# Concurrency pt. 2

Justin Reniers & Johan Sijtsma

Based on section 4-6 from ‘Resources, Concurrency and Local Reasoning’ by Peter W. O’Hearn

# Today

Recap

Process Interaction

Proof Rules

Semaphore Example

# Recap

Racy vs Race-Free

Separation Logic

Disjoint Concurrency

# Recap - Racy vs Race-free

- Racy
  - “Two concurrent processes attempt to access the same portion of state at the same time”

$$x := y + x \parallel x := x \times z$$

- Sequentially equivalent but distinguished by concurrency

$$x := x + 1; x := x + 1 \text{ and } x := x + 2$$

- Cautious or daring
- Ownership and Separation

# Recap - Separation Logic

- $s, h \vDash P$ State *s*, heap *h*
- $P * Q$ Meaning one part of the heap makes *P* hold, and another makes *Q* hold
- **emp** Heap is empty
- $a \mapsto b$ Address *a* owns value *b*

# Process Interaction

# Process Interaction

- Grammar

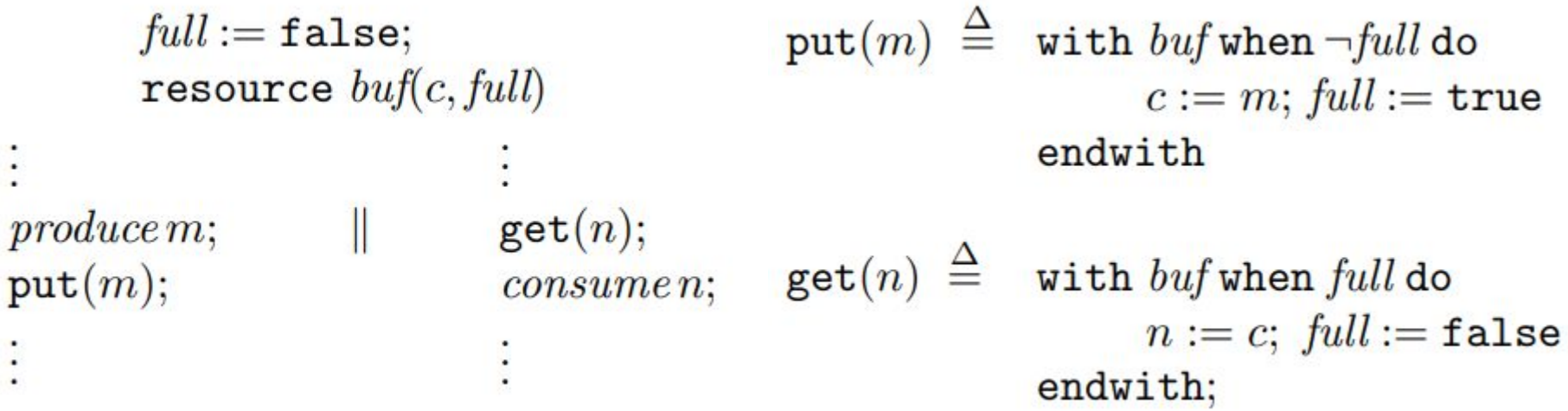
$$
\begin{array}{rcl}
C & ::= & x := E \mid x := [E] \mid [E] := F \mid x := \texttt{cons}(E_1, \ldots, E_n) \mid \texttt{dispose}(E) \\
& \mid & \texttt{skip} \mid C; C \mid \texttt{if } B \texttt{ then } C \texttt{ else } C \mid \texttt{while } B \texttt{ do } C \\
& \mid & \texttt{with } r \texttt{ when } B \texttt{ do } C \texttt{ endwith}
\end{array}
$$

- “Resource” Nomenclature

$$
\begin{array}{l}
\mathit{init}; \\
\texttt{resource } r_1(\text{variable list}), \ldots, r_m(\text{variable list}) \\
C_1 \parallel \cdots \parallel C_n
\end{array}
$$

# Process Interaction - Example

- Producer-Consumer relation



# Process Interaction - (Super) Semaphores

- Semaphores

$$\begin{aligned} \mathrm{P}(s) &= \texttt{with } s \texttt{ when } s > 0 \texttt{ do } s := s - 1 \texttt{ endwith} \\ \mathrm{V}(s) &= \texttt{with } s \texttt{ when true do } s := s + 1 \texttt{ endwith.} \end{aligned}$$

- Super Semaphores

$$\begin{aligned} \mathrm{P}'(s) &= \texttt{with } s \texttt{ when } s > 0 \texttt{ do auxiliary assignments; } s := s - 1 \texttt{ endwith} \\ \mathrm{V}'(s) &= \texttt{with } s \texttt{ when true do auxiliary assignments; } s := s + 1 \texttt{ endwith.} \end{aligned}$$

# Process Interaction - Extra Requirements

- Syntactic Restrictions
  1. A variable belongs to at most one resource
  2. If variable *x* belongs to resource *r*, it cannot appear in a parallel process except in a critical region for *r*
  3. If variable *x* is changed in one process, it cannot appear in another unless it belongs to a resource
- Do not avoid interference with pointers
  - Aliases

$$[x] := 3 \parallel [y] := 4$$

# Proof Rules

# Proof Rules

- We want to reason about a program
- $init;$
  $\texttt{resource}\ r_1(\text{variable list}), ..., r_m(\text{variable list})$
  $C_1 \parallel \cdots \parallel C_n$

- Resource Invariant $RI_{ri}$ for every resource $r_i$
- $RI_{ri}$ must satisfy that any `x := …` that is free in $RI_{ri}$ must occur in the critical region for $r_i$

# Proof Rules

- Resource Rule

$$\frac{\{P\}\mathit{init}\{RI_{r_1} * \cdots * RI_{r_m} * P'\} \qquad \{P'\}C_1 \parallel \cdots \parallel C_n\{Q\}}{\begin{array}{l}\{P\}\\ \mathit{init};\\ \texttt{resource}\ r_1(\text{variable list}), ..., r_m(\text{variable list})\\ C_1 \parallel \cdots \parallel C_n\\ \{RI_{r_1} * \cdots * RI_{r_m} * Q\}\end{array}}$$

# Proof Rules

- Parallel Composition

$$\frac{\{P_1\}\, C_1\, \{Q_1\} \;\cdots\; \{P_n\}\, C_n\, \{Q_n\}}{\{P_1 * \cdots * P_n\}\, C_1 \parallel \cdots \parallel C_n\, \{Q_1 * \cdots * Q_n\}}$$

Where no free variable in $P_i$ or $Q_i$ is changed in $C_j$ when $j$ != $i$

- Critical Regions

$$\frac{\{(P * RI_r) \wedge B\}\, C\, \{Q * RI_r\}}{\{P\}\ \texttt{with}\ r\ \texttt{when}\ B\ \texttt{do}\ C\ \texttt{endwith}\ \{Q\}}$$

No other process modifies variables free in $P$ or $Q$

# Proof Rules

- Assignment Deletion

$$\frac{\{P\}\mathtt{prog}'\{Q\}}{\{P\}\mathtt{prog}\{Q\}}$$

# Semaphore Example

# Semaphore Example

- Annotated Code

$$
\begin{array}{ccc}
\vdots & & \vdots \\
\{\mathtt{emp}\} & & \{\mathtt{emp}\} \\
\mathtt{P}(\mathit{free}); & & \mathtt{P}(\mathit{busy}); \\
\{10 \mapsto -\} & & \{10 \mapsto -\} \\
[10] := m; & \| & n := [10]; \\
\{10 \mapsto -\} & & \{10 \mapsto -\} \\
\mathtt{V}(\mathit{busy}); & & \mathtt{V}(\mathit{free}); \\
\{\mathtt{emp}\} & & \{\mathtt{emp}\} \\
\vdots & & \vdots
\end{array}
$$

# Semaphore Example

- Semaphore Invariant

$$RI_s \quad = \quad (s = 0 \wedge \mathtt{emp}) \vee (s = 1 \wedge 10 \mapsto -).$$

- Semaphore Proof Rules

$$\frac{\{(A * RI_s) \wedge s > 0\}\ s := s - 1\ \{A' * RI_s\}}{\{A\}\,\mathtt{P}(s)\{A'\}}$$

$$\frac{\{A * RI_s\}\ s := s + 1\ \{A' * RI_s)\}}{\{A\}\,\mathtt{V}(s)\{A'\}}$$

# Semaphore Example

- Obtaining $\{\mathtt{emp}\}\mathtt{P}(free)\{10 \mapsto -\}$

$$\{(\mathtt{emp} * ((free = 0 \wedge \mathtt{emp}) \vee (free = 1 \wedge 10 \mapsto -))) \wedge free > 0\}$$
$$\{free = 1 \wedge 10 \mapsto -\}$$
$$free := free - 1$$
$$\{free = 0 \wedge 10 \mapsto -\}$$
$$\{10 \mapsto - * (free = 0 \wedge \mathtt{emp})\}$$
$$\{10 \mapsto - * ((free = 0 \wedge \mathtt{emp}) \vee (free = 1 \wedge 10 \mapsto -))\}$$

# Semaphore Example

- Obtaining $\mathtt{V}(free)$ The Other Way Around

$$\{10 \mapsto - * ((free = 0 \wedge \mathtt{emp}) \vee (free = 1 \wedge 10 \mapsto -))\}$$
$$\{10 \mapsto - * (free = 0 \wedge \mathtt{emp})\}$$
$$free := free + 1$$
$$\{10 \mapsto - * (free = 1 \wedge \mathtt{emp})\}$$
$$\{\mathtt{emp} * (free = 1 \wedge 10 \mapsto -)\}$$
$$\{\mathtt{emp} * ((free = 0 \wedge \mathtt{emp}) \vee (free = 1 \wedge 10 \mapsto -))\}$$

# Semaphore Example

$\{10 \mapsto -\}$
$free := 1,\ busy := 0;$
$\{(free = 1 \wedge 10 \mapsto -) * (busy = 0 \wedge \texttt{emp})\}$
$\{RI_{free} * RI_{busy} * \texttt{emp} * \texttt{emp}\}$
`resource` $free(free),\ busy(busy);$
$\{\texttt{emp} * \texttt{emp}\}$

| | |
|---|---|
| $\{\texttt{emp}\}$ | $\{\texttt{emp}\}$ |
| `while true do` | `while true do` |
| $\{\texttt{emp} \wedge \texttt{true}\}$ | $\{\texttt{emp} \wedge \texttt{true}\}$ |
| $\{\texttt{emp}\}$ | $\{\texttt{emp}\}$ |
| $produce\ m$ | $\texttt{P}(busy);$ |
| $\{\texttt{emp}\}$ | $\{10 \mapsto -\}$ |
| $\texttt{P}(free);$ | $n := [10];$ |
| $\{10 \mapsto -\}$ | $\{10 \mapsto -\}$ |
| $[10] := m;$ ‖ | $\texttt{V}(free);$ |
| $\{10 \mapsto -\}$ | $\{\texttt{emp}\}$ |
| $\texttt{V}(busy);$ | $consume\ n$ |
| $\{\texttt{emp}\}$ | $\{\texttt{emp}\}$ |
| $\{\texttt{emp} \wedge \neg\texttt{true}\}$ | $\{\texttt{emp} \wedge \neg\texttt{true}\}$ |
| $\{\texttt{false}\}$ | $\{\texttt{false}\}$ |

$\{\texttt{false} * \texttt{false}\}$
$\{RI_{free} * RI_{busy} * \texttt{false}\}$
$\{\texttt{false}\}$

# Thank you for listening

Are there any questions