CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Exercise 6: Finite State Verification

Gregory Gay
DIT636/DAT560 - March 5, 2025

Finish In-Class Activity First!

Microwave

Consider a simple microwave controller modeled as a finite state machine using the following state variables:

- `Door: {Open, Closed}` -- sensor input indicating state of the door
- `Button: {None, Start, Stop}` -- button press (assumes at most one at a time)
- `Timer: 0...999` -- (remaining) seconds to cook
- `Cooking: Boolean` -- state of the heating element

Partial Model

```
MODULE microwave
VAR
    Door: {Open, Closed};
    Button: {None, Start, Stop};
    Timer: 0..999;
    Cooking: boolean;
ASSIGN
    init(Door) := Closed;
    init(Button) := None;
    init(Timer) := 0;
    next(Timer) :=
    case
        Timer > 0 & Cooking=TRUE : Timer - 1;
        Timer > 0 & Cooking=FALSE & Button!=Stop : Timer;
        Button=Stop : 0;
        Timer=0 : 0..999;
        TRUE: Timer;
    esac;
```

```
init(Cooking) := FALSE;
next(Cooking) :=
case
    -- Suggestion: Start by defining the
    -- conditions that would cause
    -- cooking to start. Then add conditions
    -- that would make it stop.
    -- Finally, ensure it will continue
    -- running if it is supposed to.
    (FILL THIS IN)
    TRUE: FALSE;
esac;
```

Example Properties

- CTL: The microwave shall stop cooking after the door is opened.
  - `AG (Door = Open -> AX (!Cooking))`
- LTL: It shall never be the case that the microwave can continue cooking indefinitely.
  - `G (Cooking -> F (!Cooking))`
- Formulate the other informal requirements in temporal logic.

Linear Time Logic Formulae

Formulae written with propositional variables (boolean properties), logical operators (and, or, not, implication), and a set of modal operators:

X (next)	X (weather = rain)	In the next state, it will be raining.
G (globally)	G (weather = rain)	Now and in all future states, it will be raining.
F (finally)	F (weather = rain)	Eventually, there will be a state where it is raining.
U (until)	(weather = rain) U (temperature < 0)	It will rain until the temperature drops below 0. (The value of “weather” can change once temperature is less than 0)
R (release)	weather = rain) R (temperature < 0)	It will cease to rain after the temperature drops below 0. (Both operands must be true at the same time for at least one state before the value of “weather” can change)

Computation Tree Logic Formulae

Combine one quantifiers (A, E) with a path-specific quantifier (X, G, F, U, W):

A (all)	Affects **all paths** branching out from the current state.
E (exists)	Affects **at least one path** branching out from the current state.

X (next)	X (weather = rain)	In the next state on this path, it will be raining.
G (globally)	G (weather = rain)	Now and in all future states on this path, it will be raining.
F (finally)	F (weather = rain)	Eventually on this path, there will be a state where it is raining.
U (until)	(weather = rain) U (temperature < 0)	On this path, it will rain until the temperature drops below 0. (The temperature must eventually be less than 0)
W (weak until)	weather = rain) W (temperature < 0)	On this path, it will rain until the temperature drops below 0. (The temperature could remain above 0 forever)

Try to Verify the Model and Properties

- http://nusmv.fbk.eu/
  - NuSMV homepage (tool download, tutorials, etc.)
  - Use NuSMV 2.6.
- Define `next(Cooking)` such that the two example properties hold. See if your properties hold.
  - If they don't, make sure the properties are correct.
  - Then, make sure the model is complete and correct.
- If you get stuck, a sample solution is on Canvas.

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY