Formal Verification

Lecture 4: Practical Exercise
Model Checking with NuSMV

Jacques Fleuriot
jdf@inf.ed.ac.uk
Jake Palmer
s1673264@sms.ed.ac.uk

Overview

The practical is in 2 parts:

1. Exercises with Temporal Logic
   - 1.1 Exercises on LTL
   - 1.2 Exercises on CTL
2. Verifying a First In First Out (FIFO) circuit
   - 2.1 Properties to Verify
   - 2.2 Model bug to fix
   - 2.3 Principles of LTL Model Checking

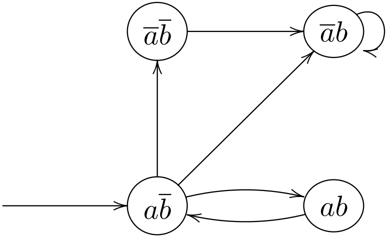
This coursework is **not** assessed

Objectives of the Practical

1. Acquire familiarity with the semantics of LTL formulas.
2. Acquire familiarity with the semantics of CTL formulas.
3. Experience in specifying a (toy) system using LTL and CTL.
4. Debugging a system using LTL model checking.
5. Exploring the principles of LTL model checking.
6. In general, gain knowledge of what model checking is capable of.

2.1 LTL

Write this model in NuSMV:



Use NuSMV to analyse some LTL properties ($\mathbf{G}\ a$, $a\ \mathbf{U}\ b$, ...):

1. Determine whether each property is valid (*i.e.*, true for all paths)
2. Get NuSMV to generate a path that satisfies the formula

2.2 CTL

Equivalence of CTL formulas *(CTL will be covered in the next lecture)*

You are given a list of pairs of CTL formulas, you have to determine whether or not each pair is equivalent.

$$\phi \equiv \psi \qquad \leftrightarrow \qquad (\forall \mathcal{M}\ s.\ \mathcal{M}, s \models \phi \leftrightarrow \mathcal{M}, s \models \psi)$$

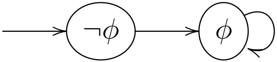
For each pair:

- If they are equivalent, then explain why, with reference to the semantics of CTL.
- If they are not equivalent, then give a NuSMV model that distinguishes them.

2.2 CTL (cont)

An example: the formulas **EF** ϕ and **EG** ϕ.

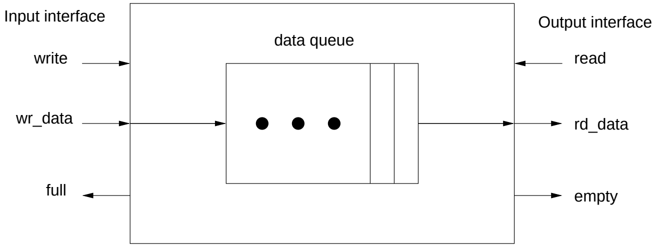
Not equivalent, this model distinguishes them:



For this model, **EF** ϕ is true, but **EG** ϕ is false.

Verifying a FIFO circuit

A FIFO circuit consists of input and output wires and a data queue:



The inputs are:

- `write`: if data should be written to the queue,
- `wr_data`: the data to be written if `write` is `TRUE`,
- `read`: if data should be read from the queue.

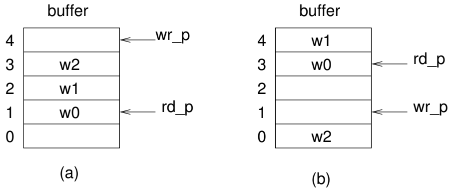
The FIFO cicruit outputs:

- `full`: if the buffer is full,
- `empty`: if the buffer is empty,
- `rd_data`: the data at the current read pointer.

Verifying a FIFO circuit

The file `fifo.smv` contains a NuSMV model for the FIFO circuit, which has 4 internal state variables buffer, rd_p, wr_p, and empty (also an output).

An example of an internal FIFO configuration is shown below.



Where will the pointers be if the buffer is empty? What if it is full?

3.2 Properties to Verify

You are given several properties to state in temporal logic, and to verify against the model using NuSMV.

1. 6 properties in LTL
2. 2 properties in CTL

Note: In many cases, you will have to refine your formalized properties with additional assumptions in order to get it to hold for the model.

But remember to stay true to the “spirit” of the property as stated in English.

3.3 Model Bug to Fix

The controller model has a bug. The bug is highlighted by the failure of the following property to verify:

> *always, if the FIFO indicates it is empty, then the read and write pointers are equal*

You have to:

1. Express this property in LTL, and get NuSMV to demonstrate a counterexample.
2. Explain what the bug is.
3. Fix the model.

You can use bounded model checking to get shorter counterexamples.

Principles of LTL model checking

Roughly, LTL model checking works by the following process:

1. Start with some formula ϕ.
2. Negate it: $\neg\phi$.
3. Convert $\neg\phi$ to a finite state automaton: a *Büchi* automaton.
4. Construct the *intersection* of the formula's automaton and the model's automaton.
5. Check to see whether the resulting automaton's language is empty.

Summary

- This time: Practical Exercise
  - LTL and CTL
  - Verifying a FIFO circuit
- Next time:
  - CTL: Computation Tree Logic
  - A *Branching-time* temporal logic