Systèmes de transitions - Modélisation TLA$^+$

Durée 1h30 - Documents autorisés

26 mars 2021

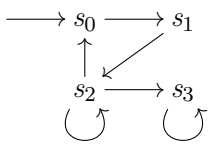
1 Questions de cours (4 points)

Soit trois variables x, y et z, qui contiennent des ensembles d'entiers.

1. Donner une action qui échange le contenu des variables x et y.
   $x' = y \land y' = x$ ($\land$ UNCHANGED z)
2. Donner un prédicat qui dit que y et z forment une partition de x (pas d'éléments en commun et l'union forme x).
   $x = y \cup z \land y \cap z = \emptyset$
3. Donner une propriété temporelle qui dit que x et y ne sont jamais égaux.
   $\Box(x \neq y)$
4. Donner une expression correspondant à une fonction définie sur x vers BOOLEAN, et qui vaut TRUE au point où la valeur est dans y ou z, et FALSE ailleurs.
   $[i \in x \mapsto i \in (y \cup z)]$

2 Exercice (4 points)

Soit le système de transition S :



Indiquer si les propriétés suivantes, exprimées en logique LTL ou CTL, sont vérifiées. Justifier les réponses (argumentaire ou contre-exemple).

	sans équité	$WF(s_2, s_3)$	$SF(s_2, s_3)$
$\Diamond s_2$			
$\Diamond\Box s_3$			
$\Box\Diamond(s_0 \lor s_3)$			
$\Diamond(s_2 \Rightarrow \bigcirc s_3)$			
$\exists\Box s_2$			
$\exists\Diamond s_3$		\	\

Notation : 1/4 point par bonne réponse ; -1/4 par ligne si absence d'explication ou explication erronée (sans jamais que la ligne compte négativement).

	sans équité	$WF(s_2, s_3)$	$SF(s_2, s_3)$	
$\Diamond s_2$	*o*	*o*	*o*	*(1)*
$\Diamond\Box s_3$	*n* $(s_0 \to s_1 \to {s_2}^{\omega})$	*n* $((s_0 \to s_1 \to {s_2}^{+})^{\omega})$	*o*	*(2)*
$\Box\Diamond(s_0 \vee s_3)$	*n* $(s_0 \to s_1 \to {s_2}^{\omega})$	*o*	*o*	*(3)*
$\Diamond(s_2 \Rightarrow \bigcirc s_3)$	*n* $(s_0 \to s_1 \to {s_2}^{\omega})$	*n* $((s_0 \to s_1 \to {s_2}^{+})^{\omega})$	*o*	*(4)*
$\exists\Box s_2$ *ou* $E\ G\ s_2$	*n*	*n*	*n*	*(5)*
$\exists\Diamond s_3$ *ou* $E\ F\ s_2$	*o*	\	\	*(6)*

(1) toute exécution commence par $s_0 \to s_1 \to s_2$.

(2) s_3 *n'est atteint dans toutes les exécutions qu'avec la SF, et ensuite on a* ${s_3}^{\omega}$.

(3) avec WF, on ne peut rester boucler sur ${s_2}^{\omega}$*, donc on va infiniment souvent en* s_0 *ou on part en* s_3 *pour y rester définitivement ; avec SF on va finalement continûment en* s_3.

(4) avec SF, on finit par faire une transition $s_2 \to s_3$.

(5) stupidement : on commence nécessairement en s_0 *donc on ne peut pas être qu'en* s_2*. La formule intéressante est* $\exists\Diamond\exists\Box s_2$ *ou* $\exists\Diamond(s_2 \wedge \exists\Box s_2)$.

(6) s_3 *est accessible depuis l'état initial.*

3 Problème : algorithme de Ricart-Agrawala (13 points [1])

L'algorithme de Ricart-Agrawala est un algorithme pour l'exclusion mutuelle, basé sur des autorisations. Quand un site souhaite obtenir l'accès exclusif, il demande l'autorisation à tous les autres sites. Quand il a obtenu l'autorisation de tous, il peut se considérer en exclusion mutuelle. Quand un site reçoit une requête d'autorisation, il répond favorablement s'il n'est pas intéressé par l'exclusion mutuelle, ou s'il est demandeur et que sa demande est postérieure à celle du requérant. Noter qu'il n'y a pas de refus : si un site est en exclusion ou s'il est demandeur avec une date antérieure, il reporte son autorisation après qu'il soit sorti d'exclusion mutuelle.

Un squelette de module TLA$^+$ `RicartAgrawala.tla` est fourni à la fin du sujet.

3.1 Module complet

1. Compléter l'action `EnvoyerAutorisation2`.

   $$\begin{array}{l} \mathit{EnvoyerAutorisation2}(i,j) == \\ \quad \wedge\ i \neq j \\ \quad \wedge\ \mathit{etat}[i] = \mathit{Demandeur} \\ \quad \wedge\ j \in \mathit{requetes}[i] \\ \quad \wedge\ \mathit{date}[j] < \mathit{date}[i] \\ \quad \wedge\ \mathit{autorisations}' = [\mathit{autorisations}\ \text{EXCEPT}\ ![j] = @ \cup \{i\}] \\ \quad \wedge\ \mathit{requetes}' = [\mathit{requetes}\ \text{EXCEPT}\ ![i] = @ \setminus \{j\}] \\ \quad \wedge\ \text{UNCHANGED}\ \langle \mathit{etat}, \mathit{temps}, \mathit{date} \rangle \end{array}$$

2. Définir le prédicat de transition `Next` qui représente toutes les transitions possibles du système.

   $$\begin{array}{ll} \mathit{Next} == & \vee\ \exists i \in \mathit{Site} : \mathit{Demander}(i) \vee \mathit{Entrer}(i) \vee \mathit{Sortir}(i) \\ & \vee\ \exists i, j \in \mathit{Site} : \mathit{EnvoyerAutorisation1}(i,j) \vee \mathit{EnvoyerAutorisation2}(i,j) \end{array}$$

1. Toutes les questions valent autant sauf la 10 qui vaut double.

3.2 Spécification

Exprimer en LTL ou CTL les propriétés suivantes (qui ne sont pas nécessairement vérifiées par le modèle) :

3. `ExclusionMutuelle` : il n'y a jamais plusieurs sites simultanément en exclusion mutuelle.
   $\Box(\forall i, j \in Site : etat[i] = Exclusif \wedge etat[j] = Exclusif \Rightarrow i = j)$
4. `AbsenceDeFamine` : tout demandeur finit par obtenir l'accès exclusif.
   $\forall i \in Site : etat[i] = Demandeur \rightsquigarrow etat[i] = Exclusif$
5. `NonContinûmentExclusif` : aucun site ne reste définitivement dans l'état exclusif.
   $\forall i \in Site : \Box\Diamond(etat[i] \neq Exclusif)$
   formule égale : $\forall i \in Site : \neg\Diamond\Box(etat[i] = Exclusif)$
   Acceptable : $\forall i \in Site : etat[i] = Exclusif \rightsquigarrow etat[i] = Hors$, *qui est plus fort que* $\forall i \in Site : etat[i] = Exclusif \rightsquigarrow etat[i] \neq Exclusif$ *(égal la première formule).*
6. `TempsNonBorné` : le temps peut prendre des valeurs arbitrairement grandes (mais ce n'est pas obligatoire, par exemple s'il cesse définitivement d'y avoir des demandes).
   $\forall v \in Nat : EF(tempsGlobal > v)$ *(ou* $\exists\Diamond(tempsGlobal > v)$
   formule de possibilité, exprimable uniquement en CTL.
7. `MinDate` : si un site est dans l'état exclusif, tous les autres demandeurs ont des dates de demande plus grandes.
   $\forall i \in Site : \Box(etat[i] = Exclusif => \forall j \in Site : (etat[j] = Demandeur \Rightarrow date[j] > date[i]))$

3.3 Équité

8. Énoncer l'équité minimale nécessaire pour que la propriété `NonContinûmentExclusif` soit vérifiée.
   Il faut que l'action Sortir(i) ait lieu pour éviter le bégaiement dans l'état Exclusif. Comme cette action est continûment faisable dans l'état exclusif, l'équité faible suffit :
   $\forall i \in Site : WF_{vars}(Sortir(i))$
9. Énoncer l'équité minimale nécessaire pour que la propriété `AbsenceDeFamine` soit vérifiée.
   Il faut éviter le bégaiement sur les états précédant Exclusif. Plus précisément, il faut assurer que les autorisations soient envoyées et que le site effectue l'entrée quand il peut. On a aussi besoin de l'équité comme précédemment sur Sortir. Comme toutes ces actions sont continûment faisables tant qu'elles ne sont pas faites, l'équité faisable suffit. Par contre, pas d'équité sur Demander, ce qui traduirait à tort qu'un site est obligé de faire des demandes.
   $\wedge \forall i \in Site : WF_{vars}(Entrer(i)) \wedge WF_{vars}(Sortir(i))$
   $\wedge \forall i, j \in Site : WF_{vars}(EnvoyerAutorisation1(i, j)) \wedge WF_{vars}(EnvoyerAutorisation2(i, j))$

3.4 Vérification

10. Dessiner le graphe de transitions (16 états) pour le cas particulier suivant : $N = 2$, $temps \leq 2$, le site 1 est demandeur en premier (le cas où c'est le site 2 étant symétrique). Dans un état, n'indiquer que les variables qui changent de valeur.
    cf à la fin, moitié gauche du graphe.

11. Comment utilise-t-on le graphe pour vérifier la propriété `ExclusionMutuelle` ?
(il n'est pas nécessaire d'avoir le graphe complet pour répondre)

Parcourir tous les états accessibles du graphe, et constater qu'il n'y a jamais un état avec deux (ou plus) sites en Exclusif.

12. Comment utilise-t-on le graphe pour vérifier la propriété `NonContinûmentExclusif` ?
(même remarque)

Vérifier qu'il n'y a pas de cycle infini où un site reste "exclusif". Ici, les seuls cycles possibles sont des boucles de bégaiement, il faut donc vérifier que, dans un état "exclusif", 'il y a toujours (au moins) une transition où l'équité s'applique, ce qui est le cas avec l'équité faible sur Sortir.

3.5 Module fourni : `RicartAgrawala.tla`

MODULE *RicartAgrawala*

EXTENDS *Naturals*, *FiniteSets*

CONSTANT N

$Site \triangleq 1 \,..\, N$

$Hors \triangleq$ "hors"
$Demandeur \triangleq$ "demandeur"
$Exclusif \triangleq$ "exclusif"
$Etat \triangleq \{Hors, Demandeur, Exclusif\}$

VARIABLES

etat,	état de chaque site : $Site \to \{Hors, Demandeur, Exclusif\}$
autorisations,	*autorisations* reçues par chaque site : $Site \to$ SUBSET $Site$
requetes,	requêtes en attente de traitement : $Site \to$ SUBSET $Site$
date,	date de la demande d'accès *exclusif* : $Site \to Nat$
temps	pour dater les demandes : Nat

$vars \triangleq \langle etat, autorisations, requetes, date, temps \rangle$

$TypeOK \triangleq$
$\land\ etat \in [Site \to Etat]$
$\land\ autorisations \in [Site \to \text{SUBSET } Site]$
$\land\ requetes \in [Site \to \text{SUBSET } Site]$
$\land\ temps \in Nat$
$\land\ date \in [Site \to Nat]$

$Init \triangleq$
$\land\ etat = [i \in Site \mapsto Hors]$
$\land\ date = [i \in Site \mapsto 0]$
$\land\ autorisations = [i \in Site \mapsto \{\}]$
$\land\ requetes = [i \in Site \mapsto \{\}]$
$\land\ temps = 0$

Met les *autorisations* reçues à vide, et dépose une requête de demande d'autorisation auprès de tous les sites. (y compris soi-même mais c'est sans importance)

$Demander(i) \triangleq$
$\land\ etat[i] = Hors$
$\land\ etat' = [etat \text{ EXCEPT } ![i] = Demandeur]$
$\land\ autorisations' = [autorisations \text{ EXCEPT } ![i] = \{\}]$
$\land\ requetes' = [j \in Site \mapsto requetes[j] \cup \{i\}]$
$\land\ date' = [date \text{ EXCEPT } ![i] = temps]$
$\land\ temps' = temps + 1$

i envoie son autorisation à j si i est non concerné par l'accès *exclusif*.
Enlève la requête de j et dépose l'autorisation de i.

$$
\begin{aligned}
&\mathit{EnvoyerAutorisation1}(i,\, j) \triangleq \\
&\quad \land\ i \neq j \\
&\quad \land\ \mathit{etat}[i] = \mathit{Hors} \\
&\quad \land\ j \in \mathit{requetes}[i] \\
&\quad \land\ \mathit{requetes}' = [\mathit{requetes} \text{ EXCEPT } ![i] = \mathit{requetes}[i] \setminus \{j\}] \\
&\quad \land\ \mathit{autorisations}' = [\mathit{autorisations} \text{ EXCEPT } ![j] = \mathit{autorisations}[j] \cup \{i\}] \\
&\quad \land\ \text{UNCHANGED } \langle \mathit{etat},\, \mathit{temps},\, \mathit{date} \rangle
\end{aligned}
$$

i envoie son autorisation à j si i est *demandeur* mais sa date est postérieure à celle de j.
Enlève la requête de j et dépose l'autorisation de i.

$\mathit{EnvoyerAutorisation2}(i,\, j) \triangleq \quad \ldots$ à compléter

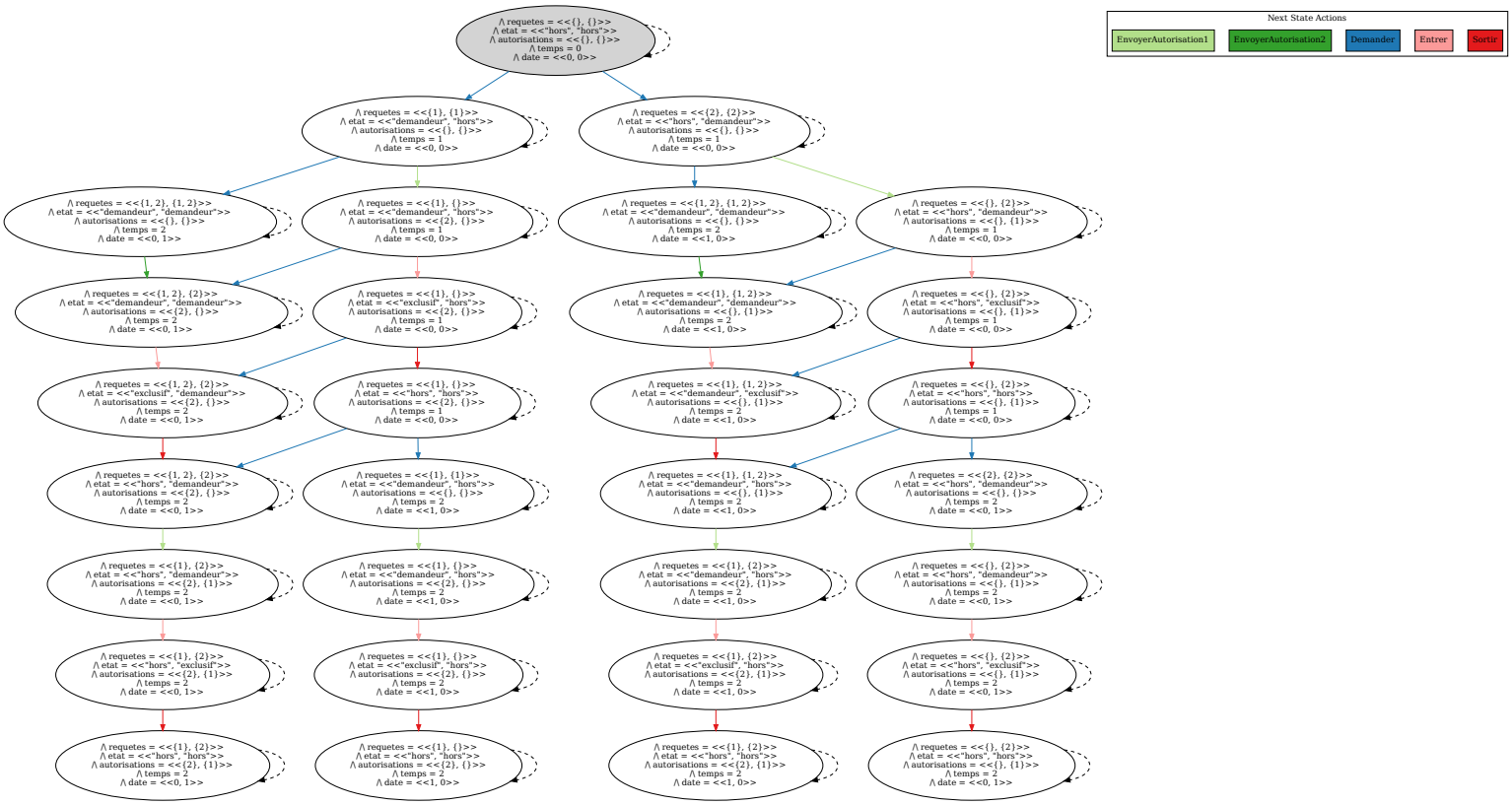
Entre dans l'état *exclusif* quand i a reçu toutes les *autorisations*.

$$
\begin{aligned}
&\mathit{Entrer}(i) \triangleq \\
&\quad \land\ \mathit{etat}[i] = \mathit{Demandeur} \\
&\quad \land\ \mathit{autorisations}[i] = \mathit{Site} \setminus \{i\} \\
&\quad \land\ \mathit{etat}' = [\mathit{etat} \text{ EXCEPT } ![i] = \mathit{Exclusif}] \\
&\quad \land\ \text{UNCHANGED } \langle \mathit{autorisations},\, \mathit{requetes},\, \mathit{date},\, \mathit{temps} \rangle
\end{aligned}
$$

Sort d'exclusion mutuelle.

$$
\begin{aligned}
&\mathit{Sortir}(i) \triangleq \\
&\quad \land\ \mathit{etat}[i] = \mathit{Exclusif} \\
&\quad \land\ \mathit{etat}' = [\mathit{etat} \text{ EXCEPT } ![i] = \mathit{Hors}] \\
&\quad \land\ \text{UNCHANGED } \langle \mathit{autorisations},\, \mathit{requetes},\, \mathit{date},\, \mathit{temps} \rangle
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{Next} \triangleq \ \ldots \\
&\mathit{Fairness} \triangleq \ \ldots \\
&\mathit{Spec} \triangleq \ \mathit{Init} \land \Box[\mathit{Next}]_{\mathit{vars}} \land \mathit{Fairness}
\end{aligned}
$$


Next State Actions
EnvoyerAutorisation1
EnvoyerAutorisation2
Demander
Entrer
Sortir