

Taller 5

Grupo 6 (F)

LU	Integrante
72/18	Manuel Panichelli
692/12	Elías Cerdeira

Ejercicio 1

```
R =  
  P0, Q0  
  P1, Q1  
  P2, Q2  
  P1, Q3  
  P3, Q2
```

Ejercicio 2

La relación es

```
R =  
  P0, Q0  
  P2, Q1  
  P3, Q2  
  P1, Q3  
  P2, Q4
```

Ejercicio 3

Estrategia ganadora representada como "árbol".

```
P, Q  
(0, 0)  
  -> a (1, 0)  
    -> a (1, 1)  
      -> tau (2, 1)  
        -> nada (2, 1)  
          -> tau (2, 2)  
            -> nada (2, 2)  
              -> c (3, 2)  
                ganó
```

```
-> tau (2, 2)
-> c (3, 2)
ganó
```

Ejercicio 5

Lo hicimos agregando ocultamiento de todo el comportamiento no observable a PRIMES

```
@ {filter[0..3].prime[2..15], end}
```

Ejercicio 6

```
Q = (a -> STOP).
P = (t -> a -> STOP) @ {a}.
```

tienen trazas a pero no son fuertemente bisimilares

Ejercicio 7

```
property CajaDeCambios = (
  pisoEmbrague -> (
    sueltoEmbrague -> CajaDeCambios |
    nuevoPalanca -> sueltoEmbrague -> CajaDeCambios
  )
).
```

Ejercicio 8

```
property UnoYDos = (uno -> STOP) + {dos}.
```fsp
```

## Ejercicio 9

En [ej9.lts](#)

## Ejercicio 10

En [ej10.lts](#)

## Ejercicio 11

Si, pues Obs permite que se haga a primero (y no b) y luego de hacer a se tiene que hacer b, no se puede hacer a de nuevo. Además, en ningún momento se puede hacer c.

## Ejercicio 12

En `ej12.lts`

```
// a. En toda traza con fair choice algún proceso se declara líder.
progress AlwaysOneLeader = {proc[i:1..N].leader}

// b. En toda traza un proceso particular se declara líder.
progress AlwaysSameLeader = {proc[1].leader}
```

## Ejercicio 13

En `ej13.lts`

No, porque R podría bloquear a Q. Por ejemplo,

```
Q = (a -> Q).
R = (a -> b -> STOP) + {a}.

||QR = (Q || R).
progress SiempreA = {a}
/*
Progress violation: SiempreA
Trace to terminal set of states:
 a
 b
Cycle in terminal set:
Actions in terminal set:
 {}
*/
```

## Ejercicio 14

- a. `[]enBase` safety porque un contraejemplo finito es `!enBase`
- b. `[](bateriaBaja => X (modoAhorro U enBase))` safety, contraej finito es `bateriaBaja, !modoAhorro`
- c. `[](paredDelante => X (girandoAIzquierda U !paredDelante))`

## Ejercicio 15

- a. no
- b. si
- c. no, podría nunca pasar alguno de los dos
- d. si

- e. no.  $(a*b)^*c / a^*(b*c)$

## Ejercicio 16

Las LTL se evalúan sobre las trazas de los LTSs. Cada una se modela como una estructura de kripke separada en la que cada acción es un nodo. Luego, nunca puede haber más de un elemento en un nodo, y como para cumplirse a y b tienen que estar los dos en un nodo, nunca puede suceder.

## Ejercicio 17

- a. `[]enBase`  
`!<>salióDeBase`
- b. `[](bateriaBaja => X (modoAhorro U enBase))`  
`[](bateríaBaja => X (modoAhorroOn ^ (!modoAhorroOff U entróABase)))`
- c. `[](paredDelante => X (girandoAIzquierda U !paredDelante))`  
`[](paredDelanteDetectado -> X girandoAIzquierda)`

## Ejercicio 18

1. verdadero
2. falso, abbbbbb...
3. falso, abbbbbb...
4. verdadero
5. falso, aacacacacac...
6. falso, abbbbbb...

## Ejercicio 20

En `ej20.lts`