Informatique théorique III
Examen hiver

Prof. Nestmann, 2005

Nom :

Durée : 225 min. **Rendu à :**

Points :

Veuillez s'il vous plaît prendre connaissance des points suivants :

- Résolvez chaque problème sur une feuille *séparée*.
- Notez votre nom et prénom sur chaque feuille rendue et rendez la donnée de l'examen.
- Les questions d'un problème sont indépendantes. Si vous ne trouvez pas la réponse à une question vous pouvez admettre le résultat et passer à la question suivante.
- On attachera la plus grande importance à la clarté, à la précision et à la concision de la rédaction. Relisez donc vos preuves d'un oeil critique et incrédule.

L'obtention d'au moins 50% des points garantit la réussite de l'examen.

---

1. Ambiguïté

(30 min., 20%)

Soit $\Sigma = \{\mathcal{A}, \mathcal{B}, \mathcal{C}, \neg, \wedge, \rightsquigarrow,), (\}$ et $G = (\{S\}, \Sigma, P, S)$ avec P tel que :

$$S \to \mathcal{A} \mid \mathcal{B} \mid \mathcal{C} \mid \neg S \mid S \wedge S \mid S \rightsquigarrow S \mid (S)$$

Les symboles $\neg, \wedge, \rightsquigarrow$ représentent réspectivement, la négation, la conjonction et l'implication logique. La grammaire G génère l'ensemble des formules de la logique propositionelle dont les variables sont $\mathcal{A}, \mathcal{B}, \mathcal{C}$.

1. Démontrer l'ambiguïté de la grammaire.
2. Donner (sans preuve) une grammaire G' telle que :
   (a) $\mathrm{L}(G') = \mathrm{L}(G)$
   (b) G' soit non ambiguë.
   (c) L'ordre de priorité des opérateurs soit, du plus faible au plus fort, $\rightsquigarrow, \wedge, \neg$.
   (d) L'opérateur $\rightsquigarrow$ soit associatif à droite, et $\wedge$ associatif à gauche. Par exemple $\mathcal{A} \rightsquigarrow \mathcal{A} \rightsquigarrow \mathcal{A}$ se lit $\mathcal{A} \rightsquigarrow (\mathcal{A} \rightsquigarrow \mathcal{A})$ et $\mathcal{A} \wedge \mathcal{A} \wedge \mathcal{A}$ se lit $(\mathcal{A} \wedge \mathcal{A}) \wedge \mathcal{A}$.
3. Définir une machine M (AFD, AFN, AAP ou MT) telle que $\mathrm{L}(M) = \mathrm{L}(G)$.

2. Racine n^{e} d'un langage (60 min., 25%)

Soit Σ un alphabet et $M = (Q, \Sigma, \delta, q_1, F)$ un AFD. On rappelle d'abord que la fonction de transition étendue $\widehat{\delta}$ a la propriété suivante :

$$\forall q \in Q \,.\, \forall u, v \in \Sigma^* \,.\, \widehat{\delta}(q, u \cdot v) = \widehat{\delta}(\widehat{\delta}(q, u), v)$$

On dira que M est *régulier à gauche* si

$$\forall u, v \in \Sigma^* \,.\, \Big(\widehat{\delta}(q_1, u) = \widehat{\delta}(q_1, v) \Rightarrow \big(\forall w \in \Sigma^* \,.\, \widehat{\delta}(q_1, w \cdot u) = \widehat{\delta}(q_1, w \cdot v)\big)\Big)$$

1. Soit $M_1 = (\{q_1, q_2, q_3\}, \{a, b\}, \delta, q_1, \{q_3\})$ l'AFD défini dans la figure 1.

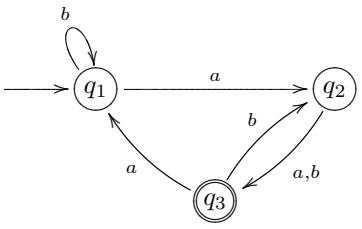

FIG. 1 – L'automate M_1

Montrer que M_1 n'est pas régulier à gauche en donnant deux mots u et v tels que $\widehat{\delta}(q_1, u) = \widehat{\delta}(q_1, v)$ et un mot w tel que $\widehat{\delta}(q_1, w \cdot u) \neq \widehat{\delta}(q_1, w \cdot v)$.

Dans la suite, on admet le théorème suivant :
Théorème Pour tout AFD M, il existe un AFD M^g régulier à gauche tel que $\mathrm{L}(M^g) = \mathrm{L}(M)$.

2. Soit $M = (Q, \Sigma, \delta, q_1, F)$ un AFD régulier à gauche.
Montrer (par induction sur n) que

$$\forall u, v \in \Sigma^* \,.\, \Big(\widehat{\delta}(q_1, u) = \widehat{\delta}(q_1, v) \Rightarrow \big(\forall n \in \mathbb{N}^* \,.\, \widehat{\delta}(q_1, u^n) = \widehat{\delta}(q_1, v^n)\big)\Big)$$

3. **Définition**$(\sqrt[n]{L})$ Soit L un langage sur Σ. On définit, pour $n \in \mathbb{N}^*$, la racine n^{e} de L par $\sqrt[n]{L} = \{u \in \Sigma^* \mid u^n \in L\}$.

Définition$(\sqrt[n]{M})$ Soit $M = (Q, \Sigma, \delta, q_1, F)$ un AFD. On définit, pour $n \in \mathbb{N}^*$,
- $F_n = \Big\{q \in Q \mid \exists u \in \Sigma^* \,.\, \big(\widehat{\delta}(q_1, u) = q \ \wedge\ \widehat{\delta}(q_1, u^n) \in F\big)\Big\}$
- $\sqrt[n]{M} = (Q, \Sigma, \delta, q_1, F_n)$

Soit $M = (Q, \Sigma, \delta, q_1, F)$ un AFD et $n \in \mathbb{N}^*$.

   (a) Montrer que $\mathrm{L}(\sqrt[n]{M}) = \Big\{w \in \Sigma^* \mid \exists u \in \Sigma^* \,.\, \big(\widehat{\delta}(q_1, u) = \widehat{\delta}(q_1, w) \ \wedge\ u^n \in \mathrm{L}(M)\big)\Big\}$.

   (b) On suppose maintenant que M est régulier à gauche
      i. Est-ce que $\sqrt[n]{M}$ est régulier à gauche ? Justifier rapidement.
      ii. Montrer que $\mathrm{L}(\sqrt[n]{M}) \subseteq \sqrt[n]{\mathrm{L}(M)}$ (en utilisant 2).
      iii. Montrer que $\sqrt[n]{\mathrm{L}(M)} \subseteq \mathrm{L}(\sqrt[n]{M})$.
      iv. En déduire que $\mathrm{L}(\sqrt[n]{M}) = \sqrt[n]{\mathrm{L}(M)}$.

4. Conclure en montrant que si $L \subseteq \Sigma^*$ est régulier, alors pour tout $n \in \mathbb{N}^*$ le langage $\sqrt[n]{L}$ est régulier. □

3. Régularité (60 min.,25%)

Rappellons que le *mot miroir* $\overline{w}$ d'un mot w est l'unique mot tel que :

$$|\overline{w}| = |w| \text{ et } \forall i \in [1, |w|] \,.\, (\overline{w})_i = (w)_{|w|+1-i}$$

Soit $\Sigma = \{a, b\}$, on considère les langages suivants avec $m \in \mathbb{N}$:

$$L_m = \left\{ uv\overline{u} \mid u \in \Sigma^+ \wedge v \in \Sigma^m \right\}$$

1. En utilisant la propriété suivante :

   $$w \in L_m \Rightarrow \left(l = |w| - m \text{ est pair } \wedge\ \forall i \in [1, l/2] \,.\, (w)_i = (w)_{|w|+1-i} \right) \qquad (\star)$$

   Montrer que pour tout $n, m \in \mathbb{N}^*$ et $n > i > 0$, on a $b^{n-i}a^m b^n \notin L_m$ (l'induction est inutile).
2. Le langage L_0 n'est pas régulier. En effet, c'est l'ensemble des palindromes de longueur paire sur Σ.

   Montrer que pour tout $m > 0$, L_m n'est pas non plus régulier. Pour cela, prendre $m > 0$, attention c'est tout ce que l'on sait de m, et montrer que L_m n'est pas régulier.
3. Considérons maintenant le langage L_* suivant :

   $$L_* = \left\{ uv\overline{u} \mid u \in \Sigma^+ \wedge v \in \Sigma^* \right\}$$

   (a) Donner une expression régulière α telle que $L_* = \mathrm{L}(\alpha)$.

   (b) Démontrer que $L_* = \mathrm{L}(\alpha)$.
4. Le langage $L = \bigcup_{m \in \mathbb{N}} L_m$ est-il régulier ? Justifier. □

4. Indécidabilité (75 min.,30%)

Dans ce problème, on s'intéresse, pour $i \in \{0, 2, 3\}$ à la décidabilité du problème INCLUS$_i$:

INCLUS$_i$: « Étant donnés deux langages $L, L' \in \mathcal{L}_i$, est-ce que $L \subseteq L'$? »

On définit aussi, pour $i \in \{0, 2, 3\}$ le problème VIDE$_i$:

VIDE$_i$: « Étant donné un langage $L \in \mathcal{L}_i$, est-ce que $L = \emptyset$? »

On rappelle que
- $\mathcal{L}_0$ est l'ensemble des langages semi-décidables (acceptés par les machines de Turing).
- $\mathcal{L}_2$ est l'ensemble des langages non-contextuels (générés par les grammaires non-contextuelles).
- $\mathcal{L}_3$ est l'ensemble des langages réguliers (acceptés par les automates finis déterministes).

0. Soit A et B deux ensembles. Montrer que $A \subseteq B \Leftrightarrow A \cap \overline{B} = \emptyset$.
1. Dans cette question, on traite le cas $i = 3$, le cas des langages réguliers.
   (a) Soit $M = (Q, \Sigma, \delta, q_0, F)$ un AFD.
   Donner un AFD $\overline{M}$ tel que $\mathrm{L}(\overline{M}) = \overline{\mathrm{L}(M)}$. Justifier.
   (b) On admet l'existence d'un algorithme qui, étant donné un AFD M détermine si, oui ou non, on a $\mathrm{L}(M) = \emptyset$ (c'est-à-dire un algorithme qui résout le problème VIDE$_3$). Décrire alors un algorithme qui résout le problème INCLUS$_3$, c'est-à-dire un algorithme qui prend en entrée deux AFD M et M' et détermine si, oui ou non, on a $\mathrm{L}(M) \subseteq \mathrm{L}(M')$.
2. Dans cette question, on traite le cas $i = 2$.
   (a) Soit $\Sigma = \{a, b\}$. Donner une grammaire G non-contextuelle telle que le langage de G soit l'ensemble de tous les mots sur Σ qui ne sont pas des palindromes, c'est-à-dire telle que $\mathrm{L}(G) = \{w \in \Sigma^* \mid w \neq \overline{w}\}$.

   Dans la suite, on admet que :
   Si Σ est un alphabet quelconque, alors $\{w \in \Sigma^* \mid w \neq \overline{w}\}$ est un langage non-contextuel (c'est le résultat précédent généralisé au cas d'un alphabet quelconque).
   (b) Montrer que s'il existe un algorithme qui, étant donné deux grammaires non-contextuelles G et G', détermine si, oui ou non, on a $\mathrm{L}(G) \subseteq \mathrm{L}(G')$, alors il existe un algorithme qui détermine si, oui ou non, une instance du PCP admet une solution.
   On pourra s'intéresser en particulier au cas où G' génère un langage de la forme $\{w \in \Sigma^* \mid w \neq \overline{w}\}$.
   Comme PCP est indécidable – et donc qu'il n'existe pas de tel algorithme pour PCP – , cela montre qu'il n'existe pas de tel algorithme pour INCLUS$_2$.
   (c) Il existe un algorithme qui résout le problème VIDE$_2$, c'est-à-dire un algorithme qui étant donné une grammaire non-contextuelle G détermine si, oui ou non, on a $\mathrm{L}(G) = \emptyset$.
   Expliquer brièvement pourquoi une construction similaire à celle faite en 1b ne peut pas s'appliquer et donc ne contredit pas le point précédent.
3. On s'intéresse maintenant au cas $i = 0$.

(a) En utilisant le théorème de Rice, montrer que le problème VIDE_0 est indécidable.

(b) En déduire, par réduction de VIDE_0 à INCLUS_0 que INCLUS_0 est indécidable. On représentera le problème VIDE_0 par le langage $\{[M] \mid \mathrm{L}(M) = \emptyset\}$ et le problème INCLUS_0 par le langage $\{[(M, M')] \mid \mathrm{L}(M) \subseteq \mathrm{L}(M')\}$ (où $[(M, M')]$ est un codage (injectif) du couple (M, M')).

□