Concurrent Programming - Exercises 5

Weak bisimilarity and observable congruence

1. Let

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

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

2. Given the weak bisimilarity $\approx$

$$\approx = \bigcup_{\text{R weak bisimulation}} R$$

   Show that

   (a) $\approx$ is an equivalence.

   (b) $\approx$ is the largest weak bisimulation.

   (c) The weak bisimilarity is coarsest than the strong bisimilarity, i.e., $\sim \subsetneq \approx$.

3. Let

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

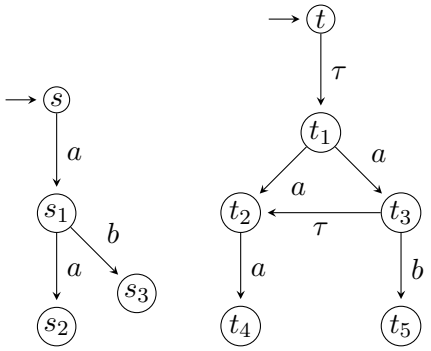
   Is true that $Spec \approx Unibad$ where $Spec := pub!.Spec?$

4. Show that for any $P, Q \in CCS$ and $\alpha \in Act$ the following equivalences hold:

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

   .

5. Considere the following LTS and show that $s \not\approx t$ using a weak bisimulation game.



6. Consider the following protocol $Protocol := acc?.del!.Protocol$ that corresponds to a comunication channel. Given the implemnetation

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

show that $(Send|Med|Rec)\backslash\{send, error, trans, ack\} \approx Protocol$. Check with Pseuco.Com.

7. Show that $a.0 + 0 \not\approx a.0 + \tau.0$ and conclude that $\approx$ is not a congruence in CCS.

8. Let $\simeq$ be the observable congruence; show that:

   - $\tau.a \not\simeq a$
   - $P|\tau.Q \not\simeq \tau.(P|Q)$

9. Consider the following algorithms $MUTEX$ for mutual exclusivity. For each one:
   (a) Write it in CCS. Use $enter_i$ and $exits_i$ for $i = 1, 2$ to indicate the critical region.
   (b) Check if the mutual exclusion seems to be working by randomly navigating through pseuco.com.
   (c) Implement them also in CCS-CAOS.
   (d) Consider now:

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

   Is true that $MUTEX \approx MutexSpec$?

   i) Peterson algorithm.
   For process P_i, $j, i = 1, 2$ and $i \neq j$.

```
while true do
    noncricital actions
    b_i ← true;
    k ← j;
    while b_j ∧ k = j do
        skip;
    end while
    critical actions
    b_i ← false
end while
```

ii) Hyman algorithm. Variables b_i are Boolean and k is an integer. For processes P_i, $j, i = 1, 2$ and $i \neq j$.

while true do
  noncricital actions
  $b_i \leftarrow$ **true;**
  while $k \neq i$ **do**
    while b_j **do**
      skip;
    end while
    $k \leftarrow i$**;**
  end while
  critical actions;
  $b_i \leftarrow$ **false**;
end while

iii) Pnueli algorithm.

Variables y_i are Boolean and start in $false$ being local. The variable s is shared and has value 0 or 1 starting in 1. For process P_i and $i = 0, 1$:

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

iv) Simple algorithm.

Variable y is shared and Boolean, and starts in 0. For process P_i and $i = 0, 1$:

while true do
  noncricital actions
  while $y = 0$ **do**
    skip;
  end while
  $y \leftarrow 0$**;**
  critical actions;
  $y \leftarrow 1$**;**
end while

10. Solve the exercises from the Pseuco.com book:

- `https://book.pseuco.com/#/read/equality/weak-bisimilarity`
- `https://book.pseuco.com/#/read/equality/observation-congruence`
- `https://book.pseuco.com/#/read/equality/workout`