NSWI101: System Behaviour Models and Verification

3. Spin

Jan Kofroň

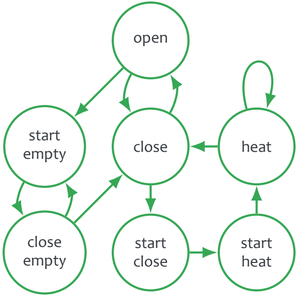
FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

Department of
Distributed and
Dependable
Systems

Today

- Spin model checker

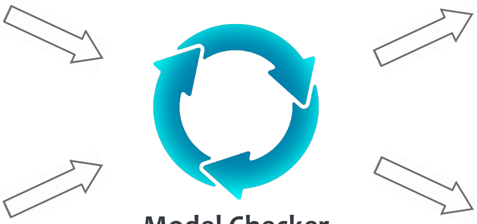
MODEL CHECKING



System model

AG (start $\rightarrow$ AF heat)

Property specification

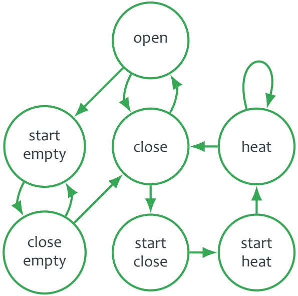


Model Checker

Property satisfied

Property violated

MODEL CHECKING



System model

AG (start → AF heat)

Property specification

Spin

Model Checker

Property satisfied

Property violated

SPIN OVERVIEW

Slides available at:

- `http://spinroot.com/spin/Doc/SpinTutorial.pdf`
- `http://spinroot.com/spin/Doc/Spin_tutorial_2004.pdf`

INTERLEAVING OF PROCESSES

Interleaving semantics:

- Each time, process is selected, and its current statement is executed
- Selected process has to be enabled
- This is repeated
- Number of all possible interleavings may be very high
  $\Longrightarrow$ state space explosion $\Longrightarrow$ not verifiable models
- Mechanism to control the interleavings would be handy

Property specification

LTL_{-X} is used in Spin

- LTL without X operator
- More efficient model checking algorithm
- Still expressive enough

Describing properties of states (or runs), not of transitions between states

Example: Alternating bit protocol – ABP

Four versions with various properties:

1. Perfect lines
2. Loosing messages
3. Fixing deadlock
4. Checking for progress

ABP VERSION 1

```
#define MAX 4;
mtype {MSG, ACK};
chan toR = [1] of {mtype, byte, bit};
chan toS = [1] of {mtype, bit};

active proctype Sender()
{
  byte data;
  bit sendb, recvb;
  sendb = 0;
  data = 0;
  do
    :: toR ! MSG(data,sendb) ->
       toS ? ACK(recvb);
    if
      :: recvb == sendb -> sendb = 1-sendb;
         data = (data+1)%MAX;
      :: else -> skip; /* resend old data */
    fi
  od
}
```

```
active proctype Receiver()
{
  byte data, exp_data;
  bit ab, exp_ab;
  exp_ab = 0;
  exp_data = 0;
  do
    :: toR ? MSG(data,ab) ->
    if
      :: (ab == exp_ab) ->
           assert(data == exp_data);
           exp_ab = 1-exp_ab;
           exp_data = (exp_data+1)%MAX;
      :: else -> skip;
    fi;
    toS ! ACK(ab)
  od
}
```

ABP VERSION 2

Adding special stealing daemon process:

```
active proctype Daemon()
{
  do
    :: toR ? _, _, _
    :: toS ? _, _
  od
}
```

ABP VERSION 3

Fixing sender model to escape from deadlock:

```
do
  :: toR ! MSG(data,sendb) ->
     if
       ::toS ? ACK(recvb) ->
          if
            :: recvb == sendb -> sendb = 1-sendb;
                                 data = (data+1)%MAX;
            :: else /* resend old data */
          fi
       ::timeout /* message lost */
     fi
od
```

ABP VERSION 4

Augmenting receiver process to detect livelock:

```
do
  :: toR ? MSG(data,ab) ->
  if
    :: (ab == exp_ab) -> assert(data == exp_data);
                         exp_ab = 1-exp_ab;
                         progress:
                         exp_data = (exp_data+1)%MAX;
    :: else -> skip;
  fi;
  toS ! ACK(ab)
od
```

Alternating bit protocol – Summary

We should be aware of all possible executions and issues in the model

If there is error due to simplification (abstraction), it can still be ok

- In our example we may know that messages *can* get lost but are *usually* delivered
- Consider possible errors beyond the ignored ones!

Model is not implementation!