Fondements de l'informatique
Logique, modèles, et calculs

Chapitre: Solutions de certains exercices

Cours INF412

de l'Ecole Polytechnique

Olivier Bournez

bournez@lix.polytechnique.fr

Version du 16 juillet 2023

Solutions de certains exercices

On trouvera dans ce chapitre les corrections de quelques un des exercices. Le lecteur est invité à faire parvenir à bournez@lix.polytechnique.fr une rédaction de la solution [1] de tout exercice qui n'est pas corrigé dans ces pages, ou toute solution qui semble plus élégante que la solution présentée.

Chapitre 1

Exercice 1.3 (page 14). On peut écrire $A \cap B^c = (A \cap A^c) \cup (A \cap B^c) = A \cap (A^c \cup B^c) = A \cap (A \cap B)^c$. De même $A \cap C^c = A \cap (A \cap C)^c$. Donc $A \cap B = A \cap C$ implique $A \cap B^c = A \cap C^c$.

Puisque $(X^c)^c = X$ pour toute partie X de E, on en déduit que $A \cap B^c = A \cap C^c$ implique $A \cap (B^c)^c = A \cap (C^c)^c$ implique $A \cap B = A \cap C$.

Exercice 1.5 (page 21). La fonction $f(x,y) = y + (0+1+2+\cdots+(x+y))$ énumère les éléments de $\mathbb{N}^2$ conformément à la figure.

La fonction f est bien bijective : soit $n \in \mathbb{N}$. Il existe un unique $a \in \mathbb{N}$ tel que $0+1+\cdots+a \leq n < 0+1+\cdots+(a+1)$. L'unique antécédent de n par f est donné par (x,y) avec $y = n-(0+1+\cdots+a)$ et $x = a-y$.

Une autre bijection entre $\mathbb{N}^2$ et $\mathbb{N}$ est donnée par la fonction $g : \mathbb{N}^2 \to \mathbb{N}$ définie par $g(x,y) = 2^x(2y+1)-1$. Le fait que ce soit une bijection découle du fait que tout nombre entier strictement positif est le produit d'une puissance de deux et d'un nombre impair.

Chapitre 2

Exercice 2.3 (page 24). Le raisonnement se fait par l'absurde. Considérons

$$X = \{k \in \mathbb{N} | P(k) \text{ est faux}\}.$$

Si X est non vide, il admet un plus petit élément n.

1. Si possible en LaTeX.

Remarque : ne peut avoir $n \neq 0$. En effet, on sait que pour $n = 0$, si en supposant pour tout entier $k < 0$ la propriété $P(k)$ on déduit $P(0)$, puisqu'il n'y a pas de $k < 0$, cela veut dire qu'on peut déduire $P(0)$ sans aucune hypothèse.

Ceci étant : $P(n-1), P(n-2), \ldots, P(0)$ doit être vrai par définition de X. On obtient une contradiction avec la propriété appliquée en n.

Exercice 2.4 (page 25). On doit d'abord se convaincre que $L^*.M$ est solution de l'équation $X = L.X \cup M$.

Pour $M = \{\epsilon\}$, cela revient à se convaincre que $L^* = L.L^* \cup \{\epsilon\}$. Cela découle de :

$$L^* = \bigcup_{n\in\mathbb{N}} L^n = L^0 \cup (\bigcup_{n\geq 1} L^n) = \{\epsilon\} \cup (\bigcup_{n\geq 0} L.L^n) = \{\epsilon\} \cup L.L^*.$$

On en déduit pour M général que $L^*.M = L.L^*.M \cup \{\epsilon\}.M$, et donc que $L^*.M$ est bien solution de l'équation $X = L.X \cup M$.

Il reste à prouver que c'est l'unique solution. Soit $X \subset \Sigma^*$ vérifiant $X = L.X \cup M$.

Pour prouver que $L^*.M \subset X$, il suffit de prouver que pour tout entier n on a $L^n.M \subset X$, puisque $L^*.M = \bigcup_{n\geq 0} L^n.M$. On prouve par récurrence sur n la propriété $P(n) : L^n.M \subset X$.

$P(0)$ est vraie car $L^0.M = \{\epsilon\}.M = M \subset M \cup L.X = X$.

Supposons $P(n)$ vraie. On a $L^{n+1}.M = L.L^n.M \subset L.X \subset L.X \cup M = X$. Donc $P(n+1)$ est vraie.

Réciproquement, on montre par récurrence la propriété $Q(n)$: tout mot w de X de longueur n appartient à $L^*.M$. Cela donne clairement l'inclusion contraire.

Pour cela, on utilise le second principe d'induction. Supposons que pour tout $k < n$, $Q(k)$ soit vraie. Soit $w \in X$ un mot de longueur n. Puisque $X = L.X \cup M$, donc cas sont à considérer. Soit $w \in M$ et on a directement $w \in L^*.M$ puisque $M \subset L^*.M$.

Soit $w \in L.X$ et on peut écrire $w = u.v$ avec $u \in L$, et $v \in X$. Puisque $\epsilon \notin L$, la longueur de u est non nulle. et donc la longueur de v est strictement plus petite que celle de w. Par hypothèse d'induction, on a $v \in L^*.M$. Donc $w = u.v \in L.L^*.M \subset L^*.M$, ce qui prouve $Q(n)$, et termine la démonstration.

Exercice 2.5 (page 28). Le langage L des expressions entièrement parenthésées formées à partir d'identificateurs pris dans un ensemble A et des opérateurs + et × correspond à la partie de $E = (A \cup \{+, \times\} \cup \{(,)\})^*$ définie inductivement par

(B) $A \subset L$;

(I) $e, f \in L \Rightarrow (e + f) \in L$;

(I) $e, f \in L \Rightarrow (e \times f) \in L$.

Exercice 2.6 (page 30). La preuve est similaire à la preuve du théorème 2.4 (qui est une généralisation de ce phénomène).

Exercice 2.7 (page 32). Soit $P(x)$ la propriété : "x est non vide et sans sommet avec un seul fils non vide". Clairement, $P(x)$ est vraie pour $x = (\emptyset, a, \emptyset)$. Si on suppose $P(g)$ et $P(d)$, pour $g, d \in ABS$, clairement $P(x)$ est aussi vraie pour $x = (g, a, d)$. La première propriété est donc vérifiée.

Soit $P(x)$ la propriété $n(x) = 2f(x) - 1$. On a $P(x)$ pour $x = (\emptyset, a, \emptyset)$, puisque $n(x) = 1$, et $f(x) = 1$, et $1 = 2 * 1 - 1$.

Supposons $P(g)$ et $P(d)$ pour $g, d \in ABS$. Considérons $x = (g, a, d)$. On a $n(x) = 1 + n(g) + n(d) = 1 + 2 * f(g) - 1 + 2 * f(d) - 1 = 2 * (f(g) + f(d)) - 1 = 2 * f(x) - 1$.

La propriété est donc vraie pour tout $x \in ABS$.

Chapitre 3

Exercice 3.17 (page 53). Il est clair que la deuxième propriété implique la première : pour toute distribution de valeurs de vérité ν, on a $\overline{\nu}(G) = 1$ si G est une tautologie, et $\overline{\nu}(F) = 0$ si $\neg F$ en est une. Dans les deux cas, $\overline{\nu}((F \Rightarrow G)) = 1$.

Supposons maintenant que la deuxième propriété soit fausse. On peut choisir une distribution de valeurs de vérité ν telle que $\overline{\nu}(\neg F) = 0$, et une distribution de valeurs de vérité ν' telle que $\overline{\nu}'(G) = 0$.

On définit une distribution de valeurs de vérité ν'', en posant pour chaque variable propositionnelle x, $\nu''(x) = \nu(x)$ si x possède au moins une occurrence dans F, et $\nu''(x) = \nu'(x)$ si x ne possède aucune occurrence dans F. Par construction cette distribution de valeurs de vérité coïncide avec ν sur F et ν' sur G. On en déduit que $\overline{\nu}''(F) = \overline{\nu}(F) = 0$ et $\overline{\nu}''(G) = \overline{\nu}'(G) = 0$. Et donc $\overline{\nu}''((F \Rightarrow G)) = 0$. La première propriété est donc nécessairement fausse.

Exercice 3.18 (page 53). Il est clair que si un graphe est coloriable avec k-couleurs, chacun de ses sous-graphes est coloriable avec k-couleurs (les mêmes). La difficulté est de prouver la réciproque.

On introduit pour chaque couple $(u, i) \in V \times \{1, 2, \ldots, k\}$ une variable propositionnelle $A_{u,i}$. On construit un ensemble Γ de formules du calcul propositionnel sur l'ensemble des variables $A_{u,i}$ qui soit satisfiable si et seulement si G est k-coloriable : l'idée est que $A_{u,i}$ code le fait que le sommet u est colorié avec la couleur i. L'ensemble Γ est défini comme $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$, où chacun des Γ_i exprime une contrainte :

— pour Γ_1 : chaque sommet possède une couleur :

$$\Gamma_1 = \{A_{u,1} \vee \cdots \vee A_{u,k} | u \in V\}.$$

— pour Γ_2 : Chaque sommet n'a pas plus qu'une couleur :

$$\Gamma_2 = \{\neg(A_{u,i} \wedge A_{u,j}) | u \in V, 1 \leq i, j \leq k, i \neq j\}.$$

— pour Γ_3 : Chaque arête n'a pas ses extrémités d'une même couleur :

$$\Gamma_3 = \{\neg(A_{u,i} \wedge A_{v,i}) | u \in V, 1 \leq i \leq k, (u, v) \in E\}.$$

En faisant ainsi, un graphe est coloriable avec k couleurs si et seulement si on peut satisfaire toutes les formules de $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$.

On va utiliser le théorème de compacité : Soit Γ_0 une partie finie de Γ. Soient $V_0 = \{u_1, \cdots, u_n\}$ les sommets u tels que $A_{u,i}$ figure dans une des formules de Γ_0. Soit $G_0 = (V_0, E_0)$ le sous graphe déterminé par V_0.

Si on suppose que l'on a un graphe dont tous les sous graphes est coloriable avec k couleurs, en particulier, cela est vrai pour G_0, et donc Γ_0, qui est un sous-ensemble des contraintes exprimant le fait que Γ_0 est k-coloriable, est satisfiable.

Puisque Γ possède toutes ses parties finies satisfiables, par le théorème de compacité, Γ est donc satisfiable. Cela veut dire que G est k-coloriable, puisque G est k-coloriable si et seulement si Γ est satisfiable.

Chapitre 4

Exercice 4.1 (page 58). On écrit :

— $F_1 : ((F \Rightarrow ((F \Rightarrow F) \Rightarrow F)) \Rightarrow (((F \Rightarrow (F \Rightarrow F)) \Rightarrow (F \Rightarrow F)))$ (instance de l'axiome 2.)
— $F_2 : ((F \Rightarrow ((F \Rightarrow F) \Rightarrow F))$ (instance de l'axiome 1.) ;
— $F_3 : ((F \Rightarrow (F \Rightarrow F)) \Rightarrow (F \Rightarrow F))$ (modus ponens à partir de F_1 et F_2) ;
— $F_4 : (F \Rightarrow (F \Rightarrow F))$ (instance de l'axiome 1.) ;
— $F_5 : (F \Rightarrow F)$ (modus ponens à partir de F_3 et F_4).

Exercice 4.2 (page 58). Une direction est facile : si $F_1, F_2, \ldots, F_n$ est une preuve de $F \Rightarrow G$ à partir de T, alors $F_1, F_2, \ldots, F_n, F, G$ est une preuve de G à partir de $T \cup \{F\}$. Réciproquement, on prouve par récurrence sur n que si il existe une preuve de longueur n de G à partir de $T \cup \{F\}$, alors il existe une preuve de $(F \Rightarrow G)$ à partir de T.

Par hypothèse de récurrence, il existe une preuve à partir de T pour chacune des formules $(F \Rightarrow F_1), \ldots, (F \Rightarrow F_n)$, ceci s'appliquant par défaut au cas $n = 1$. Considérons la formule F_n c'est-à-dire G. Trois cas sont possibles :

1. La formule G est un axiome ou une formule de T. On a alors $T \vdash G$. On sait que $(G \Rightarrow (F \Rightarrow G))$ est une instance de l'axiome 1., donc on a $\vdash (G \Rightarrow (F \Rightarrow G))$, et à fortiori $T \vdash (G \Rightarrow (F \Rightarrow G))$. Par modus ponens, on a $T \vdash (F \Rightarrow G)$.
2. G est la formule F. La correction de l'exercice précédent donne une preuve de $(F \Rightarrow F)$.
3. Il existe $i, j < n$ tels que H_j est une formule $(H_i \Rightarrow G)$. Par hypothèse de récurrence, on a $T \vdash (F \Rightarrow H_i)$ et $T \vdash (F \Rightarrow (H_i \Rightarrow G))$. Alors la suite :
   — $((F \Rightarrow (H_i \Rightarrow G)) \Rightarrow ((F \Rightarrow H_i) \Rightarrow (F \Rightarrow G)))$ (instance de l'axiome 2.) ;
   — $((F \Rightarrow H_i) \Rightarrow (F \Rightarrow G))$ (modus ponens)
   — $(F \Rightarrow G)$ (modus ponens)
   est une preuve de $(F \Rightarrow G)$ à partir de $(F \Rightarrow H_i)$ et $(F \Rightarrow (H_i \Rightarrow G))$. En concaténant cette preuve de $(F \Rightarrow H_i)$ et de $(F \Rightarrow (H_i \Rightarrow G))$ à partir de T, on obtient une preuve de $F \Rightarrow G$ à partir de T.

Exercice 4.3 (page 58). Pour la première assertion : supposons $T \cup \{F\} \vdash G$. Par le théorème de la déduction (exercice 4.2), on a $T \vdash (F \Rightarrow G)$. Or la formule $((F \Rightarrow G) \Rightarrow (\neg G \Rightarrow \neg F))$ est une instance de l'axiome 5, donc par modus ponens, on obtient $T \vdash (\neg G \Rightarrow \neg F)$, d'où par le théorème de la déduction à nouveau $T \cup \{\neg G\} \vdash \neg F$.

Supposons maintenant $T\cup\{\neg G\}\vdash\neg F$. Par ce qui précède, on obtient $T\cup\{\neg\neg F\}\vdash\neg\neg G$. Comme $\neg\neg G\Rightarrow G$ est une instance de l'axiome 4., par modus ponens on déduit $T\cup\{\neg\neg F\}\vdash G$. Or $(F\Rightarrow\neg\neg F)$ est une instance de l'axiome 3. : de là, on déduit de toute preuve d'une formule à partir de $T\cup\{\neg\neg F\}$ une preuve à partir de la même formule à partir de $T\cup\{F\}$, et on obtient finalement $T\cup\{F\}\vdash G$.

Pour la seconde assertion : on a $\{\neg F,\neg G\}\vdash\neg F$ par définition, d'où par la première assertion, $\{\neg F,F\}\vdash G$, et de là, $T\vdash G$ si à la fois F et $\neg F$ sont prouvables à partir de T.

Exercice 4.4 (page 58). On a $\{\neg G,\neg G\Rightarrow G\}\vdash G$. En utilisant l'exercice 4.3, item 1, cela revient à dire $\{\neg G,\neg G\}\vdash\neg(\neg G\Rightarrow G)$. Soit $\{\neg G\}\vdash\neg(\neg G\Rightarrow G)$. En utilisant l'exercice 4.3, item 1 à l'envers $\{\neg G\Rightarrow G\}\vdash G$.

Exercice 4.5 (page 58). Supposons $T\cup\{F\}\vdash G$ et $T\cup\{\neg F\}\vdash G$. En appliquant la première assertion de l'exercice 4.3, on obtient $T\cup\{\neg G\}\vdash\neg F$ et $T\cup\{\neg G\}\vdash F$. Par la seconde assertion de l'exercice 4.3, on obtient que toute formule est prouvable à partir de $T\cup\{\neg G\}$, et en particulier $T\cup\{\neg G\}\vdash G$. Par le théorème de la déduction (exercice 4.2), $T\vdash(\neg G\Rightarrow G)$. Il suffit alors d'utiliser l'exercice 4.4 pour déduire $T\vdash G$.

Chapitre 5

Exercice 5.1 (page 76). Seule la dernière écriture correspond à une formule : dans la première et la seconde l'arité de R_2 n'est pas respectée. Dans la troisième, la quantification $\exists R$ n'est pas sur une variable, mais sur un symbole de relation : c'est ce que l'on appelle une formule du second ordre, ce qui n'est pas considéré comme une formule valide dans la définition précédente (i.e. ce polycopié).

Exercice 5.2 (page 77). Il n'y a pas de variable libre ni d'occurrence libre dans la première formule. Dans la deuxième formule, la variable x est libre : sa première occurrence est liée, sa seconde occurrence est libre.

Exercice 5.4 (page 81). Pour éviter des notations trop lourdes, on écrira xRy pour $R(x,y)$, et on s'autorise à ne pas écrire toutes les parenthèses. Il suffit de considérer

$$(\forall x xRx)\wedge(\forall x\forall y(xRy\wedge yRx\Rightarrow x=y))\wedge(\forall x\forall y\forall z(xRy\wedge yRz\Rightarrow xRz)).$$

Exercice 5.7 (page 84). On se contentera ici d'indiquer par une flèche les implications : par exemple, $\Rightarrow$ indique que le membre gauche implique le membre droit.

1. $\Leftrightarrow$
2. $\Leftrightarrow$
3. $\Rightarrow$
4. $\Leftrightarrow$

5. $\Rightarrow$
6. $\Rightarrow$

Exercice 5.8 (page 85). Voici des formes prénexes équivalentes

$$\exists x \forall x' \forall y (P(x) \wedge (Q(y) \Rightarrow R(x')))$$

$$\forall x \forall y \exists x' (P(x') \wedge (Q(y) \Rightarrow R(x)))$$

$$\forall x \exists x' \forall y (P(x') \wedge (Q(y) \Rightarrow R(x)))$$

Chapitre 6

Exercice 6.1 (page 92). Le troisième item de la Définition 6.3 est vrai pour tout symbole de relation R et en particulier pour le symbole = d'arité 2 : en particulier, on a $\forall x_1 \forall x_1' \forall x_2 (x_1 = x_1' \Rightarrow (x_1 = x_2 \Rightarrow x_1' = x_2)$. Puisque l'on $x = x$ par le premier item de la Définition 6.3, si on a $x = y$ alors on a $y = x$ en appliquant le cas où x_1, x_2, x_1' sont respectivement x, x et y.

Exercice 6.6 (page 96). Clairement la dernière assertion découle de la seconde, puisque le première produit un modèle qui ne peut pas être le modèle standard des entiers qui satisfait les axiomes de Robinson : en effet, dans le modèle standard des entiers (dans les entiers), l'addition est commutative.

Pour la première assertion, il suffit de prouver la propriété par récurrence sur n : elle est vraie pour $n = 0$ par l'axiome $\forall x\ \mathbf{0} + x = x$, appliquée en $x = s^m(\mathbf{0})$. Supposons la propriété vraie au rang $n - 1 \geq 0$: on a $s^n(\mathbf{0}) + s^m(\mathbf{0}) = s(s^{n-1}(\mathbf{0})) + s^m(\mathbf{0})$, qui selon l'axiome $\forall x \forall y\ s(x) + y = s(x + y)$ appliquée pour $x = s^{n-1}(\mathbf{0})$ et $y = s^m(\mathbf{0})$ vaut $s(s^{n-1}(\mathbf{0}) + s^m(\mathbf{0}))$ soit $s(s^{n+m-1}(\mathbf{0}))$ par l'hypothèse de récurrence, autrement dit $s^{n+m-1+1}(\mathbf{0}) = s^{n+m}(\mathbf{0})$.

Pour la seconde assertion, il faut par conséquent construire un modèle dont l'ensemble de base contient autre chose que (seulement) les éléments $s^{(n)}(\mathbf{0})$ pour n entier. Voici une façon de procéder : on considère X un ensemble avec au moins deux éléments.

On considère la structure $\mathfrak{M}$ dont l'ensemble de base est

$$M = \mathbb{N} \cup (X \times \mathbb{Z}),$$

et où les symboles $s, +, *, =$ sont interprétés par les conditions suivantes :

- — $=$ est interprété par l'égalité. $s, +, *$ étendent les fonctions correspondantes sur $\mathbb{N}$;
- — pour $a = (x, n)$:
  - — $s(a) = (x, n + 1)$;
  - — $a + m = m + a = (x, n + m)$;
  - — $a * m = (x, n * m)$ si $m \neq 0$, et $(x, n) * 0 = 0$;
  - — $m * a = (x, m * n)$;

— pour $a = (x, n)$ et $b = (y, m)$:
  — $(x, n) + (y, m) = (x, n + m)$;
  — $(x, n) * (y, m) = (x, n * m)$.

(dans ces définitions, $\mathbb{N}$, et $\mathbb{Z}$ désignent les ensembles (standards) usuels). On vérifie que cette structure vérifie bien tous les axiomes de Robinson, et que l'addition n'y est pas commutative.

Exercice 6.9 (page 97). On se contentera de donner le schéma de la preuve : on prouve successivement

— $\forall v(v + \mathbf{0} = v)$
— $\forall v \forall v' v + s(v') = s(v + v')$
— $\forall v(v + \mathbf{1} = s(v))$ où 1 désigne $s(\mathbf{0})$
— $\forall v \forall v'(v + v' = v' + v)$

Par exemple, $\forall v(v + \mathbf{0} = v)$ se prouve en observant que $\mathbf{0} + \mathbf{0} = \mathbf{0}$ et que $\forall v((v + \mathbf{0} = v) \Rightarrow (s(v) + \mathbf{0} = s(v))$. On utilise alors le schéma des axiomes de Peano dans le cas où la formule F est la formule $v + \mathbf{0} = v$ pour déduire $\forall v(v + \mathbf{0} = \mathbf{0})$.

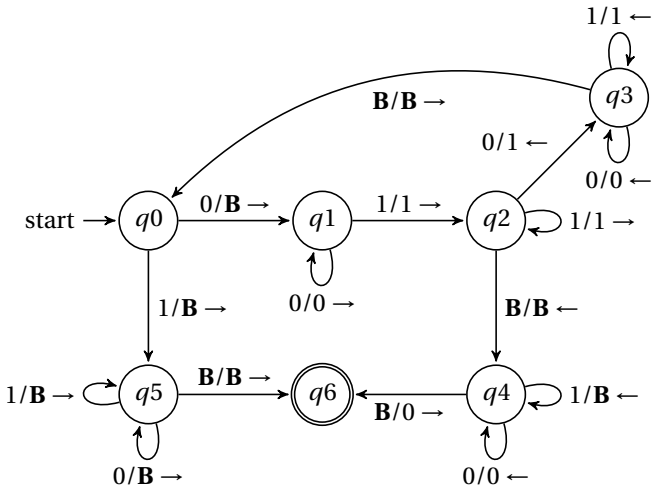
Exercice 6.12 (page 105). Considérons un nouveau symbole de constante c. Ajoutons ce symbole de constante à la signature des axiomes de Peano. Considérons $\mathscr{T}$ défini comme l'union des axiomes de Peano et des formules $\neg c = s^n(\mathbf{0})$, pour n un entier. Tout sous-ensemble fini de $\mathscr{T}$ admet un modèle, car il est inclus dans l'union des axiomes de Peano et des formules $\neg c = s^n(\mathbf{0})$ pour $1 \leq n \leq N$ pour un certain entier N : il suffit d'observer que si l'on interprète c par $N + 1$, alors on obtient un modèle de ce sous-ensemble fini.

Par le théorème de compacité, $\mathscr{T}$ admet un modèle. Ce modèle doit satisfaire $\neg c = s^n(\mathbf{0})$ pour tout entier n. L'interprétation de c n'est donc pas un entier standard. Le modèle est donc non-standard : il contient des "entiers" qui ne sont pas standard.

Exercice 6.13 (page 106). Supposons que $\mathscr{T}$ soit une théorie sur une signature dénombrable qui possède un modèle. Elle est donc cohérente : le corollaire 6.3 s'applique. Observons que la preuve du corollaire 6.3 (en fait la preuve du théorème de complétude) consiste au final à construire un modèle $\mathfrak{M}$ de $\mathscr{T}$ dont le domaine de base est constitué des termes sur la signature. Puisque l'ensemble des termes sur un signature dénombrable est dénombrable, le modèle $\mathfrak{M}$ construit est dénombrable, et est bien un modèle de $\mathscr{T}$.

Chapitre 7

Exercice 7.2 (page 116). Voici une solution : on considère une machine sur l'ensemble d'états $Q = \{q0, q1, q2, \cdots, q6\}$ avec $\Gamma = \{0, 1, \mathbf{B}\}$.



La machine est construite pour effectuer le travail suivant : elle recherche le 0 le plus à gauche et le remplace par un blanc. Elle cherche alors à droite un 1, quand elle en trouve un elle continue à droite jusqu'à trouver un 0 qu'elle remplace par un 1. La machine retourne alors à gauche pour trouver le 0 le plus à gauche qu'elle identifie en trouvant le premier blanc en se déplaçant à gauche et en se déplaçant depuis ce blanc d'une case vers la droite.

On répète le processus jusqu'à ce que :

— soit en cherchant à droite un 0, on rencontre un blanc. Alors les n 0 dans 0^m10^n ont été changés en 1 et $n+1$ des m symboles 0 ont été changés en **B**. Dans ce cas, la machine remplace les $n+1$ symboles 1 par un 0 et n blancs, ce qui laisse $m-n$ symboles 0 sur le ruban. Puisque dans ce cas, $m \geq n$, $m \ominus n = m - n$.

— ou en recommençant le cycle, la machine n'arrive pas à trouver un 0 à changer en blanc, puisque les m premiers 0 ont déjà été changés en **B**. Alors $n \geq m$, et donc $m \ominus n = 0$. La machine remplace alors tous les 1 et 0 restants par des blancs, et termine avec un ruban complètement blanc.

Chapitre 9

Exercice 9.1 (page 144). A est décidable.

En effet : Supposons que s vaut 0. Dans ce cas, A est reconnu par la machine de Turing qui compare la lettre en face de la tête de lecture à 0 et accepte si elle vaut 0, et rejette sinon.

Supposons que s vaut 1. Dans ce cas, A est reconnu par la machine de Turing qui compare la lettre en face de la tête de lecture à 1 et accepte si elle vaut 1, et rejette sinon.

Dans tous les cas, A est donc décidé par une machine de Turing.

Exercice 9.2 (page 155). Il s'agit d'une application directe du Théorème de Rice.

Exercice 9.3 (page 162). Supposons que $S \subset \mathbb{N}$ soit décidable. La fonction χ est calculable, car il suffit sur l'entrée n de déterminer si $n \in S$ et de retourner 1 si c'est le cas, 0 sinon.

Réciproquement, si χ est calculable, S est décidable : sur une entrée n, on calcule χ et on accepte (respectivement refuse) si $\chi(n) = 1$ (resp. $\chi(n) = 0$).

Supposons que $S \subset \mathbb{N}$ soit semi-décidable. La fonction est calculable, car il suffit sur l'entrée n de simuler la machine qui calcule la fonction, et d'accepter si cette simulation termine.

Réciproquement, si la fonction est calculable, S est semi-décidable : sur une entrée n, on simule le calcul de la fonction et on accepte si la simulation accepte.

Exercice 9.1 (page 162). Soit H la fonction calculable définie par : si t est un programme unaire (une machine de Turing) de A, alors $H(\langle t\rangle, n)$ donne le résultat de t sur n sinon $H(\langle t\rangle, n) = 0$. Par construction H est un interpréteur pour tous les programmes unaires de A. La fonction H est totale.

On montre que la fonction calculable totale $H'(n) = H(n,n)+1$ n'est pas dans A : en effet, sinon il y aurait un t dans A qui calcule H'. On aurait pour tout n, $H(\langle t\rangle n)$ qui serait le résultat de t sur n, soit $H'(n) = H(n,n)+1$. En particulier $H(\langle t\rangle, \langle t\rangle) = H(\langle t\rangle, \langle t\rangle)+1$. Absurde.

Ce résultat implique l'indécidabilité du problème de l'arrêt.

— En effet, supposons que $Ar(A,n)$ décide de l'arrêt de la machine de Turing A sur l'entrée n. Pour tout programme unaire f, soit f' le programme unaire qui termine toujours défini par $f'(n) = f(n)$ si $Ar(f,n)$, 0 sinon.

— Soit A défini comme l'ensemble des codages des machines de Turing de la forme si $Ar(f,n)$ alors $f(n)$ sinon 0.

— Pour tout programme unaire f qui termine toujours, f' et f calculent la même fonction.

— Toute fonction unaire calculable totale est représentée par une machine de Turing de A. C'est en contradiction avec le résultat précédent.

Exercice 9.4 (page 162). Pour tester si $x \in E$, on énumère les éléments de E jusqu'à trouver soit x, auquel cas on accepte, soit un élément plus grand, auquel cas on refuse.

Exercice 9.5 (page 162). Pour extraire un ensemble décidable infini d'un ensemble récursivement énumérable infini, il suffit d'extraire une sous-suite strictement croissante de la suite des $f(n)$: on part avec $y = max = 0$. Pour $n = 0, 1, \ldots,$

— On calcule $y = f(n)$.

— Si $y > max$ alors on fait $max := y$, et on affiche $f(n)$

Exercice 9.6 (page 163). Pour $n = 0, 1, \ldots,$ on teste si n est dans l'ensemble, et si oui, on l'affiche.

Exercice 9.7 (page 163). Pour tester si $x \in \exists A$, il suffit de tester pour $y = 0, 1, \ldots$ si $(x, y) \in A$. On s'arrête dès qu'un trouve un y.

Tout langage récursivement énumérable est énuméré par une fonction calculable f. Il correspond donc à la projection de l'ensemble $A = \{(f(n), n) | n \in \mathbb{N}\}$. Cet ensemble A est bien décidable.

Exercice 9.10 (page 164). Leur non-décidabilité découle du théorème de Rice.

Le premier est récursivement énumérable : on énumère les triples (a, b, t) avec a et b deux mots distincts, t un entier, et pour chacun on teste si A accepte a et b en temps t. Si oui, on accepte.

Le second n'est pas récursivement énumérable car son complément est récursivement énumérable : on énumère les paires (a, t) et on teste si a est accepté par A en temps t. Si oui on accepte.

Chapitre 10

Exercice 10.1 (page 168). Soit $\mathbb{N}$ le modèle standard des entiers. $Th(\mathbb{N})$ correspond à l'ensemble des formules closes F vraies sur $\mathbb{N}$.

Le théorème d'incomplétude dit qu'il existe donc des formules closes vraies de $Th(\mathbb{N})$ qui ne sont pas prouvables , ou dont la négation n'est pas prouvable à partir des axiomes de Peano, ou de toute axiomatisation "raisonnable" des entiers.

Soit F une telle formule close. Supposons sans perte de généralité que F est satisfaite sur $\mathbb{N}$.

Le théorème de complétude dit que les formules closes prouvables sont exactement celles qui sont vraies dans tous les modèles. Cela veut donc simplement dire qu'il existe d'autres modèles que $\mathbb{N}$ des axiomes de Peano : en particulier un modèle ou F n'est pas vérifiée.

Dit encore autrement, il y a un modèle des axiomes de Peano avec F vérifiée (par exemple $\mathbb{N}$), et un autre modèle où F n'est pas vérifiée.

Le théorème de complétude reste bien compatible car F n'est pas vérifiée dans tous les modèles.

Exercice 10.1 (page 170). On considère la machine de Turing S qui fait les choses suivantes :

- sur tout entrée w
  - obtenir par le théorème de récursion sa propre description $\langle S \rangle$.
  - construire la formule $\psi = \gamma_{S,\epsilon}$ (la formule γ de la preuve du cours pour la machine S)
  - énumérer les formules prouvables tant que l'on a pas produit $\gamma_{S,\epsilon}$.
  - si l'étape d'avant finit par terminer, alors accepter.

La formule $\psi = \gamma_{S,\epsilon}$ de la deuxième étape n'est pas prouvable : en effet, ψ est vraie si et seulement si S n'accepte pas le mot vide, car

- Si S trouve une preuve de ψ, alors S accepte le mot vide, et donc la formule ψ est fausse.
- (si l'arithmétique est cohérente) on ne peut pas prouver de formule fausse, donc ce cas ne peut donc pas se produire.

Maintenant, observons que si S ne trouve pas une preuve de ψ, alors S n'accepte pas le mot vide : par conséquent ψ est vraie sans être prouvable!!

Bref : ψ est vraie mais non-prouvable.

Chapitre 11

Exercice 11.1 (page 182). On va utiliser le fait que la limite existe et est positive pour prouver que $f(n) = \mathcal{O}\big(g(n)\big)$ et $f(n) = \Omega(g(n))$, conformément à la définition de Θ.

Puisque

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c > 0,$$

par la définition d'une limite, il y a un rang n_0 à partir du quel le rapport est entre $\frac{1}{2}c$ et $2c$. Donc $f(n) \leq 2cg(n)$ et $f(n) \geq \frac{1}{2}cg(n)$ pour tout $n \geq n_0$ ce qui prouve exactement ce que l'on souhaite.

Exercice 11.3 (page 183). Le calcul du maximum de n nombres se fait en temps linéaire (voir cours).

Un algorithme de tri (on prend n nombres en entrée et on doit produire en sortie ces mêmes nombres dans l'ordre croissant) comme le tri fusion se fait en temps $\emptyset(n\log n)$: le tri fusion consiste, pour trier n nombres, à partager en deux sous-ensembles de même taille (à 1 près), à trier récursivement chaque sous-ensemble, et à fusionner les résultats. La fusion de listes croissantes peut se faire en temps linéaire en la somme des tailles des listes. En effet, fusionner dans ce cas consiste à répéter l'opération suivante tant que possible : écrire le plus petit élément des deux listes, enlever cet élément de sa liste. Cela donne une complexité globale pour le tri fusion qui vérifie une équation du type $T_n = \mathcal{O}(2 * T_{n/2}) + \mathcal{O}(n)$ ce l'on peut prouver donner $\mathcal{O}\big(n\log n\big)$.

Supposons que l'on se donne n ensembles S_1, S_2, …, S_n, chacun étant un sous-ensemble de $\{1,2,\ldots,n\}$, et que l'on veuille déterminer si l'on peut trouver une paire disjointe parmi ces ensembles. Cela se résout en temps cubique : il suffit de parcourir les paires S_i et S_j, et pour chaque paire de parcourir les éléments de S_i pour savoir si ils sont dans S_j.

Les chapitres suivants contiennent de nombreux exemples de problèmes dont on ne connaît pas de solutions polynomiales : les algorithmes présentés ne sont pas polynomiaux.

Exercice 11.4 (page 183). Le temps est respectivement pour (a),

— multiplié par 4
— multiplié par 8
— multiplié par 400
— multiplié par 2 + 1
— élevé au carré

Et pour (b) :

— augmenté de $2n+1$
— augmenté de $3n^2+3n+1$
— augmenté de $200n+100$
— transformé en $(n+1)\log(n+1)$, soit essentiellement augmenté de $\mathcal{O}\left(n\log n\right)$
— multiplié par 2

Chapitre 12

Exercice 12.4 (page 204). Si P = NP, alors puisque le complément de P est P (il suffit d'inverser l'état d'acceptation et de rejet d'une machine), on doit avoir NP égal à son complément. Donc on ne peut pas avoir NP qui n'est pas égal à son complément.

Exercice 12.6 (page 205). Cet exercice est corrigé dans [Kozen, 2006], dont nous reprenons la correction ici.

Pour le premier item, c'est incorrect parce que l'on ne dit pas comment produire x (de façon déterministe) quand il existe.

Pour le second item : immédiat, car la donnée de x correspond à un certificat vérifiable en temps polynomial.

Pour le troisième item. Supposons que P = NP. Par conséquent B est dans P. En utilisant ce fait, étant donné y de longueur n, on peut faire une recherche binaire sur les chaînes de longueur n pour trouver x tel que $f(x) = y$. En effet, tout d'abord on teste si $(y,\epsilon) \in B$. Si la réponse est non, alors il n'y a pas de tel x : on s'arrête en refusant. Si la réponse est oui, on teste si $(y,0) \in B$. Si oui, il y a un x avec $f(x) = y$ dont le premier bit est 0, et si la réponse est non, alors tous les tels x ont leur premier bit mis 1. Maintenant, selon la réponse précédente, on teste si $(y,00) \in B$ ou si $(y,10) \in B$ selon ce qui est approprié. La réponse détermine le deuxième bit de x. On continue de cette façon jusqu'à tout ce que les bits d'un x avec $f(x) = y$ aient été déterminés.

Les items 4 et 5 sont immédiats par définition.

Pour le 6ième item : On détermine si l'entrée est de la forme $\phi\#t$, et si c'est le cas, on évalue ϕ sur t. Si f est inversible, alors P = NP, parce que ϕ est satisfiable si et seulement s'il existe x tel que $f(x) = \phi\#1^{|t|}$. L'autre direction de l'implication a déjà été prouvée. La dernière affirmation est alors immédiate.

Chapitre 13

Exercice 13.3 (page 218).

— Le problème est dans NP car étant donné un plan de table, on peut vérifier en temps polynomial si pour chaque chevalier, il n'est pas à coté d'un ennemi.
— Nous allons faire la réduction à partir du problème du cycle Hamiltonien
Soit $\mathscr{I} =< G = (V,E) >$ une instance du problème du cycle Hamiltonien.
Maintenant, nous allons transformer cette instance en une instance $\mathscr{I}_2$ du problème des CHEVALIERS DE LA TABLE RONDE de la façon suivante :

— chaque sommet du graphe est un chevalier.
— Deux chevaliers sont des ennemis si et seulement si il n'existe pas une arête dans G impliquant les deux sommets représentés par ces deux chevaliers

Cette transformation peut se faire en temps polynomial (nous avons construit le complémentaire du graphe G).
Il est facile de prouver que :
— Si il existe un cycle Hamiltonien dans G, alors il existe un plan de table. Il suffit de voir qu'une arête dans G correspond au fait que les deux chevaliers ne sont pas ennemis. Donc le plan de table correspond au cycle Hamiltonien.
— Si il existe un plan de table alors il existe un cycle Hamiltonien dans G.

Donc le problème des CHEVALIERS DE LA TABLE RONDE est plus difficile que le problème du cycle Hamiltonien.

Donc le problème des CHEVALIERS DE LA TABLE RONDE est NP-complet.

Exercice 13.4 (page 219).

— Le problème CHAÎNE HAMILTONIENNE est dans NP car si on se donne une chaîne, on peut vérifier en temps polynomial si elle passe une fois et une seule par chaque sommet du graphe et qu'elle a u et v comme extrémité.
— Nous allons faire la réduction à partir du problème CYCLE HAMILTONIEN. Soit $\mathscr{I} =< G = (V,E) >$ une instance du problème CYCLE HAMILTONIEN. Maintenant, nous allons transformer cette instance en une instance $\mathscr{I}'$ du problème CHAÎNE HAMILTONIENNE de la façon suivante : Nous allons construire un graphe $G' = (V',E')$ tel que
  — Soit u un sommet arbitraire de V
  — $V' := V \cup \{v\}$ tel que v est un sommet n'appartenant pas dans V
  — $E' := E \cup \{(v,\ell) : \ell$ est un voisin de u dans $G\}$

  Cette transformation peut se faire en temps polynomial (nous avons juste copier le graphe G en rajoutant un sommet et des arêtes).
  Il est facile de prouver que :
  — Si il existe un cycle hamiltonien dans G, alors il existe une chaîne Hamiltonienne dans G'.
    Soit $C = (u,\ell_1,\ldots,\ell_{n-1},u)$ un cycle hamiltonien dans G. Nous construisons la chaîne $\mathscr{P} = (u,\ell_1,\ldots,\ell_{n-1},v)$ dans le graphe G'. Cette chaîne est hamiltonienne : elle passe une fois et une seule par v et par chaque sommet de G puisque C est un cycle hamiltonien.
  — Si il existe une chaîne Hamiltonienne dans G', alors il existe un cycle hamiltonien dans G.
    Soit $\mathscr{P} = (u,\ell_1,\ldots,\ell_{n-1},v)$ une chaîne dans un graphe G'. Le cycle $C = (u,\ell_1,\ldots,\ell_{n-1},u)$ est Hamiltonien pour le graphe G.

  Donc le problème CYCLE HAMILTONIEN se réduit au problème CHAÎNE HAMILTONIENNE en temps polynomial.

Donc le problème CHAÎNE HAMILTONIENNE est NP-complet.

Exercice 13.5 (page 219).

— Le problème CHAÎNE est dans NP car étant donnée une chaîne, on peut vérifier en temps polynomial si sa taille est de longueur $n/2$ et qu'elle a u et v comme extrémité.
— Nous allons faire la réduction à partir du problème CHAÎNE HAMILTONIENNE. Soit $\mathscr{I} =< G = (V,E), u, v >$ une instance du problème CHAÎNE HAMILTONIENNE.
Nous transformons cette instance en une instance $\mathscr{I}'$ du problème CHAÎNE de la façon suivante. Nous construisons un graphe $G' = (V', E')$ tel que
G' est une copie du graphe G plus une chaîne de $|V|$ sommets dont un seul sommet de cette chaîne est voisin de u.
Cette transformation se fait en temps polynomial (nous avons juste copier le graphe G en rajoutant un sommet et des arêtes).
Il est facile de prouver que
il existe une chaîne Hamiltonienne dans G si et seulement si il existe une chaîne Hamiltonienne dans G' de longueur $\frac{|V'|}{2}$.

Donc le problème CHAÎNE HAMILTONIENNE se réduit au problème CHAÎNE ; et ce dernier est donc NP-complet.

Exercice 13.6 (page 219). Le problème ARBRE est NP-complet. Il suffit simplement de remarquer qu'une chaîne Hamiltonienne est un arbre couvrant ayant 2 feuilles.

Exercice 13.7 (page 220).

Point 1. : Soit S une couverture de sommets du graphe G. Il faut prouver que tous les sommets du graphe sont soit voisins de S ou soit dans S dans le graphe G'.

1. Comme toutes les arêtes de G ont au moins une de leurs extrémités dans S, tous les sommets de G' correspondant à une arête de G sont des voisins de S.
2. Soit v un sommet de V. Si v n'appartient pas à S, alors v est au moins une extrémité d'une arête (car G est connexe). Comme S est une couverture de sommets du graphe G, cela implique que l'autre extrémité de cette arête est dans S. Donc v est voisin d'un sommet de S. Par conséquence, tous les sommets de G sont soit dans S ou soit voisins de S.

Donc S est un ensemble dominant de G'.

Point 2. : Supposons que $S' \subseteq V$. S' domine le graphe G'. Ce qui signifie que toutes les arêtes de G ont une de ces extrémités dans S'. Donc S' est une couverture de graphe G.

Supposons que S' n'est pas un sous-ensemble de V. Il existe un sommet s n'appartenant pas à V dans S'. s est un sommet représentant une arête (u,v) dans le graphe G.

— Si les deux sommets u et v sont dans S', alors $S' \backslash \{s\}$ est un ensemble dominant de G' de cardinal plus petite que S'.
— Si un des deux sommets u et v est dans S', alors $S' \backslash \{s\} \cup \{u,v\}$ est un ensemble dominant de G' de même cardinal que S'.

On réitère le même raisonnement sur cet ensemble jusqu'à supprimer tous les sommets n'appartenant pas à V.

Point 3 :

Donnée: Un graphe non-orienté G, et un entier k.
Réponse: Décider s'il existe un ensemble dominant S de G tel que $|S| \leq k$.
Point 4 : On peut vérifier en temps polynomial si un ensemble de sommets est un ensemble dominant et si il est de cardinal inférieur à k.
Point 5. Il suffit de combiner les questions précédentes.

Exercice 13.8 (page 221). Point 1. Étant donné un sous-ensemble de sommets de K, on peut vérifier en temps polynomiale

1. si S est de cardinal inférieure à k.
2. si, pour tout sommet v de V, sa distance à un des sommets de S est inférieure à b

Point 2. Nous allons faire la réduction à partir du problème DOMINANT. Soit I une instance du problème du dominant minimum : $G = (V, E)$ et un entier k'.
Nous construisons l'instance du problème du k-CENTRE : $k' = k$ et le graphe complet $K = (V, E')$ muni de la fonction de poids sur les arêtes suivantes :

$$w(u,v) = \begin{cases} 1 & si \quad (u,v) \in E \\ 2 & si \quad (u,v) \notin E \end{cases}$$

Cette réduction se réalise en temps polynomiale et elle vérifie les propriétés suivantes :
— Si il existe un ensemble dominant de taille inférieure à k dans G alors K admet un k-centre de coût 1 ;
— Si il existe un ensemble dominant de taille supérieure à k dans G alors K admet un k-centre de coût 2 ;
On peut en déduire qu'il existe un ensemble dominant de taille inférieure à k dans G si et seulement si K admet un k-centre de coût 1.

Chapitre 14

Exercice 14.1 (page 232). On sait que NP $\subset$ PSPACE. On doit montrer inclusion inverse. On considère SAT dans PSPACE, qui est NP-complet. Il est donc NP-dur et donc aussi PSPACE-dur. Donc pour tout langage A dans PSPACE, A se réduit à SAT, et puisque SAT $\in$ NP, on a donc A dans NP.

1 Notes bibliographiques

Plusieurs corrections (et exercices) sont honteusement plus qu'empruntés de l'ouvrage [Arnold and Guessarian, 2005] pour les premiers chapitres. D'autres exercices sont inspirés de résultats démontrés dans [Cori and Lascar, 1993] ou dans [Kleinberg and Tardos, 2005], ou d'exercices de [Sipser, 1997].

Index

Bibliographie

[Arnold and Guessarian, 2005] Arnold, A. and Guessarian, I. (2005). *Mathématiques pour l'informatique*. Ediscience International.

[Cori and Lascar, 1993] Cori, R. and Lascar, D. (1993). *Logique mathématique. Volume I*. Masson.

[Kleinberg and Tardos, 2005] Kleinberg, J. and Tardos, E. (2005). *Algorithm design*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

[Kozen, 2006] Kozen, D. (2006). *Theory of computation*. Springer-Verlag New York Inc.

[Sipser, 1997] Sipser, M. (1997). *Introduction to the Theory of Computation*. PWS Publishing Company.