CSE 620: Selective Topics

Introduction to Formal Verification

Master Studies in CSE

Fall 2017

Lecture #6

Dr. Hazem Ibrahim Shehata

Assistant Professor

Dept. of Computer & Systems Engineering

Course Outline

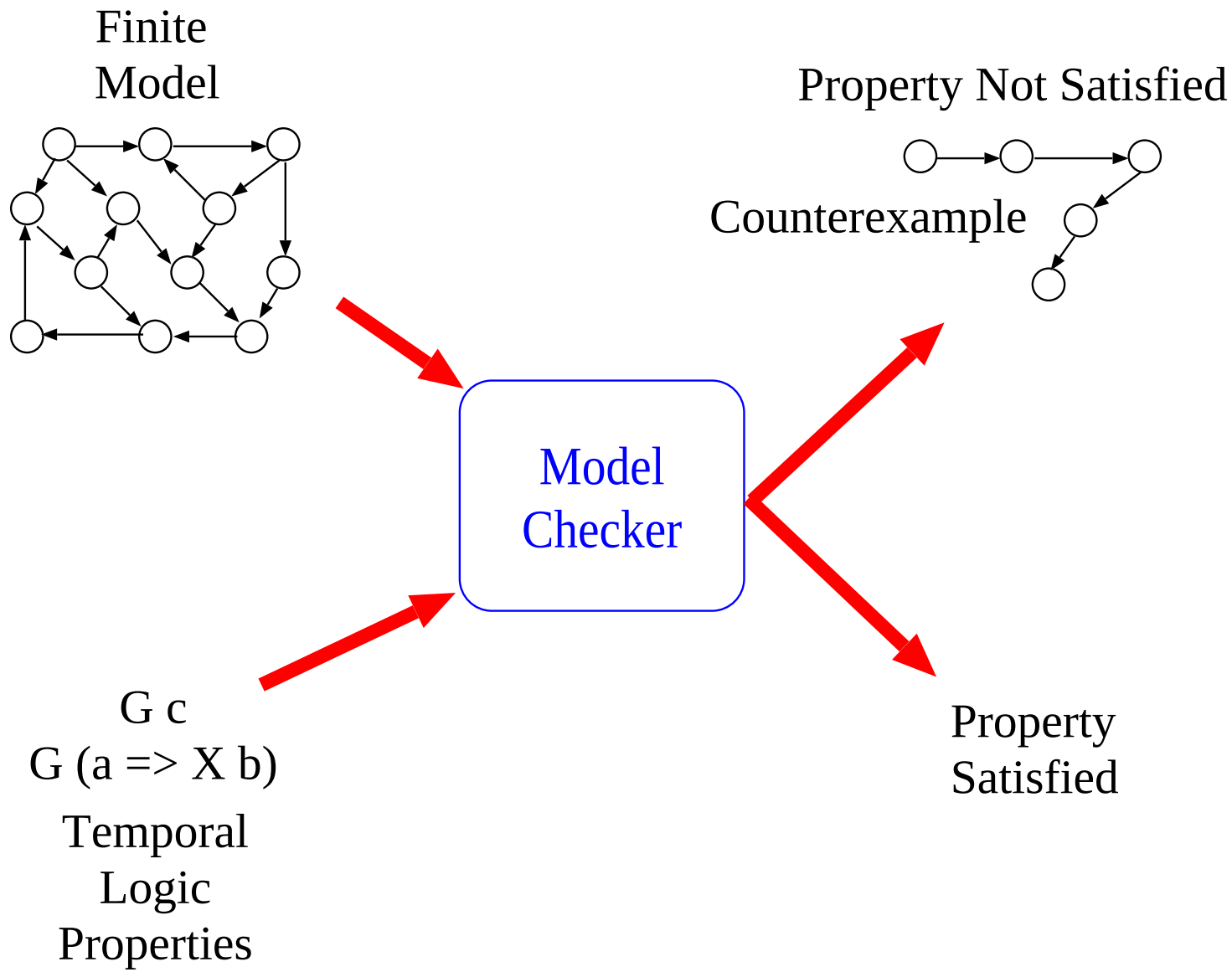
- Computational Boolean Algebra
  - —Basics
    - – Shannon Expansion
    - – Boolean Difference
    - – Quantification Operators
      - + Application to Logic Network Repair
  - —Validity Checking (Tautology Checking)
  - —Binary Decision Diagrams (BDD's)
  - —Satisfiability Checking (SAT solving)
- Model Checking
  - —Temporal Logics ➔ LTL - CTL
  - —SMV: Symbolic Model Verifier
  - —Model Checking Algorithms ➔ Explicit CTL

CS745/ECE-725: Computer-Aided Verification

Topic 4: Temporal Logic and Model Checking

http://www.student.cs.uwaterloo.ca/~cs745
uw.cs.cs745
cs745@student.cs.uwaterloo.ca
Nancy Day

Model Checking



Model Checking

A counterexample is a trace of the system that violates the property.

An error (counterexample) may indicate a problem in the system or it may demonstrate that you didn't write your property correctly.

Verification via Model Checking

$$\mathcal{M} \models \phi$$

- $\mathcal{M}$ is a model (or implementation)
- ϕ is a property (or specification)
- $\models$ is a relationship that should hold between $\mathcal{M}$ and ϕ

In model checking, the model is a Kripke structure (labelled state transition graph), and the properties are written in temporal logic. We rarely work with Kripke structures directly but rather specification notations that describe Kripke structures.

There are different temporal logics. For each temporal logic, we will define what $\models$ means.

Models and Entailment

Recall that $\models$ is overloaded and has two meanings (see Huth and Ryan [R18], p. 137, p. 149):

1. $\mathcal{M} \models \phi$ relates a model to a formula, saying that $\mathcal{M}$ satisfies the formula ϕ. This is called a satisfaction relation.
2. $\psi \models \phi$ relates two formulae, saying that forall v (i.e., for all possible models), if $v(\psi) = \mathsf{T}$ then $v(\phi) = \mathsf{T}$. This is called semantic entailment.

These two uses can be distinguished by their context.

For model checking, we are talking about the first use of $\models$.

Historical Perspective

- Burstall, Pnueli and others proposed using temporal logic (TL) for reasoning about programs (early 1970's)
- Pnueli was the first to use TL for reasoning about concurrency [R26] (1977)
- Clarke and Emerson [R9] in the early 1980's developed a way to automate temporal logic reasoning (CTL); Quielle and Sifakis [R27] also gave a model checking algorithm at this time.

Historical Perspective

- language containment approach to model checking
  - SPIN - Holzmann [R16]
  - COSPAN/Formalcheck – Kurshan [R20]
- symbolic model checking (implicit representation of the state space) – McMillan, 1987 [R6]
- lots and lots of work on handling the state space explosion problem

Another approach to model checking is symbolic trajectory evaluation (STE), developed by Bryant and Seger [R28].

Classification of Properties

- Safety:
  - Description: Something bad does not happen, or something good happens within a finite amount of time.
  - Sample prop: It's never the case that the elevator door is open while the elevator is moving.
  - Counterexample: Finite sequence, ending when the bad thing happens.
- Liveness:
  - Description: Something good happens in the future.
  - Sample prop: The elevator eventually visits every floor.
  - Counterexample: Infinite sequence in which the good thing never happens.

Every property is either pure safety, pure liveness, or safety followed by liveness.

Classification of Properties

- Fairness: something is true infinitely often, e.g., a process gives up a token infinitely often, so another process can make progress.

Fairness properties often have to be assumed in order to prove other properties.

Fairness properties are a subset of liveness properties.

Model checkers often have special provisions for dealing with fairness properties.

Temporal Logic

"Temporal logics have proven to be useful for specifying concurrent systems, because they can describe the ordering of events in time without introducing time explicitly." [R11] Temporal logic uses operators such as eventually and always.

Temporal logic is good for describing the behaviour of reactive systems.

A reactive system is one that interacts with its environment and doesn't terminate. We need to know more than their input-output behaviour. We want to check how the state of the system changes when an action occurs.

Sources for this material: Clarke, Grumberg, and Peled [R11], Peled [R25], Huth and Ryan [R17].

Temporal Logic

Temporal logic is used to describe changes in values over time.

We will discuss propositional temporal logic.

So we now consider that propositions can have different truth values at different times.

We can use all the regular logical connectives to create statements in propositional logic.

A temporal logic formula is considered relative to an initial state ("now").

Linear Temporal Logic (LTL)

Temporal logics use temporal operators. Linear temporal logic uses the following operators:

Symbol	Alternate Symbol	Informal Meaning
X	$\bigcirc$	Next
F	$\Diamond$	Eventually (in the future)
G	$\Box$	always (globally, henceforth)
U	$\bigcup$	strong until
W	**W**	weak until

LTL Syntax

If p is an atomic proposition, and f_1 and f_2 are LTL formulae, then the set of LTL formulae consists of:

1. p
2. $\neg f_1$, $f_1 \wedge f_2$, $f_1 \vee f_2$, $f_1 \Rightarrow f_2$
3. $\mathbf{X} f_1$
4. $\mathbf{G} f_1$
5. $\mathbf{F} f_1$
6. $f_1 \ \mathbf{U} \ f_2$

Brackets are used as necessary.

Weak Until

- a **W** b means that a holds at every point in time up to a point where b holds, but b may never hold.

Temporal Logic

Some operators can be expressed in terms of the other operators. For example:

$$\mathbf{F}\ f \equiv \mathbf{true}\ \ \mathbf{U}\ f$$

$$\mathbf{G}\ f \equiv \neg(\mathbf{F}\ \neg f)$$

Traditional LTL Semantics

We gave one version of the semantics of LTL as an embedding in higher order logic.

Now we will present the more traditional view of the semantics. In the more traditional view, in the satisfaction relation $\mathcal{M} \models \phi$:

- $\mathcal{M}$ is a Kripke structure,
- ϕ is a property in temporal logic,
- the satisfaction relation, $\models$, is defined by structural induction on the property

Kripke Structures

Let AP be a set of atomic propositions. A Kripke structure $\mathcal{M}$ over AP is a four tuple $\mathcal{M} = (S, S_0, R, L)$ where

1. S is a finite set of states.
2. $S_0 \subseteq S$ is the set of initial states.
3. $R \subseteq S \times S$ is a transition relation that must be total, that is $\forall s \in S.\, \exists s'.\, R(s, s')$.
4. $L : S \to 2^{\mathrm{AP}}$ is a function that labels each state with the set of atomic propositions true in that state.

Note: there are no labels on the transition arrows.

Labelling Function

$$L : S \rightarrow 2^{\mathrm{AP}}$$

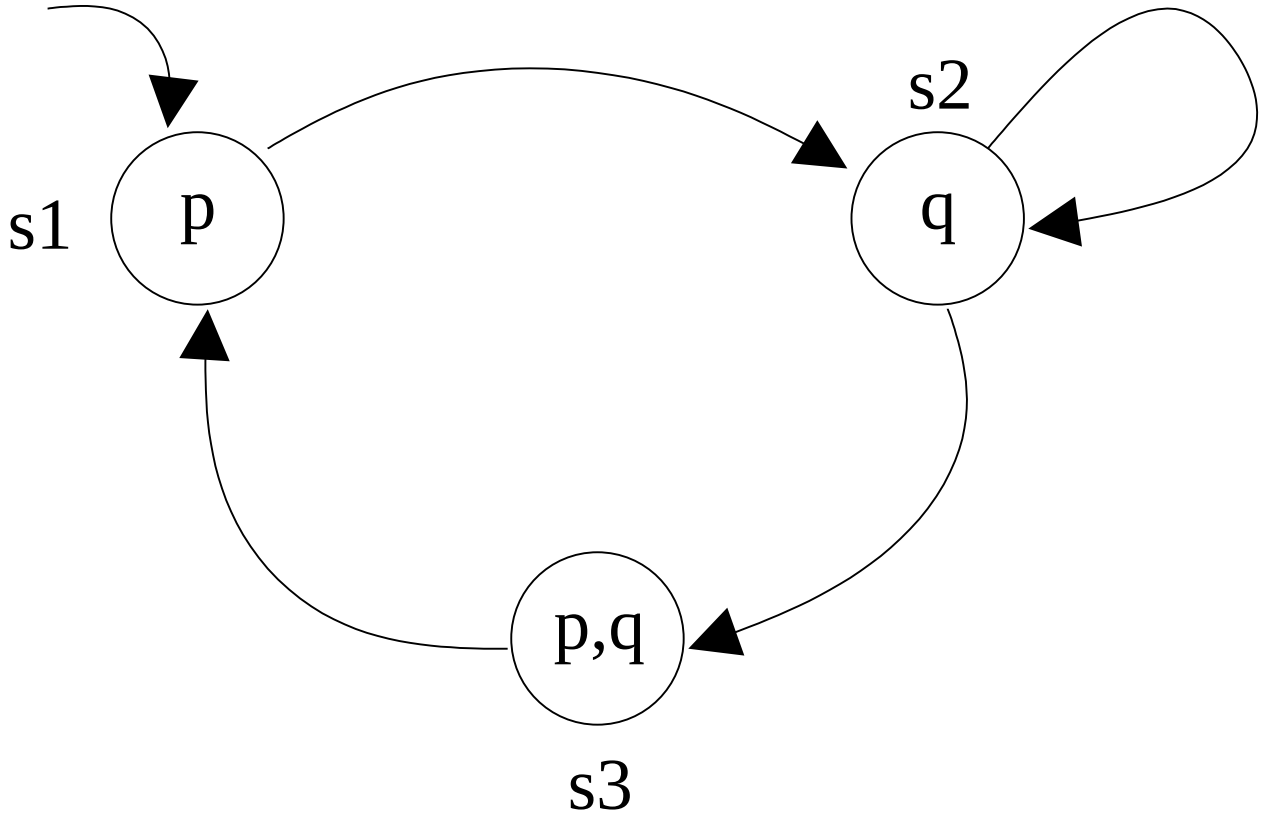
AP is the set of atomic propositions. For example, AP might be $\{p, q, r\}$.

2^{AP} is the power set of the set of atomic propositions.

$$2^{\{p,q,r\}} = \{\emptyset, \{p\}, \{q\}, \{r\}, \{p, q\}, \{p, r\}, \{q, r\}, \{p, q, r\}\}$$

L is a function that takes a state as an argument and returns the set of propositions that are true in that state.

Labelling Function Example



What is the labelling function in this example?

$L(s_1) = \{p\}, L(s_2) = \{q\}, L(s_3) = \{p, q\}$

Paths

As we saw, LTL formulae are evaluated relative to paths (i.e., an LTL formula is true or false relative to a sequence of values).

In the HOL formulation, we used propositions as functions of time to model the paths. Functions of time are often called streams.

For a Kripke structure, a path from a state s_0 is an infinite sequence $\pi = s_0 s_1 s_2 \ldots$ such that $\forall i. R(s_i, s_{i+1})$. We use π_i to denote the suffix of π starting at s_i.

LTL Semantics

$\mathcal{M}, \pi \models g$ means the LTL formula g holds on path $\pi = s_0 s_1 s_2 \ldots$ in the Kripke structure $\mathcal{M} = (S, S_0, R, L)$. The relation $\models$ is defined inductively as follows:

$$
\begin{array}{lll}
\mathcal{M}, \pi \models p & \text{iff} & p \in L(s_0) \text{ (where } p \text{ is an atomic proposition)} \\
\mathcal{M}, \pi \models \neg g & \text{iff} & \mathcal{M}, \pi \not\models g \\
\mathcal{M}, \pi \models g_1 \vee g_2 & \text{iff} & \mathcal{M}, \pi \models g_1 \text{ or } \mathcal{M}, \pi \models g_2 \\
\mathcal{M}, \pi \models g_1 \wedge g_2 & \text{iff} & \mathcal{M}, \pi \models g_1 \text{ and } \mathcal{M}, \pi \models g_2 \\
\mathcal{M}, \pi \models \mathbf{X}\, g & \text{iff} & \mathcal{M}, \pi_1 \models g \\
\mathcal{M}, \pi \models \mathbf{G}\, g & \text{iff} & \forall i \geq 0,\ \mathcal{M}, \pi_i \models g \\
\mathcal{M}, \pi \models \mathbf{F}\, g & \text{iff} & \exists i \geq 0,\ \mathcal{M}, \pi_i \models g \\
\mathcal{M}, \pi \models g_1\ \mathbf{U}\ g_2 & \text{iff} & \exists i \geq 0,\ \mathcal{M}, \pi_i \models g_2 \\
 & & \quad \text{and } \forall j.\, 0 \leq j < i \Rightarrow \mathcal{M}, \pi_j \models g_1
\end{array}
$$

LTL Semantics

We just defined $\mathcal{M}, \pi \models g$, where π is a path.

An LTL formula g is satisfied in a state s of a model $\mathcal{M}$ if g is satisfied on every path starting at s.

How do we say there is some path where a formula is true?

For example, "from any state, there is always some way to get to a state where p is true".

In LTL we can't say this property. We turn now to another temporal logic: CTL.

LTL is conceptually simpler than CTL because we only have to think about one path at a time.