IF228 - Calculabilité et Complexité

Cours #4 : réduction, théorème de Rice

Frédéric Herbreteau
frederic.herbreteau@bordeaux-inp.fr
(d'après les supports de Corentin Travers)

8 janvier 2025

Réductions

Réduction : 1er exemple

PAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n pair ?

Réduction : 1er exemple

PAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n pair ?

IMPAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n impair ?

Décidé par M_{IMPAIR}

Réduction : 1er exemple

PAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n pair ?

IMPAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n impair ?

Décidé par M_{IMPAIR}

- Soit M_{PAIR} qui sur entrée $n \in \mathbb{N}$:
  1. calcule $n + 1$ à partir de n
  2. simule M_{IMPAIR} sur $n + 1$
  3. accepte n si M_{IMPAIR} accepte $n + 1$, et rejette sinon

Réduction : 1er exemple

PAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n pair ?

IMPAIR

ENTRÉE : $n \in \mathbb{N}$
QUESTION : n impair ?

Décidé par M_{IMPAIR}

- Soit M_{PAIR} qui sur entrée $n \in \mathbb{N}$:
  1. calcule $n+1$ à partir de n
  2. simule M_{IMPAIR} sur $n+1$
  3. accepte n si M_{IMPAIR} accepte $n+1$, et rejette sinon

- M_{PAIR} décide *PAIR* car :
  - n est pair **si et seulement si** $n+1$ est impair
  - et $r(n) = n+1$ est **calculable**
  - et *IMPAIR* est décidé par M_{IMPAIR}

Réduction : 2ème exemple

SUDOKU

ENTRÉE : une grille G de Sudoku
QUESTION : G admet-elle une solution ?

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

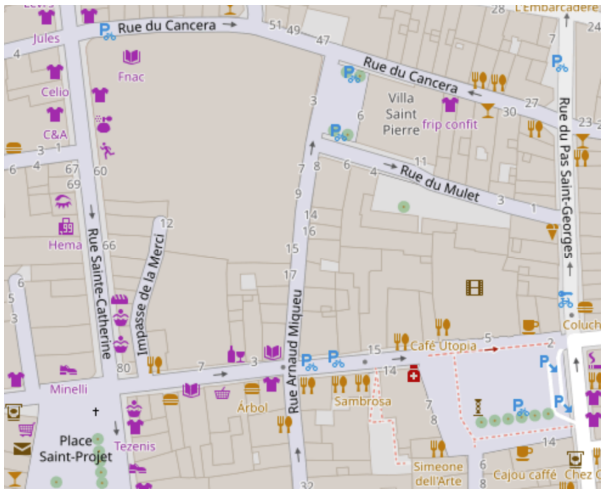
SAT

ENTRÉE : ϕ une formule de logique propositionnelle
QUESTION : ϕ est-elle satisfiable ?

Réduction : 3ème exemple

TRAJET

ENTRÉE : un plan P et deux points A et B
QUESTION : existe-t-il un chemin de A à B dans P ?



Exercice :

ENTRÉE :
QUESTION :

Réduction many-one

Soient A et B deux langages sur $\{0,1\}$.

Définition

Une **réduction many-one** de A à B est une fonction totale $r : \{0,1\}^* \to \{0,1\}^*$ telle que :

- r est **calculable**
- et pour tout mot $w \in \{0,1\}^*$, $w \in A \iff r(w) \in B$

Réduction many-one

Soient A et B deux langages sur $\{0,1\}$.

Définition

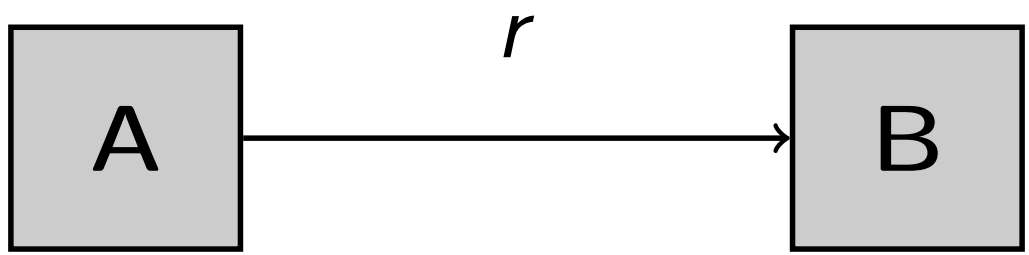
Une **réduction many-one** de A à B est une fonction totale $r : \{0,1\}^* \to \{0,1\}^*$ telle que :

- r est **calculable**
- et pour tout mot $w \in \{0,1\}^*$, $w \in A \iff r(w) \in B$

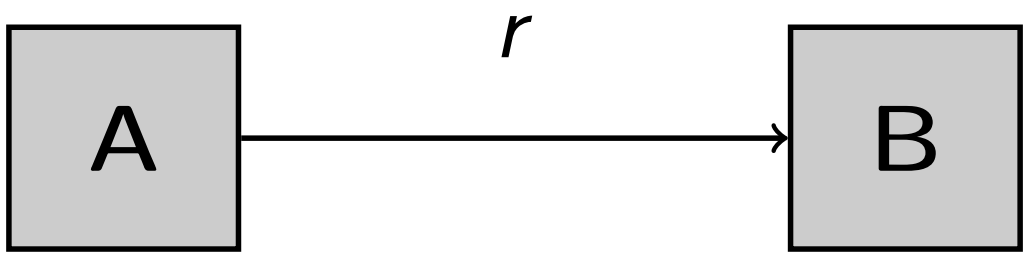
Exemple : réduction many-one de SUDOKU à SAT

- sur entrée G, r calcule Φ_G
- telle que G a une solution ssi Φ_G satisfiable

Réduction : montrer la décidabilité

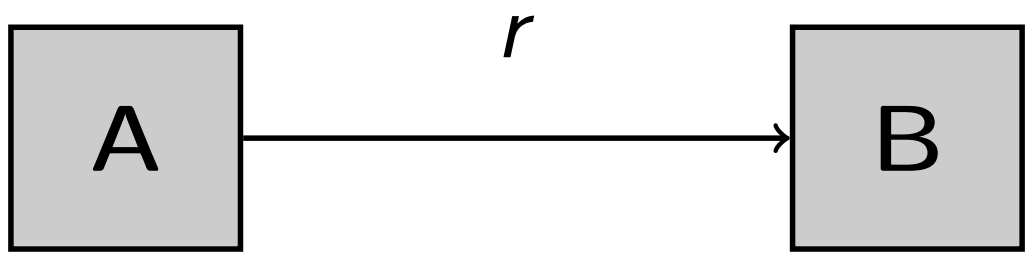


Réduction : montrer la décidabilité



décidable

Réduction : montrer la décidabilité



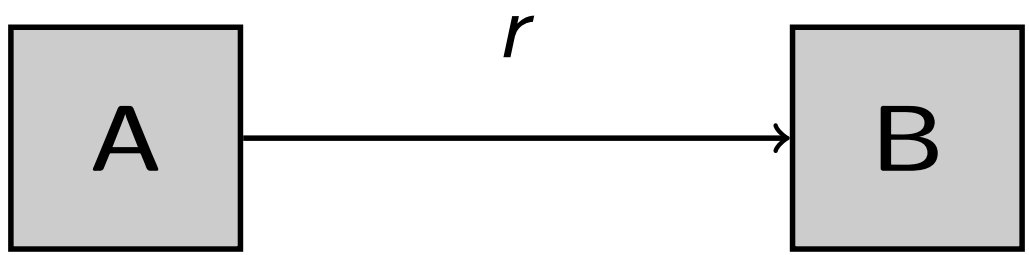
décidable

Théorème

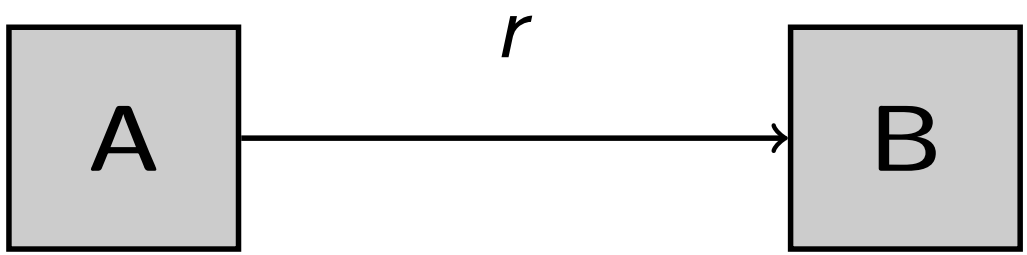
Si A ***se réduit à*** *B et B* ***est décidable****, alors A* ***est décidable****.*

Preuve

Réduction : montrer l'indécidabilité

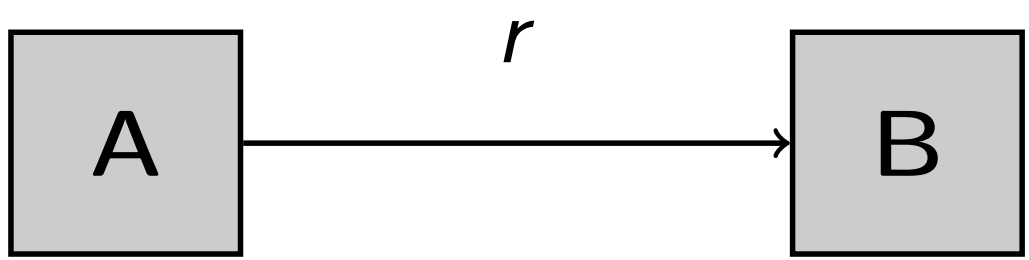


Réduction : montrer l'indécidabilité



indécidable

Réduction : montrer l'indécidabilité



indécidable

Théorème

Si A ***se réduit à*** *B et A* ***est indécidable****, alors B* ***est indécidable****.*

Preuve

Vers d'autres formes de réduction

- La réduction many-one permet de montrer la **décidabilité** et l'**indécidabilité**.
- La réduction de A à B induit un **algorithme pour** A :
  1. sur entrée w, exécuter la machine M_r qui calcule $r(w)$
  2. puis, exécuter un algorithme M_B pour B sur $r(w)$
  3. accepter/rejetter w en utilisant $w \in A \iff r(w) \in B$.
- Si r calculable **en temps polynomial**, alors A et B sont de **complexité similaire** (cf. cours suivants)
- **Limite forte :** M_A **ne peut que** : (1) calculer r(w), puis (2) exécuter M_B sur $r(w)$, et (3) accepter/rejeter w.

Turing-réduction

Soient A et B deux langages sur $\{0, 1\}$.

Un **oracle** pour B est une procédure qui pour tout mot $w \in \{0, 1\}^*$ décide si $w \in B$.

Turing-réduction

Soient A et B deux langages sur $\{0,1\}$.

Un **oracle** pour B est une procédure qui pour tout mot $w \in \{0,1\}^*$ décide si $w \in B$.

NB :

- oracle $\neq$ **procédure effective**

Turing-réduction

Soient A et B deux langages sur $\{0,1\}$.

Un **oracle** pour B est une procédure qui pour tout mot $w \in \{0,1\}^*$ décide si $w \in B$.

NB :

- oracle $\neq$ **procédure effective**
- On note $M[B]$ une MT qui **utilise** un oracle pour B

Turing-réduction

Soient A et B deux langages sur $\{0,1\}$.

Un **oracle** pour B est une procédure qui pour tout mot $w \in \{0,1\}^*$ décide si $w \in B$.

NB :

- oracle $\neq$ **procédure effective**
- On note $M[B]$ une MT qui **utilise** un oracle pour B

Définition (Turing-réduction)

A **se réduit à** B, noté $A \leq_T B$, si étant donné un oracle pour B, il existe une machine de Turing $M[B]$ qui décide A.

Turing-réduction

Soient A et B deux langages sur $\{0,1\}$.

Un **oracle** pour B est une procédure qui pour tout mot $w \in \{0,1\}^*$ décide si $w \in B$.

NB :

- oracle $\neq$ **procédure effective**
- On note $M[B]$ une MT qui **utilise** un oracle pour B

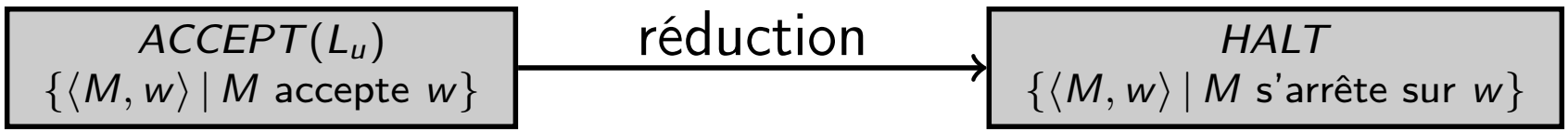
Définition (Turing-réduction)

A **se réduit à** B, noté $A \leq_T B$, si étant donné un oracle pour B, il existe une machine de Turing $M[B]$ qui décide A.

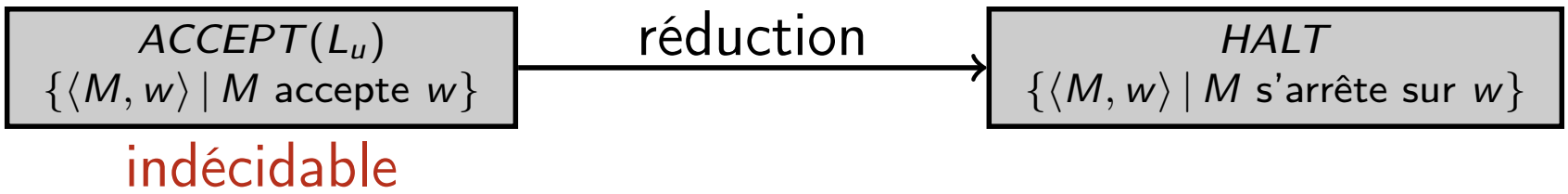
Théorème

*Si $A \leq_T B$ et A **est indécidable**, alors B **est indécidable**.*

Exemple

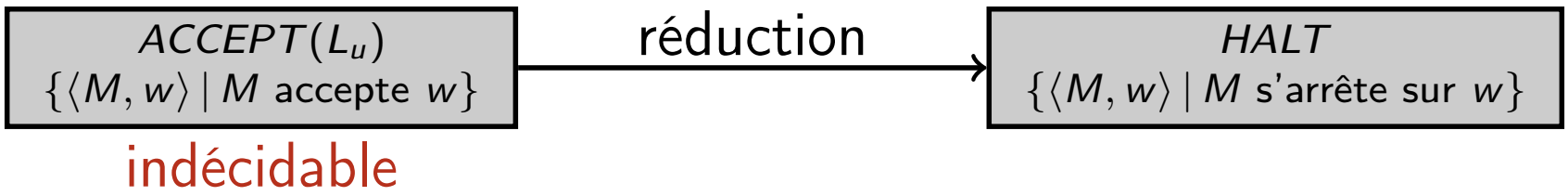

$ACCEPT(L_u)$
$\{\langle M, w\rangle \mid M \text{ accepte } w\}$
réduction
$HALT$
$\{\langle M, w\rangle \mid M \text{ s'arrête sur } w\}$

Exemple



indécidable

Exemple



indécidable

Théorème

Le langage HALT est indécidable.

Preuve

Théorème de Rice

Propriété non-triviale

Soit $L(M)$ le **langage accepté** par M.

Définition

Une **propriété** P est un ensemble de descriptions de machines de Turing tq si $L(M_1) = L(M_2)$ alors $\langle M_1 \rangle \in P \Leftrightarrow \langle M_2 \rangle \in P$.

Propriété non-triviale

Soit $L(M)$ le **langage accepté** par M.

Définition

Une **propriété** P est un ensemble de descriptions de machines de Turing tq si $L(M_1) = L(M_2)$ alors $\langle M_1 \rangle \in P \Leftrightarrow \langle M_2 \rangle \in P$.

Une propriété est **non-triviale** si elle contient au moins une, mais pas toutes les descriptions de MT

Propriété non-triviale

Soit $L(M)$ le **langage accepté** par M.

Définition

Une **propriété** P est un ensemble de descriptions de machines de Turing tq si $L(M_1) = L(M_2)$ alors $\langle M_1 \rangle \in P \Leftrightarrow \langle M_2 \rangle \in P$.

Une propriété est **non-triviale** si elle contient au moins une, mais pas toutes les descriptions de MT

Exemples (corrects ou incorrects) :

- INFINITY : l'ensemble des $\langle M \rangle$ tq $L(M)$ est infini
- REGULAR : l'ensemble des $\langle M \rangle$ tq $L(M)$ est régulier
- 3-STATES : l'ensemble des $\langle M \rangle$ tq M a 3 états
- NONE : l'ensemble vide des descriptions de MT

Théorème de Rice (1951)

$RICE_P$ où P est une propriété

ENTRÉE : la description $\langle M \rangle$ d'une MT M
QUESTION : M satisfait-elle P, i.e. est-ce que $\langle M \rangle \in P$?

Théorème de Rice (1951)

$RICE_P$ où P est une propriété

ENTRÉE : la description $\langle M \rangle$ d'une MT M
QUESTION : M satisfait-elle P, i.e. est-ce que $\langle M \rangle \in P$?

Théorème

Toute propriété ***non-triviale*** *P est* ***indécidable***

Théorème de Rice (1951)

$RICE_P$ où P est une propriété

ENTRÉE : la description $\langle M \rangle$ d'une MT M
QUESTION : M satisfait-elle P, i.e. est-ce que $\langle M \rangle \in P$?

Théorème

Toute propriété ***non-triviale*** *P est* ***indécidable***

Par réduction depuis HALT :

HALT

ENTRÉE : la description $\langle M, x \rangle$ d'une MT M et d'un mot x
QUESTION : M s'arrête-t'elle sur x ?

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y
- *Alors :*
  - M s'arrête sur $x \implies L(N_{M,x}) =$

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y
- *Alors :*
  - M s'arrête sur $x \implies L(N_{M,x}) = L(M_\alpha) \in P$

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y
- *Alors :*
  - M s'arrête sur $x \implies L(N_{M,x}) = L(M_\alpha) \in P$
  - M ne s'arrête pas sur $x \implies L(N_{M,x}) =$

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y
- *Alors :*
  - M s'arrête sur $x \implies L(N_{M,x}) = L(M_\alpha) \in P$
  - M ne s'arrête pas sur $x \implies L(N_{M,x}) = \emptyset \notin P$

HALT se réduit à $RICE_P$ (1)

Preuve.

Hypothèse : $L(M_\alpha) \neq \emptyset$ pour tout $\alpha \in P$, sinon prendre $\overline{P}$.

- Soit $\alpha \in P$ quelconque et M_α la machine correspondante
- À toute entrée $\langle M, x \rangle$ de HALT, on associe la MT $N_{M,x}$ qui sur entrée y :
  1. simule l'exécution de M sur x
  2. **puis** simule l'exécution de M_α sur l'entrée y
- *Alors :*
  - M s'arrête sur $x \implies L(N_{M,x}) = L(M_\alpha) \in P$
  - M ne s'arrête pas sur $x \implies L(N_{M,x}) = \emptyset \notin P$

$$L(N_{M,x}) \in P \iff \langle M, x \rangle \in HALT$$

HALT se réduit à $RICE_P$ (2)

Preuve (suite et fin).

Supposons qu'il existe une MT M_R **qui décide** $RICE_P$.

HALT se réduit à $RICE_P$ (2)

Preuve (suite et fin).

Supposons qu'il existe une MT M_R **qui décide** $RICE_P$.

On construit une machine M_H qui, sur entrée $\langle M, x \rangle$:

1. calcule la description $\langle N_{M,x} \rangle$ de la MT $N_{M,x}$
2. exécute M_R sur $\langle N_{M,x} \rangle$
   - si M_R accepte $\langle N_{M,x} \rangle$, alors M_H accepte $\langle M, x \rangle$
   - sinon, M_H rejette $\langle M, x \rangle$

HALT se réduit à $RICE_P$ (2)

Preuve (suite et fin).

Supposons qu'il existe une MT M_R **qui décide** $RICE_P$.

On construit une machine M_H qui, sur entrée $\langle M, x \rangle$:

1. calcule la description $\langle N_{M,x} \rangle$ de la MT $N_{M,x}$
2. exécute M_R sur $\langle N_{M,x} \rangle$
   - si M_R accepte $\langle N_{M,x} \rangle$, alors M_H accepte $\langle M, x \rangle$
   - sinon, M_H rejette $\langle M, x \rangle$

Or : $L(N_{M,x}) \in P \iff \langle M, x \rangle \in HALT$

Donc, M_H décide $HALT$, contradiction.

Application

ACC_L for a R.E. language L

ENTRÉE : la description $\langle M \rangle$ d'une MT M
QUESTION : M accepte-t-elle le langage L ?

Le problème ACC_L est indécidable.

Preuve.