Topics in Verification

Azadeh Farzan

Fall 2017
Model Checking
OVERVIEW
OVERVIEW

- Nontraditional use of nontraditional logic!
OVERVIEW

- **Nontraditional use of nontraditional logic!**
- Checking whether a formula is satisfied in a finite domain.
OVERVIEW

- **Nontraditional use of nontraditional logic!**
- Checking whether a formula is satisfied in a **finite** domain.
  - **Model:** finite-state transition system
OVERVIEW

- Nontraditional use of nontraditional logic!
- Checking whether a formula is satisfied in a finite domain.
  - Model: finite-state transition system
  - Logic: Propositional Temporal Logic.
OVERVIEW

- Nontraditional use of nontraditional logic!
- Checking whether a formula is satisfied in a finite domain.
  - Model: finite-state transition system
  - Logic: Propositional Temporal Logic.
  - Verification Procedure: exhaustively search of the state space to determine the truth of specification.
WHY MODEL CHECKING?
WHY MODEL CHECKING?
WHY MODEL CHECKING?

☐ Doesn’t aim too high!
WHY MODEL CHECKING?

- Doesn’t aim too high!
- Originally restricted to finite-state systems.
WHY MODEL CHECKING?

- Doesn’t aim too high!

- Originally restricted to finite-state systems.

- Applicable to systems with “short” descriptions.
WHY MODEL CHECKING?

☐ Doesn’t aim too high!

☐ Originally restricted to finite-state systems.

☐ applicable to systems with “short” descriptions.

☐ control-oriented systems such as hardware, protocols, ...Can we come up with them automatically?
WHY MODEL CHECKING?

- Doesn’t aim too high!
  - Originally restricted to finite-state systems.
  - Applicable to systems with “short” descriptions.
  - Control-oriented systems such as hardware, protocols, ... Can we come up with them automatically?
- Fully automatic with low computational complexity.
WHY MODEL CHECKING?

- Doesn’t aim too high!
  - Originally restricted to finite-state systems.
  - Applicable to systems with “short” descriptions.
  - Control-oriented systems such as hardware, protocols, ...Can we come up with them automatically?
- Fully automatic with low computational complexity.
- Can be viewed as an elaborate debugging tool: counterexamples.
First Step:
We need a formal model!
A transition system $TS$ is a tuple $(S, \text{Act}, \rightarrow, I, \text{AP}, L)$ where

- $S$ is a set of states,
- $\text{Act}$ is a set of actions,
- $\rightarrow \subseteq S \times \text{Act} \times S$ is a transition relation,
- $I \subseteq S$ is a set of initial states,
- $\text{AP}$ is a set of atomic propositions, and
- $L : S \rightarrow 2^{\text{AP}}$ is a labeling function.

$TS$ is called finite if $S$, $\text{Act}$, and $\text{AP}$ are finite.
Transition Systems

the likelihood with which a certain transition is selected. Similarly, when the set of initial states consists of more than one state, the start state is selected nondeterministically.

The labeling function \( L \) relates a set \( L(s) \in 2^{AP} \) of atomic propositions to any state \( s \).

Intuitively stand for exactly those atomic propositions \( a \in AP \) which are satisfied by state \( s \). Given that \( \Phi \) is a propositional logic formula, then \( s \) satisfies the formula \( \Phi \) if the evaluation induced by \( L(s) \) makes the formula \( \Phi \) true; that is:

\[
\Phi \text{ if } L(s) = \Phi.
\]

(Basic principles of propositional logic are explained in Appendix A.3, see page 915 ff.)

Example 2.2. Beverage Vending Machine

We consider an (somewhat foolish) example, which has been established as standard in the field of process calculi. The transition system in Figure 2.1 models a preliminary design of a beverage vending machine. The machine can either deliver beer or soda. States are represented by ovals and transitions by labeled edges. State names are depicted inside the ovals. Initial states are indicated by having an incoming arrow without source.

![Transition System Diagram](image)

The state space is \( S = \{ pay, select, soda, beer \} \). The set of initial states consists of only one state, i.e., \( I = \{ pay \} \). The (user) action \( insert \text{ coin} \) denotes the insertion of a coin, while the (machine) actions \( get \text{ soda} \) and \( get \text{ beer} \) denote the delivery of soda and beer, respectively. Transitions of which the action label is not of further interest here, e.g., as it denotes some internal activity of the beverage machine, are all denoted by the distinguished action symbol \( \tau \). We have:

\[
\text{Act} = \{ insert \text{ coin}, get \text{ soda}, get \text{ beer}, \tau \}.
\]

Some example transitions are:

\[
pay \xrightarrow{insert \text{ coin}} select \quad \text{and} \quad get \text{ beer} \xrightarrow{get \text{ beer}} pay.
\]

Recall that \( 2^{AP} \) denotes the power set of \( AP \).
Figure 2.3: Transition system modeling the extended beverage vending machine.
Second Step: We need a formal Specification!
There are 5 philosophers at a table sharing 5 chopsticks for eating. Each philosopher needs two chopsticks to eat.

At each point in time at most one of two neighbouring philosophers can eat.
There are 5 philosophers at a table sharing 5 chopsticks for eating.

Each philosopher needs two chopsticks to eat.

At each point in time at most one of two neighbouring philosophers can eat.

Classic deadlock scenario example!
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from $Q_0$.

Solution?
Problem: given an TS, and a target set T, is T reachable from Q₀.

Solution? Depth First Search, in O(n+m) time.
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from $Q_0$.

Solution?

Depth First Search, in $O(n+m)$ time.

**DFS(q)**

Add q to visited_states;
for each q’ such that q $\rightarrow a$ q’
if q’ in T
print "YES!"; halt;
else if q’ not in visited_states
DFS(q’)
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from Q₀.

Solution? Depth First Search, in O(n+m) time.
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from Q₀.

Solution? Depth First Search, in O(n+m) time.

What if we are interested in more sophisticated properties?
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from Q₀.

Solution? Depth First Search, in $O(n+m)$ time.

What if we are interested in more sophisticated properties?

Suggest a non-reachability property for philosophers!
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from $Q_0$.

Solution? Depth First Search, in $O(n+m)$ time.

What if we are interested in more sophisticated properties?

Suggest a non-reachability property for philosophers!
REACHABILITY

Problem: given an TS, and a target set T, is T reachable from $Q_0$.

Solution? Depth First Search, in $O(n+m)$ time.

What if we are interested in more sophisticated properties?

Suggest a non-reachability property for philosophers!

The light will always eventually turn green.