Communicating Transactions

Matthew Hennessy

joint work with Edsko de Vries, Vasileois Koutavas

FSEN11, Teheran, April 2011

TRINITY COLLEGE DUBLIN
Coláiste na Tríonóide, Baile Átha Cliath

science foundation ireland

Outline

Outline

Standard Transactions

- Transactions provide *an abstraction for error recovery* in a concurrent setting.

Standard Transactions

- Transactions provide *an abstraction for error recovery* in a concurrent setting.
- Guarantees:
  - Atomicity: Each transaction either runs in its entirety (commits) or not at all
  - Consistency: When faults are detected the transaction is automatically rolled-back
  - Isolation: The effects of a transaction are concealed from the rest of the system until the transaction commits
  - Durability: After a transaction commits, its effects are permanent

Standard Transactions

- Transactions provide *an abstraction for error recovery* in a concurrent setting.
- Guarantees:
  - Atomicity: Each transaction either runs in its entirety (commits) or not at all
  - Consistency: When faults are detected the transaction is automatically rolled-back
  - Isolation: The effects of a transaction are concealed from the rest of the system until the transaction commits
  - Durability: After a transaction commits, its effects are permanent
- Isolation:
  - good: provides coherent semantics
  - bad: limits concurrency
  - bad: limits co-operation between transactions and their environments

Communicating Transactions

- We *drop isolation* to *increase concurrency*
  - There is no limit on the communication between a transaction and its environment
- These new transactional systems guarantee:
  - Atomicity: Each transaction will either run in its entirety or not at all
  - Consistency: When faults are detected the transaction is automatically rolled-back, *together with all effects of the transaction on its environment*
  - Durability: After *all transactions that have interacted* commit, their effects are permanent (coordinated checkpointing)

Outline

Introduction

TransCCS

Liveness and safety properties

Compositional semantics

TransCCS

An extension of CCS with communicating transactions.

1. Simple language: 2 additional language constructs and 3 additional reduction rules.
2. Intricate concurrent and transactional behaviour:
   - encodes nested, restarting, and non-restarting transactions
   - does not limit communication between transactions
3. Simple behavioural theory: based on properties of systems:
   - *Safety* property: nothing bad happens
   - *Liveness* property: something good happens

TransCCS

Syntax:	P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
		$\mid$	$P \mid Q$	parallel
		$\mid$	$\nu a.P$	hiding
		$\mid$	$\mu X.P$	recursion
		$\mid$	$[\![P \rhd_k Q]\!]$	transaction (k bound in P)
		$\mid$	co k	commit

TransCCS

Syntax:	P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
		$\mid$	$P \mid Q$	parallel
		$\mid$	$\nu a.P$	hiding
		$\mid$	$\mu X.P$	recursion
		$\mid$	$[\![P \triangleright_k Q]\!]$	transaction (k bound in P)
		$\mid$	$\mathsf{co}\ k$	commit

TransCCS

Syntax:	P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
		$\mid$	$P \mid Q$	parallel
		$\mid$	$\nu a.P$	hiding
		$\mid$	$\mu X.P$	recursion
		$\mid$	$[\![P \triangleright_k Q]\!]$	transaction (k bound in P)
		$\mid$	$\mathrm{co}\ k$	commit

TransCCS

Syntax:

$$
\begin{array}{llll}
P, Q & ::= & \sum \mu_i.P_i & \text{guarded choice} \\
 & \mid & P \mid Q & \text{parallel} \\
 & \mid & \nu a.P & \text{hiding} \\
 & \mid & \mu X.P & \text{recursion} \\
 & \mid & [\![P \triangleright_k \; Q]\!] & \text{transaction } (k \text{ bound in } P) \\
 & \mid & \mathsf{co}\; k & \text{commit}
\end{array}
$$

TransCCS

Syntax:	P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
		$\mid$	$P \mid Q$	parallel
		$\mid$	$\nu a.P$	hiding
		$\mid$	$\mu X.P$	recursion
		$\mid$	$[\![P \triangleright_k Q]\!]$	transaction (k bound in P)
		$\mid$	$\mathrm{co}\ k$	commit

TransCCS

Syntax:

$$
\begin{array}{llll}
P, Q & ::= & \sum \mu_i.P_i & \text{guarded choice} \\
 & \mid & P \mid Q & \text{parallel} \\
 & \mid & \nu a.P & \text{hiding} \\
 & \mid & \mu X.P & \text{recursion} \\
 & \mid & [\![P \rhd_k Q]\!] & \text{transaction } (k \text{ bound in } P) \\
 & \mid & \text{co } k & \text{commit}
\end{array}
$$

TransCCS

Syntax:	P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
		$\mid$	$P \mid Q$	parallel
		$\mid$	$\nu a.P$	hiding
		$\mid$	$\mu X.P$	recursion
		$\mid$	$[\![P \triangleright_k Q]\!]$	transaction (k bound in P)
		$\mid$	$\text{co } k$	commit

Transaction $[\![P \triangleright_k Q]\!]$

- execute P to completion ($\text{co } k$)
- subject to random aborts
- if aborted roll back all effects of P and initiate Q

TransCCS

Syntax:

P, Q	$::=$	$\sum \mu_i.P_i$	guarded choice
	$\mid$	$P \mid Q$	parallel
	$\mid$	$\nu a.P$	hiding
	$\mid$	$\mu X.P$	recursion
	$\mid$	$[\![P \triangleright_k Q]\!]$	transaction (k bound in P)
	$\mid$	$\text{co } k$	commit

Transaction $[\![P \triangleright_k Q]\!]$

- execute P to completion ($\text{co } k$)
- subject to random aborts
- if aborted roll back all effects of P and initiate Q
- roll back includes . . . environmental impact of P

Rollbacks and Commits

Co-operating actions: $\boxed{a \longleftarrow \text{needs co-operation of} \longrightarrow \overline{a}}$

Rollbacks and Commits

Co-operating actions: $a \leftarrow$ needs co-operation of $\rightarrow \overline{a}$

$$T_a \mid T_b \mid T_c \mid P_d \mid P_e$$

where

$$\begin{aligned}
T_a &= [\![\overline{d}.\overline{b}.(\text{co } k_1 \mid a) \triangleright_{k_1} \mathbf{0}]\!] \\
T_b &= [\![\overline{c}.(\text{co } k_2 \mid b) \triangleright_{k_2} \mathbf{0}]\!] \\
T_c &= [\![\overline{e}.c.\text{co } k_3 \triangleright_{k_3} \mathbf{0}]\!] \\
P_d &= d.R_d \\
P_e &= e.R_e
\end{aligned}$$

Rollbacks and Commits

Co-operating actions: $a \leftarrow$ needs co-operation of $\rightarrow \overline{a}$

$$T_a \mid T_b \mid T_c \mid P_d \mid P_e$$

where

$$\begin{aligned} T_a &= [\![\overline{d}.\overline{b}.(\text{co } k_1 \mid a) \rhd_{k_1} \mathbb{0}]\!] \\ T_b &= [\![\overline{c}.(\text{co } k_2 \mid b) \rhd_{k_2} \mathbb{0}]\!] \\ T_c &= [\![\overline{e}.c.\text{co } k_3 \rhd_{k_3} \mathbb{0}]\!] \\ P_d &= d.R_d \\ P_e &= e.R_e \end{aligned}$$

- if T_c aborts, what roll-backs are necessary?

Rollbacks and Commits

Co-operating actions: $\boxed{a \leftarrow \text{needs co-operation of} \rightarrow \overline{a}}$

$$T_a \mid T_b \mid T_c \mid P_d \mid P_e$$

where

$$\begin{aligned}
T_a &= [\![\overline{d}.\overline{b}.(\text{co } k_1 \mid a) \triangleright_{k_1} \mathbf{0}]\!] \\
T_b &= [\![\overline{c}.(\text{co } k_2 \mid b) \triangleright_{k_2} \mathbf{0}]\!] \\
T_c &= [\![\overline{e}.c.\text{co } k_3 \triangleright_{k_3} \mathbf{0}]\!] \\
P_d &= d.R_d \\
P_e &= e.R_e
\end{aligned}$$

- if T_c aborts, what roll-backs are necessary?
- When can action a be considered permanent?

Rollbacks and Commits

Co-operating actions: $a \leftarrow$ needs co-operation of $\rightarrow \overline{a}$

$$T_a \mid T_b \mid T_c \mid P_d \mid P_e$$

where

$$\begin{aligned}
T_a &= [\![\overline{d}.\overline{b}.(\text{co } k_1 \mid a) \rhd_{k_1} \mathbb{0}]\!] \\
T_b &= [\![\overline{c}.(\text{co } k_2 \mid b) \rhd_{k_2} \mathbb{0}]\!] \\
T_c &= [\![\overline{e}.c.\text{co } k_3 \rhd_{k_3} \mathbb{0}]\!] \\
P_d &= d.R_d \\
P_e &= e.R_e
\end{aligned}$$

- if T_c aborts, what roll-backs are necessary?
- When can action a be considered permanent?
- When can code R_d be considered permanent?

Reduction semantics main rules

R-Comm

$$\frac{a_i = \overline{b}_j}{\sum_{i \in I} a_i.P_i \mid \sum_{j \in J} b_j.Q_j \rightarrow P_i \mid Q_j}$$

Communication

R-Co

$$\frac{}{\llbracket P \mid \text{co } k \triangleright_k Q \rrbracket \rightarrow P}$$

Commit

R-Ab

$$\frac{}{\llbracket P \triangleright_k Q \rrbracket \rightarrow Q}$$

Random abort

Reduction semantics main rules

R-COMM

$$\frac{a_i = \overline{b}_j}{\sum_{i \in I} a_i.P_i \mid \sum_{j \in J} b_j.Q_j \to P_i \mid Q_j}$$

Communication

R-CO

$$\frac{}{[\![P \mid \mathsf{co}\ k \triangleright_k Q]\!] \to P}$$

Commit

R-AB

$$\frac{}{[\![P \triangleright_k Q]\!] \to Q}$$

Random abort

R-EMB

$$\frac{k \notin R}{[\![P \triangleright_k Q]\!] \mid R \to [\![P \mid R \triangleright_k Q \mid R]\!]}$$

Embed

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k r]\!]$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid \llbracket \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ r \rrbracket$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k\ r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k\ a.c.\omega + e.\mho \mid r]\!]$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid \llbracket \overline{a}.\overline{c}.co\ k + \overline{e} \rhd_k r \rrbracket$$

$$\xrightarrow{\text{R-Emb}} \llbracket a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.co\ k + \overline{e} \rhd_k a.c.\omega + e.\mho \mid r \rrbracket$$

$$\xrightarrow{\text{R-Comm}} \llbracket c.\omega \mid \overline{c}.co\ k \rhd_k a.c.\omega + e.\mho \mid r \rrbracket$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\ c.\omega \mid \overline{c}.\mathrm{co}\ k \rhd_k\ a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\ \omega \mid \mathrm{co}\ k \rhd_k\ a.c.\omega + e.\mho \mid r]\!]$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\ c.\omega \mid \overline{c}.\mathrm{co}\ k \triangleright_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\ \omega \mid \mathrm{co}\ k \triangleright_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-CO}} \omega$$

Simple Example

Convention:

- ω: I am happy
- $\mho$: I am sad

$$
\begin{aligned}
& a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ r]\!] \\
\xrightarrow{\text{R-EMB}}\ & [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ a.c.\omega + e.\mho \mid r]\!] \\
\xrightarrow{\text{R-COMM}}\ & [\![\ c.\omega \mid \overline{c}.\mathrm{co}\ k \rhd_k\ a.c.\omega + e.\mho \mid r]\!] \\
\xrightarrow{\text{R-COMM}}\ & [\![\ \omega \mid \mathrm{co}\ k \rhd_k\ a.c.\omega + e.\mho \mid r]\!] \\
\xrightarrow{\text{R-CO}}\ & \omega
\end{aligned}
$$

Simple Example (a second trace)

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k r]\!]$$

Simple Example (a second trace)

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k\ r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \triangleright_k\ a.c.\omega + e.\mho \mid r]\!]$$

Simple Example (a second trace)

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k\ a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\quad \mho \quad \rhd_k\ a.c.\omega + e.\mho \mid r]\!]$$

Simple Example (a second trace)

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\text{co } k + \overline{e} \rhd_k r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\text{co } k + \overline{e} \rhd_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\quad \mho \quad \rhd_k a.c.\omega + e.\mho \mid r]\!] \quad \text{(Deadlocked)}$$

Simple Example (a second trace)

$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathsf{co}\ k + \overline{e} \rhd_k r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathsf{co}\ k + \overline{e} \rhd_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-COMM}} [\![\quad \mho \quad \rhd_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-AB}} a.c.\omega + e.\mho \mid r$$

Simple Example (a second trace)

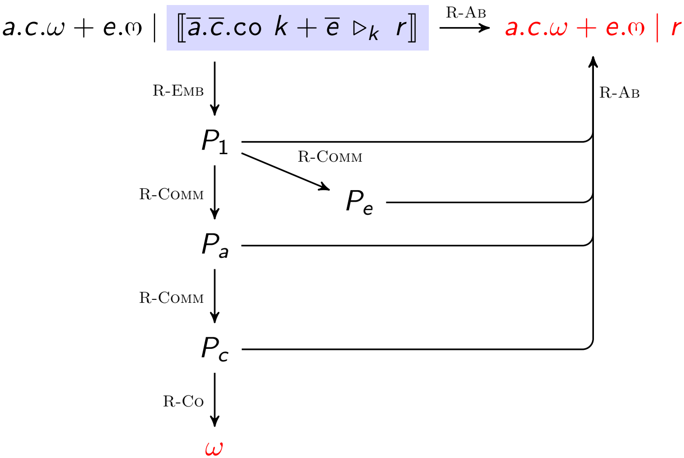
$$a.c.\omega + e.\mho \mid [\![\overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k r]\!]$$

$$\xrightarrow{\text{R-EMB}} [\![a.c.\omega + e.\mho \mid \overline{a}.\overline{c}.\mathrm{co}\ k + \overline{e} \rhd_k a.c.\omega + e.\mho \mid r]\!]$$

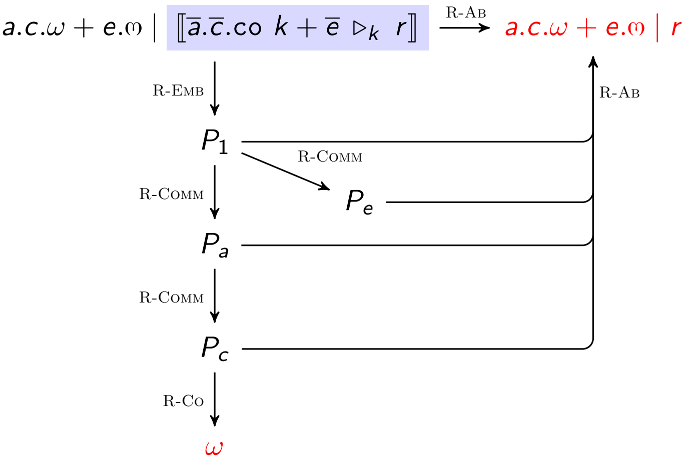
$$\xrightarrow{\text{R-COMM}} [\![\mho \rhd_k a.c.\omega + e.\mho \mid r]\!]$$

$$\xrightarrow{\text{R-AB}} a.c.\omega + e.\mho \mid r$$ (The environment is restored)

Simple Example (all traces)

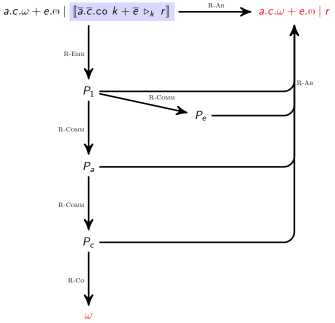


Simple Example (all traces)



Will never be sad: $\mho$ assuming r does not contain $\overline{e}$

Aborting transactions



A commit step makes the effects of the transaction permanent (**Durability**)

An abort step:

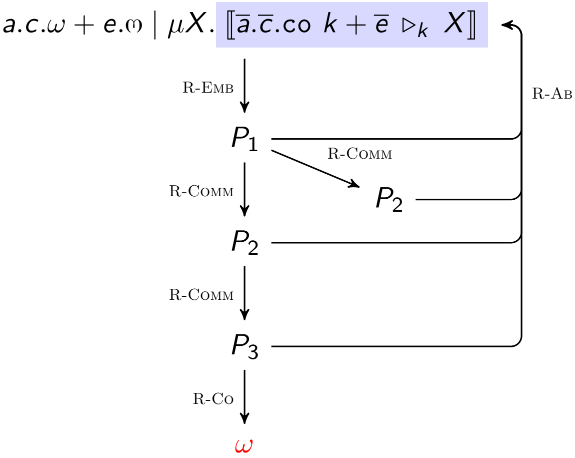
- restarts the transaction
- rolls-back embedded processes to their state before embedding (**Consistency**)
- does not roll-back actions that happened before embedding
- does not affect non-embedded processes

The behavioural theory will show the **Atomicity** property.

Restarting transactions

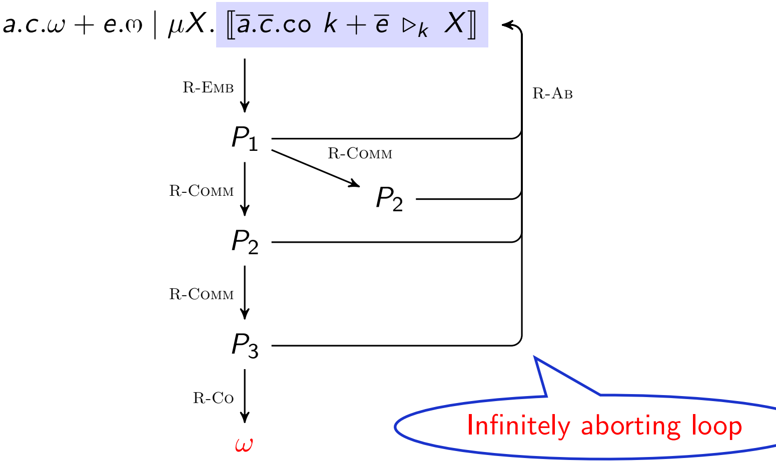
$$a.c.\omega + e.\mho \mid \mu X. \llbracket \overline{a}.\overline{c}.co\ k + \overline{e} \triangleright_k X \rrbracket$$

Restarting transactions



Will never be sad: $\mho$

Restarting transactions



Will never be sad: ℧

Double Embedding

$$[\![a.\mathrm{co}\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\overline{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!]$$

Double Embedding

$$[\![a.\mathsf{co}\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\overline{a}.\mathsf{co}\ l \mid c \rhd_l \mathbf{0}]\!]$$

Double Embedding

$$[\![a.\mathrm{co}\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[a.\mathrm{co}\ k \mid b \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!]\right]\!\!\right]$$

Double Embedding

$$[\![a.\mathsf{co}\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\overline{a}.\mathsf{co}\ l \mid c \rhd_l \mathbf{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[a.\mathsf{co}\ k \mid b \mid [\![\overline{a}.\mathsf{co}\ l \mid c \rhd_l \mathbf{0}]\!] \rhd_k [\![\overline{a}.\mathsf{co}\ l \mid c \rhd_l \mathbf{0}]\!] \right]\!\!\right]$$

Double Embedding

$$[\![a.\mathrm{co}\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[a.\mathrm{co}\ k \mid b \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[b \mid [\![a.\mathrm{co}\ k \mid \bar{a}.\mathrm{co}\ l \mid c \rhd_l a.\mathrm{co}\ k]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbf{0}]\!] \right]\!\!\right]$$

Double Embedding

$$[\![a.co\ k \mid b \rhd_k \mathbf{0}]\!] \mid [\![\bar{a}.co\ l \mid c \rhd_l \mathbf{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \Big[\!\!\Big[a.co\ k \mid b \mid [\![\bar{a}.co\ l \mid c \rhd_l \mathbf{0}]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbf{0}]\!] \Big]\!\!\Big]$$

$$\xrightarrow{\text{R-EMB}} \Big[\!\!\Big[b \mid [\![a.co\ k \mid \bar{a}.co\ l \mid c \rhd_l a.co\ k]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbf{0}]\!] \Big]\!\!\Big]$$

$$\xrightarrow{\text{R-COMM}} \Big[\!\!\Big[b \mid [\![co\ k \mid co\ l \mid c \rhd_l a.co\ k]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbf{0}]\!] \Big]\!\!\Big]$$

Double Embedding

$$\llbracket a.\mathsf{co}\ k \mid b \rhd_k \mathbb{0} \rrbracket \mid \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket$$

$$\xrightarrow{\text{R-EMB}} \left\llbracket a.\mathsf{co}\ k \mid b \mid \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket \rhd_k \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket \right\rrbracket$$

$$\xrightarrow{\text{R-EMB}} \left\llbracket b \mid \llbracket a.\mathsf{co}\ k \mid \bar{a}.\mathsf{co}\ l \mid c \rhd_l a.\mathsf{co}\ k \rrbracket \rhd_k \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket \right\rrbracket$$

$$\xrightarrow{\text{R-COMM}} \left\llbracket b \mid \llbracket \mathsf{co}\ k \mid \mathsf{co}\ l \mid c \rhd_l a.\mathsf{co}\ k \rrbracket \rhd_k \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket \right\rrbracket$$

$$\xrightarrow{\text{R-CO}} \left\llbracket b \mid \mathsf{co}\ k \mid c \rhd_k \llbracket \bar{a}.\mathsf{co}\ l \mid c \rhd_l \mathbb{0} \rrbracket \right\rrbracket$$

Double Embedding

$$[\![a.\mathrm{co}\ k \mid b \rhd_k \mathbb{0}]\!] \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[a.\mathrm{co}\ k \mid b \mid [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[b \mid [\![a.\mathrm{co}\ k \mid \bar{a}.\mathrm{co}\ l \mid c \rhd_l a.\mathrm{co}\ k]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-COMM}} \left[\!\!\left[b \mid [\![\mathrm{co}\ k \mid \mathrm{co}\ l \mid c \rhd_l a.\mathrm{co}\ k]\!] \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-CO}} \left[\!\!\left[b \mid \mathrm{co}\ k \mid c \rhd_k [\![\bar{a}.\mathrm{co}\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-CO}} b \mid c$$

Double Embedding

$$[\![a.co\ k \mid b \rhd_k \mathbb{0}]\!] \mid [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[a.co\ k \mid b \mid [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-EMB}} \left[\!\!\left[b \mid [\![a.co\ k \mid \bar{a}.co\ l \mid c \rhd_l a.co\ k]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-COMM}} \left[\!\!\left[b \mid [\![co\ k \mid co\ l \mid c \rhd_l a.co\ k]\!] \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-CO}} \left[\!\!\left[b \mid co\ k \mid c \rhd_k [\![\bar{a}.co\ l \mid c \rhd_l \mathbb{0}]\!] \right]\!\!\right]$$

$$\xrightarrow{\text{R-CO}} b \mid c$$

Outline

Safety properties

Safety: "Nothing bad will happen" [Lamport'77]

- A safety property can be formulated as a *safety test* T^{ω} which signals on channel ω when it detects the bad behaviour

Examples:

- $\mu X.(a.X + e.\omega)$ can not perform e while performing any sequence of as
- $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$ can not perform e when a followed by b is offered.

Safety properties

Safety: "Nothing bad will happen" [Lamport'77]

- A safety property can be formulated as a *safety test* T^{ω} which signals on channel ω when it detects the bad behaviour

Examples:

- $\mu X.(a.X + e.\omega)$ can not perform e while performing any sequence of as
- $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$ can not perform e when a followed by b is offered.

- P *passes the safety test* T^{ω} when $P \mid T^{\omega}$ can not output on ω
  - This is the negation of passing a "may test" [DeNicola-Hennessy'84]

Safety properties

Safety: "Nothing bad will happen" [Lamport'77]

- A safety property can be formulated as a *safety test* T^{ω} which signals on channel ω when it detects the bad behaviour

Examples:

- $\mu X.(a.X + e.\omega)$ can not perform e while performing any sequence of as
- $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$ can not perform e when a followed by b is offered.

- *P passes the safety test T^{ω}* when $P \mid T^{\omega}$ can not output on ω
  - This is the negation of passing a "may test" [DeNicola-Hennessy'84]

Examples:

- $I_3 = \mu X. \llbracket a.b.co\ k + \overline{e} \triangleright_k\ X \rrbracket$ passes safety test T^{ω}
- $I_4 = \mu X. \llbracket a.b.co\ k \mid \overline{e} \triangleright_k\ X \rrbracket$ does not pass safety test T^{ω}

Safety

Definition (Basic Observable)

$P{\Downarrow}_{\omega}$ iff there exists P' such that $P \rightarrow^* P' \mid \omega$

- Basic observable actions are *permanent*

Safety

Definition (Basic Observable)

$P{\Downarrow_{\omega}}$ iff there exists P' such that $P \rightarrow^* P' \mid \omega$

- Basic observable actions are *permanent*
- True: $[\![a.b.\mathrm{co}\ k \mid \overline{e} \rhd_k \mathbf{0}]\!] \mid (e.\omega \mid \overline{a}.\overline{b}) {\Downarrow_{\omega}}$

Safety

Definition (Basic Observable)

$P{\Downarrow_{\omega}}$ iff there exists P' such that $P \rightarrow^* P' \mid \omega$

- Basic observable actions are *permanent*
- True: $[\![a.b.co\ k \mid \overline{e} \rhd_k \mathbb{0}]\!] \mid (e.\omega \mid \overline{a}.\overline{b}) \Downarrow_{\omega}$
- False: $[\![a.b.co\ k + \overline{e} \rhd_k \mathbb{0}]\!] \mid (e.\omega \mid \overline{a}.\overline{b}) \Downarrow_{\omega}$

Safety

Definition (Basic Observable)

$P \Downarrow_{\omega}$ iff there exists P' such that $P \rightarrow^{*} P' \mid \omega$

- Basic observable actions are *permanent*
- True: $[\![a.b.\mathrm{co}\ k \mid \overline{e} \rhd_{k} \mathbf{0}]\!] \mid (e.\omega \mid \overline{a}.\overline{b}) \Downarrow_{\omega}$
- False: $[\![a.b.\mathrm{co}\ k + \overline{e} \rhd_{k} \mathbf{0}]\!] \mid (e.\omega \mid \overline{a}.\overline{b}) \Downarrow_{\omega}$

Definition (P Passes safety test T^{ω})

$$P \;\mathrm{cannot}\; T^{\omega} \quad \text{when} \quad P \mid T^{\omega} \not\Downarrow_{\omega}$$

Safety

Definition (Basic Observable)

$P{\Downarrow_{\mho}}$ iff there exists P' such that $P \rightarrow^* P' \mid \mho$

- Basic observable actions are *permanent*
- True: $[\![a.b.co\ k \mid \overline{e} \rhd_k\ \mathbf{0}]\!] \ \mid\ (e.\mho \mid \overline{a}.\overline{b}) \Downarrow_{\mho}$
- False: $[\![a.b.co\ k + \overline{e} \rhd_k\ \mathbf{0}]\!] \ \mid\ (e.\mho \mid \overline{a}.\overline{b}) \Downarrow_{\mho}$

Definition (P Passes safety test $T^{\mho}$)

$$P \text{ cannot } T^{\mho} \quad \text{when} \quad P \mid T^{\mho} \not\Downarrow_{\mho}$$

Definition (Safety Preservation)

$$S \sqsubseteq_{\text{safe}} I \quad \text{when} \quad \forall T^{\mho}.\quad S \text{ cannot } T^{\mho} \quad \text{implies} \quad I \text{ cannot } T^{\mho}$$

Safety preservation: Examples

$$
\begin{aligned}
S_{ab} &= \mu X. [\![a.b.\text{co } k \rhd_k X]\!] \\
I_3 &= \mu X. [\![a.b.\text{co } k + \overline{e} \rhd_k X]\!] \\
I_4 &= \mu X. [\![a.b.\text{co } k \mid \overline{e} \rhd_k X]\!]
\end{aligned}
$$

Safety preservation: Examples

$$
\begin{aligned}
S_{ab} &= \mu X.\,[\![a.b.co\ k \rhd_k X]\!] \\
I_3 &= \mu X.\,[\![a.b.co\ k + \overline{e} \rhd_k X]\!] \\
I_4 &= \mu X.\,[\![a.b.co\ k \mid \overline{e} \rhd_k X]\!]
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\mathrm{safe}} I_4$

Safety preservation: Examples

$$S_{ab} = \mu X. \llbracket a.b.\mathsf{co}\ k \rhd_k X \rrbracket$$

$$I_3 = \mu X. \llbracket a.b.\mathsf{co}\ k + \overline{e} \rhd_k X \rrbracket$$

$$I_4 = \mu X. \llbracket a.b.\mathsf{co}\ k \mid \overline{e} \rhd_k X \rrbracket$$

- $S_{ab} \not\sqsubseteq_{\mathrm{safe}} I_4$ use test $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$

Safety preservation: Examples

$$S_{ab} = \mu X. [\![a.b.\mathsf{co}\ k \rhd_k X]\!]$$
$$I_3 = \mu X. [\![a.b.\mathsf{co}\ k + \overline{e} \rhd_k X]\!]$$
$$I_4 = \mu X. [\![a.b.\mathsf{co}\ k \mid \overline{e} \rhd_k X]\!]$$

- $S_{ab} \not\sqsubseteq_{\mathrm{safe}} I_4$ use test $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$
- $S_{ab} \sqsubseteq_{\mathrm{safe}} I_3$

Safety preservation: Examples

$$
\begin{aligned}
S_{ab} &= \mu X. [\![a.b.\mathsf{co}\ k \rhd_k X]\!] \\
I_3 &= \mu X. [\![a.b.\mathsf{co}\ k + \overline{e} \rhd_k X]\!] \\
I_4 &= \mu X. [\![a.b.\mathsf{co}\ k \mid \overline{e} \rhd_k X]\!]
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\mathrm{safe}} I_4$ use test $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$
- $S_{ab} \sqsubseteq_{\mathrm{safe}} I_3$ – proof techniques required

Safety preservation: Examples

$$
\begin{aligned}
S_{ab} &= \mu X. \llbracket a.b.co\ k \rhd_k X \rrbracket \\
I_3 &= \mu X. \llbracket a.b.co\ k + \overline{e} \rhd_k X \rrbracket \\
I_4 &= \mu X. \llbracket a.b.co\ k \mid \overline{e} \rhd_k X \rrbracket
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\text{safe}} I_4$ use test $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$
- $S_{ab} \sqsubseteq_{\text{safe}} I_3$ – proof techniques required
- $\tau.P + \tau.Q \sqsubseteq_{\text{safe}} \llbracket P \rhd_k Q \rrbracket$, for any P, Q

Safety preservation: Examples

$$S_{ab} = \mu X. \llbracket a.b.co\ k \triangleright_k X \rrbracket$$
$$I_3 = \mu X. \llbracket a.b.co\ k + \overline{e} \triangleright_k X \rrbracket$$
$$I_4 = \mu X. \llbracket a.b.co\ k \mid \overline{e} \triangleright_k X \rrbracket$$

- $S_{ab} \not\sqsubseteq_{\text{safe}} I_4$ use test $T^{\omega} = e.\omega \mid \overline{a}.\overline{b}$
- $S_{ab} \sqsubseteq_{\text{safe}} I_3$ – proof techniques required
- $\tau.P + \tau.Q \sqsubseteq_{\text{safe}} \llbracket P \triangleright_k Q \rrbracket$, for any P, Q – proof techniques rqd

Liveness

Liveness: "Something good will eventually happen" [Lamport'77]

- A liveness property can be formulated as a *liveness test* T^ω which detects and reports good behaviour on ω.

Examples:

- $T^\omega = \overline{a}.\overline{b}.\omega$ can do an a then a b
- $\mu X.\ [\![\overline{a}.\overline{b}.(\omega \mid \text{co } l) \triangleright_l X]\!]$ can eventually do an a, b uninterrupted?
- $a.\mu X.\ [\![\overline{b}.\overline{c}.(\omega \mid \text{co } l) \triangleright_l X]\!]$ English?

Liveness

Liveness: "Something good will eventually happen" [Lamport'77]

- A liveness property can be formulated as a *liveness test* T^{ω} which detects and reports good behaviour on ω.

Examples:

- $T^{\omega} = \overline{a}.\overline{b}.\omega$ can do an a then a b
- $\mu X. \; [\![\overline{a}.\overline{b}.(\omega \mid \text{co } I) \rhd_I \; X]\!]$ can eventually do an a, b uninterrupted?
- $a.\mu X. \; [\![\overline{b}.\overline{c}.(\omega \mid \text{co } I) \rhd_I \; X]\!]$ English?

- *P passes the liveness test* T^{ω} when ω is eventually guaranteed

Liveness

Liveness: "Something good will eventually happen" [Lamport'77]

- A liveness property can be formulated as a *liveness test* T^ω which detects and reports good behaviour on ω.

Examples:

- $T^\omega = \overline{a}.\overline{b}.\omega$ can do an a then a b
- $\mu X.\ [\![\overline{a}.\overline{b}.(\omega \mid \text{co } I) \rhd_I X]\!]$ can eventually do an a, b uninterrupted?
- $a.\mu X.\ [\![\overline{b}.\overline{c}.(\omega \mid \text{co } I) \rhd_I X]\!]$ English?

- *P passes the liveness test* T^ω when ω is eventually guaranteed

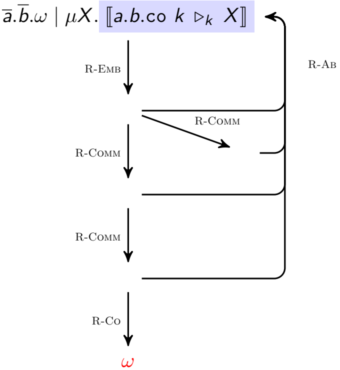
Dilemma: What does this mean?

Dilemma

Does $\mu X.\ [\![a.b.\text{co}\ k \triangleright_k X]\!]$ pass liveness test $T^{\omega}_{ab} = \overline{a}.\overline{b}.\omega$?

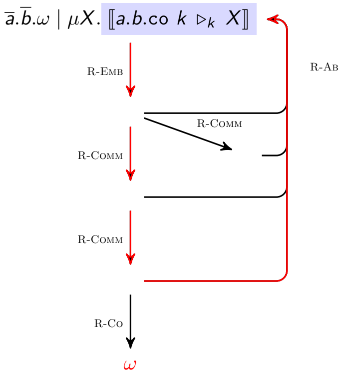
Dilemma

Does $\mu X. \llbracket a.b.co\ k \rhd_k X \rrbracket$ pass liveness test $T^{\omega}_{ab} = \overline{a}.\overline{b}.\omega$?



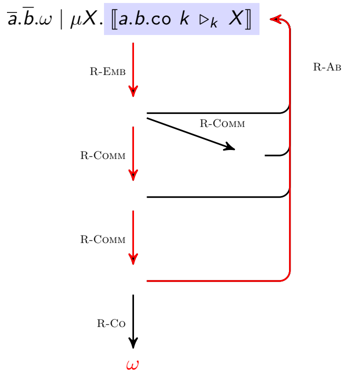
Dilemma

Does $\mu X.\, [\![a.b.\mathsf{co}\ k \triangleright_k X]\!]$ pass liveness test $T^{\omega}_{ab} = \overline{a}.\overline{b}.\omega$?



Dilemma

Does $\mu X. \llbracket a.b.\mathrm{co}\ k \rhd_k\ X \rrbracket$ pass liveness test $T^{\omega}_{ab} = \overline{a}.\overline{b}.\omega$?



- **must-testing:** NO because of infinite loop
- **should-testing:** YES

Liveness testing

Definition (P Passes liveness test T^ω [Rensink-Vogler'07])

$$P \,\mathrm{shd}\, T^\omega \quad \text{when} \quad \forall R.\quad P \mid T^\omega \rightarrow^* R \quad \text{implies} \quad R{\Downarrow_\omega}$$

Liveness testing

Definition (P Passes liveness test T^ω [Rensink-Vogler'07])

$$P \,\mathrm{shd}\, T^\omega \quad \text{when} \quad \forall R. \quad P \mid T^\omega \rightarrow^* R \quad \text{implies} \quad R{\Downarrow_\omega}$$

Examples:

- $\mu X.\, [\![a.b.\mathsf{co}\ k \ \triangleright_k\ X]\!]$ passes liveness test $T^\omega_{ab} = \overline{a}.\overline{b}.\omega$
- $[\![a.b.\mathsf{co}\ k \ \triangleright_k\ \mathbf{0}]\!]$ does not pass test T^ω_{ab}

Liveness testing

Definition (P Passes liveness test T^ω [Rensink-Vogler'07])

$$P \,\mathrm{shd}\, T^\omega \quad \text{when} \quad \forall R. \quad P \mid T^\omega \to^* R \quad \text{implies} \quad R{\Downarrow_\omega}$$

Examples:

- $\mu X. [\![a.b.co\ k \triangleright_k X]\!]$ passes liveness test $T^\omega_{ab} = \overline{a}.\overline{b}.\omega$
- $[\![a.b.co\ k \triangleright_k \mathbf{0}]\!]$ does not pass test T^ω_{ab}

Definition (Liveness preservation)

$$S \sqsubseteq_{\mathrm{live}} I \quad \text{when} \quad \forall T^\omega. \quad S \,\mathrm{shd}\, T^\omega \quad \text{implies} \quad I \,\mathrm{shd}\, T^\omega$$

Liveness preservation:Examples

$$S_{ab} = \mu X. [\![a.b.\mathrm{co}\ k \triangleright_k X]\!]$$
$$I_2 = \mu X. [\![a.b.\mathbb{0} \triangleright_k X]\!]$$
$$I_3 = \mu X. [\![a.b.\mathrm{co}\ k + \overline{e} \triangleright_k X]\!]$$

Liveness preservation:Examples

$$S_{ab} = \mu X. [\![a.b.\mathsf{co}\ k \rhd_k X]\!]$$
$$l_2 = \mu X. [\![a.b.\mathbb{0} \rhd_k X]\!]$$
$$l_3 = \mu X. [\![a.b.\mathsf{co}\ k + \overline{e} \rhd_k X]\!]$$

- $S_{ab} \not\sqsubseteq_{\mathrm{live}} l_2$

Liveness preservation:Examples

$$
\begin{aligned}
S_{ab} &= \mu X. \llbracket a.b.\mathrm{co}\ k \ \triangleright_k \ X \rrbracket \\
l_2 &= \mu X. \llbracket a.b.\mathbb{0} \ \triangleright_k \ X \rrbracket \\
l_3 &= \mu X. \llbracket a.b.\mathrm{co}\ k + \overline{e} \ \triangleright_k \ X \rrbracket
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\text{live}} l_2$ use test $T^\omega = \overline{a}.\overline{b}.\omega$

Liveness preservation:Examples

$$S_{ab} = \mu X. \llbracket a.b.co\ k \rhd_k X \rrbracket$$
$$I_2 = \mu X. \llbracket a.b.\mathbb{0} \rhd_k X \rrbracket$$
$$I_3 = \mu X. \llbracket a.b.co\ k + \overline{e} \rhd_k X \rrbracket$$

- $S_{ab} \not\sqsubsetapprox_{\text{live}} I_2$ use test $T^{\omega} = \overline{a}.\overline{b}.\omega$
- $S_{ab} \sqsubsetapprox_{\text{live}} I_3$

Liveness preservation:Examples

$$S_{ab} = \mu X. \llbracket a.b.\mathrm{co}\ k \triangleright_k X \rrbracket$$
$$I_2 = \mu X. \llbracket a.b.\mathbb{0} \triangleright_k X \rrbracket$$
$$I_3 = \mu X. \llbracket a.b.\mathrm{co}\ k + \overline{e} \triangleright_k X \rrbracket$$

- $S_{ab} \not\sqsubseteq_{\text{live}} I_2$ use test $T^\omega = \overline{a}.\overline{b}.\omega$
- $S_{ab} \sqsubseteq_{\text{live}} I_3$ – proof techniques required

Liveness preservation:Examples

$$
\begin{aligned}
S_{ab} &= \mu X.\, [\![a.b.\mathsf{co}\ k \rhd_k X]\!] \\
I_2 &= \mu X.\, [\![a.b.\mathbb{0} \rhd_k X]\!] \\
I_3 &= \mu X.\, [\![a.b.\mathsf{co}\ k + \overline{e} \rhd_k X]\!]
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\text{live}} I_2$ use test $T^\omega = \overline{a}.\overline{b}.\omega$
- $S_{ab} \sqsubseteq_{\text{live}} I_3$ – proof techniques required
- $\mu X.\, [\![P \mid \mathsf{co}\ k \rhd_k X]\!] \eqsim_{\text{live}} P$, for any P

Liveness preservation:Examples

$$
\begin{aligned}
S_{ab} &= \mu X. \llbracket a.b.\text{co } k \triangleright_k X \rrbracket \\
I_2 &= \mu X. \llbracket a.b.\mathbb{0} \triangleright_k X \rrbracket \\
I_3 &= \mu X. \llbracket a.b.\text{co } k + \overline{e} \triangleright_k X \rrbracket
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\text{live}} I_2$ use test $T^\omega = \overline{a}.\overline{b}.\omega$
- $S_{ab} \sqsubseteq_{\text{live}} I_3$ – proof techniques required
- $\mu X. \llbracket P \mid \text{co } k \triangleright_k X \rrbracket \eqsim_{\text{live}} P$, for any P – proof techniques rqd

Liveness preservation:Examples

$$
\begin{aligned}
S_{ab} &= \mu X.\, [\![a.b.\mathsf{co}\ k \triangleright_k X]\!] \\
I_2 &= \mu X.\, [\![a.b.\mathbf{0} \triangleright_k X]\!] \\
I_3 &= \mu X.\, [\![a.b.\mathsf{co}\ k + \overline{e} \triangleright_k X]\!]
\end{aligned}
$$

- $S_{ab} \not\sqsubseteq_{\mathrm{live}} I_2$ use test $T^{\omega} = \overline{a}.\overline{b}.\omega$
- $S_{ab} \sqsubseteq_{\mathrm{live}} I_3$ – proof techniques required
- $\mu X.\, [\![P \mid \mathsf{co}\ k \triangleright_k X]\!] \eqsim_{\mathrm{live}} P$, for any P – proof techniques rqd

Proof techniques:

Require characterisations using “traces” and “refusals”

Outline

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \triangleright_k \ Q]\!]^\circ$

$$a.c.\mathbb{0} \qquad\qquad | \qquad\qquad [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \triangleright_k \ e]\!]$$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \rhd_k Q]\!]^\circ$

$$a.c.\mathbb{0} \quad | \quad [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \triangleright_k Q]\!]^\circ$

$$a.c.\mathbb{0} \quad | \quad [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \triangleright_k e]\!]$$

$$\xrightarrow{\texttt{emb}\ k} [\![a.c.\mathbb{0} \triangleright_k a.c.\mathbb{0}]\!]^\circ \quad | \quad \xrightarrow{\texttt{emb}\ k} [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \triangleright_k e]\!]$$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \rhd_k Q]\!]^\circ$

	$a.c.\mathbb{0}$	$\vert$		$[\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$
$\xrightarrow{\texttt{emb}\ k}$	$[\![a.c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ$	$\vert$	$\xrightarrow{\texttt{emb}\ k}$	$[\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$
$\xrightarrow{k(a)}$	$[\![c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ$	$\vert$	$\xrightarrow{k(\overline{a})}$	$[\![\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \rhd_k Q]\!]^\circ$

$$a.c.\mathbb{0} \quad | \quad [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$$

$$\xrightarrow{\texttt{emb}\ k} [\![a.c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ \quad | \quad \xrightarrow{\texttt{emb}\ k} [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$$

$$\xrightarrow{k(a)} [\![c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ \quad | \quad \xrightarrow{k(\overline{a})} [\![\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$$

$$\xrightarrow{k(c)} [\![\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ \quad | \quad \xrightarrow{k(\overline{c})} [\![\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!]$$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \rhd_k Q]\!]^\circ$

$$
\begin{array}{rl|rl}
 & a.c.\mathbb{0} & & [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{\texttt{emb}\ k} & [\![a.c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ & \xrightarrow{\texttt{emb}\ k} & [\![\overline{a}.\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{k(a)} & [\![c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ & \xrightarrow{k(\overline{a})} & [\![\overline{c}.\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{k(c)} & [\![\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ & \xrightarrow{k(\overline{c})} & [\![\mathrm{co}\ k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{\mathrm{co}\ k} & \mathbb{0} & \xrightarrow{\mathrm{co}\ k} & \overline{a}.\mathbb{0}
\end{array}
$$

Compositional Semantics

- The embedding rule is simple but entangles the processes
- We need to reason about the behaviour of $P|Q$ in terms of P and Q
- We introduce a compositional Labelled Transition System that uses *secondary transactions*: $[\![P \rhd_k Q]\!]^\circ$

$$
\begin{array}{lll|ll}
 & a.c.\mathbb{0} & & & [\![\overline{a}.\overline{c}.\text{co } k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{\texttt{emb } k} & [\![a.c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ & & \xrightarrow{\texttt{emb } k} & [\![\overline{a}.\overline{c}.\text{co } k \mid \overline{a}.\mathbb{0} \rhd_k e]\!] \\
\xrightarrow{k(a)} & [\![c.\mathbb{0} \rhd_k a.c.\mathbb{0}]\!]^\circ & & \xrightarrow{k(\overline{a})} & [\![\overline{a}.\overline{c}.\text{co } k \mid \mathbb{0} \rhd_k e]\!] \\
\xrightarrow{\texttt{ab } k} & a.c.\mathbb{0} & & \xrightarrow{\texttt{ab } k} & e
\end{array}
$$

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

$$\mathrm{Tr}_{\mathrm{clean}}\left(\llbracket a.c.\mathrm{co}\ k \ \triangleright_k\ e \rrbracket \right) = \{\epsilon,\ \mathbf{a}\,\mathbf{c},\ \mathbf{e}\}$$

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

$$\mathrm{Tr}_{\mathrm{clean}}\left(\llbracket a.c.\mathrm{co}\ k \triangleright_k e \rrbracket\right) = \{\epsilon,\ \mathbf{a\,c},\ \mathbf{e}\}$$

$$\mathrm{Tr}_{\mathrm{clean}}\left(\mu X. \llbracket a.c.\mathrm{co}\ k \triangleright_k X \rrbracket\right) = \{\epsilon,\ \mathbf{a\,c}\}$$

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

$$\mathrm{Tr}_{\mathrm{clean}}\left(\llbracket a.c.\mathrm{co}\ k \ \triangleright_k\ e \rrbracket\right) = \{\epsilon,\ \mathbf{a}\,\mathbf{c},\ \mathbf{e}\}$$

$$\mathrm{Tr}_{\mathrm{clean}}\left(\mu X.\llbracket a.c.\mathrm{co}\ k \ \triangleright_k\ X \rrbracket\right) = \{\epsilon,\ \mathbf{a}\,\mathbf{c}\}$$

- Set of clean traces not prefix closed: atomicity

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

$$\mathrm{Tr}_{\mathrm{clean}}\left(\llbracket a.c.co\ k \triangleright_k e \rrbracket\right) = \{\epsilon,\ \mathbf{a\,c},\ \mathbf{e}\}$$

$$\mathrm{Tr}_{\mathrm{clean}}\left(\mu X. \llbracket a.c.co\ k \triangleright_k X \rrbracket\right) = \{\epsilon,\ \mathbf{a\,c}\}$$

- Set of clean traces not prefix closed: atomicity

Characterisation of May Testing:

$$P \sqsubseteq_{\mathrm{may}} Q \quad \text{iff} \quad \mathrm{Tr}_{\mathrm{clean}}(P) \subseteq \mathrm{Tr}_{\mathrm{clean}}(Q)$$

Compositional Semantics: may-testing

The behaviour of processes in TransCCS can be understood by a *simple subset of the LTS traces*:

- where *all actions are eventually committed*
- that *ignore transactional annotations* on the traces

$$\mathrm{Tr}_{\mathrm{clean}}\left([\![a.c.co\ k \triangleright_k\ e]\!] \right) = \{\epsilon,\ \mathbf{a\,c},\ \mathbf{e}\}$$

$$\mathrm{Tr}_{\mathrm{clean}}\left(\mu X. [\![a.c.co\ k \triangleright_k\ X]\!] \right) = \{\epsilon,\ \mathbf{a\,c}\}$$

- Set of clean traces not prefix closed: atomicity

Characterisation of May Testing:

$$P \sqsubseteq_{\mathrm{may}} Q \qquad \text{iff} \qquad \mathrm{Tr}_{\mathrm{clean}}(P) \subseteq \mathrm{Tr}_{\mathrm{clean}}(Q)$$

- To understand the may-testing behaviour of P we only need to consider the clean traces $\mathrm{Tr}_{\mathrm{clean}}(P)$.

Compositional semantics: should-testing

Tree Failures: [Rensink-Vogler'07]

(t, Ref) where

- t is a clean trace
- Ref is a set of clean traces — can be non-prefixed closed

Compositional semantics: should-testing

Tree Failures: [Rensink-Vogler'07]

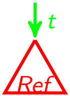
(t, Ref) where

- t is a clean trace
- Ref is a set of clean traces   can be non-prefixed closed

Tree failures of a process:

(t, Ref) is a tree failure of P when

$$\exists P'. \quad P \stackrel{t}{\Rightarrow}_{CL} P' \quad \text{and} \quad \mathcal{L}(P') \cap \mathit{Ref} = \emptyset$$



$$\mathcal{F}(P) = \{(t, \mathit{Ref}) \text{ tree failure of } P\}$$

Compositional semantics: should-testing

Tree Failures: [Rensink-Vogler'07]

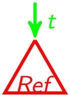
(t, Ref) where

- t is a clean trace
- Ref is a set of clean traces   can be non-prefixed closed

Tree failures of a process:

(t, Ref) is a tree failure of P when

$$\exists P'. \quad P \stackrel{t}{\Rightarrow}_{CL} P' \quad \text{and} \quad \mathcal{L}(P') \cap Ref = \emptyset$$



$$\mathcal{F}(P) = \{(t, Ref) \text{ tree failure of } P\}$$

Characterisation of should-testing:

$$S \sqsubseteq_{\text{live}} I \quad \text{iff} \quad \mathcal{F}(S) \supseteq \mathcal{F}(I)$$

Simple Examples

Let $S_{ab} = \mu X. [\![a.b.\mathsf{co}\ k \rhd_k X]\!]$ $\mathcal{L}(S_{ab}) = \{\epsilon, ab\}$

$$\mathcal{F}(S_{ab}) = \{(\epsilon, S \backslash ab), (ab, S) \mid S \subseteq A^*\}$$

Simple Examples

Let $S_{ab} = \mu X. [\![a.b.\mathsf{co}\ k \triangleright_k X]\!]$ $\mathcal{L}(S_{ab}) = \{\epsilon, ab\}$

$\mathcal{F}(S_{ab}) = \{(\epsilon, S \backslash ab), (ab, S) \mid S \subseteq A^*\}$

- $S_{ab} \eqsim_{\text{safe}} I_1 = [\![a.b.\mathsf{co}\ k \triangleright_k \mathbb{0}]\!]$ $\mathcal{L}(I_1) = \{\epsilon, ab\}$

  $S_{ab} \not\eqsim_{\text{live}} I_1$ $\mathcal{F}(I_1) = \{(\epsilon, S), (ab, S) \mid S \subseteq A^*\}$

Simple Examples

Let $S_{ab} = \mu X. \llbracket a.b.\mathsf{co}\ k \rhd_k X \rrbracket$ $\mathcal{L}(S_{ab}) = \{\epsilon, ab\}$

$$\mathcal{F}(S_{ab}) = \{(\epsilon, S \backslash ab), (ab, S) \mid S \subseteq A^*\}$$

- $S_{ab} \eqsim_{\text{safe}} I_1 = \llbracket a.b.\mathsf{co}\ k \rhd_k \mathbf{0} \rrbracket$ $\mathcal{L}(I_1) = \{\epsilon, ab\}$
  $S_{ab} \not\eqsim_{\text{live}} I_1$ $\mathcal{F}(I_1) = \{(\epsilon, S), (ab, S) \mid S \subseteq A^*\}$

- $S_{ab} \eqsim_{\text{safe}} I_2 = \mu X. \llbracket a.b.\mathsf{co}\ k + e \rhd_k X \rrbracket$ $\mathcal{L}(I_2) = \mathcal{L}(S_{ab})$
  $S_{ab} \eqsim_{\text{live}} I_2$ $\mathcal{F}(I_2) = \mathcal{F}(S_{ab})$

Summary

- TransCCS: a language for communicating/co-operative transactions
- simple reduction semantics using an *embedding* rule
- behavioural theories for preservation of
  - safety properties
  - liveness properties
- characterisations which allow
  - proofs of equivalences
  - equational laws

References:

- *Communicating Transactions*, Concur 2010
- *Liveness of Communicating Transactions*, APLAS 2010

Summary

- TransCCS: a language for communicating/co-operative transactions
- simple reduction semantics using an *embedding* rule
- behavioural theories for preservation of
  - safety properties
  - liveness properties
- characterisations which allow
  - proofs of equivalences
  - equational laws

References:

- *Communicating Transactions*, Concur 2010
- *Liveness of Communicating Transactions*, APLAS 2010

Future work:

- Reference implementation
- Extension to Haskell
- PhD Scholarship position funded by Microsoft Research, UK

THANK YOU!