Exercise 1

Which of the following expressions are correctly built CCS expressions? Why?
(Assume that A, B are process constants and a, b are channel names.)

1. $a.b.A + B$
2. $(a.Nil + \overline{a}.A) \setminus \{a, b\}$
3. $(a.Nil \,|\, \overline{a}.A) \setminus \{a, \tau\}$
4. $a.B + [a/b]$
5. $\tau.\tau.B + Nil$
6. $(a.B + b.B)[a/b, b/a]$
7. $(a.B + \tau.B)[a/\tau, b/a]$
8. $(a.B + \tau.B)[\tau/a]$
9. $(a.b.A + \overline{a}.Nil) \,|\, B$
10. $(a.b.A + \overline{a}.Nil).B$
11. $(a.b.A + \overline{a}.Nil) + B$
12. $(Nil \,|\, Nil) + Nil$

Solution of Exercise 1

1. **Correct**
2. **Correct**
3. **False**, τ can not be used in a restriction
4. **False**, relabelling can be applied only on a valid process expression
5. **Correct**
6. **Correct**
7. **False**, the relabelling function should satisfy $f(\tau) = \tau$ but here $f(\tau) = a$
8. **False**, actions cannot be relabelled to τ as $\overline{a}$ should be then relabelled to $\overline{\tau}$ and such action does not exist
9. **Correct**

10. **False**, only actions can be used as prefixes
11. **Correct**
12. **Correct**

Exercise 2*

By using the SOS rules for CCS, prove the existence of the following transitions (assume that A is a CCS constant with defining equation $A \overset{\text{def}}{=} b.a.B$):

- $(A \,|\, \overline{b}.Nil) \setminus \{b\} \xrightarrow{\tau} (a.B \,|\, Nil) \setminus \{b\}$
- $(A \,|\, \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{\overline{b}} (A \,|\, a.B)$
- $(A \,|\, \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{\overline{a}} A[a/b]$

Solution of Exercise 2

- Derivation of $(A \,|\, \overline{b}.Nil) \setminus \{b\} \xrightarrow{\tau} (a.B \,|\, Nil) \setminus \{b\}$.

$$
\text{RES}\;\dfrac{\text{COM3}\;\dfrac{\text{CON}\;\dfrac{\text{ACT}\;\dfrac{}{b.a.B \xrightarrow{b} a.B}}{A \xrightarrow{b} a.B}\; A \overset{\text{def}}{=} b.a.B \qquad \text{ACT}\;\dfrac{}{\overline{b}.Nil \xrightarrow{\overline{b}} Nil}}{(A \,|\, \overline{b}.Nil) \xrightarrow{\tau} (a.B \,|\, Nil)}}{(A \,|\, \overline{b}.Nil) \setminus \{b\} \xrightarrow{\tau} (a.B \,|\, Nil) \setminus \{b\}}\; \tau, \overline{\tau} \notin \{b\}
$$

- Derivation of $(A \,|\, \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{\overline{b}} (A \,|\, a.B)$.

$$
\text{SUM1}\;\dfrac{\text{COM2}\;\dfrac{\text{ACT}\;\dfrac{}{\overline{b}.a.B \xrightarrow{\overline{b}} a.B}}{A|\,\overline{b}.a.B \xrightarrow{\overline{b}} A|\,a.B}}{(A \,|\, \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{\overline{b}} (A \,|\, a.B)}
$$

- Derivation of $(A \mid \bar{b}.a.B) + (\bar{b}.A)[a/b] \xrightarrow{\bar{a}} A[a/b]$.

$$\text{SUM2}\ \dfrac{\text{REL}\ \dfrac{\text{ACT}\ \dfrac{}{\bar{b}.A \xrightarrow{\bar{b}} A}}{(\bar{b}.A)[a/b] \xrightarrow{\bar{a}} A[a/b]}}{(A \mid \bar{b}.a.B) + (\bar{b}.A)[a/b] \xrightarrow{\bar{a}} A[a/b]}$$

Exercise 3*

Consider the following CCS defining equations:

$$CM \stackrel{\text{def}}{=} coin.\overline{coffee}.CM$$

$$CS \stackrel{\text{def}}{=} \overline{pub}.\overline{coin}.coffee.CS$$

$$Uni \stackrel{\text{def}}{=} (CM \mid CS) \setminus \{coin, coffee\}$$

Use the rules of the SOS semantics for CCS to derive the labelled transition system for the process Uni defined above. The proofs can be omitted and a drawing of the resulting LTS is enough.

Solution of Exercise 3

LTS for the process $Uni \stackrel{\text{def}}{=} (CM \mid CS) \setminus \{coin, coffee\}$.

$$\begin{tikzcd}
Uni \arrow[d, "\overline{pub}"] \\
(CM \mid \overline{coin}.coffee.CS) \setminus \{coin, coffee\} \arrow[d, "\tau"] \\
(\overline{coffee}.CM \mid coffee.CS) \setminus \{coin, coffee\} \arrow[d, "\tau"] \\
(CM \mid CS) \setminus \{coin, coffee\} \arrow[uu, bend left=60, "\overline{pub}"]
\end{tikzcd}$$

Exercise 4

Draw (part of) the labelled transition system for the process constant A defined by

$$A \stackrel{\text{def}}{=} (a.A) \setminus \{b\}.$$

The resulting LTS should have infinitely many reachable states. Can you think of a CCS term that generates a finite LTS and intuitively has the same behaviour as A?

Solution of Exercise 4

Transition system for $A \stackrel{\text{def}}{=} (a.A) \setminus \{b\}$.

$$A \xrightarrow{\quad a \quad} A \setminus \{b\} \xrightarrow{\quad a \quad} (A \setminus \{b\}) \setminus \{b\} \xrightarrow{\quad a \quad} \big((A \setminus \{b\}) \setminus \{b\}\big) \setminus \{b\} \xrightarrow{\quad a \quad} \cdots$$

One solution could be the CCS defining equation $B \stackrel{\text{def}}{=} a.B$ which generates a finite LTS with (intuitively) the same behavior as A.

Exercise 5

Let us consider the following CCS definition of a coffee machine.

$$\text{CM} \stackrel{\text{def}}{=} coin.\overline{coffee}.\text{CM}$$

- Give a CCS process which describes a coffee machine that may behave like CM but may also steal the money it receives and fail at any time.

Solution of Exercise 5

For example like this:

$$\text{CM'} \stackrel{\text{def}}{=} coin.(\overline{coffee}.\text{CM'} + \text{CM'} + \overline{fail}.Nil) + \overline{fail}.Nil$$

Exercise 6 (optional)

1. Draw the transition graph for the process name Mutex_1 whose behaviour is given by the following defining equations.

$$\begin{aligned}
\text{Mutex}_1 &\stackrel{\text{def}}{=} (\text{User} \mid \text{Sem}) \setminus \{p, v\} \\
\text{User} &\stackrel{\text{def}}{=} \bar{p}.\text{enter}.\text{exit}.\bar{v}.\text{User} \\
\text{Sem} &\stackrel{\text{def}}{=} p.v.\text{Sem}
\end{aligned}$$

2. Draw the transition graph for the process name Mutex_2 whose behaviour is given by the defining equation

$$\text{Mutex}_2 \stackrel{\text{def}}{=} ((\text{User}|\text{Sem})|\text{User}) \setminus \{p, v\}$$

where User and Sem are defined as before. Would the behaviour of the process change if User was defined as

$$\text{User} \stackrel{\text{def}}{=} \bar{p}.\text{enter}.\bar{v}.\text{exit}.\text{User} \ \ ?$$

3. Draw the transition graph for the process name FMutex whose behaviour is given by the defining equation

$$\text{FMutex} \stackrel{\text{def}}{=} ((\text{User} \mid \text{Sem}) \mid \text{FUser}) \setminus \{p, v\}$$

where User and Sem are defined as before, and the behaviour of FUser is given by the defining equation

$$\text{FUser} \stackrel{\text{def}}{=} \bar{p}.\text{enter}.(\text{exit}.\bar{v}.\text{FUser} + \text{exit}.\bar{v}.Nil)$$

Do you think that Mutex_2 and FMutex are offering the same behaviour? Can you argue informally for your answer?