Examen du cours de L3: Calculabilité

15 novembre 2018, durée: 3 heures

Tous les documents sont autorisés, les appareils électroniques sont interdits. Les résultats vus en TD, s'ils sont utilisés, doivent être redémontrés. Pour chacun des problèmes suivants, dire s'ils sont décidables ou non. Justifier.

1. **Donnée:** Le code d'une machine de Turing M

   Question: M accepte au moins tous les mots dont la longueur est divisible par 3

2. **Donnée:** Deux machines de Turing M_1, M_2 et un mot w

   Question: $w \in L(M_1) \cap L(M_2)$.

3. **Donnée:** Une machine de Turing M, un mot w, et une transition t: $q, a \mapsto q', a', d$ de la machine M

   Question: La transition t est utilisée au plus une fois dans le calcul de M sur w.

4. **Donnée:** une machine de Turing M telle que $L(M) \subseteq \{0, 1\}$

   Question: $L(M) = \{0, 1\}$

5. **Donnée:** Les codes de deux machines de Turing M_1, M_2

   Question: pour tout w tel que M_1 ne s'arrête pas sur w, M_2 s'arrête sur w en moins de $2 \times |w|$ étapes de calcul.

6. **Donnée:** Un entier n et une machine de Turing M

   Question: Pour tout mot w, M effectue au plus n étapes de calcul sur w

7. **Donnée:** Une machine de Turing M et un mot w

   Question: Existe-t-il un entier $n \geq 1$ et une configuration γ de la machine telle que $\gamma_0 \vdash_M^* \gamma \vdash_M^{2n} \gamma$? (Autrement dit, la machine M boucle en un nombre pair d'étapes sur une configuration du calcul de M sur w).

8. **Donnée:** Une machine de Turing M qui s'arrête sur toute donnée

   Question: M accepte un mot de la forme $u \cdot u$

9. **Donnée:** Le code d'une machine de Turing M qui s'arrête sur toute donnée

   Question: M calcule en temps constant

   Note: M calcule en temps constant s'il existe un entier n tel que pour tout w, le calcul de M sur w s'arrête après au plus n étapes.

10. **Donnée:** Une fonction récursive partielle f

    Question: f est elle récursive primitive ?

    Note: f est donnée sous forme d'un arbre fini, étiqueté par les constructions Comp, Prim, Min et dont les feuilles sont étiquetées par les fonctions de base.

Solution

1. C'est indécidable, par le théorème de Rice: la question peut être reformulée en $L(M)$ contient tous les mots de longueur divisible par 3. Il s'agit d'une propriété non triviale des langages récursivement énumérables puisque Σ^* la satisfait et pas $\emptyset$.

   Notes: taux de réussite 89%

2. Indécidable: on réduit L_u, le langage universel (donnée: M, w, question $w \in L(M)$) en choisissant pour M_2 une machine qui accepte tous les mots et $M_1 = M$. Cette réduction est calculable et $w \in L(M_1) \cap L(M_2)$ si et seulement si $< M, w >\in L_u$.

   Notes: taux de réussite 90%. La réduction $f : M, w \mapsto M_1, M_2, w$ consiste juste à recopier le code de M pour M_1, ajouter le code de la machine qui s'arrête immédiatement en acceptant (pour M_2), puis recopier w. Cette fonction f elle-même n'a donc rien à voir avec la machine universelle !

3. Indécidable: on réduit le complémentaire du problème de l'arrêt. Donnée: M', w', question: M' ne s'arrête pas sur w'. Ce problème est indécidable puisque le complémentaire d'un problème indécidable est indécidable.

   On construit la machine M qui a les mêmes états que M', avec un état et supplémentaire q_f et dont les transitions sont

   - $\delta(q, a) = \delta'(q, a) = (q', a', d)$ si $q \in Q$ et $q' \notin \{\textbf{accept}, \textbf{reject}\}$
   - $\delta(q, a) = (q_f, a, , \downarrow)$ si $\delta'(q, a) = (q', a, d)$ et $q' \in \{\textbf{accept}, \textbf{reject}\}$
   - $\delta(q_f, a) = (q_f, a, \downarrow)$

   $w = w'$ et la transition t est $\delta'(q_f, a) = (q_f, a, \downarrow)$.

   La réduction est calculable: la construction ci-dessus fournit un algorithme.

   Si M' ne s'arrête pas sur w', M ne s'arrête pas sur w et la transition t n'est jamais utilisée.

   Si M' s'arrête sur w', la transition t est utilisée plus de deux fois (en fait, une infinité de fois) lors du calcul de M sur w.

   Il en résulte que M' ne s'arrête pas sur w' si et seulement si la transition t est utilisée au plus une fois dans le calcul de M sur w.

   Notes: taux de réussite 73%.

4. Indécidable: on réduit le problème de l'arrêt: donnée M, w, question: M s'arrête sur w.

   On construit la machine M' comme suit: sur la donnée x, M' teste si $x \in \{0, 1\}$, si non elle s'arrête en rejetant. Si oui, elle efface son ruban et simule M sur w. Si M s'arrête sur w, alors M' accepte.

   Le code de M' est calculable.

   Si M s'arrête sur w, alors $L(M') = \{0, 1\}$. Sinon, $L(M') = \emptyset$. Donc M s'arrête sur w ssi $L(M') = \{0, 1\}$.

Notes: taux de réussite 75%. On ne peut pas appliquer directement le théorème de Rice, à cause de la restriction sur les langages dans la donnée du problème.

5. Indécidable: on réduit le problème de l'arrêt universel de M.

   M_1 est la machine M et M_2 est la machine qui boucle sur toute entrée (donc ne s'arrête jamais).

   Les codes de M_1, M_2 sont calculables.

   Si M s'arrête sur toute donnée, alors, pour tout w tel que M ne s'arrête pas sur w, M_2 s'arrête sur w en moins de $2 \times |w|$ étapes puisqu'un tel w n'existe pas.

   Réciproquement, s'il existe un w tel que M ne s'arrête pas sur w, alors M_2 calcule en plus de $2 \times |w|$ étapes sur ce mot là.

   Ainsi, M s'arrête sur toute entrée ssi, pour tout w sur lequel M_1 ne s'arrête pas, M_2 s'arrête sur w en moins de $2 \times |w|$ étapes.

   Notes: taux de réussite 77%

6. Le problème est décidable: il suffit de construire l'arbre de toutes les transitions à profondeur n. Plus formellement, on ordonne totalement l'alphabet $\Sigma \setminus \{\$, B\}$. On ordonne les mots suivant leur longueur et, à égalité de longueur, suivant l'ordre lexicographique. Cet ordre est total, a pour plus petit élément le mot vide. Chaque mot w a un successeur $s(w)$, calculable. et on constuit une machine à plusieurs rubans, qui, sur la donnée $< M >, n$ exécute le programme suivant:

   (a) recopier n sur un ruban auxilliaire r_0
   (b) écrire ϵ sur un ruban auxilliaire r_1
   (c) Tant que le mot du ruban r_1 a une longueur inférieure ou égale à l'entier du ruban r_0 faire:
       i. À l'aide de M_u simuler (sur des rubans r_2, r_3, r_4) M sur le contenu w du ruban r_1, en s'assurant qu'au plus n étapes de M sont simulées: $e := true$, $i := n$. Tant que e et $i \neq 0$ faire: (simuler une étape de calcul de M. Si on arrive dans un état final, $e := false$. Sinon $i := i - 1$)
       ii. Si e alors s'arrêter en rejet. Sinon remplacer w sur le ruban r_1 par $s(w)$.
   (d) S'arrêter en accept.

   La machine ainsi construite s'arrête toujours puisqu'il n'y a qu'un nombre fini de mots de longueur inférieure ou égale à n (moins de $|\Sigma|^{n+1}$) et qu'au plus n étapes de M sur chaque mot sont simulées.

   La machine accepte si et seulement si M s'arrête en au plus n étapes sur tous les mots de longueur au plus n.

   Or si M s'arrête en au plus n étapes sur tous les mots de longueur au plus n, elle s'arrête sur tous les mots, quelle que soit leur longueur, puisque sur un mot w, M ne pourra lire en n étapes qu'un préfixe de longueur au plus n de w.

   La machine construite décide donc l'arrêt en moins de n étapes de M sur tous les mots.

   Notes: taux de réussite: 63%.

7. Indécidable. On réduit le problème de l'arrêt. À M, w on associe M', w où M' est la machine à deux rubans qui (sauf quand M va s'arrêter) effectue les mêmes calculs que M sur le premier ruban et, à chaque étape, écrit un symbole sur le deuxième ruban et se déplace vers la droite.

   Lorsque M va s'arrêter, M' passe dans un état spécial q_s et boucle ($\delta(q_s, a) = (q_s, a, \downarrow)$), sans rien faire sur le deuxième ruban.

   La réduction est calculable.

   De plus, si M s'arrête sur w, alors M' parvient dans une configuration $\gamma = (q_s, (w, w'), (w_2, \epsilon))$ et $\gamma \vdash_{M'} \gamma$, donc aussi $\gamma \vdash^2_{M'} \gamma$.

   Inversement, si M ne s'arrête pas sur w, la nième configuration du calcul de M' sur w est de la forme $(q, (w, w'), (\alpha^n, \epsilon))$, ce qui assure que deux configurations accessibles quelconques sont distinctes.

   Notes: taux de réussite 75%

8. Indécidable. On réduit le problème de l'arrêt. À la machine M et au mot w on associe la machine M' qui, sur une donnée x, commence par vérifier que x peut s'écrire $u \cot u$ (ceci est possible: le langage $\{u \cdot u \mid u \in \Sigma^*\}$ est récursif). Si ce n'est pas le cas, elle refuse. Si c'est le cas, elle copie w sur un deuxième ruban et effectue au plus $|u \cdot u|$ transitions de M sur w: à chaque étape de calcul de M sur le deuxième ruban, une lettre est effacée sur le premier ruban. Si le premier ruban est vide, M' s'arrête et refuse.

   Si M s'arrête avant que le crédit du premier ruban soit épuisé, M' accepte.

   M' s'arrête bien sur toute donnée, la première phase terminant toujours et la deuxième phase comportant au plus $|x|$ étapes de la machine à deux rubans.

   Si M s'arrête sur w en N étapes, alors, pour un mot u de longueur N, M' accepte $u \cdot u$. Réciproquement, si M' accepte $u \cdot u$ alors M s'arrête sur w.

   Notes: taux de réussite 63%. Il est aussi possible de réduire PCP, mais ce n'est pas plus simple.

9. Indécidable. On réduit le problème de l'arrêt. À M, w on associe la machine M' à deux rubans qui, sur la donnée x, écrit w sur le deuxième ruban, puis effectue au plus $|x|$ étapes du calcul de M sur w (cf. ci-dessus): à chaque étape du calcul de M, M' avance d'une lettre sur le premier ruban. Quand M va accepter ou refuser, M' s'arrête et, quand M' lit un blanc sur le premier ruban, elle s'arrête aussi.

   Si M s'arrête sur w en N épapes, M' (à deux rubans) effectue $2|w| + 1 + \min(N, |x|)$ étapes de calcul. On peut donc constuire M'' à un seul ruban qui effectue le même calcul en au plus $k \times (2|w| + 1 + \min(N, |x|))^2$ étapes, pour un k fixé. Le temps de calcul de M' est donc borné par $k \times (2|w| + 1 + N)^2$, qui ne dépend pas de l'entrée x: M' calcule en temps constant.

   Réciproquement, si M ne s'arrête pas sur w, pour toute entrée x, M' ne s'arrête qu'après avoir entièrement lu la donnée x. Le temps de calcul de M' n'est pas constant (il est quadratique pour la machine à un ruban M'' correspondante).

De plus M' (et donc la machine M'') s'arrête sur toute donnée.

Notes: taux de réussite: 39%. En principe, quand on parle de temps de calcul, il s'agit d'une machine à un seul ruban. Mais ici, cela ne faisait aucune différence puisque le temps constant sur une machine à deux rubans est aussi le temps constant sur une machine à un ruban. Il ne s'agit donc pas d'une erreur. En revanche, si on efface le mot d'entrée lettre à lettre, comme dans la question précédente, la réduction devien incorrecte puisque M' doit commencer par lire en entier l'entrée.

10. Indécidable. On réduit le problème de l'arrêt. Étant donnés M, w, On peut construire effectivement, comme vu en cours, une fonction récursive primitive g_M qui, au code d'une configuration $< \gamma >$ et à un entier n associe $< \gamma' >$ telle que $\gamma \vdash_M^n \gamma'$ si M ne s'est pas arrêtée et 0 sinon. On construit alors $f_{M,w} = \min\{n \mid g(< \gamma_0 >, n) = 0\}$.

    Cette fonction est récursive partielle et on peut calculer à partir de M, w l'arbre qui la représente.

    Si M s'arrête sur w, $f_{M,w}$ est une fonction constante, donc récursive primitive (elle sécrit par composées de successeurs et de la constante 0).

    Réciproquement, si M ne s'arrête pas sur w, $f_{M,w}$ est indéfinie et donc pas récursive primitive.

Notes: taux de réussite: 12%. Très peu de réponses correctes. Il ne s'agissait pas de décider si l'écriture de f comporte un min. La "fonction f" peut avoir plusieurs représentations sous forme arborescente !