CSCI 740 - Programming Language Theory

Lecture 31

Introduction to Models and Properties

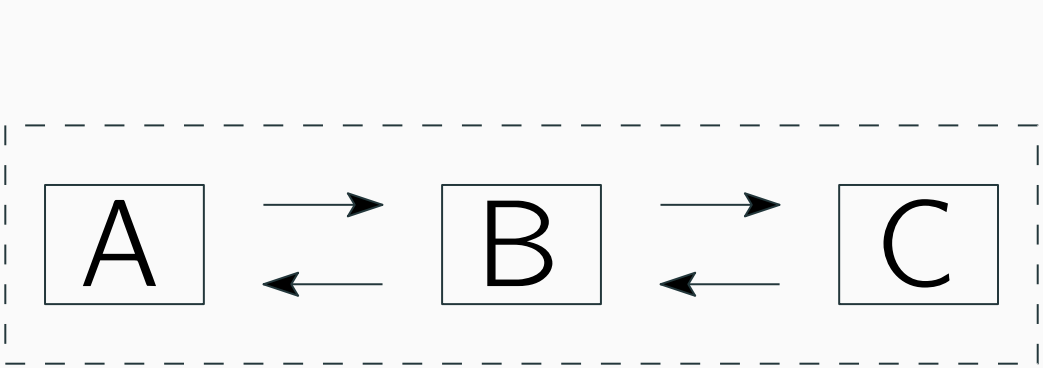
Instructor: Hossein Hojjat

November 15, 2017

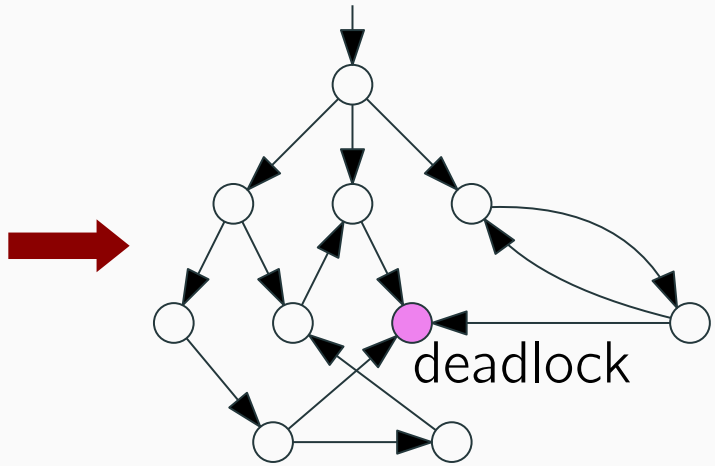
Recap

	Type Inference	Verification Condition Generation	Abstract Interpretation	**Model Checking**
Properties	Properties of Variables	Properties at Program Points	Properties at Program Points	Properties of Execution Traces
Flexible	No	Yes	No	Yes
Push-Button	Yes	No	Yes	Yes

Model Checking



Each component is modeled by a FSM



- Model Checking = systematic state-space exploration = exhaustive testing
- Model Checking = check whether the system satisfies a temporal-logic formula
  - Example: $G(p \rightarrow Fq)$ is an LTL formula
- Simple yet effective technique for finding bugs in hardware and software designs

Hardware Model Checking Today

Intel Pentium FDIV Bug

- Try $4195835 - 4195835/3145727 * 3145727$
- In 94 Pentium, did not return 0, but 256
- Cost: $400 - $500 million
- E. M. Clarke, M. Khaira, X. Zhao:
  "Word level model checking - avoiding the Pentium FDIV error" (DAC '96)

Hardware Model Checking Today

- Part of the standard toolkit for hardware design
- Intel has used it for production chips since Pentium 4
- For the Intel Core i7, most pre-silicon validation was done through formal methods
  - "Replacing Testing with Formal Verification in Intel® Core™ i7 Processor Execution Engine Validation" (CAV 2009)
  - (i.e. Model Checking + Theorem Proving)
- Many commercial products
  - IBM RuleBase, Synopsys Magellan, ...

Software Model Checking Today

- Static driver verifier now a commercial Microsoft product
  - `http://www.microsoft.com/whdc/devtools/tools/sdv.mspx`
- Java PathFinder used to verify code for mars rover
  - Java PathFinder(NASA): `http://javapathfinder.sourceforge.net/`
- This does not mean Model Checking is a solved problem
  - Far from it

Model Checking Genesis

Edmund M. Clarke, Joseph Sifakis, E. Allen Emerson
(Turing Award 2007)

- Clarke and Emerson, "Design and Synthesis of Synchronization Skeletons using branching time temporal logic", 1981

"Proof Construction is unnecessary in the case of finite state concurrent systems and can be replaced by a model-theoretic approach which will mechanically determine if the system meets a specification expressed in propositional temporal logic"

- Jean-Pierre Queille and J. Sifakis, "A temporal logic to deal with fairness in transition systems", 1982

Intellectual Roots

Two important developments preceded this paper

1. "Verification through exhaustive exploration of finite state models"
   - G. V. Bochmann and J. Gecsei, "A unified method for the specification and verification of protocols, 1977

2. "Development of Linear Temporal Logic and its application to specifying system properties"
   - A. Pnueli, The temporal semantics of concurrent programs, 1977

Model Checking

The model checking approach (as characterized by Emerson)

- Start with a program that defines a finite state graph M
- Search M for patterns that tell you whether a specification f holds
- Pattern specification is flexible
- The method is efficient in the sizes of M and hopefully also f
- The method is algorithmic

What exactly is a model?

Remember our friend $\vdash$?

- What does this mean? $\vdash x \wedge y \Rightarrow x$
- The statement above can be established through logical deduction
- Axiomatic semantics and type theory are deductive
  - The program, together with the desired properties make a theorem
  - We use deduction to prove the theorem
- What about this; is it true? $\vdash x + y = 5$
- We can not really establish this through deduction
- We can say whether it's true or false under a given model

$$[x = 3, y = 2] \models x + y = 5$$

You have seen this symbol too $\models$

- In operational semantics, the variable assignments were the model
- The program behavior was the theorem we were trying to prove under a given model

Basic Notions of Model Theory

- Consider the following sentence:

$$S := \text{The class today was awesome}$$

- Is this sentence true or false? That depends
- What class is "the class"? What day is "today"?
- We can give this sentence an Interpretation

$$I := \text{The class is CSCI 740, Today is Wednesday Nov 15}$$

- When an interpretation I makes S true we say that
  - I satisfies S
  - I is a model of S
  - $I \models S$

The Model Checking Problem

- We are interested in deciding whether $I \models S$ for the special case where
  - I is a Kripke structure
  - S is a temporal logic formula
- Today you get to learn what each of these things are

High level idea:

- Unlike axiomatic semantics, where the program was part of the theorem,
- The program will now be the model
  - Well, not the program directly, but rather a Kripke structure representing the program

Kripke Structures as Models

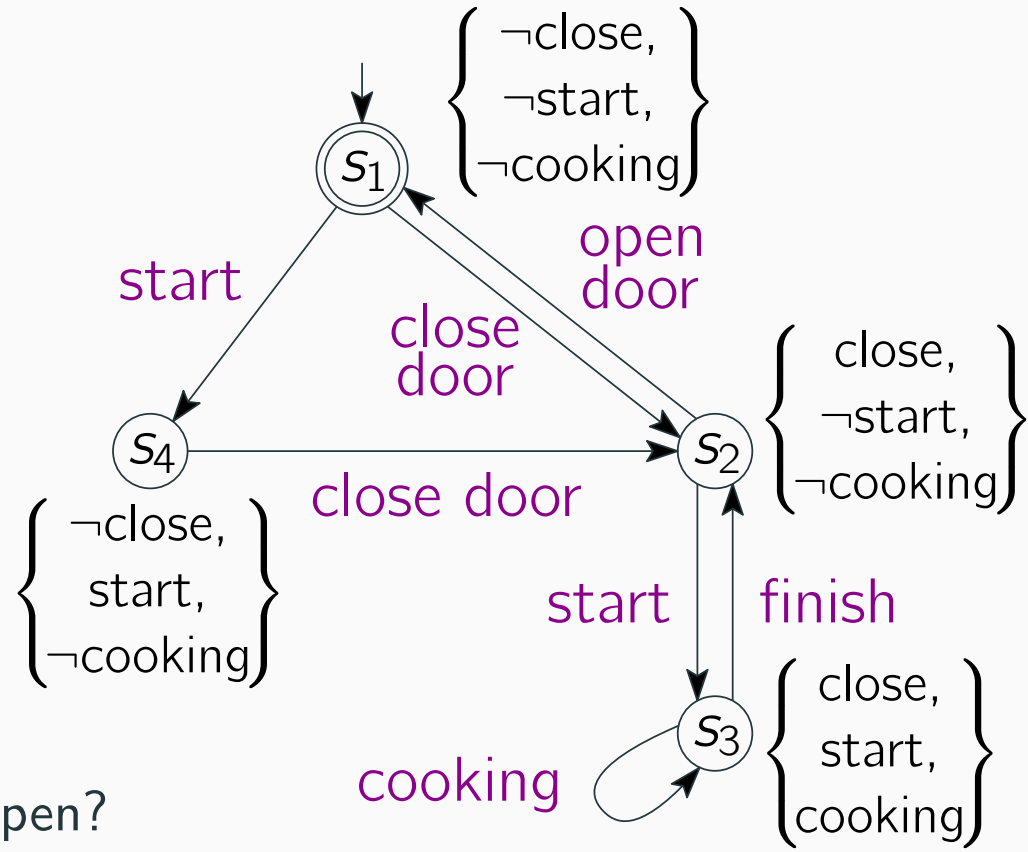
Kripke structure is a FSM with labels

Kripke structure $= (S, S_0, R, L)$

- S = finite set of states
- $S_0 \subseteq S$ = set of initial states
- $R \subseteq S \times S$ = transition relation
- $L : S \to 2^{\mathsf{AP}}$ = labels each state with a set of atomic propositions

Microwave Example

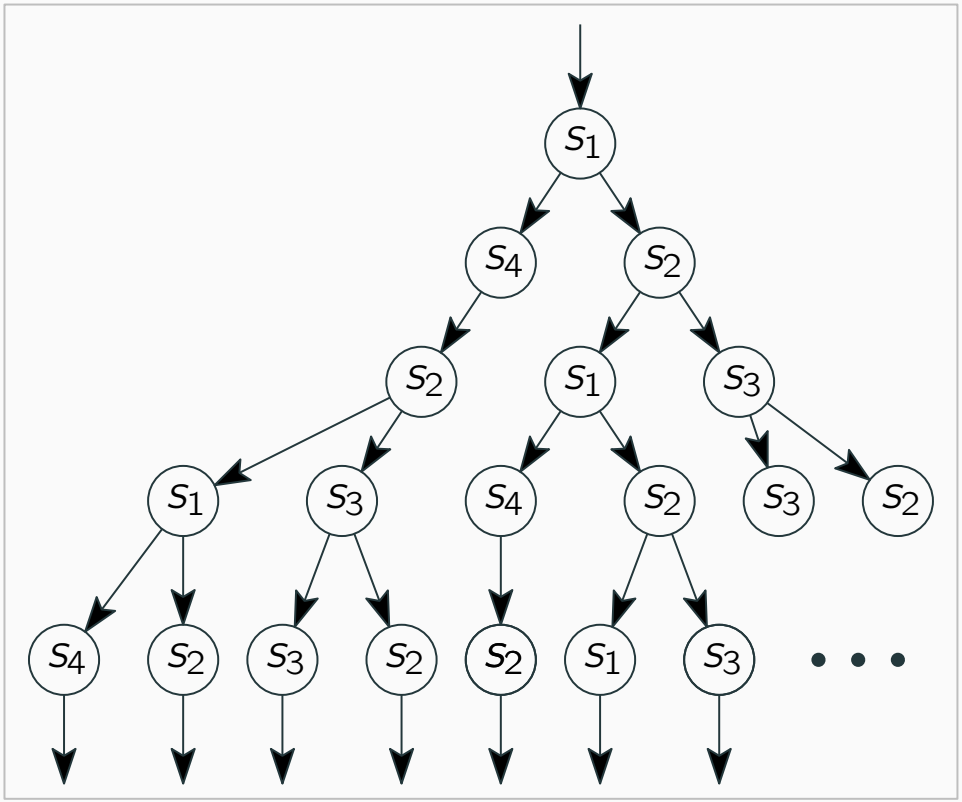
- $S = \{s_1, s_2, s_3, s_4\}$
- $S_0 = \{s_1\}$
- $R = \{(s_1, s_2), (s_2, s_1), (s_1, s_4), (s_4, s_2), (s_2, s_3), (s_3, s_2), (s_3, s_3)\}$
- $L(s_1) = \{\neg\text{close}, \neg\text{start}, \neg\text{cooking}\}$
- $L(s_2) = \{\text{close}, \neg\text{start}, \neg\text{cooking}\}$
- $L(s_3) = \{\text{close}, \text{start}, \text{cooking}\}$
- $L(s_4) = \{\neg\text{close}, \text{start}, \neg\text{cooking}\}$

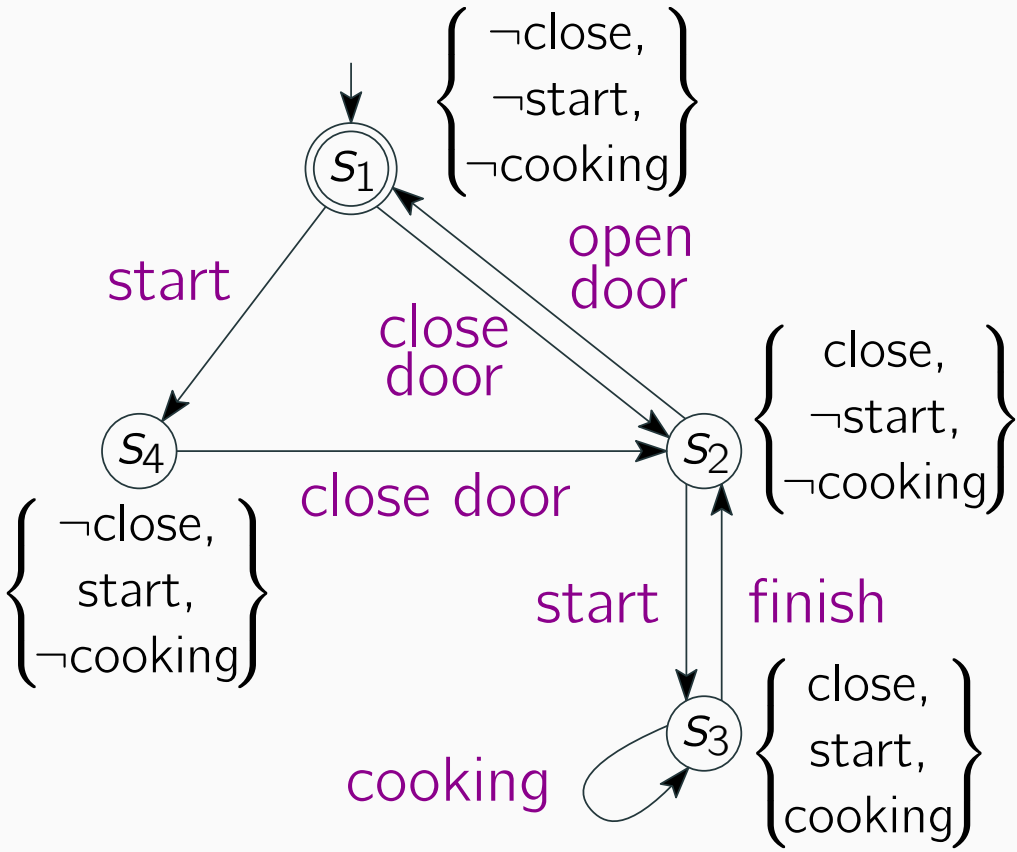


Can the microwave cook with the door open?

Microwave Example

- A Kripke structure can describe an infinite process
- We can interpret it as an infinite tree





We need a language to describe properties of paths down the computation tree

Linear Temporal Logic

- Let π be a sequence of states in a path down the tree
- $\pi := s_0, s_1, s_2, \cdots$
- Let π_i be a subsequence starting at i
- We are going to define a logic to describe properties over paths

Properties over States

State Formulas

- Can be established as true or false on a given state
- If $p \in \{\text{AP}\}$ then p is a state formula
- If f and g are state formulas, so are $(f \wedge g)$, $(\neg f)$, $(f \vee g)$
- Ex. $\neg$closed $\wedge$ cooking

Path Formulas

- A state formula p is also a path formula

$$p(\pi_i) := p(s_i)$$

- Boolean operations on path formulas are path formulas

$$(f \wedge g)(\pi_i) := f(\pi_i) \wedge g(\pi_i)$$

Path Quantifiers

- $Gf(\pi_i) :=$ globally $f(\pi_i) =$ forall $k \geq i . f(\pi_k)$
  - ($\Box$ another notation for G)
- $Ff(\pi_i) :=$ eventually $f(\pi_i) =$ exists $k \geq i$ s.t. $f(\pi_k)$
  - ($\Diamond$ another notation for F)
- $Xf(\pi_i) :=$ next $f(\pi_{i+1})$
  - ($\circ$ another notation for X)
- $f\ Ug(\pi_i) := f$ until $g =$ exists $k \geq i$ s.t.

$$g(\pi_k) \text{ and } f(\pi_j) \text{ for } i \leq j < k$$

Given a formula f and a path π, if $f(\pi)$ is true, we say $\pi \models f$