le cnam

Année universitaire 2017-2018

Sujet UE NFP108: Spécification et Modélisation Informatiques

Examen première session : 6/02/2018

Responsable : P. Courtieu – O. Pons

Durée : 3 heures

Consignes

Tous les documents sont autorisés.
Calculatrice non autorisée

Les téléphones mobiles et autres équipements communicants (exemple : PC, tablettes, etc) doivent être éteints et rangés dans les sacs pendant toute la durée de l'épreuve.

Sujet de 6 pages, celle-ci comprise.

→ *Vérifiez que vous disposez de la totalité des pages du sujet en début d'épreuve et signalez tout problème de reprographie le cas échéant.*

A B A=>B
1 1 1
1 0 0
0 1 1
0 0 1

1 Logique des propositions

Exercice 1

T: travailler E:examen réussi D:Déçu P:Pleure

Mon super coach m'explique
- Si j'ai travaillé assez alors je réussi l'examen. T => E
- Si je réussi l'examen alors je ne suis pas déçu et je pleure E => (not D /\ P)
- Et,...je suis déçu ! D

Mon coach en conclu que je ne pleure pas.
Moi je conclu que je n'ai pas assez travaillé.

Affirmation du coach: (T=>E) , (E=>(notD/\P)) , D |- not P
(conséquence logique?)

Question 1.1
Modélisez ces faits en logique des propositions.

Question 1.2
Montrez par un **contre-exemple** que le raisonnement du coach est faux. (*i.e.* que son affirmation n'est pas une conséquence des autres faits).

0 = 1 => 1 > 1

A=>B === nonA \/ B

contre-exemple: D=Vrai, E=Faux, T=Faux, P = Vrai

Question 1.3
Montrer (en DN ou par une table de vérité) que mon raisonnement est juste.

Déduction Naturelle en Logique des propositions

Exercice 2

Question 2.1
Montrez en Déduction Naturelle :

propositionnel: A/\B => C

prédicats B(x) => B(y,z), forall x y, A(x) => A(x,y)

1. $((A \wedge B) \wedge (C \wedge D)) \Rightarrow (C \vee \neg A)$
2. $(B \vee \neg B) \Rightarrow (A \Rightarrow ((\neg B => \neg A) \Rightarrow B))$
3. $(B \vee \neg B) \Rightarrow (A \Rightarrow ((\neg B => \neg A) \Rightarrow B))$
4. $A \Rightarrow (((B \Rightarrow C) \Rightarrow D) \Rightarrow (C \Rightarrow (D \wedge A)))$

Logique des prédicats

Exercice 3

Prédicats: Pere(x,y), Mere(x,y), Frere(x,y), Parent(x,y), Dieu(x)
Individus: Poséidon, Zeus, Rhéa

- si une personne est le père d'une autre c'est un parent de cette autre personne forall x y, Pere(x,y) => Parent(x,y)
- si une personne est le mère d'une autre c'est un parent de cette autre personne forall x y, Mere(x,y) => Parent(x,y)
- si deux personnes sont des dieux et ont un même parent elles sont frères. ∀ x y, (Dieu(x) /\ Dieu(y) /\ (∃ z, Parent(z,x) /\ Parent(z,y))) => Frere(x,y)
- Poséidon est un dieu et Zeus est un dieux. Dieu(Poseidon) /\ Dieu(Zeus)
- Rhéa est la mère de Poséidon et de Zeus.

Mere(Rhéa,Poseidon)/\Mere(Rhéa,Zeus)

Question 3.1
Modélisez ces informations en **logique des prédicats**

Question 3.2
Montrez en Déduction Naturelle que Poséidon et Zeus sont frères Frere(zoseidon,zeus)

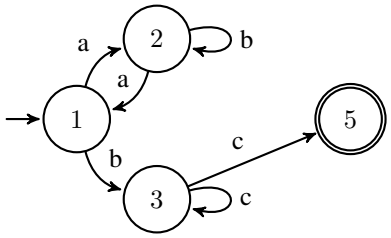
Sémantique

Soit la formule F suivante : $\forall x.\forall y.Q(x,y) \Rightarrow Q(y,x)$.
En interprétant Q par autre chose que l'égalité.

— Donnez une interprétation dans l'ensemble $String$« des chaînes de caractères » qui soit un modèle de F (ie qui la rende valide) et une qui ne soit pas un modèle.

Automates

Exercice 4 Question de cours



A = ({1,2,3,5} , {a,b,c}, 1, {5}, delta)
delta = { (1,a,2),(1,b,3), (2,b,2), (2,a,1), (3,c,3), (3,c,5) }

pas déterministe, l'état 3 a 2 flèche sortantes étiquetées c

langage infini car boucles

langage régulier par définition

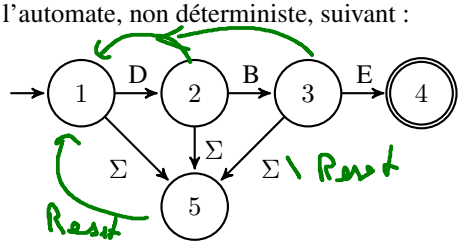
Question 4.1
Pour l'automate de la figure ci-dessus :

ababc bc abbbaabbbabc (ab*a)* bc*c

— Donnez le 5-uplet notant formellement cet automate.
— Cet automate est-il déterministe ? Justifiez votre réponse.
— Le langage de cet automate est-il fini ? Justifiez votre réponse.
— Le langage de cet automate est-il régulier ? Justifiez votre réponse.
— Donnez deux chaînes appartenant au langage de cet automate.
— Donnez une expression régulière définissant le même langage que cet automate.

Exercice 5 digicode

On veut spécifier le fonctionnement d'un digicode à 6 lettres, $\Sigma = \{A, B, C, D, E, F\}$ Reset}
Dans un premier temps on définit l'automate, non déterministe, suivant :



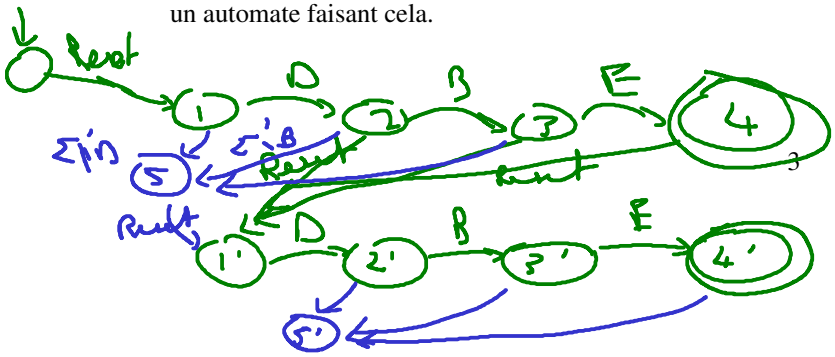
Question 5.1
Quel est le code correct du digicode ? DBE

Question 5.2
Quelle modification faut-il apporter pour que l'automate soit déterministe ? retirer l'a lettre ambiguë dans les sigma.

Question 5.3
Que se passe-t-il si on tape un code erroné ? on va dans l'état poubelle

Question 5.4
Proposez un automate dans lequel une touche RESET permet de remettre à zéro la frappe du code.

Question 5.5
On désire maintenant que le digicode permette exactement deux essais. Autrement dit si l'utilisateur effectue deux essais erronés (avec une frappe sur RESET avant chaque essai) alors le digicode se bloque définitivement. Proposez un automate faisant cela.



Annotation de programme

Rappel : `requires` spécifie une pré-condition, `ensures` spécifie une post-condition. Ces deux annotations sont des formules logiques sur les variables du programme. Par exemple : $\forall 12 < i < t.length, t[i] > 23$.

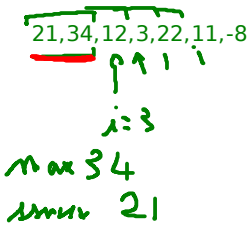
Exercice 6 Deuxième plus grande valeur d'un tableau

Le but de la fonction ci-dessous est de retourner la deuxième plus grande valeur du tableau. C'est-à-dire une valeur apparaissant dans le tableau, inférieure au maximum et supérieure à toutes les autres. Dans cette version on suppose que toutes les cases du tableau sont différentes.

L'algorithme est le suivant : on tient à jour 2 variables : `max` qui contient la valeur candidate pour le maximum du tableau, et `sousmax` qui contient la valeur candidate pour la deuxième valeur du tableau.

```
// requires:
// ensures:


int trouveDeuxième (int t[]){
  int i;
  int max;
  int sousmax;
  // initialisation de max et sousmax
  if (t[0] > t[1]) {
    max = t[0];
    sousmax = t[1];
  } else {
    max = t[1];
    sousmax = t[0];
  }
  i = 2; // On démarre à partir de la première case non vue
  while( i < t.length){
    // variant: (t.length - i)
    // invariant: max : valeur la plus élevée à gauche de i, sousmax: 2e max à gauche de i.
              forall j,  0<= j < i => t[j]<=max  exist k, 0<= k < i /\ t[k]=sousmax
              exist k, 0<= k < i /\ t[k]=max     sousmax < max
    if (t[i] > max){ // nouveau maximum          not (exist k', (0<= k' < i /\  max>t[k']>sousmax))
      sousmax = max; // on déclasse l'ancien max en sousmax
      max = t[i]; // On met à jour le max
    } else {
      if (t[i] > sousmax) { // t[i] nouveau sousmax
        sousmax = t[i];
      }
    }
    i = i + 1;
  }
}
```



Question 6.1

Spécifier SUR l'ÉNONCÉ la fonction en précisant la pré-condition (`requires`) et post-condition (`ensures`).

Notamment la pré-condition devra préciser qu'il n'y a pas de répétition dans le tableau et qu'il existe bien une deuxième valeur dans le tableau.

Question 6.2

Proposez SUR l'ÉNONCÉ un variant pour la boucle **`while`**.

Question 6.3

Proposez SUR l'ÉNONCÉ un invariant de la boucle **`while`** suffisant pour impliquer le bon fonctionnement de cette fonction.

Appendice : Règles de la déduction naturelle

$FV(\Phi)$ designe l'ensemble des variables libres dans la formule Φ

Axiomes

$$\frac{}{\Gamma, \phi \vdash \phi}\ Ax$$

Règles d'introduction

$$\frac{\Gamma \vdash \phi \ \ \Gamma \vdash \psi}{\Gamma \vdash \phi \wedge \psi}\ \wedge_i$$

$$\frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \Rightarrow \psi}\ \Rightarrow_i$$

$$\frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \vee \psi}\ \vee_{i1} \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \vee \psi}\ \vee_{i2}$$

$$\frac{\Gamma \vdash \phi \ \ \Gamma \vdash \neg\phi}{\Gamma \vdash \bot}\ \bot_i$$

$$\frac{\Gamma \vdash \phi \ \ x \notin FV(\Gamma)}{\Gamma \vdash \forall x \phi}\ \forall_i$$

$$\frac{\Gamma \vdash \phi[x := t]}{\Gamma \vdash \exists x \phi}\ \exists_i$$

Règles d'élimination

$$\frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \phi}\ \wedge_{e1} \qquad \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \psi}\ \wedge_{e2}$$

$$\frac{\Gamma \vdash \phi \Rightarrow \psi \ \ \Gamma \vdash \phi}{\Gamma \vdash \psi}\ \Rightarrow_e$$

$$\frac{\Gamma \vdash \phi \vee \psi \ \ \Gamma, \phi \vdash \theta \ \ \Gamma, \psi \vdash \theta}{\Gamma \vdash \theta}\ \vee_{e1}$$

$$\frac{\Gamma, \neg\phi \vdash \bot}{\Gamma \vdash \phi}\ \bot_e$$

$$\frac{\Gamma \vdash \forall x \phi}{\Gamma \vdash \phi[x := t]}\ \forall_e$$

$$\frac{\Gamma \vdash \exists x \phi \quad \Gamma, \phi \vdash \psi \ \ x \notin (FV(\Gamma) \cup FV(\phi))}{\Gamma \vdash \psi}\ \exists_e$$

$\neg\phi$ est une abréviation pour $\phi \Rightarrow \bot$. on donne donc aussi les règles suivantes :

$$\frac{\Gamma, \phi \vdash \bot}{\Gamma \vdash \neg\phi}\ \neg_i$$

$$\frac{\Gamma \vdash \phi \ \ \Gamma \vdash \neg\phi}{\Gamma \vdash \psi}\ \neg_e$$