The Faculty of Science & Engineering - Computer Science

CSC385 - Modelling and Verification Techniques

Exam Session: January 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: ***Contact the Exams Office***

Question 1

In the modelling and analysis of processes two crucial questions are:

- When are two processes to be considered equal?
- When does a process (playing the rôle of the implementation) refine another (playing the rôle of the specification)?

(a) Explain (that is, give the definitions of) two important notions of equivalence between processes and state how they are related to each other.

You may cite theorems discussed in class and use examples to underpin your answers.

[8 marks]

(b) Explain the notion of refinement we worked with in the lab classes and illustrate its use by an example.

[4 marks]

Question 2

Consider the following CSP process definition of a robot which repeatedly can move to the left or to the right or pick up or drop some item.

```
min = 0
max = 5
Range = {min..max-1}
left(x) = if x>0 then x-1 else x
right(x) = if x+1<max then x+1 else x

datatype Direction = L | R

channel move : Direction
channel pick, drop
channel position : Range

Robot(x) = move.L     -> Robot(left(x))  []
           move.R     -> Robot(right(x)) []
           position.x -> Robot(x)        []
           pick       -> Robot(x)        []
           drop       -> Robot(x)
```

The behaviour of the robot should be constrained in such a way, that it can hold at most `maxload` items where `maxload` is some fixed natural number (defined as a constant). The robot should be able to drop an item only if holds at least one.

Define a process `Watchload(n)` (with a parameter `n` for the number of items the robot currently holds) and a synchronisation set `X` such that the process defined by

```
Robot(min) [| X |] Watchload(0)
```

behaves as specified above, assuming it starts at position `min` with an initial load `0`.

Minor syntax errors will not be penalised.

[7 marks]

Question 3

A Halloween sweet packing machine shall fill 20 bags with 5 different kinds of sweets: A chocolate bar, a gummy bear, a candy, a Welsh cake and a raisin. Each bag shall contain exactly one sweet of each kind but the order in which the sweets are put into the bags doesn't matter.

In each work cycle a bag is opened, then the sweets are filled in, in any order, then the bag is closed and the next cycle begins. When all bags are full, the process shall stop.

Define a CSP process that models all possible behaviours of the machine.

Use the following declarations:

```
Bag = {1..20}

datatype Sweet = Choc | Bear | Candy | Cake | Raisin

channel open, close: Bag
channel fill: Bag.Sweet
```

Minor syntax errors will not be penalised.

[8 marks]

Question 4

In the labs, we modelled an authentication protocol which is informally described by the following rules:

(1) $A \longrightarrow B : \{\mathrm{N}_A, A\}_{\mathrm{pk}_B}$

(2) $B \longrightarrow A : \{\mathrm{N}_A, \mathrm{N}_B\}_{\mathrm{pk}_A}$

(3) $A \longrightarrow B : \{\mathrm{N}_B\}_{\mathrm{pk}_B}$

(a) Referring to the description above, explain this protocol. What is the purpose of the protocol? Which cryptographic mechanism is it based on?

[2 marks]

(b) We used the following encoding of messages:

`E1.v.u.n` is the initiating message encrypted with `v`'s public key. The encrypted content is `u.n` where `u` is the name of the initiating user, `v` is the name of the responding user (the person `u` wants to talk to), and `n` is a nonce created by `u` intended for `v`.

`E2.u.n.m` is the responding message encrypted with `u`'s public key. The encrypted content is `n.m` where `n` is the nonce received by the responder and `m` is the nonce generated by the responder and intended for `u`.

`E3.v.n` is the final acknowledging message with `v`'s public key. The encrypted content is `n`.

A nonce created by `u` and intended for `v` is denoted `N.u.v`.

One lab task was about modelling a honest user (`HU(u)`) in their rôle as an initiator of the communication (called A in the informal description and called 'active user', `UA(u)`, in the encoding).

Explain what is wrong with the following solution and correct it:

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

Explain why your solution is correct.

[3 marks]

(c) Describe an attack on this protocol using a similar style of rules as above.

[2 marks]

(d) Describe how the protocol can be amended to prevent this attack.

[1 mark]

(e) We modelled the amended protocol by a process `System`, specified its correctness by a process `SPEC`, and checked its correctness by a suitable assertion:

```
SYSTEM = ENV [| {| send, receive |} |] USERS

SPEC = know ? n : noncesI -> SPEC

assert SPEC [T= System \ {| send, receive |}
```

In the following, 'explain' means giving an informal explanation showing an understanding of the modelling.

(i) Explain what the process `System` means (describe its components and explain the way they are connected).

(ii) Explain what the assertion means. Explain all parts of it, in particular, explain the meaning of `\ {| send, receive |}` and why this part is necessary.

(iii) Which correctness property is guaranteed if the execution of this assertion successfully terminates (that is, with the result "Passed")?

[6 marks (2 marks each)]

Question 5

(a) Describe the main elements of program extraction from constructive proofs. Your description should include:

(i) A short description of the principle program extraction is based on and an explanation of the difference between constructive and classical logic.

(ii) Examples illustrating the correspondence between formulas and proof rules on the one hand and types and program constructs on the other hand.

(iii) A simple example of a problem which program extraction can be applied to.

[6 marks (2 marks each)]

(b) Briefly discuss the advantages and disadvantages of program extraction compared with program verification. **[3 marks]**.

End of Paper