Bordeaux INP
ENSEIRB
MATMECA

Calculabilité & Complexité
Devoir Maison n° 1

21 février 2021

Rémi DEHENNE
Sébastien DELPEUCH
Tom MOËNNE-LOCCOZ
Aurélien MOINEL

Exercice 1 : Clôture

Soient R l'ensemble des langages récursifs et RE l'ensemble des langages récursivement énumérables.

Q1.a R est-il clos pour l'opération $\cap$?

Pour tous langages $L1$ et $L2$ de R, montrons que $L = (L1 \cap L2) \in R$.

Puisque $L1$ et $L2$ appartiennent à R, il existe deux machines de Turing $MT1$ et $MT2$ qui décident respectivement de ces langages. Nous proposons la machine de Turing MT formée de la « concaténation » des « sous-machines » $MT1$ et $MT2$.

Pour distinguer ces deux sous-machines au sein de MT, nous suffixons leurs états (lecture et écriture) par leur numéro et nous employons le nombre maximal de rubans nécessaires entre les deux machines.

Sur une entrée quelconque, la machine MT emploie d'abord les transitions de la machine $MT1$ grâce à une transition de l'état q_0 à l'état q_{0_1}. Si l'état q_{no_1} est atteint, la machine MT passe dans l'état q_{no} et termine. Si l'état q_{yes_1} est atteint, les bandes de travail sont remises à zéro et toutes les têtes de lecture sont remises sur leur *start symbol*, avant que les transitions de la machine $MT2$ soient employées. Lorsque les états q_{no_2} ou q_{yes_2} sont atteints, la machine MT passe dans l'état q_{no} ou q_{yes} et termine.

Ainsi, pour tout mot w appartenant à $L = L1 \cap L2$, la machine MT simule successivement les machines $MT1$ et $MT2$ qui acceptent toutes deux successivement le mot en temps fini, donnant ainsi lieu à un état terminal q_{yes} en temps fini. En effet, le mot appartient simultanément aux deux langages, donc ces machines terminent toutes deux sur un état acceptant. Au contraire, si w n'appartient pas simultanément à $L1$ et $L2$, la machine MT termine en temps fini dans un état terminal q_{no}, en étant passée par une ou deux machines terminant toutes deux en temps fini.

La machine MT décide donc du langage $L = L1 \cap L2$. Le langage L est donc lui aussi récursif et ce en toute généralité concernant $L1$ et $L2$ de R. L'ensemble R est donc bien clos pour l'opération $\cap$.

Q1.b R est-il clos pour l'opération $\setminus$?

Pour tous langages $L1$ et $L2$ de R, montrons que $L = (L1 \setminus L2) \in R$.

Pour une machine de Turing $MT2$ qui décide d'un langage $L2$ quelconque, il est possible de créer la machine $\overline{MT2}$ qui décide du langage $\overline{L2}$ (l'ensemble des mots n'appartenant pas à $L2$). Pour cela, on conserve dans $\overline{MT2}$ toutes les transitions de $MT2$, mais les changements d'état vers q_{no} sont redirigés vers q_{yes} et inversement. Le langage $\overline{L2}$ est donc lui aussi récursif.

Or, pour tous langages $L1$ et $L2$ de R, $L = L1 \setminus L2 = L1 \cap \overline{L2}$. Les langages $L1$ et $\overline{L2}$ sont donc tous deux des langages de R et R est clos par $\cap$, donc $L = L1 \setminus L2$ appartient à R pour tous $L1$ et $L2$ de R.

L'ensemble R est donc bien clos pour l'opération $\setminus$.

Q2.a RE est-il clos pour l'opération $\cap$?

Pour tous langages $L1$ et $L2$ de RE, montrons que $L = (L1 \cap L2) \in RE$.

Puisque $L1$ et $L2$ appartiennent à RE, il existe deux machines de Turing $MT1$ et $MT2$ qui acceptent respectivement ces deux langages. Nous proposons la machine de Turing MT formée de la « concaténation » des machines $MT1$ et $MT2$ vue précédemment.

Ainsi, pour tout mot w appartenant à $L = L1 \cap L2$, la machine MT simule successivement les machines $MT1$ et $MT2$, qui acceptent toutes deux le mot en temps fini dans l'état q_{yes}. En effet, le mot appartient simultanément aux deux langages, donc ces machines terminent toutes deux sur un état acceptant. Au contraire, si w n'appartient pas simultanément à $L1$ et $L2$:

— soit la machine MT termine en temps fini dans un état terminal q_{no}, en étant « passée par » une ou deux machines terminant toutes deux en temps fini ;

— soit l'une des deux machines $MT1$ ou $MT2$ ne termine pas, et MT ne termine pas.

Dans ces deux cas où le mot est rejeté, la machine ne termine pas sur l'état q_{yes}.

La machine MT accepte donc le langage $L = L1 \cap L2$. Le langage L est donc lui aussi récursivement énumérable, et ce en toute généralité concernant $L1$ et $L2$ de R. L'ensemble RE est donc bien clos pour l'opération $\cap$.

Q2.b RE est-il clos pour l'opération $\setminus$?

Montrons qu'il existe $L1$ et $L2$ de RE tels $L = (L1 \setminus L2) \in RE$.

Supposons par l'absurde qu'il existe un langage $L2$ tel que $L2 \in RE$ et $L2 \notin R$, et que RE est clos pour l'opération $\setminus$.

Dans ce cas, pour tout langage $L1 \in RE$, $L = (L1 \setminus L2) \in RE$. En particulier, le langage Σ^* appartient à RE puisqu'il est accepté par l'automate trivial qui possède une unique transition de l'état q_0 à q_{yes}, quelle que soit la lecture. Cet automate accepte donc tous les mots, donc le langage Σ^* : par conséquent, $(\Sigma^* \setminus L2) \in RE$.

Or, $(\Sigma^* \setminus L2) = \overline{L2}$, donc $\overline{L2} \in RE$. Cependant, si $L2$ et $\overline{L2}$ sont récursivement énumérables, ils doivent être récursifs (vu en cours). Or, nous avons supposé que $L2 \notin R$: c'est absurde. De ce fait, soit il n'existe pas de langage dans RE qui n'appartient pas à R, soit RE n'est pas clos pour l'opération $\setminus$, soit les deux.

Cependant, le langage *HALT* étudié en cours est un exemple de langage récursivement énumérable qui n'est pas récursif. RE n'est donc pas clos pour l'opération $\setminus$.

Annexe : Procédure d'effaçage en temps fini

Ajouter une bande « d'effaçage ». Pendant toutes les transitions de $MT1$ au sein de MT, avancer sur le bande « d'effaçage » (R), puis à la lecture de q_{yes_1}, reculer (L) et pour chaque autre bande de travail avancer (R), répéter.

Lorsque la tête de lecture supplémentaire arrive sur le *start symbol*, faire effacer et reculer (L) sur tous les rubans de travail sauf ceux déjà sur leur *start symbol* qui ne font rien, jusqu'à ce que toutes les têtes de lecture soient sur le *start symbol*.

À ce moment-là, passer dans l'état q_{0_2} qui commande le départ de la machine $MT2$ au sein de la machine MT.

Cela prend moins de 3 fois le nombre de coups que la première machine a pris avant de terminer. Puisque cette dernière a terminé en temps fini, cet effaçage se fait lui aussi en temps fini. De plus, cela garantit que les bandes sont complètement réinitialisés puisque les caractères sont effacés au moins à partir de « nombre de coups » après le *start symbol*, ce qui est le plus loin que la machine $MT1$ ait pu écrire.

Exercice 2 : Énumération

Soit $E \subseteq \mathbb{N}$ un ensemble.

Q3 Démontrez que si E est fini alors il est récursif.

Pour tout singleton $\{e\}$, il existe une machine de Turing triviale qui décide du langage formé par ce singleton.

Pour tous les éléments, de E, nous construisons une telle machine et nous « concaténons » ces sous-machines pour former une machine MT. Cette machine simule à tour de rôle ces sous-machines, et termine sur q_{yes} dès qu'une sous-machine termine sur l'état q_{yes_i}, ou termine sur l'état q_{no} si la dernière sous-machine termine sur l'état $q_{no_{|E|}}$. Entre l'exécution de chaque sous-machine, les bandes de travail sont remises à zéro, et toutes les têtes de lectures sont repositionnées au *start symbol*.

Cette machine de Turing accepte exactement tous les éléments de E et aucun autre, et termine forcément en temps fini, en tant que « concaténation » de sous-machines terminant toutes en temps fini. MT décide donc du langage E en toute généralité sur les langages finis.

Les langages finis sont donc récursifs.

Q4 On suppose maintenant que E est infini. Démontrez que E est récursif si et seulement si E est énumérable dans l'ordre croissant. Une machine de Turing *énumère* un ensemble S si et seulement si elle écrit sur sa bande de sortie tous les éléments de S et seulement ceux-là (la machine ne s'arrête donc pas si S est infini).

Montrons d'abord que si E est récursif, il est énumérable dans l'ordre croissant.

Si E est récursif, il existe une machine de Turing $MT1$ qui décide de E. Nous pouvons alors fabriquer à partir de $MT1$ une machine de Turing $MT2$ qui énumère E dans l'ordre croissant. $MT2$ est constituée des rubans de $MT1$ sauf du ruban d'entrée, qui est remplacé par un ruban « compteur ». Un ruban de sortie est ajouté. À l'initialisation, $MT2$ écrit la valeur 0 sur le ruban « compteur » et passe dans l'état q_{00}. Les transitions de $MT1$ sont conservées dans $MT2$ à la distinction que la lecture sur le ruban « compteur » remplace la lecture sur le ruban d'entrée. De plus, l'état initial des transitions de $MT1$ dans $MT2$ est q_{00}.

Si ces transitions terminent par l'état q_{yes}, la valeur écrite sur le ruban « compteur » est réécrite sur le ruban de sortie, puis les rubans de travail sont réinitialisés sauf le ruban « compteur ». Si ces transitions terminent par l'état q_{no}, les rubans sont directement réinitialisés (sauf le ruban « compteur »).

Après la réinitialisation, la valeur sur le ruban compteur est incrémentée, et $MT2$ repasse dans l'état q_{00}.

Nous remarquons que tous les entiers auront la possibilité d'être écrit dans l'ordre croissant, ce qui garantit que notre énumération ait lieu dans l'ordre croissant.

De plus, nous savons que les entiers sur le ruban de sortie sont tous ceux et uniquement ceux appartenant à E. En effet, pour tout entier $e \in E$, puisqu'il y a terminaison de la machine $MT1$ en temps fini, et qu'il y a un nombre fini (e) d'éléments à décider avant e, le ruban « compteur » arrive à la valeur e en temps fini. De plus, lorsque le ruban « compteur » arrive à e, la machine $MT1$ accepte nécessairement e ce qui entraîne son écriture sur le ruban de sortie.

A l'inverse, pour tout entier $e \notin E$, lorsque le ruban « compteur » arrive à la valeur e, la machine $MT1$ le refuse nécessairement donc sa valeur n'est jamais écrite sur le ruban de sortie.

Cette machine de Turing $MT2$ énumère donc E dans l'ordre croissant, et ce en toute généralité sur $E \in \mathbb{N}$ infini.

Dans l'autre sens, si E est énumérable dans l'ordre croissant, il existe une machine de Turing $MT1$ qui énumère E dans l'ordre croissant. Nous pouvons donc créer une machine de Turing $MT2$ qui simule le comportement interne de $MT1$. À chaque écriture d'un nouveau nombre o de $MT1$ au sein de $MT2$, il y a comparaison entre l'entrée e de $MT2$ et o.

Si $e > o$, la machine $MT2$ au sein de $MT1$ reprend son énumération.

Si $e = o$, $MT2$ passe dans l'état q_{yes} et termine. Cela aura pris un temps fini (succession d'écritures ou non des mots de 0 jusqu'à e compris, chacune en temps fini).

Si $e < o$, la machine passe dans l'état q_{no}, puisqu'il est impossible que le nombre recherché appartienne au langage (ce langage étant énuméré dans l'ordre croissant).

Dans ce cas, notre machine termine aussi en temps fini, étant donné que $MT1$ énumère les entiers d'un ensemble infini sur $\mathbb{N}$, qui est lui-même un ensemble bien-fondé, donc qui ne contient pas de plus grand élément. Cela implique qu'il existe o' le plus petit mot supérieur à w appartenant à E. Pour terminer sur l'entrée E, notre machine nécessitera $o' + 1$ écritures ou non de mots. Tous ces passages ayant lieu en temps fini, et puisqu'il y a un nombre fini de passages, la machine termine en temps fini.

Il y a équivalence entre le fait que $e \in E$ et que $MT2$ termine sur q_{yes}. En effet, si $e \in E$, le nombre sera forcément écrit sur le ruban de sortie, amenant alors à l'état q_{yes} de $MT2$. À l'inverse, si $MT2$ termine dans l'état q_{yes}, c'est nécessairement que $MT1$ a écrit e sur le ruban de sortie, ce qui implique que $e \in E$ puisque $MT1$ n'écrit sur le ruban de sortie que les mots appartenant à E.

De même, il y a équivalence entre le fait que $e \notin E$ et que $MT2$ termine sur q_{no}. En effet, si $e \notin E$, nous avons vu que $MT1$ au sein de $MT2$ finira forcément par écrire un nombre o' plus grand que e ce qui engendrera une transition vers l'état q_{no} et la terminaison. À l'inverse, si $MT2$ termine dans l'état q_{no}, c'est nécessairement que $MT1$ a écrit un entier $o' > e$ sur le ruban de sortie, ce qui implique que $e \notin E$ puisque $MT1$ écrit sur le ruban de sortie tous les mots appartenant à E dans l'ordre croissant.

Notre machine décide donc du langage E.

Il est donc équivalent que, si $E \in \mathbb{N}$ et E infini, E soit énumérable dans l'ordre croissant et E soit récursif.