Model Checking General Temporal Formulas

Next, we consider methods for model checking general LTL formulas.

Let $\mathcal{D}$ be an FDS and φ an LTL formula. Assume we wish to check whether $\mathcal{D} \models \varphi$. We proceed along the following steps:

- Construct the temporal tester $T(\neg\varphi)$. This is an FDS whose computations are all the sequences falsifying φ.
- Form the parallel composition $\mathcal{D} \parallel\!| \; T(\neg\varphi)$. This is an FDS whose computations are all computations of $\mathcal{D}$ which violate φ.
- Check whether the composition $\mathcal{D} \parallel\!| \; T(\neg\varphi)$ is feasible. $\mathcal{D} \models \varphi$ iff $\mathcal{D} \parallel\!| \; T(\neg\varphi)$ is infeasible.

It only remains to describe the construction of a tester $T(\psi)$ for a general LTL formula ψ.

Operations on FDS's: Asynchronous Parallel Composition

The asynchronous parallel composition of systems $\mathcal{D}_1$ and $\mathcal{D}_2$, denoted by $\mathcal{D}_1 \parallel \mathcal{D}_2$, is given by $\mathcal{D} = \langle V, \Theta, \rho, \mathcal{J}, \mathcal{C} \rangle$, where

$$
\begin{array}{ccl}
V & = & V_1 \quad \cup \quad V_2 \\
\Theta & = & \Theta_1 \quad \wedge \quad \Theta_2 \\
\rho & = & \left(\begin{array}{cl} & (\rho_1 \wedge \mathit{pres}(V_2 - V_1)) \\ \vee & (\rho_2 \wedge \mathit{pres}(V_1 - V_2)) \end{array}\right) \\
\mathcal{J} & = & \mathcal{J}_1 \quad \cup \quad \mathcal{J}_2 \\
\mathcal{C} & = & \mathcal{C}_1 \quad \cup \quad \mathcal{C}_2
\end{array}
$$

The predicate $\mathit{pres}(U)$ stands for the assertion $U' = U$, implying that all the variables in U are preserved by the transition.

Asynchronous parallel composition represents the interleaving-based concurrency which is the assumed concurrency in shared-variables models.

Claim 5. $\mathcal{D}(P_1 \parallel P_2) \quad \sim \quad \mathcal{D}(P_1) \parallel \mathcal{D}(P_2)$

That is, the FDS corresponding to the program $P_1 \parallel P_2$ is equivalent to the asynchronous parallel composition of the FDS's corresponding to P_1 and P_2.

Synchronous Parallel Composition

The synchronous parallel composition of systems $\mathcal{D}_1$ and $\mathcal{D}_2$, denoted by $\mathcal{D}_1 \mathbin{|||} \mathcal{D}_2$, is given by the FDS $\mathcal{D} = \langle V, \Theta, \rho, \mathcal{J}, \mathcal{C} \rangle$, where

$$
\begin{array}{lllll}
V & = & V_1 & \cup & V_2 \\
\Theta & = & \Theta_1 & \wedge & \Theta_2 \\
\rho & = & \rho_1 & \wedge & \rho_2 \\
\mathcal{J} & = & \mathcal{J}_1 & \cup & \mathcal{J}_2 \\
\mathcal{C} & = & \mathcal{C}_1 & \cup & \mathcal{C}_2
\end{array}
$$

Synchronous parallel composition can be used for hardware verification, where it is the natural operator for combining two circuits into a composed circuit. Here we use it for model checking of LTL formulas.

Claim 6. *The sequence σ of V-states is a computation of the combined $\mathcal{D}_1 \mathbin{|||} \mathcal{D}_2$ iff $\sigma \Downarrow_{V_1}$ is a computation of $\mathcal{D}_1$ and $\sigma \Downarrow_{V_2}$ is a computation of $\mathcal{D}_2$.*

Here, $\sigma \Downarrow_{V_i}$ denotes the sequence obtained from σ by restricting each of the states to a V_i-state.

Temporal Testers

Let φ be a temporal formula over vocabulary U, and let $x \notin U$ be a boolean variable disjoint from U.

In the following, let $\sigma : s_0, s_1, \ldots$ be an infinite sequence of states over $U \cup \{x\}$. We say that x matches φ in σ if, for every position $j \geq 0$, the value of x at position j is true iff $(\sigma, j) \models \varphi$.

A temporal tester for φ is an FDS $T(\varphi)$ over $U \cup \{x\}$, satisfying the requirement:

The infinite sequence σ is a computation of $T(\varphi)$ iff x matches φ in σ.

A consequence of this definition is that every infinite sequence π of U-states can be extended into a computation σ of $T(\varphi)$ by interpreting x at position $j \geq 0$ of σ as 1 iff $(\pi, j) \models \varphi$.

Construction of Temporal Testers

A formula φ is called a principally temporal formula (PTF) if the main operator of p is temporal. A PTF is called a basic temporal formula if it contains no other PTF as a proper sub-formula.

We start our construction by presenting temporal testers for the basic temporal formulas.

A Tester for $\bigcirc p$

The tester for the formula $\bigcirc p$ is given by:

$$T(\bigcirc p) : \begin{cases} V : & Vars(p) \ \cup \ \{x\} \\ \Theta : & 1 \\ \rho : & x \quad = \quad p' \\ \mathcal{J} = \mathcal{C} : & \emptyset \end{cases}$$

Claim 7.
$T(\bigcirc p)$ is a temporal tester for $\bigcirc p$.

Proof:
Let σ be a computation of $T(\bigcirc p)$. We will show that x matches $\bigcirc p$ in σ. Let $j \geq 0$ be any position. By the transition relation, $x = 1$ at position j iff $s_{j+1} \models p$ iff $(\sigma, j) \models \bigcirc p$.

Let σ be an infinite sequence such that x matches $\bigcirc p$ in σ. We will show that σ is a computation of $T(\bigcirc p)$. For any position $j \geq 0$, $x = 1$ at j iff $(\sigma, j) \models \bigcirc p$, iff $s_{j+1} \models p$. Thus, x satisfies $x = p'$ at every position j. ▪

A Tester for $p\mathcal{U}q$

The tester for the formula $p\mathcal{U}q$ is given by:

$$T(p\mathcal{U}q): \begin{cases} V: & \mathit{Vars}(p,q) \ \cup \ \{x\} \\ \Theta: & 1 \\ \rho: & x \quad = \quad q \ \vee \ (p \ \wedge \ x') \\ \mathcal{J}: & q \ \vee \ \neg x \\ \mathcal{C}: & \emptyset \end{cases}$$

Claim 8.
$T(p\mathcal{U}q)$ is a temporal tester for $p\mathcal{U}q$.

Proof:
Let σ be a computation of $T(p\,\mathcal{U}\,q)$. We will show that x matches $p\,\mathcal{U}\,q$ in σ. Let $j \geq 0$ be any position. Consider first the case that $s_j \models x$ and we will show that $(\sigma, j) \models p\,\mathcal{U}\,q$. According to the transition relation, $s_j \models x$ implies that either $s_j \models q$ or $s_j \models p$ and $s_{j+1} \models x$. If $s_j \models q$ then $(\sigma, j) \models p\,\mathcal{U}\,q$ and we are done. Otherwise, we apply the same argument to position $j+1$. Continuing in this manner, we either locate a $k \geq j$ such that $s_k \models q$ and $s_i \models p$ for all i, $j \leq i < k$, or we have $s_i \models \neg q \ \wedge \ p \ \wedge \ x$ for all $i \geq j$. If we locate a stopping k then, obviously $(\sigma, j) \models p\,\mathcal{U}\,q$ according to the semantic definition of the $\mathcal{U}$ operator. The other case in which both $\neg q$ and x hold over all positions beyond j is impossible since it violates the justice requirement demanding that σ contains infinitely many positions at which either q is true or x is false.

Proof Continued

Next we consider the case that σ is a computation of $T(p\mathcal{U}q)$ and $(\sigma, j) \models p\mathcal{U}q$, and we have to show that $s_j \models x$. According to the semantic definition, there exists a $k \geq j$ such that $s_k \models q$ and $s_i \models p$ for all i, $j \leq i < k$. Proceeding from k backwards all the way to j, we can show (by induction if necessary) that the transition relation implies that $s_t \models x$ for all $t = k, k-1, \ldots, j$.

In the other direction, let σ be an infinite sequence such that x matches $p\mathcal{U}q$ in σ. We will show that σ is a computation of $T(p\mathcal{U}q)$. From the semantic definition of $\mathcal{U}$ it follows that $(\sigma, j) \models p\mathcal{U}q$ iff either $s_j \models q$ or $s_j \models p$ and $(\sigma, j+1) \models p\mathcal{U}q$. Thus, if $x = (p\mathcal{U}q)$ at all positions, the transition relation $x = q \lor (p \land x')$ holds at all positions. To show that x satisfies the justice requirement $q \lor \neg x$ it is enough to consider the case that σ contains only finitely many q-positions. In that case, there must exist a cutoff position $c \geq 0$ such that no position beyond c satisfies q. In this case, $p\mathcal{U}q$ must be false at all positions beyond c. Consequently, x is false at all positions beyond c and is therefore false at infinitely many positions. ▟

Why Do We Need the Justice Requirement

Reconsider the temporal tester for $p\mathcal{U}q$:

$$T(p\mathcal{U}q): \begin{cases} V: & \mathit{Vars}(p,q) \ \cup \ \{x\} \\ \Theta: & 1 \\ \rho: & x \quad = \quad q \ \vee \ (p \ \wedge \ x') \\ \mathcal{J}: & q \ \vee \ \neg x \\ \mathcal{C}: & \emptyset \end{cases}$$

We wish to show that the justice requirement $q \vee \neg x$ is essential for the correctness of the construction. Consider a state sequence $\sigma : s_0, s_1, \ldots$ in which q is identically false and p is identically true at all positions. In this case, the transition relation reduces to the equation

$$x = x'.$$

This equation has two possible solutions, one in which x is identically false and the other in which x is identically true at all positions. Only $x = 0$ matches $p\mathcal{U}q$. This is also the only solution which satisfies the justice requirement.

Thus, the role of the justice requirement is to select among several solutions to the transition relation equation, a unique one which matches the basic temporal formula at all positions.

A Tester for $p\mathcal{W}q$

A supporting evidence for the significance of the justice requirements is provided by the tester for the formula $p\mathcal{W}q$:

$$T(p\mathcal{W}q): \begin{cases} V: & \mathit{Vars}(p,q) \ \cup \ \{x\} \\ \Theta: & 1 \\ \rho: & x \quad = \quad q \ \vee \ (p \ \wedge \ x') \\ \mathcal{J}: & \neg p \ \vee \ x \\ \mathcal{C}: & \emptyset \end{cases}$$

Note that the transition relation of $T(p\mathcal{W}q)$ is identical to that of $T(p\mathcal{U}q)$, and they only differ in their respective justice requirements.

The role of the justice requirement in $T(p\mathcal{W}q)$ is to eliminate the solution $x = 0$ over a computation in which $p = 1$ and $q = 0$ at all positions.

Testers for the Derived Operators

Based on the testers for $\mathcal{U}$ and $\mathcal{W}$, we can construct testers for the derived operators $\diamondsuit$ and $\square$. They are given by

$$T(\diamondsuit\, p): \begin{cases} V: & \mathit{Vars}(p) \ \cup \ \{x\} \\ \Theta: & 1 \\ \rho: & x \quad = \quad p \ \vee \ x' \\ \mathcal{J}: & p \ \vee \ \neg x \\ \mathcal{C}: & \emptyset \end{cases} \qquad T(\square\, p): \begin{cases} V: & \mathit{Vars}(p) \ \cup \ \{x\} \\ \Theta: & 1 \\ \rho: & x \quad = \quad p \ \wedge \ x' \\ \mathcal{J}: & \neg p \ \vee \ x \\ \mathcal{C}: & \emptyset \end{cases}$$

A formula such as $\diamondsuit\, p$ can be viewed as a "promise for an eventual p". The justice requirement $p \vee \neg x$ can be interpreted as suggesting:

Either fulfill all your promises or stop promising.

Note that once $x = 0$ in the tester $T(\diamondsuit\, p)$, it remains 0 and requires $p = 0$ ever after.

Testers for the Past Basic Formulas

The following are testers for the basic past formulas $\ominus p$ and $p\mathcal{S}q$:

$$T(\ominus p): \begin{cases} V: & \mathit{Vars}(p) \ \cup \ \{x\} \\ \Theta: & x = 0 \\ \rho: & x' \quad = \quad p \\ \mathcal{J}: & \emptyset \\ \mathcal{C}: & \emptyset \end{cases} \qquad T(p\mathcal{S}q): \begin{cases} V: & \mathit{Vars}(p,q) \ \cup \ \{x\} \\ \Theta: & x = q \\ \rho: & x' \quad = \quad q' \ \vee \ (p' \wedge x) \\ \mathcal{J}: & \emptyset \\ \mathcal{C}: & \emptyset \end{cases}$$

Note that testers for past formulas are not associated with any fairness requirements. On the other hand, they have a non-trivial initial conditions.

Testers for Compound Temporal Formulas

Up to now we only considered testers for basic formulas. The construction for non-basic formulas is based on the following reduction principle. Let $f(\varphi)$ be a temporal formula containing one or more occurrences of the basic formula φ. Then the temporal tester for $f(\varphi)$ can be constructed according to the following recipe:

$$T(f(\varphi)) \quad = \quad T(f(x_\varphi)) \parallel\!\mid T(\varphi)$$

where, x_φ is the boolean output variable of $T(\varphi)$, and $f(x_\varphi)$ is obtained from $f(\varphi)$ by replacing every instance of φ by x_φ.

Following this recipe the temporal tester for an arbitrary formula f can be decomposed into a synchronous parallel composition of smaller testers, one for each basic formula nested within f.

Example: A Tester for $\Diamond\Box\, p$

Following is a tester for the formula $\Diamond\Box\, p$ which is obtained by computing the parallel composition $T(\Diamond\, x_{\Box}\,) \mathrel{|||} T(\Box\, p)$.

$$T(\Diamond\Box\, p): \begin{cases} V: & \mathit{Vars}(p)\ \cup\ \{x_{\Diamond}\,, x_{\Box}\,\} \\ \Theta: & 1 \\ \rho: & (x_{\Box}\, = p\ \wedge\ x_{\Box}\,')\ \wedge\ (x_{\Diamond}\, = x_{\Box}\ \vee\ x_{\Diamond}\,') \\ \mathcal{J}: & \{\neg p\ \vee\ x_{\Box}\,,\quad x_{\Box}\ \vee\ \neg x_{\Diamond}\,\} \\ \mathcal{C}: & \emptyset \end{cases}$$

The output variable of $\Diamond\Box\, p$ is $x_{\Diamond}$.

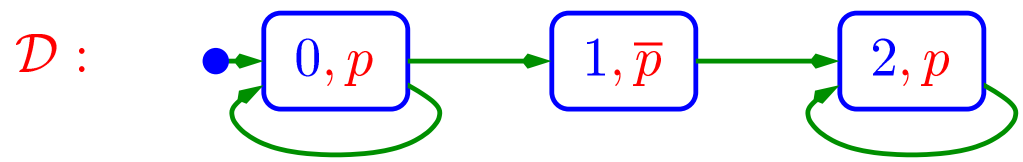
Model Checking General Temporal Formulas

To check whether $\mathcal{D} \models \varphi$, perform the following steps:

- Construct the temporal tester $T(\varphi)$.
- Form the combined system $C = \mathcal{D} \parallel\!\mid T(\varphi) \parallel\!\mid [\Theta : \neg x_\varphi]$, where $[\Theta : \neg x_\varphi]$ is a trivial FDS which imposes the initial condition $\neg x_\varphi$, implying that φ is false at the initial states.
- Check whether C is feasible.
- Conclude $\mathcal{D} \models \varphi$ iff C is infeasible.

Example

Consider the following system:

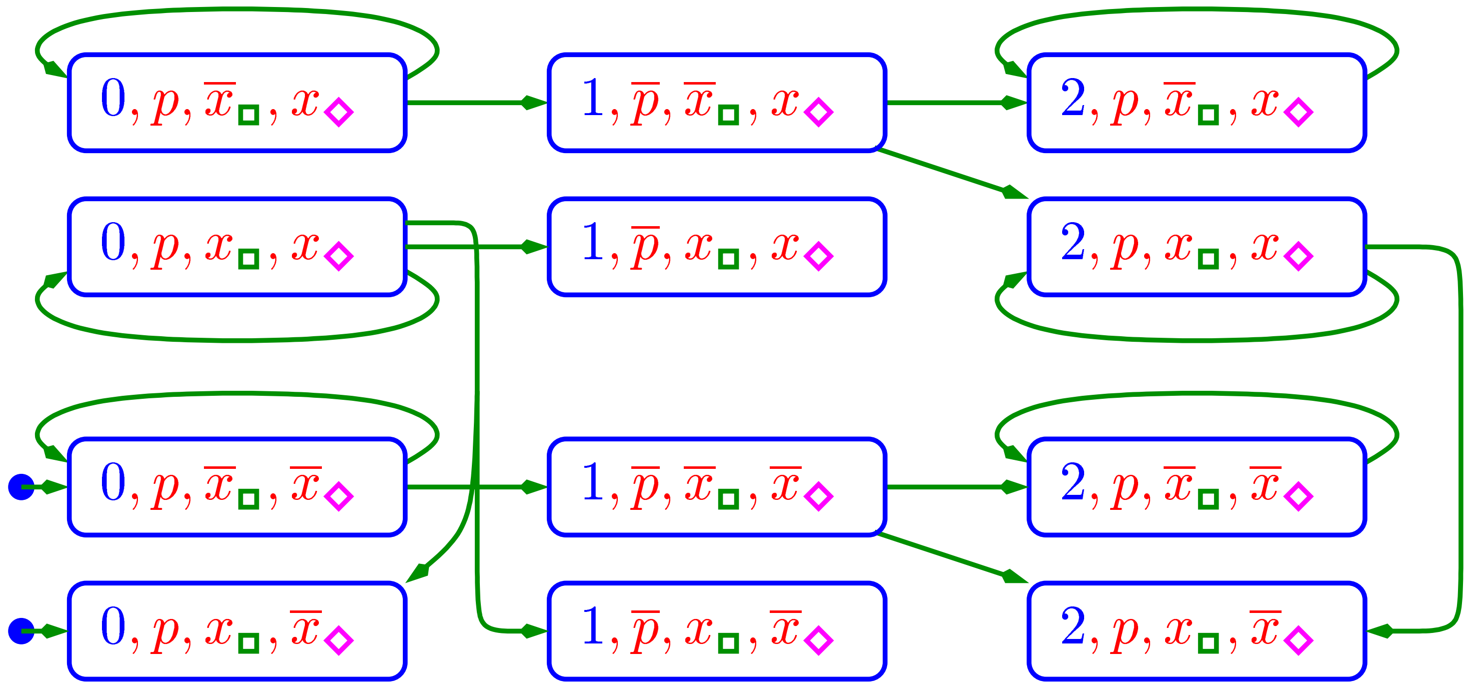


For which we wish to verify the property $\Diamond \Box\, p$.

Example: Continued

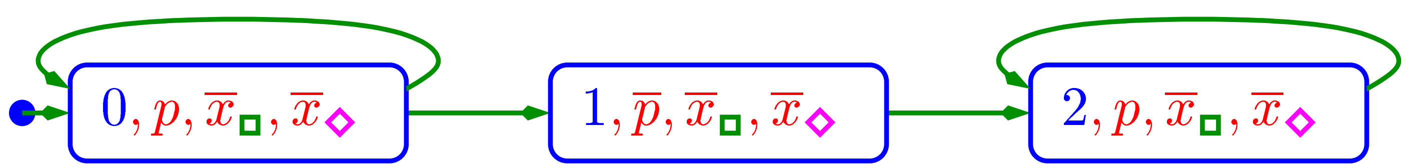
Composing the system with the temporal tester $T(\Diamond\Box\, p)$, we obtain:

C :



with the justice requirements $\neg p \vee x_\Box$ and $x_\Box \vee \neg x_\Diamond$.

Eliminating all unreachable states and states with no successors, we are left with:



State 2 is eliminated because it does not have a path leading to a $\neg p \vee x_\Box$ -state. Then state 1 is eliminated. having no successors. Finally, 0 is eliminated because it cannot reach a $\neg p \vee x_\Box$ -state. Nothing is left, hence the system satisfies the property $\Diamond\Box\, p$.

Correctness of the Algorithms

Claim 9.
For an FDS $\mathcal{D}$ *and temporal formula* φ, $\mathcal{D} \models \varphi$ *iff* $C : \mathcal{D} \parallel\!| \, T(\varphi) \parallel\!| \, [\Theta : \neg x_{\varphi}]$ *is infeasible*

Proof:
The proof is based on the observation that every computation of the combined system C is a computation of $\mathcal{D}$ which satisfies the negation of φ. Therefore, the existence of such a computation shows that not all computations of $\mathcal{D}$ satisfy φ, and therefore, φ is not valid over $\mathcal{D}$. ■