

# Systemes de transitions - Modelisation TLA<sup>+</sup>

Durée 1h45 - Documents autorisés

31 mars 2017

## 1 Questions de cours (2 points)

Soit quatre variables  $x, y, S, f$ .  $x$  et  $y$  sont des entiers,  $S$  est un ensemble d'entiers et  $f \in [S \rightarrow \text{Nat}]$ .

Répondre aux questions suivantes en respectant la syntaxe TLA<sup>+</sup>.

1. Donner une expression qui représente le sous-ensemble des éléments de  $S$  qui sont plus grands que  $x$  et plus petits que  $y$ .
2. Donner une action qui ajoute  $x$  à  $S$  à condition que  $y$  soit déjà dans  $S$ .
3. Donner une propriété temporelle qui exprime que  $S$  est toujours un ensemble d'entiers.
4. Donner une propriété temporelle qui exprime qu'une valeur de la fonction  $f$  sera un jour supérieure à  $x$ .

## 2 Exercice (5 points)

Soit le module TLA<sup>+</sup> ci-dessous définissant le système de transitions **Spec**.

```
MODULE examen15b_test
EXTENDS Naturals, TLC
VARIABLES x, y

Init ≜ x ∈ {0, 1} ∧ y = x

action1 ≜ x = y ∧ x' = 0 ∧ y' = 1
action2 ≜ x = y ∧ x' = 1 ∧ y' = 0
action3 ≜ x + y = 1 ∧ x' = y ∧ UNCHANGED y
action4 ≜ x ≠ y ∧ x' = 0 ∧ y' = 2

Next ≜ action1 ∨ action2 ∨ action3 ∨ action4
Fairness ≜ WF<x, y>(action1) ∧ SF<x, y>(action4)
Spec ≜ Init ∧ □[Next]<x, y> ∧ Fairness
```

1. Donner le graphe d'exécution correspondant.
2. Les propriétés suivantes, exprimées en logique LTL ou CTL, sont-elles vérifiées (donner une justification informelle)?

(a)  $\Box(x + y \leq 2)$

(b)  $\Box\Diamond(x = 1)$

(c)  $\Diamond\Box(y = 2)$

(d)  $\exists\Box(x = 0)$

(e)  $\forall\Diamond(y = 2)$

(f)  $\forall\Box(x \neq y \Rightarrow \forall\Diamond x = y)$

### 3 Problème : algorithme de sélection du maximum (13 points <sup>1</sup>)

On souhaite modéliser et vérifier un algorithme qui détermine la valeur maximale possédée par  $N$  sites. Initialement, chaque site possède une valeur quelconque, et l'algorithme finit par atteindre un état où tous les sites sont d'accord sur la valeur maximale. Note : les sites n'ont pas nécessairement une valeur distincte, et plusieurs sites peuvent posséder la valeur maximale.

Nous allons étudier plusieurs descriptions de l'algorithme, de plus en plus déterministes.

Un squelette de module TLA<sup>+</sup> `maximum1.tla` est fourni en 3.6.

#### 3.1 `maximum1.tla` Module simple non déterministe

1. Compléter le prédicat `Init` définissant les états initiaux possibles, de la façon la plus générale. Les valeurs initiales possédées par les sites devront être quelconques dans  $0..V - 1$ .
2. Donner le prédicat de transition `Next`, spécifiant toutes les transitions possibles du problème modélisé.
3. Préciser la ou les propriétés d'équité minimale nécessaires pour atteindre une solution.

#### 3.2 Spécification

Exprimer en TLA<sup>+</sup> les propriétés suivantes (qui ne sont pas nécessairement vérifiées par le modèle) :

4. `ValinitEstConstant` : le tableau `valinit` est constant (son contenu ne change pas).
5. `ValeursParmiValeursInitiales` : les valeurs actuelles connues sont un sous ensemble des valeurs initiales.
6. `ValeurEstCroissant` : pour chaque site, sa valeur courante est croissante.
7. `NonNul` : un site qui démarre avec une valeur nulle prendra une valeur non nulle.
8. `ÇaTermine` : on atteint un point où il n'y a plus qu'une seule valeur commune à tous les sites.
9. `ÇaTermineAvecMax` : on atteint un point où il n'y a plus qu'une seule valeur commune à tous les sites, et cette valeur est la valeur maximale des valeurs initiales.

#### 3.3 `maximum2.tla` : Choix non arbitraire

Au lieu de se comparer avec n'importe quel autre site, un site ne se compare qu'avec son prédécesseur (i.e. le site  $i$  se compare avec  $prec(i)$ ).

10. Modifier le modèle TLA<sup>+</sup> fourni pour cela.
11. Donner la propriété à vérifier pour montrer que `maximum2` est un raffinement de `maximum1`.

#### 3.4 Vérification

12. Dessiner le graphe de transitions pour `maximum2` en supposant qu'il y a 4 sites qui ont pour valeurs initiales respectives 0/0/3/1 (9 états).

---

1. Toutes les questions valent autant.

13. Comment vérifie-t-on la propriété `ValeursParmiValeursInitiales` en examinant le graphe des transitions ?
14. Comment vérifie-t-on la propriété `ÇaTermine` en examinant le graphe des transitions ?
15. La propriété `ÇaTermineAvecMax` est-elle toujours vérifiée par ce modèle, quelque soit les valeurs initiales des sites ?

### 3.5 maximum3.tla : Choix ordonné

La version 2 est plus déterministe que la version 1 mais n'évite pas des transitions superflues où un site se compare à multiples reprises avec le site précédent. Nous allons imposer un ordre obtenu par un jeton circulant. Seul le site qui a le jeton peut se comparer avec son prédécesseur. On montrera qu'il suffit de faire 2 tours pour garantir que tous les sites possèdent le maximum global. Noter que le jeton ne transporte pas de valeur, seulement l'autorisation de se comparer.

On ajoute une variable `jeton` qui contient le site où est le jeton, et une variable `tour` dans 0..2. Initialement, le jeton est sur le site 0 et `tour` vaut 0.

16. Modifier l'action `changer(i)` pour que la mise à jour de `valeur` n'ait lieu que si le site a le jeton et que celui-ci a fait moins de 2 tours. Cette action `changer` doit aussi transmettre le jeton au site suivant et incrémenter si nécessaire `tour`.
17. Ajouter une action `passer(i)` pour transmettre aussi le jeton (et incrémenter `tour` si nécessaire) quand la comparaison des valeurs échoue (= quand le site a déjà une valeur plus grande que son prédécesseur).
18. Donner une propriété exprimant que si `tour` vaut 2, alors tous les sites ont la même valeur, laquelle est le maximum des valeurs initiales.
19. Donner une propriété spécifiant que `tour` est croissant.
20. Montrer qu'il suffit effectivement de 2 tours pour parvenir à la solution.

### 3.6 Module fourni : maximum1.tla

---

MODULE *maximum1*

---

Calcule le maximum des valeurs situés sur les sites, en comparant sa *valeur* avec la *valeur* d'un autre site (quelconque)

EXTENDS *Naturals*, *FiniteSets*

CONSTANTS  $N$ , nbre de sites  
 $V$  valeur maximale possédée

$Site \triangleq 0 \dots N - 1$  les sites  
 $Valeur \triangleq 0 \dots V - 1$  les valeurs possédées

VARIABLES

$valinit$ , valeur initialement possédée par chaque site (constante)  
 $valeur$  valeur maximale actuellement connue par le site

$vars \triangleq \langle valinit, valeur \rangle$

$max(S) \triangleq \text{CHOOSE } x \in S : \forall y \in S : y \leq x$  maximum d'un ensemble  
 $range(f) \triangleq \{f[x] : x \in \text{DOMAIN } f\}$  codomaine d'une fonction  
 $prec(i) \triangleq (i + N - 1) \% N$  site précédent

---

$TypeInvariant \triangleq \square (\wedge valinit \in [Site \rightarrow Valeur]$   
 $\wedge valeur \in [Site \rightarrow Valeur])$

---

$Init \triangleq \wedge \dots$  INITIALISATION  $valinit$  À COMPLÉTER  
 $\wedge valeur = valinit$

$changer(i, j) \triangleq$  le site  $i$  récupère la  $valeur$  du site  $j$  si elle est plus grande  
 $\wedge valeur[i] < valeur[j]$   
 $\wedge valeur' = [valeur \text{ EXCEPT } ![i] = valeur[j]]$   
 $\wedge \text{UNCHANGED } valinit$

$Next \triangleq \dots$  À DÉFINIR

$Fairness \triangleq \dots$  À DÉFINIR

$Spec \triangleq \wedge Init$   
 $\wedge \square [Next]_{vars}$   
 $\wedge Fairness$

---