TD n° 10

Réductions et indécidabilité de la logique du premier ordre

FONCTIONS ET LANGAGES NON RECURSIFS : RÉDUCTIONS

Exercice 10.1. Que pensez-vous des énoncés suivants.

1. $\nexists M_{halt} : \forall \langle M \rangle, w : M_{halt}(\langle M \rangle, w) = halt(\langle M \rangle, w)$
2. $\forall \langle M \rangle, w : \nexists M_{halt} : M_{halt}(\langle M \rangle, w) = halt(\langle M \rangle, w)$

Exercice 10.2. Extrait du cours :

On dit qu'un problème A se réduit à un problème B si, connaissant une réponse à B, on peut trouver une réponse à A. Plus formellement, on dira qu'un langage A est **Turing-réductible** à un langage B si il existe une machine de Turing avec **oracle** B, qui résout le problème A. Une machine de Turing avec oracle B est une machine qui peut, au cours de son calcul, obtenir autant de réponses qu'elle souhaite sur des questions d'appartenance au langage B : est-ce que tel $w \in B$? est-ce que tel autre $w' \in B$? Et l'oracle pour le langage B lui donne des réponses oui/non instantanément.

1. Comment intégrer formellement ce concept aux machines de Turing ? C'est-à-dire, comment définir les appels à l'oracle, et comment définir la façon dont une MT obtient une réponse de l'oracle ?
2. Donner le diagramme d'une machine de Turing avec oracle $B = \{w \in \{0,1\}^* \mid w \text{ est un palindrome}\}$ qui reconnait le langage $A = \{aw \in \{0,1\}^* \mid w \text{ est un palindrome}\}$.
3. Est-ce utile d'autoriser une machine à avoir plusieurs oracles ?
4. Ecrire chacune des réductions suivantes, et indiquer ce que l'on peut en déduire quant à la calculablité/récursivité/décidabilité des fonctions/langages/problèmes :
   (a) réduire L à $aL = \{aw \mid w \in L\}$ pour tout langage L ;
   (b) réduire $L_u = \{\langle M \rangle \# w \mid M \text{ accepte le mot } w\}$
   à $A = \{\langle M \rangle \mid M \text{ s'arrête quand on la lance sur l'entrée vide}\}$;
   (c) réduire le calcul de la fonction **halt** à celui de $f : \langle M \rangle \times w \mapsto \begin{cases} 0 \text{ si } M(bw) \uparrow \\ 1 \text{ sinon} \end{cases}$;
   (d) réduire L_u à $B = \{\langle M \rangle \mid a \in L(M)\}$;
   (e) réduire $L_{stupide} = \{a\}$ à L_u ;
   (f) réduire L_u à $C = \{\langle M \rangle \mid M \text{ s'arrête si on la lance sur l'entrée } abba\}$;
   (g) réduire B à $D \times E$ avec $D = \{\langle M \rangle \mid b \in L(M)\}$ et $E = \{\langle M \rangle \mid a \in L(M) \text{ ou } b \in L(M)\}$.
   (h) réduire L_u à $F = \{\langle M \rangle \mid M(w) \uparrow \text{ pour tout } w\}$;

Exercice 10.3. Le but de cet exercice est de démontrer le résultat suivant :
Théorème. Le problème de savoir si $\Gamma \models \varphi$ dans la logique du premier ordre est indécidable.
Via une réduction depuis L_u.

1. Quelle est le type de l'entrée de notre réduction, quelle est le type de la sortie, et quelle est la relation qui doit exister entre l'entrée et la sortie ?

Soit $M = (Q^M, \Sigma^M, \Gamma^M, \delta^M, q_0^M, B^M, q_F^M)$ et w une instance du problème d'appartenance à L_u. Nous définissons le langage $S = (S_f, S_r)$ avec $S_f = \{(\varepsilon, 0)\} \cup \{(a,1) \mid a \in \Gamma^M\}$ et $S_r = \{(f_q, 2) \mid q \in Q^M\}$. Voici l'idée que nous allons suivre : les variables x correspondront à des mots sur Γ^M, ε sera utilisé pour le mot vide, $a(w)$ sera le mot aw, et $f_q(x,y)$ signifiera que M, sur l'entrée w, peut atteindre la configuration $\bar{x}qy$ (avec $\bar{x}$ le mot miroir de x).

2. Quelle terme représente le mot *abba* ?
3. Quelle formule représente le fait que la machine M atteint l'état q avec la tête de lecture sur la première lettre du mot *abba* ?
4. Quelle formule représente la configuration initiale de M ? Doit-on l'ajouter à Γ ?
5. Quel est le but φ ?

Nous allons maintenant ajouter des formules dans Γ, de façon à ce que la formule correspondant à la configuration $uqav$ soit vraie si et seulement si $\iota_w \vdash^* uqav$.

6. Supposons que M possède la transition suivante : $\delta^M(q_0^M, a) = (q_1^M, b, R)$. Quelle formule faut-il ajouter pour que l'on puisse appliquer la règle de résolution afin d'en déduire que la formule représentant la configuration $bq_1^M w_1 w'$ (avec $w = w_0 w_1 w_2 \dots w_n$ et $w' = w_2 \dots w_n$) est conséquence logique de Γ si $w_0 = a$?
7. Etant donnée M, donner l'ensemble des règles à ajouter pour encoder toutes ses transitions.

Il faut également rajouter les règles suivantes à Γ pour permettre d'écrire sur les symboles initiallement blancs :

pour tout $q \in Q^M$ on ajoute la formule $\forall x : f_q(x, \varepsilon) \iff f_q(x, B)$
pour tout $q \in Q^M$ on ajoute la formule $\forall y : f_q(\varepsilon, y) \iff f_q(B, \varepsilon)$

8. Donner Γ, φ et les étapes du calcul de la résolution qui permettent de conclure que la machine suivante atteint l'état $bq_F^M bb$ sur l'entrée ab :

$$\delta^M(q_0^M, a) = (q_1^M, b, R) \qquad \delta^M(q_1^M, b) = (q_2^M, b, R) \qquad \delta^M(q_2^M, B) = (q_F^M, b, L)$$

9. Argumenter le fait que cette réduction est correcte (c'est-à-dire que la relation attendue entre l'entrée et la sortie est bien vérifiée).

Exercice 10.4. Le but de cet exercice est de montrer que le modèle des machines de Turing à plusieurs rubans et plusieurs têtes de lecture (une tête de lecture indépendante par ruban, et un seul état pour toute la machine) est équivalent au modèle des machines de Turing.

En fonction de l'état et du symbole lu sur chacun des rubans, la machine peut
- changer d'état,
- écrire un symbole sur chaque ruban,
- déplacer chaque tête de lecture vers la droite ou la gauche indépendament les unes des autres.

1. Donner le type de la fonction de transition δ, et donner un exemple de transition.

Dans l'état initial, l'entrée est écrite sur le premier ruban et tous les autres rubans sont vides. Un mot est accepté si et seulement si la machine entre dans l'état final q_F au cours du calcul.

2. Comment simuler une MT *multi-tape* avec une MT *mono-tape* ?

Exercice 10.5. Le but de cet exercice est de montrer que le modèle des machines de Turing non-déterministes est équivalent au modèle des machines de Turing.

Une machine de Turing non-déterministe peut, à une étape de temps donnée (c'est à dire dans un état et pour un symbole lu), avoir plusieurs transitions possibles. Un calcul d'une machine de Turing est une succession de choix parmi l'ensemble des transitions possibles. Un mot est accepté si et seulement si il existe au moins un calcul qui mène à l'état final.

1. Donner le type de la fonction de transition δ, et donner un exemple de transition.
2. Donner une machine de Turing non déterministe qui reconnaisse le langage $L = \{ww \mid w \in \{a,b\}^*\}$.
3. Soit $r = \max\{|\delta(q,x)| \mid q \in Q \text{ et } x \in \Gamma\}$. Que représente q ?
4. Que pourrait représenter une suite finie de lettres sur l'alphabet $R = \{1,\ldots,r\}$?
5. Peut-on construire une machine de Turing qui énumère (c'est-à-dire écrit un à un et se place dans un état particulier q_e à chaque fois que son ruban contient un mot à énumérer) tous les mots finis sur l'alphabet R ?
6. Montrer que tout langage reconnu par une MT non-déterministe est reconnu par une MT déterministe (indication : trois rubans avec chacun leur tête semble un bon choix).