## Verificação Formal de Software - Exercícios 2

### Paralelismo e Comunicação

1. Mostra que o operador de *handshaking* $||$ em que a sincronização de dois sistemas é nas ações comuns (i.e na interseção dos seus conjuntos de ações) é associativa:

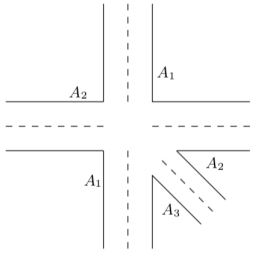
$$(T_1||T_2)||T_3 = T_1||(T_2||T_3).$$

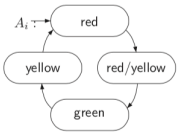
2. Considera os processos $P_i$ para $i = 1, 2, 3$ que partilham uma variável inteira $x$, têm variáveis locais $r_i$ e onde cada processo tem as seguintes instruções:

   **for** $k_i = 1, \ldots, 10$ **do**
   $r_i := x;$
   $r_i := r_i + 1;$
   $x := r_i;$

   e considera a seguinte programa paralelo $P = P_1||P_2||P_3$ com $x$ inicialmente em 0. Será possível que $P$ tem uma execução que termina com $x = 2$ ?

3. Considera a seguinte junção de ruas e um semáforo com a especificação à direita.




   (a) Escolhe ações e etiqueta as transições do sistema de transições do semáforo de forma adequada.

   (b) Determina um sistema de transições para um controlador $C$ que permita que a luz verde alterne da seguinte forma: $A_1, A_2, A_3, A_1, A_2, A_3, \ldots$.

   (c) Esboça o sistema de transições para $A_1||A_2||A_3||C$

4. Mostra que o operador de *handshaking* $||$ em que a sincronização de dois sistemas é nas ações comuns (i.e na interseção dos seus conjuntos de ações) é associativa:

$$(T_1||T_2)||T_3 = T_1||(T_2||T_3).$$

5. (Exclusão Mútua I) Para o protocolo de Perterson para dois processos $P_1$ e $P_2$ dado nas aulas, constroí o $PG_1|||PG_2$ e $T(PG_1|||PG_2)$.

6. (Exclusão Mútua II) O programa seguinte é um protocolo de exclusão mútua de dois processos de Pnuelli. Existe uma única variável $s$ que pode tomar os valores 0 ou 1, tendo inicialmente o valor 1. Para além disso cada processo tem uma variável Booleana local $y$ que inicialmente é 0. O programa para o processo $P_i$ para $i = 1, 2$ é o seguinte:

**while true do**
// seccao nao critica
$(y_i, s) := (1, i)$
**wait until** $((y_{1-i} = 0) \vee (s \neq i))$;
// seccao critica
$y_i := 0$

onde $(y_i, s) := (1, i)$ é uma atribui 1 a $y_i$ e $i$ a $s$, num único passo (ação atómica).

(a) Determina o grafo de programa de um processo $P_i$ (considerando localizações diferentes para as secções não crítica e a crítica) e o correspondente sistema de transições.

(b) Constrói o sistema de transições $TS(P_1|||P_2)$ sobre o espaço de estados $(l_i, l_j, y_1, y_2, s)$. Sugestão: Para diminuir o número de estados, sempre que se tenha uma transição para $(l_i, l_j, y_1, y_2, 0)$ considerar o estado $(l_j, l_i, y_2, y_1, 1)$.

(c) Verifica se o algoritmo garante a propriedade da exclusão mútua.

7. Considera o seguinte algoritmo de eleição: Para $n \in \mathbb{N}$, $n$ processos $P_1, \ldots, P_n$ estão organizados num anel topológico onde cada processo está ligado por um canal assíncrono e unidirecional ao seu vizinho (para $1 \leq i < n$, $P_i$ comunica com $P_{i+1}$ e $P_n$ com $P_1$). Cada processo tem um identificador $id(P_i) \in \{1, \ldots, n\}$ que é guardado na variável local $id_i$. Pretende-se eleger o processo com identificador mais alto. Cada processo executa o seguinte algoritmo, onde $m_i$ é uma variável local, inicialmente em 0.

SEND($id_i$); ▷ envia a sua identidade ao processo seguinte
**while true do**
RECEIVE($m_i$)
**if** $(m_i = id_i)$ **then**
EXIT ▷ o processo i é o eleito
**if** $(m_i > id_i)$ **then**
SEND($m_i$);

(a) Modela este protocolo como um sistema de canais com $n$ processos. Sugestão: implementa em Promela para testar

(b) Indica um fragmento de execução inicial de $TS([P_1|P_2|P_3])$ até que um processo entre na instrução SEND no ciclo **while**. Supõe que $id_i = i$, para $1 \leq i \leq 3$.