Licence Informatique, semestre 5 — 2021–22

Eléments de logique pour l'informatique (Info 315) — 17 novembre 2021

Partiel - 27 octobre 2021

L'examen dure 2 heures. L'énoncé est composé de 6 pages, le barème est indicatif.
Toutes les réponses (sauf QCM) devront être clairement justifiées.
Les questions marquées **LDD** sont optionnelles pour les étudiants de la licence Informatique (parcours informatique et Miage) mais comptent dans le barème des étudiants de LDD Informatique, Mathématiques (dont magistère).
Les questions marquées **Licence** sont optionnelles pour les étudiants de LDD Informatique, Mathématiques (dont magistère), leur résultat est admis pour la suite.
Le seul document autorisé est une page A4 manuscripte recto-verso.
Ne pas cacheter les copies !
Correction :

Exercice 1 *QCM (5,5 points)*

Le nom et le numéro d'étudiant doivent être reportés sur l'énoncé du QCM que vous rendrez avec votre copie (utiliser un style bleu ou noir pour cocher les cases). Sauf indication contraire, une bonne réponse rapporte $\frac{1}{4}$, une mauvaise réponse retire $\frac{1}{8}$ point, l'absence de réponse vaut 0 point.
Correction : *Voir correction individuelle des QCM.*

Exercice 2 *Transformation de formules du calcul propositionnel (9 points)*

Le **ou-exclusif**, noté $\oplus$ a pour table de vérité

p	q	$p \oplus q$
V	V	F
V	F	V
F	V	V
F	F	F

Propriétés admises de $\oplus$. On utilisera sans le justifier le fait que le ou-exclusif est commutatif ($p \oplus q \equiv q \oplus p$) et associatif ($(p \oplus q) \oplus r \equiv p \oplus (q \oplus r)$), ainsi que la propriété $p \oplus p \equiv \perp$. On peut écrire des formules $A_1 \oplus \ldots \oplus A_n$ sans se soucier des parenthèses ou de l'ordre dans lequel les formules A_i sont écrites.

Ou-exclusif d'une liste de formules. Si E est une liste de formules $[A_1; \ldots; A_n]$, on note $\bigoplus(E)$ la formule qui est le ou-exclusif des formules de la liste soit $\bigoplus(E) \stackrel{\text{def}}{=} A_1 \oplus \ldots \oplus A_n$. Par convention le ou-exclusif d'une liste vide de formules représente la formule $\perp$.

Monome. On utilise dans la suite le notion de **monome**. Un monome est une formule qui est soit $\top$ soit qui ne contient que des variables propositionnelles et des conjonctions.

L'objet de l'exercice est de montrer que toute formule propositionnelle est équivalente au ou-exclusif d'un ensemble de monomes.

1. Donner la table de vérité des formules $\top \oplus p$ et $\perp \oplus p$, en déduire des formes équivalentes à ces formules en utilisant les connecteurs logiques habituels.

   Correction :

p	$\top \oplus p$	$\perp \oplus p$
V	F	V
F	V	F

   $\top \oplus p \equiv \neg p$ *et* $\perp \oplus p \equiv p$

2. **Licence** Montrer que la formule $(p \oplus q) \wedge r$ est équivalente à $(p \wedge r) \oplus (q \wedge r)$ et que la formule $(p \vee q)$ est équivalente à $p \oplus q \oplus (p \wedge q)$

   Correction :

   On vérifie que les tables de vérité de $(p \oplus q) \wedge r$ et $(p \wedge r) \oplus (q \wedge r)$ sont identiques ainsi que celles de $p \oplus q \oplus (p \wedge q)$ et de $(p \vee q)$

p	q	r	$(p \oplus q) \wedge r$	$(p \wedge r) \oplus (q \wedge r)$
_	_	F	F	F
V	V	V	F	F
V	F	V	V	V
F	V	V	V	V
F	F	V	F	F

p	q	$p \oplus q$	$p \wedge q$	$p \oplus q \oplus (p \wedge q)$
V	V	F	V	V
V	F	V	F	V
F	V	V	F	V
F	F	F	F	F

3. Soit une formule qui s'écrit $A_0 \oplus \ldots \oplus A_n$. Montrer que pour toute interprétation I, on a $I \models A_0 \oplus \ldots \oplus A_n$ si et seulement si l'ensemble $\{i \leq n | I \models A_i\}$ des indices des formules qui sont vraies dans I a un nombre impair d'éléments.

   Correction : *On raisonne par récurrence sur n pour montrer $I \models A_0 \oplus \ldots \oplus A_n$ ssi $|\{i \leq n | I \models A_i\}|$ est impair*

   — *si $n = 0$ alors la propriété s'écrit $I \models A_0$ ssi $|\{i \leq 0 | I \models A_0\}|$ est impair. $\{i \leq 0 | I \models A_0\}$ contient 0 ou 1 éléments, son cardinal est impair ssi il contient 1 élement et donc ssi $I \models A_0$. Le propriété est donc vérifiée pour $n = 0$.*

   — *On suppose la propriété vraie pour n et on veut la montrer pour $n + 1$*

   — *si $I \models A_{n+1}$ alors*

   — *$I \models (A_0 \oplus \ldots \oplus A_n) \oplus A_{n+1}$ ssi $I \not\models A_0 \oplus \ldots \oplus A_n$ (propriété de $\oplus$)*

   — *on a $I \not\models A_0 \oplus \ldots \oplus A_n$ ssi $|\{i \leq n | I \models A_i\}|$ est pair (hypothèse de récurrence)*

   — *on a aussi $|\{i \leq n | I \models A_i\}|$ est pair ssi $|\{i \leq n+1 | I \models A_i\}|$ est impair (car $I \models A_{n+1}$)*

   d'où le résultat souhaité

   — *si $I \not\models A_{n+1}$, le raisonnement est analogue*

$$
\begin{aligned}
I \models (A_0 \oplus \ldots \oplus A_n) \oplus A_{n+1} \quad & \textit{ssi } I \models A_0 \oplus \ldots \oplus A_n \\
& \textit{ssi } |\{i \leq n | I \models A_i\}| \textit{ est impair} \\
& \textit{ssi } |\{i \leq n + 1 | I \models A_i\}| \textit{ est impair}
\end{aligned}
$$

4. Résultats sur les monomes (définition au début de l'exercice)

   (a) Donner une formule sur les variables propositionnelles p, q qui **n'est pas** un monome.

   (b) Montrer que deux monomes sont logiquement équivalents si et seulement ils contiennent exactement les mêmes variables propositionnelles. En déduire une représentation informatique des monomes qui permette simplement de tester l'équivalence de deux monomes.

   (c) Combien y a-t-il de monomes différents qui ne sont pas équivalents entre eux si on a 3 variables propositionnelles ? même question avec n variables propositionnelles.

   (d) Donner une forme équivalente plus simple de la formule $m_1 \oplus m_2 \oplus p$ lorsque m_1 et m_2 sont deux monomes équivalents.

   Correction :

   (a) La formule $p \wedge \neg q$ n'est pas un monome (présence d'une négation)

   (b) Soit un monome m, si une variable x apparait dans le monome, alors comme il n'y a que des conjonctions, le monome est faux dans toute interprétation dans laquelle x est faux. Un monome est vrai dans toute interprétation qui rend vraies toutes les variables du monome. Si une variable x apparait dans le monome m_1 et pas dans le monome m_2 soit I l'interprétation qui rend faux la variable x et vraies toutes les autres variables. On a I rend faux le monome m_1 et vrai le monome m_2. Les deux monomes ne sont donc pas équivalents.

   On peut donc représenter un monome comme l'ensemble des variables propositionnelles qui apparaissent dans le monome (l'ensemble vide correspond au monome $\top$). L'equivalence des monomes se fait par test d'égalité entre les ensembles.

(c) On a autant de monomes non equivalents que de sous-ensembles de variables. Avec trois variables différentes, il y a $2^3 = 8$ *monomes différents qui ne sont pas equivalents entre eux et* 2^n *dans le cas général de* n *variables.*

(d) si $m_1 \equiv m_2$ *alors* $m_1 \oplus m_2 \oplus p \equiv (m_1 \oplus m_1) \oplus p \equiv \bot \oplus p \equiv p$

5. On souhaite maintenant transformer une formule en ou-exclusif d'un ensemble de monomes.
   (a) Soient p, q deux variables propositionnelles et la formule $P \overset{\text{def}}{=} (\top \oplus p \oplus (p \wedge q)) \wedge (\top \oplus q)$. En utilisant les équivalences établies dans les questions précédentes, donner une forme équivalente à P qui ne contient que des ou-exclusifs et des monomes non equivalents entre eux.

   (b) Que se passe-t-il si on veut faire la même opération pour la formule $(\top \oplus p) \wedge p$? donner une forme équivalente.

   (c) La distributivité de la conjonction sur le ou-exclusif, établie à la question 2 se généralise :

$$(p_1 \oplus \ldots \oplus p_n) \wedge (q_1 \oplus \ldots \oplus q_k) \equiv \bigoplus [p_i \wedge q_j | 1 \leq i \leq n, 1 \leq j \leq k]$$

   On note `mand` l'opération qui prend en arguments deux listes de monomes $L = [l_1; \ldots; l_n]$ et $M = [m_1; \ldots; m_k]$ et renvoie la liste des monomes combinés $[l_i \wedge m_j | 1 \leq i \leq n, 1 \leq j \leq k]$
   On a : $\bigoplus(\texttt{mand}(L, M)) \equiv \bigoplus(L) \wedge \bigoplus(M)$
   Donner les équations récursives d'une fonction `fnal` qui prend en argument une formule P du calcul propositionnel sur les connecteurs usuels en **forme normale de négation** et qui renvoie une liste de monomes telle que $P \equiv \bigoplus(\texttt{fnal}(P))$.

   (d) Expliquer comment transformer la liste de monomes pour qu'elle ne contienne pas deux monomes équivalents en préservant l'équivalence. On peut alors considérer la liste comme un ensemble de monomes.

Correction :

(a) On utilise la distributivité puis les règles de simplification des connecteurs $\wedge$ *et* $\oplus$.

$$\begin{aligned} P &\equiv (\top \wedge \top) \oplus (\top \wedge q) \oplus (p \wedge \top) \oplus (p \wedge q)) \oplus (p \wedge q \wedge \top) \oplus (p \wedge q \wedge q) \\ &\equiv \top \oplus q \oplus p \oplus (p \wedge q) \oplus (p \wedge q) \oplus (p \wedge q) \\ &\equiv \top \oplus q \oplus p \oplus (p \wedge q) \oplus \bot \\ &\equiv \top \oplus q \oplus p \oplus (p \wedge q) \end{aligned}$$

(b) On a $(\top \oplus p) \wedge p \equiv (\top \wedge p) \oplus (p \wedge p) \equiv p \oplus p \equiv \bot$ *La formule est equivalente à* $\bot$ *que l'on peut écrire comme le ou-exclusif de deux monomes identiques mais pas comme un ou-exclusif de monomes non equivalents.*

(c) On part d'une formule en forme normale de négation, il suffit donc de traiter le cas de littéraux, des conjonctions et des disjonctions.
On utilise le fait que le ou-exclusif d'une liste vide de formules est $\bot$*, que* $\neg P \equiv \top \oplus P$*, la propriété de* ***mand*** $\bigoplus(\mathit{mand}(L, M)) \equiv \bigoplus(L) \wedge \bigoplus(M)$ *et que* $p \vee q \equiv p \oplus q \oplus (p \wedge q)$

$$\begin{aligned} \mathit{fnal}(\bot) &= [] \\ \mathit{fnal}(\top) &= [\top] \\ \mathit{fnal}(x) &= [x] \\ \mathit{fnal}(\neg x) &= [\top; x] \\ \mathit{fnal}(A \wedge B) &= \mathit{mand}(\mathit{fnal}(A), \mathit{fnal}(B)) \\ \mathit{fnal}(A \vee B) &= \ \textbf{soit } L = \mathit{fnal}(A), M = \mathit{fnal}(B) \\ & \quad\ \textbf{dans } L + M + \mathit{mand}(L, M) \end{aligned}$$

(d) On a vu que si $m_1 \equiv m_2$ *alors* $m_1 \oplus m_2 \oplus p \equiv p$*, on peut donc supprimer de la liste tout couple de monomes qui sont équivalents. Pour construire une version simplifiée équivalente d'une liste, on parcourt cette liste, à chaque monome* m *on regarde s'il y a un monome equivalent* m' *dans le reste de la liste* L*. Si oui on continue en simplifiant la liste* L *privée de* m'*, sinon on garde* m *dans la liste simplifiée et on continue avec la liste* L*. La liste simplifiée ne contient plus de doublons, l'ordre n'a pas d'importance, on peut donc la considérer comme un ensemble.*

6. **LDD** On a montré que toute formule était équivalente à une formule qui était le ou-exclusif d'un ensemble de monomes non-équivalents. Dans la suite on parlera d'ensemble simplifié de monomes pour un ensemble de monomes qui ne contient pas deux monomes équivalents.

   (a) Si on se donne n variables propositionnelles, combien peut-on former d'**ensembles simplifiés de monomes** ?

   (b) En utilisant un argument de cardinalité, en déduire que deux formules $\bigoplus(E_1)$ et $\bigoplus(E_2)$ sont équivalentes si et seulement si les deux ensembles simplifiés de monomes E_1 et E_2 sont égaux.

   (c) En déduire une méthode qui étant donné un ensemble simplifié de monomes E, renvoie si la formule $\bigoplus(E)$ est valide, si elle est satisfiable et non-valide ou si elle est insatisfiable.

   Correction :

   (a) On a vu qu'avec n variables, on pouvait construire 2^n monomes non equivalents donc on peut construire 2^{2^n} ensembles de tels monomes.

   (b) Il y a aussi 2^{2^n} fonctions booléennes à n arguments. On a montré que l'on pouvait associer à chaque fonction booléenne un ensemble de monomes équivalents on a donc une bijection entre les deux ensembles. Deux formules $\bigoplus(E_1)$ et $\bigoplus(E_2)$ sont équivalentes si et seulement si $E_1 = E_2$.

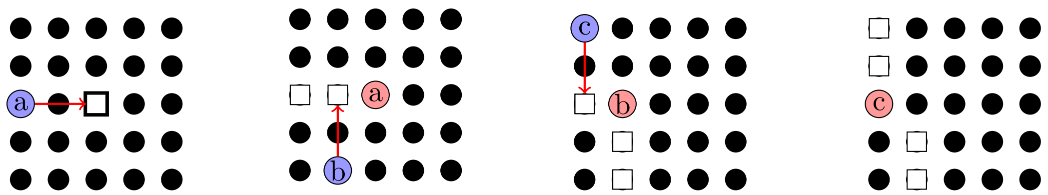
   (c) Si E est un ensemble simplifié de formules on a $\bigoplus(E)$ est valide ssi $\bigoplus(E) \equiv \top$ ssi $E = \{\top\}$ et $\bigoplus(E)$ est insatisfiable ssi $\bigoplus(E) \equiv \bot$ ssi $E = \emptyset$. Finalement $\bigoplus(E)$ est satisfiable et non-valide ssi E est non vide et contient un élément différent de $\top$.

Exercice 3 *Modélisation au premier ordre (5,5 points)*

On modélise le jeu de solitaire. Un plateau de jeu comporte des cases disposées en grille sur lesquelles sont positionnés des pions. Au départ il y a une seule case vide. Un pion peut se déplacer de deux cases suivant l'une des quatre directions (Nord, Sud, Est, Ouest) à condition que la case d'arrivée soit vide et que la case intermédiaire contienne un pion. Ce pion intermédiaire est alors retiré du plateau. Après une étape de jeu, la case de destination qui était vide initialement est occupée mais la case d'origine du pion ainsi que la case intermédiaire sont vides.

L'objectif est de n'avoir à la fin plus qu'un seul pion sur le plateau.

On représente les cases vides par des carrés blancs et les cases occupées par des ronds noirs. La figure ci-dessous contient 4 états successifs du jeu en indiquant avec une flèche le déplacement choisi et en identifiant avec une lettre les pions qui se déplacent.



On modélise de manière logique ce jeu pour une forme de grille quelconque. Les objets de notre théorie représentent soit des positions sur la grille, soit des directions de déplacement soit des états du jeu.

On se donne la signature suivante :

— 4 constantes `N`, `S`, `E`, `O` qui représentent les sens de déplacement ;

— une constante `null` pour représenter une position qui n'est pas dans la grille ;

— une fonction `suivant` à deux arguments : le premier argument représente une position et le second une direction, le résultat est une position correspondant au déplacement d'une seule case dans la direction donnée ; si le déplacement "sort" de la grille alors le résultat est `null` ;

— deux prédicats unaires, `pos` qui est vrai lorsque l'argument est une position (incluant `null`) et `etat` qui est vrai lorsque l'argument représente un état du jeu ;

— un prédicat binaire `pion` : le premier argument s représente un état du jeu et le second une position p, le prédicat sera vrai lorsque s est un état et p est une position différente de `null` et que, dans cet état du jeu, la case de position p sur la grille est occupée par un pion ;

— un prédicat binaire pour l'égalité noté de manière infixe. On pourra utiliser la notation $t \neq u$ comme abréviation de la formule $\neg(t = u)$.

1. Traduire les formules suivantes en langue naturelle (français ou anglais)
   (a) $\forall x, \neg\mathtt{etat}(x) \Rightarrow \neg\mathtt{pos}(x) \Rightarrow (x = \mathtt{N}) \vee (x = \mathtt{S}) \vee (x = \mathtt{E}) \vee (x = \mathtt{O})$
   (b) $\forall p, \mathtt{suivant}(p, E) \neq \mathtt{null} \Rightarrow \mathtt{suivant}(\mathtt{suivant}(p, E), O) = p$
   (c) $\forall s, \mathtt{etat}(s) \Rightarrow \exists p, \mathtt{pion}(s, p)$

   Correction :

   (a) Tout objet qui n'est ni un état ni une position est égal à l'une des quatre directions $\mathtt{N}$*,* $\mathtt{S}$*,* $\mathtt{E}$ *ou* $\mathtt{O}$*.*

   (b) Si le déplacement de la position p *dans la direction* $\mathtt{E}$ *reste dans la grille, alors un nouveau déplacement dans la direction* $\mathtt{O}$ *revient à la position initiale*

   (c) Dans tout état, il existe une position occupée par un pion.

2. Traduire les propriétés suivantes en des formules de logique sur la signature donnée
   (a) Le prédicat `pion` est vrai seulement si le premier argument est un état et le second une position qui n'est pas `null`.
   (b) La fonction `suivant` appliquée à une position et une direction renvoie une position et si le premier argument est la position `null` alors le résultat est également `null`.
   (c) Dans tout état du jeu, il existe une position telle que les déplacements dans les 4 directions autour de cette position restent dans la grille.

   Correction :

   (a) $\forall s\, p, \mathtt{pion}(s, p) \Rightarrow \mathtt{etat}(s) \wedge \mathtt{pos}(p) \wedge p \neq \mathtt{null}$

   (b) $\forall p\, d, (\mathtt{pos}(p) \Rightarrow \mathtt{pos}(\mathtt{suivant}(p, d))) \wedge \mathtt{suivant}(\mathtt{null}, d) = \mathtt{null}$

   (c) $\forall s, \mathtt{etat}(s) \Rightarrow \exists p, \mathtt{pos}(p) \wedge \forall d, (d = \mathtt{N} \vee d = \mathtt{S} \vee d = \mathtt{E} \vee d = \mathtt{O}) \Rightarrow \mathtt{suivant}(p, d) \neq \mathtt{null}$

3. Soient des variables s, t, p et d, dans cette question on va écrire des formules paramétrées par certaines de ces variables (les variables à utiliser sont précisées dans chaque question).

   On pourra introduire des abréviations pour des expressions (termes ou formules) qui apparaissent plusieurs fois. De même on peut utiliser dans une question les formules introduites dans les questions préalables.

   (a) Ecrire une formule logique $\mathtt{gagnant}(s)$ qui dépend de la variable s et qui exprime la propriété que l'état du jeu s est un état gagnant (un seul pion restant).
   (b) Ecrire une formule logique $\mathtt{coup\text{-}possible}(s, p, d)$ qui utilise les variables s, p et d et qui exprime le fait qu'il est possible dans l'état du jeu s de déplacer un pion de la position p dans la direction d en respectant les règles du jeu. On exprimera l'ensemble des contraintes :
   — le pion ne sort pas de la grille,
   — on va d'une position occupée par un pion à une position vide en passant au dessus d'une position occupée par un pion
   (c) Ecrire une formule logique $\mathtt{result}(s, t)$ qui utilise les variables s et t et qui exprime le fait que t est un état du jeu après avoir joué un coup dans l'état s, c'est-à-dire qu'il existe une position p et une direction d que l'on pouvait jouer et t est le résultat de cette étape de jeu. On précisera dans la formule les positions dans la grille qui ont changé et celles qui ne changent pas.

   Correction :

   (a) $\mathtt{gagnant}(s) \stackrel{\text{def}}{=} \exists p, \mathtt{pion}(s, p) \wedge \forall q, \mathtt{pion}(s, q) \Rightarrow p = q$

(b) *On utilise la notation* $\texttt{suivant2}(p,d)$ *pour représenter le terme* $\texttt{suivant}(\texttt{suivant}(p,d),d)$

$$\texttt{coup-possible}(s,p,d) \stackrel{\text{def}}{=} \texttt{suivant}(p,d) \neq \texttt{null} \wedge \texttt{suivant2}(p,d) \neq \texttt{null} \\ \wedge \texttt{pion}(s,p) \wedge \texttt{pion}(s,\texttt{suivant}(p,d)) \wedge \neg\texttt{pion}(s,\texttt{suivant2}(p,d))$$

(c) $$\texttt{result}(s,t) \stackrel{\text{def}}{=} \exists p, \exists d, \texttt{coup-possible}(s,p,d) \\ \wedge\neg\texttt{pion}(t,p) \wedge \neg\texttt{pion}(t,\texttt{suivant}(p,d)) \wedge \texttt{pion}(t,\texttt{suivant2}(p,d)) \\ \wedge\forall q, (q \neq p \wedge q \neq \texttt{suivant}(p,d) \wedge q \neq \texttt{suivant2}(p,d)) \Rightarrow (\texttt{pion}(s,q) \Leftrightarrow \texttt{pion}(t,q))$$

4. **LDD** Donner les axiomes de la théorie de l'égalité pour la signature.

Correction : *On a les axiomes d'équivalence et de congruence par rapport aux symboles de fonction (*$\texttt{suivant}$*) et de prédicat (*$\texttt{etat}$*,* $\texttt{pos}$ *et* $\texttt{pion}$*)*

— $\forall x, x = x$
— $\forall x\, y, x = y \Rightarrow y = x$
— $\forall x\, y\, z, x = y \wedge y = z \Rightarrow x = z$
— $\forall p\, q\, d\, e, p = q \wedge d = e \Rightarrow \texttt{suivant}(p,d) = \texttt{suivant}(q,e)$
— $\forall p\, q, p = q \Rightarrow \texttt{pos}(p) \Rightarrow \texttt{pos}(q)$
— $\forall s\, t, s = t \Rightarrow \texttt{etat}(s) \Rightarrow \texttt{etat}(t)$
— $\forall s\, t\, p\, q, s = t \wedge p = q \Rightarrow \texttt{pion}(s,p) \Rightarrow \texttt{pion}(t,q)$