# TD 9 – Calculabilité et Indécidabilité

**Exercice 1.** *PCP*

$\Sigma$ est un alphabet fini et $P$ un ensemble fini de paires de mots sur $\Sigma$. Le Problème de Correspondance de Post associé à $\Sigma, P$ est l'existence d'une suite non vide $(v_i, w_i)_i$ d'éléments de $P$ telle que la concaténation des $v_i$ soit égale à la concaténation des $w_i$. Le Problème de Correspondance de Post Modifié est celui de l'existence d'une telle suite lorsque le premier terme est fixé.

**1.** Résoudre PCP pour les instances suivantes :
   1. $P = (aab, ab), (bab, ba), (aab, abab)$
   2. $P = (a, ab), (ba, aba), (b, aba), (bba, b)$
   3. $P = (ab, bb), (aa, ba), (ab, abb), (bb, bab)$
   4. $P = (a, abb), (aab, b), (b, aa), (bb, bba)$

**2.** Montrer que si $\Sigma$ ne contient qu'une lettre le problème est décidable.

**3.** Montrer l'équivalence entre PCP et PCPM.

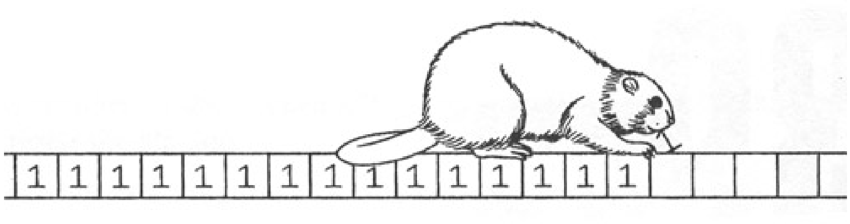
**4.** Peut-on se passer des couples de la forme $(w, w)$ dans PCPM ?

**5.** Montrer que PCP est indécidable.

**Exercice 2.** *Ceux-là font travailler l'imagination sans raconter d'histoire...*

On considère des machines de Turing à une bande bi-infinie et à $n$ états, plus un état d'arrêt noté $H$. L'alphabet est $\Sigma = \{0, 1\}$, où 0 fait également office de symbole blanc. La tête de lecture ne peut pas rester sur place.

On appelle *Castor Affairé*, ou *Busy Beaver* [Tibor RADÓ, 1962], la fonction qui à $n$ associe le nombre maximal de 1 qu'il est possible d'obtenir par une machine de Turing à $n$ états dont la bande ne contient initialement que des 0 et **qui s'arrête** (et telle que définie ci-dessus). On note $\mathbf{BB}(n)$ cette fonction. On définit de la même façon la fonction *Max Shift* telle que $\mathbf{MS}(n)$ est le nombre maximal possible de transitions pour $n$ états.




**1.** Combien existe-t-il de machines à 2 symboles et $n$ états (plus l'état d'arrêt $H$) ?

**2.** Combien existe-t-il de machines à 2 symboles et à 2 états ? Construire une (ou des) machines qui atteignent les scores $\mathbf{BB}(2)$ et $\mathbf{MS}(2)$. Proposer des machines candidates pour $\mathbf{BB}(3)$ et $\mathbf{MS}(3)$.

Actuellement, nous connaissons les valeurs exactes de $\mathbf{BB}(n)$ et $\mathbf{MS}(n)$ uniquement pour quelques petites valeurs de $n$. On peut constater (voir Tab. 1) que ces fonctions croissent très vite : en fait elles croissent plus vite que n'importe quelle fonction calculable.

| $n$ | $\mathbf{BB}(n)$ | $\mathbf{MS}(n)$ |
|---|---|---|
| 2 | 4 | 6 |
| 3 | 6 | 21 |
| 4 | 13 | 107 |
| 5 | $\geq$ 4098 | $\geq$ 47 176 870 |
| 6 | $\geq 3,5 \times 10^{18276}$ | $\geq 7,4 \times 10^{36534}$ |

TABLE 1 – Les premières valeurs de **BB** et **MS**.

3. Montrer que la fonction **BB** n'est pas calculable par une machine de Turing.
   *Indice : on pourra considérer la composition de machines de Turing, dont une qui double le nombre de 1.*
4. Montrer de deux façons différentes que la fonction **MS** n'est pas calculable.

**Exercice 3.** *0-τ*

THÉORÈME DE RICE [HENRY RICE, 1953] : Toute propriété non triviale (c'est-à-dire qui n'est ni toujours vraie ni toujours fausse) sur les langages récursivement énumérables est indécidable.

1. Donner des exemples de propriétés non triviales sur les langages récursivement énumérables.
2. En quoi le problème de l'ARRÊT est-il un cas particulier du théorème de Rice ?
3. Démontrer le théorème de Rice.
4. Existe-t-il une machine de Turing qui décide si la machine donnée en entrée ne s'arrête sur aucune entrée ?
5. Existe-t-il une machine de Turing qui décide si le domaine de la machine donnée en entrée est infinie ?
6. Existe-t-il une machine de Turing qui décide si les deux machines passées en argument calculent la même fonction ?

**Exercice 4.** *Pouvez vous travailler chez St Macloud ?*

Un *jeu de tuiles de Wang* est un ensemble fini de tuiles carrées, chacun des côté des carrés étant coloriés (ou étiquetés). Deux tuiles peuvent s'assembler de manière valide selon deux côtés si ces deux côtés ont la même couleur (il est interdit de faire tourner les tuiles). Un *pavage* est un recouvrement du plan tel que chaque assemblage de tuile deux-à-deux est valide. Montrer que le problème suivant est indécidable :

Problème TILING

> *entrée* : $T$ un jeu fini de tuiles de Wang et $t_0$ une tuile.
>
> *sortie* : peut-on paver le plan avec $T$ en partant de $t_0$ ?

**Exercice 5.** *Plus fort que la machine*

Les fonctions suivantes sont-elles calculables ?

1. la fonction partielle $R$ définie par $R(\mathfrak{M}, x) = n_{\rightarrow}$ où $n_{\rightarrow}$ est le nombre de mouvements vers la droite que fait la tête de la machine $\mathfrak{M}$ lors du calcul sur l'entrée $x$.
2. la fonction partielle $C$ définie par $C(\mathfrak{M}, x) = n_c$ où $n_c$ est le nombre de cases utilisées par la machine $\mathfrak{M}$ lors du calcul sur l'entrée $x$.
3. la fonction $Q$ définie par $Q(\mathfrak{M}, x) = n_q$ où $n_q$ est le nombre d'états utilisés par la machine $\mathfrak{M}$ lors du calcul sur l'entrée $x$.
4. l'entrée $n$ utilisera un alphabet composé des quantificateurs $\forall$ et $\exists$, des variables $\{x_1, \ldots, x_k\}$ à interpréter sur $\mathbb{N}$, les symboles $+$ et $=$ et les connecteurs booléens $\wedge, \vee, \neg$. Par exemple : $\forall x_1, x_2 \quad (\exists x_3 \quad x_1 + x_2 = x_3) \wedge (\exists x_4 \quad x_2 = x_4)$. $n$ est une entrée *valide* si $n$ représente une formule sans variable libre (toutes les variables qui apparaissent sont précédemment quantifiées).

   La fonction $f$ est la suivante : si $n$ est valide, $f(n)$ renvoie une preuve que la formule $n$ est vraie ou fausse, et sinon $f(n)$ est le symbole blanc.
5. l'entrée $n$ représente un polynôme à coefficients entiers une (resp. à plusieurs) variable(s) et $f(n)$ renvoie une racine entière du polynôme si cela existe, et $\varepsilon$ sinon.