe : définition

$M = (Q, \Gamma, \Sigma, B, \delta, q_i, F)$ est une machine de Turing avec :

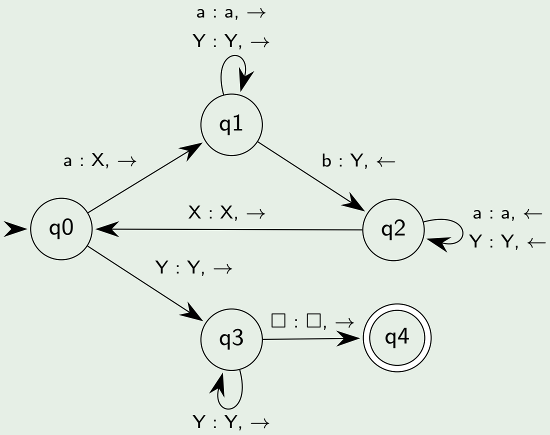
- Q un ensemble fini d'état,
- Γ l'alphabet de la bande,
- $\Sigma \subset \Gamma$ l'alphabet des mots en entrée,
- $B \in \Gamma \setminus \Sigma$ un caractère spécial "blanc",
- $\delta : \quad Q \times \Gamma \to Q \times \gamma \times \{\leftarrow, \rightarrow\}$ une fonction de transition, $(q, \sigma) \mapsto (p, \sigma', d)$
- $q_i \in Q$ l'état initial,
- $F \subset Q$ l'ensemble des états acceptant.

$$\delta(q, \sigma) = (p, \sigma', d)$$

signifie que si M est dans l'état q et que sa tête de lecture lit σ sur la bande, alors M passe dans l'état p, remplace σ par σ' sur la bande et effectue le déplacement $d \in \{\leftarrow, \rightarrow\}$.

Premier exemple

Le diagramme suivant spécifie une machine de Turing déterministe :



Que donne l'exécution sur le mot *aabb* ?

Définition : Configuration

Une configuration de la machine de Turing $M = (Q, \Gamma, \Sigma, B, \delta, q_i, F)$ est un triplet $(u, q, v) \in \Gamma^* \times Q \times \Gamma^*$.

Interprétation : la machine se trouve dans l'état q, la bande contient le mot uv et la tête de lecture est sur la première lettre de v.

Définition : relation de dérivation entre configurations

On dit que la configuration C' se dérive en une étape de la configuration C pour la machine M, et on note :

$$C \vdash_M C'$$

lorsqu'on obtient C' à partir de C en une transition de M. On note $\vdash_M^*$ la clôture transitive réflexive de cette relation.

Dans notre premier exemple, on a successivement :

$$(\varepsilon, q_0, aabb) \vdash_M (X, q_1, abb) \vdash_M (Xa, q_1, bb) \vdash_M (X, q_2, aYb)$$

Définition : Calcul

Le calcul de la machine de Turing $M = (Q, \Gamma, \Sigma, B, \delta, q_i, F)$ sur le mot $w \in \Sigma^*$ est la suite de configurations :

$$C_0 = (\varepsilon, q_i, w) \vdash_M C_1 \vdash_M C_2 \vdash_M \dots$$

obtenues par dérivations successives à partir de la configuration initiale $C_0 = (\varepsilon, q_i, w)$.

Il y a 3 possibilités mutuellement exclusives :

1. le calcul passe par un état acceptant,
2. le calcul s'arrête avant,
3. le calcul est infini et ne passe jamais par un état acceptant.

Définition : mot, langage accepté par une MT

Un mot w est accepté par par la machine de Turing $M = (Q, \Gamma, \Sigma, B, \delta, q_i, F)$ si et seulement si on a

$$(\varepsilon, q_i, w) \vdash_M^* (u, f, v), \text{ avec } f \in F$$

Le langage accepté par M est l'ensemble des mots acceptés par M.

Définition : langage décidé par une machine de Turing

Un langage L est décidé par un machine de Turing M lorsque L est le langage accepté par M et que M n'a pas d'exécution infinie.

Exercice

1. Décrire des machines de Turing acceptant les langages suivants :
   1. $L_1 = \{a^n b^n c^n, \ n \in \mathbb{N}\}$
   2. $L_2 = \left\{ww^R, \ w \in \{a, b\}^*\right\}$
2. Les langages L_1 et L_2 sont-ils décidés par les machines de Turing proposées ?
3. Les complémentaires des langages L_1 et L_2 sont-ils décidables ?

Théorème : machines de Turing universelles

Il existe des machines de Turing prenant en entrée une chaîne "M$w", où M décrit une machine de Turing et w un mot, et qui acceptent si et seulement si la machine de Turing codée par M accepte le mot w.

De telles machines de Turing sont appelées machine de Turing universelles.

Thèse de Turing-Church

« Les langages reconnus par une procédure effective sont ceux décidés par une machine de Turing. »

Commentaires

- Nature épistémologique de cet énoncé :
  - pas un théorème : la notion de procédure effective n'est pas définie
  - plutôt une loi au sens des sciences expérimentales : non démentie par les expériences déjà faites (les autres modèles de calculs essayés)
- Arguments : les autres modèles de calculs ne décident pas davantage de langages :
  - On peut rajouter des rubans, ou du non déterminisme au modèle des machines de Turing, on sait construire une machine de Turing déterministe à un ruban qui décide le même langage,
  - Lambda calcul, fonctions récursives, machines RAM, etc. décident les mêmes langages

Définition : langages récursivement énumérable

Un langage est récursivement énumérable (dans RE, semi-decidable) si et seulement si il est accepté par une machine de Turing.

Définition : langages récursifs

Un langage est récursif (dans R, décidable) si et seulement si il est décidé par une machine de Turing.

On a donc par définition l'inclusion (on verra qu'elle est stricte) des langages récursifs dans les langages récursivement énumérables.

Théorème

$L \in RE \Leftrightarrow$ il existe une machine de Turing qui énumère les mots de L, c'est à dire qui les écrits successivement (éventuellement en se répétant) sur la bande.

L'argument de la cardinalité

L'ensemble des machines de Turing est dénombrable : elle peuvent être encodées avec des mots. L'ensemble des langages ne l'est pas. Il existe donc des langages non récursivement énumérable.

L'argument diagonal

Soit $(M_i)_{i\in\mathbb{N}}$ une énumération des machines de Turing, et $(w_i)_{i\in\mathbb{N}}$ une énumération des mots. Alors :

1. $D = \{w_i,\ M_i \text{ accepte } w_i\}$ est dans RE.
2. $\overline{D} = \{w_i,\ M_i \text{ n'accepte pas } w_i\}$ n'est pas dans RE.
3. $\overline{D} \in RE \setminus R$.

Preuve

1. On énumère les couples (M_j, w_j) jusqu'à reconnaître w_i, on lance alors l'exécution de M_i sur w_i et on accepte si et seulement si celle-ci accepte.
2. Par l'absurde, si $\overline{D} \in RE$, soit M_j la MT qui l'accepte, alors $w_j \in \overline{D} \Leftrightarrow w_j \in D$, contradiction.
3. Si D était dans R, alors $\overline{D}$ le serait aussi.

Principe d'une réduction

On dispose d'un problème de référence P prouvé « difficile », et on veut montrer qu'un nouveau problème P' est « au moins aussi difficile ».

- On suppose qu'on a une solution pour P'
- On montre que la résolution de toute instance de P peut se ramener « facilement » à celle d'une instance de P'.

On obtient ainsi une preuve que P' est au moins aussi difficile que P puisque notre résolution utilise la solution pour P', et que toute solution générale pour P a été prouvée difficile.

Théorème : $LU \notin R$

Le langage universel

$$LU = \{M\$w,\ M \text{ accepte } w\}$$

n'est pas décidable : $LU \notin R$.

Preuve

On fait une réduction du problème P du langage D au problème P' de la décision de LU.
Supposons qu'un machine de Turing M décide LU. Alors soit w une instance de P :

1. On énumère les couples $M_i\$w_i$ jusqu'à reconnaître w.
2. On utilise notre hypothétique solution à P' sur l'instance $M_i\$w_i$.

Ainsi, D est décidable, contradiction. Donc $LU \notin R$.

Théorème : le problème de l'arrêt n'est pas décidable

Le langage du problème de l'arrêt

$$H = \{M\$w,\ M \text{ s'arrête sur } w\}$$

n'est pas décidable : $H \notin R$.

preuve

On fait une réduction du problème P du langage universel LU au problème $P' = H$ de l'arrêt.

Preuve de $H \notin R$, fin

Par l'absurde, supposons qu'une machine de Turing M_H décide H. Soit $M\$w$ une instance de P :

1 On lance M_H sur l'entrée $M\$w$ pour savoir si l'exécution de M sur w se termine.

2.a Si la réponse est non, on refuse l'entrée (exécution infinie, w n'est pas accepté par M).

2.b Si la réponse est oui, on lance l'exécution de M sur w, et à la fin de l'exécution on accepte si et seulement si M a accepté w.

Ainsi $LU \in R$, contradiction. Donc $H \notin R$.