

CS256/Winter 2009 Lecture #4

Zohar Manna

Example: Proving a Congruence

For temporal formulas φ and ψ , show

$$\diamond \square \varphi \wedge \diamond \square \psi \approx \diamond (\square \varphi \wedge \square \psi)$$

We have to show

$$\diamond \square \varphi \wedge \diamond \square \psi \Rightarrow \diamond (\square \varphi \wedge \square \psi)$$

and

$$\diamond \square \varphi \wedge \diamond \square \psi \Leftarrow \diamond (\square \varphi \wedge \square \psi)$$

\Rightarrow The left-to-right entailment is valid:

Consider arbitrary σ and j such that

$$(\sigma, j) \models \diamond \square \varphi \wedge \diamond \square \psi.$$

Thus

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

and

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

Example: Proving a Congruence (Cont'd)

$$\diamond \square \varphi \wedge \diamond \square \psi \approx \diamond (\square \varphi \wedge \square \psi)$$

Unraveling the definition of \square , we get

$$\exists k_1 \geq j. \forall k'_1 \geq k_1. (\sigma, k'_1) \models \varphi$$

and

$$\exists k_2 \geq j. \forall k'_2 \geq k_2. (\sigma, k'_2) \models \psi.$$

This implies that

$$\begin{aligned} & k = \max\{k_1, k_2\} \\ & \overbrace{\exists k \geq j.} \quad \forall k' \geq k. \\ & \quad (\sigma, k') \models \varphi \text{ and } (\sigma, k') \models \psi. \end{aligned}$$

So

$$\exists k \geq j. (\sigma, k) \models (\square \varphi \wedge \square \psi).$$

That is,

$$(\sigma, j) \models \diamond (\square \varphi \wedge \square \psi).$$

\Leftarrow The right-to-left entailment is valid.

4-3

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

$$\diamond \varphi \sim \diamond \ominus \varphi \quad \diamond \varphi \not\sim \diamond \ominus \varphi$$

We shall prove: (1) $\diamond \varphi \Rightarrow \diamond \ominus \varphi$ is valid;

Thus $\diamond \varphi \rightarrow \diamond \ominus \varphi$ is valid.

(2) $\diamond \ominus \varphi \rightarrow \diamond \varphi$ is valid.

(3) $\diamond \ominus \varphi \Rightarrow \diamond \varphi$ is not valid.

(1) $\diamond \varphi \Rightarrow \diamond \diamond \varphi$ is valid:

Consider arbitrary σ and j such that

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

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

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

By def. $\exists i \geq j. (\sigma, i) \models \diamond \varphi.$

Therefore $(\sigma, j) \models \diamond \diamond \varphi.$

(2) $\diamond \Box \varphi \rightarrow \diamond \varphi$ is valid:

Consider arbitrary σ such that

$$(\sigma, 0) \models \diamond \Box \varphi.$$

Then $\exists i \geq 0. (\sigma, i) \models \Box \varphi.$

Hence $\exists i \geq 0. \exists k: 0 \leq k \leq i. (\sigma, k) \models \varphi.$

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

Therefore $(\sigma, 0) \models \diamond \varphi.$

(3) $\diamond \Box \varphi \Rightarrow \diamond \varphi$ is **not** valid. Counterexample:

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

Then $(\sigma, 1) \models \diamond \Box p,$

but $(\sigma, 1) \not\models \diamond p.$

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

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

if for every $j \geq 0$

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

Example:

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

is a z -variant of

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

Semantics of Quantification

For temporal formula φ :

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

Examples

Let x, y be flexible variables

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

for $\sigma: \langle x: 1, y: 2 \rangle, \langle x: 2, y: 3 \rangle, \langle x: 3, y: 4 \rangle, \dots$

Take a y -variant

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

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

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

Examples

Let x be a flexible variable
 u be a rigid variable

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

Consider $\sigma: \langle x: 1, u: 0 \rangle, \langle x: 2, u: 0 \rangle, \langle x: 3, u: 0 \rangle, \dots$

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

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

(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.

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

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

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

$$\sigma : \langle x: 0, u: 2 \rangle, \langle x: 1, u: 2 \rangle, \langle x: 2, u: 2 \rangle, \dots$$

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

There is no position j such that

$$(\sigma, j) \models \forall u. x \neq u \quad (\text{take } u = x)$$

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

i.e., $\sigma \not\models \diamond(\forall u.(x \neq u))$

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

Take an arbitrary u -variant of σ :

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

and consider two cases

case $a = 0$	case $a \neq 0$
$\langle \sigma'_a, 1 \rangle \models x \neq u$	$\langle \sigma'_a, 0 \rangle \models x \neq u$
\Downarrow	\Downarrow
$\langle \sigma'_a, 0 \rangle \models \Diamond(x \neq u)$	$\langle \sigma'_a, 0 \rangle \models \Diamond(x \neq u)$
$\underbrace{\hspace{15em}}$	
$\langle \sigma, 0 \rangle \models \forall u. \Diamond(x \neq u)$	
i.e., $\boxed{\sigma \models \forall u. \Diamond(x \neq u)}$	

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:
 - \square and \boxminus have conjunction character
 - \diamond and \lozenge 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

$$\forall u.(\varphi \wedge \psi) \approx \forall u.\varphi \wedge \forall u.\psi$$

$$\exists u.(\varphi \vee \psi) \approx \exists u.\varphi \vee \exists u.\psi$$

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

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

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

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

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

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

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, ...”

- 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

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

where t_1, t_2, t_3 are integers.

Temporal Logic + Programs

P-Validity

Given a program P :

- For a state formula q :

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

Example:

$\models x = 1 \rightarrow x > 0$

$P \models q$ (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 $\sigma : s_0, s_1, s_2, \dots, s_i, \dots$ of P .

Example

local x : integer where $x = 1$

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

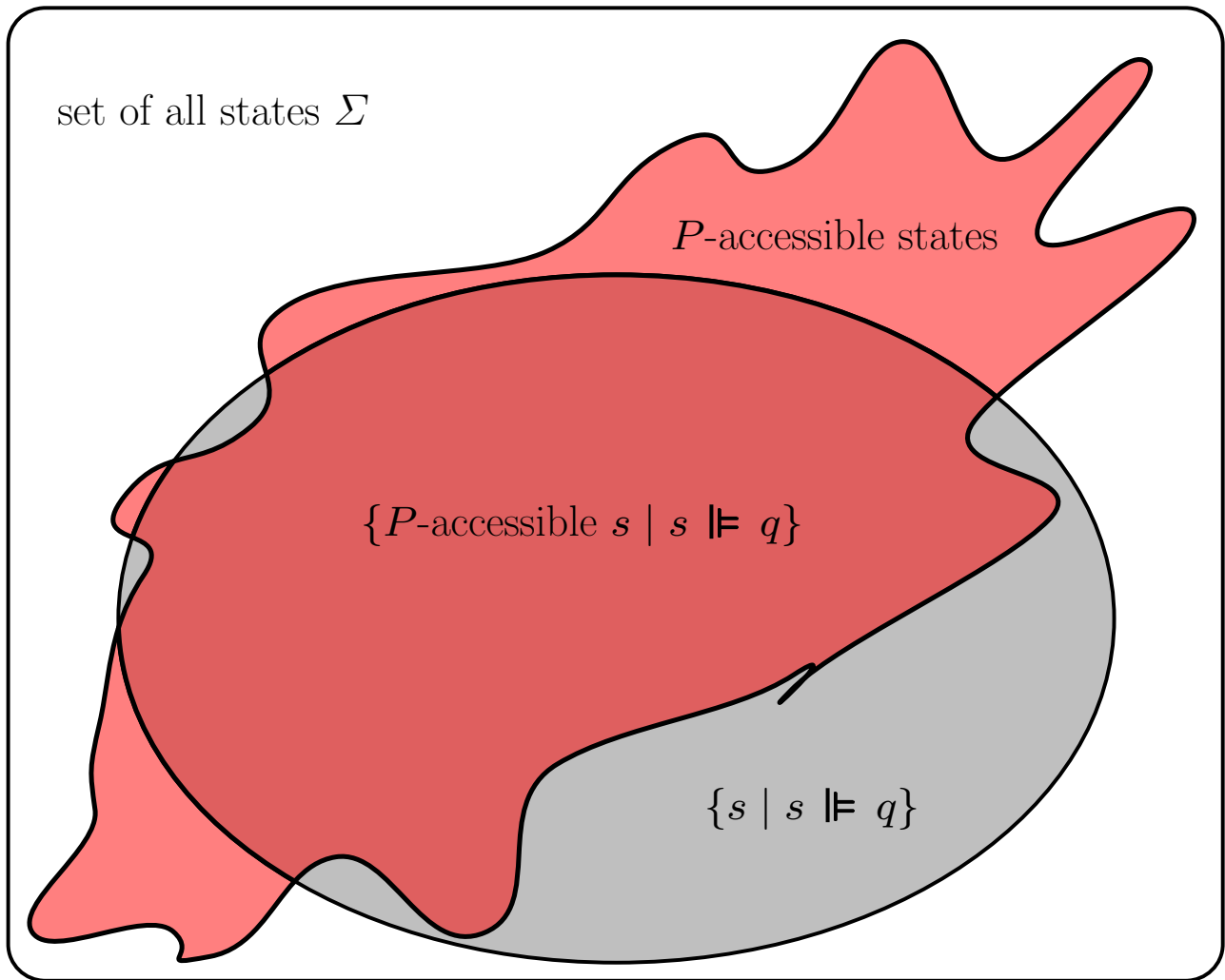
$P \models x = 1 \vee x = 2$

$P \models at_{\ell_2} \rightarrow x = 1$

Recall: at_{ℓ_2} stands for $[\ell_2] \in \pi$

P -Validity (Con't)

$$P \models q$$



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:

$\models \square p \vee \diamond \neg p$

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

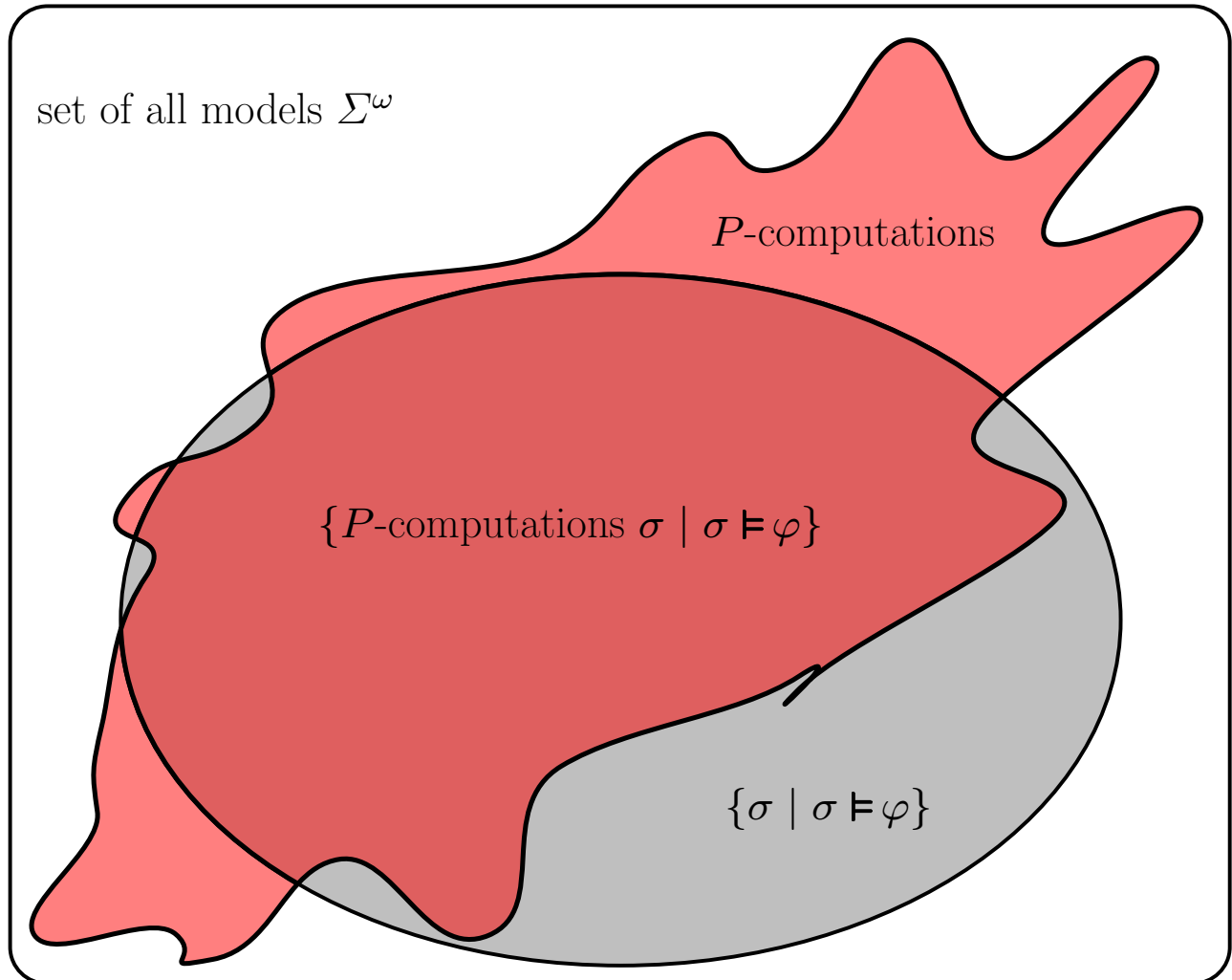
Example:

local x : integer where $x = 1$

$$\left[\begin{array}{l} \ell_0: \text{loop forever do} \\ \left[\begin{array}{l} \ell_1: \text{await } x = 1 \\ \ell_2: x := 2 \end{array} \right] \end{array} \right] \parallel \left[\begin{array}{l} m_0: \text{loop forever do} \\ \left[\begin{array}{l} m_1: \text{await } x = 2 \\ m_2: x := 1 \end{array} \right] \end{array} \right]$$
$$P \models \square \diamond (x = 1) \wedge \square \diamond (x = 2)$$
$$P \models at_{\ell_1} \Rightarrow \diamond at_{\ell_2}$$

P -Validity (Con't)

$$P \models \varphi$$



P-Validity (Con't)

	general	program <i>P</i>
state formula <i>q</i>	$\models q$ state valid “ <i>q</i> holds in all states” $x = 1 \rightarrow x > 0$	$P \models q$ <i>P</i>-state valid “ <i>q</i> holds in all <i>P</i> -accessible states” $x = 1 \vee x = 2$
temporal formula φ	$\vDash \varphi$ valid “ φ holds in first position of every sequence” $\Box p \vee \Diamond \neg p$	$P \vDash \varphi$ <i>P</i>-valid “ φ holds in first position of every <i>P</i> -computation” $at_l_1 \Rightarrow \Diamond at_l_2$

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

P-Validity (Con't)

For state formula q :

$$\begin{aligned} \models q &\iff \vDash \Box q \\ P \models q &\iff P \vDash \Box q \\ \models q &\implies P \models q \quad \text{but not vice-versa} \end{aligned}$$

For temporal formula φ :

$$\vDash \varphi \implies P \vDash \varphi \quad \text{but not vice-versa}$$

Specification of Properties

- property Π of P = set of models
- Π is specified by temporal formula p
if for every model σ , $\sigma \in \Pi$ iff $\sigma \models p$
- P has property Π if
$$\{P\text{-computations}\} \subseteq \{\Pi\text{-models}\}$$

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.

Broad classification: Safety – Progress

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

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

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