

TP d'introduction au model-checking

Cette série d'exercices est inspirée de la documentation du model-checker SPIN.

Prise en main de SPIN

Exclusion mutuelle

Exercice 1

Considérez ci-après le modèle Promela (**Produce-Consumer**) d'un algorithme d'exclusion mutuelle. Ouvrir ce modèle dans xspin et répondre aux questions suivantes en utilisant spin.

1. Quelle la fonction ou sémantique de l'expression **atomic**?
2. Expliquez les fonctions **request** et **release** et la sémantique de l'expression **inline**
3. Expliquez la fonction de l'expression **assert**.
4. Expliquez la fonction de la variable **who**.
5. Donnez FSM de ce modèle (Produce-Consumer)

```

mtype = {P, C,N}; /* symbols used */
mtype turn = P; /* shared variable */
pid who

inline request(x, y, z){
    atomic(x == y -> x = z ; who = _pid)
}

Inline release (x, y) {
    atomic{ x= y; who = 0}
}

active [2] proctype producer() {

do
:: request(turn, P, N) -> /* Guard */
    printf("P%d\n", _pid);
    assert(who == _pid);
    release(turn, C)
od

}

Active [2] proctype consumer() {

do
:: request(turn, P, N) -> /* Guard */
    printf("P%d\n", _pid);
    assert(who == _pid);
    release(turn, P)
od

}

```

Exercice 2

Considérez le modèle Promela (**Dekker**) d'un algorithme d'exclusion mutuelle – ci-après. Ouvrir ce modèle dans xspin et répondre aux questions suivantes en utilisant spin.

1. Donnez la FSM de ce modèle
2. Quelle est la fonction de la variable **flag** ?
3. Cet algorithme est-il correct?

```
bit    turn;
bool   flag[2];
byte   cnt;

active [2] proctype mutex()
{
    pid i, j;

    i = _pid;
    j = 1 - _pid;
again:
    flag[i] = true;
    do
    :: flag[j] ->
        if
        :: turn == j ->
            flag[i] = false;
            (turn != j) -> /* wait until true */
            flag[i] = true
        :: else ->
            skip           /* do nothing */
        fi
    :: else ->
        break             /* break from loop */
    od;

    cnt++;
    assert(cnt == 1);      /* critical section */
    cnt--;

    turn = j;
    flag[i] = false;
    goto again
}
```

Transfert de fichier

Exercice 3

Voir ci-après le modèle Promela (**transfer1**) du protocole de transfert de fichier.

1. Donnez la FSM de ce modèle
2. Que représentent les déclarations suivantes **chan M = [1] of {mtype}** et **chan w = [1] of {mtype}**?
3. Quelle est la sémantique des instructions **W!ini** et **M?ack**
4. Expliquez le bloc **if...fi** du processus Mproc et **do...od** du processus Wproc.

```
mtype = { ini, ack, dreq, data, shutup, quiet, dead };
```

```
chan M = [1] of { mtype };
```

```
chan W = [1] of { mtype };
```

```
active proctype Mproc()
```

```
{
    W!ini;                /* connection      */
    M?ack;                /* handshake      */

    timeout ->           /* wait           */
    if                   /* two options:   */
    :: W!shutup          /* start shutdown */
    :: W!dreq;           /* or request data */
    M?data ->           /* receive data   */
    do
        :: W!data        /* send data      */
        :: W!shutup;     /* or shutdown    */
        break
    od
    fi;

    M?shutup;            /* shutdown handshake */
    W!quiet;
    M?dead
}
```

```
active proctype Wproc()
```

```
{
    W?ini;                /* wait for ini   */
    M!ack;                /* acknowledge    */

    do                   /* 3 options:     */
    :: W?dreq ->         /* data requested */
        M!data          /* send data      */
    :: W?data ->         /* receive data   */
        skip            /* no response    */
    :: W?shutup ->      /* start shutdown */
        M!shutup;
        break
    od;

    W?quiet;
    M!dead
}
```

Exercice 4 (Algorithme de Peterson)

Exprimer l'algorithme suivant en PROMELA et vérifier avec Spin s'il satisfait les conditions requises pour traiter le problème de l'exclusion mutuelle.

var drapeau : tableau [0..1] de **bool** init faux;

tour : 0..1 init quelconque;

processus : **repeter**

Section non critique;

drapeau[i]=vrai;

tour=i;

tant que drapeau[j] et tour ≠ j **faire**

ftq

Section Critique;

drapeau[i]=faux;

jusqu'a faux

Exercice 5 (Algorithme de Dijkstra)

L'algorithme suivant est conçu pour un nombre quelconque **n** de processus. L'exprimer en PROMELA et vérifier avec Spin s'il satisfait les conditions requises pour traiter le problème de l'exclusion mutuelle. Vous pourrez utiliser 3 et 4 comme valeurs de **n**.

var

drapeau : tableau [0..n-1] de {inactif, demande, enSC} init inactif;

tour : 0..n-1 init quelconque;

processus :

repeter

Section non critique;

repeter

drapeau[i]=demande;

tant que tour ≠ i **faire**

si drapeau[tour]=inactif

```

        alors tour=i;

    fsi

ftq
drapeau[i]=enSC;

j=0;

tant que (j<n) et (j=i ou drapeau[j]≠enSC) faire
    j=j+1;

ftq

jusqu'a j≥n

Section Critique;

drapeau[i]=inactif;

jusqu'a faux

```

Exercice 6 (TrafficLight)

On souhaite réaliser un programme de contrôle de carrefour de circulation automobile à quatre voies et (donc) quatre feux synchronisés. Le système de contrôle sera composé plusieurs processus concurrents :

1. Un processus Lamp pour chaque lampe de feu (avec 4 états : allumé, éteint, clignotant et en panne)
2. Un processus Light qui regroupe trois lampes : rouge, orange, vert et décrit les interactions entre ces lampes
3. Deux processus Control qui se chargent des interactions entre les deux feux d'une même voie. Les deux processus devront se synchroniser.

Question 1 Décrire les propriétés informelles attendues pour ce système

Question 2 Proposez une FSM pour ce problème

Question 3 Proposer un modèle du système sous la forme d'un fichier TrafficLight.pml pour Promela. Décrire sous la forme d'assertions les propriétés formelles attendues pour le système. Vérifier que le système respecte toutes les assertions proposées.

Exercice 7 (Le modèle de lecteurs/rédacteurs)

Il s'agit de résoudre le problème d'accès à une ressource commune (un fichier par exemple) devant être partagée par deux types de processus :

- Les lecteurs qui consultent le fichier sans le modifier. Ils peuvent donc l'accéder en parallèle

- Les rédacteurs qui accèdent au fichier pour le modifier. Ils doivent accéder au fichier en exclusion mutuelle entre eux et en exclusion mutuelle avec les lecteurs.

Proposer un modèle Promela pour ce problème.