

A Game-Theoretic Approach to Concurrent Separation Logic

Paul-André Melliès¹ Léo Stefanescu²

November 28, 2016

¹IRIF/CNRS/Université Paris Diderot

²ENS Lyon

Syntax

$$B ::= \mathbf{true} \mid \mathbf{false} \mid B \wedge B \mid B \vee B \mid E = E'$$
$$E ::= 0 \mid 1 \mid \dots \mid x \mid E + E \mid E - E \mid E * E$$
$$C ::= x := E \mid x := [y] \mid [x] := [y]$$
$$\mid \mathbf{while} \ B \ \mathbf{do} \ C \mid \mathbf{if} \ B \ \mathbf{then} \ C \ \mathbf{else} \ C$$
$$\mid \mathbf{resource} \ r \ \mathbf{do} \ C \mid \mathbf{with} \ r \ \mathbf{when} \ B \ \mathbf{do} \ C$$
$$\mid x := \mathbf{alloc}(E) \mid \mathbf{dispose}(E)$$
$$\mid C; C \mid C \parallel C$$

Predicates

$$P, Q, R, J ::= B \mid P \vee Q \mid P \wedge Q \mid \neg P \mid P \Rightarrow Q \mid \forall X.P \mid \text{Own}_p(x) \\ \mid \exists X.P \mid \mathbf{emp} \mid E_1 \xrightarrow{p} E_2 \mid P * Q$$

Semantics

$$\sigma \models E_1 = E_2 \iff \llbracket E_1 \rrbracket(\sigma) = \llbracket E_2 \rrbracket(\sigma) \wedge \text{fv}(E_1 = E_2) \subseteq \text{dom}(\sigma)$$

$$\sigma \models P \wedge Q \iff \sigma \models P \text{ et } \sigma \models Q$$

$$\sigma \models P_1 * P_2 \iff \exists \sigma_1, \sigma_2, \sigma = \sigma_1 \uplus \sigma_2 \wedge \sigma_1 \models P_1 \wedge \sigma_2 \models P_2$$

$$\sigma \models \text{Own}_p(x) \iff \exists v \in \mathbf{Val}, \sigma(x) = (v, p)$$

“Classic” Concurrent Separation Logic

Judgments

$$\Gamma \vdash \{P\} C \{Q\}$$

Contexts

$$\Gamma ::= () \mid \Gamma, r : J$$

$$\frac{\Gamma \vdash \{P_1\} C_1 \{Q_1\} \quad \Gamma \vdash \{P_2\} C_2 \{Q_2\}}{\Gamma \vdash \{P_1 * P_2\} C_1 \parallel C_2 \{Q_1 * Q_2\}} \text{PAR}$$

$$\frac{\Gamma, r : J \vdash \{P\} C \{Q\}}{\Gamma \vdash \{P * J\} \text{resource } r \text{ do } C \{Q * J\}} \text{RES}$$

$$\frac{\Gamma \vdash \{(P * J) \wedge B\} C \{Q * J\}}{\Gamma, r : J \vdash \{P\} \text{with } r \text{ when } B \text{ do } C \{Q\}} \text{WHEN}$$

Example: Producer – Consumer

$POP(n) : \text{with } r_n \text{ when } n > 0$

do $n--$; $f := f.1$

$ADD(n) : \text{with } r_n \text{ when true}$

do $n++$; $b := b.1$

$r_n : \{\text{Own}_{0.5}(f) * \text{Own}_{0.5}(b) * \text{Own}_T(n_S)$

$* \text{listseg } n \text{ } f \text{ } b\}$

$\{\text{Own}_T(tc) * \text{Own}_T(front) * \text{Own}_{0.5}(f)$

$* \text{Own}_T(n_P) * (front = f \wedge \mathbf{emp})\}$

$\{\text{Own}_T(tp) * \text{Own}_T(back) * \text{Own}_{0.5}(b)$

$* \text{Own}_T(n_V) * (back = b \wedge back \mapsto _, _)\}$

while true do

$tc := front$;

$POP(n)$;

$front := front.1$

consume($tc.0$)

dispose(tc)

while true do

$back.0 := \text{produce}()$

$tp := \text{cons}()$

$back.1 := tp$;

$ADD(n)$;

$back := tp$

Traces

The semantics of a command C is a transition system $\llbracket C \rrbracket$ of traces of the form:

$$\mathfrak{s}_1 \xrightarrow{env} \mathfrak{s}_2 \xrightarrow{m_1} \mathfrak{s}_3 \xrightarrow{env} \mathfrak{s}_4 \xrightarrow{m_2} \mathfrak{s}_5 \xrightarrow{env} \dots \xrightarrow{env} \mathfrak{s}_{2p} \xrightarrow{m_p} \mathfrak{s}_{2p+1} \xrightarrow{env} \mathfrak{s}_{2p+2}$$

where $\mathfrak{s} = (\sigma, L)$ and m_i are instructions:

- $x := E$
- $x := [y]$
- $[x] := y$
- **nop**
- $P(r)$
- $V(r)$
- $x := \text{alloc}(E)$
- $\text{dispose}(E)$

Transition System

$\mathbf{T} = (T, |T|)$ with $T \subseteq \mathbf{Traces}$ and $|T| \subseteq T$, such that T is closed by odd prefixes.

Sequential Composition

$$[[C]; [C']] = [[C]] \cup \{t \cdot t' \mid t \in [[C]], t' \in [[C']] \text{ and } \partial_1 t = \partial_0 t'\}$$

$$|[C]; [C']| = \{t \cdot t' \mid t \in [[C]], t' \in [[C']] \text{ and } \partial_1 t = \partial_0 t'\}$$

Parallel composition

$[[C \parallel C']]$ is the set of interleavings of the traces $[[C]]$ and of $[[C']]$

Toward the Soundness Theorem

- The graph of separated states
- Separation Games
- Strategies, Winning Strategies
- Soundness Theorem

The Graph of Separated States

Definition (The Graph of Separated States)

Its *nodes* are all the tuples

$$(\sigma_C, \sigma, \sigma_F) \in \mathbb{S} \times (\mathbf{LockName} \rightarrow \mathbb{S} + \{C, F\}) \times \mathbb{S}$$

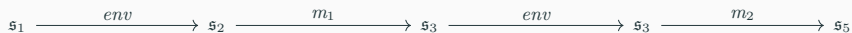
such that

$$\sigma_C * \left\{ \bigotimes_{r \in \text{dom}(\sigma)} \sigma(r) \right\} * \sigma_F$$

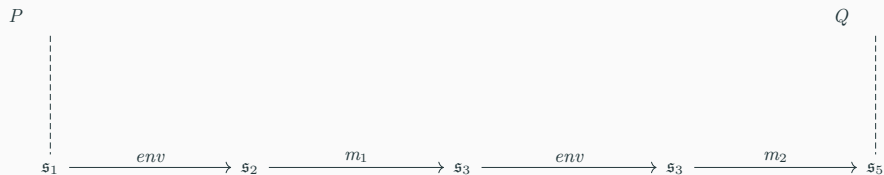
is well defined, and it has two kinds of edges:

- the Eve edges: $(\sigma_C, \sigma, \sigma_F) \xrightarrow{r:\sigma_U} (\sigma'_C, \sigma \uplus [r \mapsto \sigma_U], \sigma_F),$
- the Adam edges $(\sigma_C, \sigma, \sigma_F) \xrightarrow{r:\sigma_L} (\sigma_C * \sigma_L, \sigma', \sigma'_F).$

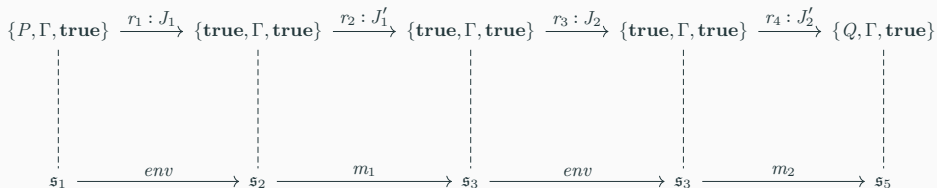
The Game Induced by $\Gamma \vdash \{P\} C \{Q\}$ and $t \in \llbracket C \rrbracket$



The Game Induced by $\Gamma \vdash \{P\} C \{Q\}$ and $t \in \llbracket C \rrbracket$

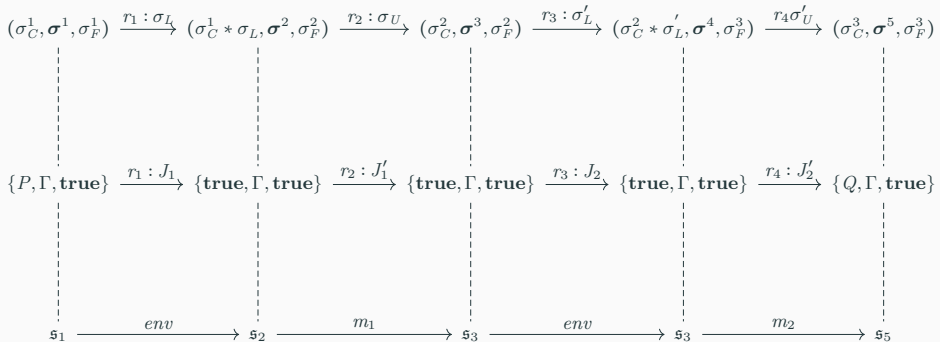


The Game Induced by $\Gamma \vdash \{P\} C \{Q\}$ and $t \in \llbracket C \rrbracket$



with $J_i = \Gamma(\text{lock}^+(m_i))$ and $J'_i = \Gamma(\text{lock}^-(m_i))$.

The Game Induced by $\Gamma \vdash \{P\} C \{Q\}$ and $t \in \llbracket C \rrbracket$



with $J_i = \Gamma(\text{lock}^+(m_i))$ and $J'_i = \Gamma(\text{lock}^-(m_i))$.

Soundness Theorem

Theorem (Soundness)

Suppose that $\Gamma \vdash \{P\} C \{Q\}$ and let $t \in \llbracket C \rrbracket$.

Then there exists a winning strategy for the induced game.

Corollary

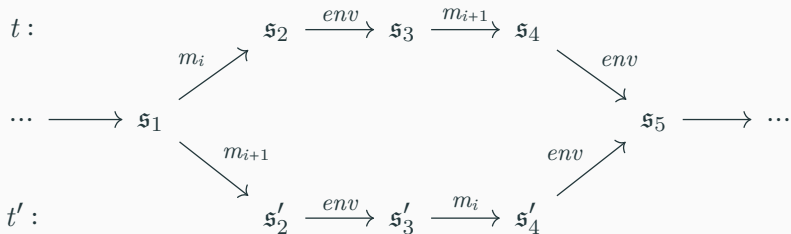
Suppose $\emptyset \vdash \{P\} C \{Q\}$, let $t \in \llbracket C \rrbracket$ a trace where the Environment is *silent*.

$$s_1 \xrightarrow{env} s_1 \xrightarrow{m_1} s_2 \xrightarrow{env} s_2 \xrightarrow{m_2} \dots \xrightarrow{m_{k-1}} s_k \xrightarrow{env} s_k$$

Then $s_1 \models P * \mathbf{true} \Rightarrow s_k \models Q * \mathbf{true}$.

Data Races & True Concurrency

- We can add a partial order \leq on instructions in traces (*program-order* + *synchronizes-with*).
- If $m_1 \not\#_t m_2$ in a trace t , we can define a commuting tile as:



- 2 problems:
 - Environnement can add *synchronizes-with* dependencies;
 - we must constraint on the Environnement transitions in t' .

Resource Tracking

Adam's edges become

$$(\sigma_C, \sigma, \sigma_F) \xrightarrow[\rho]{r:\sigma_L} (\sigma_C * \sigma_L, \sigma', \sigma'_F)$$

where $\rho \subseteq \mathbf{LockName}$ contains the locks touched by Env:

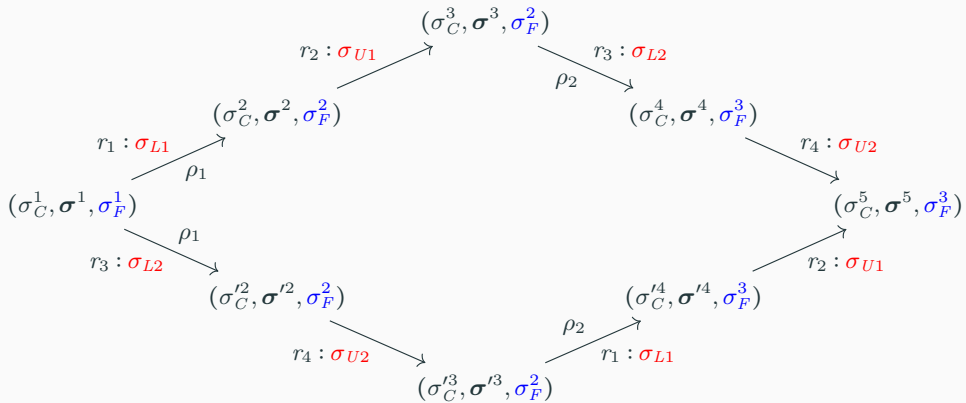
$$\forall r \in \mathbf{LockName} \setminus \rho, \sigma'(r) = \sigma(r).$$

Definition (Concurrent Instructions)

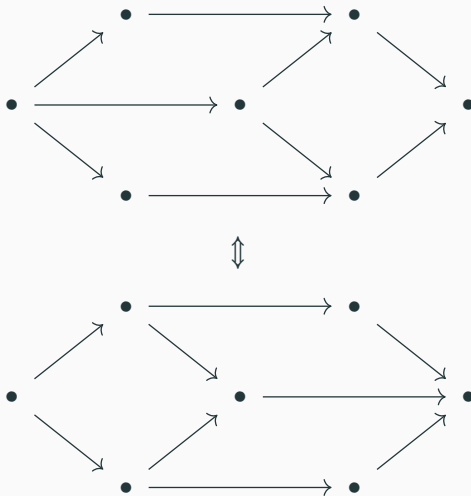
$$\begin{array}{ccccccc} p: & n_1 & \xrightarrow{r_1:\sigma_{U1}} & n_2 & \xrightarrow[\rho]{r_2:\sigma_L} & n_3 & \xrightarrow{r_3:\sigma_{U2}} & n_4 \\ & \vdots & & \vdots & & \vdots & & \vdots \\ t: & \mathfrak{s}_1 & \xrightarrow{m_1} & \mathfrak{s}_2 & \xrightarrow{env} & \mathfrak{s}_3 & \xrightarrow{m_2} & \mathfrak{s}_4 \end{array}$$

$$m_1 \#_p m_2 \quad \Leftrightarrow \quad m_1 \#_t m_2 \quad \wedge \quad (\rho \cap (\text{lock}(m_1) \cup \text{lock}(m_2))) = \emptyset$$

Commuting Tile



From Local to Global: the Cube



Merci !