

Examen de Logique 1 – HLIN402 – session 1

Michel Leclère

9 mai 2019

Durée : 2h. 1 feuille A4 manuscrite recto-verso autorisée. Pas de calculatrice.

Question 1 (5 points). *Syntaxe*

Soit la formule :

$$F = \neg((b \rightarrow \neg\neg a) \vee (c \rightarrow d)) \vee ((a \wedge \neg c) \wedge (b \vee \neg d \rightarrow b \wedge \neg d))$$

- a. Dessinez l'arborescence de F .
- b. Dites si F est valide, contingente ou insatisfiable en **justifiant votre réponse**.

Contingente

Donnez l'un des modèles $I_1(b) = I_1(c) = 1$ $I_1(a) = I_1(d) = 0$, ou $I_2(a) = I_2(d) = 1$ $I_2(b) = I_2(c) = 0$ ou $I_3(a) = I_3(b) = I_3(d) = 1$ $I_3(c) = 0$.

Et l'un contre-modèle pour l'une des 13 autres interprétations.

- c. Quelles sont les sous-formules de F dont le connecteur racine est une négation ?

$\neg((b \rightarrow a) \vee (c \rightarrow d))$, $\neg\neg a$, $\neg a$, $\neg c$ et $\neg d$

- d. Définissez par induction la fonction $ssFbfNeg$ qui associe à toute formule bien formée l'ensemble de ses sous-formules ayant une négation comme connecteur racine.

(base) si $F \in S \cup \{\top, \perp\}$, $ssFbfNeg(F) = \emptyset$

(cons) si $F = \neg G$, $ssFbfNeg(F) = \{F\} \cup ssFbfNeg(G)$

si $F = (G \text{ c } H)$ où c est un connecteur binaire, $ssFbfNeg(F) = ssFbfNeg(G) \cup ssFbfNeg(H)$

- e. Traduisez en Scheme $ssFbfNeg$ (cf. annexe pour rappel des fonctions utiles).

```
(define (ssFbfNeg f)
  (cond ((atomicFbf f) ())
        ((negFbf f) (set-union (list f) (ssFbfNeg (fils f))))
        (else (set-union (ssFbfNeg (filsG f)) (ssFbfNeg (filsD f))))))
```

Question 2 (3 points). *Questions de cours*

- a. Rappelez la définition de l'insatisfiabilité.

F insatisfiable ssi $\forall I, v(F, I) = 0$

- b. Rappelez la définition de la conséquence logique.

$\{H_1, H_2, \dots, H_n\} \models C$ ssi $\forall I$ tel que $\forall i \in 1..n, v(H_i, I) = 1$, on a $v(C, I) = 1$

- c. Comment peut-on reformuler un problème de conséquence logique $\{H_1, H_2, \dots, H_n\} \models C$ (où les H_i et C sont des formules bien formées) à l'aide de l'insatisfiabilité ?

$H_1 \wedge H_2 \wedge \dots \wedge H_n \wedge \neg C$ insatisfiable.

- d. Prouvez que si cette propriété d'insatisfiabilité est établie alors la conséquence logique l'est aussi. **Soignez la présentation de la preuve.**

Question 3 (3 points). *Modélisation*

Soit le règlement suivant d'un club écossais :

1. Tout membre non écossais porte des chaussettes orange ;
2. Tout membre porte une jupe ou ne porte pas de chaussettes orange ;
3. Les membres mariés ne sortent pas le dimanche ;
4. Un membre sort le dimanche si et seulement si il est écossais ;
5. Tout membre qui porte une jupe est écossais et marié ;
6. Tout membre écossais porte une jupe

On cherche à savoir s'il peut y avoir un membre dans ce club, c'est à dire si son règlement est consistant.

- a. Modélisez chacune des règles de ce club (après avoir précisé quel sens vous donnez à vos symboles propositionnels).

e = "être écossais"
 o = "porter des chaussettes oranges"
 j = "porter une jupe"
 m = "être marié"
 d = "sortir le dimanche"
 $R_1 = \neg e \rightarrow o$
 $R_2 = j \vee \neg o$ (ou éventuellement $(j \wedge o) \vee (\neg j \wedge \neg o)$)
 $R_3 = m \rightarrow \neg d$
 $R_4 = d \leftrightarrow e$
 $R_5 = j \rightarrow m \wedge e$
 $R_6 = e \rightarrow j$

- b. Précisez quel problème de logique des propositions permet de répondre à la question.

$\{R_1, R_2, R_3, R_4, R_5, R_6\}$ est inconsistant (ou la conjonction est insatisfiable).

- c. Résolvez ce problème à l'aide de la méthode de votre choix et répondez à la question.

$\{R_1, R_2, R_3, R_4, R_5, R_6\}$ est inconsistant (ou la conjonction est insatisfiable) et 0,5 point pour la justification par une méthode. Par exemple avec la méthode de résolution, on a :

$C_1 = e \vee o$
 $C_2 = j \vee \neg o$ (ou éventuellement $(j \wedge o) \vee (\neg j \wedge \neg o)$)
 $C_3 = \neg m \vee \neg d$
 $C_4 = \neg d \vee e$
 $C_5 = \neg e \vee d$
 $C_6 = \neg j \vee m$
 $C_7 = \neg j \vee e$
 $C_8 = \neg e \vee j$
 $C_9 = \text{Res}(C_1, C_2) = e \vee j$
 $C_{10} = \text{Res}(C_8, C_9) = j$
 $C_{11} = \text{Res}(C_6, C_{10}) = m$
 $C_{12} = \text{Res}(C_7, C_{10}) = e$
 $C_{13} = \text{Res}(C_3, C_{11}) = \neg d$
 $C_{14} = \text{Res}(C_5, C_{12}) = d$
 $C_{15} = \text{Res}(C_{13}, C_{14}) = \perp$

Question 4 (4 points). *Méthodes de preuve*

Montrez que la formule suivante est **valide** :

$$(((r \vee t) \rightarrow (r \vee s)) \rightarrow (r \vee (t \rightarrow s)))$$

- a. En utilisant la méthode de résolution.

on prend la négation de la formule, la mise sous forme clausale détaillée, et un arbre de résolution qui conduit à la clause vide.

- b. En utilisant le système LK des séquents de Gentzen (cf. règles en annexe).

séquent à prouver : $\vdash (((r \vee t) \rightarrow (r \vee s)) \rightarrow (r \vee (t \rightarrow s)))$ puis faire l'arbre de preuve.

Pour chacune de ces méthodes, vous détaillerez bien les étapes du raisonnement. **Soignez la présentation.**

Question 5 (5 points). Exercice de synthèse

On se propose de remplacer les deux règles d'inférence à gauche du système LK par une règle compilée en exploitant l'équivalence sémantique suivante : $(P \leftrightarrow Q) \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$.

- a. Prouver l'équivalence logique précédente.

table de vérité à 4 lignes dont les colonnes correspondant aux 2 formules sont égales.

- b. Développer "au maximum" l'arbre de preuve du séquent $\Gamma, (A \rightarrow B) \wedge (B \rightarrow A) \vdash \Delta$.

arbre de preuve dont les 2 feuilles non "fermées" sont les antécédents de la règle de superdédution ci-après.

- c. Prouver que l'utilisation de la règle de superdédution

$$\frac{\Gamma, A, B \vdash \Delta \quad \Gamma \vdash A, B, \Delta}{\Gamma, A \leftrightarrow B \vdash \Delta} \leftrightarrow_g$$

est adéquate vis-à-vis de la sémantique de la logique des propositions c'est-à-dire prouver que, soit $\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_n\}$ et $\Delta = \{\delta_1, \delta_2, \dots, \delta_m\}$, on a :

*Si $\gamma_1 \wedge \gamma_2 \wedge \dots \wedge \gamma_n \wedge (A \leftrightarrow B) \models \delta_1 \vee \delta_2 \vee \dots \vee \delta_m$ alors
d'une part, $\gamma_1 \wedge \gamma_2 \wedge \dots \wedge \gamma_n \wedge A \wedge B \models \delta_1 \vee \delta_2 \vee \dots \vee \delta_m$
et d'autre part, $\gamma_1 \wedge \gamma_2 \wedge \dots \wedge \gamma_n \models A \vee B \vee \delta_1 \vee \delta_2 \vee \dots \vee \delta_m$.*

A Annexe règles d'inférence du système LK

$\frac{}{\Gamma, P \vdash \Delta, P} ax$	$\frac{\Gamma, P \vdash \Delta \quad \Gamma, Q \vdash \Delta}{\Gamma, P \vee Q \vdash \Delta} \vee_g$
$\frac{}{\Gamma \vdash \Delta, \top} \top_d$	$\frac{\Gamma \vdash \Delta, P, Q}{\Gamma \vdash \Delta, P \vee Q} \vee_d$
$\frac{}{\Gamma, \perp \vdash \Delta} \perp_g$	$\frac{\Gamma \vdash \Delta, P \quad \Gamma, Q \vdash \Delta}{\Gamma, P \rightarrow Q \vdash \Delta} \rightarrow_g$
$\frac{\Gamma \vdash \Delta, P}{\Gamma, \neg P \vdash \Delta} \neg_g$	$\frac{\Gamma, P \vdash \Delta, Q}{\Gamma \vdash \Delta, P \rightarrow Q} \rightarrow_d$
$\frac{\Gamma, P \vdash \Delta}{\Gamma \vdash \Delta, \neg P} \neg_d$	$\frac{\Gamma \vdash \Delta, P \quad \Gamma, Q \vdash \Delta}{\Gamma, P \leftrightarrow Q \vdash \Delta} \leftrightarrow_{g1}$
$\frac{\Gamma, P, Q \vdash \Delta}{\Gamma, P \wedge Q \vdash \Delta} \wedge_g$	$\frac{\Gamma \vdash \Delta, Q \quad \Gamma, P \vdash \Delta}{\Gamma, P \leftrightarrow Q \vdash \Delta} \leftrightarrow_{g2}$
$\frac{\Gamma \vdash \Delta, P \quad \Gamma \vdash \Delta, Q}{\Gamma \vdash \Delta, P \wedge Q} \wedge_d$	$\frac{\Gamma, P \vdash \Delta, Q \quad \Gamma, Q \vdash \Delta, P}{\Gamma \vdash \Delta, P \leftrightarrow Q} \leftrightarrow_d$

B Annexe : fonctions utiles TP Scheme

```
(define (neg? f) (eq? f '!))

(define (and? f) (eq? f '^))

(define (or? f) (eq? f 'v))

(define (imp? f) (eq? f '->))

(define (equ? f) (eq? f '<->))

(define (top? f) (eq? f 'Top))

(define (bot? f) (eq? f 'Bot))

(define (symbLog? f) (or (top? f) (bot? f) (and? f) (or? f) (neg? f) (imp? f) (equ? f)))

(define (conBin? f) (or (and? f) (or? f) (imp? f) (equ? f)))

(define (symbProp? f) (and (symbol? f) (not (symbLog? f))))

(define (atomicFbf? f) (or (symbProp? f) (top? f) (bot? f)))

(define (fbf? f)
  (cond ((atomicFbf? f) #t )
        ((list? f) (cond ((and (= (length f) 2) (neg? (car f))) (fbf? (cadr f)))
                          ((and (= (length f) 3) (conBin? (car f))) (and (fbf? (cadr f)) (fbf? (caddr f))))
                          (else #f)))
        (else #f)))

(define (conRac f) (car f))

(define (fils f) (cadr f))

(define (filsG f) (cadr f))

(define (filsD f) (caddr f))

(define (negFbf? f) (and (not (atomicFbf? f)) (neg? (conRac f))))

(define (andFbf? f) (and (not (atomicFbf? f)) (and? (conRac f))))

(define (orFbf? f) (and (not (atomicFbf? f)) (or? (conRac f))))

(define (impFbf? f) (and (not (atomicFbf? f)) (imp? (conRac f))))

(define (equFbf? f) (and (not (atomicFbf? f)) (equ? (conRac f))))
```

Vous pouvez également utiliser les fonctions de manipulation d'ensemble utilisées dans le TP (set-union ens1 ens2), (set-intersect ens1 ens2), (set-subtract ens1 ens2), (set-add ens val), (set-remove ens val), (set=? ens1 ens2).