Topics in Verification

AZADEH FARZAN
WINTER 2016
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?
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 $(S, Act, \rightarrow, I, AP, L)$ 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.
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 \( s \).

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.

![Diagram of a beverage vending machine](image)

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:

\[ \text{Act} = \{ \text{insert\_coin}, \text{get\_soda}, \text{get\_beer}, \tau \} \]

Some example transitions are:

\[ \text{pay} \xrightarrow{\text{insert\_coin}} \text{select} \]

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!
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.
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?
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.
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?
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?

The light will always eventually turn green.
ONE TS AS A SPEC FOR ANOTHER TS!

Given a TS M for the model and a TS S for the specification:
ONE TS AS A SPEC FOR ANOTHER TS!

Given a TS M for the model and a TS S for the specification:

Question: Is every behaviour of M a behaviour of S?
ONE TS AS A SPEC FOR ANOTHER TS!

Given a TS M for the model and a TS S for the specification:

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

\[ L(M) \subseteq L(S) \]
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 $S$. 

**ONE TS AS A SPEC FOR ANOTHER TS!**
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

\[ \varphi ::= \text{true} \mid a \mid \varphi_1 \land \varphi_2 \mid \neg \varphi \mid \varnothing \varphi \mid \varphi_1 \mathbf{U} \varphi_2 \]
LTL SYNTAX

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

\( a \in AP \)
The until operator allows to derive the temporal modalities eventually.

Other Boolean connectives such as disjunction \( \lor \) and conjunction \( \land \) stand for the state labels \( a \) and \( b \), respectively. The temporal operator \( \square \) is a unary prefix operator and requires a single LTL formula as argument. Formula \( \phi \) holds at all future states.

The Boolean connectives \( \lor \), \( \land \), and \( \neg \) bind equally strong. The temporal operator \( \square \) binds more strongly than the binary ones. The until operator \( \phi_1 U \phi_2 \) stands for the state label \( \phi_2 \) at which an \( \phi_1 \)-state is visited. This thus amounts to \( \phi_1 \) eventually \( \phi_2 \).

The \( \diamond \) operator (or: exclusive or) is defined as \( \diamond \phi = \text{true} U \phi \). This is equivalent to the fact that \( \phi \) holds infinitely often, that is, there is a moment \( j \) such that \( \phi \) holds at the current moment, if there is some future moment for which \( \phi \) holds. The \( \boxdot \) operator is defined as \( \boxdot \phi = \neg \diamond \neg \phi \).

The until operator expresses that \( \phi_1 \) occurs infinitely often before \( \phi_2 \) occurs. The \( \diamond \) operator expresses that \( \phi \) holds eventually in the future.
LTL: INTUITION
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:

$$\Box(\neg \text{crit}_1 \lor \neg \text{crit}_2)$$

This formula expresses that always ($\Box$) at least one of the two processes is not in the critical section ($\neg \text{crit}_i$).

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

$$\Diamond a$$

Figure 5.1: Intuitive semantics of temporal modalities.
LTL is interpreted over paths.

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

\[ \sigma \models \text{true} \]

\[ \sigma \models a \quad \text{iff} \quad a \in A_0 \quad \text{(i.e.,} \ A_0 \models a) \]

\[ \sigma \models \varphi_1 \land \varphi_2 \quad \text{iff} \quad \sigma \models \varphi_1 \quad \text{and} \quad \sigma \models \varphi_2 \]

\[ \sigma \models \neg \varphi \quad \text{iff} \quad \sigma \not\models \varphi \]

\[ \sigma \models \Diamond \varphi \quad \text{iff} \quad \sigma[1...]=A_1A_2A_3... \models \varphi \]

\[ \sigma \models \varphi_1 U \varphi_2 \quad \text{iff} \quad \exists j \geq 0. \ \sigma[j...] \models \varphi_2 \quad \text{and} \quad \sigma[i...] \models \varphi_1, \text{ for all } 0 \leq i < j \]
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.

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

LTL’s \( \models \) is the smallest relation satisfying the above rules.
The semantics of the combinations of  

\[ \forall j \phi \]  

stands for  

\[ (\phi) \]  

follows from:  

\[ \sigma \models \Diamond \varphi \iff \Diamond \sigma \varphi \]  

\[ \sigma \models \Box \varphi \iff \Box \sigma \varphi \]  

\[ \sigma \models \Box \Diamond \varphi \iff \Diamond \Box \sigma \varphi \]  

\[ \sigma \models \Diamond \Box \varphi \iff \Box \Diamond \sigma \varphi \]
The semantics of the combinations of LTL-formulae with respect to a satisfaction relation \( \sigma \) can be regarded in order to be able to refer to the truth-value of the subformula \( \varphi \).

For the derived operators

\[ \sigma \models \Diamond \varphi \text{ iff } \exists j \geq 0. \sigma[j...] \models \varphi \]

\[ \sigma \models \Box \varphi \text{ iff } \forall j \geq 0. \sigma[j...] \models \varphi \]

\[ \sigma \models \Box \Diamond \varphi \text{ iff } \]

\[ \sigma \models \Diamond \Box \varphi \text{ iff } \]
\[\sigma \models \Diamond \varphi \iff \exists j \geq 0. \, \sigma[j...] \models \varphi\]

\[\sigma \models \Box \varphi \iff \forall j \geq 0. \, \sigma[j...] \models \varphi\]

\[\sigma \models \Box \Diamond \varphi \iff \exists j. \, \sigma[j...] \models \varphi\]

\[\sigma \models \Diamond \Box \varphi \iff \forall j. \, \sigma[j...] \models \varphi\]
EXAMPLES

mutual exclusion:
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:

$$\square (\neg \text{crit}_1 \lor \neg \text{crit}_2)$$

This formula expresses that always ($\square$) at least one of the two processes is not in its critical section ($\neg \text{crit}_i$).
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:

$$\Box (\neg \text{crit}_1 \lor \neg \text{crit}_2)$$

This formula expresses that always ($\Box$) at least one of the two processes is not in its critical section ($\neg \text{crit}_i$).

The liveness requirement stating that each process $P_i$ is infinitely often in its critical section mutual exclusion:

once red, the light cannot become green immediately:
EXAMPLES

mutual exclusion:

\[ \Box ( \neg \text{crit}_1 \lor \neg \text{crit}_2 ) \]

once red, the light cannot become green immediately:

\[ \Box ( \text{red} \rightarrow \neg \bigcirc \text{green} ) \]
EXAMPLES

mutual exclusion:
\[ \Box (\neg \text{crit}_1 \lor \neg \text{crit}_2) \]

once red, the light cannot become green immediately:
\[ \Box (\text{red} \rightarrow \neg \Diamond \text{green}) \]

every request will eventually lead to a response:
EXAMPLES

mutual exclusion: \( \Box (\neg \text{crit}_1 \lor \neg \text{crit}_2) \)

once red, the light cannot become green immediately: \( \Box (\text{red} \rightarrow \neg \Diamond \text{green}) \)

every request will eventually lead to a response: \( \Box (\text{request} \rightarrow \Diamond \text{response}) \)
EXAMPLES

mutual exclusion:

\[ \square ( \neg \text{crit}_1 \lor \neg \text{crit}_2 ) \]

once red, the light cannot become green immediately:

\[ \square ( \text{red} \rightarrow \neg \lozenge \text{green} ) \]

ev\ry request will eventually lead to a response:

\[ \square ( \text{request} \rightarrow \Diamond \text{response} ) \]

once red, the light always becomes green eventually after being yellow for some time:
EXAMPLES

mutual exclusion: \(\Box(\neg crit_1 \lor \neg crit_2)\)

once red, the light cannot become green immediately: \(\Box(red \rightarrow \neg \Diamond green)\)

every request will eventually lead to a response: \(\Box(request \rightarrow \Diamond response)\)

once red, the light always becomes green eventually after being yellow for some time:
\(\Box(red \rightarrow \Diamond (red \lor (yellow \land \Diamond (yellow \lor green))))\)
LTL semantics can be lifted from paths to states and transition systems.
LTL semantics can be lifted from paths to states and transition systems.

\[ s \models \phi \iff \forall \pi : \pi[0] = s \implies \pi \models \phi \]
LTL semantics can be lifted from paths to states and transition systems.

\[ s \models \phi \iff \forall \pi : \pi[0] = s \implies \pi \models \phi \]

\[ TS \models \phi \iff \forall s \in I : s \models \phi \]
EXAMPLES

Thus, \( TS \models \phi \) if and only if \( s_0 \models \phi \) for all initial states \( s_0 \) of \( TS \).

\[
\begin{align*}
\{a, b\} & \quad s_1 \quad \{a, b\} \\
\{a, b\} & \quad s_2 \quad \{a\} \\
\{a\} & \quad s_3
\end{align*}
\]

Figure 5.3: Example for semantics of LTL.

Example 5.8. Semantics of LTL

Consider the transition system \( TS \) depicted in Figure 5.3 with the set of propositions \( AP = \{a, b\} \). For example, we have that \( TS \models \Box a \), since all states are labelled with \( a \), and hence, all traces of \( TS \) are words of the form \( A_0 A_1 A_2 ... \) with \( a \in A_i \) for all \( i \geq 0 \).

Thus, \( s_i \models \Box a \) for \( i = 1, 2, 3 \). Moreover:

\( s_1 \models \Diamond (a \land b) \) since \( s_2 \models a \land b \) and \( s_2 \) is the only successor of \( s_1 \), \( s_3 \not\models \Diamond (a \land b) \) as \( s_3 \in \text{Post}(s_2) \) and \( s_3 \not\models a \land b \).

This yields \( TS \not\models \Diamond (a \land b) \) as \( s_3 \) is an initial state for which \( s_3 \not\models \Diamond (a \land b) \). As another example:

\( TS \models \Box (\neg b \rightarrow \Box (a \land \neg b)) \), since \( s_3 \) is the only \( \neg b \) state, \( s_3 \) cannot be left anymore, and \( a \land \neg b \) in \( s_3 \) is true. However, \( TS \not\models b \lor (a \land \neg b) \), since the initial path \( (s_1 s_2) \omega \) does not visit a state for which \( a \land \neg b \) holds.

Remark 5.9. Semantics of Negation

For paths, it holds \( \pi \models \phi \) if and only if \( \pi \not\models \neg \phi \). This is due to the fact that Words(\neg \phi) = \( 2^{AP} \) \setminus \text{Words}(\phi) \).

However, the statements \( TS \not\models \phi \) and \( TS \models \neg \phi \) are not equivalent in general. Instead, we have \( TS \models \neg \phi \) implies \( TS \not\models \phi \). Note that \( TS \not\models \phi \) iff Traces(\( TS \)) \not\subseteq \text{Words}(\phi) \).

Note that \( TS \not\models \phi \) iff Traces(\( TS \)) \not= \emptyset \).
Consider the transition system $\text{TS}$.

**Example 5.8. Semantics of LTL**

For paths, it holds $\text{Post}(\pi) = \emptyset$ if and only if $\text{Post}(\pi) = \emptyset$. Thus, $s_i = s_i$ implies $\phi$.

**Remark 5.9. Semantics of Negation**

However, the statements $\phi$ are words of the form $\omega$. Note that $\phi$ are words of the form $\omega$. Hence, all traces of the initial path $(\omega)$. And hence, all traces $(\omega)$.

For all initial states $s_i$, $s_i$ is the only successor of $s_i$. Moreover:

$$\forall s_i, s_i = s_i$$

The transition system is an initial state for which $\phi$. Therefore, that a fact that $s_i$ is an initial state for which $\phi$. Since $s_i$ cannot be left anymore, and $\text{Post}(\pi)$ is an initial state for which $\phi$. Therefore, that a fact that $s_i$ is an initial state for which $\phi$.

Consider the transition system $\text{TS}$.

**Example 5.8. Semantics of LTL**

For paths, it holds $\text{Post}(\pi) = \emptyset$ if and only if $\text{Post}(\pi) = \emptyset$. Thus, $s_i = s_i$ implies $\phi$.

**Remark 5.9. Semantics of Negation**

However, the statements $\phi$ are words of the form $\omega$. Note that $\phi$ are words of the form $\omega$. Hence, all traces of the initial path $(\omega)$. And hence, all traces $(\omega)$.

For all initial states $s_i$, $s_i$ is the only successor of $s_i$. Moreover:

$$\forall s_i, s_i = s_i$$

The transition system is an initial state for which $\phi$. Therefore, that a fact that $s_i$ is an initial state for which $\phi$. Since $s_i$ cannot be left anymore, and $\text{Post}(\pi)$ is an initial state for which $\phi$.

Consider the transition system $\text{TS}$.

**Example 5.8. Semantics of LTL**

For paths, it holds $\text{Post}(\pi) = \emptyset$ if and only if $\text{Post}(\pi) = \emptyset$. Thus, $s_i = s_i$ implies $\phi$.

**Remark 5.9. Semantics of Negation**

However, the statements $\phi$ are words of the form $\omega$. Note that $\phi$ are words of the form $\omega$. Hence, all traces of the initial path $(\omega)$. And hence, all traces $(\omega)$.

For all initial states $s_i$, $s_i$ is the only successor of $s_i$. Moreover:

$$\forall s_i, s_i = s_i$$

The transition system is an initial state for which $\phi$. Therefore, that a fact that $s_i$ is an initial state for which $\phi$. Since $s_i$ cannot be left anymore, and $\text{Post}(\pi)$ is an initial state for which $\phi$.

Consider the transition system $\text{TS}$.

**Example 5.8. Semantics of LTL**

For paths, it holds $\text{Post}(\pi) = \emptyset$ if and only if $\text{Post}(\pi) = \emptyset$. Thus, $s_i = s_i$ implies $\phi$.

**Remark 5.9. Semantics of Negation**

However, the statements $\phi$ are words of the form $\omega$. Note that $\phi$ are words of the form $\omega$. Hence, all traces of the initial path $(\omega)$. And hence, all traces $(\omega)$.

For all initial states $s_i$, $s_i$ is the only successor of $s_i$. Moreover:

$$\forall s_i, s_i = s_i$$

The transition system is an initial state for which $\phi$. Therefore, that a fact that $s_i$ is an initial state for which $\phi$. Since $s_i$ cannot be left anymore, and $\text{Post}(\pi)$ is an initial state for which $\phi$.
**EXAMPLES**

\[
s_1 \models \Box (a \land b) \quad \text{and} \quad s_2 \not\models \Box (a \land b)
\]
Consider the transition system since the initial path (s, 0) as satisfies ¬1(a ∧ b). However, words (s, 0) cannot be left anymore, and traces (s, 0) ∩ TS = ∅. For example, we have that state a is true. Instead, we equivalent in general. Instead, we label with □, since a label with □.
EXAMPLES

\[ s_1 \models \Diamond (a \land b) \]
\[ s_2 \not\models \Diamond (a \land b) \]
\[ TS \not\models \Diamond (a \land b) \]
\[ s_3 \not\models \Diamond (a \land b) \]
NEGATION?

\[ \pi \models \phi \iff \pi \not\models \neg \phi \]
NEGATION?

\[ \pi \models \phi \iff \pi \not\models \neg \phi \]

This is for paths, what about transition systems?
This is for paths, what about transition systems?
To illustrate this, consider the transition system depicted in Figure 5.5 which represents a semaphore-based solution to the mutual exclusion problem; see also Example 3.9 on page 98. Each state of the system can be represented as follows:

- \( s_1 \) with input \( \{a\} \) and transition \( TS \not\models \diamond a \)
- \( s_0 \) with no input \( \emptyset \) and transition \( TS \not\models \neg \diamond a \)
- \( s_2 \) with no input \( \emptyset \)

Thus, it is possible that a transition system (or a state) satisfies neither \( \phi \) nor \( \lnot \phi \). This is caused by the fact that there might be paths in the system. In this case, consider the transition system depicted in Figure 5.5.

\[
\pi \models \phi \iff \pi \not\models \lnot \phi
\]

This is for paths, what about transition systems?
**EQUIVALENCE OF LTL FORMULAE**

**Definition.** two LTL formulas are equivalent iff:

\[ \forall \pi : \pi \models \phi_1 \iff \pi \models \phi_2 \]
### Equivalence of LTL Formulae

The equivalence of LTL (Linear Temporal Logic) formulae is determined by the truth values of the formulae under all interpretations. Two LTL formulae are said to be equivalent if they have the same truth-value under all interpretations. For example, it seems useless to distinguish between $\neg
\neg a$ and $a$, although these formulae are syntactically different.

**Definition 5.17. Equivalence of LTL Formulae**

Two LTL formulae $\phi_1, \phi_2$ are equivalent, denoted $\phi_1 \equiv \phi_2$, if $|\phi_1| = |\phi_2|$.

As LTL subsumes propositional logic, equivalences of propositional logic also hold for LTL, e.g., $\neg \neg \phi \equiv \phi$ and $\phi \land \phi \equiv \phi$.

In addition, there exist a number of equivalence laws for temporal modalities. These include the equivalence laws indicated in Figure 5.7. We explain some of these equivalence laws.

- **Duality Rule**
  
  $\neg \Box \phi \equiv \Box \neg \phi$

- **Idempotency Law**
  
  $\Diamond \Diamond \phi \equiv \Diamond \phi$

- **Absorption Law**

  $\Diamond \Box \Diamond \phi \equiv \Box \Diamond \Diamond \phi$

  $(\Diamond \phi \land \Diamond \psi) \equiv \Diamond (\phi \land \Diamond \psi)$

- **Expansion Law**

  $\Diamond (\phi \lor \psi) \equiv \Diamond \phi \lor \Diamond \psi$

  $\Box (\phi \land \Box \psi) \equiv \Box \phi \land \Box \Box \psi$

- **Distributive Law**

  $\Diamond (\phi \lor \psi) \equiv (\Diamond \phi) \lor (\Diamond \psi)$

  $\Box (\phi \land \Box \psi) \equiv (\Box \phi) \land (\Box \Box \psi)$

  $\Diamond (\phi \land \psi) \equiv (\Diamond \phi) \land (\Diamond \psi)$

  $\Box (\phi \land \Box \psi) \equiv (\Box \phi) \land (\Box \Box \psi)$
EXPANSION LAWS

\[ \varphi U \psi \equiv \psi \lor (\varphi \land \Box (\varphi U \psi)) \]
\[ \Diamond \psi \equiv \psi \lor \Box \Diamond \psi \]
\[ \Box \psi \equiv \psi \land \Box \Box \psi \]
Lemma. Until is the least solution to the expansion law.

\[
\begin{align*}
\varphi \mathbin{U} \psi &\equiv \psi \lor (\varphi \land \bigcirc (\varphi \mathbin{U} \psi)) \\
\Diamond \psi &\equiv \psi \lor \bigcirc \Diamond \psi \\
\Box \psi &\equiv \psi \land \bigcirc \Box \psi
\end{align*}
\]
EXPANSION LAWS

Lemma. Until is the least solution to the expansion law.

The following equation has many solutions:

$$X = \psi \lor (\phi \land \Diamond X)$$

Until is the smallest set that satisfies this equation.
Lemma. Until is the least solution to the expansion law.

The following equation has many solutions:

\[ X = \psi \lor (\phi \land \diamond X) \]

Until is the smallest set that satisfies this equation.

Note that we are using the notions of sets (of paths) and formulas interchangeably, by referring to the set of paths that satisfy a given formula.
CATEGORIZATION OF LTL PROPERTIES

☐ Safety : nothing bad should happen.

☐ Liveness : something good will happen in future.
CATEGORIZATION OF LTL PROPERTIES

☐ Safety: nothing bad should happen.

• Invariants: a condition $\Phi$ holds for all reachable states.

☐ Liveness: something good will happen in future.
CATEGORIZATION OF LTL PROPERTIES

- **Safety**: nothing bad should happen.

- **Invariants**: a condition $\Phi$ holds for all reachable states.

  $\Phi = \neg \text{crit}_1 \lor \neg \text{crit}_2$

- **Liveness**: something good will happen in future.
CATEGORIZATION OF LTL PROPERTIES

- **Safety**: nothing bad should happen.

- **Invariants**: a condition $\Phi$ holds for all reachable states.

- **Safety Properties**: any infinite path does not have a bad finite prefix.

- **Liveness**: something good will happen in future.

\[ \Phi = \neg \text{crit}_1 \lor \neg \text{crit}_2 \]
CATEGORIZATION OF LTL PROPERTIES

- **Safety**: nothing bad should happen.

- **Invariants**: a condition $\Phi$ holds for all reachable states.

  $\Phi = \neg \text{crit}_1 \lor \neg \text{crit}_2$

- **Safety Properties**: any infinite path does not have a bad finite prefix.

  $\square (\text{red} \rightarrow \neg \lozenge \text{green})$

- **Liveness**: something good will happen in future.
CATEGORIZATION OF LTL PROPERTIES

Safety: nothing bad should happen.

- Invariants: a condition \( \Phi \) holds for all reachable states.

- Safety Properties: any infinite path does not have a bad finite prefix.

Liveness: something good will happen in future.

\[ \Phi = \neg \text{crit}_1 \lor \neg \text{crit}_2 \]

\[ \Box(\text{red} \rightarrow \neg \circ \text{green}) \]

\[ \Box(\text{request} \rightarrow \Diamond \text{response}) \]
CATEGORIZATION OF LTL PROPERTIES

☐ **Safety**: nothing bad should happen.

- **Invariants**: a condition $\Phi$ holds for all reachable states.

- **Safety Properties**: any infinite path does not have a bad finite prefix.

☐ **Liveness**: something good will happen in future.

Any finite prefix can be extended to a trace that satisfies the property.
CATEGORIZATION OF LTL PROPERTIES

☐ Safety and Liveness are almost disjoint.
CATEGORIZATION OF LTL PROPERTIES

Safety and Liveness are almost disjoint.

set of all paths over AP is both safety and liveness!
CATEGORIZATION OF LTL PROPERTIES

☐ Safety and Liveness are almost disjoint.

set of all paths over AP is both safety and liveness!

☐ Every linear time property can be written in terms of a safety property and a liveness property.

\[ P = P_{\text{Safe}} \cup P_{\text{Live}} \]
FAIRNESS

```plaintext
global nat s, t
local nat m
while(true):
    m=t++ // Acquire a ticket
while(m>s): // Busy wait
    skip
// Critical section
s++ // Exit critical
```
unconditional LTL fairness:

every process gets its turn infinitely often.
unconditional LTL fairness:

\[ \square \Diamond \psi \]

every process gets its turn infinitely often.

Definition 5.25. LTL Fairness Constraints and Assumptions

Let \( \Phi \) and \( \Psi \) be propositional logic formulae over \( \text{AP} \).

1. An unconditional LTL fairness constraint is an LTL formula of the form:

\[ \text{ufair} = \square \Diamond \Psi \]

2. A strong LTL fairness condition is an LTL formula of the form:

\[ \text{sfair} = \square \Diamond \Phi \rightarrow \square \Diamond \Psi \]

3. A weak LTL fairness constraint is an LTL formula of the form:

\[ \text{wfair} = \Diamond \square \Phi \rightarrow \square \Diamond \Psi \]

An LTL fairness assumption is a conjunction of LTL fairness constraints (of any arbitrary type).

For instance, a strong LTL fairness assumption denotes a conjunction of strong LTL fairness constraints, i.e., a formula of the form:

\[ \bigwedge_{0 \leq i \leq k} (\square \Diamond \Phi_i \rightarrow \square \Diamond \Psi_i) \]

for propositional logical formulae \( \Phi_i \) and \( \Psi_i \) over \( \text{AP} \). Weak and unconditional LTL fairness assumptions are defined in a similar way.
unconditional LTL fairness:

\[ \Box \Diamond \Psi \]

every process gets its turn infinitely often.

strong LTL fairness:

\[ \Box \Diamond \Phi \rightarrow \Box \Diamond \Psi \]

every process that is enabled infinitely often gets its turn infinitely often.
unconditional LTL fairness:

\[
\square \Diamond \Psi
\]

every process gets its turn infinitely often.

strong LTL fairness:

\[
\square \Diamond \Phi \quad \rightarrow \quad \square \Diamond \Psi
\]

every process that is enabled infinitely often gets its turn infinitely often.
**FAIRNESS**

unconditional LTL fairness:

\[ \Box \Diamond \Psi \]

*every process gets its turn infinitely often.*

strong LTL fairness:

\[ \Box \Diamond \Phi \rightarrow \Box \Diamond \Psi \]

*every process that is enabled infinitely often gets its turn infinitely often.*

weak LTL fairness:

*every process that is continuously enabled from a certain time instant on gets its turn infinitely often.*
unconditional LTL fairness:

\[ \square \Diamond \Psi \]

every process gets its turn infinitely often.

strong LTL fairness:

\[ \square \Diamond \Phi \rightarrow \square \Diamond \Psi \]

every process that is enabled infinitely often gets its turn infinitely often.

weak LTL fairness:

\[ \Diamond \square \Phi \rightarrow \square \Diamond \Psi \]

every process that is continuously enabled from a certain time instant on gets its turn infinitely often.
unconditional LTL fairness:

**every process gets its turn infinitely often.**

\[ \square \Diamond \psi \]

strong LTL fairness:

**every process that is enabled infinitely often gets its turn infinitely often.**

\[ \square \Diamond \phi \rightarrow \square \Diamond \psi \]

weak LTL fairness:

**every process that is continuously enabled from a certain time instant on gets its turn infinitely often.**

\[ \Diamond \square \phi \rightarrow \square \Diamond \psi \]

**Fair satisfaction relation:** \( TS \models F \phi \)
Corollary 3.18. Trace Equivalence and LT Properties

Let $TS$ and $TS'$ be transition systems without terminal states and with the same set of atomic propositions. Then:

$$\text{Traces}(TS) = \text{Traces}(TS') \iff TS \text{ and } TS' \text{ satisfy the same LT properties.}$$

There thus does not exist an LT property that can distinguish between trace-equivalent transition systems. Stated differently, in order to establish that the transition systems $TS$ and $TS'$ are not trace-equivalent it suffices to find one LT property that holds for one but not for the other.

Example 3.19. Two Beverage Vending Machines

Consider the two transition systems in Figure 3.8 that both model a beverage vending machine. For simplicity, the observable action labels of transitions have been omitted. Both machines are able to offer soda and beer. The left transition system models a beverage machine that after insertion of a coin nondeterministically chooses to either provide soda or beer. The right one, however, has two selection buttons (one for each beverage), and after insertion of a coin, nondeterministically blocks one of the buttons. In either case, the user has no control over the beverage obtained—the choice of beverage is under full control of the vending machine.

Let $AP = \{\text{pay}, \text{soda}, \text{beer}\}$. Although the two vending machines behaved differently, it is not difficult to see that they exhibit the same traces when considering $AP$, as for both machines traces are alternating sequences of pay and either soda or beer. The vending machines are thus trace-equivalent. By Corollary 3.18 both vending machines satisfy exactly the same LT properties. Stated differently, it means that there does not exist an LT property that distinguishes between the two vending machines.
These two transition systems satisfy the same set of LTL formulas. But they function in different ways.
LIMITATIONS OF LTL

Corollary 3.18. Trace Equivalence and LT Properties

Let \( TS \) and \( TS' \) be transition systems without terminal states and with the same set of atomic propositions. Then:

\[
\text{Traces}(TS) = \text{Traces}(TS') \iff \text{TS and TS' satisfy the same LT properties.}
\]

There thus does not exist an LT property that can distinguish between trace-equivalent transition systems. Stated differently, in order to establish that the transition systems \( TS \) and \( TS' \) are not trace-equivalent it suffices to find one LT property that holds for one but not for the other.

Example 3.19. Two Beverage Vending Machines

Consider the two transition systems in Figure 3.8 that both model a beverage vending machine. For simplicity, the observable action labels of transitions have been omitted. Both machines are able to offer soda and beer. The left transition system models a beverage machine that after insertion of a coin nondeterministically chooses to either provide soda or beer. The right one, however, has two selection buttons (one for each beverage), and after insertion of a coin, nondeterministically blocks one of the buttons. In either case, the user has no control over the beverage obtained—the choice of beverage is under full control of the vending machine.

Let \( AP = \{\text{pay}, \text{soda}, \text{beer}\} \).

Although the two vending machines behave differently, it is not difficult to see that they exhibit the same traces when considering \( AP \), as for both machines traces are alternating sequences of \( \text{pay} \) and either \( \text{soda} \) or \( \text{beer} \). The vending machines are thus trace-equivalent. By Corollary 3.18 both vending machines satisfy exactly the same LT properties. Stated differently, it means that there does not exist an LT property that distinguishes between the two vending machines.

These two transition systems satisfy the same set of LTL formulas. But they function in different ways.

They are trace equivalent
STEP 3: MODEL CHECKING AGAINST AN LTL PROPERTY
WE NEED TO LEARN A BIT ABOUT AUTOMATA ON INFINITE WORDS ...