CS256/Spring 2008 — Lecture #4

Zohar Manna

Example: Proving a Congruence

For temporal formulas φ and ψ, show

$$\Diamond\Box\varphi \wedge \Diamond\Box\psi \;\approx\; \Diamond(\Box\varphi \wedge \Box\psi)$$

We have to show

$$\Diamond\Box\varphi \wedge \Diamond\Box q \Rightarrow \Diamond(\Box\varphi \wedge \Box\psi)$$

and

$$\Diamond\Box\varphi \wedge \Diamond\Box\psi \Leftarrow \Diamond(\Box\varphi \wedge \Box\psi)$$

$\Rightarrow$ The left-to-right entailment is valid:
Consider arbitrary σ and j such that

$$(\sigma, j) \models \Diamond\Box\varphi \wedge \Diamond\Box\psi.$$

Thus

$$\exists k_1 \geq j.\ (\sigma, k_1) \models \Box\varphi$$

<u>and</u>

$$\exists k_2 \geq j.\ (\sigma, k_2) \models \Box\psi$$

Example: Proving a Congruence (Cont'd)

$$\Diamond \Box \varphi \wedge \Diamond \Box \psi \;\approx\; \Diamond(\Box \varphi \wedge \Box \psi)$$

Unraveling the definition of $\Box$, we get

$$\exists k_1 \geq j.\ \forall k_1' \geq k_1.\ (\sigma, k_1') \vDash \varphi$$

<u>and</u>

$$\exists k_2 \geq j.\ \forall k_2' \geq k_2.\ (\sigma, k_2') \vDash \psi.$$

This implies that

$$\overbrace{\exists k \geq j.}^{k=max\{k_1,k_2\}} \quad \forall k' \geq k.\ (\sigma, k') \vDash \varphi \ \underline{\text{and}}\ (\sigma, k') \vDash \psi.$$

So

$$\exists k \geq j.\ (\sigma, k) \vDash (\Box \varphi \wedge \Box \psi).$$

That is,

$$(\sigma, j) \vDash \Diamond(\Box \varphi \wedge \Box \psi).$$

$\Leftarrow$ The right-to-left entailment is valid. All implications in the first part hold in reverse, so the entailment is valid.

Example: Proving an Equivalence / Disproving a Congruence

For temporal logic formulas φ and ψ, show

$$\Diamond\, \varphi \;\sim\; \Diamond\, \diamondminus\, \varphi \qquad\qquad \Diamond\, \varphi \;\not\approx\; \Diamond\, \diamondminus\, \varphi$$

We shall prove: (1) $\Diamond\, \varphi \Rightarrow \Diamond\, \diamondminus\, \varphi$ is valid;

Thus $\Diamond\, \varphi \rightarrow \Diamond\, \diamondminus\, \varphi$ is valid.

(2) $\Diamond\, \diamondminus\, \varphi \rightarrow \Diamond\, \varphi$ is valid.

(3) $\Diamond\, \diamondminus\, \varphi \Rightarrow \Diamond\, \varphi$ is <u>not</u> valid.

(1) $\Diamond\varphi \Rightarrow \Diamond \overleftarrow{\Diamond}\varphi$ is valid:

Consider arbitrary σ and j such that

$$(\sigma, j) \models \Diamond\varphi.$$

Then $\quad \exists i \geq j.\ (\sigma, i) \models \varphi.$

Hence $\quad \exists i \geq j.\ \underbrace{\exists k\colon 0 \leq k \leq i.}_{k=i}\ (\sigma, k) \models \varphi.$

By def. $\quad \exists i \geq j.\ (\sigma, i) \models \overleftarrow{\Diamond}\varphi.$

Therefore $\quad (\sigma, j) \models \Diamond \overleftarrow{\Diamond}\varphi.$

(2) $\Diamond \diamondminus \varphi \rightarrow \Diamond \varphi$ is valid:

Consider arbitrary σ such that

$$(\sigma, 0) \vDash \Diamond \diamondminus \varphi.$$

Then $\quad \exists i \geq 0.\ (\sigma, i) \vDash \diamondminus \varphi.$

Hence $\quad \exists i \geq 0.\ \exists k\colon 0 \leq k \leq i.\ \ (\sigma, k) \vDash \varphi.$

Hence $(k = i)$ $\quad \exists k \geq 0.\ (\sigma, k) \vDash \varphi.$

Therefore $\quad (\sigma, 0) \vDash \Diamond \varphi.$

(3) $\Diamond \diamondminus \varphi \Rightarrow \Diamond \varphi$ is **not** valid. Counterexample:

Take $\varphi\colon p$ (propositional symbol) $\sigma = \langle\ s_0\colon p,\ \ s_1\colon \neg p,\ \ s_2\colon \neg p,\ \ s_3\colon \neg p,\ \ \ldots\ \rangle$ and $j = 1$

Then $\quad (\sigma, 1) \vDash \Diamond \diamondminus p,$

but $\quad (\sigma, 1) \nvDash \Diamond p.$

Rigid and Flexible Variables

Variables in the vocabulary are partitioned into:

Rigid Variables:
Rigid variable has the same value in all states of a sequence σ

Flexible Variables:
The values of a flexible variable may be different in different states of a sequence σ.

- system variables are generally flexible (except for variables declared as *in* in an SPL program)
- auxiliary variables (used in specification) are usually rigid

`Example:`

"every value placed in x is eventually copied to z"

$$\forall u.\ (x = u \ \Rightarrow\ \Diamond(z = u))$$

u is a rigid auxiliary variable

Temporal Logic: Quantification

Definition:

Model $\sigma' :\ s'_0,\ s'_1,\ s'_2,\ \ldots$ is a u-variant of

$$\sigma :\ s_0,\ s_1,\ s_2,\ \ldots$$

if for every $j \geq 0$

s'_j agrees with s_j on the interpretation of all variables $y \in V - \{u\}$

Example:

$$\sigma' \colon \langle x\colon 0, y\colon 1, \boxed{z\colon 0}\rangle,\ \langle x\colon 1, y\colon 2, \boxed{z\colon 1}\rangle,\ \langle x\colon 2, y\colon 3, \boxed{z\colon 4}\rangle,\ \ldots$$

is a z-variant of

$$\sigma \colon \langle x\colon 0, y\colon 1, \boxed{z\colon 0}\rangle,\ \langle x\colon 1, y\colon 2, \boxed{z\colon 0}\rangle,\ \langle x\colon 2, y\colon 3, \boxed{z\colon 0}\rangle,\ \ldots$$

Semantics of Quantification

For temporal formula φ:

- $(\sigma, j) \models \exists u.\varphi \quad \Longleftrightarrow$
  $(\sigma', j) \models \varphi$ for some σ', a u-variant of σ

- $(\sigma, j) \models \forall u.\varphi \quad \Longleftrightarrow$
  $(\sigma', j) \models \varphi$ for all σ', a u-variant of σ

Examples

Let x, y be flexible variables

$$\sigma \vDash \exists y.\, \Box(y = x^2)$$

for σ: $\langle x\colon 1,\ y\colon 2\rangle,\ \langle x\colon 2,\ y\colon 3\rangle,\ \langle x\colon 3,\ y\colon 4\rangle,\ \ldots$

Take a y-variant

$$\sigma' :\ \langle x : 1,\ \boxed{y : 1}\rangle,\ \langle x : 2,\ \boxed{y : 4}\rangle,\ \langle x : 3,\ \boxed{y : 9}\rangle,\ \ldots$$

We have $(\sigma', 0) \vDash \Box(y = x^2)$

Therefore, $(\sigma, 0) \vDash \exists y.\, \Box(y = x^2)$

Examples

Let x be a flexible variable
u be a rigid variable

$$\boxed{\sigma \not\models \exists u.\ \Box(u = x^2)}$$

Consider σ: $\langle x\colon 1,\ u\colon 0\rangle,\ \langle x\colon 2,\ u\colon 0\rangle,\ \langle x\colon 3,\ u\colon 0\rangle,\ \ldots$

Since u is rigid, every u-variant must be of the form

$$\sigma'\colon\ \langle x\colon 1,\ \boxed{u\colon a}\rangle,\ \langle x\colon 2,\ \boxed{u\colon a}\rangle,\ \langle x\colon 3,\ \boxed{u\colon a}\rangle,\ \ldots$$

(with u having the same value in all states)

There is no u-variant σ' such that

$$(\sigma', 0) \models \Box(u = x^2)$$

Therefore, $(\sigma, 0) \not\models \exists u.\ \Box(u = x^2)$

Examples

Let u be a rigid variable.

$$\diamondsuit(\forall u.\ p) \not\approx \forall u.\ \diamondsuit p$$

i.e., $\diamondsuit(\forall u.\ p) \leftrightarrow \forall u.\ \diamondsuit p$ is not valid

Take $p : x \neq u$ with x flexible

$$\sigma\colon\ \langle x\colon 0,\ u\colon 2\rangle,\ \langle x\colon 1,\ u\colon 2\rangle,\ \langle x\colon 2,\ u\colon 2\rangle,\ \ldots$$

- left side: $\diamondsuit(\forall u.(x \neq u))$

  There is no position j such that
  $(\sigma, j) \models \forall u\ .\ x \neq u$ (take $u = x$)

  Therefore $(\sigma, 0) \not\models \diamondsuit(\forall u.(x \neq u))$

  i.e., $\boxed{\sigma \not\models \diamondsuit(\forall u.(x \neq u))}$

- right side: $\forall u.\ \diamond(x \neq u)$

Take an arbitrary u-variant of σ:

$$\sigma'_a\colon\ \langle x\colon 0,\ \boxed{u\colon a}\rangle,\ \langle x\colon 1,\ \boxed{u\colon a}\rangle,\ \langle x\colon 2,\ \boxed{u\colon a}\rangle, \ldots$$

and consider two cases

case $a = 0$	case $a \neq 0$
$\langle \sigma'_a, 1\rangle \vDash x \neq u$	$\langle \sigma'_a, 0\rangle \vDash x \neq u$
$\Downarrow$	$\Downarrow$
$\langle \sigma'_a, 0\rangle \vDash \diamond(x \neq u)$	$\langle \sigma'_a, 0\rangle \vDash \diamond(x \neq u)$

$$\langle \sigma, 0\rangle \vDash \forall u.\ \diamond(x \neq u)$$

i.e., $\boxed{\sigma \vDash \forall u.\ \diamond(x \neq u)}$

Therefore,

$$\diamond(\forall u.p) \leftrightarrow \forall u.\ \diamond(x \neq u)$$

is not valid.

Conjunction and Disjunction Characters

- For first-order logic:
  $\wedge$ has conjunction character

  $\vee$ has disjunction character

- For Temporal Logic:

$\Box$ and $\boxminus$	have conjunction character
$\Diamond$ and $\diamondminus$	have disjunction character
$\mathcal{U}, \mathcal{W}, \mathcal{S}, \mathcal{B}$	have conjunction character w.r.t. the first argument and disjunction character w.r.t. the second argument

- For Quantifiers:
  $\forall$ has conjunction character

  $\exists$ has disjunction character

Congruences

$$\forall u.(\varphi \land \psi) \quad \approx \quad \forall u.\varphi \;\land\; \forall u.\psi$$

$$\exists u.(\varphi \lor \psi) \quad \approx \quad \exists u.\varphi \;\lor\; \exists u.\psi$$

$$\Box(\forall u.\varphi) \quad \approx \quad \forall u.\,\Box\,\varphi$$

$$\Diamond(\exists u.\varphi) \quad \approx \quad \exists u.\,\Diamond\,\varphi$$

$$\varphi\,\mathcal{U}\,(\exists u.\psi) \quad \approx \quad \exists u.(\varphi\,\mathcal{U}\,\psi) \qquad (u \text{ not free in } \varphi)$$

$$(\forall u.\varphi)\,\mathcal{U}\,\psi \quad \approx \quad \forall u.(\varphi\,\mathcal{U}\,\psi) \qquad (u \text{ not free in } \psi)$$

$$\varphi\,\mathcal{W}\,(\exists u.\psi) \quad \approx \quad \exists u.(\varphi\,\mathcal{W}\,\psi) \qquad (u \text{ not free in } \varphi)$$

$$(\forall u.\varphi)\,\mathcal{W}\,\psi \quad \approx \quad \forall u.(\varphi\,\mathcal{W}\,\psi) \qquad (u \text{ not free in } \psi)$$

Expressibility

There are properties that cannot be specified by a quantifier-free temporal logic formula.

Example:

Specify the property
"x assumes the value 0 only, if ever, at even positions"
i.e., "at positions $0, 2, 4, \ldots$"

- cannot be expressed in quantifier-free TL

- can be expressed in (quantified) TL

  Quantifying over flexible boolean variable b:
  $\exists b[b \;\wedge\; \Box(b \leftrightarrow \neg \bigcirc b) \;\wedge\; \Box(x = 0 \rightarrow b)]$.
  $\forall b[b \;\wedge\; \Box(b \leftrightarrow \neg \bigcirc b) \;\rightarrow \Box(x = 0 \rightarrow b)]$.

  Why not

$$x = 0 \wedge \Box[x = 0 \rightarrow \bigcirc \bigcirc (x = 0)]?$$

Temporal vs First-Order

TL formula

$$\Box(p \;\rightarrow\; \Diamond[r \;\wedge\; \Diamond q])$$

can be transformed into FOL formula

$$(\forall t_1 \geq 0)\left[\; p(t_1) \rightarrow (\exists t_2)\left[\begin{array}{c} t_1 \leq t_2 \;\wedge\; r(t_2) \;\wedge \\ (\exists t_3)(t_2 \leq t_3 \;\wedge\; q(t_3)) \end{array}\right]\right]$$

where t_1, t_2, t_3 are integers.

Temporal Logic + Programs

P-Validity

Given a program P:

- For a state formula q:

  $\Vdash q$ (q is state valid)
  if q holds in all states

Example:
$\Vdash x = 1 \rightarrow x > 0$

$P \Vdash q$ (q is state valid over P
q is P-state valid)
if q holds over all P-accessible states

Recall : State s_i is a P-accessible state if it is in some computation $\sigma : s_0, s_1, s_2, \ldots, s_i, \ldots$ of P.

Example

local x: **integer where** $x = 1$

$$\left[\begin{array}{l} \ell_0\text{: } \textbf{loop forever do} \\ \quad \left[\begin{array}{l} \ell_1\text{: } \textbf{await } x = 1 \\ \ell_2\text{: } x := 2 \end{array}\right] \end{array}\right] \; || \; \left[\begin{array}{l} m_0\text{: } \textbf{loop forever do} \\ \quad \left[\begin{array}{l} m_1\text{: } \textbf{await } x = 2 \\ m_2\text{: } x := 1 \end{array}\right] \end{array}\right]$$

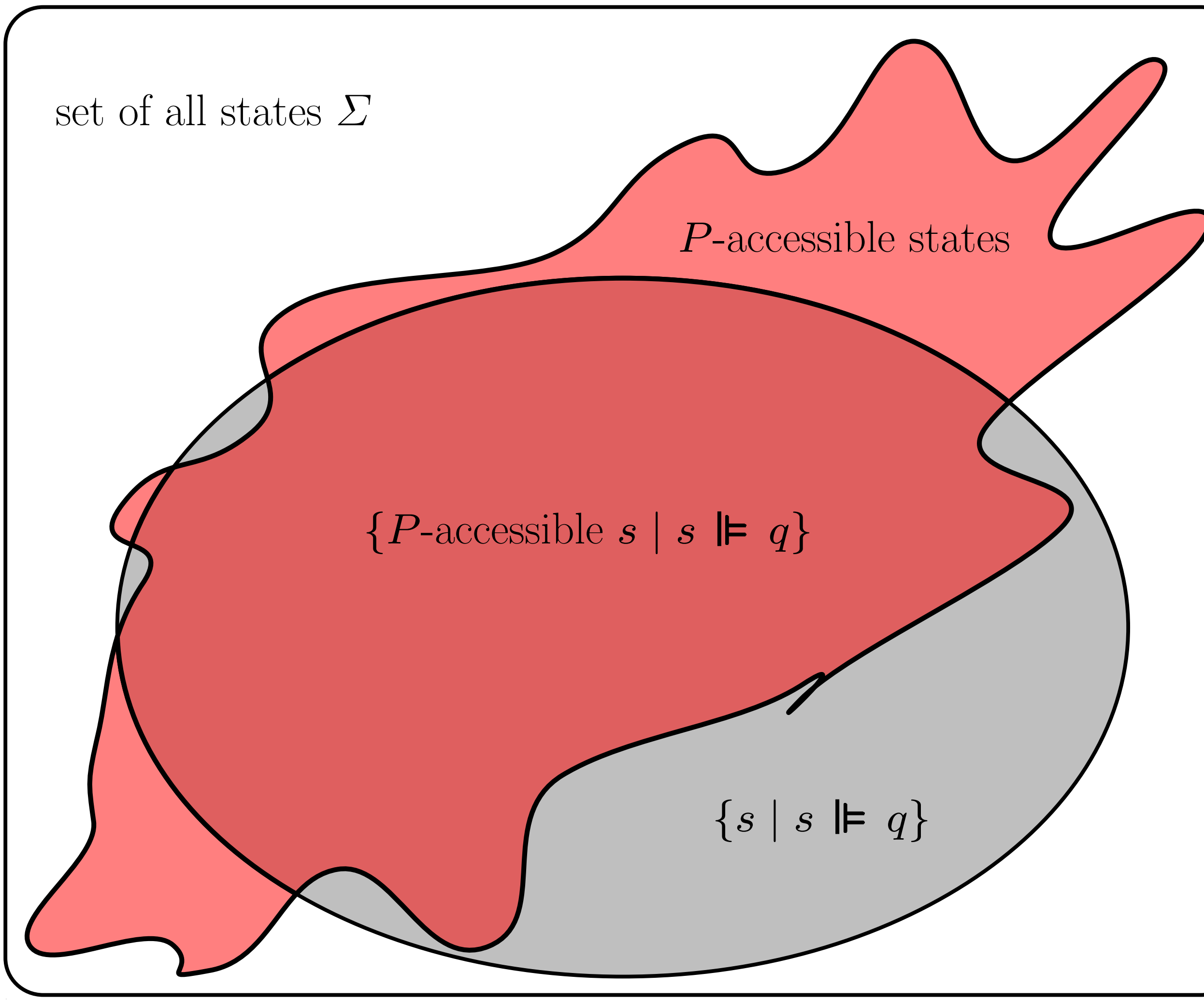
$P \; \Vdash \; x = 1 \lor x = 2$

$P \; \Vdash \; at\_\ell_2 \; \rightarrow \; x = 1$

Recall: at_ℓ_2 stands for $[\ell_2] \in \pi$

P-Validity (Con't)

$P \Vdash q$



P-Validity (Con't)

Given a program P:

- For a temporal formula φ:

  $\models \varphi$ (φ is valid)
  if φ holds in the first state of every model (i.e., every infinite sequence of states)

Example:

$\models \Box p \lor \Diamond \neg p$

P-Validity (Con't)

$P \models \varphi$ (φ is valid over P, φ is P-valid)
if φ holds in the first state of
every P-computations

Example:

$$\textbf{local } x\text{: } \textbf{integer where } x = 1$$

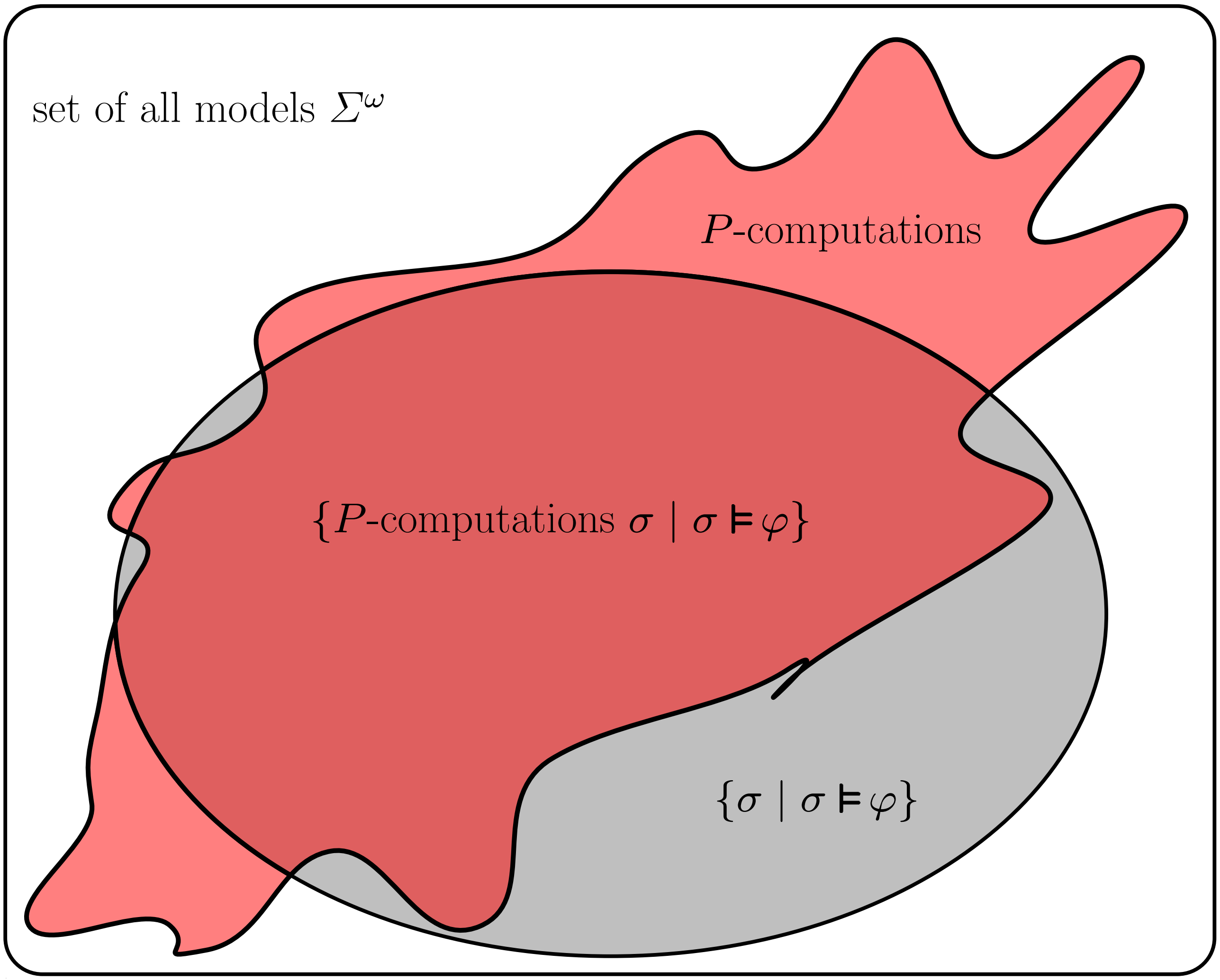
$$\left[\begin{array}{l} \ell_0\text{: } \textbf{loop forever do} \\ \quad \left[\begin{array}{l} \ell_1\text{: } \textbf{await } x = 1 \\ \ell_2\text{: } x := 2 \end{array}\right] \end{array}\right] \| \left[\begin{array}{l} m_0\text{: } \textbf{loop forever do} \\ \quad \left[\begin{array}{l} m_1\text{: } \textbf{await } x = 2 \\ m_2\text{: } x := 1 \end{array}\right] \end{array}\right]$$

$P \models \square \diamondsuit (x = 1) \wedge \square \diamondsuit (x = 2)$

$P \models at\_\ell_1 \Rightarrow \diamondsuit\, at\_\ell_2$

P-Validity (Con't)

$P \models \varphi$



P-Validity (Con't)

	general	program P
state formula q	$\Vdash q$ **state valid** "q holds in all states" $x = 1 \rightarrow x > 0$	$P \Vdash q$ **P-state valid** "q holds in all P-accessible states" $x = 1 \lor x = 2$
temporal formula φ	$\vDash \varphi$ **valid** "φ holds in first position of every sequence" $\Box p \lor \Diamond \neg p$	$P \vDash \varphi$ **P-valid** "φ holds in first position of every P-computation" $at\_\ell_1 \Rightarrow \Diamond at\_\ell_2$

Similarly,
P-satisfiability, P-equivalence,
P-congruence

P-Validity (Con't)

For state formula q:

$$
\begin{array}{rcl}
\Vdash q & \longleftrightarrow & \models \Box q \\
P \Vdash q & \longleftrightarrow & P \models \Box q \\
\Vdash q & \longrightarrow & P \Vdash q \quad \text{but not vice-versa}
\end{array}
$$

For temporal formula φ:

$$
\models \varphi \quad \longrightarrow \quad P \models \varphi \quad \text{but not vice-versa}
$$

Specification of Properties

- property $\mathit{\Pi}$ of P = set of models

- $\mathit{\Pi}$ is specified by temporal formula p
  if for every model σ, $\sigma \in \mathit{\Pi}$ iff $\sigma \models p$

- P has property $\mathit{\Pi}$ if

$$\{P\text{-computations}\} \subseteq \{\mathit{\Pi}\text{-models}\}$$

Classification of TL formulas

Reason for classification:

each class is associated with a proof principle for verifying that a given program satisfies a property specifiable by a formula in the class.

<u>Broad classification: Safety – Progress</u>

Safety

- all finite prefixes of a computation satisfy a certain requirement.

- "no bad things will happen"

- violation can be detected in finite time

- satisfaction of a safety property does not depend on the fairness conditions:
  a safety formula φ holds on all P-computations iff φ holds on all P-runs,
  i.e., a safety property cannot distinguish P-computations and P-runs.

- topic of textbook

Progress (liveness)

- "something good will happen eventually"

- always depends on fairness conditions in non-trivial cases, because the set of P-runs includes the sequence

$$s_0 \xrightarrow{\tau_I} s_1 \xrightarrow{\tau_I} s_2 \xrightarrow{\tau_I} \ldots$$

  i.e., the idling transition is the only transition ever taken