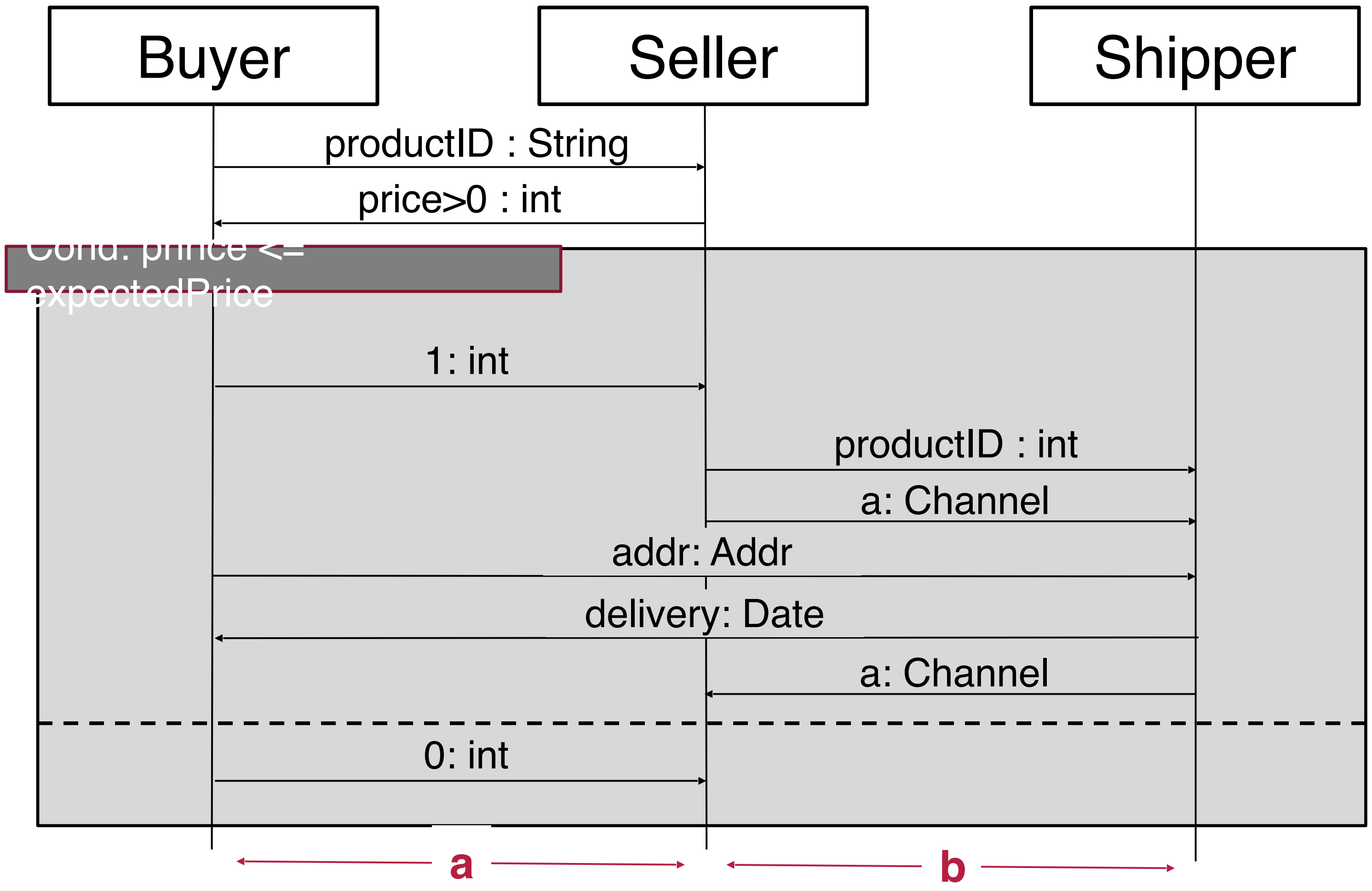
AUTOMATED VERIFICATION FOR RACE-FREE CHANNELS WITH IMPLICIT AND EXPLICIT SYNCHRONIZATION

Andreea Costea, Wei-Ngan Chin, Florin Craciun, Shengchao Qin

March 2017

CONTEXT – COMMUNICATION PROTOCOLS



Possible problems:

- deadlock
- unexpected communication
- transmission race

STATE OF THE ART

Behavioural types:

- Generic types[1]: types and type environments as abstract processes, and then guarantee deadlock-freedom of process by checking the corresponding type environment.
- Behavioral separation[2]: extends separation logics and substructural types to higher order imperative concurrent programs in order to discipline interference
- Session types[3,4]: Global and local types to describe communication and ensure deadlock freedom and race-freedom in the context of message passing

[1] IGARASHI , A. and KOBAYASHI , N., "A Generic Type System for the Pi-Calculus," Theoretical Computer Science, vol. 311, no. 1, pp. 121 – 163, 2004.

[2] CAIRES , L. and SECO , J. C., "The Type Discipline of Behavioral Separation," in POPL 2013.

[3] HONDA , K., VASCONCELOS , V. T., and KUBO , M., "Language primitives and type discipline for structured communication-based programming," in ESOP '98.

[4] HONDA , K., YOSHIDA , N., and CARBONE , M., "Multiparty Asynchronous Session Types," POPL 2008.

STATE OF THE ART (CONT.)

Logics with channel primitives:

- CSL for copyless message passing: an extension of separation for bidirectional communication between two players using global contracts
- CSL for pipelined parallelization[6]: an extension of separation logic which supports multiple players communicating through a single shared channel
- Chalice[8] with support for message passing[7]: modular verification to prevent deadlocks of programs which mix message passing and locking.

[5] V ILLARD , J., L OZES , É., and C ALCAGNO , C., "Proving copyless message passing," in APLAS 2009 , pp. 194–209, Springer.

[6] BELL , C. J., APPEL , A. W., and WALKER , D., "Concurrent Separation Logic for Pipelined Parallelization," in SAS 2010, pp. 151–166, Springer.

[7] LEINO , K. R. M., MÜLLER , P., and SMANS , J., "Deadlock-Free Channels and Locks," in ESOP 2010, pp. 407–426, Springer.

[8] LEINO , K. R. M. and MÜLLER , P., "A Basis for Verifying Multi-Threaded Programs," in ESOP 2009 pp. 378–393, Springer.

AUTOMATED VERIFICATION FOR RACE-FREE CHANNELS WITH IMPLICIT AND EXPLICIT SYNCHRONIZATION

AUTOMATED VERIFICATION FOR RACE-FREE CHANNELS WITH IMPLICIT AND EXPLICIT SYNCHRONIZATION

EXAMPLE 1

A→ B: c("Yes") ; A→ C: c("No")

A	B	C
…	…	…
`send(c, "Yes");`	`a = receive(c);`	`a = receive(c);`
`send(c, "No");`		

Communication assumptions:
shared FIFO *message queues*
unbounded queue
asynch communication

Who reads "Yes"?
Race on reading from c!
Current approaches declare this protocol as UNSAFE

AUTOMATED VERIFICATION FOR RACE-FREE CHANNELS WITH IMPLICIT AND EXPLICIT SYNCHRONIZATION

EXAMPLE 1

A→ B: c(“Yes”) ; A →C: c(“No”)

A	B	C
…	…	…
`send(c, “Yes”);`	`a = receive(c);`	
`send(c, “No”);`	`notifyAll(w);`	`wait(w);`
		`a = receive(c);`

Communication assumptions:
shared FIFO *message queues*
unbounded queue
asynch communication

Introduce a proof obligation on event ordering to prove that
B ***happens-before*** C

AUTOMATED VERIFICATION FOR RACE-FREE CHANNELS WITH IMPLICIT AND EXPLICIT SYNCHRONIZATION

EXAMPLE 2

A → B: c(“Yes”) ; C → B: c(“No”)

Race on writing to c!

Introduce a proof obligation on event ordering to prove that

A ***happens-before*** C

GOAL

$S_1 \quad R_1\!: c(\ldots)\ ;\ \ldots;\ S_2 \quad R_2\!: c(\ldots)$

To ensure race-freedom on c, prove that:

S_1 happens-before S_2

and

R_1 happens-before R_2

MERCURIUS: A LOGIC FOR PROTOCOL SPECIFICATION

$$
\begin{array}{llcl}
\textit{Single transmission} & \mathtt{T} & ::= & \mathtt{S} \xrightarrow{i} \mathtt{R} : \mathtt{c}\langle \mathtt{v} \cdot \Delta \rangle \\
\textit{Global protocol} & \mathtt{G} & ::= & \mathtt{T} \\
\textit{Concurrency} & & | & \mathtt{G} * \mathtt{G} \\
\textit{Choice} & & | & \mathtt{G} \vee \mathtt{G} \\
\textit{Sequencing} & & | & \mathtt{G} \,;\, \mathtt{G} \\
\textit{Guard} & & | & \ominus(\Psi) \\
\textit{Assumption} & & | & \oplus(\Psi) \\
\textit{Inaction} & & | & \mathtt{emp}
\end{array}
$$

$$
\textit{(Parties)}\ \mathtt{P}, \mathtt{S}, \mathtt{R} \in \mathcal{R}\mathtt{ole} \quad \textit{(Channels)}\ \mathtt{c} \in \mathcal{C}\mathtt{han} \quad \textit{(Messages)}\ \mathtt{v} \cdot \Delta \quad \textit{(Labels)}\ i \in \mathtt{Nat}
$$

WELL-FORMEDNESS (*)

[**Well-Formed Concurrency**] A protocol specification, $G_1 * G_2$, is said to be well-formed with respect to $*$ if and only if $\forall \mathsf{c} \in G_1 \implies \mathsf{c} \notin G_2$, and vice versa.

WELL-FORMEDNESS (∨)

(a) *(same first channel)* $\forall \mathtt{c}_1 \in i_{\mathtt{k}}, \mathtt{c}_2 \in \mathtt{l}_{\mathtt{j}} \Rightarrow \mathtt{c}_1{=}\mathtt{c}_2$;

(b) *(same first sender* $\mathtt{S}$*)* $\forall \mathtt{S}_1 \in i_{\mathtt{k}}, \mathtt{S}_2 \in \mathtt{l}_{\mathtt{j}} \Rightarrow \mathtt{S}_1{=}\mathtt{S}_2 \wedge \mathtt{S}{=}\mathtt{S}_1$;

(c) *(same first receiver* $\mathtt{R}$*)* $\forall \mathtt{R}_1 \in i_{\mathtt{k}}, \mathtt{R}_2 \in \mathtt{l}_{\mathtt{j}} \Rightarrow \mathtt{R}_1{=}\mathtt{R}_2 \wedge \mathtt{R}{=}\mathtt{R}_1$;

(d) *(mutually exclusive "first" messages)*

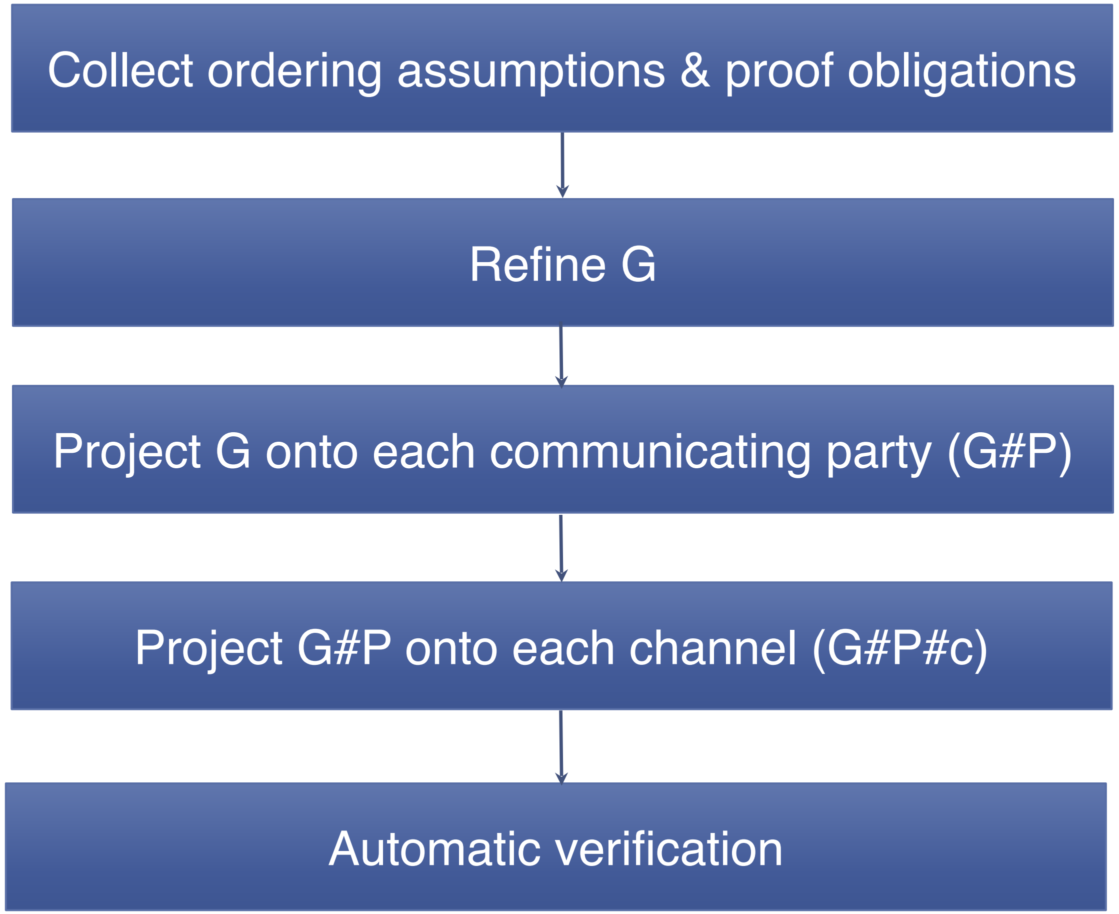
$$\forall \mathtt{j}, \mathtt{k} {\in} \{i_1, .., i_{\mathtt{n}}, \mathtt{l}_1, .., \mathtt{l}_{\mathtt{m}}\} \Rightarrow \mathtt{UNSAT}(\Delta_{\mathtt{j}} \wedge \Delta_{\mathtt{k}}) \vee \mathtt{j}{=}\mathtt{k};$$

(e) *(same roles)* $\forall \mathtt{P} \in \mathtt{G}_1 \vee \mathtt{G}_2 \Rightarrow \mathtt{P}{=}\mathtt{S} \vee \mathtt{P}{=}\mathtt{R}$, *with peers* $\mathtt{S}$ *and* $\mathtt{R}$ *the roles referenced by conditions (b) and (c), respectively;*

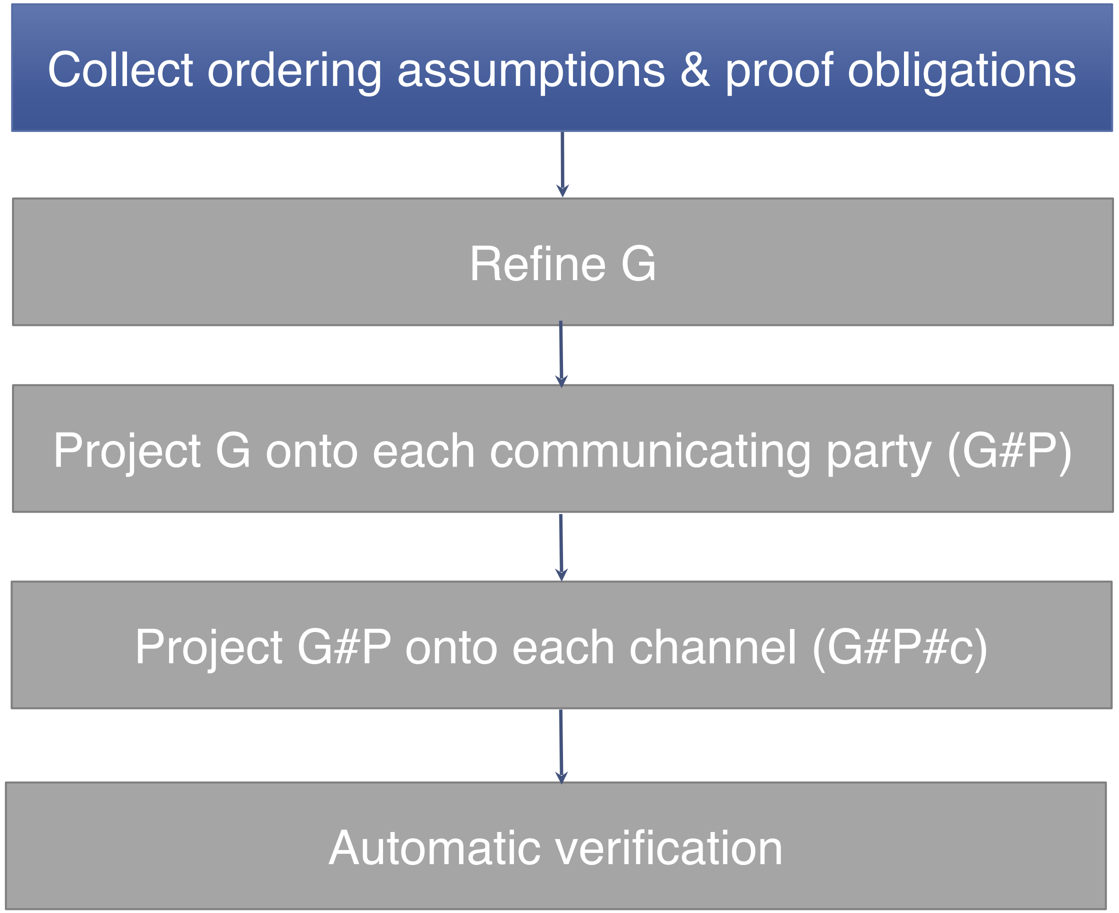
(f) *(recursive well-formedness)* $\mathtt{G}_1$ *and* $\mathtt{G}_2$ *are well-formed with respect to* $\vee$.

OVERVIEW OF OUR APPROACH

Given G:



OVERVIEW OF OUR APPROACH



ORDERING ASSUMPTIONS

Local events | Global orderings

Communicates-before for

$S \xrightarrow{i} R : c\langle v \cdot \Delta \rangle$

$$\oplus(S \xrightarrow{i} R) \Rightarrow \oplus(S^{(i)}) \wedge \oplus(R^{(i)}) \wedge \oplus(S^{(i)} \prec_{CB} R^{(i)})$$

Happens-before for any two adjacent events on the same party

$P \xrightarrow{i_1} R_1 : c_1\langle v \cdot \Delta_1 \rangle; \ldots; P \xrightarrow{i_2} R_2 : c_2\langle v \cdot \Delta_2 \rangle$

$S_1 \xrightarrow{i_1} P : c_1\langle v \cdot \Delta_1 \rangle; \ldots; S_2 \xrightarrow{i_2} P : c_2\langle v \cdot \Delta_2 \rangle$

$P \xrightarrow{i_1} R_1 : c_1\langle v \cdot \Delta_1 \rangle; \ldots; S_2 \xrightarrow{i_2} P : c_2\langle v \cdot \Delta_2 \rangle$

$S_1 \xrightarrow{i_1} P : c_1\langle v \cdot \Delta_1 \rangle; \ldots; P \xrightarrow{i_2} R_2 : c_2\langle v \cdot \Delta_2 \rangle$

$$\oplus(P^{(i_1)} \prec_{HB} P^{(i_2)})$$

RACE-FREE ASSERTIONS

$$\mathrm{S}_1 \xrightarrow{i_1} \mathrm{R}_1 : \mathrm{c}\langle \mathrm{v}\cdot\Delta_1\rangle; \ldots; \mathrm{S}_2 \xrightarrow{i_2} \mathrm{R}_2 : \mathrm{c}\langle \mathrm{v}\cdot\Delta_2\rangle$$

Proof-obligation to check race-freedom of c:

$$\ominus\left(\mathrm{S}_1^{(i_1)} \prec_{\mathrm{HB}} \mathrm{S}_2^{(i_2)} \wedge \mathrm{R}_1^{(i_1)} \prec_{\mathrm{HB}} \mathrm{R}_2^{(i_2)}\right)$$

ORDERINGS CONSTRAINT SYSTEM

$$
\begin{array}{lll}
\textit{Send/Recv Event} & \mathtt{E} & ::= \mathtt{P}^{(i)} \\
\textit{Ordering Constraints} & \vartheta & ::= \mathtt{E} \prec_{\mathtt{CB}} \mathtt{E} \mid \mathtt{E} \prec_{\mathtt{HB}} \mathtt{E} \\
\textit{Race-Free Assertions} & \Psi & ::= \mathtt{E} \mid \mathtt{not}(\mathtt{E}) \mid \vartheta \mid \Psi \wedge \Psi \mid \Psi \vee \Psi \mid \mathtt{E} \Rightarrow \Psi
\end{array}
$$

$$
\begin{array}{ll}
\Pi \models \mathtt{P}^{(i)} & \text{iff } \mathtt{P}^{(i)} \in \Pi \\
\Pi \models \mathtt{not}(\mathtt{P}^{(i)}) & \text{iff } \mathtt{P}^{(i)} \notin \Pi \\
\Pi \models \mathtt{P}_1^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_2^{(i_2)} & \text{iff } \Big(\bigwedge_{\Psi_j \in \Pi} \Psi_j\Big) \Rightarrow^* \mathtt{P}_1^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_2^{(i_2)} \\
\Pi \models \Psi_1 \wedge \Psi_2 & \text{iff } \Pi \models \Psi_1 \text{ and } \Pi \models \Psi_2 \\
\Pi \models \Psi_1 \vee \Psi_2 & \text{iff } \Pi \models \Psi_1 \text{ or } \Pi \models \Psi_2 \\
\Pi \models \mathtt{E} \Rightarrow \Psi & \text{iff } \Pi \models \mathtt{E} \Rightarrow \Pi \models \Psi
\end{array}
$$

Constraint propagation lemmas:

$$
\mathtt{P}_1^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_2^{(i_2)} \wedge \mathtt{P}_2^{(i_2)} \prec_{\mathtt{HB}} \mathtt{P}_3^{(i_3)} \Rightarrow \mathtt{P}_1^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_3^{(i_3)} \quad \text{(HB-HB)}
$$

$$
\mathtt{P}_1^{(i_1)} \prec_{\mathtt{CB}} \mathtt{P}_2^{(i_1)} \wedge \mathtt{P}_2^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_3^{(i_2)} \Rightarrow \mathtt{P}_1^{(i_1)} \prec_{\mathtt{HB}} \mathtt{P}_3^{(i_2)} \quad \text{(CB-HB)}
$$

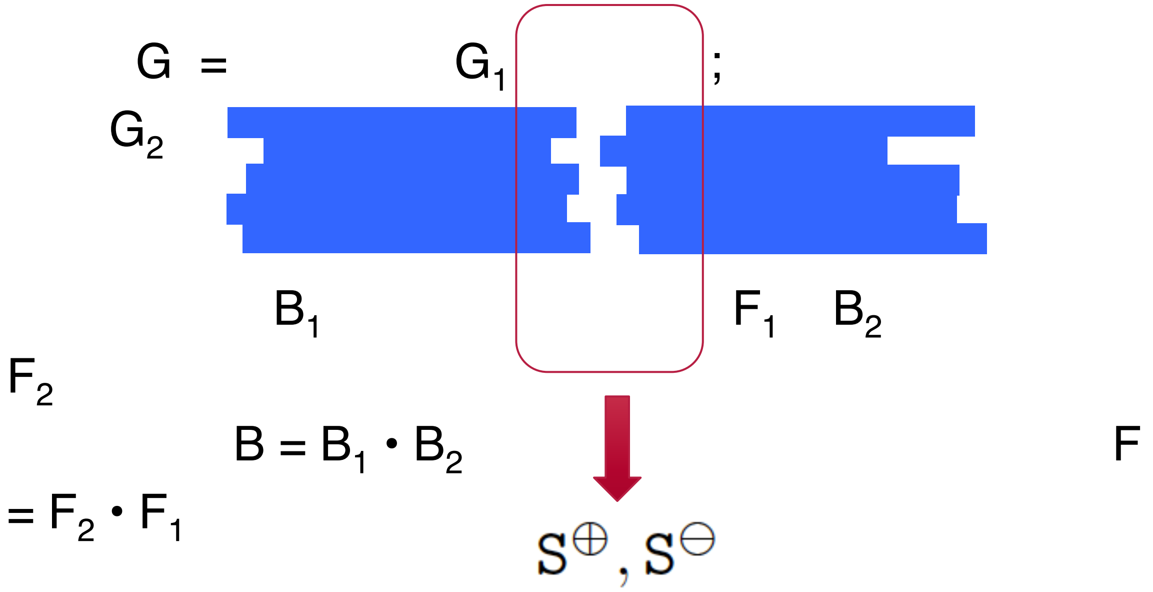
COLLECTION – BUILDING AND MERGING SUMMARIES

Summary := B^{border} x F^{border}

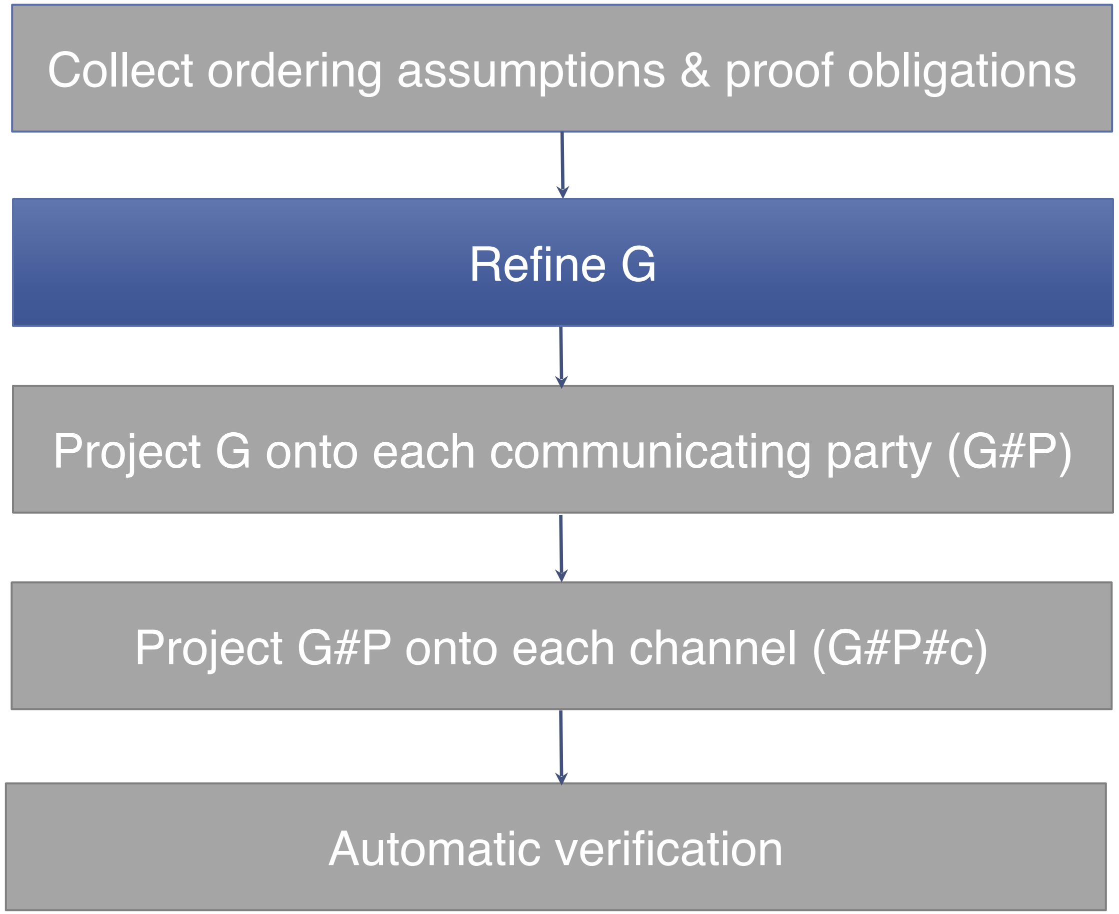
Border := M^{events} x M^{trans}

M^{events} := Role $\longrightarrow$ Events

M^{trans} := Chan $\longrightarrow$ Trans



OVERVIEW OF OUR APPROACH



EXAMPLE 3

$$(\mathtt{A} \xrightarrow{1} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_1 \rangle) \,;\, (\mathtt{A} \xrightarrow{2} \mathtt{B} : \mathtt{c}_2\langle \mathtt{t}_2 \rangle) \,;\, (\mathtt{B} \xrightarrow{3} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_3 \rangle)$$

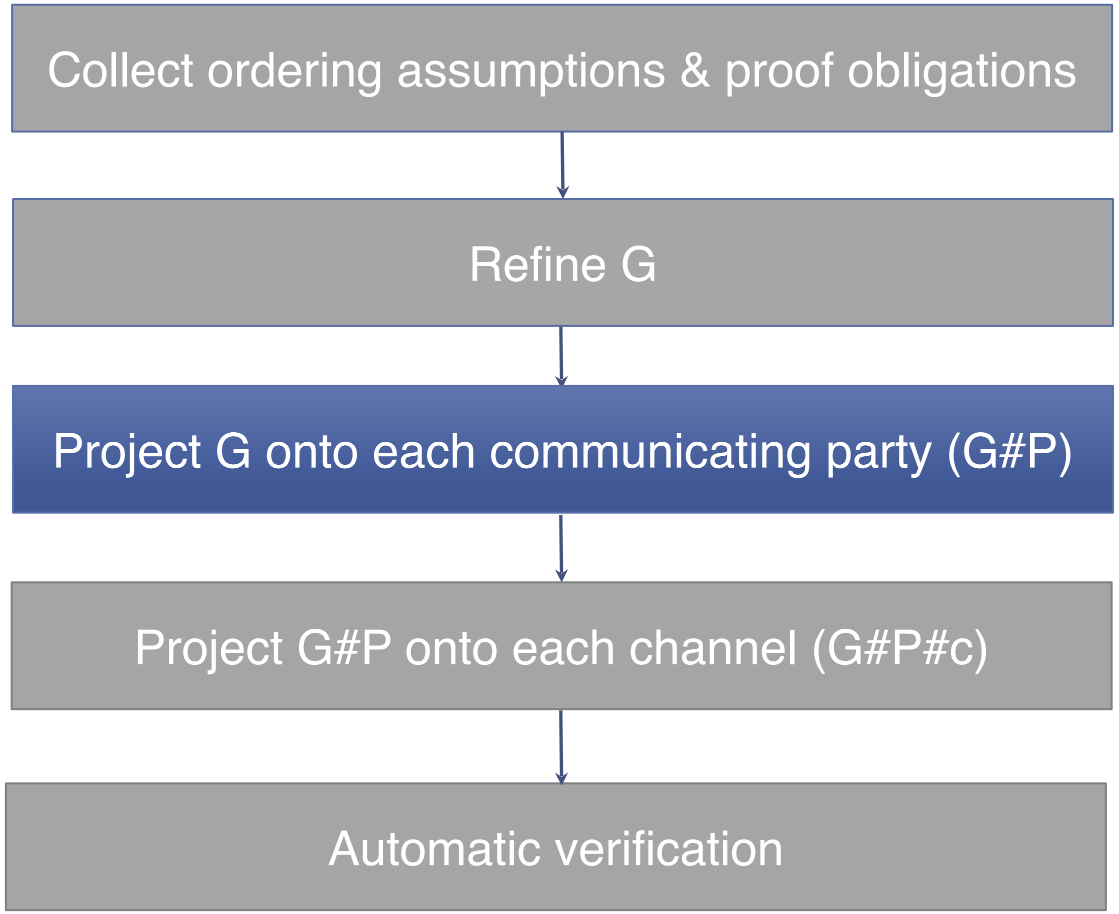
Spec

$\mathtt{A} \xrightarrow{1} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_1 \rangle;$
$\mathtt{A} \xrightarrow{2} \mathtt{B} : \mathtt{c}_2\langle \mathtt{t}_2 \rangle;$
$\mathtt{B} \xrightarrow{3} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_3 \rangle$

(i) Refinement

$\mathtt{A} \xrightarrow{1} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_1 \rangle; \oplus(\mathtt{A}^{(1)}); \oplus(\mathtt{C}^{(1)}); \oplus(\mathtt{A}^{(1)} \prec_{\mathtt{CB}} \mathtt{C}^{(1)});$
$\mathtt{A} \xrightarrow{2} \mathtt{B} : \mathtt{c}_2\langle \mathtt{t}_2 \rangle; \oplus(\mathtt{A}^{(2)}); \oplus(\mathtt{B}^{(2)}); \oplus(\mathtt{A}^{(1)} \prec_{\mathtt{HB}} \mathtt{A}^{(2)}); \oplus(\mathtt{A}^{(2)} \prec_{\mathtt{CB}} \mathtt{B}^{(2)});$
$\mathtt{B} \xrightarrow{3} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_3 \rangle; \oplus(\mathtt{B}^{(3)}); \oplus(\mathtt{C}^{(3)}); \oplus(\mathtt{B}^{(2)} \prec_{\mathtt{HB}} \mathtt{B}^{(3)}); \oplus(\mathtt{C}^{(1)} \prec_{\mathtt{HB}} \mathtt{C}^{(3)}) \oplus(\mathtt{B}^{(3)} \prec_{\mathtt{HB}} \mathtt{C}^{(3)}); \ominus(1 \prec_{\mathtt{HB}} 3)$

OVERVIEW OF OUR APPROACH



GLOBAL SPEC $\longrightarrow$ PER PARTY (LANGUAGE)

Local protocol	$\mathtt{L^p} ::=$
Send/Receive	$\mathtt{c!v} \cdot \Delta \mid \mathtt{c?v} \cdot \Delta$
HO variable	$\mid \mathtt{V}$
Concurrency	$\mid \mathtt{L^p} * \mathtt{L^p}$
Choice	$\mid \mathtt{L^p} \vee \mathtt{L^p}$
Sequence	$\mid \mathtt{L^p};\mathtt{L^p}$
Guard/Assumption	$\mid \ominus(\Delta) \mid \oplus(\Delta)$

GLOBAL SPEC $\longrightarrow$ PER PARTY (PROJECTION RULES)

$$(\mathrm{P}_1 \xrightarrow{i} \mathrm{P}_2 : c\langle\Delta\rangle)\lfloor_{\mathrm{P}} := \begin{cases} c!v \cdot \Delta & \text{if } \mathrm{P}{=}\mathrm{P}_1 \\ c?v \cdot \Delta & \text{if } \mathrm{P}{=}\mathrm{P}_2 \\ \mathtt{emp} & \text{otherwise} \end{cases}$$

$$(\mathrm{G}_1 {*} \mathrm{G}_2)\lfloor_{\mathrm{P}} := (\mathrm{G}_1)\lfloor_{\mathrm{P}} * (\mathrm{G}_2)\lfloor_{\mathrm{P}}$$

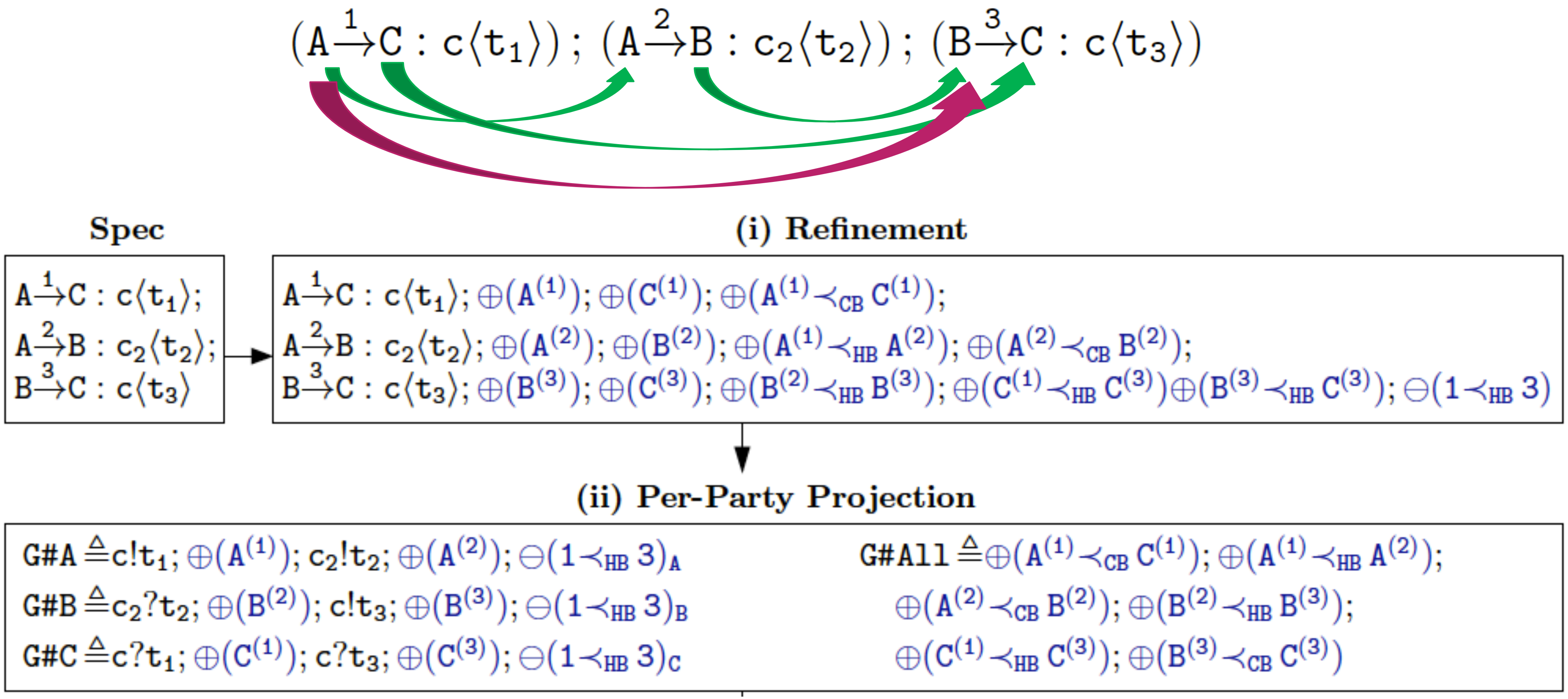
$$(\mathrm{G}_1 {\vee} \mathrm{G}_2)\lfloor_{\mathrm{P}} := (\mathrm{G}_1)\lfloor_{\mathrm{P}} \vee (\mathrm{G}_2)\lfloor_{\mathrm{P}}$$

$$(\mathrm{G}_1 ; \mathrm{G}_2)\lfloor_{\mathrm{P}} := (\mathrm{G}_1)\lfloor_{\mathrm{P}} \, ; \, (\mathrm{G}_2)\lfloor_{\mathrm{P}}$$

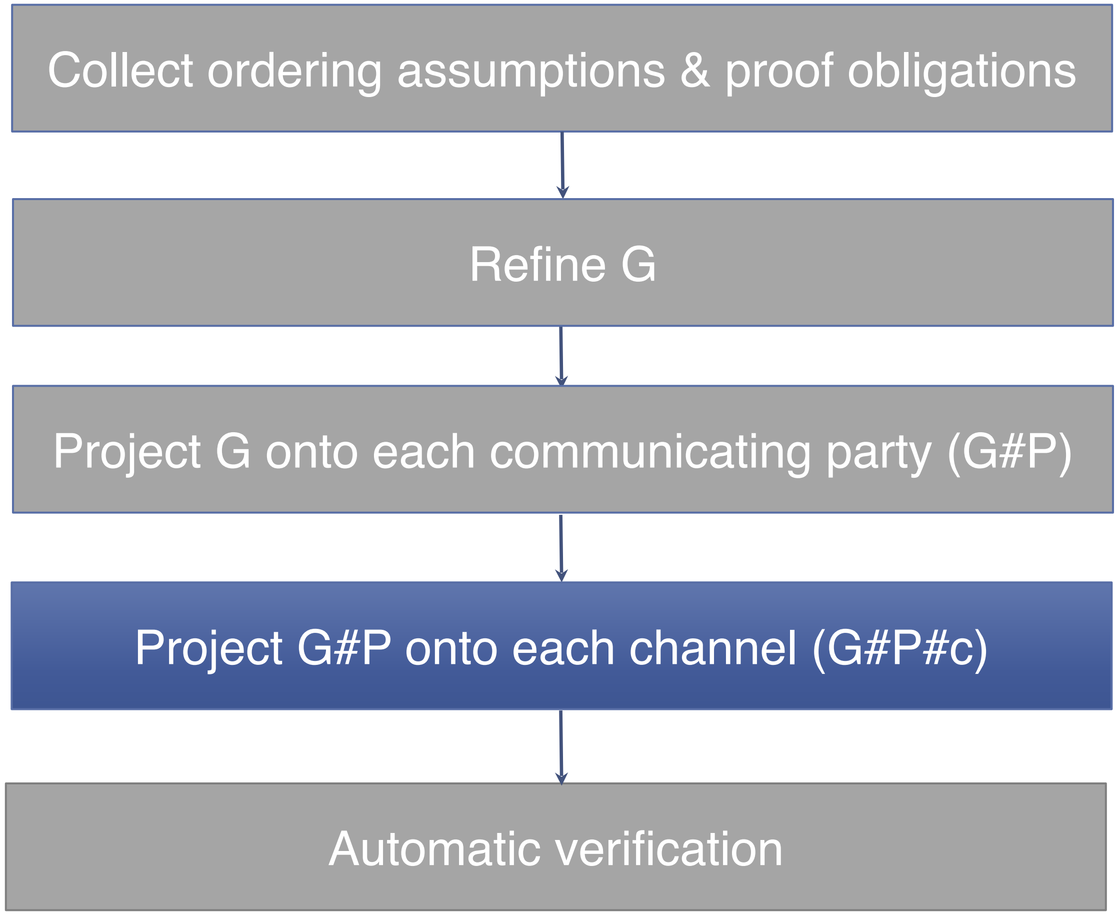
$$(\oplus(\mathrm{P}_1^{(i)}))\lfloor_{\mathrm{P}} := \begin{cases} \oplus(\mathrm{P}^{(i)}) & \text{if } \mathrm{P}{=}\mathrm{P}_1 \\ \mathtt{emp} & \text{otherwise} \end{cases}$$

$$(\ominus(\mathrm{P}_1^{(i_1)} \prec_{\mathrm{HB}} \mathrm{P}_2^{(i_2)}))\lfloor_{\mathrm{P}} := \begin{cases} \ominus(\mathrm{P}_1^{(i_1)} \prec_{\mathrm{HB}} \mathrm{P}_2^{(i_2)}) & \text{if } \mathrm{P}{=}\mathrm{P}_2 \\ \oplus(\mathrm{P}_1^{(i_1)} \prec_{\mathrm{HB}} \mathrm{P}_2^{(i_2)}) & \text{otherwise} \end{cases}$$

EXAMPLE 3: PER PARTY SPEC



OVERVIEW OF OUR APPROACH



PER PARTY $\longrightarrow$ PER CHANNEL (LANGUAGE)

Local protocol	$\mathtt{L} ::=$
Send/Receive	$!\mathtt{v} \cdot \Delta \mid ?\mathtt{v} \cdot \Delta$
HO variable	$\mid \mathtt{V}$
Choice	$\mid \mathtt{L} \vee \mathtt{L}$
Sequence	$\mid \mathtt{L};\mathtt{L}$
Guard/Assumption	$\mid \ominus(\Delta) \mid \oplus(\Delta)$

PER PARTY $\longrightarrow$ PER CHANNEL (PROJECTION RULES)

$$(c_1!v\cdot\Delta)\lfloor_c := \begin{cases} !v\cdot\Delta \text{ if } c{=}c_1 \\ \text{emp} \quad \text{otherwise}\end{cases} \qquad (L_1^p{*}L_2^p)\lfloor_c := \begin{cases} (L_j^p)\lfloor_c \text{ if } c{\in}L_j,\ j{=}1 \text{ or } 2 \\ \text{emp} \quad \text{otherwise}\end{cases}$$

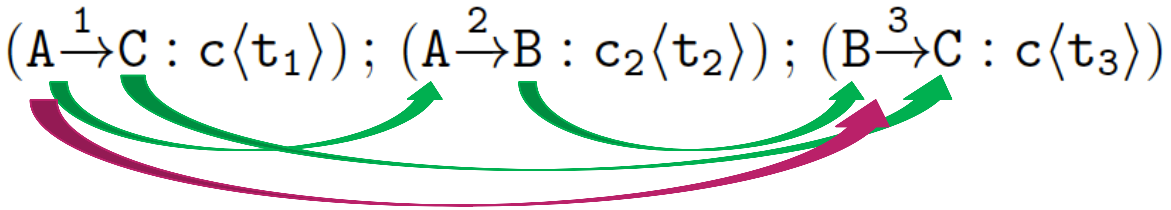
$$(c_1?v\cdot\Delta)\lfloor_c := \begin{cases} ?v\cdot\Delta \text{ if } c{=}c_1 \\ \text{emp} \quad \text{otherwise}\end{cases} \qquad \begin{array}{lcl} (L_1^p{\vee}L_2^p)\lfloor_c & := & (L_1^p)\lfloor_c \vee (L_2^p)\lfloor_c \\ (L_1^p;L_2^p)\lfloor_c & := & (L_1^p)\lfloor_c\,;\,(L_2^p)\lfloor_c \end{array}$$

$$(\oplus(P^{(i)}))\lfloor_c := \begin{cases} \oplus(P^{(i)}) \text{ if } c{\in}i \\ \ominus(P^{(i)}) \text{ otherwise}\end{cases}$$

$$(\ominus(P_1^{(i_1)} \prec_{HB} P_2^{(i_2)}))\lfloor_c := \begin{cases} \ominus(P_1^{(i_1)} \prec_{HB} P_2^{(i_2)}) \text{ if } c{\in}i_2 \\ \text{emp} \qquad \text{otherwise}\end{cases}$$

$$(\oplus(P_1^{(i_1)} \prec_{HB} P_2^{(i_2)}))\lfloor_c := \begin{cases} \oplus(P_1^{(i_1)} \prec_{HB} P_2^{(i_2)}) \text{ if } c{\in}i_2 \\ \text{emp} \qquad \text{otherwise}\end{cases}$$

EXAMPLE 3: PER CHANNEL SPEC



Spec

$A \xrightarrow{1} C : c\langle t_1 \rangle;$
$A \xrightarrow{2} B : c_2\langle t_2 \rangle;$
$B \xrightarrow{3} C : c\langle t_3 \rangle$

(i) Refinement

$A \xrightarrow{1} C : c\langle t_1 \rangle; \oplus(A^{(1)}); \oplus(C^{(1)}); \oplus(A^{(1)} \prec_{CB} C^{(1)});$
$A \xrightarrow{2} B : c_2\langle t_2 \rangle; \oplus(A^{(2)}); \oplus(B^{(2)}); \oplus(A^{(1)} \prec_{HB} A^{(2)}); \oplus(A^{(2)} \prec_{CB} B^{(2)});$
$B \xrightarrow{3} C : c\langle t_3 \rangle; \oplus(B^{(3)}); \oplus(C^{(3)}); \oplus(B^{(2)} \prec_{HB} B^{(3)}); \oplus(C^{(1)} \prec_{HB} C^{(3)}) \oplus(B^{(3)} \prec_{HB} C^{(3)}); \ominus(1 \prec_{HB} 3)$

(ii) Per-Party Projection

$G\#A \triangleq c!t_1; \oplus(A^{(1)}); c_2!t_2; \oplus(A^{(2)}); \ominus(1 \prec_{HB} 3)_A$
$G\#B \triangleq c_2?t_2; \oplus(B^{(2)}); c!t_3; \oplus(B^{(3)}); \ominus(1 \prec_{HB} 3)_B$
$G\#C \triangleq c?t_1; \oplus(C^{(1)}); c?t_3; \oplus(C^{(3)}); \ominus(1 \prec_{HB} 3)_C$

$G\#All \triangleq \oplus(A^{(1)} \prec_{CB} C^{(1)}); \oplus(A^{(1)} \prec_{HB} A^{(2)});$
$\oplus(A^{(2)} \prec_{CB} B^{(2)}); \oplus(B^{(2)} \prec_{HB} B^{(3)});$
$\oplus(C^{(1)} \prec_{HB} C^{(3)}); \oplus(B^{(3)} \prec_{CB} C^{(3)})$

(ii) Per-Channel Projection

$G\#A\#c \triangleq !t_1; \oplus(A^{(1)}); \oplus(A^{(1)} \prec_{HB} B^{(3)}); \oplus(C^{(1)} \prec_{HB} C^{(3)})$
$G\#B\#c \triangleq \ominus(B^{(2)}); !t_3; \oplus(B^{(3)}); \ominus(A^{(1)} \prec_{HB} B^{(3)}); \oplus(C^{(1)} \prec_{HB} C^{(3)})$
$G\#C\#c \triangleq ?t_1; \oplus(C^{(1)}); ?t_3; \oplus(C^{(3)}); \ominus(C^{(1)} \prec_{HB} C^{(3)}); \oplus(A^{(1)} \prec_{HB} B^{(3)})$

$G\#A\#c_2 \triangleq \ominus(A^{(1)}); !t_2; \oplus(A^{(2)})$
$G\#B\#c_2 \triangleq ?t_2; \oplus(B^{(2)})$

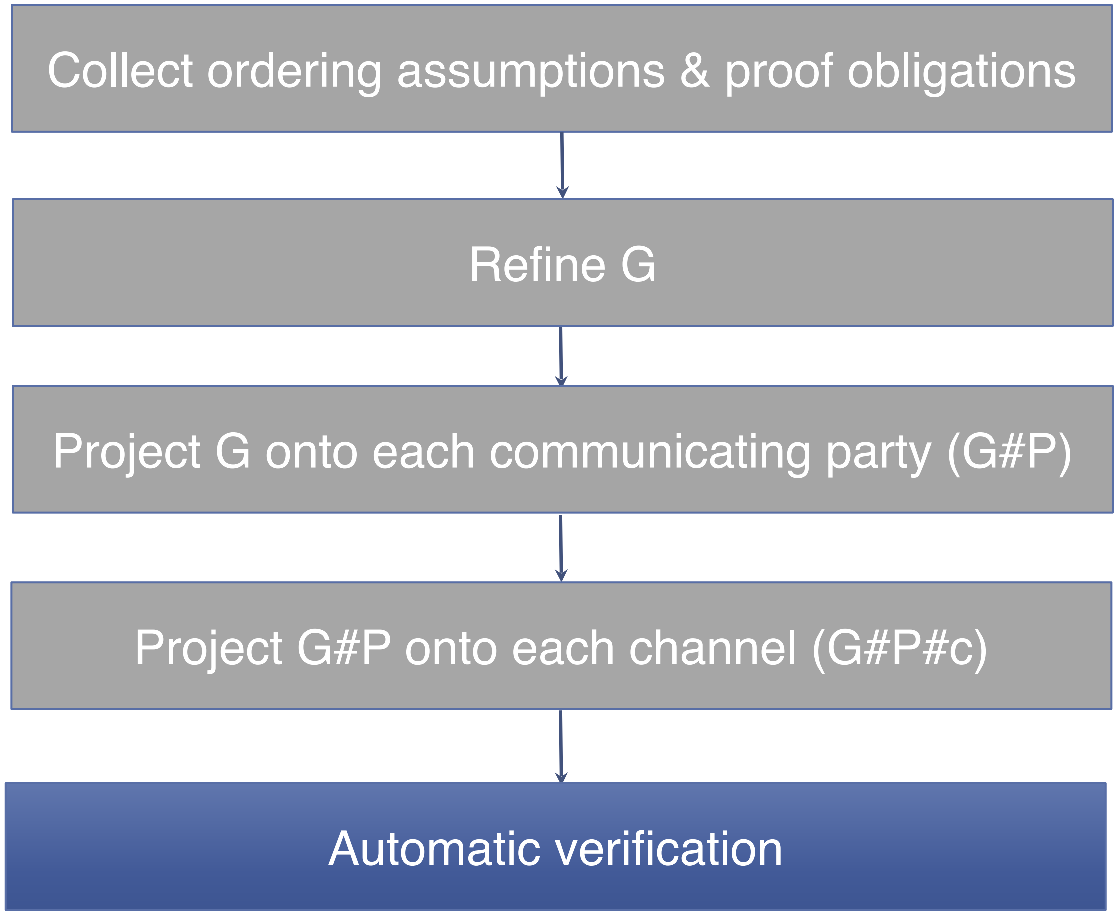
EXAMPLE 3: PER CHANNEL SPEC

$$(\mathtt{A} \xrightarrow{1} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_1 \rangle)\ ;\ (\mathtt{A} \xrightarrow{2} \mathtt{B} : \mathtt{c}_2\langle \mathtt{t}_2 \rangle)\ ;\ (\mathtt{B} \xrightarrow{3} \mathtt{C} : \mathtt{c}\langle \mathtt{t}_3 \rangle)$$

$$\ominus(1 \prec_{HB} 3)_{\mathtt{B}} = \ominus(\mathtt{A}^{(1)} \prec_{HB} \mathtt{B}^{(3)}); \oplus(\mathtt{C}^{(1)} \prec_{HB} \mathtt{C}^{(3)})$$
$$\ominus(1 \prec_{HB} 3)_{\mathtt{C}} = \ominus(\mathtt{C}^{(1)} \prec_{HB} \mathtt{C}^{(3)}); \oplus(\mathtt{A}^{(1)} \prec_{HB} \mathtt{B}^{(3)})$$

$$\ominus(1 \prec_{HB} 3)_{\mathtt{A}} = \oplus(\mathtt{A}^{(1)} \prec_{HB} \mathtt{B}^{(3)}); \oplus(\mathtt{C}^{(1)} \prec_{HB} \mathtt{C}^{(3)})$$

OVERVIEW OF OUR APPROACH



COMMUNICATION PRIMITIVES

[OPEN]

$$\frac{V_{pre}{=}\mathcal{C}(c_1, P_1, L_1) * \ldots * \mathcal{C}(c_1, P_n, L_n) \qquad V_{post}{=}\mathcal{C}(c, P_1, L_1) * \ldots * \mathcal{C}(c, P_n, L_n)}{\{V_{pre}\}\ \mathbf{c} = \mathbf{open}()\ \mathtt{with}\ (c_1, P_1..P_n)\ \{V_{post} \wedge \mathtt{bind}(c, c_1)\}}$$

[CLOSE]

$$\frac{V_{pre}{=}\mathcal{C}(c, P_1, \mathtt{emp}) * \ldots * \mathcal{C}(c, P_n, \mathtt{emp}) \qquad V_{post}{=}\mathcal{C}(c_1, P_1, \mathtt{emp}) * \ldots * \mathcal{C}(c_1, P_n, \mathtt{emp})}{\{V_{pre} \wedge \mathtt{bind}(c, c_1)\}\ \mathbf{close}(\mathbf{c})\ \{V_{post}\}}$$

[SEND]

$$\{\mathcal{C}(c, P, !v \cdot V(v); L) * V(x) \wedge \mathtt{Peer}(P)\}\ \mathbf{send}(\mathbf{c}, \mathbf{x})\ \{\mathcal{C}(c, P, L)\}$$

[RECV]

$$\{\mathcal{C}(c, P, ?v \cdot V(v); L) \wedge \mathtt{Peer}(P)\}\ \mathbf{x} = \mathbf{receive}(\mathbf{c})\ \{V(x) * \mathcal{C}(c, P, L)\}$$

EXAMPLE 3 - VERIFICATION

$$\{\texttt{Common}(\texttt{G\#All}) * \texttt{Party}(\texttt{A}, \texttt{G\#A}) * \texttt{Party}(\texttt{B}, \texttt{G\#B}) * \texttt{Party}(\texttt{C}, \texttt{G\#C})\}$$
$$(\texttt{Code}_\texttt{A} \mid\mid \texttt{Code}_\texttt{B} \mid\mid \texttt{Code}_\texttt{C})$$
$$\{\texttt{Party}(\texttt{A}, \texttt{emp}) * \texttt{Party}(\texttt{B}, \texttt{emp}) * \texttt{Party}(\texttt{C}, \texttt{emp})\}$$

“Release” lemma:

$$\texttt{Party}(\texttt{B}, \texttt{G\#B}) \Leftrightarrow \mathcal{C}(\texttt{c}, \texttt{B}, \texttt{G\#B\#s}) * \mathcal{C}(\texttt{c}_1, \texttt{B}, \texttt{G\#B\#c}_1)$$

“Join-emp” lemma:

$$\texttt{Party}(\texttt{B}, \texttt{emp}) \Leftrightarrow \mathcal{C}(\texttt{c}, \texttt{B}, \texttt{emp}) * \mathcal{C}(\texttt{c}_1, \texttt{B}, \texttt{emp})$$

FINAL REMARKS

Race-freedom via implicit & explicit synchronization

Ordering constraint system

Expressive session logic, which goes beyond types

More in the technical report

- Well-formedness of * and $\vee$
- Explicit synchronization specifications
- Recursion
- Full constraint system
- Entailment rules

WAIT-NOTIFYALL PRIMITIVES

$$\frac{\boxed{\textbf{CREATE}}\quad \mathtt{V}{=}\bigwedge_{\mathtt{j}\in\{2..n\}}\oplus(\mathtt{E_j} \Rightarrow \mathtt{E_1} \prec_{\mathtt{HB}} \mathtt{E_j})}{\{\mathtt{emp}\}\ \mathbf{w} = \mathbf{create}()\ \mathbf{with}\ \mathbf{E_1}, \overline{\mathbf{E_2..E_n}}\ \{\mathtt{NOTIFY}(\mathtt{w}, \ominus(\mathtt{E_1})) * \mathtt{WAIT}(\mathtt{w}, \mathtt{V})\}}$$

$$\boxed{\textbf{NOTIFY-ALL}}$$

$$\{\mathtt{NOTIFY}(\mathtt{w}, \ominus(\mathtt{E_1})) \wedge \mathtt{E_1}\}\ \mathbf{notifyAll}(\mathbf{w})\ \{\mathtt{NOTIFY}(\mathtt{w}, \mathtt{emp})\}$$

$$\frac{\boxed{\textbf{WAIT}}\quad \mathtt{V^{rel}}{=}\oplus(\mathtt{E_2} \Rightarrow \mathtt{E_1} \prec_{\mathtt{HB}} \mathtt{E_2})}{\{\mathtt{WAIT}(\mathtt{w}, \mathtt{V^{rel}}) \wedge \mathtt{not}(\mathtt{E_2})\}\ \mathbf{wait}(\mathbf{w})\ \{\mathtt{WAIT}(\mathtt{w}, \mathtt{emp}) * \mathtt{V^{rel}}\}}$$

(Wait lemma) $\oplus(\mathtt{E_2} \Rightarrow \mathtt{E_1} \prec_{\mathtt{HB}} \mathtt{E_2}) \wedge \mathtt{E_2} \Rightarrow \mathtt{E_1} \prec_{\mathtt{HB}} \mathtt{E_2}$

(Distribute-waits lemma) $\mathtt{WAIT}(\mathtt{w}, \bigwedge_{\mathtt{j}\in\{2..n\}} \Psi_{\mathtt{j}}) \Rightarrow \bigwedge_{\mathtt{j}\in\{2..n\}} \mathtt{WAIT}(\mathtt{w}, \Psi_{\mathtt{j}})$

Deadlock-check:

$$\mathtt{NOTIFY}(\mathtt{w}, \ominus(\mathtt{E_1})) * \mathtt{WAIT}(\mathtt{w}, \mathtt{emp}) \Rightarrow \mathtt{false}$$

MERCURIUS: SPECIFICATION LANGUAGE

Symbolic pred.	$pred$	$::= p(\mathtt{root}, v^*) \equiv \Phi \mid p(P^*, v^*) \equiv G$
Formula	Φ	$::= \bigvee \Delta \qquad \Delta ::= \exists\, v^* \cdot \kappa \wedge \pi \mid \Delta * \Delta$
Separation	κ	$::= \mathtt{emp} \mid v \mapsto d(v^*) \mid p(v^*) \mid C(v, P, L) \mid \kappa * \kappa \mid V$
Pure	π	$::= v : t \mid b \mid a \mid \phi \wedge \phi \mid \phi \vee \phi \mid \neg \phi$
		$\mid \exists v \cdot \phi \mid \forall v \cdot \phi \mid \gamma$
Pointer eq./diseq.	γ	$::= v{=}v \mid v{=}\mathtt{null} \mid v{\neq}v \mid v{\neq}\mathtt{null}$
Boolean	b	$::= \mathtt{true} \mid \mathtt{false} \mid b{=}b \qquad a ::= s{=}s \mid s{\leq}s \mid V{=}\Delta$
Presburger Arith.	s	$::= k^{\mathtt{int}} \mid v \mid k^{\mathtt{int}} \times s \mid s + s \mid -s$

where $k^{\mathtt{int}}$: integer constant; v : first order variable;
V : second-order variable; P : session role
d : name of a user-defined data structure
L : local protocol (defined in Fig. 5)