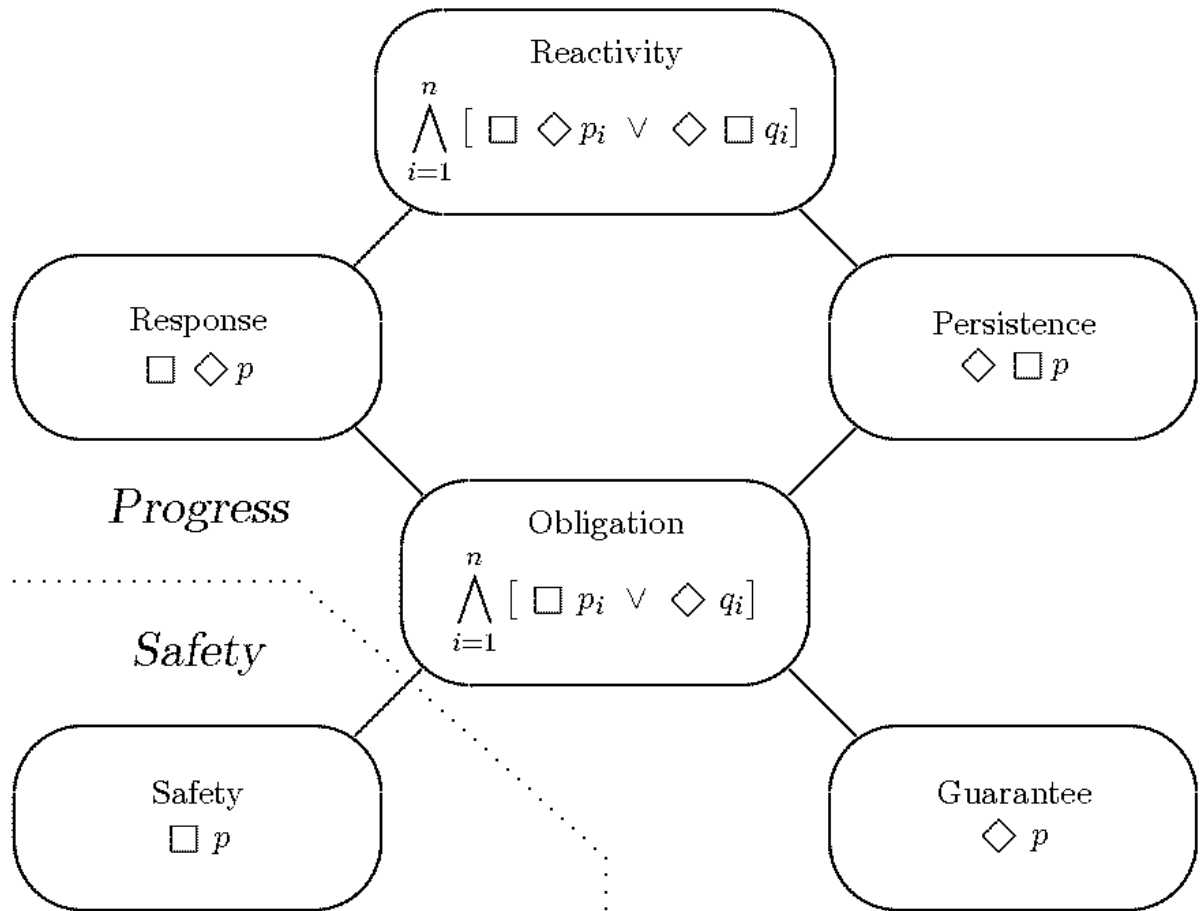


CS256/Spring 2008 — Lecture #5

Zohar Manna

Classification Diagram (Fig. 0.18)

- For each $\kappa \in \{\text{safety, guarantee, obligation response, persistence, reactivity}\}$ 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



Closure of Classes

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

Persistence: closure under \wedge, \vee

$$\diamond \square p \wedge \diamond \square q \sim \diamond \square (p \wedge q)$$

$$\diamond \square p \vee \diamond \square q \sim \diamond \square (q \vee \ominus (p \mathcal{S} (p \wedge \neg q)))$$

Response: closure under \wedge, \vee

$$\square \diamond p \vee \square \diamond q \sim \square \diamond (p \vee q)$$

$$\square \diamond p \wedge \square \diamond q \sim \square \diamond (q \wedge \ominus ((\neg q) \mathcal{S} p))$$

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

Guarantee: closure under \wedge, \vee

$$\diamond p \vee \diamond q \sim \diamond (p \vee q)$$

$$\diamond p \wedge \diamond q \sim \diamond (\ominus p \wedge \ominus q)$$

Safety: closure under \wedge, \vee

$$\square p \wedge \square q \sim \square (p \wedge q)$$

$$\square p \vee \square q \sim \square (\boxminus p \vee \boxminus q)$$

Duality of classes

- Safety vs. Guarantee

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

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

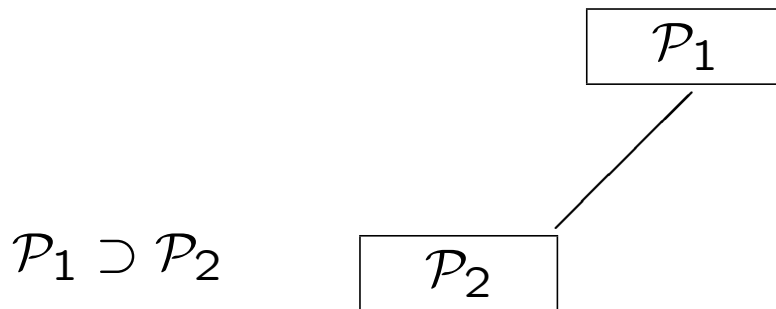
- Response vs. Persistence

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

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

Classification Diagram

- strict inclusion between boxes



Example: Obligation \subset Persistence

$$(\Box p_i \vee \Diamond q_i) \sim \Diamond \Box (\Box p_i \vee \Diamond q_i)$$

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

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

In Reactivity

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

Note:

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

$$p \sim \Box(\textit{first} \rightarrow p)$$

$$p \sim \Diamond(\textit{first} \wedge p)$$

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

$$\bigcirc p \sim \Box(\ominus \textit{first} \rightarrow p)$$

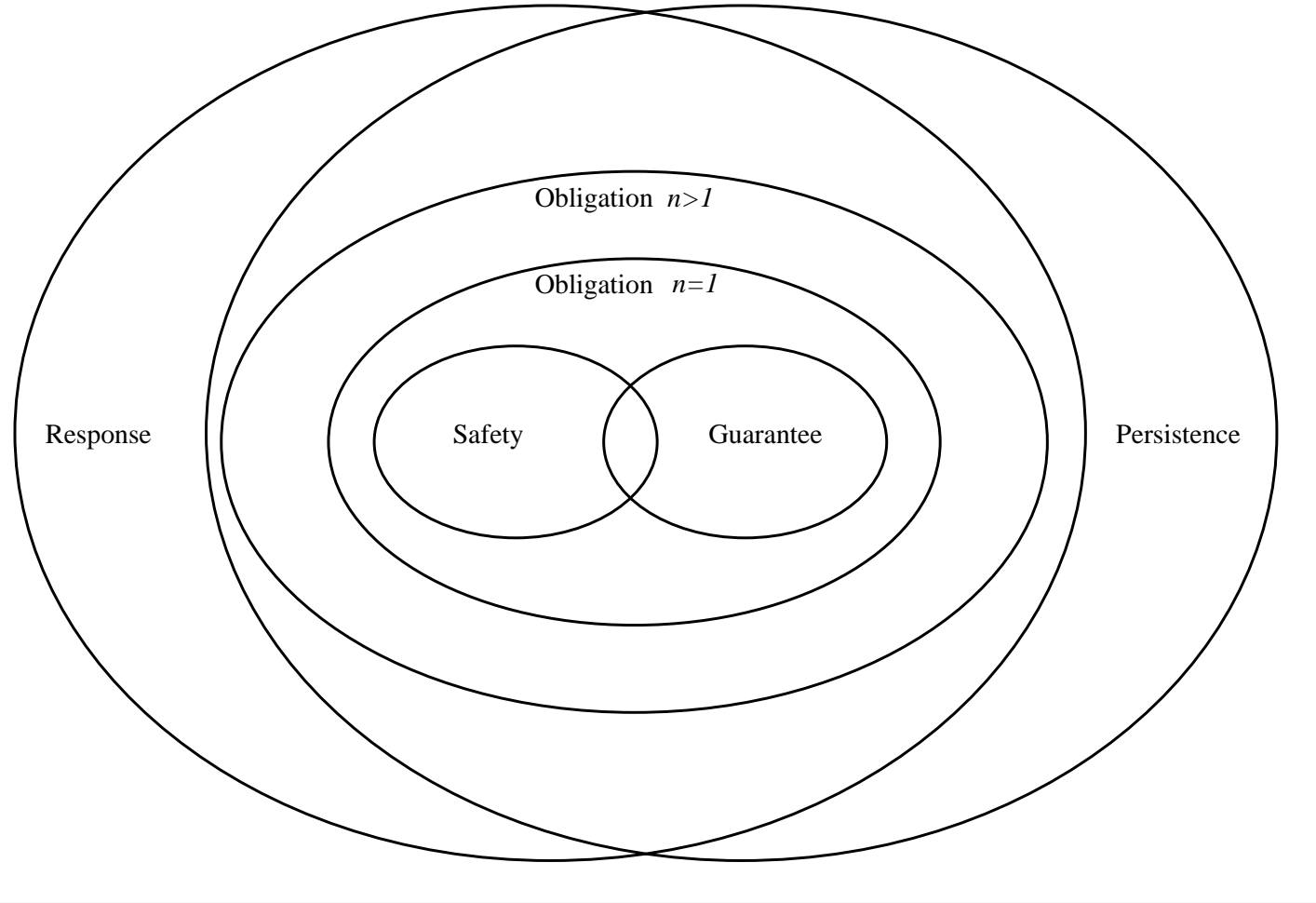
$$\bigcirc p \sim \Diamond(\ominus \textit{first} \wedge p)$$

$$\bigcirc \bigcirc p \sim \Box(\ominus \ominus \textit{first} \rightarrow p)$$

$$\bigcirc \bigcirc p \sim \Diamond(\ominus \ominus \textit{first} \wedge p)$$

Reactivity $n > 1$

Reactivity $n = 1$



Example Formulas

- Safety $\Box p$

conditional safety

$$p \rightarrow \Box q \sim \Box(\Diamond(p \wedge \textit{first}) \rightarrow q)$$

$$p \Rightarrow \Box q \sim \Box(\Diamond p \rightarrow q)$$

waiting-for

$$p \mathcal{W} q \sim \Box(\Diamond \neg p \rightarrow \Diamond q)$$

- Guarantee $\Diamond p$

conditional guarantee

$$p \rightarrow \Diamond q \sim \Diamond(\Diamond(\textit{first} \wedge p) \rightarrow q)$$

until

$$p \mathcal{U} q \sim \Diamond(q \wedge \widehat{\Box} p)$$

Example formulas (Con't)

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

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

- Response $\Box \Diamond p$

response

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

justice

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

where

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

Example formulas (Con't)

- Persistence $\diamond \square p$

conditional stabilization

$$p \Rightarrow \diamond \square q \quad \sim \quad \diamond \square (\diamond p \rightarrow q)$$

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

compassion

$$\square \diamond \textit{enabled}(\tau) \rightarrow \square \diamond \textit{last-taken}(\tau)$$

insistence

$$\square \diamond p \Rightarrow \diamond q \quad \sim \quad \square \diamond q \vee \diamond \square \neg p$$

| |
|--|
| Temporal Logic and Programs: Examples |
|--|

Control Predicates

$$at_l \quad [l] \in \pi$$

$$at_l_{i,j} \quad at_l_i \vee at_l_j$$

$$at_l_{i\dots j} \quad at_l_i \vee at_l_{i+1} \vee \dots \vee at_l_j$$

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

$P_1 ::$

$$\left[\begin{array}{l} \text{local } t_1 : \text{integer} \\ l_0 : \text{ while } y_1 > (n - k) \text{ do} \\ \quad \left[\begin{array}{l} l_1 : \text{ request}(r) \\ \quad \left[\begin{array}{l} l_2 : t_1 := b \cdot y_1 \\ l_3 : b := t_1 \end{array} \right] \\ l_4 : \text{ release}(r) \\ l_5 : y_1 := y_1 - 1 \end{array} \right] \\ l_6 : \end{array} \right]$$

$P_2 ::$

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

Program BINOM : Total Correctness

- termination

$$\diamond [at_{-l_6} \wedge at_{-m_7}]$$

- partial correctness

$$\square \left[at_{-l_6} \wedge at_{-m_7} \rightarrow b = \binom{n}{k} \right]$$

- total correctness

$$\diamond \left[at_{-l_6} \wedge at_{-m_7} \wedge b = \binom{n}{k} \right]$$

Program BINOM : Auxiliary Properties

- global invariant

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

- deadlock freedom

$$\square[[at_{-l_1} \wedge at_{-m_2}] \rightarrow r = 1]$$

- fault freedom

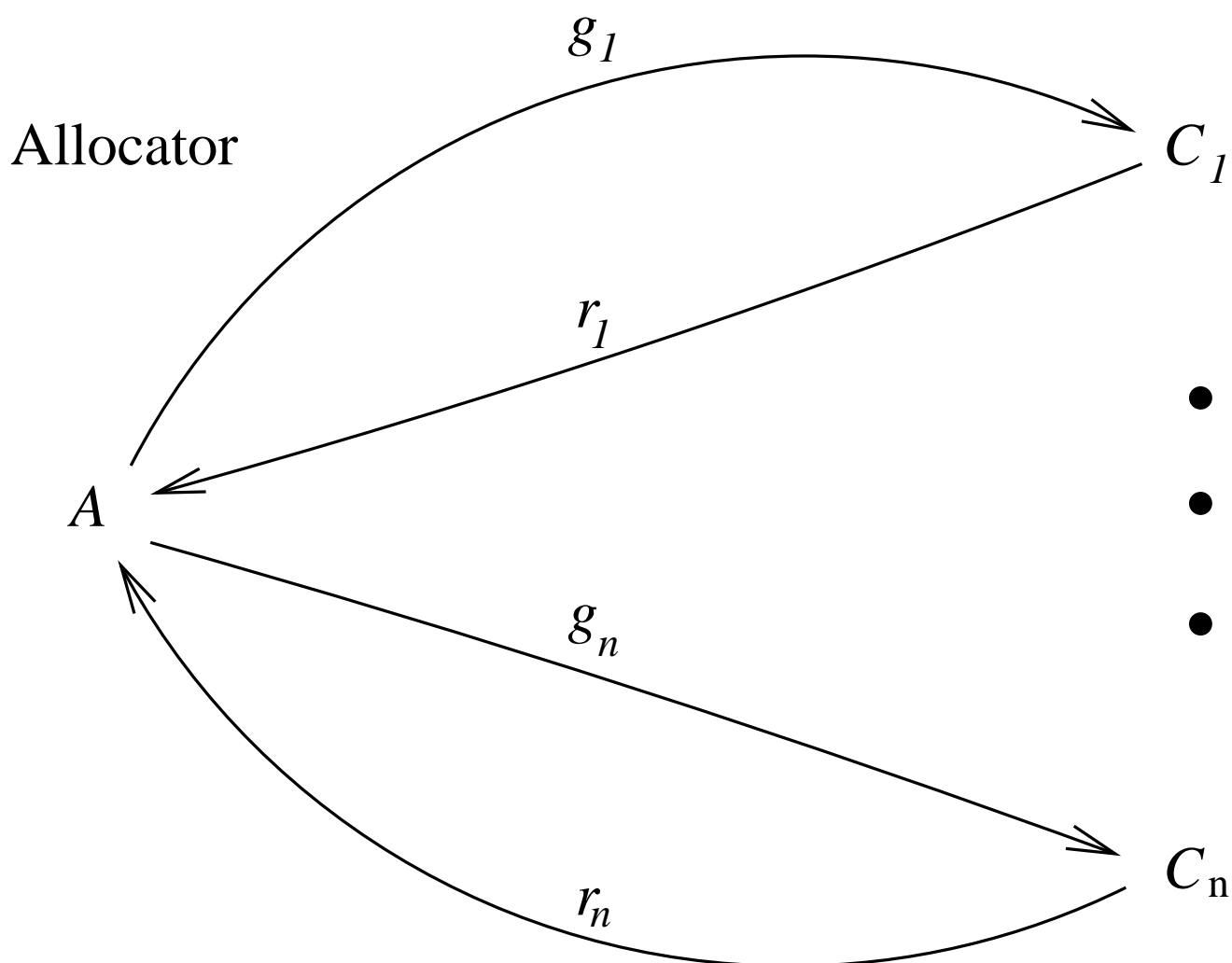
$$\square[at_{-m_3} \rightarrow [y_2 \neq 0 \wedge (b \bmod y_2) = 0]]$$

- mutual exclusion

$$\square[\neg(at_{-l_{2..4}} \wedge at_{-m_{3..5}})]$$

Example 2: Resource-Allocator Program

Consumers



C_i requests — $r_i := T$

A grants — $g_i := T$

C_i releases — $r_i := F$

A acknowledges — $g_i := F$

Properties

- mutual exclusion

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

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

- conformance with protocol

$$(\neg g_i) \Rightarrow (\neg g_i) \mathcal{W} (\neg g_i \wedge r_i)$$

$$r_i \Rightarrow r_i \mathcal{W} (r_i \wedge g_i)$$

$$g_i \Rightarrow g_i \mathcal{W} (g_i \wedge \neg r_i)$$

$$(\neg r_i) \Rightarrow (\neg r_i) \mathcal{W} (\neg r_i \wedge \neg g_i)$$

- 1-bounded overtaking

$$r_i \Rightarrow (\neg g_j) \mathcal{W} g_j \mathcal{W} (\neg g_j) \mathcal{W} g_i$$

for every $j, j \neq i$

- liveness

$$r_i \Rightarrow \Diamond g_i$$

$$g_i \Rightarrow \Diamond (\neg r_i)$$

$$(\neg r_i) \Rightarrow \Diamond (\neg g_i)$$

Example 3A: Program PRIME

```

local  $y$ : integer where  $y = 1$ 
       $\ell_0$  : loop forever do
           $\vdots$ 
           $\dots$   $||$   $\left[ \begin{array}{l} \vdots \\ \ell_5 : \mathbf{print}(y) \\ \ell_6 : \vdots \end{array} \right]$   $||$   $\dots$ 

```

output: 2, 3, 5, 7, 11, 13, \dots

$$\underbrace{\text{printed}(u)}_j : \underbrace{\ominus at_l_5}_{j-1} \wedge \underbrace{at_l_6}_j \wedge \underbrace{y = u}_j$$

Why $\ominus at_l_5$?

- **only primes**

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

- **all primes**

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

- **monotonicity**

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

where u, u' are rigid

Asynchronous Communication

sending event (predicate)

$$[\alpha \leftarrow v]: \quad \neg first \wedge \alpha = \alpha^- \bullet v$$

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

$$\begin{array}{ccc}
 \alpha & \alpha \Leftarrow e & [\alpha \leftarrow v] & & [\alpha \not\leftarrow v] \\
 | & \hline & | & \hline & | \\
 j - 1 & & j & & j + 1
 \end{array}$$

receiving event

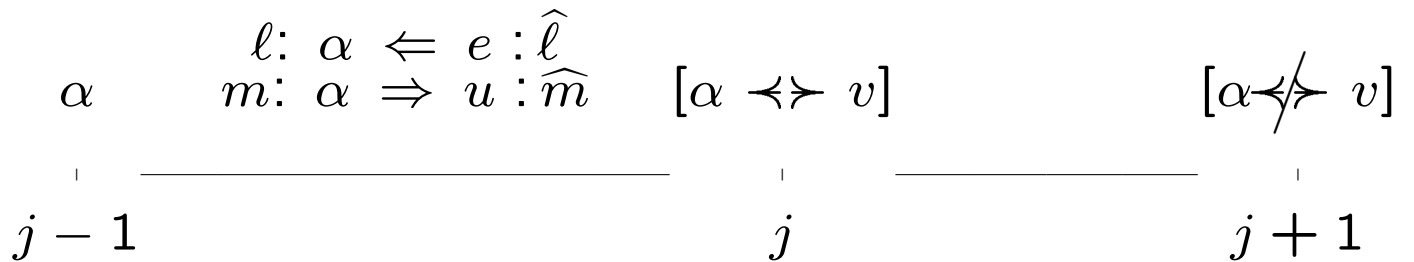
$$[\alpha \succ v]: \quad \neg first \wedge v \bullet \alpha = \alpha^-$$

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

$$\begin{array}{ccc}
 \alpha & \alpha \Rightarrow u & [\alpha \succ v] & & [\alpha \not\succ v] \\
 | & \hline & | & \hline & | \\
 j - 1 & & j & & j + 1
 \end{array}$$

Synchronous Communication

$$[\alpha \leftarrow\!\!\rightarrow v]$$



$$[\alpha \leftarrow\!\!\rightarrow v]: \bigvee_{\langle l, m \rangle} \left[\begin{array}{l} \{[l], [m]\} \subseteq \pi^- \wedge \\ \pi = (\pi^- - \{[l], [m]\}) \cup \{[\hat{l}], [\hat{m}]\} \wedge \\ v = e^- \end{array} \right]$$

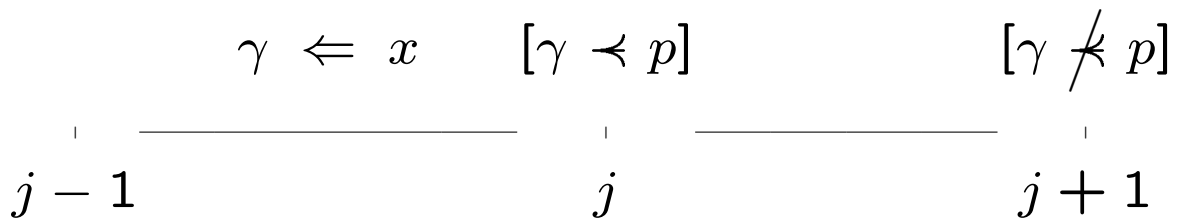
“A synchronous communication has just taken place”

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

Example 3B: Program PRIME

$\ell_0 : \text{ loop forever do}$
 $\dots \parallel \left[\begin{array}{c} \vdots \\ \gamma \Leftarrow x \\ \vdots \end{array} \right] \parallel \dots$

output channel γ : 2, 3, 5, 7, 11, 13, ...



- **only primes**

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

- **all primes**

$$\forall u. \text{prime}(u) \rightarrow \Diamond[\gamma \Leftarrow u]$$

- **monotonicity**

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

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

Verification: Motivating Example

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

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

P_1 ::

l_0 : **loop forever do**

| | |
|---------|-------------------------|
| l_1 : | noncritical |
| l_2 : | $y_1 := \text{T}$ |
| l_3 : | await $\neg y_2$ |
| l_4 : | critical |
| l_5 : | $y_1 := \text{F}$ |

||

P_2 ::

m_0 : **loop forever do**

| | |
|---------|-------------------------|
| m_1 : | noncritical |
| m_2 : | $y_2 := \text{T}$ |
| m_3 : | await $\neg y_1$ |
| m_4 : | critical |
| m_5 : | $y_2 := \text{F}$ |

Peterson's Algorithm (Con't)

- protection variables: y_1, y_2

| | | |
|--|---|---------------------|
| $y_1 = F, y_2 = F$ | — | initially |
| $l_2 : y_1 := T$ | — | P_1 is interested |
| $l_3 : \mathbf{await} (\neg y_2) \vee \dots$ | — | P_1 waits |
| $l_5 : y_1 := F$ | — | P_1 resets y_1 |

Problem: potential deadlock

may reach l_3, m_3 with $y_1 = y_2 = T$

$$\begin{aligned} \dots &\rightarrow \langle \{l_2, m_2\}, \overset{y_1}{F}, \overset{y_2}{F} \rangle \xrightarrow{l_2} \langle \{l_3, m_2\}, T, F \rangle \\ &\xrightarrow{m_2} \langle \{l_3, m_3\}, T, T \rangle \xrightarrow{\tau_I} \dots \end{aligned}$$

Fix: add signature variable s

| | | |
|--|---|-------------------------|
| $l_2 : s := 1$ | — | P_1 requests priority |
| $l_3 : \mathbf{await} \dots \vee (s \underset{\uparrow}{=} 2)$ | — | P_1 has priority |
| P_2 was the last to request priority | | |

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

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

l_0 : **loop forever do**

P_1 :: $\left[\begin{array}{l} l_1 : \text{noncritical} \\ l_2 : (y_1, s) := (\text{T}, 1) \\ l_3 : \text{await } (\neg y_2) \vee (s = 2) \\ l_4 : \text{critical} \\ l_5 : y_1 := \text{F} \end{array} \right]$

||

m_0 : **loop forever do**

P_2 :: $\left[\begin{array}{l} m_1 : \text{noncritical} \\ m_2 : (y_2, s) := (\text{T}, 2) \\ m_3 : \text{await } (\neg y_1) \vee (s = 1) \\ m_4 : \text{critical} \\ m_5 : y_2 := \text{F} \end{array} \right]$

Peterson's Algorithm
Properties of version 2

Mutual Exclusion

$$\square \neg(at_{-l_4} \wedge at_{-m_4})$$

1-Bounded Overtaking for P_1

$$at_{-l_3} \Rightarrow (\neg at_{-m_4}) \mathcal{W} at_{-m_4} \mathcal{W} (\neg at_{-m_4}) \mathcal{W} at_{-l_4}$$

Accessibility for P_1

$$at_{-l_2} \Rightarrow \diamond at_{-l_4}$$

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 $s = 1$

l_0 : **loop forever do**

$P_1 ::$

| |
|--|
| l_1 : noncritical |
| l_2 : $s := 1$ |
| l_3 : $y_1 := \text{T}$ |
| l_4 : await $(\neg y_2) \vee (s = 2)$ |
| l_5 : critical |
| l_6 : $y_1 := \text{F}$ |

||

m_0 : **loop forever do**

$P_2 ::$

| |
|--|
| m_1 : noncritical |
| m_2 : $s := 2$ |
| m_3 : $y_2 := \text{T}$ |
| m_4 : await $(\neg y_1) \vee (s = 1)$ |
| m_5 : critical |
| m_6 : $y_2 := \text{F}$ |

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

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

Problem: violation of mutual exclusion

$$\begin{aligned} \dots &\rightarrow \langle \{\ell_3, m_3\}, \overset{s}{2}, \overset{y_1}{F}, \overset{y_2}{F} \rangle \xrightarrow{m_3} \langle \{\ell_3, m_4\}, 2, F, T \rangle \\ &\xrightarrow{m_4} \langle \{\ell_3, m_5\}, 2, F, T \rangle \xrightarrow{\ell_3} \langle \{\ell_4, m_5\}, 2, T, T \rangle \\ &\xrightarrow{\ell_4} \langle \{\ell_5, m_5\}, 2, T, T \rangle \rightarrow \dots \end{aligned}$$

Fix: reverse the statements

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

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

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

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

ℓ_0 : **loop forever do**

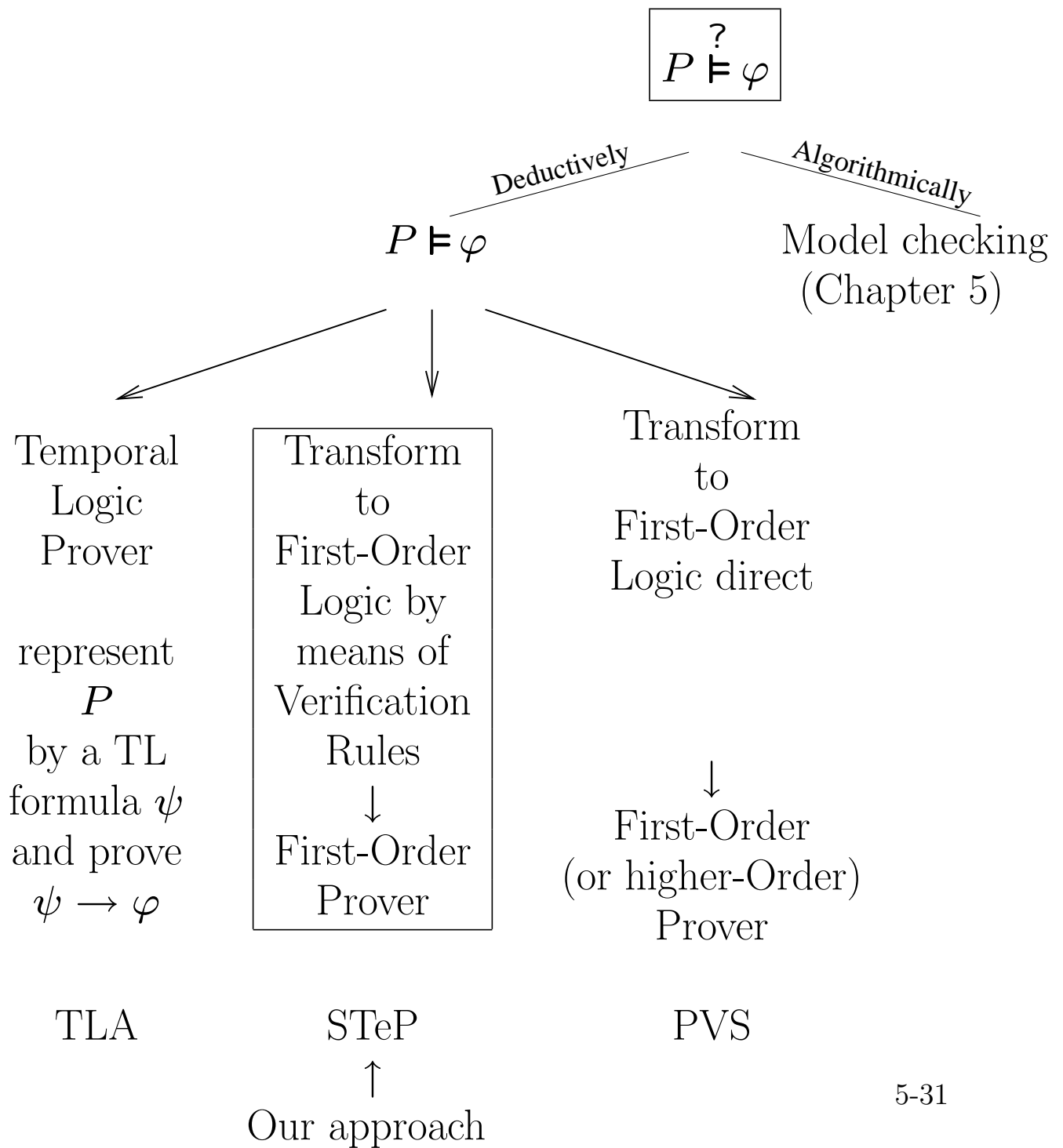
P_1 :: $\left[\begin{array}{l} \ell_1 : \text{noncritical} \\ \boxed{\ell_2 : y_1 := \text{T}} \\ \boxed{\ell_3 : s := 1} \\ \ell_4 : \text{await } (\neg y_2) \vee (s = 2) \\ \ell_5 : \text{critical} \\ \ell_6 : y_1 := \text{F} \end{array} \right]$

||

m_0 : **loop forever do**

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

Proving temporal properties of reactive systems



Proving temporal properties of reactive systems (Con't)

Textbook: Proof methods for safety properties:

$$P \models \Box \varphi, \quad \text{where } \varphi \text{ is a past formula} \\ \text{(no future operators)}$$

Chapters 1, 2: Proof methods for invariance properties:

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

Vol III of textbook Proof methods for progress properties.