CS3211 Parallel and Concurrent Programming – Guidelines for tutorial 1 (25-29 Jan 2010)

Possible flow of any tutorial:

i) **Discussion of any topics in past lectures which students found hard (you can go through some lecture slides, or prepare your own explanation – up to you).**
ii) **Discussion of some exercises on the recent lectures. These exercises are given on-the-spot and discussed. In other words, they are *not* for grading the students – they are for online discussion and forcing the students to think clearly on the concepts.**
iii) **Any questions the students have about assignments (not applicable for Tut 1)**

Content:

The general idea of concurrency, and Promela – a modeling language for concurrent systems (the first two lectures in cs3211 – check Lesson Plan and Workbin in IVLE please).

Materials:

Lecture slides for the first two lectures.

Readings given in IVLE E-reserves (only 1 download per person allowed for copyright reasons, so once you click it please remember to save the file).

Possible material to be revised in tutorial:

Channel communication in Promela

Solutions to the Dining philosopher's problem and their encoding in Promela.

Sample Exercises:

[Please conduct these as an interactive discussion, rather than an evaluation. Please also make it clear to the students that they are not being evaluated for their performance in these exercises, so that they are not afraid to make mistakes while answering.]

1. Consider the following Promela specification of a simple producer and consumer. Assume that both of them access a buffer which serves as a shared data structure.

```
toktype = {P, C};
toktype turn = P;

active proctype producer()
{
  do
  :: (turn == P) -> /*produce 1 item*/
                    turn = C
  od
}
```

```
active proctype consumer()
{
  do
  :: (turn == C) -> /* consume 1 item */
                    turn = P
  od
}
```

If there is only one producer process and one consumer process, will an interleaved execution violate mutual exclusion of access to

the shared buffer ? To answer this question, construct the global state transition system by hand after assigning labels to the control locations of the processes.

[Note: This is again to challenge the students to understand well the semantics of the concurrent programming language. When you introduce the question – you need to tell them what are the states of the global transition system. In general, a global state is a valuation of --- the program counters of the processes and the variables. In this case, a global state is a valuation of --- (program counter of producer, program counter of consumer, value of turn)

End-of-note]

2. Suppose we instantiate the producer and consumer modules to concrete processes in SPIN as follows.

```
init{
  atomic{ run producer(); run producer(); run consumer(); run consumer(); }
}
```

Can any two of the above SPIN processes be allowed to access the shared buffer at the same time – that is, violation of mutual exclusion? Explain your answer.

[Note: This is, in fact, possible since two producers might access the shared buffer at the same time. The turn only distinguishes between producers and consumers.

End-of-note]