Sorbonne Université
ANNÉE 2019-2020

LRC - Examen réparti n° 1

M1 ANDROIDE
M1 DAC

Durée 2h - documents et calculatrices non autorisés

Le barème n'est donné qu'à titre indicatif

1 Logique classique

Nous considérons dans les exercices 1 et 5 les six propositions suivantes :

(a) "Les Ménines" est un tableau peint par Velázquez.

(b) Tous les peintres ont peint au moins un tableau.

(c) Les tableaux sont peints par un peintre et un seul.

(d) "Les Ménines" est une série de tableaux peints par Picasso.

(e) Picasso s'est inspiré de Velázquez, mais Picasso n'est pas Velázquez.

(f) Velázquez n'a peint aucun tableau de la série "Les Ménines"

Exercice 1 – Représentation en logique des prédicats du premier ordre – *3 points*

Traduire en logique des prédicats du premier ordre les six propositions *a, b, c, d, e, f* en faisant appel aux prédicats unaires $peintre(x)$ et $tableau(x)$ ainsi qu'aux prédicats binaires $a\_peint(x, y)$, $inspiré\_par(x, y)$, $dans\_série(x, y)$ et $eq(x, y)$.

Exercice 2 – Démonstration avec la règle de résolution – *3 points*

Soit $S = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11}, C_{12}\}$ l'ensemble des 12 clauses suivantes où x, y et z sont des variables universellement quantifiées et a, b, "$LesMénines''$, $Picasso$ et $Vélazquez$ des constantes :

$C_1 : \neg tableau(x) \vee \neg a\_peint(y, x) \vee \neg a\_peint(z, x) \vee eq(y, z)$

$C_2 : tableau(\text{“}LesMénines'')$

$C_3 : \neg tableau(x) \vee a\_peint(b, x)$

$C_4 : inspiré\_par(Picasso, Vélazquez)$

$C_5 : a\_peint(Velázquez, \text{“}LesMénines'')$

$C_6 : \neg dans\_série(x, \text{“}LesMénines'') \vee tableau(x)$

$C_7 : \neg peintre(x) \vee a\_peint(x, a)$

$C_8 : \neg dans\_série(x, \text{“}LesMénines'') \vee a\_peint(Picasso, x)$

$C_9 : \neg eq(Picasso, Velázquez)$

$C_{10} : \neg peintre(x) \vee tableau(a)$

$C_{11} : \neg tableau(x) \vee peintre(b)$

$C_{12} : \neg tableau(x) \vee \neg dans\_série(x, \text{“}LesMénines'') \vee \neg a\_peint(Vélazquez, x)$

Montrer en utilisant la règle de résolution que $C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11} \vdash C_{12}$.

Exercice 3 – Démonstration avec la méthode des tableaux – *3 points*

Montrer avec la méthode des tableaux (rappelée en annexe), en justifiant la réponse et en précisant à chaque étape la règle appliquée et la formule traitée, si la formule
$F = (A \to B) \to ((B \to C) \to (A \to C))$ est valide, satisfiable ou insatisfiable.

Remarque : A, B et C sont trois variables propositionnelles.

2 Prolog

Exercice 4 – Écriture d'un programme en PROLOG – *4 points*

1. Programmer la fonction `appartient(X, Y, Z)` qui instancie `X` avec un élément de la liste `Y` et `Z` avec la liste `Y` sans l'élément `X`.
   Exemple : l'appel à `appartient(X, ['a', 'b', 'c'], Z)` aura trois solutions :
   1) `X = 'a'; Z = ['b', 'c']`
   2) `X = 'b'; Z = ['a', 'c']`
   3) `X = 'c'; Z = ['a', 'b']`
2. Écrire en Prolog le prédicat `permutation(X, Y)` qui instancie `Y` avec toutes les permutations de la liste `X`.
   Remarque : la solution peut faire appel à la fonction `appartient` de la question précédente.
   Exemple : l'appel à `permutation(['a', 'b', 'c'], Y)` aura 6 solutions :
   1) `Y = ['a', 'b', 'c']`
   2) `Y = ['a', 'c', 'b']`
   3) `Y = ['b', 'a', 'c']`
   4) `Y = ['b', 'c', 'a']`
   5) `Y = ['c', 'a', 'b']`
   6) `Y = ['c', 'b', 'a']`

3 Logique de description

Exercice 5 – Représentation en logique de description – *3 points*

Représenter les trois propositions a, b, c décrite dans la partie 1 en logique de description $\mathcal{ALCIQ}$, en distinguant explicitement la TBox et la ABox et en utilisant les concepts $Tableau$ et $Peintre$ ainsi que le rôle a_peint.

Remarque : on rappelle que $\mathcal{I}$ signifie que l'on peut avoir des inversions de rôle et que $\mathcal{Q}$ indique que l'on peut avoir des restrictions qualifiées sur les cardinalités, par exemple $\exists^{>=3} a\_enfant.Femme$ qui désigne les personnes qui ont plus de trois filles.

4 Logique modale

Exercice 6 *1 point*

On sait que dans la logique K, on a $\neg\Box\neg\phi \equiv \Diamond\phi$, où ϕ est une formule quelconque.
Mais est-il vrai que $\neg\Box\Box\neg\phi \equiv \Diamond\Diamond\phi$? Justifiez votre réponse.

Exercice 7 — *3 points*

On considère la logique S4, définie par les axiomes (T) $\Box\phi \rightarrow \phi$, et (4) $\Box\phi \rightarrow \Box\Box\phi$. Les propriétés correspondantes dans les structures de Kripke sont la réflexivité et la transitivité.

1. Montrer que $\Diamond\Box\phi \rightarrow \phi$ n'est pas valide dans cette logique.
2. Montrer que $\Diamond\Diamond\phi \rightarrow \Diamond\phi$ est valide dans cette logique.

5 Annexe

5.1 Méthode des tableaux sémantiques pour la logique des propositions

La méthode des tableaux sémantiques permet d'établir si un ensemble de fomules logiques est valide, satisfiable ou insatisfiable.

5.1.1 Composantes

La méthode des tableaux est basée sur des règles syntaxiques de décomposition, qui distinguent deux types de formules, nommés α et β.

Nom	Formule α	α_1	α_2
$R_{\neg\neg}$	$\neg\neg\varphi$	φ	φ
$R_{\wedge}$	$\varphi_1 \wedge \varphi_2$	φ_1	φ_2
$R_{\neg\vee}$	$\neg(\varphi_1 \vee \varphi_2)$	$\neg\varphi_1$	$\neg\varphi_2$
$R_{\neg\rightarrow}$	$\neg(\varphi_1 \rightarrow \varphi_2)$	φ_1	$\neg\varphi_2$
$R_{\leftrightarrow}$	$\varphi_1 \leftrightarrow \varphi_2$	$\varphi_1 \rightarrow \varphi_2$	$\varphi_2 \rightarrow \varphi_1$

Nom	Formule β	β_1	β_2
$R_{\vee}$	$\varphi_1 \vee \varphi_2$	φ_1	φ_2
$R_{\neg\wedge}$	$\neg(\varphi_1 \wedge \varphi_2)$	$\neg\varphi_1$	$\neg\varphi_2$
$R_{\rightarrow}$	$\varphi_1 \rightarrow \varphi_2$	$\neg\varphi_1$	φ_2
$R_{\neg\leftrightarrow}$	$\neg(\varphi_1 \leftrightarrow \varphi_2)$	$\neg(\varphi_1 \rightarrow \varphi_2)$	$\neg(\varphi_2 \rightarrow \varphi_1)$

5.1.2 Satisfiabilité

La recherche d'un modèle pour un ensemble de formules $\mathcal{F}$ par la méthode des tableaux peut être représentée de différentes façons, nous utilisons ici une forme arborescente.

- Initialisation : créer un nœud racine, étiqueté par l'ensemble $\mathcal{F}$ et marqué comme non traité
- Décomposition itérative : choisir un nœud non traité et le marquer comme traité
  - — si l'étiquette du nœud contient deux littéraux complémentaires, marquer le nœud comme fermé
  - — sinon, si toutes les formules associées au nœud sont des variables propositionnelles, marquer le nœud comme ouvert
  - — sinon, choisir une formule F de l'étiquette du nœud
    - — si elle est de type α
      - — créer un sous-nœud marqué comme non traité
      - — lui associer l'étiquette $\mathcal{F} \setminus \{F\} \cup \{\alpha_1, \alpha_2\}$ où α_1 et α_2 sont les formules obtenues par réécriture de F
    - — sinon (si elle est de type β)
      - — créer deux sous-nœuds marqués comme non traités
      - — leur associer respectivement les étiquettes $\mathcal{F} \setminus \{F\} \cup \{\beta_1\}$ et $\mathcal{F} \setminus \{F\} \cup \{\beta_2\}$ où β_1 et β_2 sont les formules obtenues par réécriture de F

Si l'arbre contient une feuille ouverte, alors $\mathcal{F}$ est satisfiable.
Si toutes les feuilles de l'arbre sont fermées, alors $\mathcal{F}$ est insatisfiable.