II - Problème des cheminots

Premier cas :

Pierre=FAUX ;

```
Processus Péruvien
Début
   Tant que VRAI faire
   Début
          /*Peruvien arrête train*/
(a1)      Tant que (Pierre=VRAI) faire
          Début
             /*Péruvien fait la sieste*/
          Fin
          /*Péruvien cherche et mets la pierre*/
(b1)      Pierre=VRAI ;
          <SC>
             /*Péruvien passe*/
          </SC>
          /*Péruvien retourne au panier et enlève la pierre*/
(c1)      Pierre=FAUX ;
          /*Péruvien repart*/
   Fin
Fin
```

```
Processus Bolivien
Début
   Tant que VRAI faire
   Début
          /*Bolivien arrête train*/
(a2)      Tant que (Pierre=VRAI) faire
          Début
             /*Bolivien fait la sieste*/
          Fin
          /*Bolivien cherche et mets la pierre*/
(b2)      Pierre=VRAI ;
          <SC>
             /*Bolivien passe*/
          </SC>
          /*Bolivien retourne au panier et enlève la pierre*/
(c2)      Pierre=FAUX ;
          /*Bolivien repart*/
   Fin
Fin
```

Propriété 1 :

Pierre : FAUX

Péruvien	Bolivien	Pierre
(a1)		
	(a2)	
(b1)		VRAI
	(b2)	VRAI
<SC>		
	<SC>	

PP1 NON VÉRIFIÉE

Propriété 2 :

Si Péruvien demande la <SC> et est le seul à la demander, il peut y entrer car Pierre=FAUX : il n'y a donc pas d'attente.
Idem pour Bolivien.
PP2 VÉRIFIÉE

Propriété 3 :

Supposons Péruvien en <SC> et Bolivien en attente :
=>Pierre=VRAI
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre=FAUX. Bolivien peut maintenant, sortir de son attente active en (a2).
PP3 VÉRIFIÉE

Interblocage :

Pierre est initialisé à FAUX, et est affecté à FAUX lors du protocole de sortie des deux processus : il ne peut jamais être VRAI lorsque les deux processus avancent dans leur instruction (a1) et (a2).
PAS D'INTERBLOCAGE

Equité :

Supposons Péruvien en <SC> et Bolivien en attente :
=>Pierre=VRAI
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre=FAUX.
Péruvien passe en protocole d'entrée puis redemande la <SC> : Il l'obtient.
NON EQUITABLE

Dans ce premier cas, la propriété 1 n'est pas respecté ; si les deux processus peuvent entrer en section critique simultanément, les deux trains peuvent entrer en collision… Il n'y a pas de priorité donné à qui que ce soit, c'est pour cela qu'il y a un accident.

Deuxième cas :

Pierre=FAUX ;

```
Processus Péruvien
Début
   Tant que VRAI faire
   Début
         /*Peruvien arrête train*/
(a1)     Tant que (Pierre=FAUX) faire
         Début
            /*Péruvien fait la sieste*/
         Fin
         <SC>
            /*Péruvien passe*/
         </SC>
         /*Péruvien retourne au panier et enlève la pierre*/
(b1)     Pierre=FAUX ;
         /*Péruvien repart*/
   Fin
Fin
```

```
Processus Bolivien
Début
   Tant que VRAI faire
   Début
         /*Bolivien arrête train*/
(a2)     Tant que (Pierre=VRAI) faire
         Début
            /*Bolivien fait la sieste*/
         Fin
         <SC>
            /*Bolivien passe*/
         </SC>
         /*Bolivien retourne au panier et met une pierre*/
(b2)     Pierre=VRAI;
         /*Bolivien repart*/
   Fin
Fin
```

Propriété 1 :

Supposons Péruvien et Bolivien en <SC>
• Péruvien en <SC> => Pierre=VRAI
• Bolivien en <SC> => Pierre=FAUX
CONTRADICTION, PP1 VÉRIFIÉE

Propriété 2 :

Après initialisation, si Bolivien demande la <SC> et est le seul à la demander, il peut y entrer car Pierre=FAUX : il n'y a donc pas d'attente.
En revanche, si Péruvien demande la <SC> et est le seul à la demander, il ne peut pas y entrer. Il est obligé d'attendre que Péruvien mette Pierre à VRAI.
PP2 NON VÉRIFIÉE

Propriété 3 :

Supposons Péruvien en <SC> et Bolivien en attente :
=> Pierre=VRAI
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre=FAUX. Bolivien peut maintenant sortir de son attente active en (a2).
PP3 VÉRIFIÉE

Interblocage :

La variable Pierre est booléenne, elle ne peut prendre que VRAI ou FAUX :
- Si Pierre=VRAI alors Péruvien passe.
- Si Pierre=FAUX alors Bolivien passe.
PAS D'INTERBLOCAGE

Equité :

Supposons Péruvien en <SC> et Bolivien en attente :
=>Pierre=VRAI
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre=FAUX.
Péruvien passe en protocole d'entrée puis redemande la <SC> : Il ne l'obtient pas, Bolivien entre en <SC>.
ÉQUITABLE

Dans ce deuxième cas, la propriété 2 n'est pas respecté ; si un processus est prêt à entrer en section critique, il ne peut pas y acceder sans attendre l'autre. Dans la situation des cheminots, c'est pour cette raison qu'il n'y a pas autant de trains qu'avant.. La condition d'attente est trop contraignante, il faut la modifier si on veut obtenir un acces à la sectionc ritique plus facile.

Troisième cas :

Pierre1=FAUX, Pierre2=FAUX ;

```
Processus Péruvien
Début
   Tant que VRAI faire
   Début
            /*Peruvien arrête train et met sa pierre*/
(a1)        Pierre1=VRAI ;
(b1)        Tant que (Pierre2=VRAI) faire
            Début
(c1)           Pierre1=FAUX ;
               /*Péruvien enlève sa pierre et fais la sieste*/
            Fin
            /*Peruvien remet sa pierre */
            Pierre1=VRAI
            <SC>
               /*Péruvien passe*/
            </SC>
            /*Péruvien retourne au panier et enlève sa pierre*/
(c1)        Pierre1=FAUX ;
            /*Péruvien repart*/
   Fin
Fin
```

```
Processus Bolivien
Début
   Tant que VRAI faire
   Début
            /*Bolivien arrête train et met sa pierre*/
(a1)        Pierre2=VRAI ;
(b2)        Tant que (Pierre1=VRAI) faire
            Début
               /*Bolivien enlève sa pierre et fais la sieste*/
(c2)           Pierre2=FAUX ;
            Fin
            /*Bolivien remet sa pierre*/
            Pierre2=VRAI ;
            <SC>
               /*Bolivien passe*/
            </SC>
            /*Bolivien retourne au panier et enlève sa pierre*/
(c2)        Pierre=FAUX ;
            /*Bolivien repart*/
   Fin
Fin
```

Propriété 1 :

Supposons Péruvien en <SC> :
=>Pierre1=VRAI
=>Pierre2=FAUX
Supposons Bolivien en <SC> :
=>Pierre2=VRAI
=>Pierre1=FAUX
CONTRADICTION, PP1 VÉRIFIÉE

Propriété 2 :

Après initialisation, si Péruvien demande la <SC> et est le seul à la demander, il peut y entrer car Pierre2=FAUX : il n'y a donc pas d'attente.
De même si Bolivien demande la <SC> et est le seul à demander, il y entre : Pierre1=FAUX.
PP2 VÉRIFIÉE

Propriété 3 :

Supposons Péruvien en <SC> et Bolivien en attente :
=> Pierre1=VRAI
=> Pierre2=FAUX
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre1=FAUX. Bolivien peut maintenant sortir de son attente active en (b2) et entrer en <SC>.
PP3 VÉRIFIÉE

Interblocage :

Les variables Pierre1 et Pierre2 sont booléennes, elle ne peuvent prendre comme valeur que VRAI ou FAUX. Le changement de valeur qui permet de sortir de la condition d'attente se trouve dans le processus opposé à celui qui attend : dès lors que l'un des deux processus change la valeur pour l'autre il débloque ce dernier et l'envoie en <SC>.
PAS D'INTERBLOCAGE

Equité :

Supposons Péruvien en <SC> et Bolivien en attente :
=> Pierre1=VRAI
=> Pierre2=FAUX
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre1=FAUX.
Péruvien repasse en protocole d'entrée puis redemande la <SC> : Il l'obtient, en passant devant le Bolivien.
NON ÉQUITABLE

Solution proposé :

La solution au problème consiste à implementer l'algorithme de Peterson pour les cheminots :

Pierre1=FAUX ; Pierre 2=FAUX ; tour=1

```
Processus Péruvien
Début
   Tant que VRAI faire
   Début
         /*Peruvien arrête train et met sa pierre*/
(a1)     Pierre1=VRAI ;
         /*Peruvien donne le tour à Bolivien*/
(b1)     tour=2 ;
(c1)     Tant que (Pierre2=VRAI ET tour=2) faire
         Début
            /*Péruvien enlève sa pierre et fais la sieste*/
         Fin
         <SC>
            /*Péruvien passe*/
         </SC>
         /*Péruvien retourne au panier et enlève sa pierre*/
(d1)     Pierre1=FAUX ;
         /*Péruvien repart*/
   Fin
Fin
```

```
Processus Bolivien
Début
   Tant que VRAI faire
   Début
         /*Bolivien arrête train et met sa pierre*/
(a2)     Pierre2=VRAI ;
         /*Bolivien donne le tour à Péruvien*/
(b2)     tour=1 ;
(c2)     Tant que (Pierre1=VRAI ET tour=1) faire
         Début
            /*Bolivien enlève sa pierre et fais la sieste*/
         Fin
         <SC>
            /*Bolivien passe*/
         </SC>
         /*Bolivien retourne au panier et enlève sa pierre*/
(d2)     Pierre2=FAUX ;
         /*Bolivien repart*/
   Fin
Fin
```

Propriété 1 :

Supposons Péruvien en <SC> :
=>Pierre1=VRAI
=>Pierre2=FAUX
=>tour=2
Supposons Bolivien en <SC> :
=>Pierre2=VRAI
=>Pierre1=FAUX
=>tour=1
CONTRADICTION, PP1 VÉRIFIÉE

Propriété 2 :

Après initialisation, si Péruvien demande la <SC> et est le seul à la demander, il peut y entrer car Pierre2=FAUX : il n'y a donc pas d'attente.
De même si Bolivien demande la <SC> et est le seul à demander, il y entre : Pierre1=FAUX.
PP2 VÉRIFIÉE

Propriété 3 :

Supposons Péruvien en <SC> et Bolivien en attente :
=> Pierre1=VRAI
=> Pierre2=FAUX
=> tour=2
Péruvien sort de <SC> et éxécute son protocole de sortie : Pierre1=FAUX. Bolivien peut maintenant sortir de son attente active en (c2) et entrer en <SC>
PP3 VÉRIFIÉE

Interblocage :

La variable tour est booléenne : il est impossible pour elle d'avoir deux valeurs ne même temps. Sachant qu'une parti nécessaire de la condition d'attente en découle, il est impossible d'avoir les deux processus en attente.
PAS D'INTERBLOCAGE

Equité :

La variable tour sert à donner la priorité à un processus. Si c'est le tour du processus 2 alors celui-ci est prioritaire à la section critique si le processus 1 n'y est pas déjà. Un processus peut toutefois s'éxecuter plusieurs fois si l'autre ne souhaite pas acceder à la section critique.
ÉQUITABLE

Dans ce dernier cas, l'ensemble des propriétés est respecté : on peut appliquer cet manière de faire à notre problème ; cela devrait régler les problèmes féroviaires dans les Andes.

Les codes ADA de ces questions sont disponibles en annexe, nous avons jugé plus facile de démontrer les propriétés sous forme de pseudo-code (c'est aussi plus agréable à l'oeil).