Resource tracking concurrent games

Aurore Alcolei, Pierre Clairambault, Olivier Laurent
ENS Lyon, LIP

Chocola seminar, Lyon – January, 24 2019

Time analysis

P =

```
newref r in
    wait(2) || wait(1)
    !r      || r := true
            || wait(2)
```

Semantics.

$$M \Downarrow v \qquad\qquad [\![M]\!] = [\![v]\!]\ , \qquad\qquad v \in \{\mathsf{true}, \mathsf{false}\}$$

Time analysis

P =

$$
\begin{array}{l}
\text{newref r in} \\
\quad \begin{array}{l|l}
\text{wait}(2) & \text{wait}(1) \\
\text{!r} & \text{r := true} \\
 & \text{wait}(2)
\end{array}
\end{array}
$$

Semantics.

$$M \Downarrow^{t} v \qquad \llbracket M \rrbracket = \llbracket v \rrbracket^{t}, \qquad v \in \{\text{true}, \text{false}\}$$

Q. What is the minimal amount of time necessary to run P?
- $\rightarrow$ to get true?
- $\rightarrow$ to get false?

What programs?

Concurrency
(interleavings)

Replication

Parametrisation

(shared)
Memory

Typed λ-calculus
(Higher order functions)

[Ghica05]
Slot games

[LMMP13]
Weighted relational
model

What programs?

Concurrency
(interleavings)

Replication

Parametrisation

(shared)
Memory

Typed λ-calculus
(Higher order functions)

[Ghica05]
Slot games

[LMMP13]
Weighted relational
model

What programs?

Concurrency (multi-core)

Replication

Parametrisation

(shared) Memory

Typed λ-calculus
(Higher order functions)

[Ghica05]
Slot games

[LMMP13]
Weighted relational model

The $\mathbb{R}$-IPA language

- Types:

$$
\begin{array}{lcl}
\mathcal{B} & := & \textbf{com} \mid \textbf{bool} \mid \textbf{mem}_R \mid \textbf{mem}_W \\
A, B & := & \mathcal{B} \mid A \multimap B
\end{array}
$$

- Syntax:

$$
\begin{array}{lcl}
M, N & := & \mid x \mid \lambda x.t \mid MN \\
 & & \mid \text{true} \mid \text{false} \mid \text{ifte} \;\; b \; M \; N \\
 & & \mid \text{skip} \mid M; N \mid M \;\; || \; N \mid \bot \\
 & & \mid \text{wait}(r) \text{ with } r \in \mathbb{R} \\
 & & \mid \text{newref } r \text{ in } M \mid !M \mid M := \text{true}
\end{array}
$$

Affine typing.

$$
\frac{\Gamma, r : \textbf{mem}_R, r : \textbf{mem}_W \vdash M : A}{\Gamma \vdash \text{newref } r \text{ in } M : A}
\qquad
\frac{\Gamma \vdash M : \textbf{bool} \quad \Delta \vdash N : \textbf{com}}{\Gamma, \Delta \vdash M \;||\; N : \textbf{bool}}
$$

Expressivity - examples

Coin

newref r in (r:= true $\|$!r)

Strictness testing.

$\lambda\ f^{\mathbf{com} \multimap \mathbf{com}}$. newref r in
(f (r:=true)) ; !r

Parallelism testing.

$\lambda\ f^{\mathbf{com} \multimap \mathbf{com} \multimap \mathbf{com}}$. newref x,y,$z_1$,$z_2$ in
f (if (!x) then (skip) else (z_1 := true ; y := true))
(if (!y) then (skip) else (z_2 := true ; x := true)) ;
(!z_1) and (!z_2)

The $\mathbb{R}$-IPA language (2)

- Interleaving-based small-step operational semantics:

$$\langle M, s, t\rangle, \quad \mathcal{L} \vdash M : B \text{ with } B \in \mathcal{B},\ t \in \mathbb{R},\ s : \mathcal{L} \to \{\text{true}, \text{false}\}$$

Reduction rules:

$$\langle (\lambda x.M)N, s, t\rangle \to \langle t[N/x], s, t\rangle \qquad \langle \text{wait}(r), s, t\rangle \to \langle \text{skip}, s, t + r\rangle$$

$$\langle \text{skip}; M, s, t\rangle \to \langle M, t\rangle \qquad \langle \text{skip} \mid\mid M, s, t\rangle \to \langle M, s, t\rangle$$

$$\langle v := \text{true}, s, t\rangle \to \langle \text{skip}, s[v \mapsto \text{true}], t\rangle \qquad \ldots$$

Contextual rules:

$$\frac{\langle M, s, t\rangle \to \langle M', s', t'\rangle}{\langle M||N, s, t\rangle \to \langle M'||N, s', t'\rangle} \qquad \ldots$$

Definition

For $v \in \{\text{true}, \text{false}, \text{skip}\}$, $\quad M \Downarrow^{t} v \quad$ iff $\quad \langle M, \emptyset, 0\rangle \to^{*} \langle v, s, t\rangle$.

Slot games [Ghica05]

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

0

Slot games [Ghica05]

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true        2
             ||  wait(2)
```

wait(2),

Slot games [Ghica05]

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true          2
             ||  wait(2)
```

wait(2), !r,

Slot games [Ghica05]


```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true        3
             ||  wait(2)
```

wait(2), !r, wait(1),

Slot games [Ghica05]

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true        3
             ||  wait(2)
```

wait(2), !r, wait(1), r:=true,

Slot games [Ghica05]

newref r in

wait(2)	wait(1)	
!r	r := true	5
	wait(2)	

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false

Slot games [Ghica05]

newref r in

wait(2)	wait(1)	
!r	r := true	2
	wait(2)	

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false

wait(2),

Slot games [Ghica05]

newref r in

wait(2)	wait(1)	
!r	r := true	3
	wait(2)	

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false
wait(2), wait(1),

Slot games [Ghica05]

newref r in
wait(2) || wait(1)
!r || r := true
|| wait(2)

3

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false
wait(2), wait(1), r:=true,

Slot games [Ghica05]

```
newref r in
    wait(2) || wait(1)
    !r      || r := true        3
            || wait(2)
```

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false
wait(2), wait(1), r:=true, !r,

Slot games [Ghica05]

newref r in			
wait(2)	wait(1)		
!r	r := true		5
	wait(2)		

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false

wait(2), wait(1), r:=true, !r, wait(2) $P \Downarrow^5$ true

...

Slot games [Ghica05]

newref r in

wait(2)	wait(1)
!r	r := true
	wait(2)

5

wait(2), !r, wait(1), r:=true, wait(2) $\quad P \Downarrow^5$ false
wait(2), wait(1), r:=true, !r, wait(2) $\quad P \Downarrow^5$ true
...

$[\![P]\!] = \{$run ② ① ② ff, run ② *red*① ② tt, $\ldots\}$

Computational adequacy: $\quad M \Downarrow^r v \quad$ iff $\quad \exists t \in [\![M]\!]$ st $|t| = r$

Slot games [Ghica05]

newref r in

wait(2)	wait(1)
!r	r := true
	wait(2)

5

wait(2), !r, wait(1), r:=true, wait(2) $P \Downarrow^5$ false

wait(2), wait(1), r:=true, !r, wait(2) $P \Downarrow^5$ true

...

$[\![P]\!] = \{\text{run } ⑤ \text{ ff}, \text{ run } ⑤ \text{ tt}\}$

Computational adequacy: $M \Downarrow^r v$ iff $\exists t \in [\![M]\!]$ st $|t| = r$

True concurrency?

Q: What about multicore systems?

```
newref (r,false) in
  wait(2) || wait(1)
  !r      || r := true        0
          || wait(2)
```

True concurrency?

Q: What about <u>multicore</u> systems?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true         2
             ||  wait(2)
```

True concurrency?

Q: What about multicore systems?

```
newref (r,false) in
    wait(2)   ||  wait(1)
    !r        ||  r := true          2
              ||  wait(2)
```

True concurrency?

Q: What about <u>multicore</u> systems?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true        2
             ||  wait(2)
```

True concurrency?

Q: What about multicore systems?

newref (r,false) in

wait(2)	wait(1)
!r	r := true
	wait(2)

4

$P \Downarrow^4$ false !

True concurrency?

Q: What about <u>multicore</u> systems?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true          1
             ||  wait(2)
```

True concurrency?

Q: What about multicore systems?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true        1
             ||  wait(2)
```

True concurrency?

Q: What about multicore systems?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

3

True concurrency?

Q: What about <u>multicore</u> systems?

```
newref (r,false) in
    wait(2) || wait(1)
    !r      || r := true
            || wait(2)
```

3

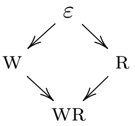
$P \Downarrow^{3}$ true !

True concurrency?

Q: What about multicore systems?

$$\frac{\langle M, s, t\rangle \to^{*} \langle M', s', t'\rangle \quad \langle N, s, t\rangle \to^{*} \langle N', s'', t''\rangle}{\langle M \mid\mid N, s, t\rangle \to^{*} \langle M' \mid\mid N', s' \bowtie s'', \max(t', t'')\rangle} \; s, s' \text{ non interfering}$$

$s : \mathcal{L} \to$



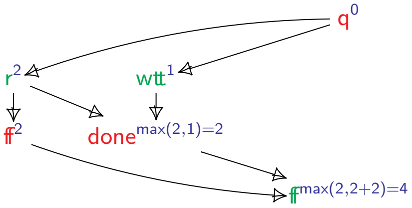
- s, s' are non interfering iff $\forall \ell \in \mathcal{L}$, $s(\ell) \leq s(\ell')$ or $s'(\ell) \leq s(\ell)$
- $s' \bowtie s''(\ell) = \bigvee(s'(\ell), s''(\ell))$

True concurrency?

newref (r,false) in

wait(2)	wait(1)
!r	r := true
	wait(2)

$P \Downarrow^4$ false

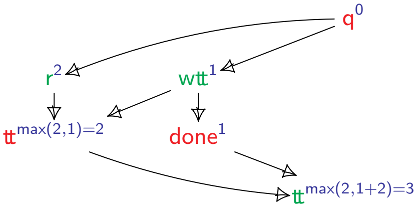


[CC16] + time annotations

True concurrency?

```
newref (r,false) in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$P \Downarrow^3$ true

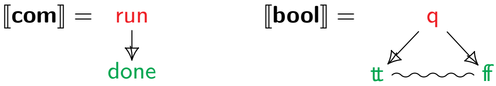


[CC16] + time annotations

Resource tracking concurrent games

Concurrent games

- Types as arenas



Definitions

An arena $(|A|, \leq_A, \#_A, \mathrm{pol}_A)$ is an event structure with polarity:

- $(|A|, \leq_A)$ a causal relation (**partial order**, with finite histories $[a]$),
- $\#_A$ a binary conflict relation (up-closed),
- $\mathrm{pol}_A : A \to \{-, +\}$,

that is

- **negative**: $\mathrm{pol}(\min(A)) = \{-\}$
- **well-threaded**: for all $a \in A$, $\min([a])$ is unic.

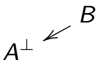
$\mathscr{C}(A)$ is the set of **configurations**: down-closed compatible subsets of A.

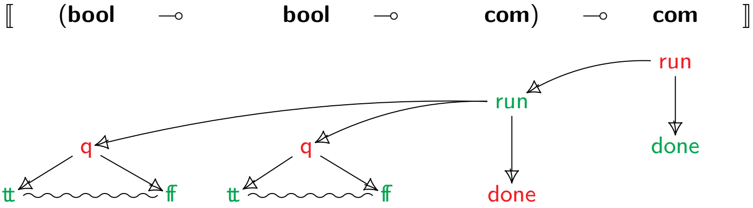
Concurrent arenas

Constructions on arenas.

- If A is a arena, $A^\perp$ has the same structure with polarity inverted.
- If A, B are arenas, $A \otimes B$ has events $|A| + |B|$, and components inherited.

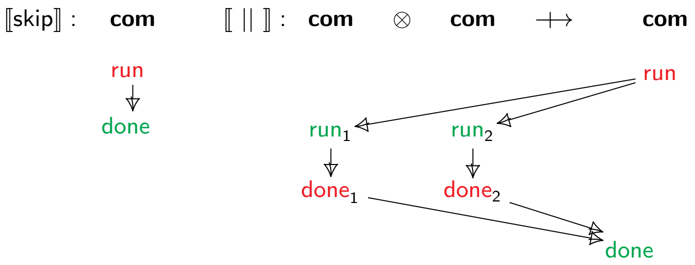
Linear map.

- If A, B are arena and B is rooted, $A \multimap B$ is ,
  i.e. $A^\perp \otimes B$ with extra relation $\min(B) \leq a$ for all $a \in |A|$.



Concurrent games

- Programs as strategies



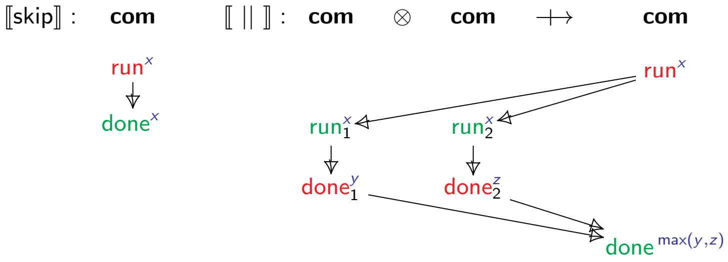
Definition

A play $(q, \leq_q) : A$ is a partial order s.t.:

- (rule respecting) $\mathscr{C}(q) \subseteq \mathscr{C}(A)$
- (courteous) $a \rightarrow b$
- (well-threaded)

Annotated concurrent games

- Programs as annotated strategies



Definition

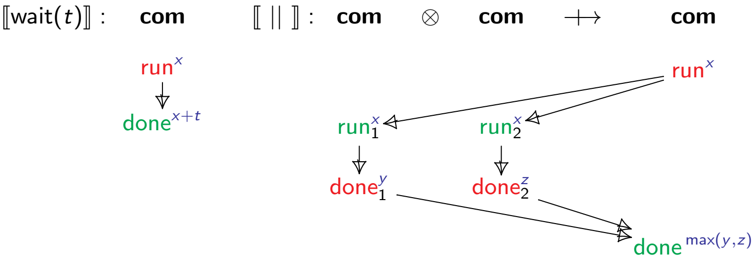
A play $(q, \leq_q) : A$ is a partial order s.t.:

- (rule respecting) $\mathscr{C}(q) \subseteq \mathscr{C}(A)$
- (courteous) $a \rightarrow b$
- (well-threaded)

A $\mathbb{R}$-annotation for q is a mapping $\lambda : (s : |q|^+) \to (\mathbb{R}^{[s]^-} \to \mathbb{R})$.

Annotated concurrent games

- Programs as annotated strategies



Definition

A play $(q, \leq_q) : A$ is a partial order s.t.:

- (rule respecting) $\mathscr{C}(q) \subseteq \mathscr{C}(A)$
- (courteous) $a \rightarrow b$
- (well-threaded)

A $\mathbb{R}$-annotation for q is a mapping $\lambda : (s : |q|^+) \to (\mathbb{R}^{[s]^-} \to \mathbb{R})$.

Concurrent games

- Programs as strategies

$[\![\text{skip}]\!]$: **com** $\qquad\qquad$ $[\![\text{coin}]\!]$: **bool**

Definition

A strategy $\sigma : A$ is a non empty set of plays that is:

- down-closed,
- (receptive) closed under $(-)$-extensions.

A strategy from A to B is $\sigma : A^{\perp} \otimes B$, written $\sigma : A \mathrel{\rightarrow\!\!\!+\!\!\!\!>} B$.

Concurrent games

- Programs as strategies

$[\![\text{skip}]\!]$: **com** $\quad\quad$ $[\![\text{coin}]\!]$: **bool**

Definition

A strategy $\sigma : A$ is a non empty set of plays that is:

- down-closed,
- (receptive) closed under $(-)$-extensions.

A strategy from A to B is $\sigma : A^{\perp} \otimes B$, written $\sigma : A \twoheadrightarrow B$.

Composition. $\tau \odot \sigma : A \twoheadrightarrow C$ is defined for all $\sigma : A \twoheadrightarrow B, \tau : B \twoheadrightarrow C$.

Annotated concurrent games

- Programs as annotated strategies

$[\![\mathrm{wait}(t)]\!]$: **com** $\qquad [\![\, \mathrm{P} \,]\!]$: **bool**

Definition

A $\mathbb{R}$-strategy $\sigma : A$ is a non empty set of $\mathbb{R}$-annotated plays that is:

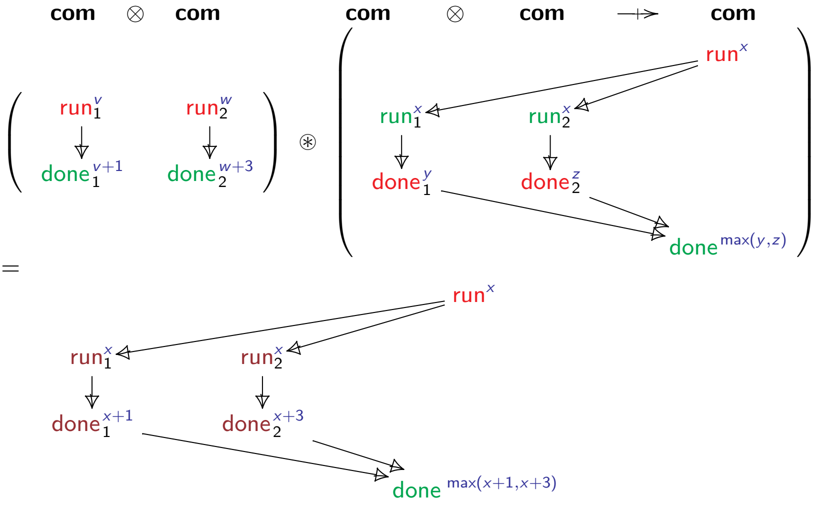
- down-closed,
- (receptive) closed under $(-)$-extensions.

A $\mathbb{R}$-strategy from A to B is $\sigma : A^{\perp} \otimes B$, written $\sigma : A \rightarrow\!\!\!\!\!+\!\!\!\!\!> B$.

Composition. $\tau \odot \sigma : A \rightarrow\!\!\!\!\!+\!\!\!\!\!> C$ is defined for all $\sigma : A \rightarrow\!\!\!\!\!+\!\!\!\!\!> B, \tau : B \rightarrow\!\!\!\!\!+\!\!\!\!\!> C$.

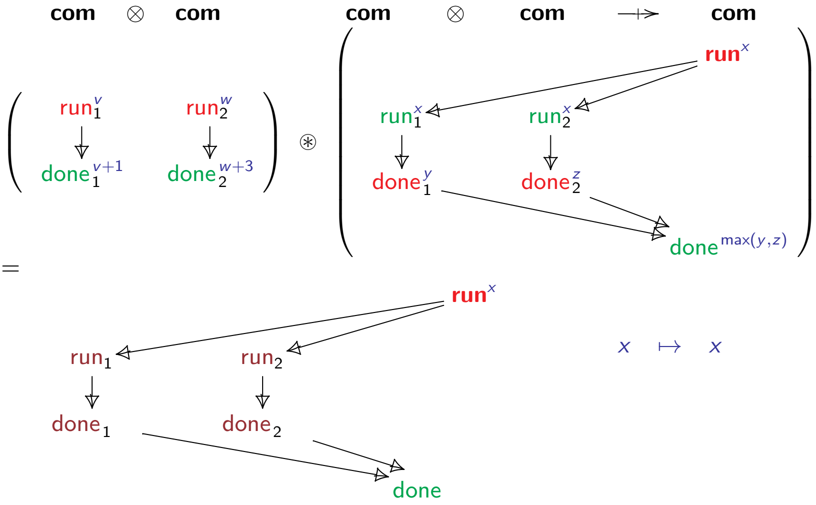
Composition example

$$[\![\mathsf{wait}(1) \mid\mid \mathsf{wait}(3)]\!] \quad = \quad [\![\ \mid\mid\ ]\!] \odot ([\![\mathsf{wait}(1)]\!] \otimes [\![\mathsf{wait}(3)]\!])$$



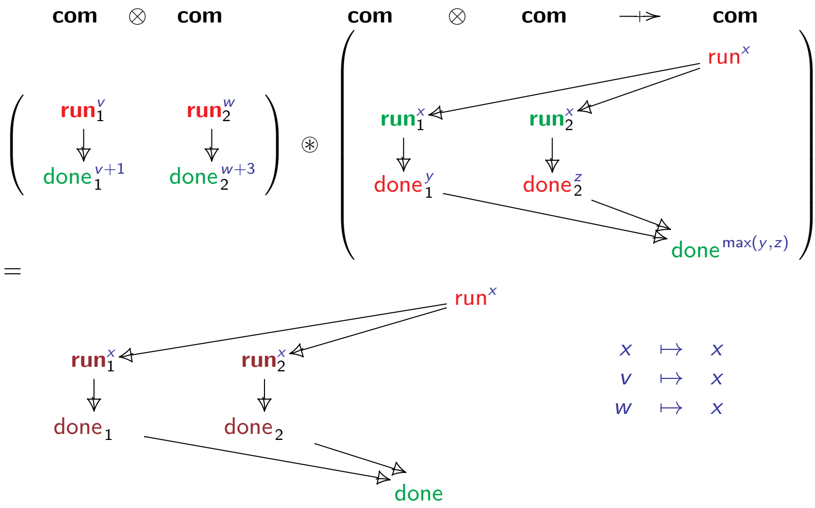
Composition example

$$[\![\mathrm{wait}(1) \parallel \mathrm{wait}(3)]\!] \quad = \quad [\![\ \parallel\ ]\!] \odot ([\![\mathrm{wait}(1)]\!] \otimes [\![\mathrm{wait}(3)]\!])$$



Composition example

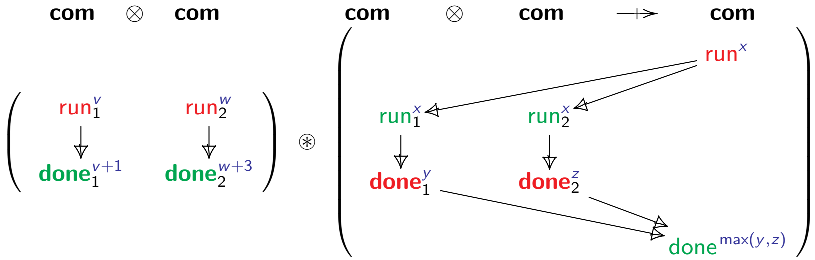
$$[\![\text{wait}(1) \parallel \text{wait}(3)]\!] \quad = \quad [\![\ \parallel\ ]\!] \odot ([\![\text{wait}(1)]\!] \otimes [\![\text{wait}(3)]\!])$$



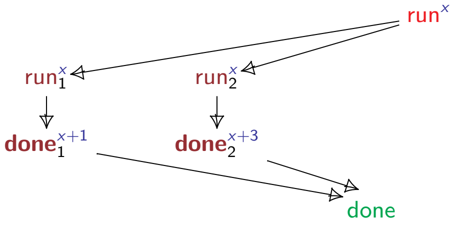
$$x \mapsto x$$
$$v \mapsto x$$
$$w \mapsto x$$

Composition example

$$[\![\text{wait}(1) \mid\mid \text{wait}(3)]\!] \quad = \quad [\![\ \mid\mid\ ]\!] \odot ([\![\text{wait}(1)]\!] \otimes [\![\text{wait}(3)]\!])$$



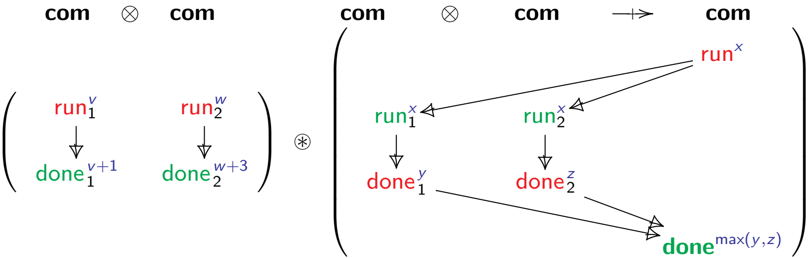
$=$

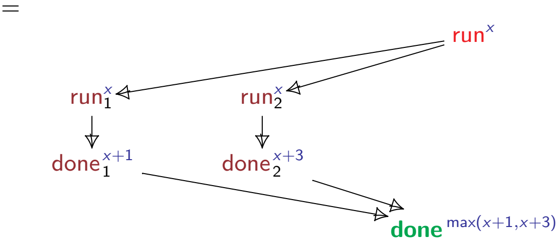


$$\begin{array}{lcl} x & \mapsto & x \\ v & \mapsto & x \\ w & \mapsto & x \\ y & \mapsto & x+1 \\ z & \mapsto & x+3 \end{array}$$

Composition example

$$[\![\mathrm{wait}(1) \mid\mid \mathrm{wait}(3)]\!] \quad = \quad [\![\ \mid\mid\ ]\!] \odot ([\![\mathrm{wait}(1)]\!] \otimes [\![\mathrm{wait}(3)]\!])$$

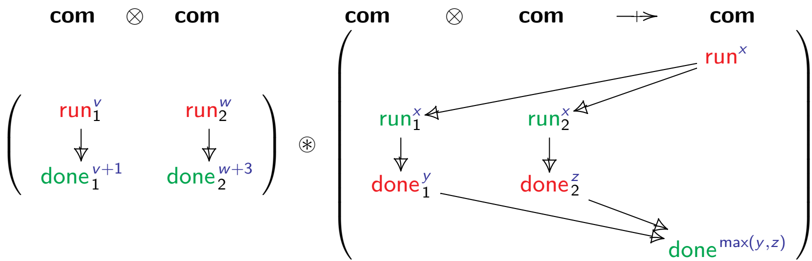




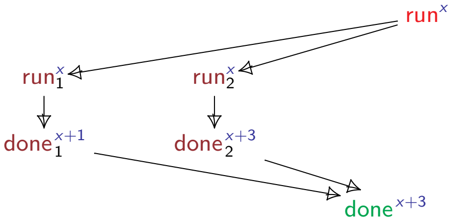
$$\begin{aligned} x &\mapsto x \\ v &\mapsto x \\ w &\mapsto x \\ y &\mapsto x+1 \\ z &\mapsto x+3 \end{aligned}$$

Composition example

$$[\![\text{wait}(1) \mid\mid \text{wait}(3)]\!] \quad = \quad [\![\ \mid\mid\ ]\!] \odot ([\![\text{wait}(1)]\!] \otimes [\![\text{wait}(3)]\!])$$



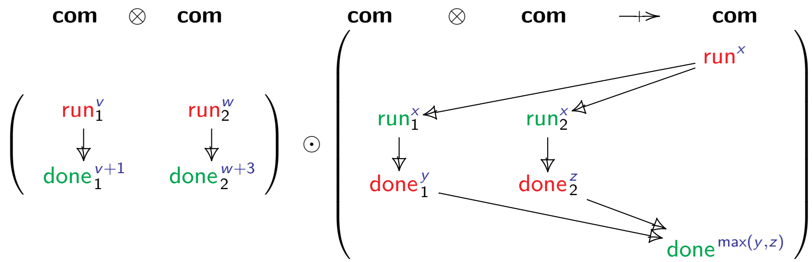
$=$



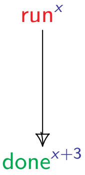
$$\begin{aligned} x &\mapsto x \\ v &\mapsto x \\ w &\mapsto x \\ y &\mapsto x+1 \\ z &\mapsto x+3 \end{aligned}$$

Composition example

$$[\![\text{wait}(1) \mid\mid \text{wait}(3)]\!] \quad = \quad [\![\ \mid\mid\ ]\!] \odot ([\![\text{wait}(1)]\!] \otimes [\![\text{wait}(3)]\!])$$



$=$



$$\begin{array}{lcl} x & \mapsto & x \\ v & \mapsto & x \\ w & \mapsto & x \\ y & \mapsto & x+1 \\ z & \mapsto & x+3 \end{array}$$

$\mathbb{R}$-CG$_{_}$

Theorem

Arenas and strategies form a symetric monoidal closed category (smcc) with products ;

In fact,

- CG is a compact closed category
  - $\rightarrow$ CG$_{_}$ is a smcc with products

$\rightarrow$ Σ-CG is a compact closed category [CSL18]

$\rightarrow$ $(\Sigma\text{-CG})/\equiv$ is a compact closed category

$\rightarrow$ $\mathbb{R}$-CG $:= (\Sigma_{\mathbb{R}}\text{-CG})/\equiv$ is a compact closed category

$\rightarrow$ $\mathbb{R}$-CG$_{_}$ is a smcc with products

Theorem

Arenas and strategies form a symetric monoidal closed category (smcc) with products ;

$$[\![\Gamma \vdash M : A]\!] : [\![\Gamma]\!] \twoheadrightarrow [\![A]\!]$$

$$\begin{array}{lll} M, N & := & \mid x \mid \lambda x.t \mid MN & \checkmark \\ & & \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{ifte} \;\; b \; M \; N & \checkmark \\ & & \mid \mathsf{skip} \mid M; N \mid M \;\; || \; N \mid \bot & \checkmark \\ & & \mid \mathsf{wait}(r) \text{ with } r \in \mathbb{R} & \checkmark \\ & & \mid \mathsf{newref}\; x \text{ in } M \mid !M \mid M := \mathsf{true} & \end{array}$$

Interpretation: Shared memory

$[\![\text{newref } r \text{ in } M]\!]$

$= \quad [\![cell]\!] \odot [\![M]\!]$

$[\![mem]\!] = \quad [\![mem_R]\!] \quad \otimes \quad [\![mem_W]\!]$

$[\![\text{cell}]\!] \quad : \quad \textbf{mem}$

Interpretation: Shared memory

$[\![\text{newref } r \text{ in } M]\!]$

$= \quad [\![cell]\!] \odot [\![M]\!]$

$[\![mem]\!] = \quad [\![mem_R]\!] \quad \otimes \quad [\![mem_W]\!]$

$[\![\text{cell}]\!] \quad : \quad \textbf{mem}$

r

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

$[\![\text{cell}]\!]$: **mem**

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

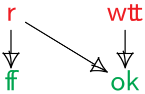
$[\![\text{cell}]\!]$: **mem**

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![\mathit{cell}]\!] \odot [\![M]\!]$$

$$[\![\mathit{mem}]\!] = [\![\mathit{mem}_R]\!] \otimes [\![\mathit{mem}_W]\!]$$

$$[\![\text{cell}]\!] : \textbf{mem}$$



Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] \quad = \quad [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = \quad [\![mem_R]\!] \quad \otimes \quad [\![mem_W]\!]$$

$$[\![\text{cell}]\!] \quad : \quad \mathbf{mem}$$

wtt

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![\mathit{cell}]\!] \odot [\![M]\!]$$

$$[\![\mathit{mem}]\!] = [\![\mathit{mem}_R]\!] \otimes [\![\mathit{mem}_W]\!]$$

$[\![\text{cell}]\!]$: **mem**

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

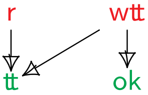
$$[\![\text{cell}]\!] \quad : \quad \textbf{mem}$$

Interpretation: Shared memory

$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$

$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$

$[\![\text{cell}]\!] : \mathbf{mem}$



Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

$$[\![\text{cell}]\!] \quad : \quad \textbf{mem}$$

r wtt

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

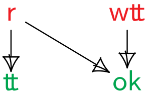
$[\![\text{cell}]\!]$: **mem**

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

$$[\![\text{cell}]\!] : \textbf{mem}$$



Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![\mathit{cell}]\!] \odot [\![M]\!]$$

$$[\![\mathit{mem}]\!] = [\![\mathit{mem}_R]\!] \otimes [\![\mathit{mem}_W]\!]$$

$$[\![\text{cell}]\!] \quad : \quad \textbf{mem}$$

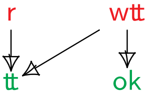
r wtt

Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

$$[\![\text{cell}]\!] : \textbf{mem}$$

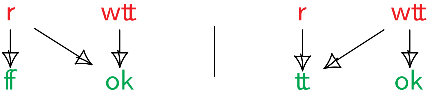


Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

$[\![\text{cell}]\!]$: **mem**

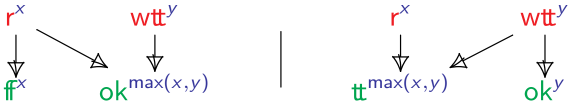


Interpretation: Shared memory

$$[\![\text{newref } r \text{ in } M]\!] = [\![cell]\!] \odot [\![M]\!]$$

$$[\![mem]\!] = [\![mem_R]\!] \otimes [\![mem_W]\!]$$

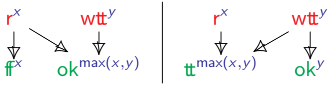
$$[\![\text{cell}]\!] : \textbf{mem}$$



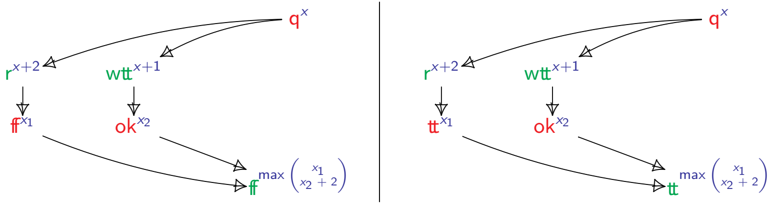
Soundness

newref r in
wait(2) !r || wait(1) r := true wait(2)

$[\![\text{cell}]\!]$: **mem**



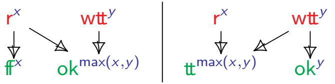
$[\![P']\!]$: **mem** $\rightarrow\!\!\!\!+\!\!\!\!\rightarrow$ **bool**



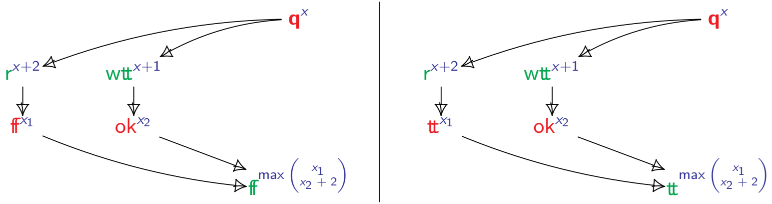
Soundness

```
newref  r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\mathrm{cell}]\!]$: **mem**



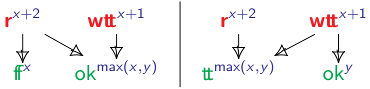
$[\![P']\!]$: **mem** $\twoheadrightarrow$ **bool**



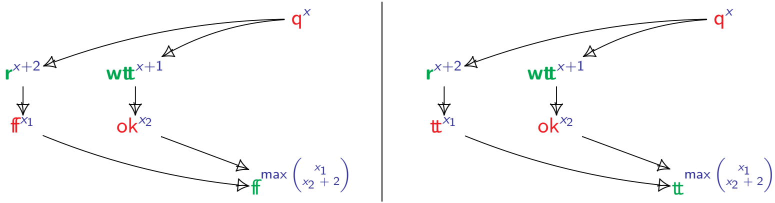
Soundness

```
newref  r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\text{cell}]\!]$: **mem**



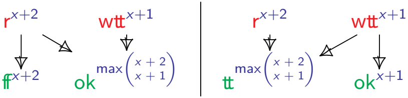
$[\![P']\!]$: **mem** $\not\rightarrow$ **bool**



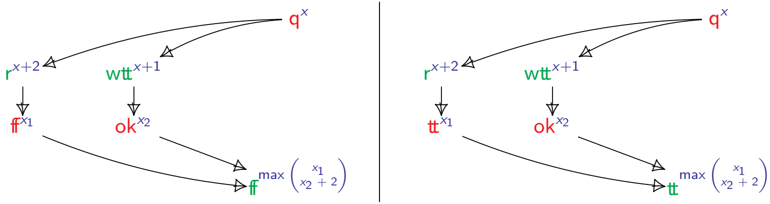
Soundness

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\text{cell}]\!]$: **mem**



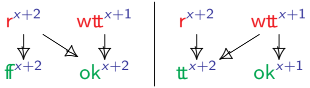
$[\![P']\!]$: **mem** $\twoheadrightarrow$ **bool**



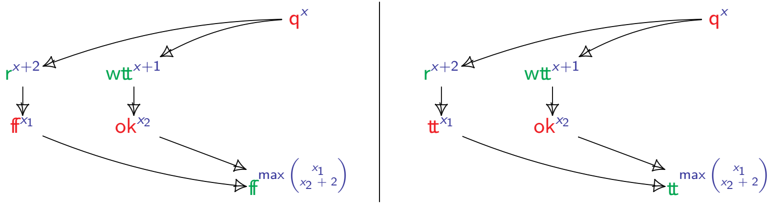
Soundness

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\text{cell}]\!]$: **mem**

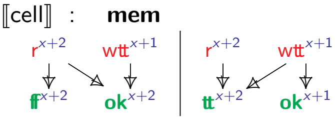


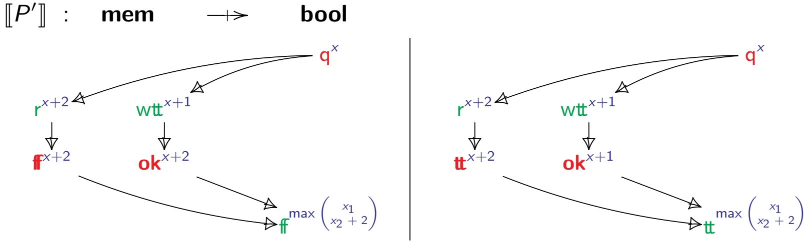
$[\![P']\!]$: **mem** $\twoheadrightarrow$ **bool**



Soundness

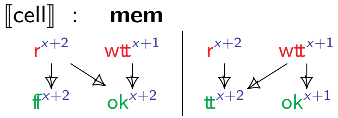
```
newref r in
    wait(2) || wait(1)
    !r      || r := true
            || wait(2)
```

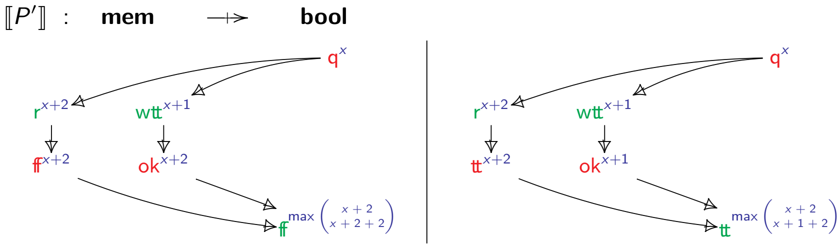




Soundness

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

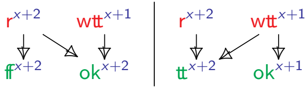




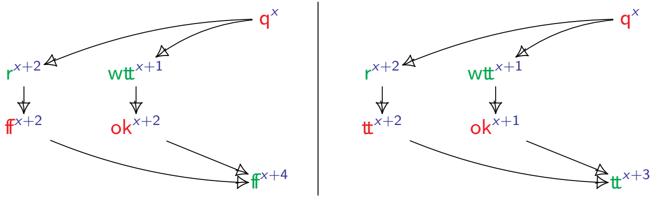
Soundness

```
newref  r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\text{cell}]\!]$: **mem**



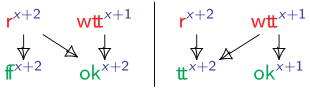
$[\![P']\!]$: **mem** $\rightarrow\!\!\!\!\!\!+$ **bool**



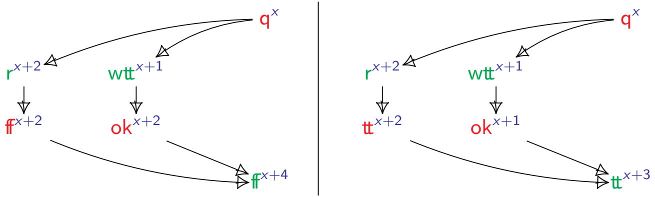
Soundness

```
newref  r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
             ||  wait(2)
```

$[\![\mathrm{cell}]\!]$: **mem**



$[\![P']\!]$: **mem** $\multimap$ **bool**



Theorem

If $M \Downarrow^{t} v$ then $q^{x} \rightharpoonup v^{x+t'} \in [\![M]\!]$ with $t' \leq t$.

What resources?

Theorem

If $M \Downarrow^{r} v$ *then* $q^{x} \rightharpoonup v^{x+r'} \in [\![M]\!]$ *with* $r' \leq r$.

$$\text{wait(r)},\ \ r \in \mathbb{R} \qquad \rightsquigarrow \qquad \text{consume(r)},\ \ r \in \mathcal{R}$$

Def. Resource bimonoid: $(\mathcal{R}, 0, ;, \|, \leq)$

- $(\mathcal{R}, 0, ;, \leq)$ ordered monoid
- $(\mathcal{R}, 0, \|, \leq)$ commutative ordered monoid
- $\|$ is idempotent, i.e. $r \parallel r = r$

	$\mathcal{R}$	;	$\|$
time	$\mathbb{R}$	$+$	max

What resources?

Theorem

If $M \Downarrow^{r} v$ *then* $q^{x} \multimap v^{x+r'} \in [\![M]\!]$ *with* $r' \leq r$.

$$\mathsf{wait}(\mathsf{r}),\ \ r \in \mathbb{R} \qquad \rightsquigarrow \qquad \mathsf{consume}(\mathsf{r}),\ \ r \in \mathcal{R}$$

Def. Resource bimonoid: $(\mathcal{R}, 0, ;, \|, \leq)$

- $(\mathcal{R}, 0, ;, \leq)$ ordered monoid
- $(\mathcal{R}, 0, \|, \leq)$ commutative ordered monoid
- $\|$ is idempotent, i.e. $r \| r = r$

	$\mathcal{R}$	;	$\|$
time	$\mathbb{R}$	$+$	max
parametric time	$\mathbb{R} \to \mathbb{R}$	$+$	max
permission	$\mathcal{P}(P)$	$\cup$	$\cup$
~~energy~~	~~$\mathbb{R}$~~	~~max~~	~~$+$~~

Resource tracking concurrent games

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true
    wait(1) || wait(2)
```

0

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true          2
    wait(1)  ||  wait(2)
```

Adequacy?

Q: What is the minimal amount of time necessary to get true?

newref r in			
wait(2)	‖	wait(1)	
!r	‖	r := true	2
wait(1)	‖	wait(2)	

Adequacy?

Q: What is the minimal amount of time necessary to get true?

newref r in			
wait(2)	‖	wait(1)	
!r	‖	r := true	2
wait(1)	‖	wait(2)	

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true        4
    wait(1) || wait(2)
```

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true        1
    wait(1) || wait(2)
```

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true        1
    wait(1) || wait(2)
```

Adequacy?

Q: What is the minimal amount of time necessary to get true?

newref r in		
wait(2)	wait(1)	
!r	r := true	3
wait(1)	wait(2)	

Adequacy?

Q: What is the minimal amount of time necessary to get true?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true        3
    wait(1) || wait(2)
```

Adequacy?

Q: What is the minimal amount of time necessary to get true?

newref r in		
wait(2)	wait(1)	
!r	r := true	4
wait(1)	wait(2)	

Adequacy?

Q: What is the minimal amount of time necessary to get true?

newref r in

wait(2)	wait(1)
!r	r := true
wait(1)	wait(2)

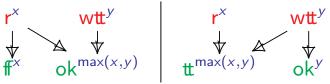
4

$P \Downarrow^4 \text{true}$

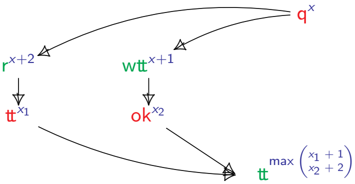
Adequacy?

```
newref r in
    wait(2) || wait(1)
    !r      || r := true
    wait(1) || wait(2)
```

$[\![\text{cell}]\!]$: **mem**



$[\![P']\!]$: **mem** $\to$ **bool**

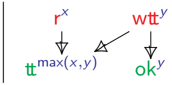


Adequacy?

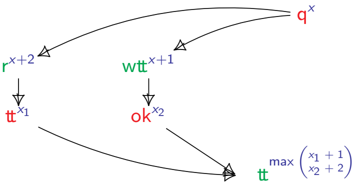
newref r in

wait(2)	wait(1)
!r	r := true
wait(1)	wait(2)

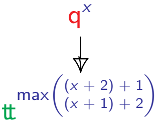
$[\![\text{cell}]\!]$: **mem**



$[\![P']\!]$: **mem** $\to$ **bool**



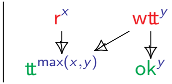
$[\![P]\!]$: **bool**



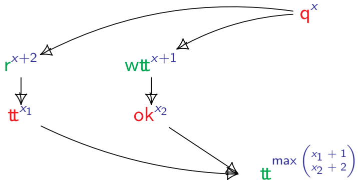
Adequacy?

```
newref r in
    wait(2)  ||  wait(1)
    !r       ||  r := true
    wait(1)  ||  wait(2)
```

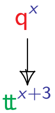
$[\![\text{cell}]\!]$: **mem**



$[\![P']\!]$: **mem** $\rightarrow$ **bool**



$[\![P]\!]$: **bool**



Is $[\![\ ]\!]$ degenerated?

Adequacy (Time)

An efficient small-step semantics:

$$\langle \mathsf{wait}(t_1 + t_2), s, x\rangle \rightarrow \langle \mathsf{wait}(t_2), s, x + t_1\rangle \qquad \ldots$$

```
newref r in
    wait(2)  ||  wait(1)
   !r        ||  r := true
    wait(1)  ||  wait(2)
```

0

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ *then* $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$$\langle \mathsf{wait}(t_1 + t_2), s, x\rangle \to \langle \mathsf{wait}(t_2), s, x + t_1\rangle \qquad \ldots$$

```
newref r in
   wait(2) || wait(1)
  !r       || r := true          0
   wait(1) || wait(2)
```

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ then $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$\langle \mathsf{wait}(t_1 + t_2), s, x \rangle \rightarrow \langle \mathsf{wait}(t_2), s, x + t_1 \rangle$ ...

newref r in

wait(1)	wait(0)
!r	r := true
wait(1)	wait(2)

1

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ *then* $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$\langle \mathsf{wait}(t_1 + t_2), s, x\rangle \to \langle \mathsf{wait}(t_2), s, x + t_1\rangle$ . . .

newref r in

wait(1)	wait(0)
!r	r := true
wait(1)	wait(2)

1

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ then $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$\langle \mathsf{wait}(t_1 + t_2), s, x \rangle \to \langle \mathsf{wait}(t_2), s, x + t_1 \rangle \qquad \ldots$

```
newref r in
     wait(1) || wait(0)
  !r         || r := true
     wait(1) || wait(2)
```

1

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ then $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$$\langle \mathsf{wait}(t_1 + t_2), s, x\rangle \to \langle \mathsf{wait}(t_2), s, x + t_1\rangle \qquad \ldots$$

```
newref r in
      wait(0) || wait(0)
    !r        || r := true
      wait(1) || wait(1)
```

2

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ *then* $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$\langle \mathsf{wait}(t_1 + t_2), s, x \rangle \to \langle \mathsf{wait}(t_2), s, x + t_1 \rangle$ . . .

```
newref r in
     wait(0) || wait(0)
    !r       || r := true        2
     wait(1) || wait(1)
```

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ *then* $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$$\langle \mathrm{wait}(t_1 + t_2), s, x\rangle \to \langle \mathrm{wait}(t_2), s, x + t_1\rangle \qquad \ldots$$

```
newref r in
      wait(0)  ||  wait(0)
     !r        ||  r := true        2
      wait(1)  ||  wait(1)
```

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ *then* $M \Downarrow^t v$.

Adequacy (Time)

An efficient small-step semantics:

$\langle \mathrm{wait}(t_1 + t_2), s, x\rangle \to \langle \mathrm{wait}(t_2), s, x + t_1\rangle$ . . .

newref r in

wait(0)	wait(0)
!r	r := true
wait(0)	wait(0)

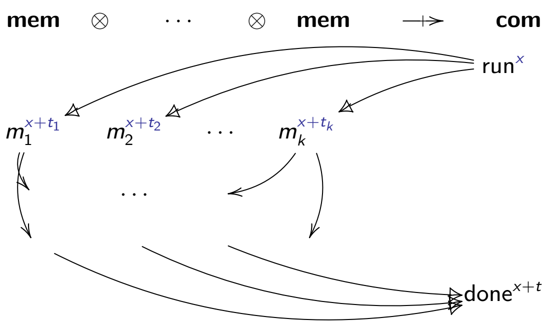
3

Theorem

If $q^x \multimap v^{x+t} \in [\![M]\!]$ then $M \Downarrow^t v$.

Proof sketch

A witness for $q^x \multimap \mathrm{done}^{x+t} \in [\![M]\!]$

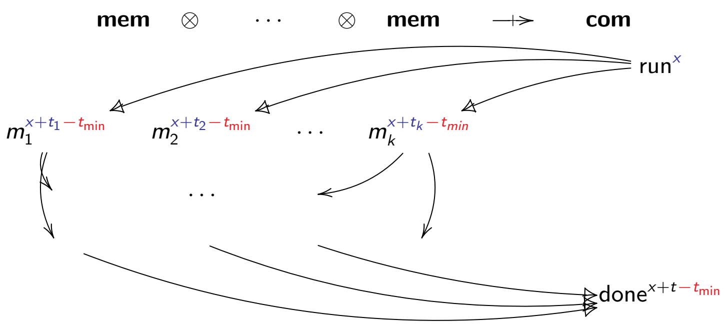


can be read as

1. If $t_i = 0$ then M can perform m_i at no cost or other memory operations.
2. Else M can reduce in high parallelism to case 1.

Proof sketch

A witness for $q^x \multimap \mathrm{done}^{x+t} \in [\![M]\!]$



can be read as

1. If $t_i = 0$ then M can perform m_i at no cost or other memory operations.
2. Else M can reduce in high parallelism to case 1.

Conclusion

- Concurrent games with annotations:
  - $\rightarrow$ A sound model for $\mathcal{R}$-IPA,
  - $\rightarrow$ Adequate for $\mathbb{R}$-IPA.

- Future work:
  - Replication;
  - Reusable resources?
  - Resources with effects?