Examen de programmation logique et fonctionnelle

Licence informatique

Université Bourgogne Europe - UFR Sciences et Techniques - année 2024/2025

Documents autorisés : trois feuilles A4 recto-verso avec contenu libre.

Ordinateurs, calculatrices et appareils connectés interdits.

Logique propositionnelle

Q1 (3 points)

Soient les deux formules suivantes :

- $\Sigma : a \rightarrow (a \wedge b)$
- $\Omega : (a \vee b) \rightarrow a$

Répondez par oui ou par non aux questions suivantes, sans donner de justification (0 si au moins trois réponses fausses, sinon -1 point par réponse fausse) :

1. Σ est-elle cohérente ?
2. Σ est-elle une tautologie ?
3. Ω est-elle cohérente ?
4. Ω est-elle une tautologie ?
5. Σ est-elle conséquence logique de Ω ?
6. Ω est-elle conséquence logique de Σ ?

Q2 (1 point)

Donnez tous les modèles de la formule suivante :

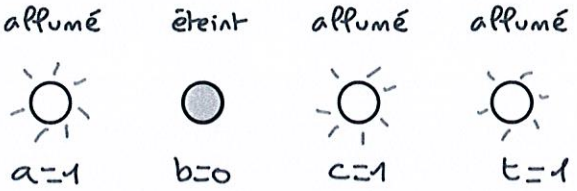
$a \wedge (\neg a \vee b) \wedge (\neg b \vee \neg a \vee c) \wedge (\neg b \vee d) \wedge (\neg d \vee \neg b \vee \neg e)$

Aucune justification n'est demandée.

Q3 (2 points)

Soient 3 variables propositionnelles a, b, c représentant les états respectifs de trois lampes blanches distinctes, et une variable t représentant l'état d'une lampe rouge. Une variable a la valeur 1 si et seulement si la lampe associée est allumée.

Voici un exemple de situation avec les valeurs des variables



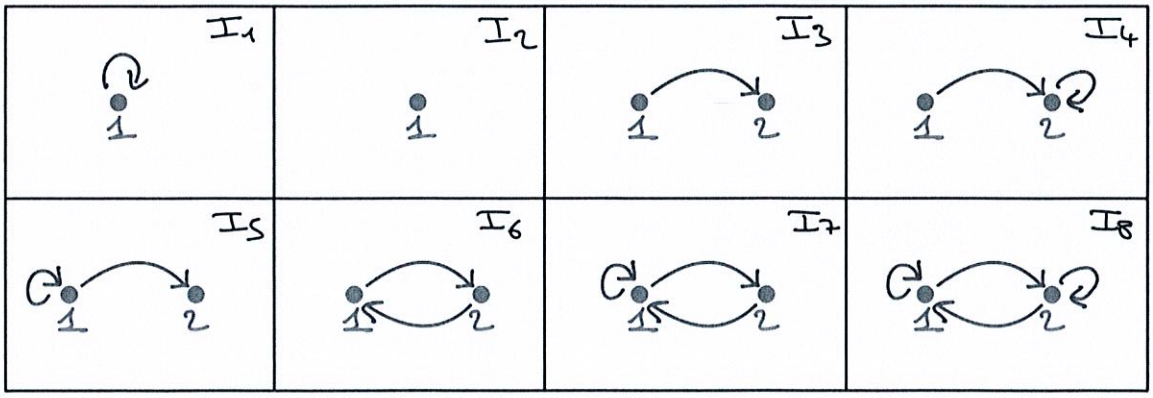
Donnez une formule modélisant la proposition "la lampe rouge est allumée si et seulement si au moins une des lampes blanches est allumée"

Logique du premier ordre

Q4 (2 points)

Soit la formule $\Sigma = \forall X \forall Y\, p(X, Y) \rightarrow p(Y, X)$.

Parmi les interprétations suivantes, lesquelles sont des modèles de Σ ? Aucune justification n'est demandée. (0 si au moins quatre réponses fausses, sinon -0,5 point par réponse fausse) :



Q5 (2 points)

Soit la formule $\Omega = \forall X \forall Y [p(X, Y) \rightarrow \exists Z\, p(Z, X)]$.

Donnez 4 contre-modèles de Ω avec des domaines d'interprétation de cardinalité 2. Présentez vos solutions sous la forme de graphes, comme dans l'exercice précédent. Aucune justification n'est demandée.

Q6 (2 points)

Donnez une formule du premier ordre modélisant l'énoncé suivant :

> Toute personne active connait au moins une personne active.

Sachant que `active(P)` est vrai si et seulement si `P` est une personne active et que `connait(X,Y)` est vrai si et seulement si `X` connait `Y`.

PLOLOG

Q7 (2 points)

On se place dans le contexte de cellules qui se reproduisent par division. Lors de ce processus, une cellule appelée *progénitrice* se divise en deux *cellules filles*. Toutes des cellules issues d'une même cellule C par une ou plusieurs divisions successives sont appelées *descendantes* de C. Les descendantes de C sont donc ses filles, les filles de ses filles, etc.

On suppose que le prédicat `fille/2`, tel que `fille(X,Y)` est vrai si et seulement si `Y` est une fille de `X`, est déjà défini sous la forme de faits.

1. Définissez le prédicat `descendante/2` tel que `descendante(X,Y)` est vrai si et seulement si `Y` est une descendante de `X`.
2. Définissez le prédicat `morte/1` tel que `morte(X)` est vrai si et seulement si `X` est une cellule ayant au moins une fille (donc n'existe plus, car la reproduction fait disparaitre la progénitrice).

Q8 (1 points)

On dispose d'un prédicat `concat/3` qui permet de concaténer deux listes et qui est défini de la manière suivante :

```
concat([],B,B).
concat([T|R],B,[T|Q]) :- concat(R,B,Q).
```

Définissez le prédicat `split/3` tel que si `L` est une liste, alors le but `split(L,P,E)` fait remonter dans `E` le dernier élément de `L` et dans `P` la liste de tous les éléments de `L` sauf le dernier. Par exemple, le but `split([1,2,3],P,E)` doit avoir pour résultat `P=[1,2]`, `E=3`.

Le prédicat `split` doit être défini en une seule clause non récursive, en utilisant le prédicat `concat`.

Q9 (2 points)

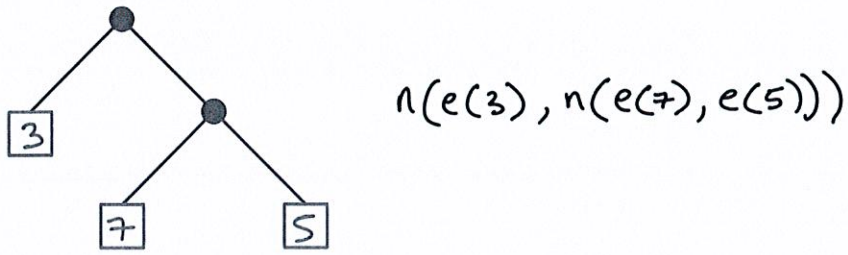
On suppose que le prédicat `split` de l'exercice précédent est disponible et utilisable. Ce prédicat permet de décomposer une liste en son dernier élément et la liste de tous les autres éléments, appelée préfixe. Il permet aussi, en changeant les slots d'entrée et de sortie, de reconstituer une liste à partir d'un préfixe et d'un dernier élément connus.

Vous devez définir un prédicat `ech/2` tel que si `L` est une liste ayant au moins deux éléments, alors `ech(L,S)` fait remonter dans `S` la liste obtenue en échangeant le premier et le dernier éléments de `L` sans modifier le reste de la liste. Par exemple, le but `ech([1,2,3,4],S)` a pour résultat `S = [4,2,3,1]`.

Vous devez définir ce prédicat de manière non récursive, en utilisant le prédicat `split`. (Deux appels à `split` sont nécessaires.)

Q10 (1 point)

On s'intéresse aux arbres binaires à nœuds non étiquetés et à feuilles étiquetées par des entiers. Ces arbres sont représentés par les termes fonctionnels construits avec les symboles `e/1` et `n/2`. Voici un exemple d'arbre et de sa représentation par un terme fonctionnel.



Définissez le prédicat `val/2` tel que si `T` est un arbre tel que décrit plus haut, alors le but `val(T,X)` fait remonter dans `X` toutes les valeurs des feuilles de `T`. Par exemple, le but ...

```
val(n(e(3),n(e(7),e(5))), X).
```

... a pour résultats `X=3`, `X=7`, `X=5`.

Q11 (2 points)

On reprend les arbres de l'exercice précédent. On représente un chemin partant de la racine par une liste de symboles `g` et `d`, où `g` signifie "aller à gauche" et `d` "aller à droite". Un chemin est dit complet s'il aboutit à une feuille. Par exemple, dans l'arbre `n(e(3),n(e(7),e(5)))` donné en exemple dans l'exercice précédent, il y a trois chemins complets, à savoir `[g]`, `[d,g]` et `[d,d]`, qui se terminent respectivement par les feuilles de valeurs 3, 7 et 5.

Définissez le prédicat `chem/3` tel que si `T` représente un arbre tel que décrit dans l'exercice précédent, alors `chem(T,C,X)` fait remonter dans `C` un chemin complet de `T` et dans `X` la valeur de la feuille à laquelle ce chemin aboutit. Par exemple le but ...

```
chem(n(e(3),n(e(7),e(5))), C, V).
```

... a pour résultats :

- `C = [g], V = 3`
- `C = [d,g], V = 7`
- `C = [d,d], V = 5`