Examen du cours de L3: logique et calculabilité

16 janvier 2013, durée: 3 heures

Tous les documents sont autorisés, les appareils électroniques sont interdits. Les 11 questions sont indépendantes et approximativement listées par difficulté croissante. Les premières questions ont des solutions très courtes (moins d'une page de corrigé pour l'ensemble des 7 premières questions). La dernière question est bonus..

Pour chacun des problèmes suivants, dire s'ils sont décidables ou non. Justifier. (NB: quelques définitions sont données ou rappelées en fin d'énoncé).

1. **Donnée:** Un arbre fini Π, étiqueté par des clauses du premier ordre, une clause C, un ensemble fini de clauses S

   Question: Est-ce que Π est une preuve de C par résolution et factorisation binaire à partir des hypothèses S ?

2. **Donnée:** Deux ensembles finis de clauses du premier ordre S_1, S_2

   Question: Est ce que, pour toute clause $C \in S_2$, $S_1 \models C$?

3. **Donnée:** Une machine de Turing quelconque M_1, et une machine de Turing M_2, qui calcule en temps polynomial

   Question: $L(M_1) \cap L(M_2) = \emptyset$?

4. **Donnée:** Un ensemble fini de clauses de Horn S du premier ordre

   Question: La clause vide est déductible de S par résolution et factorisation binaires ?

5. **Donnée:** Une machine de Turing M telle que, pour tout état q, $\delta(q, B) = q', B, \leftarrow$ (elle écrit toujours B quand B est lu sur le ruban) et un mot w.

   Question: M s'arrête-t-elle sur la donnée w ?

6. **Donnée:** Une machine de Turing M

   Question: $L(M)$ contient deux mots w_1, w_2 tels que $|w_2| = 2 \times |w_1| + 1$?

7. **Donnée:** Deux machines de Turing M_1, M_2

   Question: $L(M_1) \subseteq L(M_2)$ ou $L(M_2) \subseteq L(M_1)$?

8. **Donnée:** Un automate cellulaire $\mathcal{A}$

   Question: L'automate s'arrête-t-il en partant de la configuration γ_0 telle que $\gamma_0(0) = q_0$ et $\gamma(z) = q_1$ pour $z \neq 0$?

9. **Donnée:** Une machine de Turing M qui s'arrête sur toutes les données.

   Question: $L(M)$ est infini ?

10. **Donnée:** Un ensemble fini de clauses de Horn S sur le seul prédicat binaire P et qui sont ou bien des formules réduites à un littéral, ou bien des clauses binaires $P(u, v) \to P(u', v')$ avec u sous-terme de u' et v sous-terme de v'.

    Question: S satisfaisable ?

11. **Donnée:** Une fonction récursive primitive f à n arguments

    Question: Pour tous $k_1, \ldots, k_n \in \mathbb{N}$, $f(k_1, \ldots, k_n) = 0$?

Définitions

- Une machine de Turing M *calcule en temps polynomial*, s'il existe un polynome P_M à coefficients entiers, tel que, pour tout mot w, le nombre d'étapes de calcul de M sur la donnée w est borné par $P_M(|w|)$.
- Une *clause de Horn* (en calcul des prédicats) est une disjonction de littéraux, dont au plus un est positif. On peut donc écrire une clause de Horn $\neg\phi_1 \vee \cdots \vee \neg\phi_n \vee \phi$ ou bien $\neg\phi_1 \vee \cdots \vee \neg\phi_n$, où $\phi_1, \ldots, \phi_n, \phi$ sont des formules atomiques.
- Un *automate cellulaire* est donné par un ensemble fini d'états Q, deux états distincts $q_0, q_1 \in Q$ et une application δ de Q^3 dans $Q \uplus \{s\}$. Une *configuration* de l'automate est une application γ de $\mathbb{Z}$ dans Q. Étant donnés un automate cellulaire $\mathcal{A}$ et une configuration γ, la configuration suivante de $\mathcal{A}$ est la configuration γ' telle que, pour tout $z \in \mathbb{Z}$, $\gamma'(z) = \delta(\gamma(z-1), \gamma(z), \gamma(z+1))$. On note alors $\gamma \vdash_{\mathcal{A}} \gamma'$. L'automate *s'arrête en partant de la configuration* γ_0, s'il existe une suite de configurations $\gamma_0 \vdash_{\mathcal{A}} \cdots \vdash_{\mathcal{A}} \gamma_n$ et un entier $z \in \mathbb{Z}$ tels que $\gamma_n(z) = s$.

Solution

1. C'est décidable: il s'agit de vérifier que Π est bien un arbre de preuve de $S \vdash C$. Pour cela, il faut et il suffit que:

   (a) Les feuilles sont étiquetées par des renommages des clauses de S

   (b) La racine est étiquetée par un renommage C

   (c) Pour chaque noeud étiqueté par C_1, qui n'est pas une feuille, ou bien il a un fils unique, étiqueté C_2 tel que C_1 est obtenu par factorisation binaire de C_2, ou bien il a exactement deux fils étiquetés C_2, C_3 et C_1 est obtenu par résolution binaire à partir de C_2, C_3

   Il suffit donc de parcourir l'arbre en effectuant ces vérifications, qui sont toutes décidables; par exemple pour la résolution binaire, il suffit de parcourir les littéraux des deux clauses parentes et, pour chaque paire de littéraux complémentaires unifiables, tester si la clause obtenue est, à renommage près, la clause obtenue par résultion sur les deux littéraux en question. Au moins un des tests doit être un succès.

2. C'est indécidable. On réduit le problème de la satisfaisabilité d'un ensemble fini de clauses du premier ordre S. En choisissant $S_2 = \{\bot\}$ et $S_1 = S$, on note que S est satisfaisable si et seulement si, pour toute clause $C \in S_2$, $S_1 \models C$.

3. C'est indécidable. On réduit le probème du vide du langage reconnu par une machine de Turing M (problème indécidable par le théorème de Rice): on choisit pour M_2 une machine qui accepte tous les mots (elle calcule en temps constant) et $M_1 = M$. On a bien $L(M_1) \cap L(M_2) = \emptyset$ ssi $L(M) = \emptyset$.

4. C'est indécidable: la satisfaisabilité d'un ensemble fini de clauses de Horn du premier ordre a été montrée indécidable: on peut remarquer que, dans la preuve du cours d'indécidabilité de la satisfaisabilité d'une formule du premier ordre, la formule construite est une conjonction de clauses de Horn. Or, par complétude réfutationnelle de la résolution+ factorisation binaires, un ensemble de clauses S est insatisfaisable ssi on peut en déduire la clause vide.

5. C'est décidable: Sur la donnée w, toutes les configurations accessibles sont de la forme (q, w_1, w_2) où $w_1 w_2 \in \$\Sigma^{|w|}(\epsilon + B)$. Il y en a donc au plus $|\Sigma|^{|w|+2} \times (|w| + 2) \times (|Q| + 2)$. Donc la machine M ne s'arrête pas sur w si et seulement si, il existe une configuration accessible γ telle que $\gamma_0 \vdash^* \gamma \vdash^+ \gamma$.

   On peut alors décider l'arrêt de la manière suivante: sur $< M, w >$, M_0 calcule d'abord un compteur $c = |\Sigma|^{|w|+2} \times (|w|+2) \times (|Q|+2)$. On simule ensuite au plus c étapes de M sur w. Si M passe dans un état d'arrêt, alors M_0 accepte. Si le nombre d'étapes de calcul atteint c sans que M se soit arrêté, M_0 rejette.

6. C'est indécidable: il suffit d'appliquer le théorème de Rice. Cette propriété est en effet une propriété non-triviale des langages récursivement énumérables puisque le langage vide ne la satisfait pas, alors que le langage plein la satisfait.

7. C'est indécidable: Par le théorème de Rice, le problème de savoir si, étant donné M, $L(M) \subseteq a^*$ ou bien $a^* \subseteq L(M)$ est indécidable. En effet, cette propriété est non triviale puisque a^* la satisfait et b^* ne la satisfait pas. On peut ensuite réduire ce problème à celui de l'énoncé en choisissant pour M_2 une machine qui accepte a^*.

8. C'est indécidable: on réduit le problème de l'arrêt sur le mot vide. Étant donnée la machine $M = (Q, q_0, \Sigma, \{\$, B\}, \delta)$, on construit l'automate cellulaire $\mathcal{A}$ comme suit: $Q_{\mathcal{A}} = Q \times \Sigma \uplus \Sigma$, $q_0^{\mathcal{A}} = (q_0, \$)$, $q_1 = B$ et $\delta_{\mathcal{A}}$ est défini par:

   - $\delta_{\mathcal{A}}(a, b, c) = b$ si $a, b, c \in \Sigma$

- $\delta_{\mathcal{A}}(a,b,(q,c)) = \begin{cases} (q',b) & \text{Si } \delta(q,c) = q',c',\leftarrow, \quad q' \notin \{\textbf{accept},\textbf{reject}\} \\ s & \text{Si } \delta(q,c) = q',c',\leftarrow, \quad q' \in \{\textbf{accept},\textbf{reject}\} \\ b & \text{Sinon} \end{cases}$
- $\delta(a,(q,b),c) = \begin{cases} (q',b') & \text{Si } \delta(q,b) = q',b',\downarrow, \quad q' \notin \{\textbf{accept},\textbf{reject}\} \\ s & \text{Si } \delta(q,b) = q',b',\downarrow, \quad q' \in \{\textbf{accept},\textbf{reject}\} \\ b' & \text{Si } \delta(q,b) = q',b',d, \quad d \in \{\leftarrow,\rightarrow\} \end{cases}$
- $\delta((q,a),b,c) = \begin{cases} (q',b) & \text{Si } \delta(q,a) = q',a',\rightarrow, \quad q' \notin \{\textbf{accept},\textbf{reject}\} \\ s & \text{Si } \delta(q,a) = q',a',\rightarrow, \quad q' \in \{\textbf{accept},\textbf{reject}\} \\ b & \text{Sinon} \end{cases}$
- s dans tous les autres cas.

Le codage $c(\gamma)$ d'une configuration $\gamma = (q, w_1, w_2)$ de M est défini par:

- $c(\gamma)(n) = a$ si $0 \leq n \leq |w_1| - 1$ et a est la $n+1$ième lettre de w_1
- $c(\gamma)(|w_1|) = (q,a)$ si a est la première lettre de w_2, ou bien B si $w_2 = \epsilon$
- $c(\gamma)(n) = a$ si $|w_1| < n < |w_1| + |w_2|$ et a est la lettre en position $n - |w_1| + 1$ dans w_2
- $c(\gamma)(z) = B$ dans les autres cas.

On montre alors que $\gamma \vdash_M \gamma'$ entraine $c(\gamma) \vdash_{\mathcal{A}} c(\gamma')$. De plus. si γ est une configuration finale, il existe $z \in \mathbb{Z}$ (la position de la tête de lecture dans la configuration finale) telle que$c(\gamma)(z) = s$ et $c(\gamma_0) = \gamma_0^{\mathcal{A}}$. Il en résulte que M s'arrête sur le mot vide si et seulement si $\mathcal{A}$ s'arrête en partant de $\gamma_0^{\mathcal{A}}$.

9. C'est indécidable. On réduit le problème de l'arrêt. Soit M et w une instance de ce problème. On construit M' comme suit:

   - M' prend en entrée un compteur c.
   - M' simule c étapes de M sur w.
   - Si M s'arrête en moins de c étapes, M' accepte, sinon elle rejette.

   M' accepte c ssi M s'arrête en moins de c étapes sur w. $L(M') = \{c \in \mathbb{N} | c \geq \text{longueur du calcul de } M \text{ sur } w\}$ est donc infini ssi M s'arrête sur w.

10. C'est indécidable. On réduit le problème de correspondance de Post modifié. Soit $(u_1, v_1), \ldots, (u_n, v_n)$ une instance de PCP modifié. À chaque lettre de l'alphabet, on associe un symbole de fonction d'arité 1. $\widetilde{u}(x)$ est un terme à une variable x défini par récurrence par $\widetilde{\epsilon} = x$, $\widetilde{a \cdot u} = a(\widetilde{u})$.

    On choisit pour ensemble de clauses S:

    - Les littéraux $P(\widetilde{u_1}(0), \widetilde{v_1}(0))$, $\neg P(x,x)$
    - Les clauses $P(x,y) \rightarrow P(\widetilde{u_i}(x), \widetilde{v_i}(y))$ pour tout i.

    S est satisfaisable ssi S a un modèle de Herbrand (par le théorème de Herbrand). Par récurrence sur k, tout modèle de S contient

    $$\mathcal{H} = \{P(u_{i_k}\widetilde{\cdots} u_{i_1}(0), v_{i_k}\widetilde{\cdots} v_{i_1}(0)) \mid i_1 = 1, i_2, \ldots, i_k \in \{1, \ldots, n\}\}$$

    Ainsi, si S a un modèle, alors ce modèle satisfait $\forall x. \neg P(x,x)$ et donc, pour toute séquence $i_1 = 1, i_2, \ldots, i_k$, $u_{i_k} \cdots u_{i_1} \neq v_{i_k} \cdots v_{i_1}$, et donc PCP modifié n'a pas de solution. Réciproquement, si S n'a pas de modèle, alors $\mathcal{H} \not\models S$. Or, par construction, $\mathcal{H}$ satisfait toutes les clauses, excepté peut-être $\forall x. \neg P(x,x)$. Il existe donc une formule atomique de $\mathcal{H}$ de la forme $P(t,t)$, ce qui entraine que PCP modifié a une solution.

11. C'est indécidable. Idée de la preuve: On réduit le problème de l'arrêt sur le mot vide. M une machine de Turing. On construit la fonction récursive primitive g qui à n associe le codage de la configuration de M après n étapes de calcul, si M ne s'arrête pas en moins de n étapes et 0 sinon: $g(0)$ est le codage de la configuration initiale et $g(n+1)$ s'obtient à partir de $g(n)$ en utilisant uniquement des tests d'égalité, la multiplication, l'addition, le quotient et le reste. g est ainsi récursive primitive.

    Soit $f(n) = (g(n) = 0)$. autrement dit, $f(n) = 0$ si M ne s'arrête pas en moins de n étapes et $f(n) = 1$ sinon. f est la fonction nulle si et seulement si M ne s'arrête pas sur le mot vide.