Tutorial 1 - Solutions

Exercise 1

The possible values stored in the variable x are 1 or 6. Note that there are *four* different executions.

Exercise 2

Since $R \cup E$ is indeed a reflexive relation and contains R, it suffices to show that $R \cup E$ is minimal. Suppose R'' is a binary relation such that (i) $R \subseteq R''$ and (ii) R'' is reflexive. We will show that $R \cup E \subseteq R''$. Let $(x, y) \in R \cup E$. Then at least one of the following must hold. Either $(x, y) \in R$ or $(x, y) \in E$. If $(x, y) \in R$, then $(x, y) \in R''$ because of (i). If $(x, y) \in E$, then $x = y$ so $(x, y) \in R''$ because of (ii). Hence $R \cup E \subseteq R''$.

Exercise 3

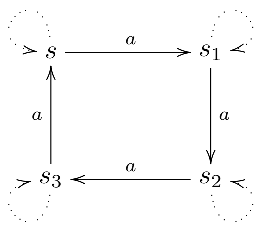
- Let $(x, y) \in R \cup R^{-1}$. If $(x, y) \in R$ then by the definition of R^{-1}, $(y, x) \in R^{-1}$, which establishes that $(y, x) \in R \cup R^{-1}$. If $(x, y) \in R^{-1}$ then by the definition of R^{-1}, $(y, x) \in R$, which establishes that $(y, x) \in R \cup R^{-1}$.
- Arguing that $R \cup R^{-1}$ is a symmetric closure of R is done similarly to Exercise 2. Let R'' be a binary relation that contains R and is symmetric. Then it is easy to see that $R \cup R^{-1} \subseteq R''$ by arguing as in Exercise 2.

Exercise 4*

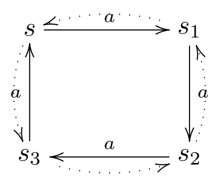
-
  - $\mathit{Proc} = \{s, s_1, s_2, s_3\}$
  - $\mathit{Act} = \{a\}$
  - $\xrightarrow{a} = \{(s, s_1), (s_1, s_2), (s_2, s_3), (s_3, s)\}$

  In the following three diagrams the dotted arrows indicate the pairs of states that must be added to the relation $\xrightarrow{a}$ in order to obtain the indicated closures.

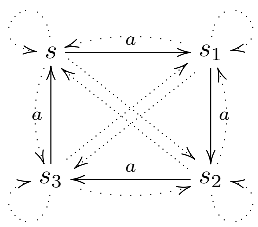
- Reflexive closure



- Symmetric closure



- Transitive closure



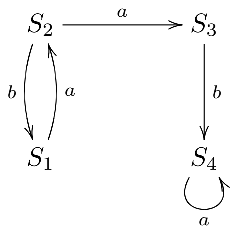
Example 5

For example like this:

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

Example 6

- Yes, there are at most $|Proc|^2$ elements in $\xrightarrow{a}$.
- An LTS with four states and two actions (you probably have a different one).



- CCS description of the LTS

$$\begin{aligned}
S_1 &\stackrel{\text{def}}{=} a.S_2 \\
S_2 &\stackrel{\text{def}}{=} a.S_3 + b.S_1 \\
S_3 &\stackrel{\text{def}}{=} b.S_4 \\
S_4 &\stackrel{\text{def}}{=} a.S_4
\end{aligned}$$

- Describing a finite LTS using CCS can be done as follows. For each $S \in Proc$ add a (possibly recursive) definition of a new process constant S as follows:
  - If $S \not\longrightarrow$ then add the defining equation $S \stackrel{\text{def}}{=} Nil$, otherwise
  - let $S \xrightarrow{a_1} S_1,\ S \xrightarrow{a_2} S_2,\ \ldots,\ S \xrightarrow{a_n} S_n$ be all the transitions going out of S. Then add the following defining equation $S \stackrel{\text{def}}{=} a_1.S_1 + a_2.S_2 + \cdots + a_n.S_n$.