Model Checking
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?

- Doesn’t aim too high!
- Originally restricted to finite-state systems.
- Applicable to systems with “short” descriptions.
- Control-oriented systems such as hardware, protocols, ...
- 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 $(S, Act, \rightarrow, I, AP, L)$ where

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

$TS$ is called finite if $S$, $Act$, and $AP$ are finite.
Example

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 stands for exactly those 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:

$s \models \Phi$ if $L(s) \models \Phi$.

(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.

Figure 2.1: A transition system of a simple beverage vending machine.

The state space is $S = \{\text{pay}, \text{select}, \text{soda}, \text{beer}\}$. The set of initial states consists of only one state, i.e., $I = \{\text{pay}\}$. The (user) action insert coin denotes the insertion of a coin, while the (machine) actions get soda and get 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:

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

Some example transitions are:

$\text{pay} \xrightarrow{\text{insert coin}} \text{select}$ and $\text{pay} \xrightarrow{\text{get beer}} \text{pay}$.

1 Recall that $2^{AP}$ denotes the power set of $AP$. 


Example

Figure 2.3: Transition system modeling the extended beverage vending machine.
Second Step: We need a formal Specification!
Example: Dining Philosophers

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? 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.
Option 1 for properties beyond reachability ...
Given a TS $M$ for the model and a TS $S$ for the specification:

Question: Is every behaviour of $M$ a behaviour of $S$?

$L(M) \subseteq L(S)$

Solvable in PSpace: linear in $M$ and exponential in $S$. 
Best choice: new logic!
Alternative: Temporal Logic

- Language for describing properties of infinite sequences.
- Extension of propositional logic.
- Uses temporal operators to describe sequencing properties.
Linear Temporal Logic
LTL Syntax

\[ \varphi ::= \text{true} \mid a \mid \varphi_1 \land \varphi_2 \mid \neg \varphi \mid \Diamond \varphi \mid \varphi_1 \mathcal{U} \varphi_2 \]

\[ a \in AP \]

\[ \Diamond \varphi \overset{\text{def}}{=} \text{true} \mathcal{U} \varphi \quad \Box \varphi \overset{\text{def}}{=} \neg \Diamond \neg \varphi \]
Before proceeding with the formal semantics of LTL, we present some examples.

**Example 5.2. Properties for the Mutual Exclusion Problem**

Consider the mutual exclusion problem for two concurrent processes $P_1$ and $P_2$, say. Process $P_i$ is modeled by three locations: (1) the noncritical section, (2) the waiting phase which is entered when the process intends to enter the critical section, and (3) the critical section. Let the propositions $\text{wait}_i$ and $\text{crit}_i$ denote that process $P_i$ is in its waiting phase and critical section, respectively.

The safety property stating that $P_1$ and $P_2$ never simultaneously have access to their critical sections can be described by the LTL-formula:

$$□(¬\text{crit}_1 ∨ ¬\text{crit}_2)$$

This formula expresses that always ($□$) at least both processes are not in their critical sections ($¬\text{crit}_i$).

The liveness requirement stating that each process $P_i$ is infinitely often in its critical section can be described by the LTL-formula:

$$\text{always} □ a$$

where $a$ represents the critical section.
LTL Semantics

LTL is interpreted over paths.

These paths are (infinite) words labeled with subset of the atomic propositions (AP) that are true at each letter.

\[
\begin{align*}
\sigma & \models \text{true} \\
\sigma & \models a \quad \text{iff} \quad a \in A_0 \quad (\text{i.e., } A_0 \models a) \\
\sigma & \models \varphi_1 \land \varphi_2 \quad \text{iff} \quad \sigma \models \varphi_1 \text{ and } \sigma \models \varphi_2 \\
\sigma & \models \neg \varphi \quad \text{iff} \quad \sigma \not\models \varphi \\
\sigma & \models \bigcirc \varphi \quad \text{iff} \quad \sigma[1...] = A_1A_2A_3... \models \varphi \\
\sigma & \models \varphi_1 \bigcup \varphi_2 \quad \text{iff} \quad \exists j \geq 0. \ \sigma[j...] \models \varphi_2 \text{ and } \sigma[i...] \models \varphi_1, \text{ for all } 0 \leq i < j
\end{align*}
\]

LTL’s \(\models\) is the smallest relation satisfying the above rules.
\( \sigma \models \Diamond \varphi \) iff \( \exists j \geq 0. \; \sigma[j\ldots] \models \varphi \)

\( \sigma \models \Box \varphi \) iff \( \forall j \geq 0. \; \sigma[j\ldots] \models \varphi \)

\( \sigma \models \Box \Diamond \varphi \) iff \( \exists j. \; \sigma[j\ldots] \models \varphi \)

\( \sigma \models \Diamond \Box \varphi \) iff \( \forall j. \; \sigma[j\ldots] \models \varphi \)
More Examples in Class