Sólo mira si hay coches en la dirección contraria (puede crear inanición)

Monitor

north: value
south: value

no_north: condition
no_south: condition

$\{Inv: north \geq 0 \land south \geq 0 \land (north > 0 \rightarrow south = 0) \land (south > 0 \rightarrow north = 0)\}$

wants_enter (dir) {

    {Inv}

    if (dir == norte) {

        no_south.wait (south == 0)
          {Inv $\land$ south = 0}
        north += 1
        {Inv} ← podemos aumentar north porque sabemos que south = 0

    } else {

        no_north.wait (north == 0)
          {Inv $\land$ north = 0}
        south += 1
        {Inv} ← podemos aumentar south porque sabemos que north = 0

    }

}

leaves_tunnel (dir) { esta función se llama cuando hay al menos un coche en el túnel

    if (dir == norte) { ← hay al menos un coche dirección norte
        {Inv $\land$ north > 0}
        north -= 1
        {Inv}
        no_north.release() ← Disminuimos north por lo que puede ocurrir que north = 0 y hay que despertar los coches dirección sur que están esperando.

    } else { ← hay al menos un coche dirección sur
        {Inv $\land$ south > 0}
        south -= 1
        {Inv}
        no_south.release() ← Disminuimos south por lo que puede ocurrir que south = 0 y hay que despertar los coches dirección norte que están esperando.

    }

}

Deja pasar en función de los coches que ya hayan pasado.

Monitor

south : value
north : value
waiting_south : value
waiting_north : value
had_passed_south : value
had_passed_north : value

no_south : condition
no_north : condition

$\{Inv: south \geq 0 \land north \geq 0 \land (north > 0 \rightarrow south = 0) \land$
$\land (south > 0 \rightarrow north = 0) \land had\_passed\_south \geq 0 \land$
$\land had\_passed\_north \geq 0 \land (had\_passed\_south > 0 \rightarrow$
$\rightarrow had\_passed\_north = 0) \land (had\_passed\_north > 0 \rightarrow$
$\rightarrow had\_passed\_south = 0) \land waiting\_south \geq 0 \land$
$\land waiting\_north \geq 0\}$

```
wants_enter(dir){
    if (dir = norte){
        {Inv}
        waiting_north += 1
        no_south.wait(south = 0 ∧ (had_passed_north < K ∨ waiting_south = 0))
        waiting_north -= 1 ← lo acabamos de aumentar así que se puede disminuir
        had_passed_south = 0
        north += 1 ← sabemos que south = 0 así que north puede ser > 0.
        had_passed_north += 1 ← sabemos que had_passed_south = 0 así que podemos aumentar.
        {Inv}
    } else {
        {Inv}
        waiting_south += 1
        no_north.wait(north = 0 ∧ (had_passed_south < K ∨ waiting_north = 0))
        waiting_south -= 1 acabamos de aumentarlo así que se puede disminuir.
        had_passed_north = 0
        south += 1 ← sabemos que north = 0 así que puede aumentar.
        had_passed_south += 1 ← sabemos que had_passed_north = 0 así que puede aumentar.
        {Inv}
    }
}
```

```
leaves_tunnel(dir){ ← sólo se llama a esta función si hay algún coche en
    {Inv ∧ dir > 0 }                                              el túnel.
    dir -= 1 ← podemos disminuir porque dir > 0
    no_dir.release() ← puede ocurrir dir = 0, hay que despertar coches.
```