JUDIT NIETO PARLA

PRÁCTICA 2 : PUENTE DE AMBITE

PRIMERA VERSIÓN

Tenemos un puente compartido por coches y peatones. La anchura del puente no permite el paso de vehículos en ambos sentidos. Además, por motivos de seguridad, los peatones y los coches no pueden compartir el puente. Los peatones pueden pasar en sentido contrario.

cars_north ⇝ coches que quieren entrar desde el norte
cars_south ⇝ coches que quieren entrar desde el sur
pedestrian ⇝ peatones que quieren entrar (dirección indiferente)
dentro_norte ⇝ coches del norte dentro del puente
dentro_sur ⇝ coches del sur dentro del puente
dentro_peaton ⇝ peatones dentro del puente

Definimos las siguientes funciones

```
def adelante_coches_norte():
    dentro_sur==0 ∧ dentro_peaton==0
def adelante_coches_sur():
    dentro_norte==0 ∧ dentro_peaton==0
def adelante_peaton():
    dentro_norte==0 ∧ dentro_sur==0
```

que nos indican cuando pueden pasar los coches del norte, del sur y los peatones, respectivamente.

Declaramos las variables condición

abierto_norte = CV
abierto_sur = CV
abierto_peaton = CV

Si quiere entrar un coche en dirección sur, esperamos hasta que pueda hacerlo, es decir, hasta que no haya dentro un coche del norte ni un peaton

```
abierto_sur.wait_for(adelante_coches_sur)
```

Si quiere entrar un coche en dirección norte, esperamos hasta que pueda hacerlo, es decir, hasta que no haya dentro un coche del norte ni un peaton.

abierto_norte.wait_for(adelante_coches_norte)

Lo mismo ocurre con el peaton, podrá entrar al puente cuando no haya ningún coche dentro.

abierto_peaton.wait_for(adelante_peaton)

De esta manera podemos demostrar que el puente es seguro.

$$\{Inv \equiv dento\_sur \geq 0 \wedge dentro\_norte \geq 0 \wedge dentro\_peaton \geq 0 \wedge$$
$$dento\_sur > 0 \rightarrow (dentro\_norte = 0 \wedge dentro\_peaton = 0) \wedge$$
$$dentro\_norte > 0 \rightarrow (dentro\_sur = 0 \wedge dentro\_peaton = 0) \wedge$$
$$dentro\_peaton > 0 \rightarrow (dentro\_sur = 0 \wedge dentro\_norte = 0)\}$$

No hay deadlocks ya que cuando dento del puente no hay ningún coche ni ningún peaton se hace saber con ".notify_all()" pudiendo entrar al puente aquellos coches o peatones que hubiera esperando.

En este caso, nos encontramos con problemas de inanición. Por ello, hemos implementado una segunda versión.

SEGUNDA VERSIÓN

Esta segunda versión es simplemente una ampliación de la primera. Ahora, habrá un turno de entrada.

- 0 para los coches del norte
- 1 para los coches del sur
- 2 para los peatones
- -1 libre

De esta forma, tendremos

```
def adelante_norte():
    dentro_sur == 0 ∧ dentro_peaton == 0 ∧ (turno == 0 ∨ turno == -1)

def adelante_sur():
    dentro_norte == 0 ∧ dentro_peaton == 0 ∧ (turno == 1 ∨ turno == -1)

def adelante_peaton():
    dentro_sur == 0 ∧ dentro_norte == 0 ∧ (turno == 2 ∨ turno == -1)
```

Como en el caso anterior, aseguramos que el puente es seguro.

$$
\begin{aligned}
\{INV \equiv\ & dentro\_sur \geq 0 \wedge dentro\_norte \geq 0 \wedge dentro\_peaton \geq 0 \wedge \\
& dentro\_sur > 0 \rightarrow (dentro\_norte == 0 \wedge dentro\_peaton == 0 \wedge \\
& \qquad \wedge (turno == 1 \vee turno == -1)) \wedge \\
& dentro\_norte > 0 \rightarrow (dentro\_sur == 0 \wedge dentro\_peaton == 0 \wedge \\
& \qquad \wedge (turno == 0 \vee turno == -1)) \wedge \\
& dentro\_peaton > 0 \rightarrow (dentro\_sur == 0 \wedge dentro\_norte == 0 \wedge \\
& \qquad (turno == 2 \vee turno == -1))\}
\end{aligned}
$$

Si un proceso está despierto y puede pasar, esperará a que pasen el puente los que estaban pasando anteriormente o directamente pasa. Si un proceso está dormido en su turno, esperará y después se despertará para poder pasar. Con esto hemos solucionado los problemas de inanición.