914 : Décidabilité et Indécidabilité. Exemples

Idée : Pour résoudre un problème informatique, on essaie de trouver un algorithme... peut-on toujours y arriver ?

I - Problèmes décidables : définition(s)

1) Notion de Problème [Car p 127]

Déf 1 : Un Problème de décision est la donné d'un ensemble d'instances E et de $P \subset E$ pour lesquelles la réponse est "oui".

Par un codage $x \in E \mapsto \langle x \rangle \in \Sigma^*$, on se ramène au cas où P est un langage : $L_P = \{\langle x \rangle, x \in P\}$.

2) Machines de Turing et calculabilité [Car p 145]

On considère des MT déterministes, contenant les états q_0 (initial) q_+ (accepteur) q_- (puit).

Déf 2 : $L \subset \Sigma^*$ est décidé par M si $L = \mathcal{L}(M)$ et si M n'a pas d'exécution infinie ;

Remarque 3 : L est décidable ssi $\bar{L}$ l'est.

Exemple 4 : Les problèmes de décision sur les graphes sont finis donc décidables. faux

Remarque 5 : Une machine de Turing sans exécution infinie permet aussi de calculer des fonctions $f : \mathbb{N}^? \to \mathbb{N}^k$.

On peut montrer que f est calculable ssi $L_f = \{\langle n, f(n) \rangle, n \in \mathbb{N}^?\}$ est décidable

3) Robustesse du modèle

D'autres notions de calculabilité ont été développées en parallèle des Machines de Turing. On peut citer :

- Les fonctions récursives
- Le λ-calcul
- Le modèle des machines RAM

culture !

Théorème 6 : Une fonction est récursive ssi elle est calculable par MT.

Thèse de Turing-Church : Tout modèle cherchant à définir ce qui est calculable "de manière effective" sera équivalent à ceux-ci

II - De la décidabilité à l'indécidabilité

1) En théorie des langages

On considère le problème du mot sur un automate (au sens large) : $LU = \{\langle \mathcal{A}, w \rangle, w \in \mathcal{L}(\mathcal{A})\}$ Chomsky

Prop 7 :
- Sur les automates finis, LU est décidable
- Sur les automates à pile, LU est encore décidable
- Sur les machines de Turing linéairement bornées, LU est décidable [Car, p 157]
- En revanche, le problème LU général, sur les machines de Turing, est indécidable. [Car, p 146]

expressivité

décidabilité

Remarque 8 : Il existe une machine de Turing dite Universelle telle que $LU = \mathcal{L}(M)$, mais elle va avoir des exécutions infinies.

Théorème 9 (de Rice) [Car / wol]
Toute problème non trivial sur les langages acceptés par MT est indécidable

Conséquence (culture) : Montrer qu'un programme va fonctionner de la façon qu'on veut sur toute entrée est indécidable de façon générale. On doit faire appel à des heuristiques adaptées au programme...

2) En logique du 1^{er} ordre

Définition 10 : Soit T une théorie du premier ordre.
- T est dite récursive si $T \subset \Sigma^*$ est un langage décidable
- T est dite décidable si $L_T = \{F \in \Sigma^* / T \models F\}$ est décidable

alors Thm(T) récursif

Proposition 11 : • On considère des théories finies, ayant un unique symbole de prédicat $=$, on ajoute les fonctions $+$ et $\times$
- la théorie de l'égalité est décidable
- L'Arithmétique de Presburger est décidable [DEV]
- L'Arithmétique de Peano est indécidable

Théorème 12 (Church)
Une théorie égalitaire possédant au moins un autre prédicat binaire est indécidable (admis)

- Ici aussi, lorsqu'on augmente l'expressivité de notre modèle, on est confronté ~~plus~~ rapidement à l'indécidabilité.

III - Démontrer l'Indécidabilité

Remarque 3bis : L est indécidable $\iff \overline{L}$ est indécidable

On prouve que $\overline{LU}$ est indécidable par un argument diagonal [Car]

Exemple 14 : On peut énumérer les mots de Σ^* et les MT.
Soit $L_0 := \{w_i \in \Sigma^* / w_i \notin \mathcal{L}(M_i)\}$, alors L_0 n'est le langage d'aucune MT. [Wol]

1) Le concept de réduction [Car/Wol]

Soit L_1 indécidable et L_2 dont on veut prouver l'indécidabilité. On dit que L_1 se réduit à L_2 si l'on peut théoriquement construire une MT décidant L_1 à partir d'une machine décidant L_2.

Remarque 15 : Cela signifie que L_2 "contient" L_1, ou que c'est un problème "plus général" que celui ~~de~~ L_1.
On dit que L_2 est plus dur que L_1.
Deux problèmes qui se réduisent l'un à l'autre sont dits équivalents.

Exemple 16 L_0 et $\overline{LU}$ sont équivalents, $\overline{L_0}$ et LU aussi. [Car + Wol].

2) Un exemple bien utile [Car] p150

Définition 17 • Une instance du Problème de Correspondance de Post est la donnée de m paires (u_i, v_i) de mots sur Σ.
- Une instance est positive s'il existe $(i_1, \dots i_n) \in [\![1, m]\!]^n, n \in \mathbb{N}$ tels que $u_{i_1} \dots u_{i_n} = v_{i_1} \dots v_{i_n}$.

Définition 18 : Le problème PCPM est la variante de PCP qui impose $i_1 = 1$

Proposition 19 : PCP et PCPM sont équivalents

Théorème 20 : Le Problème de Correspondance de Post (Modifié) est indécidable.

IV - Problèmes Indécidables

1) Liés aux machines de Turing [Wol p146-188]

- Le problème de l'arrêt $H = \{\langle M, w\rangle,\ M \text{ s'arrête sur } w\}$

et ses variantes : arrêt sur mot vide $\{\langle M\rangle,\ M \text{ s'arrête sur } \varepsilon\}$, arrêt existentiel $\{\langle M\rangle,\ \exists x \in \Sigma^*,\ M \text{ s'arrête sur } x\}$ et universel...

- Les autres problèmes "classiques" entrent dans le cadre du Thm de Rice.

2) Liés aux grammaires algébriques [Car p153 + cours]

Théorème 2.1 : Déterminer si l'intersection de deux langages algébriques est vide est un problème indécidable [DEV]

Conséquence : problèmes indécidables :

- inclusion, égalité entre deux langages algébriques
- Déterminer si l'intersection est algébrique
- Déterminer si une grammaire est ambigüe

3) En logique du 1^er^ ordre

- Le calcul des prédicats $(T = \emptyset)$ est indécidable !
- [Baader] : Une théorie équationnelle dont le système de réécriture est confluent et terminant est décidable.

Mais déterminer si un système est terminant, ou confluent, sont deux problèmes indécidables !

On peut aussi citer pour la culture le 10° Problème de Hilbert : Soit P un polynôme à coefficients dans $\mathbb{Z}$, P admet-il une racine entière ?

ou d'autres problèmes de dominos dérivés de Post.

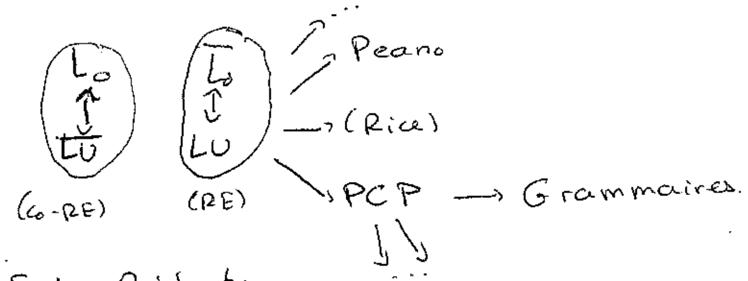


Fig 1 : Réductions

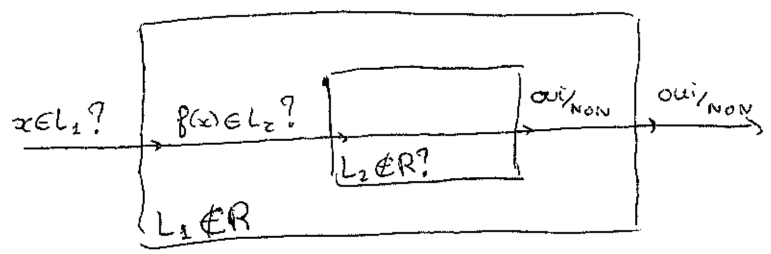
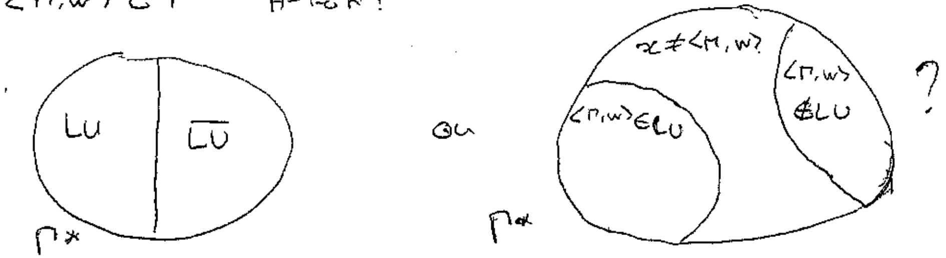


Fig 2 : Principe de Réduction.

NB : Soit Γ l'alphabet qui code les machines et les mots pour LU : $\langle M, w\rangle \in \Gamma^*$. A-t-on :



Arithmétique de Presburger

Th : (Presburger)

La théorie au premier ordre des entiers munis de l'addition est décidable.

<u>Preuve:</u> Soit φ une formule de l'arithmétique de Presburger. On pose L_φ l'ensemble des instances positives de φ.

<u>Montrons que L_φ est un langage rationnel:</u>

- Supposons que φ est sous forme prénexe :

$$\varphi = Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \psi$$

Alors pour $0 \leq k \leq n$, on définit :

$$\varphi_k = Q_{k+1} x_{k+1} \cdots Q_n x_n \psi$$

où $\varphi_0 = \varphi$ et $\varphi_n = \psi$

Dans φ_k, les variables $x_1, \dots, x_k$ sont libres, on écrira donc le cas échéant : $\varphi_k(x_1, \dots, x_k)$

- <u>Codage</u> : Nous allons coder des k-uplet d'entiers en binaire.

$$\begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_k \end{pmatrix} = \begin{pmatrix} \lambda_{1,0} \\ \vdots \\ \ \end{pmatrix} \begin{pmatrix} \lambda_{1,1} \\ \vdots \\ \ \end{pmatrix} \cdots \begin{pmatrix} \lambda_{1,n} \\ \vdots \\ \ \end{pmatrix}$$

où $\lambda_{i,j}$ est le j-ème bits dans l'écriture en base 2 de λ_i. Le bit de poid faible est à gauche.

On peut toujours compléter la suite avec $\begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$ à droite.

On prend comme codage, la séquence de longueur minimale.

• Posons $X_k = \left\{ \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \;\middle|\; \varphi_k(x_1, \dots, x_k) \text{ est vraie} \right\}$.

On procède par récurrence descendante sur k :

<u>Init</u> : Construisons A_n.

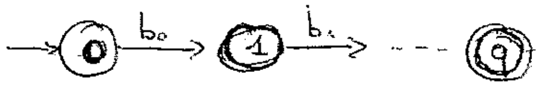
φ est une combinaison booléenne de formule atomique de la forme :

- $x_i = cte$ - $x_i = x_j$ - $x_i + x_j = x_p$

De plus, pour φ_1 et φ_2 des formules, on a :
- $L_{\varphi_1 \wedge \varphi_2} = L_{\varphi_1} \cap L_{\varphi_2}$
- $L_{\varphi_1 \vee \varphi_2} = L_{\varphi_1} \cup L_{\varphi_2}$
- $L_{\neg\varphi} = \overline{L_\varphi}$

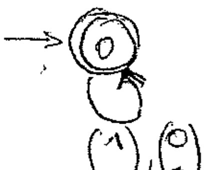
La classe des langages rationnels étant close pour $\cup$, $\cap$, et $\overline{}$, il suffit de montrer que X_k est rationnel en supposant que φ_k est une formule atomique.

<u>$x_i = n$</u> :

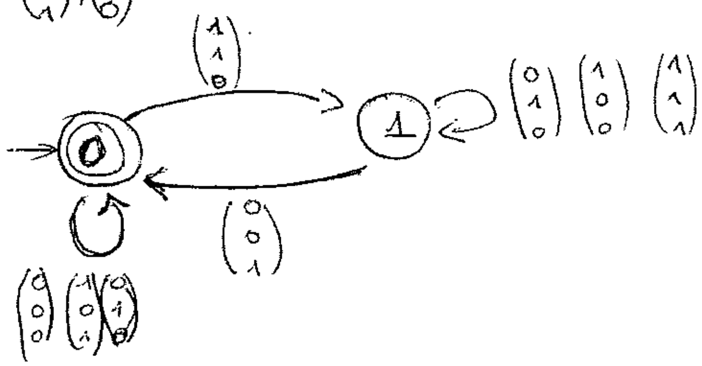


où b_i est le bit de poids i dans l'écriture en base 2 de n.

<u>$x_i = x_j$</u> :



<u>$x_i + x_j = x_p$</u> :



<u>Récurrence</u> : Supposons construit l'automate A_{k+1}

on a : $\varphi_k = Q_{k+1} x_{k+1} \varphi_{k+1}$

Si $Q_{k+1} = \forall$, alors $\varphi_k = \neg \exists\, x_{k+1} \neg \varphi_{k+1}$

Comme la classe des langages rationnels est close par complémentation
Il suffit de faire la preuve dans le cas où $Q_{k+1} = \exists$

On considère l'automate A_{k+1} :

- A_k a les mêmes états que A_{k+1}. Il a également les mêmes états initiaux.
- Si $q \xrightarrow{\begin{pmatrix} a_1 \\ \vdots \\ a_k \\ a_{k+1} \end{pmatrix}} q'$ est une transition de A_{k+1}, alors $q \xrightarrow{\begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix}} q'$ est une transition de A_k
- Pour tout état final q de A_{k+1}, les états $\tilde{q}$ tq $\tilde{q} \xrightarrow{\begin{pmatrix} 0 \\ | \\ 0 \end{pmatrix}^* \begin{pmatrix} 0 \\ | \\ 0 \\ 1 \end{pmatrix}} q$ dans A_{k+1} sont des états finaux de A_k.

<u>Conclusion</u> : A_0 est un automate qui accepte au moins un mot si et seulement si φ est vraie.

De plus, le problème de savoir si "le langage reconnu par un automate fini est vide" est décidable.

Donc l'arithmétique de Presburger est décidable.

Th : Le problème de savoir si $L_G(S) \cap L_{G'}(S) = \emptyset$ pour deux grammaires G et G' données est indécidable.

Preuve : Nous allons faire une réduction de ce problème au problème de correspondance de Post que nous admettrons indécidables.

Soit $\begin{pmatrix} u_1 \\ v_1 \end{pmatrix}, \dots, \begin{pmatrix} u_m \\ v_m \end{pmatrix}$ une instance au problème de Post, avec u_i des mots sur Σ.

Soit $A = \{a_1, \dots, a_m\}$ tq $A \cap \Sigma = \emptyset$.

On pose alors : $L_u = \{ u_{i_1} \dots u_{i_n} a_{i_n} \dots a_{i_1} \mid n > 0,\ 1 \leq i_1, \dots, i_n \leq m \}$
sinon $\varepsilon \in L_u \cap L_v$

Ce nouveau langage est algébrique car engendré par

$G : S \longrightarrow \sum_{i=1}^{m} u_i S' a_i$ ~~+ε~~ pour enlever ε

$S' \longrightarrow S + \varepsilon$

De même on définit $L_v = \{ v_{i_1} \dots v_{i_n} a_{i_n} \dots a_{i_1} \mid n \geq 0,\ 1 \leq i_1, \dots, i_n \leq m \}$

Supposons que $L_u \cap L_v \neq \emptyset$, alors $\exists w \in L_u \cap L_v$.

donc $\exists\, i_1, \dots, i_p$ tq $w = u_{i_1} \dots u_{i_p} a_{i_p} \dots a_{i_1}$

$\exists\, j_1, \dots, j_q$ tq $w = v_{j_1} \dots v_{j_q} a_{i_q} \dots a_{i_1}$

On a donc, en lisant la fin de w, que $p = q$,

et que $\forall k \in [\![0, p]\!]$, $i_k = j_k$

et donc $\begin{pmatrix} u_{i_1} \\ v_{i_1} \end{pmatrix}, \ldots, \begin{pmatrix} u_{i_p} \\ v_{i_p} \end{pmatrix}$ est une solution

au problème de correspondance de Post.

Réciproquement, si $\begin{pmatrix} u_{i_1} \\ v_{i_1} \end{pmatrix}, \ldots, \begin{pmatrix} u_{i_p} \\ v_{i_p} \end{pmatrix}$ est une

solution au problème de Post, alors

$w = u_{i_1} \ldots u_{i_p} a_{i_p} \ldots a_{i_1} = v_{i_1} \ldots v_{i_p} a_{i_p} \ldots a_{i_1}$

donc $w \in L_u \cap L_v$ et $L_u \cap L_v \neq \emptyset$.

Donc on ne peut pas décider du vide de $L_u \cap L_v$.

Or si on pouvait décider du vide entre deux langages

algébriques par deux grammaires quelconque, on saurait

décider du vide de $L_u \cap L_v$, pour u et v donnés.

Donc on ne peut pas décider du vide de l'intersection

entre deux langages algébriques.