# NSWI101: System Behaviour Models and Verification

## Lab 03 – Spin exercises

Jan Kofroň

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

# Recall: Spin

Explicit state model checker

- Generates all states of the model to verify

Input language – Promela

- Set of processes with interleaving statements
- Communicating via global variables and channels

**Finite state models only!**

# Example of Promela

```
bool turn, flag[2];
byte ncrit;

active [2] proctype user()
{
        assert(_pid == 0 || _pid == 1);
        again:
        flag[_pid] = 1;
        turn = _pid;
        (flag[1 - _pid] == 0 || turn == 1 - _pid);
        ncrit++;
        assert(ncrit == 1);
        /* critical section */
        ncrit--;
        flag[_pid] = 0;
        goto again;
}
```

# GUI for Spin

- Several implementations
- The best one (and sort-of official) is **iSpin**
  - Tcl script, TclTk interpreter required
  - For windows I recommend ActiveTcl
  - Be sure to set paths to both spin.exe and gcc.exe (I used cygwin)

# Evaluating Search Complexity – Simulation

How many reachable states does the following naïve Promela model generate?

```
init {
        byte i = 0;
        do
        :: i = i + 1;
        od
}
```

```
$ spin -p -l ex1a.pml
```

# Evaluating Search Complexity – Verification

Now we verify the model:

```
$ spin -a ex1a.pml
$ gcc -o pan pan.c
$ ./pan
```

# EXERCISE

Estimate how many reachable states there are for the following model.
Draw the complete reachability tree.

```
#define N 2
init {
        chan dummy = [N] of { byte };
        do
        :: dummy!85
        :: dummy!170
        od
}
```

# Exercise – Evaluation

```
$ spin -m -a ex1b.pml   # use -m to ignore buffer overflow
$ gcc -o pan pan.c
$ ./pan
```

# Exercise – Contd.

What happens if you set N to 3? Express the number of states as a function of N. Use the formula to calculate how many states there will be if you set N to 14? Check your prediction:

```
$ spin –m –a ex1b.pml
$ gcc –o pan pan.c
$ ./pan
```

# Comments on Memory usage I.

The efficiency of the conventional reachability analysis is determined by the state space storage functions. To study this, repeat the last verification run with a smaller and a bigger hash table for storing reachable states:

```
$ pan -w10 # hash table with 210 slots ...
$ pan -w20 # hash table with 220 slots ...
```

# Comments on Memory usage II.

Bit-state hashing method

- Probabilistic approach
- Uses all available (specified) memory
- Might miss some states

```
$ spin –m –a ex.1b.pml          # as before
$ gcc –DBITSTATE –o pan pan.c   # different
$ ./pan
```

# Exercise: Producer-Consumer Model

Describe producer/consumer problem in Promela using channels and check the model for invalid end states (deadlocks) and channels' buffer overruns

- i.e., suppose channels are not blocked (messages get lost instead) and you must control the number of messages within the channel by hand