Observations and Equivalence of Systems

Let $U \subseteq V$ be a subset of state variables and s be a V-state. We denote by $s \Downarrow_U$ the U-state, called the projection of s on U, which is obtained by restricting the interpretation of variables to the variables in U.

For a V-state sequence

$$\sigma : \quad s_0, s_1, \ldots,$$

we denote by $\sigma \Downarrow_U$ the projected U-state sequence

$$\sigma \Downarrow_U : \quad s_0 \Downarrow_U, s_1 \Downarrow_U, \ldots.$$

An $\mathcal{O}$-state sequence Ω is called an observation of the FDS $\mathcal{D}$ if $\Omega = \sigma \Downarrow_{\mathcal{O}}$ for some σ, a computation of σ. We denote by $Obs(\mathcal{D})$ the set of observations of FDS $\mathcal{D}$.

Systems $\mathcal{D}_1$ and $\mathcal{D}_2$ are said to be equivalent, denoted $\mathcal{D}_1 \sim \mathcal{D}_2$, if their sets of abservations are identical. That is,

$$Obs(\mathcal{D}_1) = Obs(\mathcal{D}_2)$$

Feasibility and Viability of Systems

An FDS $\mathcal{D}$ is said to be feasible if $\mathcal{D}$ has at least one computation.

A finite or infinite sequence of states is defined to be a run of an FDS $\mathcal{D}$ if it satisfies the requirements of initiality and consecution but not necessarily any of the fairness requirements.

The FDS $\mathcal{D}$ is defined to be viable if any finite run of $\mathcal{D}$ can be extended to a computation of $\mathcal{D}$.

Claim 7. *Every* FDS *derived from an* SPL *program is viable.*

Note that if $\mathcal{D}$ is a viable system, such that its initial condition $\Theta_{\mathcal{D}}$ is satisfiable, then $\mathcal{D}$ is feasible.

Operations on FDS's: Asynchronous Parallel Composition

Systems $\mathcal{D}_1$ and $\mathcal{D}_2$ are compatible if $V_1 \cap V_2 = \mathcal{O}_1 \cap \mathcal{O}_2$.

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

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

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

Synchronous Parallel Composition

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

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

Synchronous parallel composition is useful for the modeling and verification of hardware designs. It is also useful for augmenting systems with auxiliary monitors.

Claim 9. *Let σ be an infinite $(V_1 \cup V_2)$-state sequence. Sequence σ is a computation of $\mathcal{D}_1 \;|||\; \mathcal{D}_2$ iff ($\sigma\Downarrow_{V_1}$ is a computation of $\mathcal{D}_1$ and $\sigma\Downarrow_{V_2}$ is a computation of $\mathcal{D}_2$).*

Requirement Specification Language: Temporal Logic

Assume an underlying (first-order) assertion language. The predicate at_ℓ_i, abbreviates the formula $\pi_j = \ell_i$, where ℓ_i is a location within process P_j.

A temporal formula is constructed out of state formulas (assertions) to which we apply the boolean operators $\neg$ and $\vee$ and various temporal operators, such as:

$\Box$ – always $\Diamond$ – eventually

A model for a temporal formula p is an infinite sequence of states $\sigma : s_0, s_1, \ldots$, where each state s_j provides an interpretation for the variables of p.

Semantics of LTL

Given a model σ, we define the notion of a temporal formula p holding at a position $j \geq 0$ in σ, denoted by $(\sigma, j) \models p$:

- For an assertion p,
  $(\sigma, j) \models p \iff s_j \models p$
  That is, we evaluate p locally on state s_j.
- $(\sigma, j) \models \neg p \iff (\sigma, j) \not\models p$
- $(\sigma, j) \models p \vee q \iff (\sigma, j) \models p$ or $(\sigma, j) \models q$
- $(\sigma, j) \models \Box\, p \iff (\sigma, k) \models p$ for all $k \geq j$
- $(\sigma, j) \models \Diamond\, p \iff (\sigma, k) \models p$ for some $k \geq j$

If $(\sigma, 0) \models p$ we say that p holds over σ and write $\sigma \models p$. p is satisfiable if it holds over some model. p is (temporally) valid if it holds over all models.

Formulas p and q are equivalent, denoted $p \sim q$, if $p \leftrightarrow q$ is valid. They are called congruent, denoted $p \approx q$, if $\Box\,(p \leftrightarrow q)$ is valid. If $p \approx q$ then p can be replaced by q in any context.

We write $p \Rightarrow q$ as an abbreviation for $\Box\,(p \rightarrow q)$.

Reading Exercises

Following are some temporal formulas φ and what they say about a sequence $\sigma : s_0, s_1, \ldots$ such that $\sigma \models \varphi$:

- $\Box\, p$ — All states within σ satisfy p. Previously, we denoted this property by $Inv(p)$.
- $p \rightarrow \Diamond\, q$ — If p holds at s_0, then q holds at s_j for some $j \geq 0$.
- $\Box\, (p \rightarrow \Diamond\, q)$ — Every p is followed by a q. Also written as $p \Rightarrow \Diamond\, q$. Previously, we denoted this property by $p \rightsquigarrow q$.
- $\Box\, \Diamond\, q$ — The sequence σ contains infinitely many q's.
- $\Diamond\, \Box\, q$ — All but finitely many states in σ satisfy q. Property q eventually stabilizes.

Temporal Specification of Properties

Formula φ is $\mathcal{D}$-valid, denoted $\mathcal{D} \models \varphi$, if all computations of $\mathcal{D}$ satisfy φ. Such a formula specifies a property of $\mathcal{D}$.

Following is a temporal specification of the main properties of program MUX-SEM.

- Mutual Exclusion – No computation of the program can include a state in which process P_1 is at ℓ_3 while P_2 is at m_3. Specifiable by the formula

$$\Box\, \neg(at\_\ell_3 \ \wedge \ at\_m_3)$$

- Accessibility for P_1 – Whenever process P_1 is at ℓ_2, it shall eventually reach it's critical section at ℓ_3. Specifiable by the formula

$$\Box\, (at\_\ell_2 \ \rightarrow \ \Diamond\, at\_\ell_3)$$

Full Temporal Logic – The Basic Operators

$\bigcirc$ – Next $\ominus$ – Previous
$\mathcal{U}$ – Until $\mathcal{S}$ – Since

Their semantics:

- $(\sigma, j) \models \bigcirc p \iff (\sigma, j+1) \models p$
- $(\sigma, j) \models p\ \mathcal{U}\ q \iff$ for some $k \geq j, (\sigma, k) \models q,$ and for every i such that $j \leq i < k, \quad (\sigma, i) \models p$
- $(\sigma, j) \models \ominus p \iff j > 0$ and $(\sigma, j-1) \models p$
- $(\sigma, j) \models p\ \mathcal{S}\ q \iff$ for some $k \leq j, (\sigma, k) \models q,$ and for every i such that $j \geq i > k, \quad (\sigma, i) \models p$

All other temporal operators can be defined in terms of these 4 as follows:

$\Diamond p = 1\ \mathcal{U}\ p$	– Eventually
$\Box p = \neg \Diamond \neg p$	– Henceforth
$p\ \mathcal{W}\ q = \Box p \vee (p\ \mathcal{U}\ q)$	– Waiting-for, Unless, Weak Until
$\Diamond\!\!-\ p = 1\ \mathcal{S}\ p$	– Sometimes in the past
$\Box\!\!-\ p = \neg \Diamond\!\!- \neg p$	– Always in the past
$p\ \mathcal{B}\ q = \Box\!\!- p \vee (p\ \mathcal{S}\ q)$	– Back-to, Weak Since
$\circledcirc p = \neg \ominus \neg p$	– Weak Previous

Expressive Completeness

Every (propositional) temporal formula φ can be translated into a first-order logic with monadic predicates over the naturals ordered by $<$ (1st-order theory of linear order).

For example, the 1st-order translation of $p \Rightarrow \Diamond q$ is

$$\forall t_1 \geq 0 : (p(t_1) \rightarrow \exists t_2 \geq t_1 : (q(t_2)))$$

Can every 1st-order formula be translated into temporal logic?

W. Kamp [Kamp68] has shown that the answer is negative if we only allow $\Box$ and $\Diamond$ in our temporal formulas. But then proceeded to show that:

Claim 10. *Every 1st-order formula can be translated into a temporal formula in the logic* $\mathcal{L}(\mathcal{U}_>, \mathcal{S}_>)$.

[GPSS81] has shown that

Claim 11. *Every 1st-order formula can be translated into a temporal formula in the logic* $\mathcal{L}(\bigcirc, \mathcal{U})$.

This also shows that the past operators add no expressive power.

Classification of Formulas/Properties

A formula of the form $\Box\, p$ for some past formula p is called a safety formula.

A formula of the form $\Box\Diamond\, p$ for some past formula p is called a response formula.

An equivalent characterization is the form $p \Rightarrow \Diamond\, q$. The equivalence is justified by

$$\Box\,(p \rightarrow \Diamond\, q) \qquad \sim \qquad \Box\Diamond\,((\neg p)\ \mathcal{B}\ q)$$

Both formulas state that either there are infinitely many q's, or there there are no p's, or there is a last q-position, beyond which there are no further p's.

A property is classified as a safety/response property if it can be specified by a safety/response formula.

Every temporal formula is equivalent to a conjunction of a reactivity formulas, i.e.

$$\bigwedge_{i=1}^{k} (\Box\Diamond\, p_i \ \vee\ \Diamond\Box\, q_i)$$