Systèmes Concurrents - Partie II 20/04/2022

Durée : 1 Heure Documents Autorisés
Les 2 exercices sont indépendants. Barème indicatif et non contractuel (14+6)

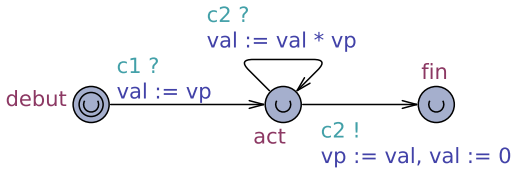
1 Puzzle

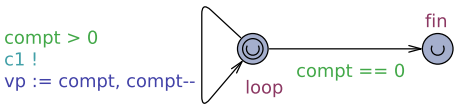
On considère le système Uppaal décrit ci-dessous

```
chan c1, c2;
int vp;
process cell(chan &c1, chan &c2) {
int val;
state debut, act, fin;
urgent debut, act, fin;
init debut;
trans
    debut -> act { sync c1 ?; assign val := vp;  },
    act -> act { sync c2 ?; assign val := val * vp;  },
    act -> fin { sync c2 !; assign vp := val, val := 0;  };}

process new(const int nb,  chan &c1) {
int compt := nb;
state loop, fin;
urgent loop,fin;
init loop;
trans
    loop -> loop { guard compt > 0; sync c1 !;
                   assign vp := compt, compt--;  },
    loop -> fin { guard compt == 0;  };}

reveil := new(4,c1) ;
cA := cell(c1,c2);
cB := cell(c1,c2);
cC := cell(c1,c2);
cD := cell(c1,c2);
system  reveil, cA, cB, cC, cD;
```





1.1 Questions :

1.1.1 Compréhension du système (8/20)

1. Proposez un scenario menant à un état du système où l'instance `reveil` et 3 instances du process `cell` sont dans l'état `fin`.
   Vous décrirez la suite des états du système constituant ce scenario en mentionnant l'état des variables (locales et partagées) en adoptant, par exemple, la représentation ci-dessous :

	vp	Reveil	cA	cB	cC	cD
Initial :	0	(Compt = 4, loop)	(val = 0, debut)	(val = 0, debut)	(val = 0, debut)	(val = 0, debut)
s1 : s2 : .../...	4	(Compt = 3, loop)	(val = 0, debut)	(val = 4, act)	(val = 0, debut)	(val = 0, debut)

2. Quelle est la forme générale d'un état de blocage ?
3. Sans faire de calcul, donnez le nombre d'états de blocage.
4. Quelle fonction permet de calculer ce système ?

1.1.2 Propriétés (6/20)

Donnez une interprétation aux propriétés suivantes ; vous indiquerez si elles sont vraies ou fausses. Pour les propriétés fausses, vous justifierez rapidement vos réponses.

1. $E <>\ cA.fin\ and\ cB.fin\ and\ cC.fin\ and\ cD.act$
2. $E <>\ cA.fin\ and\ cB.fin\ and\ cC.fin\ and\ cD.fin$
3. $reveil.loop\ -->\ reveil.fin$
4. $A <>\ (cA.val + cB.val + cC.val + cD.val == 24)$

2 Communication Asynchrone (6/20)

On considère le système UPPAAL représenté ci-contre

```
chan EtoB, BtoR;
int vp;

process osb(chan &in, chan &out){
int loc;
state   empty,full;
urgent  empty,full;
init    empty;
trans
 empty -> full{sync in? ;
               assign loc := vp;},
 full  ->  empty{sync out! ;
                 assign vp := loc;};}

process sender(chan &out){
state   serv;
init    serv;
trans  serv -> serv{sync out! ;
              assign vp := 1};}

process receiver(chan &in){
int loc := 0;
state   serv;
init    serv;
trans serv -> serv{sync in? ;
                   assign loc :=vp;};}

 emetteur :=  sender(EtoB);
 tamp :=      osb(EtoB,BtoR);
 recepteur := receiver(BtoR);
system emetteur, tamp, recepteur;
```

Indications :
Le processus `osb` correspond à une mémoire tampon (buffer) à une place : losrque le buffer est vide, il peut accepter et mémoriser la valeur qu'on lui confie (il est alors plein), un buffer plein peut envoyer la valeur qu'il mémorise et retourner à l'état vide.
Le système décrit ci-contre permet de mettre en oeuvre une communication asynchrone entre l'`emetteur` et le `récepteur`.

Questions :

1. Sans modifier les processus actuels [a], proposez un système permettant une communication par fifo de taille 3 entre l'`émetteur` et le `récepteur`.
2. A quoi correspondrait l'instantiation suivante ?

```
.../...
emetteur1 :=  sender(EtoB);
emetteur2 :=  sender(EtoB);
tamp1 :=      osb(EtoB,EtoQ);
tamp2 :=      osb(EtoQ,BtoR);
recepteur1 := receiver(BtoR);
recepteur2 := receiver(BtoR);
```

a. Vous pouvez uniquement ajouter de nouveaux canaux et de nouvelles instances de processus