Exam 2 Testing Techniques (271001) — 14 April 2008

To make this exam, you are allowed to have a copy of the lecture notes and the slides. Nothing else. Indicate your name on each separate page that you hand in.

The division of the points is as follows:

exercise 1: 5
exercise 2: 25
exercise 3: 20 We wish you a lot of success!
exercise 4: 30
exercise 5: 20

Exercise 1 (Security testing) In his lecture, Marc Witteman discussed several security attacks. Give two arguments why these attacks can be considered as testing. Also explain a difference between a testing (in the sense of this course) and a security testing.

Exercise 2 (Testing preorders) Consider the LTSs Q_1, Q_2, Q_3, Q_4 below. Their labels sets are $\{a, b, c, d, e, f\}$. Recall that the preorders that we consider are: trace inclusion, completed trace inclusion, testing preorder $\leq_{te}$ and refusal preorder $\leq_{rf}$.

1. Fill in the following table. In each entry (i, j) of the table (i are rows, j are colums), write the strongest preorder that holds between processes (Q_i, Q_j) (i.e. $Q_i \leq Q_j$). Note:
   - Example: if we have trace inclusion between Q_i and Q_j, but not completed trace inclusion (and hence they are not in the testing preorder and not refusal preorder), then write "trace inclusion" at table position (Q_i, Q_j).
   - Only fill in the blank positions in the table.
   - If they are not included in any preorder, write a "x".
   - Copy the table on your answer sheet.

voor dezelfde matrix -

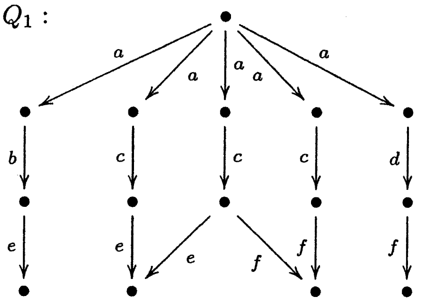
2. For all process pairs (Q_i, Q_j), consider the weakest preorder such that (Q_i, Q_j) are not in this preorder. Provide a distinguishing behavior (i.e. trace, completed trace, refusal pair or refusal trace) that shows that they are not in this preorder.

j

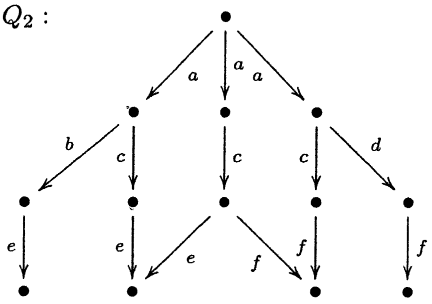
i

	Q_1	Q_2	Q_3	Q_4
Q_1	-			
Q_2	-	-		
Q_3	-	-	-	

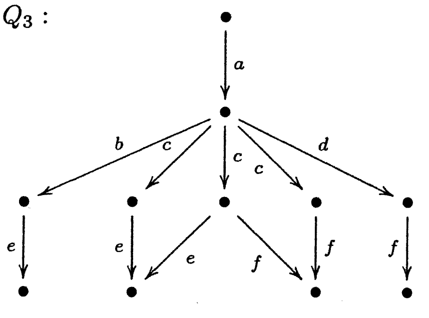
Q_1 :



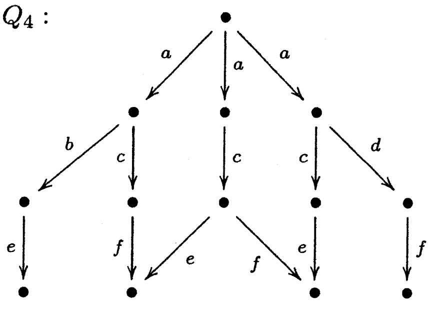
Q_2 :



Q_3 :

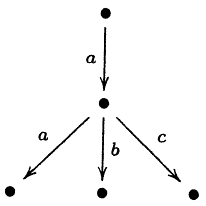


Q_4 :



Exercise 3 (Testing Equivalences and preorders) In the following exercise, you have to create LTSs over label set $\{a, b, c\}$ with the following properties:

1. p_3 and q_3 are trace equivalent, but not testing equivalent and a distinguishing environment is given by

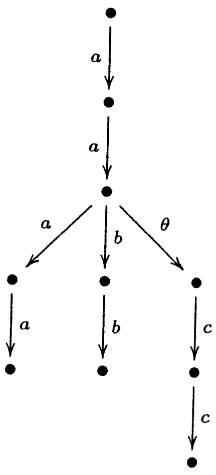


2. p_5 and q_5 are trace equivalent, but not testing equivalent, and

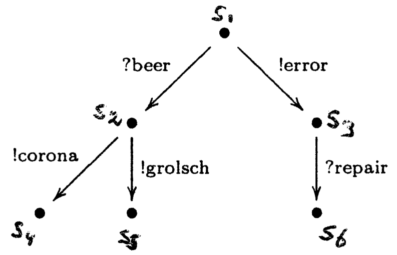
$$p_5 \textbf{ after } aaa \textbf{ refuses } \{a, b, c\}$$
$$q_5 \textbf{ after } aaa \textbf{ refuses } \{a, b\}$$
$$q_5 \textbf{ after } aaa \textbf{ refuses } \{b, c\}$$
$$q_5 \textbf{ after } aaa \textbf{ refuses } \{a, c\}$$

3. p_6 and q_6 are testing equivalent, but not refusal equivalent and $a\{b, c\}a\{b, c\}$ is a failure trace of p_6, but not of q_6.

4. p_7 and q_7 are testing equivalent, but not refusal equivalent, and a distinguishing environment is given by

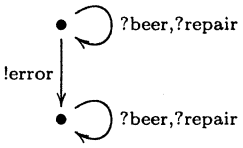


Exercise 4 (Test derivation for ioco) A company that produces beer tenders provides the following specification S, with $L_I = \{?\text{beer}, ?\text{repair}\}$ and $L_U = \{!\text{grolsch}, !\text{corona}, !\text{error}\}$.

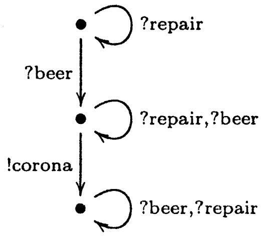


1. Determine the quiescent states in S and construct the suspension automaton (i.e. add δ-loops where needed).
2. Give test suites T_1, T_2, T_3 for S with the following properties. Use the ioco test derivation algorithm whenever possible.
   (a) T_1 is sound, but not complete[1].
   (b) T_2 is not sound.
   (c) T_3 is sound and complete.
3. Which of the following implementations i_1, i_2 and i_3 conform to S w.r.t. ioco? For those that are incorrect, provide a test case that exhibits the error.

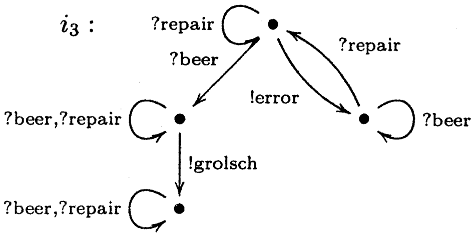
i_1 :



i_2 :



i_3 :



4. Give an implementation IOLTS i_4 that is ioco-correct with respect to S and which is able to provide a corona after a repair.

[1] Recall that the lecture notes uses the word "exhaustive" for this concept

Exercise 5 (True or false?) Are the following statements true? If so, give a proof otherwise give a counter example. Here, p, q are IOLTSs with the same input and output labels, p is input-enabled. Action a is an output action of p and q. Do we have

1. $\mathit{Straces}(p) = \mathit{Straces}(q)$, then $\mathit{qtraces}(p) = \mathit{qtraces}(q)$
2. Assume that q contains only output actions. Then p **ioco** $q \implies p \leq_{tr} q$
3. Assume that q is input-enabled. Then $p \leq_{te} q \implies p$ **ioco** q
4. If p **ioco** q, then (**hide** a **in** p) **ioco** (**hide** a **in** q).