

Programação Concorrente - Exercícios 5

bissimilaridade Fraca (ou Equivalência Observável) e Congruência Observável

1. Sendo

$$\begin{aligned} Buffer &:= put?.get?.Buffer \\ BufferL &:= put?.pass!.BufferL \\ BufferR &:= pass?.get?.BufferR \end{aligned}$$

Mostra que $(BufferL|BufferR)\{\{pass!, pass?\} \approx (Buffer|Buffer)$.

2. Sendo a bissimilaridade fraca \approx dado por

$$\approx = \bigcup_{R \text{ é bissimulação fraca}} R$$

Mostra que

- (a) A relação \approx é de equivalência.
- (b) A relação \approx é maior bissimulação fraca.
- (c) A bissimilaridade fraca é estritamente mais grosseira que a bissimilaridade forte, i.e $\sim \subseteq \approx$.

3. Seja

$$\begin{aligned} CMB &:= coin?.coffee!.CMB + coin?.CMB \\ CS &:= pub!.coin!.coffee?.CS \\ UniBad &:= (CMB|CS)\{\{coin, coffee\} \end{aligned}$$

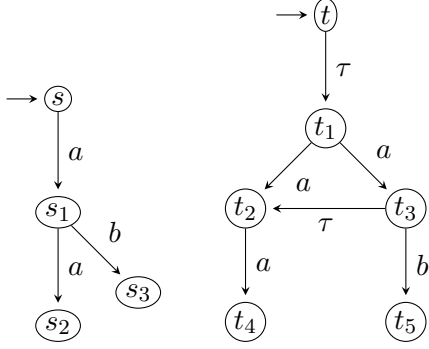
Será que $Spec \approx Unibad$ onde $Spec := pub!.Spec$?

4. Mostra que para qualquer $P, Q \in CCS$ e $\alpha \in Act$ es seguintes equivalências verificam-se (As chamas leis τ de Milner):

$$\begin{aligned} \alpha.\tau.P &\approx \alpha.P \\ P + \tau.P &\approx \tau.P \\ \alpha.(P + \tau.Q) &\approx \alpha.(P + \tau.Q) + \alpha.Q \end{aligned}$$

Nota: constrói bissimulações fracas apropriadas.

5. Considera o seguinte sistema de transições e mostra que $s \not\approx t$ mostrando que um jogo de bissimilaridade fraca onde o atacante tem uma estratégia universal ganhadora, isto é, ganha para todas as possíveis escolhas do defesa.



6. Considera o seguinte protocolo $Protocol := acc?.del!.Protocol$ que corresponde a um canal em que se envia e recebe mensagens (como um buffer duma posição). Seja a seguinte implementação

$$\begin{aligned}
 Send &:= acc?.Sending \\
 Sending &:= send!.Wait \\
 Wait &:= ack?.Send + error?.Sending \\
 Rec &:= trans?.Del \\
 Del &:= del!.Ack \\
 Ack &:= ack!.Rec \\
 Med &:= send?.Med1 \\
 Med1 &:= \tau.Err + trans!.Med \\
 Err &:= error!.Med
 \end{aligned}$$

Mostra que $(Send|Med|Rec) \setminus \{send, error, trans, ack\} \approx Protocol$. Verifica no Pseudo.Com.

7. Mostra que $a.0 + 0 \not\approx a.0 + \tau.0$ e concluí que \approx não é uma congruência em CCS .
8. Sendo \simeq a congruência observável mostra que
- $\tau.a \not\approx a$
 - $P|\tau.Q \not\approx \tau.(P|Q)$
9. A seguir apresentam-se vários algoritmos $MUTEX$ para a exclusão mútua. Para cada um deles:
- (a) Implementa o algoritmo directamente em CCS explicando sucintamente os processos usados e qual o processo que corresponde ao algoritmo (que deverá ter o nome correspondente). Assinala a zona crítica com as ações $enter_i$ e $exit_i$ para $i = 1, 2$.
 - (b) Fornece um ficheiro correspondente para o pseudo.com e testa para verificar (traços aleatórios) que realmente a exclusão mútua ocorre (ou não). Qual o problema caso não?
 - (c) Implementa também em CAAL.
 - (d) Supõe que se modela a entrada e a saída da zona crítica por

$$MutexSpec := enter1.exit1.MutexSpec + enter2.exit2.MutexSpec$$

Será verdade que $MUTEX \approx MutexSpec$?

(e) Implementa o algoritmo na linguagem Pseudo (e/ou CAAL) e testa também a exclusão mútua usando traços aleatórios.

i) Modelar o algoritmo de Peterson com dois processos para a exclusão mútua em CCS.

Para o processo P_i , $j, i = 1, 2$ e $i \neq j$.

```
while true do
  noncritical actions
   $b_i \leftarrow \text{true};$ 
   $k \leftarrow j;$ 
  while  $b_j \wedge k = j$  do
    skip;
  critical actions
   $b_i \leftarrow \text{false}$ 
```

ii) Considera o algoritmo de Hyman para a exclusão mútua. As variáveis b_i são booleanas e k é inteira. Para o processo P_i , $j, i = 1, 2$ e $i \neq j$.

```
while true do
  noncritical actions
   $b_i \leftarrow \text{true};$ 
  while  $k \neq i$  do
    while  $b_j$  do
      skip;
     $k \leftarrow i;$ 
  critical actions;
   $b_i \leftarrow \text{false};$ 
```

iii) Considera o algoritmo de Pnueli para a exclusão mútua. As variáveis y_i são Booleanas e começam em *false* sendo locais. A variável s é partilhada e toma os valores 0 ou 1 começando em 1. Para o processo P_i e $i = 0, 1$:

```
while true do
  noncritical actions
   $y_i \leftarrow \text{true};$ 
   $s \leftarrow i;$ 
  while  $\neg(y_{1-i} = 0 \vee (s \neq i))$  do
    skip;
  critical actions;
   $y_i \leftarrow \text{false};$ 
```