Swansea University
Department of Computer Science

U Berger
October 10, 2022

CSC385/CSCM85 Modelling and Verification Techniques

Lab 1

Choice and parallelism

The two tasks below may be done in groups of up to four students. In order to obtain full marks, all tasks must be completed. For fewer or incompletely accomplished tasks, partial marks will be awarded. Marks will be awarded only for work that is demonstrated during the lab session.

This lab assignment is worth 20 marks out of 100 lab marks. It needs to be completed and assessed on the 11th or the 14th of October 2022.

Task 1. (Understanding recursion and replicated choice).

We work with `channel c : Int`, that is, the alphabet (set of possible events) we are considering is $\{c.i | i \text{ an integer}\}$.

Write a parameterised process Perm(n) such that for every natural number n, Perm(n) has as maximal traces[1] exactly the words of the form $\langle c.f(1), c.f(2), ..., c.f(n)\rangle$ where $f : \{1, ..., n\} \to \{1, ..., n\}$ is a permutation. This means that the set of maximal traces of Perm(n) must must consist of all possible rearrangements of the word $\langle c.1, \ldots, c.n\rangle$.

For example, Perm(3) should have as maximal traces the words

$$\langle c.1, c.2, c.3\rangle, \langle c.1, c.3, c.2\rangle, \langle c.2, c.1, c.3\rangle, \langle c.2, c.3, c.1\rangle, \langle c.3, c.2, c.1\rangle, \langle c.3, c.1, c.2\rangle$$

Demonstrate the correctness of your solution by displaying the graphs of, say, Perm(3), Perm(4), Perm(5).

Hint: Define Perm(n) with the help of another process that is parameterised by a finite set of integers. This helper process should, for every finite set $S = \{k_1, ..., k_m\}$ of integers, have as maximal traces exactly the words of the form $\langle c.f(k_1), c.f(k_2), ..., c.f(k_m)\rangle$ where $f : S \to S$ is a permutation. This means that the set of maximal traces of the helper process should consist of all possible rearrangements of the word $\langle c.k_1, \ldots, c.k_m\rangle$.

10 marks

[1]The notion of a maximal trace is explained in detail at the end of this lab sheet.

Task 2. The point of this task is to study a process with a 'hidden' deadlock and to practise the use of the generalised parallel operator as well assertions.

Declare `channel a, b, d, e` and implement the processes

$$
\begin{aligned}
P &= a \to e \to P \,\square\, b \to d \to P \\
Q &= e \to d \to Q \\
R &= P \,[\![\{d, e\}]\!]\, Q
\end{aligned}
$$

The expression $P \,[\![\{d, e\}]\!]\, Q$ is written in machine syntax as

```
P [| {| d, e |} |] Q
```

Explore the behaviours of the processes P, Q, R using `:probe` and `:graph`.

Furthermore, use FDR to check whether the processes P, Q, R are

(a) deadlock free (type `assert P :[deadlock free]` in FDR's terminal or in your file `lab1.csp`, etc.),

(b) deterministic (type `assert P :[deterministic]`, etc.).

(c) Modify one of the processes P, Q or R, by changing one character, such the result of testing determinacy changes. Confirm the change using FDR.

10 marks

Detailed explanation of the maximal traces of a process. A trace of a process P is called *maximal* if it cannot be extended to a longer trace of P. For example, if

```
P = a -> b -> STOP
```

then the set of all traces of P is $\{\langle\rangle, \langle a\rangle, \langle a, b\rangle\}$, and $\langle a, b\rangle$ is the only maximal trace of P. For example, $\langle a\rangle$ is also a trace of P but not a maximal one, since it can be extended to the longer trace $\langle a, b\rangle$.