Théorie de la complexité — Travaux Dirigés n. 4

1 Définition du problème de correspondance de Post

Cette feuille est dédiée à un nouveau problème de décision. Ici, on commence par le définir. On montrera plus tard son indécidabilité.

Définition 1 (Problème de correspondance de Post (PCP))
Le problème est défini comme suit :

ENTRÉE : Un alphabet Σ, un entier $n \geqslant 1$ et n paires de mots $(u_1, v_1), \ldots, (u_n, v_n) \in \Sigma^* \times \Sigma^*$.
QUESTION : Existe-t-il une séquence non vide et finie d'indices $i_1, \ldots i_k \in \{1, \ldots, n\}$ telle que :

$$u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$$

Exercices

Exercice 1 On considère les instances de **PCP** suivantes :

1. $u_1 = abc \quad u_2 = ca \quad u_3 = b$
   $v_1 = a \quad v_2 = bcca \quad v_3 = ccb$

2. $u_1 = aab \quad u_2 = cdabdc \quad u_3 = d$
   $v_1 = aa \quad v_2 = a \quad v_3 = bdcd$

3. $u_1 = c \quad u_2 = cc \quad u_3 = ababa$
   $v_1 = ab \quad v_2 = cccc \quad v_3 = a$

4. $u_1 = abb \quad u_2 = b \quad u_3 = a$
   $v_1 = a \quad v_2 = abb \quad v_3 = bb$

Pour chaque instance, donner une solution (elles en ont toutes une). ▪

Exercice 2 On considère l'instance de **PCP** suivante :

$$u_1 = aab \quad u_2 = aa \quad u_3 = b \quad u_4 = bab$$
$$v_1 = aa \quad v_2 = ab \quad v_3 = a \quad v_4 = ba$$

Montrer qu'il n'y a aucune solution. ▪

Exercice 3 Montrer que **PCP** restreint à l'alphabet unaire $\{1\}$ est décidable. ▪

Exercice 4 Montrer que **PCP** est semi-décidable. ▪

Exercice 5 — Problème d'inclusion de Post. Soit u, v deux mots. On écrit $u \subseteq v$ quand u est un *sous-mot* de v, c'est-à-dire qu'on peut obtenir u à partir v en effaçant des lettres (par exemple $aaa \subseteq ababca$).

On considère le problème d'inclusion de Post qui est une version légèrement modifiée du problème de correspondance de Post :

ENTRÉE : Un alphabet Σ, un entier $n \geqslant 1$ et n paires de mots $(u_1, v_1), \ldots, (u_n, v_n) \in \Sigma^* \times \Sigma^*$.

QUESTION : Existe-t-il une séquence non vide et finie d'indices $i_1, \ldots i_k \in \{1, \ldots, n\}$ tels que :

$$u_{i_1} \cdots u_{i_k} \subseteq v_{i_1} \cdots v_{i_k}$$

Le but de l'exercice est de montrer que ce problème est décidable (contrairement à **PCP** qu'on va montrer indécidable plus bas).

1. Soient v, v', u et u' des mots. Montrez que si $u\,u' \subseteq v\,v'$, alors $u \subseteq v$ ou $u' \subseteq v'$.
2. Montrez que si le problème d'inclusion de Post a une solution, alors il existe une solution qui a pour longueur 1 (*i.e.*, la séquence d'indices est de longueur 1).
3. Montrez que le problème d'inclusion de Post est décidable. ∎

2 Indécidabilité de PCP

Le but de cette section est de montrer que le problème de correspondance de Post est indécidable. On commence par énoncer le résultat dans le théorème suivant.

Théorème 2

Le problème de correspondance de Post est indécidable.

Le Théorème 2 est prouvé par réduction du problème de l'arrêt pour les machines de Turing (**HALT**) vers le problème de correspondance de Post (**PCP**). En effet, puisqu'on sait déjà que **HALT** est indécidable, une telle réduction prouve que **PCP** est également indécidable.

Pour des raisons de commodité technique, on présente cette réduction en deux étapes. On commence par introduire une version modifiée de **PCP** qu'on note **MPCP**. On prouve ensuite séparément que **HALT** se réduit à **MPCP** d'une part, et que **MPCP** se réduit à **PCP** d'autre part. En composant ces deux réductions, on obtient la réduction voulue de **HALT** vers **PCP**.

Intérêt de MPCP

En elle-même, la définition de **MPCP** est artificielle. Notre unique motivation pour introduire ce problème est qu'il rend notre preuve plus simple à rédiger. Comme souvent en mathématiques, on sépare un énoncé à prouver en plusieurs énoncés plus simples.

Commençons par définir **MPCP**. L'unique différence avec le problème de correspondance de Post original est qu'on impose la première paire de mots qui doit être utilisée dans la séquence solution.

Définition 3 (Problème de correspondance de Post modifié (MPCP))

Le problème est défini comme suit :

ENTRÉE : Un alphabet Σ, un entier $n \geqslant 1$ et n paires de mots $(u_1, v_1), \ldots, (u_n, v_n) \in \Sigma^* \times \Sigma^*$.

QUESTION : Existe-t-il une séquence (possiblement vide) finie d'indices $i_1, \ldots i_k \in \{1, \ldots, n\}$ telle que :

$$u_1\,u_{i_1} \cdots u_{i_k} = v_1\,v_{i_1} \cdots v_{i_k}$$

On remarquera que **PCP** et **MPCP** prennent les mêmes objets en entrée. Cependant, une entrée qui admet une solution pour **PCP** n'en admet pas forcément une pour **MPCP** : la question posée par **MPCP** est plus contraignante puisque on doit nécessairement utiliser la paire (u_1, v_1) comme premier élément de notre séquence solution.

Les deux exercices qui suivent demandent de donner les deux réductions annoncées plus haut. Comme nous l'avons expliqué, leur existence implique le Théorème 2 : **PCP** est indécidable.

Exercice 6 Donner une réduction de **MPCP** vers **PCP**.

Indication. On pourra utiliser l'observation suivante. Considérons un alphabet Σ qui contient un symbole spécial $\# \in \Sigma$ et une séquence $(u_1, v_1), \ldots, (u_n, v_n) \in \Sigma^* \times \Sigma^*$. Supposons que les deux propriétés suivantes sont vraies :

- *Le mot u_1 commence par la lettre "#", et les autres mots u_i pour $i \geqslant 2$ commencent par une autre lettre.*
- *Tous les mots v_i pour $i \geqslant 1$ commencent par la lettre "#".*

Alors, s'il existe une séquence non vide et finie d'indices $i_1, \ldots i_k \in \{1, \ldots, n\}$ tels que : $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$, on a forcément $i_1 = 1$. ▪

Exercice 7 Donner une réduction de **HALT** vers **MPCP**.

Indication. Étant une entrée pour **HALT** (i.e., *une machine de Turing M et un mot d'entrée w pour M), la réduction doit construire une entrée pour* **MPCP**, *c'est-à-dire une liste de paires de mots $(u_1, v_1), \ldots, (u_n, v_n)$. Par définition, cette construction doit satisfaire l'équivalence suivante : M s'arrête sur w si et seulement si il existe une séquence (possiblement vide) finie d'indices $i_1, \ldots i_k \in \{1, \ldots, n\}$ telle que $u_1 u_{i_1} \cdots u_{i_k} = v_1 v_{i_1} \cdots v_{i_k}$.*

L'intuition est que quand cette séquence existe, le mot $u_1 u_{i_1} \cdots u_{i_k} = v_1 v_{i_1} \cdots v_{i_k}$ encode l'execution (finie) de M sur w. ▪

Exercices : utilisation de PCP

Exercice 8 Montrer que PCP restreint à l'alphabet binaire $\{0, 1\}$ reste indécidable. ▪

Dans les deux exercices suivants, on va utiliser PCP pour prouver que certains problèmes des *grammaires hors-contexte* sont indécidables. On commence par définir ce nouvel objet.

On utilise les grammaires hors-contexte pour définir des langages. Formellement, une grammaire hors-contexte est un tuple $G = (\Sigma, V, S, R)$ tel que,

- Σ est un alphabet (la grammaire définit un langage $L(G) \subseteq \Sigma^*$).
- $V = \{X, Y, Z, \ldots, \}$ est un ensemble fini de variables.
- $S \in V$ est une variable de départ.
- R est un ensemble de règles de réécriture. Chaque règle s'écrit de la façon suivante :

$$X \to w \qquad \text{avec } X \in V \text{ et } w \in (V \cup \Sigma)^*$$

Par exemple $G = (\{a, b\}, \{X\}, X, R)$ avec $R = \{X \to ab, X \to aXb\}$ est une grammaire hors-contexte.

À toute grammaire hors-contexte $G = (\Sigma, V, S, R)$, on associe un langage $L(G) \subseteq \Sigma^*$ de la façon suivante. Soient u, v deux mots dans $(\Sigma \cup V)^*$ (leurs lettres sont soient des variables de V soit des lettres de Σ). On note :

$$u \to v$$

si il existe une règle de réécriture $X \to w$ dans R, et $u_1, u_2 \in (\Sigma \cup V)^*$ tels que :

- $u = u_1 X u_2$, et,
- $v = u_1 w u_2$.

En d'autre termes, on obtient v à partir de u en remplaçant la variable X par le mot w (*i.e.*, en appliquant la règle $X \to w$).

Enfin, on note $\xrightarrow{*}$ la clôture transitive de $\rightarrow$: $u \xrightarrow{*} v$ si et seulement si il existe $u_0, \ldots, u_n$ tels que,

$$u = u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \cdots \rightarrow u_n = v$$

On peut maintenant définir le langage des mots engendrés par la grammaire G :

$$L(G) = \{v \in \Sigma^* \mid S \xrightarrow{*} v\}$$

Exercice 9 On considère la grammaire $G = (\{a,b\}, \{X\}, X, R)$ où $R = \{X \rightarrow ab, X \rightarrow aXb\}$. Quel est le langage défini par G ? ▪

Exercice 10 — Intersection vide de langages hors-contexte. On considère le problème d'intersection vide pour les langages hors-contexte :

ENTRÉE	:	Deux grammaires hors-contexte G, H
QUESTION	:	$L(G) \cap L(H) = \emptyset$?

Montrer que ce problème est indécidable.

Suggestion : on peut réduire le complémentaire du problème de correspondance de Post au problème d'intersection vide pour les langages hors-contexte. C'est-à-dire qu'étant donné instance de PCP, on doit expliquer comment construire deux langages hors-contexte L_1, L_2 tels que $L_1 \cap L_2 = \emptyset$ si et seulement si l'instance du PCP n'a pas de solution. ▪

Exercice 11 — ♠ Universalité de langages hors-contexte. On considère le problème problème d'universalité pour les langages hors-contexte :

ENTRÉE	:	Une grammaire hors-contexte G sur un alphabet Σ
QUESTION	:	$L(G) = \Sigma^*$?

Montrer que le problème d'universalité pour les langages hors-contexte est indécidable.

Suggestion : on peut réduire le complémentaire du problème de correspondance de Post au problème d'universalité de langages hors-contexte. Pour cela, on peut considérer une instance de PCP qu'on notera $(u_1, v_1), \ldots, (u_n, v_n) \in \Sigma^* \times \Sigma^*$. Étant donné cette instance, on peut construire un nouvel alphabet $\Delta = \Sigma \uplus \{1, \ldots, n\} \uplus \{\#\}$ et un langage hors-contexte L qui contient tous les mots de Δ^*, sauf ceux de la forme

$$\overleftarrow{u}_{i_k}\ i_k\ \ldots \overleftarrow{u}_{i_2}\ i_2\ \overleftarrow{u}_{i_1}\ i_1\ \#\ i_1\ v_{i_1}\ i_2\ v_{i_2}\ \ldots\ i_k\ v_{i_k}$$

ou $k \geqslant 1, 1 \leqslant i_1, i_2, \ldots, i_k \leqslant n$, $\overleftarrow{u}_j$ est le miroir de u_j, et $u_{i_1}\ u_{i_2}\ \ldots\ u_{i_k} = v_{i_1}\ v_{i_2}\ \ldots\ v_{i_k}$. En d'autres termes ces mots sont ceux qui codent une solution pour l'instance de PCP. On aura donc que $L = \Delta^*$ si et seulement si l'instance de PCP n'admet pas de solution. ▪