The Faculty of Science & Engineering - Computer Science

CSC385 - Modelling and verification Techniques

Exam Session: May/June 2022

Time Allowed: 2 Hours

Do NOT turn over your question paper until instructed to do so.

Exam Paper Information

None

Special Instructions

None

Specific Items

Dictionaries - Candidates may only refer to the English and Welsh Dictionaries available

Calculators - Candidates may NOT use a calculator

Open Book - This is NOT an Open Book Exam

Contact Details

Originator: ***Ulrich Berger***

Contact Number: ***3380*** **or** ***Contact the Exams Office***

Question 1

The following Haskell program concatenates two lists:

```
conc []       ys = ys
conc (x : xs) ys = x : (conc xs ys)
```

Explanation of the program for those who are not familiar with Haskell:

- The first line says that concatenating the empty list with a list `ys` yields `ys`.
- The second line says that the concatenation of a list with head `x` and tail `xs` with a list `ys` yields a list with head `x` and a tail obtained by (recursively) concatenating `xs` and `ys`.

Briefly describe what it means to verify this program.

In particular, state the correctness property and indicate what method you would use to prove it.

You do not need to carry out the proof.

You may express the correctness property using the functions

- `length` where `length xs` computes the length of the list `xs`,
- and `xs !! i` which computes the `i`th element of the list `xs`.

[5 marks]

Question 2

Consider the following labelled transition systems, both over the alphabet $\{a, b\}$:

$$\{P \xrightarrow{a} P, P \xrightarrow{b} P, P \xrightarrow{a} Q, Q \xrightarrow{b} Q, Q \xrightarrow{b} P\}$$

$$\{R \xrightarrow{a} R, R \xrightarrow{b} R\}$$

(a) For each of the processes P and R say whether it is deadlock free.

[2 marks]

(b) For each of the processes P and R say whether it is deterministic.

[2 marks]

(c) Are P and R trace equivalent? Justify your answer.

[4 marks]

(d) Decide whether the states P and Q are bisimilar, by either giving a winning strategy for the defender, or else a winning strategy for the attacker in the bisimulation game.

[6 marks]

Question 3

Suppose you are given a tool that can test determinism, deadlock freeness and bisimilarity of processes. Furthermore, suppose you are given two processes P and Q that are known to be deterministic.
Describe how the tool can be used to test whether P and Q are trace equivalent. Say which theoretical result makes this possible.

[5 marks (3 for description, 2 for theoretical result)]

Question 4

Consider the following processes describing a ticket machine and a customer:

The ticket machine, call it M, can do an action r (for 'ready') after which it is in the state MR, where it accepts a coin c (that is, it can do an action c) after which it is back at state M.

Alternatively, M can print a ticket (do an action p) after which it is in the state MP. Then it issues a ticket (action t) after which it is back at state M.

The customer, call her C, can pay with a coin (that is, do a c action) after which she is in the state CC. In the state CC she can take a ticket (do action t) after which she is back at state C.

(a) Write down the labelled transition systems of the ticket machine and the customer, either by giving the sets of transitions, or by drawing the corresponding graphs.

[2 marks]

(b) Write down CSP processes that define the ticket machine and the customer.

[2 marks]

(c) Write down a CSP process CM that describes the customer interacting with the ticket machine.

[2 marks]

(d) Decide whether the process CM is deadlock free. Justify your answer.

[2 marks]

Question 5

During 30 days of lockdown there are no lectures but Bob has to perform five activities every day: cooking (c), eating (r), walking the dog (d), working on his project (p), and phoning his girlfriend (g).

He can perform these activities in any order, except that eating must immediately follow cooking.

Describe Bob's activities during lockdown as a CSP process. After lockdown has ended, the process shall successfully terminate.

Minor syntax errors will not be penalised.

[10 marks]

Question 6

In the labs, we modelled the Needham-Schroeder-Lowe authentication protocol.

Carla did pretty well but she got the 6th and 8th task slightly wrong.

Explain what is wrong with Carla's solution, correct it and explain why yours is correct.

```
channel send, receive : Message
channel know : Nonce

HU(u) = UA(u) [] UP(u)

-- Task 1
-- Define a process UA(u) ("User Active") that models a honest user u
-- who plays the role of the initiator of the communication.

UA(u) = [] v : allUsersBut(u) , n : {N.u.v}@
        send.E1.v.u.n     ->
        receive.E2.u.n?m  ->
        send.E3.v.m       ->
        HU(u)

-- Task 2
-- Define a process UP(u) ("User Passive") that models a honest
-- user u who plays the role of the responder to a
-- communication request.

UP(u) = [] v : allUsersBut(u), m : {N.u.v} @
        receive.E1.u.v?n  ->
        send.E2.v.n.m  ->
        receive.E3.u.m  ->
        HU(u)

-- Task 3
-- Define a process Knowledge(lns), where lns is a parameter
-- ranging over lists of nonces, that does all the actions
-- know.n, for n is in lns, and then successfully terminates.

Knowledge(lns) = if null(lns)
                 then SKIP
                 else know.head(lns) -> Knowledge(tail(lns))

-- Task 4
-- Model the intruder.

Intruder = IA [] IP   -- Intruder without memory

-- The active intruder makes up any message and sends it.

IA = send ? msg : genMessagesI({}) ->
     Intruder

-- The passive intruder receives any message, and sends any message
-- he can compose by analysing the received message.

-- Before recursing to Intruder, report the nonces the intruder has learnt
-- from the received message.

IP = receive ? msg ->
      ( Knowledge(learn(msg)) ;
        send ? msg' : genMessagesI({msg}) ->
        Intruder )

-- The process describing user u
-- where u may be the intruder or an honest user:

U(u) = if u == I then Intruder else HU(u)

-- Task 5
-- Define a process USERS that models all users
-- running independently in parallel.

USERS = ||| u : allUsers @ U(u)
```

```
-- Task 6
-- Model the environment. All the environment does, is to transfer
-- any message from the send channel to the receive channel.

ENV = send ? msg -> receive . msg -> ENV

-- Model the process SYSTEM of all users interacting with
-- the environment.

System = USERS [| {| send, receive, know |} |] ENV

-- Task 7
-- Define a specification SPEC that can be used to check whether
-- there is an attack, that is, whether the intruder is able to
-- learn a nonce he is not supposed to know.

SPEC = know ? n : noncesI -> SPEC

-- Task 8
-- Do the model checking. Analyse the trace that leads to failure
-- and construct the attack from that.

assert SPEC [T= System
```

[8 marks (2 for each correction, 2 for each explanation)]

End of Paper