Lab3: Model-based Testing and Automated Test-case Generation

Mimmi Lindgren, Katrina Liang

February 2023

1 UML diagram

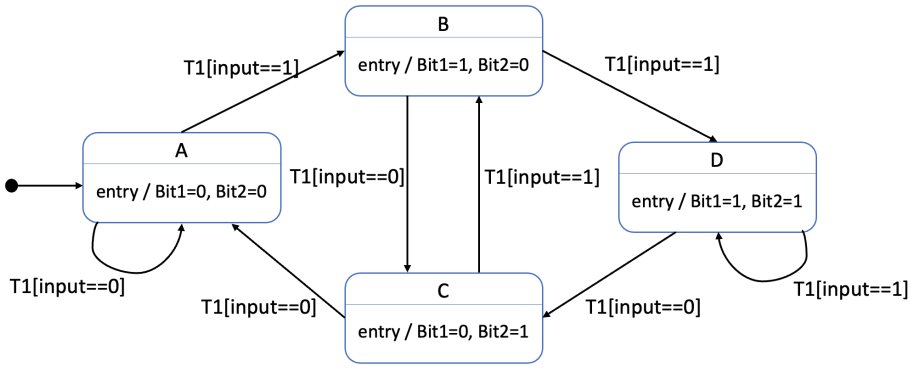

Figure 1: A UML diagram of the 2-bit shift register.

2 2-bit shift register in NuSMV

```
-- no counterexample found with bound 0
-- no counterexample found with bound 1
-- no counterexample found with bound 2
-- no counterexample found with bound 3
-- no counterexample found with bound 4
-- no counterexample found with bound 5
-- no counterexample found with bound 6
-- no counterexample found with bound 7
-- no counterexample found with bound 8
-- no counterexample found with bound 9
-- no counterexample found with bound 10
```

```
-- no counterexample found with bound 0
-- specification  G (Bit1 <->  X Bit1)    is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 1.1 <-
    Bit1 = FALSE
    Bit2 = FALSE
    state = s1
  -> Input: 1.2 <-
    input = TRUE
  -> State: 1.2 <-
    Bit1 = TRUE
    state = s2
```

Because the first specification $G(Bit1 <- > XBit2)$ should always be evaluated to TRUE, we did not get a counterexample from the tests. // The first test with bound 0 probably tests the initial state. For specification $G(Bit1 <- > XBit1)$, it will be evaluated to true until we change the state. We can see that when we go to another state with input = TRUE, the specification was evaluated to false.

3 NC of carcontroller.smv

3.1 Run the original carcontroller.smv for NC

```
-- specification  G !(state = stop)    is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 1.1 <-
    state = stop
```

Since we only have a trap property for the stop state, with a total of 3 states/nodes, we achieve $1/3 \approx 33\%$ node coverage.

3.2 Extract a test case from the counterexample for Stop

A test case based on the counterexample for the state Stop is shown in Table 1.

	t = 0
inputs	null
outputs	stop

Table 1: Test case for the node Stop.

3.3 Extract test cases from the counterexample for Slow ans Fast

A test case based on the counterexample for the state Slow is shown in Table 2, and for the state Fast in Table 3.

```
-- specification  G !(state = slow)    is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 2.1 <-
    state = stop
  -> Input: 2.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 2.2 <-
    state = slow
```

	t = 0	t = 1
inputs	null	accelerate = TRUE; brake = FALSE;
outputs	stop	slow

Table 2: Test case for the node Slow.

```
-- specification  G !(state = fast)    is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 3.1 <-
    state = stop
  -> Input: 3.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 3.2 <-
    state = slow
  -> Input: 3.3 <-
  -> State: 3.3 <-
    state = fast
```

	t = 0	t = 1	t = 2
inputs	null	accelerate = TRUE; brake = FALSE;	null
outputs	stop	slow	fast

Table 3: Test case for the node Fast.

4 EC of carcontroller.smv

4.1 Run the original carcontroller.smv for EC

```
-- no counterexample found with bound 0
-- specification  G ((state = stop & accelerate) ->  X !(state = slow))   is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 4.1 <-
    state = stop
  -> Input: 4.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 4.2 <-
    state = slow
```

Since we only have a trap property for the edge from stop state to slow state with accelerate, with a total of 6 edges, we achieve $1/6 \approx 16,6\cdots\%$ node coverage.

4.2 Extract a test case from the counterexample for Stop

A test case based on the counterexample for the edge accelerate between stop and slow is shown in Table 4.

	t = 0	t = 1
inputs	null	accelerate = TRUE; brake = FALSE;
outputs	stop	slow

Table 4: Test case for the edge accelerate between stop and slow.

4.3 100% EC as LTL formulas

The following shows the 5 additional trap propoerties needed for 100% EC:

```
LTLSPEC
    G( state=slow & !accelerate & !brake -> X(!(state=stop)))

LTLSPEC
    G( state=slow & brake & !accelerate -> X(!(state=stop)))

LTLSPEC
    G( state=fast & brake -> X(!(state=stop)))

LTLSPEC
    G( state=fast & !accelerate -> X(!(state=slow)))
```

```
LTLSPEC
    G( state=slow & accelerate -> X(!(state=fast)))
```

Counterexamples and test cases

```
-- no counterexample found with bound 0
-- no counterexample found with bound 1
-- specification  G (((state = slow & !accelerate) & !brake) ->
X !(state = stop))  is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -- Loop starts here
  -> State: 2.1 <-
    state = stop
  -> Input: 2.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 2.2 <-
    state = slow
  -> Input: 2.3 <-
    accelerate = FALSE
  -> State: 2.3 <-
    state = stop
-- no counterexample found with bound 0
-- no counterexample found with bound 1
-- specification  G (((state = slow & brake) & !accelerate) ->
X !(state = stop))    is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -- Loop starts here
  -> State: 3.1 <-
    state = stop
  -> Input: 3.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 3.2 <-
    state = slow
  -> Input: 3.3 <-
    accelerate = FALSE
    brake = TRUE
  -> State: 3.3 <-
    state = stop
-- no counterexample found with bound 0
```

```
-- no counterexample found with bound 1
-- no counterexample found with bound 2
-- specification  G ((state = fast & brake) ->  X !(state = stop))  is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -- Loop starts here
  -> State: 7.1 <-
    state = stop
  -> Input: 7.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 7.2 <-
    state = slow
  -> Input: 7.3 <-
  -> State: 7.3 <-
    state = fast
  -> Input: 7.4 <-
    accelerate = FALSE
    brake = TRUE
  -> State: 7.4 <-
    state = stop
-- no counterexample found with bound 0
-- no counterexample found with bound 1
-- no counterexample found with bound 2
-- specification  G ((state = fast & !accelerate) ->  X !(state = slow))  is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 8.1 <-
    state = stop
  -> Input: 8.2 <-
    accelerate = TRUE
    brake = FALSE
  -- Loop starts here
  -> State: 8.2 <-
    state = slow
  -> Input: 8.3 <-
  -> State: 8.3 <-
    state = fast
  -> Input: 8.4 <-
    accelerate = FALSE
  -> State: 8.4 <-
    state = slow
-- no counterexample found with bound 0
-- no counterexample found with bound 1
```

```
-- specification  G ((state = slow & accelerate) ->  X !(state = fast))  is false
-- as demonstrated by the following execution sequence
Trace Description: BMC Counterexample
Trace Type: Counterexample
  -> State: 9.1 <-
    state = stop
  -> Input: 9.2 <-
    accelerate = TRUE
    brake = FALSE
  -> State: 9.2 <-
    state = slow
  -> Input: 9.3 <-
  -> State: 9.3 <-
    state = fast
```

	t = 0	t = 1	t = 2
inputs	null	accelerate = TRUE; brake = FALSE;	accelerate = FALSE;
outputs	stop	slow	stop

Table 5: Test case for the edge !accelerate between slow and stop.

	t = 0	t = 1	t = 2
inputs	null	accelerate = TRUE; brake = FALSE;	accelerate = FALSE; brake = TRUE;
outputs	stop	slow	stop

Table 6: Test case for the edge brake between slow and stop.

	t = 0	t = 1	t = 2	t = 3
inputs	null	accelerate = TRUE; brake = FALSE;	null	accelerate = FALSE; brake = TRUE;
outputs	stop	slow	fast	stop

Table 7: Test case for the edge brake between fast and stop.

	t = 0	t = 1	t = 2	t = 3
inputs	null	accelerate = TRUE; brake = FALSE;	null	accelerate = FALSE;
outputs	stop	slow	fast	slow

Table 8: Test case for edge !accelerate between fast and slow.

	t = 0	t = 1	t = 2
inputs	null	accelerate = TRUE; brake = FALSE;	null
outputs	stop	slow	fast

Table 9: Test case for edge accelerate between slow and fast.