CS4710: Model-Driven Software Development, Spring 2020
Computer Science Department
Michigan Technological University

SPIN Group Assignment #1: Parallel Swap
Deadline: February 23, 2020 at mid-night

Problem description: Consider a shared memory program with *N* processes that have access to an array A[] of integers of size *N*. You should initialize the array A[] with *distinct* non-negative integer values. Each process can read and write to any array cell.

Specifications: Each process P_i should randomly pick a value *j* modulo *N* and then swap A[i] with A[j]. After the swap, P_i terminates. After the termination of all processes, the array A[] must contain a permutation of its initial values. Notice that, A[] must not contain duplicate, nor should there be a missing value in A[] after termination. The safety and liveness requirements of the program are as follows:

- *Liveness*: All processes must eventually terminate.
- *Safety* (*data race-freedom*): No two processes should simultaneously access the same array cell where one of them performs a write operation. In this problem, data race-freedom would imply no duplicate/missing values.

Single Instruction Multiple Data (SIMD) model:
There are different models of parallel computing. One approach is SIMD where a single piece of code is replicated in different threads/processes, but each process may run the code on different input data. Each process may decide which part of the code to execute based on its own process identifier (a.k.a. *pid*). For example, in a system of N processes, we may decide to assign different tasks to even and odd processes. Such a system can be captured as a *parameterized* SIMD process as follows:

```
active [N] proctype myProc(){
        if
                :: (_pid % 2 == 0) ; { // here you write the code for even processes }
                :: (_pid % 2 != 0) ; { // here you write the code for odd processes }
        fi;
}
```

Notice that the code of this proctype is parameterized in terms of process ID; i.e., _pid.

Deliverables: You must turn in the following material:

1. **Promela model**: You should design a model of the Parallel Swap in Promela. Your solution must be in the SIMD (Single Instruction Multiple Data) model of parallelism where all processes have a symmetric code that is parameterized in terms of _pid. (40 points)
2. **Property specification in LTL and their verification**: You should specify the safety and liveness properties as LTL expressions, and verify them using SPIN. (30 points)
3. **Written report**: Your team must turn in a written report that explains your Promela model, your LTL expressions and your verification activities. (30 points)

Notes:

1. *The part of the code that performs the swap operation must NOT be in an atomic block. You are allowed to introduce new variables if needed*.
2. *Your code must not serialize the execution of processes. That is, if two pairs of processes have no intersections in terms of the array cells they want to swap, then they should be able to execute concurrently. For instance, the swapping that processes 1 and 3 do can be done concurrently with the swapping that processes 2 and 5 do because the set of memory cells the pairs of processes access are disjoint.*

Extra Credit: After you verify the safety and liveness properties, revise your Promela code such that each proctype performs the swapping for an unbounded number of times. Note that your code must be *non-terminating*. (50 points)

- Are the LTL properties still satisfied? Explain what you experience.
- Should the proctypes be synchronized after each round of swapping?

Presentation: All teams should present their work in class on February 25th. For that, you can create slides and/or walk through your Promela code, and run it in SPIN.