Fondements de l'informatique. Examen
Durée: 3h

Sujet proposé par Olivier Bournez
Version 4
(corrigé)

Les 4 parties sont indépendantes, et peuvent être traitées dans un ordre quelconque. On pourra admettre le résultat d'une question pour passer aux questions suivantes. On pourra utiliser tous les résultats et les théorèmes démontrés ou énoncés en cours ou en petite classe ou dans le polycopié sans chercher à les redémontrer.

Il est possible d'avoir la note maximale sans répondre à toutes les questions. La difficulté des questions n'est pas une fonction linéaire ni croissante de leur numérotation.

La qualité de votre argumentation et de votre rédaction est une partie importante de votre évaluation.

Tous les graphes considérés sont non-orientés.

1 Machines de Turing

On fixe un alphabet fini Σ. Σ^* désigne l'ensemble des mots sur l'alphabet Σ.

Question 1. *Les problèmes suivants sont-ils décidables ? Justifier.*

- *Déterminer si le langage accepté par une machine de Turing M est co-fini : on dit qu'un langage est* co-fini *si son complémentaire (dans Σ^*) est fini.*
- *Déterminer si une machine de Turing accepte son entrée en utilisant au plus 2^{2^n} cases du ruban, où n est la longueur du mot en entrée.*
- *On fixe une fonction calculable $g : \mathbb{N} \to \mathbb{N}$. Déterminer si une machine de Turing accepte son entrée en utilisant au plus $2^{g(n)}$ cases du ruban, où n est la longueur du mot en entrée. Même question si l'on suppose que $g : \mathbb{N} \to \mathbb{N}$ est quelconque (c'est-à-dire, possiblement non-calculable).*

Solution : Le premier problème est indécidable par une application directe du théorème de Rice : le problème est non-trivial car il y a bien des langages finis et co-finis récursivement énumérables. Par exemple, le langage $L = \{0\}$ et son complémentaire.

Le second problème est un cas particulier du troisième pour $g(n) = 2^n$ qui est calculable.

Le troisième problème est décidable, quand g est calculable car il suffit de simuler la machine en la restreignant à au plus $2^{g(n)}$ cases du ruban pour décider la propriété (si jamais la simulation réalise que l'on repasse par la même configuration, alors on sait que la machine que l'on simule ne termine pas, et donc n'accepte pas son entrée).

Si g est possiblement non-calculable, le problème peut être indécidable. Considérer par exemple la fonction $g : \mathbb{N} \to \mathbb{N}$ avec $g(n)$ qui vaut 2 ou 1 suivant si la machine de Turing numéro n termine ou non sur l'entrée vide, et une machine de Turing M qui à l'étape 1 déplace sa tête de lecture d'une case vers la droite, puis à l'étape 2 déplace sa tête de lecture d'une case vers la droite et accepte. M accepte 1^n en au plus $g(n)$ étapes si et seulement si la machine de Turing numéro n termine. Décider le problème reviendrait à savoir décider si une machine de Turing donnée s'arrête. □

Une machine de Turing avec ruban semi-infini est une machine de Turing dont le programme est tel que la tête de lecture ne va jamais à gauche de sa position initiale.

Question 2. *Montrer que le problème de savoir si une machine de Turing semi-infinie accepte un mot w est indécidable.*

Solution : Le problème de l'arrêt des machines de Turing se réduit à ce problème : on transforme une machine de Turing M en une machine M' qui la simule de la façon suivante : à chaque fois que le ruban de M contient w, le ruban de M' contient $\#w$ mais avec $\#$ en face de la position initiale de la tête de lecture de M : M' commence par remplacer l'entrée w par $\#w$ en décalant w d'une case vers la droite, et en insérant un symbole $\#$ tout à gauche. A chaque fois que la tête de lecture de M souhaite aller à gauche et voudrait écrire un symbole a sur le symbole $\#$, M' décale le contenu w du ruban d'une case sur la droite et écrit a juste à droite de $\#$ pour que le ruban devienne $\#aw$. En faisant ainsi, M' simule M, et est une machine avec ruban semi-infini. M accepte w si et seulement si M' accepte w. M' s'obtient de façon calculable à partir de M. □

2 Espaces vectoriels

Soit E un $\mathbb{R}$-espace vectoriel. On dit que E est *ordonnable* s'il existe une relation d'ordre total $\leq$ sur E qui est compatible avec la structure d'espace vectoriel, c'est-à-dire :

— pour tout $x, x', y, y' \in E$, si $x \leq x'$ et $y \leq y'$ alors $x + y \leq x' + y'$;

— pour tout $x \in E$ et pour tout $\lambda \in \mathbb{R}$, si $x \geq 0$ et $\lambda \geq 0$ alors $\lambda x \geq 0$.

Question 3. *Construire un ensemble $\mathcal{F}_E$ de formules du calcul **propositionnel** sur une signature que vous préciserez tel que $\mathcal{F}_E$ est satisfiable si et seulement si E est ordonnable.*

Solution : A chaque couple $x, y \in E$, on associe une variable propositionnelle $X_{x,y}$. Soit $\mathcal{F}_E = \{R_x : x \in E\} \cup \{S_{x,y} : x \neq y \in E\} \cup \{T_{x,y,z} : x, y, z \in E\} \cup \{U_{x,y,x',y'} : x, y, x', y' \in E\} \cup \{V_{x,\lambda} : x \in E \text{ et } \lambda \in \mathbb{R}^+\}$ où

$$R_x : X_{x,x},$$

$$S_{x,y} : \neg(X_{x,y} \wedge X_{y,x})$$

$$T_{x,y,z} : X_{x,y} \wedge Y_{y,z} \Rightarrow X_{x,z}$$

$$U_{x,y,x',y'} : X_{x,x'} \wedge Y_{y,y'} \Rightarrow X_{x+y,x'+y'}$$

et

$$V_{x,\lambda} : X_{x,0} \Rightarrow X_{\lambda x,0}$$

On vérifie que E est ordonnable si et seulement si $\mathcal{F}_E$ est satisfiable. □

Question 4. *Montrer qu'un $\mathbb{R}$-espace vectoriel E est ordonnable si et seulement si tous ses sous-espaces vectoriels de dimension finie*[1] *le sont.*

Solution : Il est évident que si E est ordonnable alors tous ses sous-espaces de dimension finie le sont.

Réciproquement : Supposons que tous les sous-espaces de dimension finie de E sont ordonnables et montrons que $\mathcal{F}_E$ est satisfiable. Soit G un sous-ensemble fini de $\mathcal{F}_E$. Soit $B = \{x \in E : \exists y \in E \text{ tel que } X_{x,y} \text{ ou } X_{y,x} \text{ figure dans l'une des formules de } G\}$. B est un sous-ensemble fini de E. Soit E' le sous-espace vectoriel engendré par B. E' est ordonnable, donc $\mathcal{F}_{E'}$ est satisfiable. Comme $G \subset \mathcal{F}_{E'}$, G est aussi satisfiable. Par le théorème de compacité, $\mathcal{F}_E$ est satisfiable. □

1. On rappelle qu'un sous-espace vectoriel de dimension finie de E est le plus petit sous-espace vectoriel engendré par (i.e. qui contient) un nombre fini d'éléments de E.

3 Chemins avec paires interdites

Etant donnés un graphe $G = (V, E)$, et un ensemble S de paires de sommets (u_1, v_1), (u_2, v_2), ..., $(u_k, v_k) \in V \times V$, on dit qu'un chemin de G évite les paires de S s'il visite au plus un sommet de chaque paire de S.

Question 5. *On considère le graphe G dont les sommets sont $V = \{s, t, u, v\}$ et les arêtes sont $E = \{(s, u), (s, v), (u, t), (v, t)\}$. Quels sont les chemins entre s et t qui évitent $S = \{(u, v)\}$?*

Solution : Un tel chemin visite les sommets s, u, t ou s, v, t. □

Question 6. *Construire un graphe à 5 sommets $V = \{s, t, u_1, u_2, u_3\}$ et un ensemble de paires S tels que :*

- *les chemins entre s et t qui évitent S passent nécessairement par l'un des trois sommets u_1, u_2, ou u_3 ;*
- *pour chaque sommet u_i pour $i \in \{1, 2, 3\}$, il y a au moins un chemin entre s et t qui évite S et passe par u_i.*

Solution : On considère $E = \{(s, u_1), (u_1, t), (s, u_2), (u_2, t), (s, u_3), (u_3, t)\}$ et $S = \emptyset$. □

Question 7. *Démontrer que le problème suivant est NP-complet :*

- *Données : un graphe $G = (V, E)$, deux sommets $s, t \in V$, et un ensemble S de paires $(u_1, v_1), (u_2, v_2), \ldots, (u_k, v_k) \in V \times V$.*
- *Question : Existe-t-il un chemin entre s et t dans le graphe G qui évite les paires de S.*

On pourra utiliser la NP-complétude du problème 3-SAT.

Solution : Le problème est dans NP, car la donnée d'un chemin constitue un certificat vérifiable en temps polynomial.

Pour montrer la complétude, on effectue une réduction à partir du problème 3-SAT que l'on sait NP-complet.

La question 5 fournit un graphe $G_1(s, u, v, t)$ qui permet de coder le fait que l'on a soit x_i soit $\neg x_i$ pour une variable propositionnelle.

La question 6 fournit un graphe $G_2(s, u_1, u_2, u_3, t)$ qui permet de coder une clause.

En concaténant chacun des graphes, on obtient la réduction.

Plus précisément, étant donnée une conjonction de clauses $C = C_1 \wedge C_2 \cdots \wedge C_\ell$ sur les variables propositionnelles $x_1, x_2, \ldots, x_n$, avec $C_i = l_{i,1} \vee l_{i,2} \vee l_{i,3}$, où chaque $l_{i,j}$ est une variable propositionnelle ou sa négation, on construit le graphe qui contient les graphes $G_1(s, x_1, \neg x_1, t_1)$, $G_1(t_1, x_2, \neg x_2, t_2), \ldots, G_1(t_{n-1}, x_n, \neg x_n, t_n), G_2(t_n, l_{1,1}, l_{1,2}, l_{1,3}, t_{n+1}), G_2(t_{n+1}, l_{2,1}, l_{2,2}, l_{2,3}, t_{n+2})$, $\ldots G_2(t_{n+\ell-1}, l_{\ell,1}, l_{\ell,2}, l_{\ell,3}, t)$.

On prend l'ensemble S comme l'union de toutes les paires où figure une variable et sa négation : pour le dire plus formellement, S est constitué de toutes les paires $(x_i, \neg x_i)$, toutes les paires $(x_i, l_{i',j})$ où $l_{i',j} = \neg x_i$, et de toutes les paires $(\neg x_i, l_{i',j})$ où $l_{i',j} = x_i$.

Un chemin entre s et t qui évite S par construction satisfait C. Réciproquement, en chemin entre s et t qui évite S doit passer par chacun des x_i ou son complémentaire, et va indiquer comment satisfaire C selon les clauses traversées. □

4 Jouons aux dominos

Le problème de la correspondance de Post (PCP) est un problème de décision sur des *dominos* qui fut introduit par Emil Post en 1946. Il apparaît souvent dans des démonstrations d'indécidabilité : nous verrons un exemple plus bas concernant un problème sur les matrices.

Nous allons considérer plusieurs variantes de ce problème, et étudier leur difficulté.

On fixe un alphabet Σ.

On appelle "*domino*" un couple (U, V) où U et V sont des mots sur un alphabet Σ : on va représenter un tel couple graphiquement sous la forme $\boxed{\begin{array}{|c|}\hline U \\ \hline V \\ \hline \end{array}}$.

Etant donné une suite finie S de dominos, le problème est de savoir si l'on peut poser ces dominos l'un à la suite de l'autre (dans n'importe quel ordre) de telle sorte que le mot qui apparaît en haut soit le même que le mot qui apparaît en bas : on appelle cela une *correspondance*.

On souhaite savoir s'il existe un algorithme qui, étant donnée une suite finie S de dominos, décide si il existe une correspondance pour cette suite : on appelle cela le problème de correspondance de Post.

4.1 Sans répétition

Supposons que l'on a le droit d'utiliser chaque domino de S au plus une fois [2].

Par exemple, si l'on part de l'ensemble

$$S = \left\{ \boxed{\begin{array}{|c|}\hline ba \\ \hline ac \\ \hline \end{array}}, \boxed{\begin{array}{|c|}\hline a \\ \hline ab \\ \hline \end{array}}, \boxed{\begin{array}{|c|}\hline ca \\ \hline a \\ \hline \end{array}} \right\}$$

on peut construire la correspondance suivante :

$$\boxed{\begin{array}{|c|}\hline a \\ \hline ab \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline ba \\ \hline ac \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline ca \\ \hline a \\ \hline \end{array}} :$$

on a bien en haut et en bas le même mot, à savoir *abaca*.

Si cela aide de le dire très formellement : étant donné une suite de dominos [3]

$$S = \left\{ \boxed{\begin{array}{|c|}\hline U_1 \\ \hline V_1 \\ \hline \end{array}}, \boxed{\begin{array}{|c|}\hline U_2 \\ \hline V_2 \\ \hline \end{array}}, \ldots, \boxed{\begin{array}{|c|}\hline U_p \\ \hline V_p \\ \hline \end{array}} \right\}, \tag{1}$$

le problème de la correspondance de Post **sans répétition** consiste à déterminer s'il existe une suite non vide d'indices $i_1, i_2, \ldots, i_k$ dans $\{1, 2, \ldots, p\}$ telle que

$$U_{i_1} U_{i_2} \ldots U_{i_k} = V_{i_1} V_{i_2} \ldots V_{i_k}.$$

$$\left(\text{ graphiquement : } \boxed{\begin{array}{|c|}\hline U_{i_1} \\ \hline V_{i_1} \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline U_{i_2} \\ \hline V_{i_2} \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline U_{i_3} \\ \hline V_{i_3} \\ \hline \end{array}} \ldots \boxed{\begin{array}{|c|}\hline U_{i_k} \\ \hline V_{i_k} \\ \hline \end{array}} \right)$$

avec $i_k \neq i_l$ pour $k \neq l$.

Question 8. *Montrer que le problème de la correspondance de Post sans répétition est décidable.*

Solution : Etant donné S, il y a un nombre finie de possibilités de suites injectives dans $\{1, 2, \ldots, p\}$. Il suffit de tester pour chacune si cela satisfait

$$\boxed{\begin{array}{|c|}\hline U_{i_1} \\ \hline V_{i_1} \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline U_{i_2} \\ \hline V_{i_2} \\ \hline \end{array}} \boxed{\begin{array}{|c|}\hline U_{i_3} \\ \hline V_{i_3} \\ \hline \end{array}} \ldots \boxed{\begin{array}{|c|}\hline U_{i_k} \\ \hline V_{i_k} \\ \hline \end{array}}.$$

□

On établira dans la section 4.6 qu'il est NP-complet.

2. On n'interdit pas cependant qu'un même domino puisse avoir plusieurs occurrences dans S.

3. Pour les plus puristes, et en lien avec la note de bas de page 2. (page précédente) : la notation de la suite S dans l'écriture sous forme ensembliste (1) est possiblement la notation d'un multi-ensemble plutôt qu'un ensemble : par rapport à un ensemble, on autorise un même élement à apparaitre deux fois ou plus dans un multi-ensemble.

4.2 Avec répétitions

Jusqu'à la section 4.6, on s'autorise à utiliser plusieurs fois un même domino de la suite S. Par exemple, si l'on part de la suite

$$S_1 = \left\{ \boxed{\begin{array}{c} b \\ ca \end{array}}, \boxed{\begin{array}{c} a \\ ab \end{array}}, \boxed{\begin{array}{c} ca \\ a \end{array}}, \boxed{\begin{array}{c} abc \\ c \end{array}} \right\}$$

on peut construire la correspondance suivante :

$$\boxed{\begin{array}{c} a \\ ab \end{array}} \boxed{\begin{array}{c} b \\ ca \end{array}} \boxed{\begin{array}{c} ca \\ a \end{array}} \boxed{\begin{array}{c} a \\ ab \end{array}} \boxed{\begin{array}{c} abc \\ c \end{array}} :$$

on a bien en haut et en bas le même mot, à savoir *abcaaabc* (et on a utilisé plusieurs fois un même domino).

Tous les ensembles S ne possèdent pas de correspondance :

Question 9. *On part cette fois de la suite*

$$S_2 = \left\{ \boxed{\begin{array}{c} abc \\ ab \end{array}}, \boxed{\begin{array}{c} ca \\ a \end{array}}, \boxed{\begin{array}{c} acc \\ ba \end{array}} \right\}$$

Est-il possible d'obtenir une correspondance ?

Solution : C'est impossible, car le mot en haut possède nécessairement plus de lettres strictement que le mot en bas. □

Si cela aide de le dire formellement : étant donné un ensemble de dominos

$$S = \left\{ \boxed{\begin{array}{c} U_1 \\ V_1 \end{array}}, \boxed{\begin{array}{c} U_2 \\ V_2 \end{array}}, \ldots, \boxed{\begin{array}{c} U_p \\ V_p \end{array}} \right\},$$

le problème de la correspondance de Post (avec répétitions) consiste à déterminer s'il existe une suite non vide d'indices $i_1, i_2, \ldots, i_k$ dans $\{1, 2, \ldots, p\}$ telle que

$$U_{i_1} U_{i_2} \ldots U_{i_k} = V_{i_1} V_{i_2} \ldots V_{i_k}$$

$$\left(\text{ graphiquement : } \boxed{\begin{array}{c} U_{i_1} \\ V_{i_1} \end{array}} \boxed{\begin{array}{c} U_{i_2} \\ V_{i_2} \end{array}} \boxed{\begin{array}{c} U_{i_3} \\ V_{i_3} \end{array}} \cdots \boxed{\begin{array}{c} U_{i_k} \\ V_{i_k} \end{array}} \right).$$

4.3 Réduction à la variante modifiée

On s'intéresse à la variante suivante, que l'on appelle problème de correspondance de Post **modifié** : on impose que $i_1 = 1$, c'est-à-dire, on impose le premier domino.

Si cela aide de le dire très formellement : étant donné un ensemble de dominos

$$S = \left\{ \boxed{\begin{array}{c} U_1 \\ V_1 \end{array}}, \boxed{\begin{array}{c} U_2 \\ V_2 \end{array}}, \ldots, \boxed{\begin{array}{c} U_p \\ V_p \end{array}} \right\},$$

le problème de la correspondance de Post **modifié** consiste à déterminer s'il existe une suite non vide d'indices $i_2, i_3, \ldots, i_k$ dans $\{1, 2, \ldots, p\}$ telle que

$$U_1 U_{i_2} \ldots U_{i_k} = V_1 V_{i_2} \ldots V_{i_k}.$$

$$\left(\text{ graphiquement : } \boxed{\begin{array}{c} U_1 \\ V_1 \end{array}} \boxed{\begin{array}{c} U_{i_2} \\ V_{i_2} \end{array}} \boxed{\begin{array}{c} U_{i_3} \\ V_{i_3} \end{array}} \cdots \boxed{\begin{array}{c} U_{i_k} \\ V_{i_k} \end{array}} \right).$$

Soit $u = u_1 u_2 \dots u_n$ un mot de longueur n sur l'alphabet Σ. Soient $*$ et $\diamond$ deux nouvelles lettres : $* \notin \Sigma$, $\diamond \notin \Sigma$.

On définit $*u$, $u*$ et $*u*$ comme les mots suivants :

$$*u = *u_1 * u_2 * u_3 \cdots * u_n$$

$$u* = u_1 * u_2 * u_3 \cdots * u_n*$$

$$*u* = *u_1 * u_2 * u_3 \cdots * u_n*$$

En d'autres termes, $*u$ (respectivement : $u*$, $*u*$) ajoute le symbole $*$ avant (resp. après, avant et après) chaque lettre du mot u.

Question 10. *On considère*

$$S_1^* = \left\{ \boxed{\begin{array}{c} *a \\ *a*b* \end{array}}, \boxed{\begin{array}{c} *b \\ c*a* \end{array}}, \boxed{\begin{array}{c} *a \\ a*b* \end{array}}, \boxed{\begin{array}{c} *c*a \\ a* \end{array}}, \boxed{\begin{array}{c} *a*b*c \\ c* \end{array}}, \boxed{\begin{array}{c} *\diamond \\ \diamond \end{array}} \right\}$$

(Observer comment S_1^* s'obtient à partir de S_1 en début de la section 4.2 en appliquant les opérations ci-dessus en haut et en bas sur les dominos).

Expliquer comment les correspondances[4] *de S_1^* s'obtiennent à partir de celles de S_1 et réciproquement.*

Solution : Une correspondance de S_1^* débute nécessairement par le premier domino en raison de la lettre $*$: tous les dominos ont ce symbole tout à gauche en haut, et c'est le seul avec ce symbole tout à gauche en bas. De façon symétrique, une correspondance se termine par le symbole $\diamond$. En effet tous les dominos ont en bas comme dernier symbole $*$, et le dernier domino d'une correspondance ne peut être que $\boxed{\begin{array}{c} *\diamond \\ \diamond \end{array}}$.

Ensuite, chaque domino impose que tous les symboles en position paire (sauf le dernier $\diamond$) correspondent au symbole $*$. Si on efface les symboles $*$ et $\diamond$ dans la correspondance, c'est nécessairement une correspondance pour S_1.

Réciproquement, toute correspondance pour S_1 s'étend en une correspondance pour S_1^* en intercalant un symbole $*$ entre chaque lettre et en ajoutant le symbole $\diamond$ à la fin. □

Question 11. *Démontrer que le problème de Post modifié se réduit au problème de Post.*

Solution : On transforme une instance du problème de Post modifié en une instance du problème de Post comme dans la question précédente.

Concrètement pour chaque ensemble de dominos

$$S = \left\{ \boxed{\begin{array}{c} t_1 \\ b_1 \end{array}}, \boxed{\begin{array}{c} t_1 \\ b_1 \end{array}}, \boxed{\begin{array}{c} t_2 \\ b_2 \end{array}}, \boxed{\begin{array}{c} t_3 \\ b_3 \end{array}}, \dots, \boxed{\begin{array}{c} *\diamond \\ \diamond \end{array}} \right\},$$

on peut considérer

$$S^* = \left\{ \boxed{\begin{array}{c} *t_1 \\ *b_1* \end{array}}, \boxed{\begin{array}{c} *t_1 \\ b_1* \end{array}}, \boxed{\begin{array}{c} *t_2 \\ b_2* \end{array}}, \boxed{\begin{array}{c} *t_3 \\ b_3* \end{array}}, \dots, \boxed{\begin{array}{c} *\diamond \\ \diamond \end{array}} \right\}.$$

Par le raisonnement précédent, les correspondances de S sont en bijection avec celles de S^* en intercalant un symbole $*$ entre chaque lettre et en ajoutant le symbole $\diamond$ à la fin.

Une correspondance de S^* commence nécessairement par son premier domino $\boxed{\begin{array}{c} *t_1 \\ *b_1 \end{array}}$.

Cela donne donc bien une réduction du problème de correspondance de Post modifié vers le problème de correspondance de Post (la transformation de S en S^* est bien calculable).

□

4. On parle bien des correspondances "normales", i.e. au sens des questions précédentes, pas nécessairement "modifées" comme dans le texte qui précède.

4.4 Indécidabilité de la variante modifiée

Question 12. *Démontrer que le problème de la correspondance de POST modifié est indécidable.*

Etant donnés une machine de Turing M semi-infinie et un mot w on cherchera à construire un ensemble S_M de dominos tel qu'une correspondance de S_M décrit un calcul de M sur l'entrée w. On pourra inclure dans S_M les dominos

$$\boxed{\begin{array}{c}\#\\ \#q_0w\#\end{array}}, \boxed{\begin{array}{c}a\\ a\end{array}}, \boxed{\begin{array}{c}\#\\ B\#\end{array}}, \boxed{\begin{array}{c}aq_a\\ q_a\end{array}}, \boxed{\begin{array}{c}q_aa\\ q_a\end{array}}, \boxed{\begin{array}{c}q_a\#\#\\ \#\end{array}}$$

et d'autres dominos bien choisis que l'on décrira : w est un mot, #,a une lettre, B le symbole de blanc, q_0 et q_a l'état initial et acceptant de M.

Solution : On utilise les notations du cours pour décrire la machine de Turing M. q_a désigne l'état d'acceptation. On peut supposer sans perte de généralité que la machine de Turing déplace toujours sa tête de lecture à chaque étape.

Pour tout $a, a' \in \Gamma$, et pour tout état $q, q' \in Q$ avec $q \neq q_r$, si $\delta(q, a) = (q', a', \rightarrow)$, on ajoute $\boxed{\begin{array}{c}qa\\ a'q'\end{array}}$ à S_M.

Pour tout $a, a' \in \Gamma$, et pour tout état $q, q' \in Q$ avec $q \neq q_r$, si $\delta(q, a) = (q', a', \leftarrow)$, on ajoute $\boxed{\begin{array}{c}cqa\\ q'ca'\end{array}}$ à S_M.

Le jeu de domino est tel que qu'une correspondance va décrire en bas (et donc aussi en haut) le diagramme espace temps du calcul de M sur w : une correspondance sera nécessairement de la forme $\#q_0w\#C_1\#C_2 \ldots \#C_n\# \ldots \#q_a\#\#$ où $C_1, C_2, \ldots, C_n$ code la suite des configurations du calcul de la machine M sur le mot w jusqu'à atteindre l'état accepteur q_a à l'étape n. Les dominos $\boxed{\begin{array}{c}aq_a\\ q_a\end{array}}, \boxed{\begin{array}{c}q_aa\\ q_a\end{array}}$ permettent alors de "manger" les symboles en haut pour compléter et terminer avec le domino $\boxed{\begin{array}{c}q_a\#\#\\ \#\end{array}}$.

<il faut l'expliquer et le justifier le plus clairement possible pour obtenir la note maximale pour cette question, et cela nécessite des explications dont nous avons ici simplement donné l'idée>. □

Par conséquent, le problème de la correspondance de Post est indécidable.

4.5 Application : Problème de matrices

Un ensemble H de matrices 3×3 à coefficients entiers est dit *mortel* si la matrice nulle peut s'obtenir comme un produit d'un nombre fini de matrices de cet ensemble (on s'autorise à utiliser plusieurs fois la même matrice de l'ensemble H dans le produit).

Le but de cette partie est de prouver qu'il est indécidable de déterminer si un ensemble H de matrices est mortel.

On considere un ensemble H de matrices 3×3 qui contient les matrices

$$S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad T = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

ainsi que des matrices de la forme

$$W_j = \begin{bmatrix} p_j & 0 & 0 \\ 0 & r_j & 0 \\ q_j & s_j & 1 \end{bmatrix} \tag{2}$$

pour certains entiers p_j, q_j, r_j, s_j.

On note $[x, y, z]$ pour un vecteur ligne.

En observant que

$$[a, b, c]S = a[1, 0, 1]$$

$$[a, b, c]T = (a - b)[1, -1, 0]$$

et que toutes les matrices W_j sont inversibles, un raisonnement assez simple (sur les types de produits possibles avec un tel ensemble H et sur les images et noyaux de S et T) démontre le fait suivant que l'on admettra : *une condition nécessaire et suffisante pour obtenir un produit nul avec les matrices de l'ensemble H est qu'il existe un produit X des matrices W_j tel que*

$$[1, 0, 1]X = [h, h, 1]$$

pour un $h > 0$ (et dans ce cas $SXT = 0$).

Illustrons une relation entre la multiplication de matrices et la concaténation de mots par un exemple :

$$[123, 12, 1] \begin{bmatrix} 1000 & 0 & 0 \\ 0 & 100 & 0 \\ 223 & 32 & 1 \end{bmatrix} = [123223, 1232, 1]$$

Question 13. *Etant donnée une paire de mots $< U, V >$, définir une matrice $W(U, V)$ de la forme W_j (i.e. de la forme* (2)*) telle si X, Y, U, V sont des mots sur l'alphabet $\{1, 2, 3\}$, alors*

$$[X, Y, 1]W(U, V) = [X.U, Y.V, 1]$$

où . désigne la concaténation, et les mots sont interprétés comme des entiers de la façon évidente.

Solution : Considérer

$$W(U, V) = \begin{bmatrix} p & 0 & 0 \\ 0 & r & 0 \\ q & s & 1 \end{bmatrix}$$

où q, s sont les nombres obtenus en écrivant les mots U, V et p et r sont les entiers obtenus en considérant 1 suivi du nombre de zéros qui correspondent au nombre de lettres de U et V respectivement. □

Question 14. *Démontrer qu'il est indécidable de savoir si un ensemble H de matrices est mortel.*

Solution : On utilise une réduction à partir du problème de la correspondance de Post que l'on a démontré dans les questions précédentes indécidable. On peut supposer sans perte de généralité que l'alphabet vérifie $\Sigma = \{2, 3\}$.

Soit $K = \left\{\boxed{\frac{U_1}{V_1}}, \boxed{\frac{U_2}{V_2}}, \ldots, \boxed{\frac{U_n}{V_n}}\right\}$ un ensemble de dominos avec l'alphabet $\Sigma = \{2, 3\}$. On considère l'ensemble $H(K)$ de matrices

$$\{S, T, W(U_j, V_j), W(U_j, 1.V_j), \text{ pour } j = 1 \ldots n\}.$$

$H(K)$ est mortel si et seulement s'il y a un produit X de W de $H(K)$ tel que $[1, 0, 1]X = [h, h, 1]$ pour un certain h par la remarque plus haut (l'observation admise).

Un tel mot doit avoir comme première lettre 1, suivi d'un 2 et de 3 seulement, et donc un tel produit existe si et seulement si le problème de la correspondance de post admet une solution.

Le problème est dans NP car la donnée d'un produit constitue un certificat vérifiable en temps polynomial : observer que cela se vérifie bien en temps polynomial car la taille des nombres

impliqués dans le produit de matrice reste bien polynomial en la taille des nombres dans le produit.

Cela montre que le problème de la correspondance de post se réduit à ce problème : la réduction est bien calculable en temps polynomial. □

4.6 Retour sur les versions sans répétition

On revient sur la variante du problème de la correspondance de Post **sans répétition**, considérée dans la section 4.1 (rappel : on peut bien avoir plusieurs fois un même domino dans la suite S, mais chaque domino ne peut être utilisé qu'une fois dans une correspondance).

Question 15. *Montrer que le problème de correspondance de Post sans répétition est NP-complet.*

Solution : Le problème est clairement dans NP car la donnée du choix du produit est un certificat vérifiable en temps polynomial : il suffit de vérifier que le produit est nul.

On peut démontrer le fait que le problème est NP-difficile directement (en revenant à la définition) de la façon suivante : étant donné un problème NP-complet, il est décidé par une machine de Turing M non-déterministe qui fonctionne en temps $p(n)$. On construit le jeux de dominos correspondant considéré dans la réduction dans la réponse de la question 12. Le nombre de dominos construits est une fonction de la taille de l'alphabet et du nombre d'états de la machine de Turing M. Le seul domino dont la taille peut être grande est le domino $\begin{array}{|c|}\hline \# \\ \hline \#q_0w\# \\ \hline \end{array}$. Le jeu de dominos nécessaire pour simuler $p(n)$ étapes de M est $f(p(n))$ où $f(n)$ est une fonction polynomiale qui exprime le nombre de dominos pour simuler n étapes de M.

On aura bien w accepté par M si et seulement si l'on a une instance positive au problème de correspondance de Post sans répétition.

(en fait, le jeu de dominos construit répète bien plusieurs fois certains dominos, mais on utilisera au plus un exemplaire de chacun de ces dominos). □

Question 16. *On ne s'autorise pas les répétitions de matrices. Montrer qu'il est décidable de savoir si un ensemble H de matrices est mortel sans répétition. Montrer que le problème est NP-complet.*

Solution : Le problème est décidable, car il n'y a qu'un nombre fini de produits possibles de matrices sans répétition, et il suffit de tester pour chacun s'il est nul.

La réduction produite dans la réponse à la question 14 produit en fait aussi une réduction du problème de la correspondance de Post sans répétition à ce problème.

(la réduction est bien calculable en temps polynomial).

Par la question précédente, le problème est donc NP-complet. □