Programação Ciber-Física

[TPC-1] CCS & Equivalências

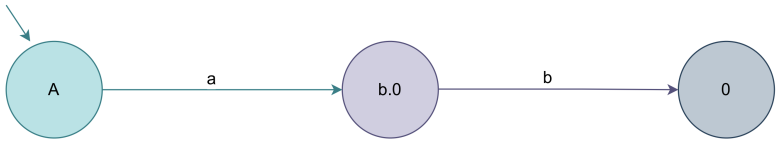
Ana Paula Oliveira Henriques
Mestrado em Engenharia Informática
Universidade do Minho, Departamento de Informática
4710-057 Braga, Portugal
e-mail: pg50196@alunos.uminho.pt

1º Exercício

Para o primeiro exercício prático, foi solicitado o desenho do sistema de transição que traduz cada um dos seguintes processos CCS.

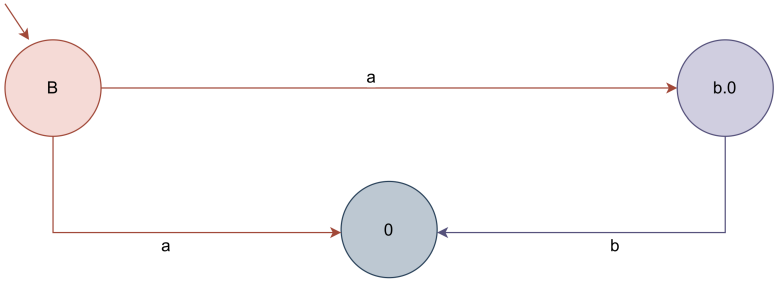
1.1 $A = a.b.0$

O desenho do sistema de transição proposto é:



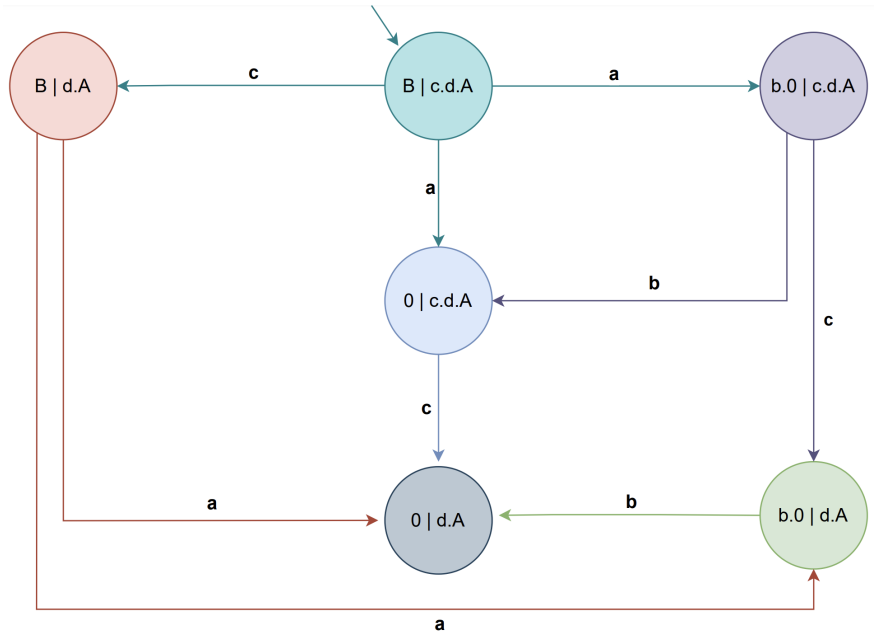
1.2 $B = A + a.0$

Como definido na alínea anterior, $A = a.b.0$ e, como tal, $B = a.b.0 + a.0$, obtendo o consequente desenho do sistema de transição:



1.3 $C = (B \mid c.d.A) \setminus \{d\}$

Sabendo que $B = a.b.0 + a.0$ e que a transição d, a partir de qualquer estado, foi excluída, temos o seguinte sistema de transição:

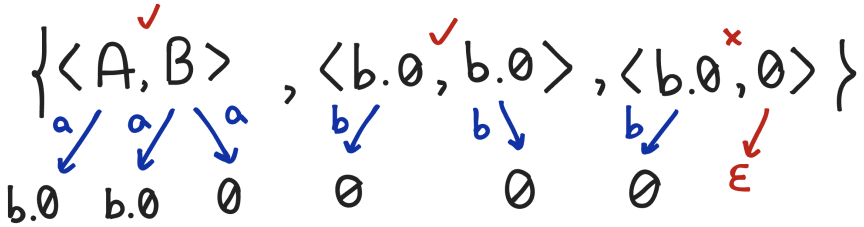


2º Exercício

No segundo exercício prático, é pedido para ter em consideração os processos A e B do exercício anterior, para responder às próximas alíneas.

2.1 $A \lesssim B$?

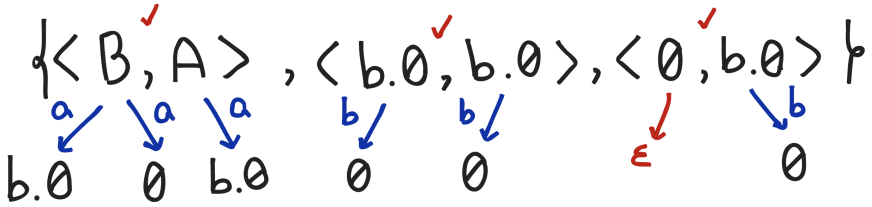
Para saber se B simula A, é necessário verificar se cada transição de B corresponde à transição de A e se essa capacidade é mantida durante todo o tempo de vida do sistema ao qual pertence o estado B.



Como podemos ver, existe uma transição de $b.0$ que o 0 não consegue imitar já que este não faz nada. Como tal, B não é capaz de simular A.

2.2 $B \lesssim A$?

Neste caso, o objetivo é verificar o contrário, ou seja, se A simula B.



Ao contrário da alínea anterior, aqui temos $b.0$ que consegue imitar o comportamento de 0 porque este não faz nada. Logo, A é capaz de simular B.

2.3 $A \sim B$?

Finalmente, devemos analisar se A é bissimilar a B. Ora, esta afirmação é verdadeira se e somente se A conseguir simular B e B conseguir simular A. Como comprovámos nas duas alíneas anteriores, embora A seja capaz de imitar B, B já não pode prometer a mesma coisa. Portanto, A e B não são bissimilares.

3º Exercício [Hard]

Chegando ao exercício 3, o proposto é provar que, para todos os processos CCS P e Q, a afirmação a seguir é verdadeira:

$$P + Q \sim Q + P$$

Comecemos por verificar se, para todos os processos P e Q, $Q + P$ tem a capacidade de simular $P + Q$, sendo esta relação representada por R_1.

$$\mathrm{R}_1 = \{\ \langle P + Q,\ Q + P \rangle \mid P, Q \in \mathbb{P}\ \}$$

Por um lado, sabemos que, se P através de α resultar em P', pela regra *sum-1*, $P + Q$ através de α também resulta em P'. Por outro lado, se Q por α resultar em Q', segundo a regra *sum-2*, $P + Q$ por α também resulta em Q'.

$$\frac{P_1 \xrightarrow{\alpha} P_1'}{P_1 + P_2 \xrightarrow{\alpha} P_1'}\ (\text{sum-1}) \qquad \frac{P_2 \xrightarrow{\alpha} P_2'}{P_1 + P_2 \xrightarrow{\alpha} P_2'}\ (\text{sum-2})$$

O mesmo se aplica a $Q + P$ que, se aplicarmos (1) *sum-1* ou (2) *sum-2* ao se verificar que (1) P por α dá P' ou (2) Q por α dá Q', então sabemos que, também através de α, $Q + P$ resulta em (1) Q' ou (2) P', respetivamente. A relação pela qual $P + Q$ simula $Q + P$ é representada por R_2.

$$\mathrm{R}_2 = \{\ \langle Q + P,\ P + Q\rangle \mid P, Q \in \mathbb{P}\ \}$$

Quanto aos outros processos i do conjunto, podemos seguir o mesmo raciocínio e concluir que $(P_i + Q_i) \sim (Q_i + P_i)$ para qualquer processo i. As relações R_1 e R_2 são reflexivas, transitivas e simétricas. Deste modo, assumindo que R (também reflexiva, simétrica e transitiva) caracteriza a relação de bissimulação, podemos afirmar que $P + Q \sim Q + P$ é válido para qualquer processo P e Q.

$$\begin{gathered} R = \{\ \langle P, P\rangle \mid P \in \mathbb{P}\ \} \\ \vee \\ \{\ \langle P, Q\rangle \mid P, Q \in \mathbb{P}\ \} \\ \vee \\ \{\ \langle Q, P\rangle \mid P, Q \in \mathbb{P}\ \} \\ \vee \\ \{\ \langle Q, Q\rangle \mid Q \in \mathbb{P}\ \} \\ \vee \\ \{\ \langle P + Q,\ Q + P\rangle \mid P, Q \in \mathbb{P}\ \} \\ \vee \\ \{\ \langle Q + P,\ P + Q\rangle \mid P, Q \in \mathbb{P}\ \} \end{gathered}$$

4º Exercício

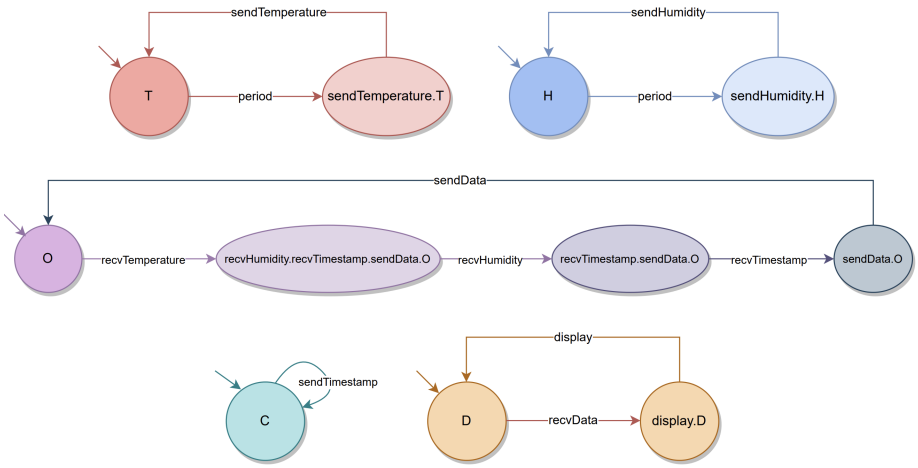
- **T**: Um sensor de temperatura que envia periodicamente um valor de temperatura;
- **H**: Um sensor de humidade que envia periodicamente um valor de humidade;
- **C**: Um relógio que envia um timestamp com a hora atual;
- **O**: Um orquestrador que recebe um valor de temperatura, seguido de um valor de humidade e de um timestamp, e envia este pacote de dados;
- **D**: Uma exibição que recebe dados do orquestrador e display o conteúdo.

4.1 $T\ ?\ H\ ?\ C\ ?\ O\ ?\ D\ ?$

Cada componente pode ser especificado desta forma:

```
T = period.sendTemperature.T
H = period.sendHumidity.H
C = sendTimestamp.C
O = recvTemperature.recvHumidity.recvTimestamp.sendData.O
D = recvData.display.D
```

Assim, o sistema de transição composto por estes 5 componentes está ilustrado na figura abaixo:



4.2 *S ?*

A solução encontrada para alínea é:

$$S = (\ T \mid H \mid C \mid O \mid D\ ) \setminus \{display\}$$

O componente S é, por isso, composto pelos 5 componentes iniciais em paralelo, impondo a sincronização de todas as ações à exceção da *display*.

4.3 *S2 ?*

A posterior sequência de acontecimentos descreve o sistema S2:

1. O sensor de humidade informa o sensor de temperatura, depois
2. O sensor de temperatura informa o timestamp, depois
3. O timestamp envia todos os dados para o display; e finalmente
4. O display avisa o sensor de humidade para reiniciar o processo.

Atendendo que já não existe o componente O (*Orchestrator*), propõe-se a seguinte representação da nova variação S2:

```
H = humidity.T
T = temperature.C
C = timestamp.D
D = display.D + restart.H
```

4.4 [Hard]

Na figura seguinte, temos a especificação do modelo S:

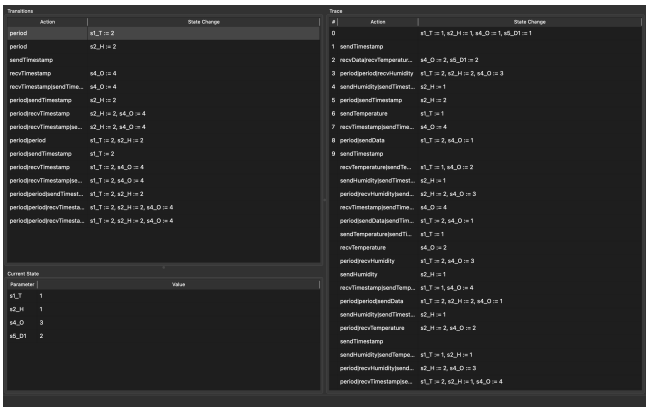
```
act period, sendTemperature, recvTemperature, sendHumidity, recvHumidity, sendTimestamp, recvTimestamp, sendData, recvData, display;

proc T = period.sendTemperature.T;
proc H = period.sendHumidity.H;
proc C = sendTimestamp.C;
proc O = recvTemperature.recvHumidity.recvTimestamp.sendData.O;
proc D = recvData.display.D;
proc SWB = T || H || C || O || D;

proc S = block{display},SWB};

init S;
```

Fazendo uma simulação da tal especificação, obteve-se o posterior *trace* aleatório:



Uma possível propriedade de verificação (de *safety*) é:

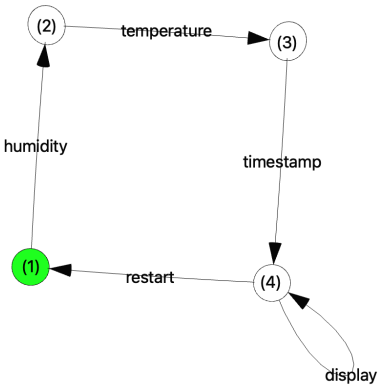
$$\texttt{A[](display => } \neg\texttt{period } /\backslash\ \neg\texttt{sendTemperature } /\backslash\ \neg\texttt{recvTemperature } /\backslash\ \neg\texttt{sendHumidity } /\backslash\ \neg\texttt{recvHumidity } /\backslash\ \neg\texttt{sendTimestamp } /\backslash\ \neg\texttt{recvTimestamp } /\backslash\ \neg\texttt{sendData } /\backslash\ \neg\texttt{recvData}$$

A representação gráfica e a especificação do modelo S2 encontram-se abaixo:

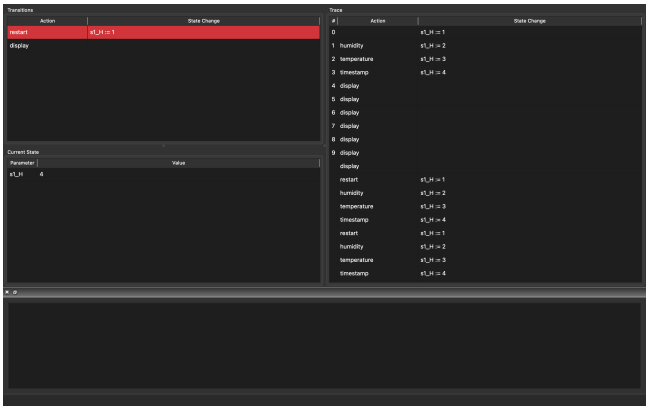
```
act humidity,temperature, timestamp, display, restart;

proc H = humidity.T;
proc T = temperature.C;
proc C = timestamp.D;
proc D = display.D + restart.H;
proc S2 = H + T + C + D;

init D;
```



Sendo feita uma simulação da tal especificação, obteve-se o posterior *trace* aleatório:



Uma possível propriedade de verificação (de *safety*) é:

```
A[]( (humidity => A<>(restart)) /\ restart => A<>(humidity) )
```