CS256/Winter 2009 Lecture #5

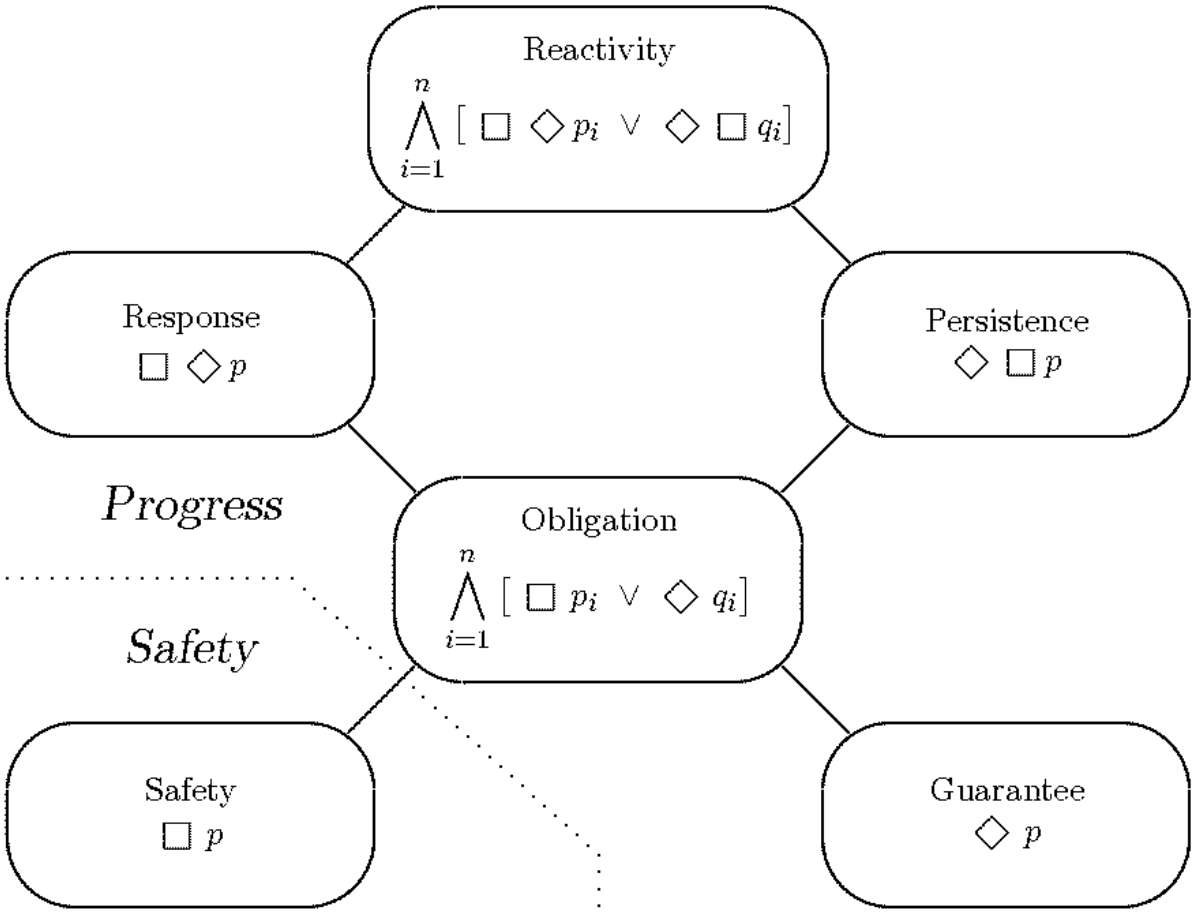
Zohar Manna

Classification Diagram (Fig. 0.18)

- For each $\kappa \in \{$safety, guarantee, obligation
  response, persistence, reactivity$\}$
  the κ class of temporal formulas is
  characterized by a canonical κ-formula,
  with p, q, p_i, q_i – past formulas

- A formula is a κ-formula if it is
  equivalent to a canonical κ-formula

- A property is a κ-property if it is
  specifiable by a κ-formula


Reactivity
$\bigwedge_{i=1}^{n} [\ \Box\ \Diamond p_i\ \vee\ \Diamond\ \Box\ q_i]$
Response
$\Box\ \Diamond p$
Persistence
$\Diamond\ \Box p$
Progress
Obligation
$\bigwedge_{i=1}^{n} [\ \Box\ p_i\ \vee\ \Diamond\ q_i]$
Safety
Safety
$\Box\ p$
Guarantee
$\Diamond\ p$

Closure of Classes

Reactivity: closure under $\wedge, \vee, \neg$

Persistence: closure under $\wedge, \vee$

$$\Diamond\Box p \wedge \Diamond\Box q \sim \Diamond\Box(p \wedge q)$$
$$\Diamond\Box p \vee \Diamond\Box q \sim \Diamond\Box(q \vee \ominus(p \mathcal{S}(p \wedge \neg q)))$$

Response: closure under $\wedge, \vee$

$$\Box\Diamond p \vee \Box\Diamond q \sim \Box\Diamond(p \vee q)$$
$$\Box\Diamond p \wedge \Box\Diamond q \sim \Box\Diamond(q \wedge \ominus((\neg q)\mathcal{S}p))$$

Obligation: closure under $\wedge, \vee, \neg$

Guarantee: closure under $\wedge, \vee$

$$\Diamond p \vee \Diamond q \sim \Diamond(p \vee q)$$
$$\Diamond p \wedge \Diamond q \sim \Diamond(\dot{\Diamond} p \wedge \dot{\Diamond} q)$$

Safety: closure under $\wedge, \vee$

$$\Box p \wedge \Box q \sim \Box(p \wedge q)$$
$$\Box p \vee \Box q \sim \Box(\dot{\Box} p \vee \dot{\Box} q)$$

Duality of classes

- Safety vs. Guarantee

  $\neg \Box p \sim \Diamond \neg p$

  $\neg \Diamond p \sim \Box \neg p$

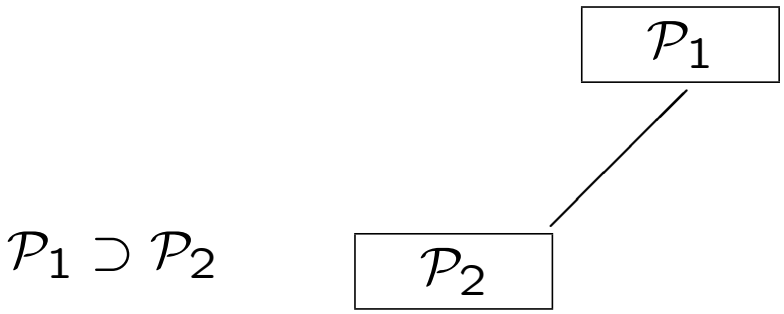
- Response vs. Persistence

  $\neg \Box \Diamond p \sim \Diamond \Box \neg p$

  $\neg \Diamond \Box p \sim \Box \Diamond \neg p$

Classification Diagram

- strict inclusion between boxes



Example: Obligation $\subset$ Persistence

$$(\Box\, p_i \vee \Diamond\, q_i) \;\sim\; \Diamond\, \Box (\boxminus\, p_i \vee \diamondminus\, q_i)$$

Theorem: Every *quantifier free temporal* formula is equivalent to a *reactivity* formula.

Classification Diagram Con't

- strict inclusion between conjunctions
  (Obligation and Reactivity)

  In Obligation

$$\bigwedge_{i=1}^{n+1} [\, \Box\, p_i \vee \Diamond\, q_i \,] \quad \supset \quad \bigwedge_{i=1}^{n} [\, \Box\, p_i \vee \Diamond\, q_i \,]$$

  In Reactivity

$$\bigwedge_{i=1}^{n+1} [\, \Box \Diamond\, p_i \vee \Diamond \Box\, q_i \,] \quad \supset \quad \bigwedge_{i=1}^{n} [\, \Box \Diamond\, p_i \vee \Diamond \Box\, q_i \,]$$

Note:

Properties specified by state formulas are safety properties and guarantee properties, since

$$p \sim \Box(first \rightarrow p)$$

$$p \sim \Diamond(first \wedge p)$$

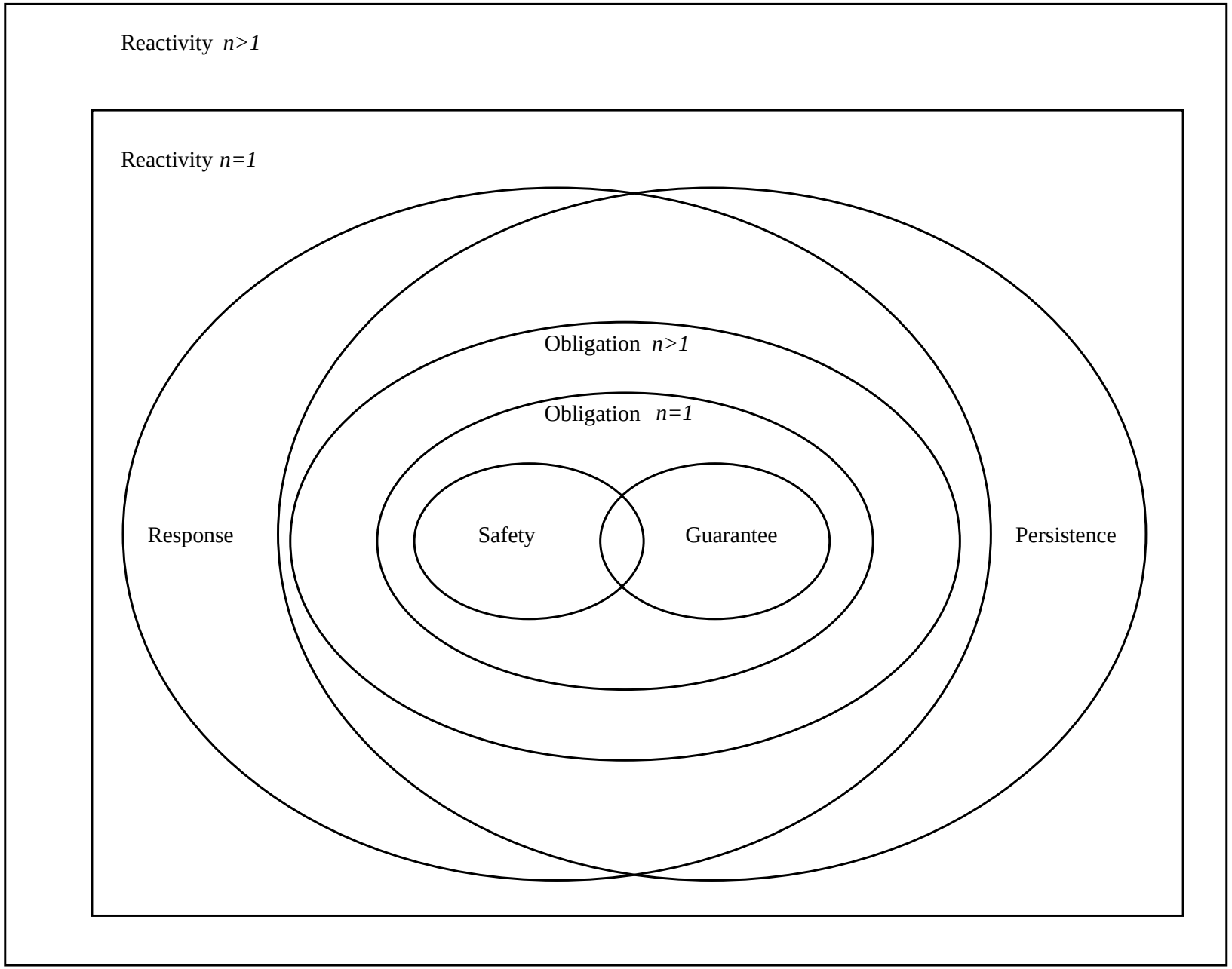
but also $\bigcirc p, \bigcirc \bigcirc p, \ldots$ since

$$\bigcirc p \sim \Box(\ominus first \rightarrow p)$$

$$\bigcirc p \sim \Diamond(\ominus first \wedge p)$$

$$\bigcirc \bigcirc p \sim \Box(\ominus \ominus first \rightarrow p)$$

$$\bigcirc \bigcirc p \sim \Diamond(\ominus \ominus first \wedge p)$$


Reactivity n>1
Reactivity n=1
Obligation n>1
Obligation n=1
Response
Safety
Guarantee
Persistence

Example Formulas

- Safety $\Box\, p$

  conditional safety

  $$p \rightarrow \Box\, q \quad \sim \quad \Box(\diamondminus(p \wedge first) \rightarrow\ q)$$
  $$p \Rightarrow \Box\, q \quad \sim \quad \Box(\diamondminus\, p\ \rightarrow\ q)$$

  waiting-for

  $$p\, \mathcal{W}\, q \quad \sim \quad \Box(\diamondminus\, \neg p\ \rightarrow\ \diamondminus\, q)$$

- Guarantee $\Diamond\, p$

  conditional guarantee

  $$p \rightarrow \Diamond\, q \quad \sim \quad \Diamond\Big(\diamondminus(first \wedge p) \rightarrow q\Big)$$

  until

  $$p\, \mathcal{U}\, q \quad \sim \quad \Diamond(q \wedge \widehat{\boxminus} p)$$

Example formulas (Con't)

- Obligation $\bigwedge_{i=1}^{n+1} (\Box\, p_i \vee \Diamond\, q_i)$

$$p\, \mathcal{W}\, (\Diamond\, q) \quad \sim \quad \Box\, p \;\vee\; \Diamond\, q$$

- Response $\Box \Diamond\, p$

  response

$$p \Rightarrow \Diamond\, q \quad \sim \quad \Box \Diamond \Big((\neg p)\, \mathcal{B}\, q \Big)$$

  justice

$$\Box \Diamond (\neg \mathit{enabled}(\tau) \vee \mathit{last\text{-}taken}(\tau))$$

  where

$$\mathit{enabled}(\tau): \quad \exists V' \,.\, \rho_\tau(V, V')$$

Example formulas (Con't)

- Persistence $\Diamond\Box p$

  conditional stabilization

  $$p \Rightarrow \Diamond\Box q \quad \sim \quad \Diamond\Box(\hat{\Diamond} p \rightarrow q)$$

- Reactivity $\bigwedge_{i=1}^{n+1} (\Diamond\Box p_i \vee \Box\Diamond q_i)$

  compassion

  $$\Box\Diamond enabled(\tau) \rightarrow \Box\Diamond last\text{-}taken(\tau)$$

  insistence

  $$\Box\Diamond p \Rightarrow \Diamond q \quad \sim \quad \Box\Diamond q \vee \Diamond\Box \neg p$$

Temporal Logic and Programs:
Examples

Control Predicates

at_ℓ $\quad [\ell] \in \pi$

$at\_\ell_{i,j}$ $\quad at\_\ell_i \ \vee \ at\_\ell_j$

$at\_\ell_{i...j}$ $\quad at\_\ell_i \ \vee \ at\_\ell_{i+1} \ \vee \ \ldots \ \vee \ at\_\ell_j$

Example 1: Program BINOM

Compute the binomial coefficient $\binom{n}{k}$

where $0 \leq k \leq n$

$$b = \frac{n \cdot (n-1) \cdots y_1 \cdots (n-k+1)}{1 \cdot 2 \cdots y_2 \cdots k}$$

property of integers:

a product of m consecutive integers
is evenly divisible by $m!$

Program BINOM (Fig. 120)

in	k, n	: **integer**	**where**	$0 \le k \le n$
local	y_1, y_2, r	: **integer**	**where**	$y_1 = n, y_2 = 1, r = 1$
out	b	: **integer**	**where**	$b = 1$

$$
P_1 :: \left[\begin{array}{l}
\textbf{local } t_1\textbf{: integer} \\
\ell_0: \ \textbf{while } y_1 > (n-k) \textbf{ do} \\
\qquad \left[\begin{array}{l}
\ell_1: \ \textbf{request}(r) \\
\boxed{\begin{array}{l}\ell_2: \ t_1 := b \cdot y_1 \\ \ell_3: \ b := t_1\end{array}} \\
\ell_4: \ \textbf{release}(r) \\
\ell_5: \ y_1 := y_1 - 1
\end{array}\right] \\
\ell_6:
\end{array}\right]
$$

$$
\Big|\Big|
$$

$$
P_2 :: \left[\begin{array}{l}
\textbf{local } t_2\textbf{: integer} \\
m_0: \ \textbf{while } y_2 \le k \textbf{ do} \\
\qquad \left[\begin{array}{l}
m_1: \ \textbf{await } (y_1 + y_2) \le n \\
m_2: \ \textbf{request}(r) \\
\boxed{\begin{array}{l}m_3: \ t_2 := b \textbf{ div } y_2 \\ m_4: \ b := t_2\end{array}} \\
m_5: \ \textbf{release}(r) \\
m_6: \ y_2 := y_2 + 1
\end{array}\right] \\
m_7:
\end{array}\right]
$$

Program BINOM : Total Correctness

- termination

$$\Diamond\, [at\_\ell_6 \ \wedge \ at\_m_7]$$

- partial correctness

$$\Box \left[at\_\ell_6 \ \wedge \ at\_m_7 \ \rightarrow \ b = \binom{n}{k} \right]$$

- total correctness

$$\Diamond \left[at\_\ell_6 \ \wedge \ at\_m_7 \ \wedge \ b = \binom{n}{k} \right]$$

Program BINOM : Auxiliary Properties

- global invariant

$$\Box[(n-k) \leq y_1 \leq n \ \wedge \ 1 \leq y_2 \leq k+1]$$

- deadlock freedom

$$\Box[[at\_\ell_1 \ \wedge \ at\_m_2] \ \rightarrow \ r = 1]$$

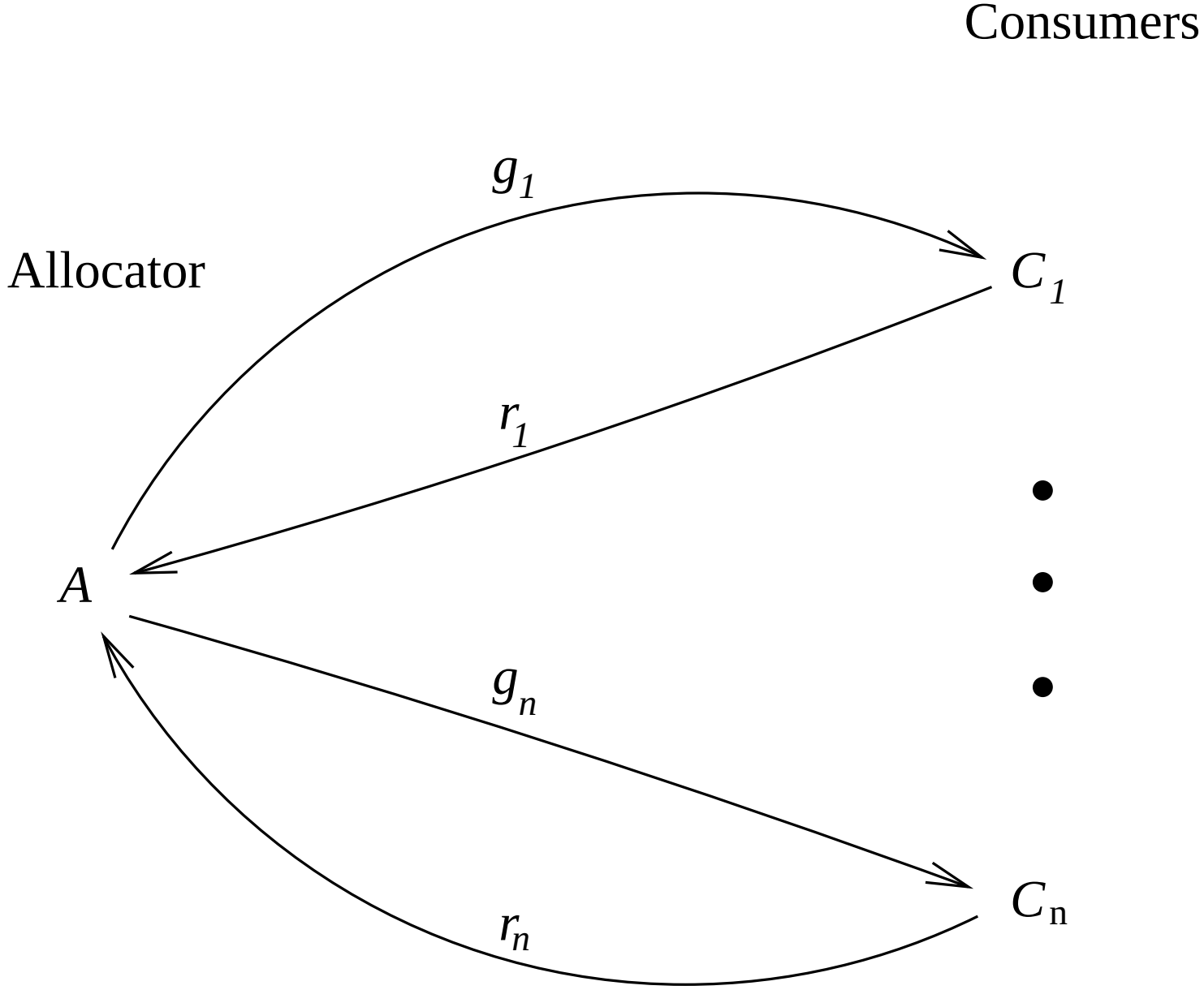
- fault freedom

$$\Box[at\_m_3 \ \rightarrow \ [y_2 \neq 0 \ \wedge \ (b \bmod y_2) = 0]]$$

- mutual exclusion

$$\Box[\neg(at\_\ell_{2..4} \ \wedge \ at\_m_{3..5})]$$

Example 2: Resource-Allocator Program



C_i requests	—	$r_i := \text{T}$
A grants	—	$g_i := \text{T}$
C_i releases	—	$r_i := \text{F}$
A acknowledges	—	$g_i := \text{F}$

Properties

- mutual exclusion

$$\Box(\sum g_i \leq 1)$$

$1 \rightarrow \text{T}, \quad 0 \rightarrow \text{F}$

- conformance with protocol

$$\begin{aligned}(\neg g_i) &\Rightarrow (\neg g_i)\ \mathcal{W}\ (\neg g_i \wedge r_i)\\ r_i &\Rightarrow r_i\ \mathcal{W}\ (r_i \wedge g_i)\\ g_i &\Rightarrow g_i\ \mathcal{W}\ (g_i \wedge \neg r_i)\\ (\neg r_i) &\Rightarrow (\neg r_i)\ \mathcal{W}\ (\neg r_i \wedge \neg g_i)\end{aligned}$$

- 1-bounded overtaking

$$r_i \Rightarrow (\neg g_j)\ \mathcal{W}\quad g_j\ \mathcal{W}\ (\neg g_j)\ \mathcal{W}\ g_i$$
for every j, $j \neq i$

- liveness

$$\begin{aligned}r_i &\Rightarrow \Diamond g_i\\ g_i &\Rightarrow \Diamond(\neg r_i)\\ (\neg r_i) &\Rightarrow \Diamond(\neg g_i)\end{aligned}$$

Example 3A: Program PRIME

$$\begin{array}{l}\textbf{local}\ \ y\text{: }\ \textbf{integer}\ \ \textbf{where}\ y = 1 \\ \qquad\quad \ell_0 :\ \ \textbf{loop forever do} \\ \cdots \Big|\Big| \begin{bmatrix} & \vdots \\ \ell_5 : & \textbf{print}(y) \\ \ell_6 : & \vdots \end{bmatrix} \Big|\Big| \cdots \end{array}$$

output: 2, 3, 5, 7, 11, 13, $\cdots$

$$\underbrace{\text{printed}(u)}_{j}\text{:}\ \ \underbrace{\ominus\, at\_\ell_5}_{j-1} \wedge \underbrace{at\_\ell_6}_{j} \wedge \underbrace{y = u}_{j}$$

Why $\ominus\, at\_\ell_5$?

- **only primes**

  $\forall u.\ \ \text{printed}(u) \ \Rightarrow\ \text{prime}(u)$

- **all primes**

  $\forall u.\ \ \text{prime}(u) \ \rightarrow\ \Diamond\, \text{printed}(u)$

- **monotonicity**

  $\forall u, u'.\ \ \text{printed}(u) \ \Rightarrow$
  $\widehat{\Box}(\text{printed}(u') \ \rightarrow\ u' > u)$

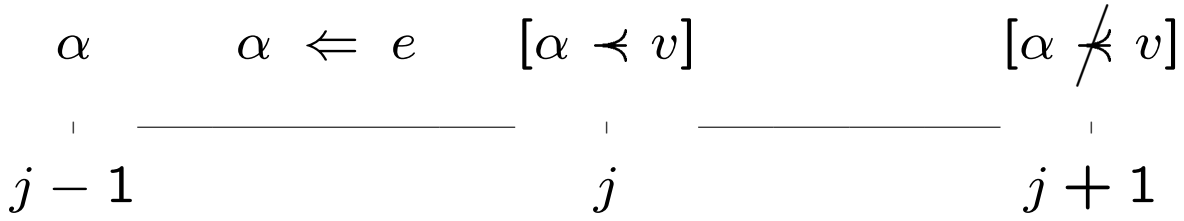
where u,u' are rigid

Asynchronous Communication

sending event (predicate)

$$[\alpha \prec v]\colon \quad \neg \mathit{first} \ \wedge \ \alpha = \alpha^{-} \bullet v$$

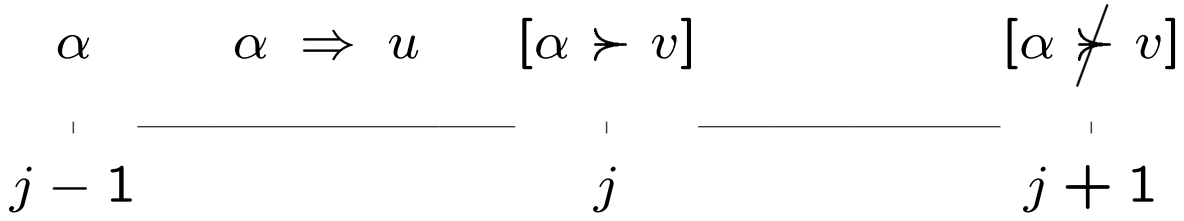
"The value v (of e) has just been sent to α"



receiving event

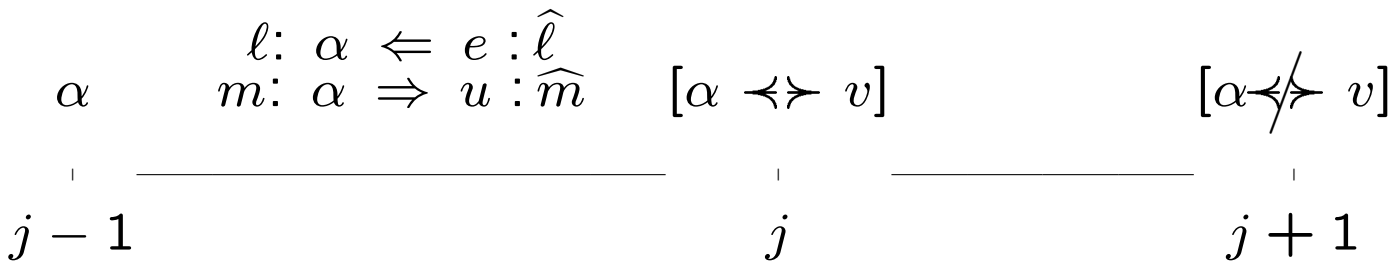
$$[\alpha \succ v]\colon \quad \neg \mathit{first} \ \wedge \ v \bullet \alpha = \alpha^{-}$$

"The value v has just been received (in u)
from channel α"



Synchronous Communication

$[\alpha \prec\!\succ v]$



$$[\alpha \prec\!\succ v]\colon \bigvee_{\langle \ell, m \rangle} \left[\begin{array}{l} \{[\ell\,], [m]\} \subseteq\ \pi^- \quad \wedge \\ \\ \pi\ =\ (\pi^- - \{[\ell], [m]\}) \cup \{[\widehat{\ell}], [\widehat{m}]\} \wedge \\ \\ v = e^- \end{array} \right]$$

"A synchronous communication has just taken place"

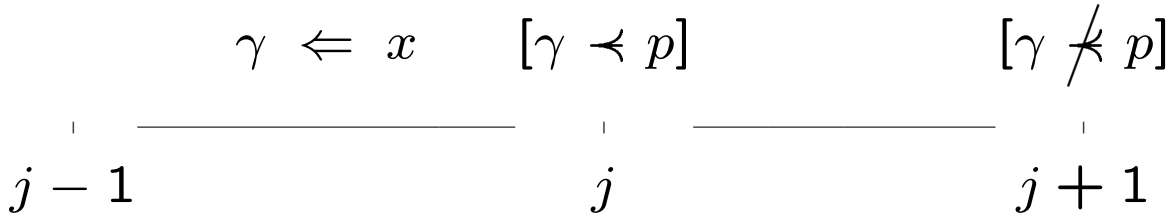
Note: $\bigvee$ ranges over all pairs of parallel $\Rightarrow, \Leftarrow$ statements on α.

Example 3B: Program PRIME

$$\ell_0: \quad \textbf{loop forever do}$$

$$\cdots \,||\, \begin{bmatrix} \vdots \\ \gamma \Leftarrow x \\ \vdots \end{bmatrix} \,||\, \cdots$$

output channel γ: 2, 3, 5, 7, 11, 13, $\cdots$



- **only primes**

  $\forall u.\ \ [\gamma \prec u] \ \Rightarrow\ \text{prime}(u)$

- **all primes**

  $\forall u.\ \ \text{prime}(u) \ \rightarrow\ \diamondsuit[\gamma \prec u]$

- **monotonicity**

  $\forall m, m'.\ ([\gamma \prec m] \ \wedge\ \widehat{\diamondsuit}[\gamma \prec m']) \ \Rightarrow\ m' < m$

Why $\widehat{\diamondsuit}$ and not $\diamondsuit$?

Verification: Motivating Example

Peterson's Algorithm
(for mutual exclusion)
Version 1 – INCORRECT

local y_1, y_2: **boolean** **where** $y_1 = \text{F}, y_2 = \text{F}$

ℓ_0 : **loop forever do**

$$P_1 :: \begin{bmatrix} \ell_1 : & \textbf{noncritical} \\ \ell_2 : & y_1 := \text{T} \\ \ell_3 : & \textbf{await } \neg y_2 \\ \ell_4 : & \textbf{critical} \\ \ell_5 : & y_1 := \text{F} \end{bmatrix}$$

$||$

m_0 : **loop forever do**

$$P_2 :: \begin{bmatrix} m_1 : & \textbf{noncritical} \\ m_2 : & y_2 := \text{T} \\ m_3 : & \textbf{await } \neg y_1 \\ m_4 : & \textbf{critical} \\ m_5 : & y_2 := \text{F} \end{bmatrix}$$

Peterson's Algorithm (Con't)

- protection variables: y_1, y_2

$y_1 = \text{F}$, $y_2 = \text{F}$	—	initially
$\ell_2:\ y_1 := \text{T}$	—	P_1 is interested
$\ell_3:\ \textbf{await}\ (\neg y_2) \vee \ldots$	—	P_1 waits
$\ell_5:\ y_1 := \text{F}$	—	P_1 resets y_1

Problem: potential deadlock
may reach ℓ_3, m_3 with $y_1 = y_2 = \text{T}$

$$\ldots \rightarrow \langle\{\ell_2, m_2\}, \overset{y_1}{\text{F}}, \overset{y_2}{\text{F}}\rangle \overset{\ell_2}{\rightarrow} \langle\{\ell_3, m_2\}, \text{T}, \text{F}\rangle$$
$$\overset{m_2}{\rightarrow} \langle\{\ell_3, m_3\}, \text{T}, \text{T}\rangle \overset{\tau_I}{\rightarrow} \ldots$$

Fix: add signature variable s

$\ell_2:\ s := 1$	—	P_1 requests priority
$\ell_3:\ \textbf{await}\ \ldots \vee (s \underset{\uparrow}{=} 2)$	—	P_1 has priority

P_2 was the last to request priority

Peterson's Algorithm
Version 2 (signature variable) – CORRECT

$$\textbf{local}\quad \begin{array}{lll} y_1, y_2: & \textbf{boolean} & \textbf{where } y_1 = \text{F}, y_2 = \text{F} \\ s: & \textbf{integer} & \textbf{where } s = 1 \end{array}$$

$$P_1 :: \quad \ell_0 : \ \textbf{loop forever do} \quad \left[\begin{array}{ll} \ell_1 : & \textbf{noncritical} \\ \ell_2 : & (y_1,\ s) := (\text{T},\ 1) \\ \ell_3 : & \textbf{await } (\neg y_2) \vee (s = 2) \\ \ell_4 : & \textbf{critical} \\ \ell_5 : & y_1 := \text{F} \end{array}\right]$$

$$\Big|\Big|$$

$$P_2 :: \quad m_0 : \ \textbf{loop forever do} \quad \left[\begin{array}{ll} m_1 : & \textbf{noncritical} \\ m_2 : & (y_2,\ s) := (\text{T},\ 2) \\ m_3 : & \textbf{await } (\neg y_1) \vee (s = 1) \\ m_4 : & \textbf{critical} \\ m_5 : & y_2 := \text{F} \end{array}\right]$$

Peterson's Algorithm

Properties of version 2

Mutual Exclusion

$$\Box \neg (at\_\ell_4 \ \wedge \ at\_m_4)$$

1-Bounded Overtaking for P_1

$$at\_\ell_3 \ \Rightarrow \ (\neg at\_m_4) \, \mathcal{W} \, at\_m_4 \, \mathcal{W} \, (\neg at\_m_4) \, \mathcal{W} \, at\_\ell_4$$

Accessibility for P_1

$$at\_\ell_2 \ \Rightarrow \ \Diamond \, at\_\ell_4$$

Peterson's Algorithm
Version 3 (split assignments) – INCORRECT

local y_1, y_2: **boolean** **where** $y_1 = \text{F}, y_2 = \text{F}$
s : **integer** **where** $s = 1$

$P_1 ::$

ℓ_0 : **loop forever do**

$$\left[\begin{array}{ll} \ell_1: & \textbf{noncritical} \\ \boxed{\begin{array}{ll} \ell_2: & s := 1 \\ \ell_3: & y_1 := \text{T} \end{array}} & \\ \ell_4: & \textbf{await } (\neg y_2) \vee (s = 2) \\ \ell_5: & \textbf{critical} \\ \ell_6: & y_1 := \text{F} \end{array}\right]$$

$\| \|$

$P_2 ::$

m_0 : **loop forever do**

$$\left[\begin{array}{ll} m_1: & \textbf{noncritical} \\ \boxed{\begin{array}{ll} m_2: & s := 2 \\ m_3: & y_2 := \text{T} \end{array}} & \\ m_4: & \textbf{await } (\neg y_1) \vee (s = 1) \\ m_5: & \textbf{critical} \\ m_6: & y_2 := \text{F} \end{array}\right]$$

$$(y_1, s) := (\text{T}, 1) \quad \rightarrow \quad \boxed{\begin{array}{ll} \ell_2\text{:} & s := 1 \\ \ell_3\text{:} & y_1 := \text{T} \end{array}}$$

$$(y_2, s) := (\text{T}, 2) \quad \rightarrow \quad \boxed{\begin{array}{ll} m_2\text{:} & s := 2 \\ m_3\text{:} & y_2 := \text{T} \end{array}}$$

Problem: violation of mutual exclusion

$$\begin{array}{lll} \ldots & \rightarrow & \langle \{\ell_3, m_3\},\ 2,\ \overset{s}{\text{F}},\ \overset{y_1}{\text{F}},\ \overset{y_2}{\text{F}} \rangle \\ & \overset{m_3}{\rightarrow} & \langle \{\ell_3, m_4\},\ 2,\ \text{F},\ \text{T} \rangle \\ & \overset{m_4}{\rightarrow} & \langle \{\ell_3, m_5\},\ 2,\ \text{F},\ \text{T} \rangle \\ & \overset{\ell_3}{\rightarrow} & \langle \{\ell_4, m_5\},\ 2,\ \text{T},\ \text{T} \rangle \\ & \overset{\ell_4}{\rightarrow} & \langle \{\ell_5, m_5\},\ 2,\ \text{T},\ \text{T} \rangle \\ & \rightarrow & \ldots \end{array}$$

Fix: reverse the statements

$$(y_1, s) := (\text{T}, 1) \quad \rightarrow \quad \boxed{\begin{array}{ll} \ell_2\text{:} & y_1 := \text{T} \\ \ell_3\text{:} & s := 1 \end{array}}$$

$$(y_2, s) := (\text{T}, 2) \quad \rightarrow \quad \boxed{\begin{array}{ll} m_2\text{:} & y_2 := \text{T} \\ m_3\text{:} & s := 2 \end{array}}$$

Peterson's Algorithm
Version 4 (reverse statements)
CORRECT???

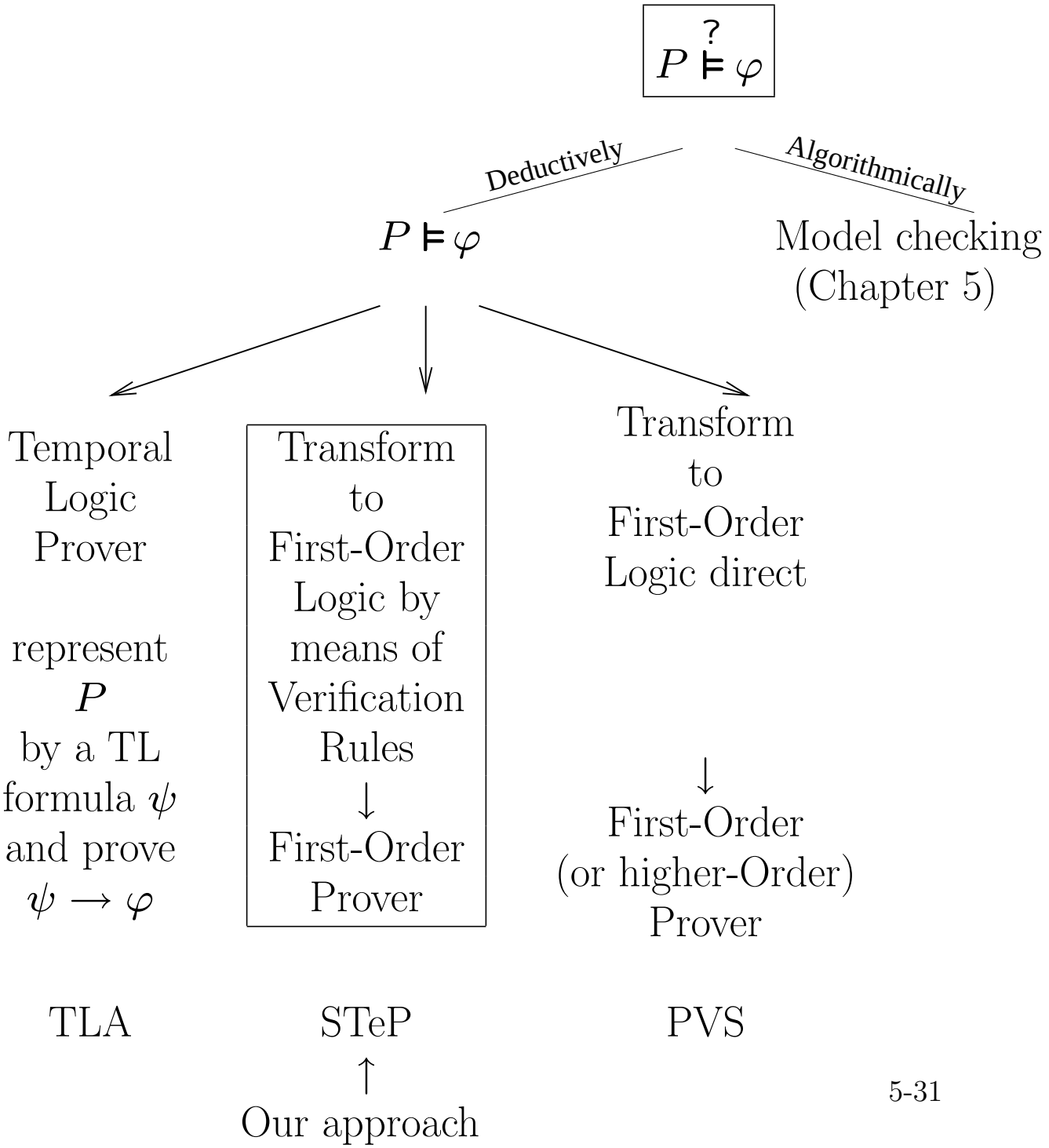
$$\textbf{local}\quad y_1, y_2\colon\ \textbf{boolean}\quad \textbf{where } y_1 = \text{F}, y_2 = \text{F}$$
$$\qquad\quad s\ \colon\ \textbf{integer}\quad \textbf{where } s = 1$$

$$P_1 :: \quad \ell_0 :\ \textbf{loop forever do} \left[\begin{array}{ll} \ell_1 : & \textbf{noncritical} \\ \boxed{\begin{array}{ll} \ell_2 : & y_1 := \text{T} \\ \ell_3 : & s := 1 \end{array}} & \\ \ell_4 : & \textbf{await } (\neg y_2) \vee (s = 2) \\ \ell_5 : & \textbf{critical} \\ \ell_6 : & y_1 := \text{F} \end{array}\right]$$

$$||$$

$$P_2 :: \quad m_0 :\ \textbf{loop forever do} \left[\begin{array}{ll} m_1 : & \textbf{noncritical} \\ \boxed{\begin{array}{ll} m_2 : & y_2 := \text{T} \\ m_3 : & s := 2 \end{array}} & \\ m_4 : & \textbf{await } (\neg y_1) \vee (s = 1) \\ m_5 : & \textbf{critical} \\ m_6 : & y_2 := \text{F} \end{array}\right]$$

Proving temporal properties of reactive systems



Proving temporal properties of reactive systems (Con't)

Textbook: Proof methods for safety properties:

$P \models \Box\, \varphi$, where φ is a past formula (no future operators)

Chapters 1, 2: Proof methods for invariance properties:

$P \models \Box\, q$, where q is a state formula (no temporal operators)

Vol III of textbook: Proof methods for progress properties.