Formal Modeling, Verification and Refinement of Long Running Transactions

Ji Wang

National Laboratory for Parallel and Distributed Processing, Changsha, China

Joint work with Zhenbang Chen and Zhiming Liu

Agenda

- Background and motivation
- Compensating CSP (cCSP)
- Non-determinism and deadlock
- Livelock and refinement
- Algebraic laws
- Conclusion and next step

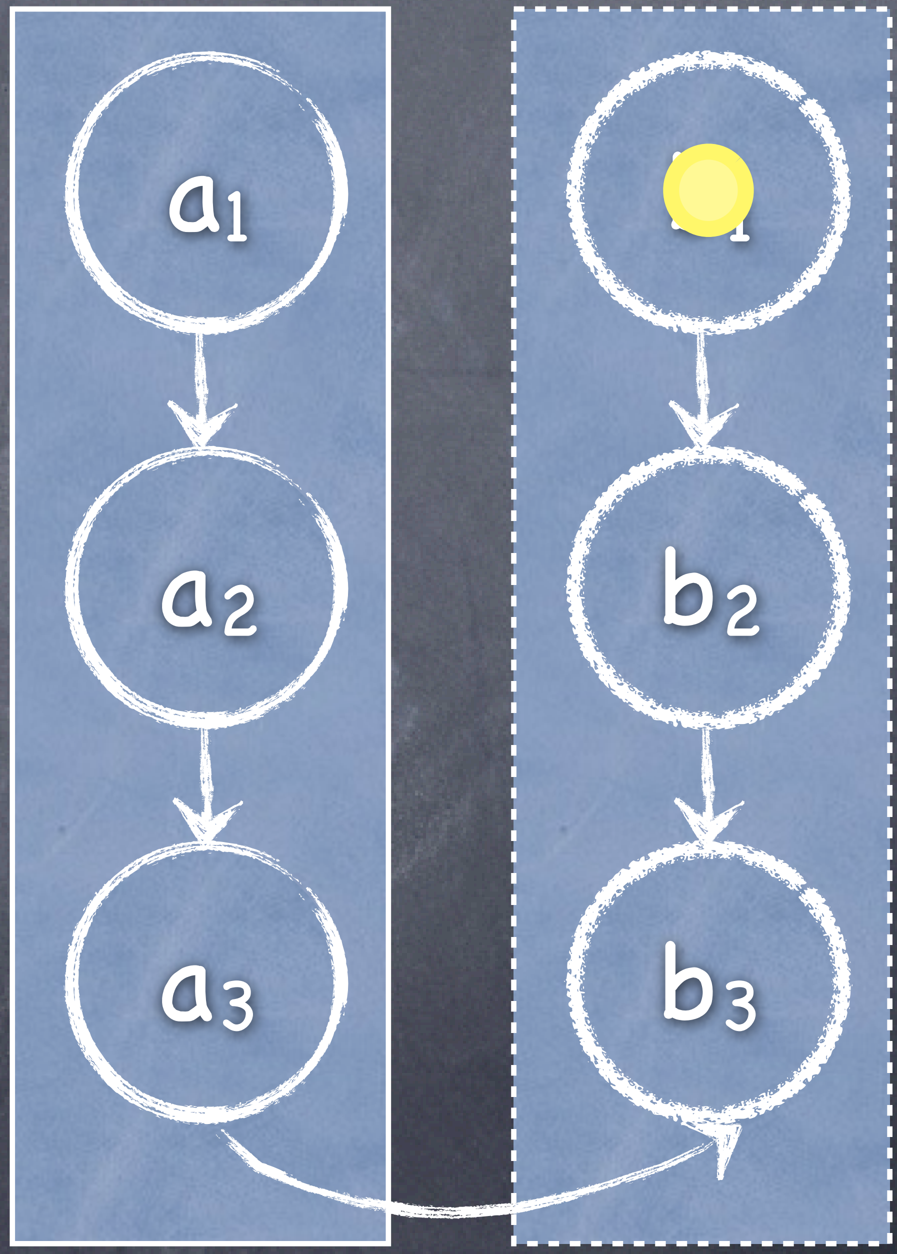
Long-Running Transactions

- Database
  - Long-lived transactions
  - Small ACID transactions
- SAGAS
  - 1987, SIGMOD
- Compensation

Compensation

In Database

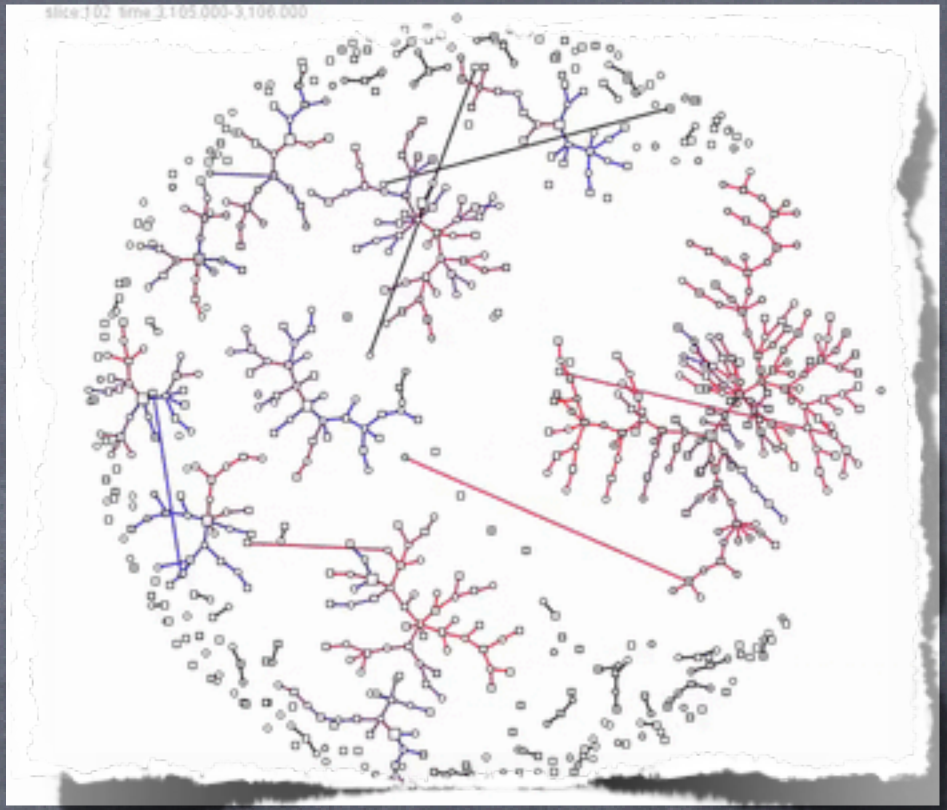
- An activity has its compensation activity
- In case of a failure, use compensations
- Atomicity and consistency



Error -> Failure

In Distributed Computing

- World wide distributed organizations
- Coordinate to accomplish a task

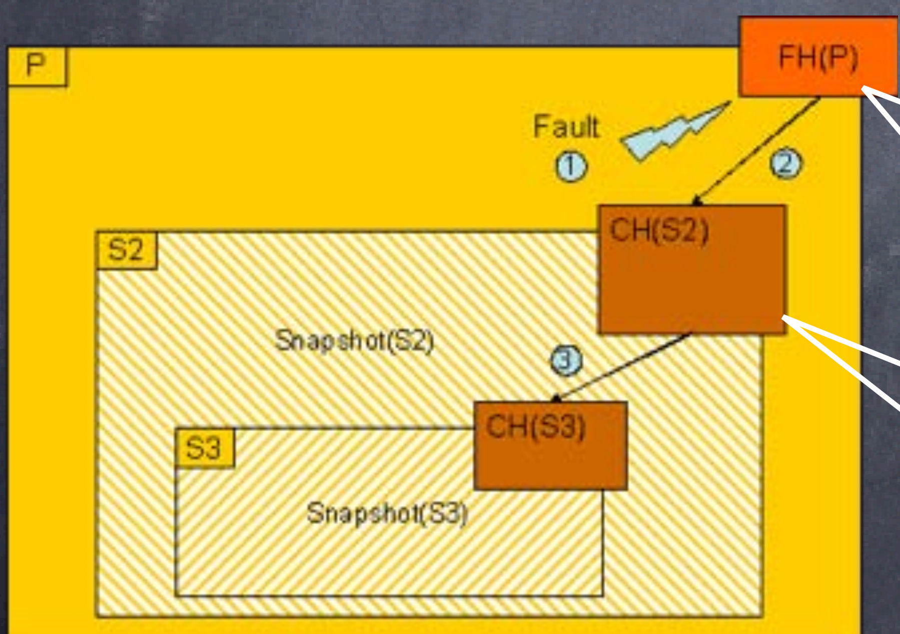


Long Running Transactions

How to ensure consistency in case of a failure?

Orchestration Programming in SOC

- WS-BPEL
  - Compensation based fault handling
  - Flexible recovery mechanisms for LRTs



Fault Handler

Compensation Handler

Ensure an acceptive consistency of composite Web Services

WS-BPEL 2.0, OASIS Standard, 11 April 2007

Orchestration Programming in SOC

- WS-BPEL OASIS
  - Compensation based fault handling
  - Flexible recovery mechanisms for LRTs
- Formal languages
  - cCSP, StAC, SAGAs Calculi, etc.

Formal Modeling and Verification

- Modeling
  - Rigorous semantic foundation
  - Formal semantics for industrial languages
  - Basis for verification
- Verification
  - Ensure the correctness of LRTs
  - Improve the reliability of LRT designs

Compensating CSP (cCSP)

Michael Butler, C.A.R. Hoare and Carla Ferreira. A Trace Semantics for Long-Running Transactions. **25 Years Communicating Sequential Processes**, LNCS 3525, 2004.

Compensating CSP (cCSP)

- Process language
  - CSP extension for modeling LRTs
  - Basic operators
- Two types of processes
  - Standard & Compensable
- Terminated trace semantics

cCSP Syntax and Example

$$P ::= a \mid P;P \mid P \Box P \mid P \| P \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2) ; \mathbf{throww}]$

cCSP Syntax and Example

$$P ::= a \mid P;P \mid P \Box P \mid P \| P \mid P \triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

Example $[(a_1 \div b_1; a_2 \div b_2)\ ;\ \textbf{throww}]$

a_1

b_1

cCSP Syntax and Example

$$P \quad ::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2) \ ; \ \mathbf{throww}]$

a_1 a_2

b_1 b_2

cCSP Syntax and Example

$$P \quad ::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2) \; ; \; \mathbf{throww}]$

$a_1 \quad a_2$ ☹

$b_1 \quad b_2$

cCSP Syntax and Example

$$P \quad ::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2)\ ;\ \mathbf{throww}]$

a_1 a_2 ☹ b_2

b_1

cCSP Syntax and Example

$$P ::= a \mid P;P \mid P\Box P \mid P\|P \mid P \triangleright P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP\Box PP \mid PP\|PP \mid PP \boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2)$; **throww**$]$

$a_1 \quad a_2 \quad$ ☹ $\quad b_2 \quad b_1$

Terminated Trace Semantics

$$P \quad ::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$T(a) =_{def} \{\langle a, \surd \rangle\}$

$T(\textbf{skip}) =_{def} \{\langle \surd \rangle\}$ $T(\textbf{throw}) =_{def} \{\langle ! \rangle\}$ $T(\textbf{yield}) =_{def} \{\langle \surd \rangle, \langle ? \rangle\}$

$T(P;Q) =_{def} \{s_1 \; ; \; s_2 \mid s_1 \in T(P), \; s_2 \in T(Q)\}$

$T(P \| Q) =_{def} \{s \mid \exists \; s_1 \in T(P), \; s_2 \in T(Q), \; s \in s_1 \; \| \; s_2\}$

$T(P \triangleright Q) =_{def} \{s_1 \triangleright s_2 \mid s_1 \in T(P), \; s_2 \in T(Q)\}$

Examples

$$P \quad ::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

T(a) = {<a, √>}

T(a;b) = {<a, b, √>} T(a;**throw**;b) = {<a, !>}

T(a || b) = {<a, b, √>, <b, a, √>}

T((a;**throw**) || b) = {<a, b, !>, <b, a, !>}

T((a;**throw**) || (**yield**;b)) = {<a, !>, <b, a, !>, <a, b, !>}

T(a ▷ b) = {<a, √>} T((a;**throw**) ▷ b) = {<a, b, √>}

Terminated Trace Semantics

$$P ::= a \mid P;P \mid P\Box P \mid P\|P \mid P\triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP ::= P\div P \mid PP;PP \mid PP\Box PP \mid PP\|PP \mid PP\boxtimes PP \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$T(P\div Q) =_{def} \{s_1\div s_2 \mid s_1\in T(P),\ s_2\in T(Q)\}$

if $s_1 = t\hat{\ }\surd$, $s_1\div s_2 = (s_1, s_2)$, else $(s_1, \langle\surd\rangle)$

T(**skipp**) $=_{def}$ **skip** ÷ **skip**

T(**throww**) $=_{def}$ **throw** ÷ **skip**

T(**yieldd**) $=_{def}$ **yield** ÷ **skip**

Examples

$T(a\div b) = \{(\langle a,\surd\rangle, \langle b,\surd\rangle)\}$

$T((a;\textbf{throw})\div b) = \{(\langle a,!\rangle, \langle\surd\rangle)\}$

Terminated Trace Semantics

$$P \quad ::= \quad a \mid P;P \mid P\Box P \mid P\|P \mid P\triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P\div P \mid PP;PP \mid PP\Box PP \mid PP\|PP \mid PP\boxtimes PP \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$T(PP;QQ) =_{def} \{(p, p') ; (q, q') \mid (p, p') \in T(P), (q, q') \in T(Q)\}$

if $p = t\hat{\ }\surd$, $(p, p') ; (q, q') = (p;q,\ q';p')$,
else, $(p, p') ; (q, q') = (p, p')$,

$T(a_1 \div b_1; a_2 \div b_2) = \{(\langle a_1, a_2, \surd\rangle, \langle b_2, b_1, \surd\rangle)\}$

Terminated Trace Semantics

$$P ::= a \mid P;P \mid P\Box P \mid P\|P \mid P\triangleright P \mid [PP] \mid \mathbf{skip} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP ::= P\div P \mid PP;PP \mid PP\Box PP \mid PP\|PP \mid PP\boxtimes PP \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$T([PP]) =_{def} \{s_1\hat{\ }s_2 \mid (s_1\hat{\ }!, s_2) \in T(PP)\} \cup \{s_1\hat{\ }\surd \mid (s_1\hat{\ }\surd, s_2) \in T(PP)\}$

Examples

$T(a\div b) = \{(\langle a, \surd\rangle, \langle b, \surd\rangle)\}$ $T([a\div b]) = \{\langle a, \surd\rangle\}$

$T(a\div b;\mathbf{throww}) = \{(\langle a, !\rangle, \langle b, \surd\rangle)\}$ $T([a\div b;\mathbf{throww}]) = \{\langle a, b, \surd\rangle\}$

Semantics Example

$$
\begin{aligned}
P \quad &::= \quad a \mid P;P \mid P \Box P \mid P \| P \mid P \triangleright P \mid [PP] \mid \textbf{skip} \mid \\
&\quad\;\; \textbf{throw} \mid \textbf{yield} \\
PP \quad &::= \quad P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \\
&\quad\;\; \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}
\end{aligned}
$$

Example $[(a_1 \div b_1 ; a_2 \div b_2) \; ; \; \textbf{throww}]$

$a_1 \quad a_2 \quad$ ☹ $\quad b_2 \quad b_1$

Trace Semantics: $\{\langle a_1, a_2, b_2, b_1, \surd \rangle\}$

Theoretical Issues of cCSP

$$
\begin{array}{lll}
P & ::= & a \mid P;P \mid P \Box P \mid P \| P \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \\
 & & \mathbf{throw} \mid \mathbf{yield} \\
 & & \\
PP & ::= & P \div P \mid PP;PP \mid PP \Box PP \mid PP \| PP \mid PP \boxtimes PP \mid \\
 & & \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}
\end{array}
$$

- Concurrent systems
  - Non-determinism & Deadlock & Livelock
- Reason
  - Trace semantics, no synchronization, no recursion
- Refinement

Non-determinism and Deadlock

Zhenbang Chen and Zhiming Liu. An Extended cCSP with Stable Failures Semantics. **7th International Colloquium on Theoretical Aspects of Computing (ICTAC'10)**, LNCS 6255, 2010.

Non-determinism & Deadlock

- Extend the syntax of cCSP
  - Internal and external choices
  - Synchronization, hiding and renaming

$$
\begin{aligned}
P \quad ::= \quad & a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \parallel_X P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \\
& \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}
\end{aligned}
$$

$$
\begin{aligned}
PP \quad ::= \quad & P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \parallel_X PP \mid PP \setminus X \mid \\
& PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}
\end{aligned}
$$

Non-determinism & Deadlock

- Extend the syntax of cCSP
  - Internal and external choices
  - Synchronization, hiding and renaming
- A stable failures semantics
  - Use refusals to model deadlocks

Basic Idea of a Failure-based Semantics

A failure (s, X)

- s: One trace s that a process can execute
- X: The set of the events that the process refuses to perform after executing s

a ; b

- refuse to execute any event except a
- refuse to execute any event except b
- refuse to execute any event except √
- refuse to execute any event finally

$\{(\langle\rangle, X) \mid a \notin X\} \cup \{(\langle a \rangle, X) \mid b \notin X\} \cup \{(\langle a, b \rangle, X) \mid \surd \notin X\} \cup \{(\langle a, b, \surd \rangle, X) \mid X \subseteq \Sigma\}$

A process **deadlocks** if it refuses to perform any event after executing a **non-terminated** trace

Semantic Models

- Standard processes

$$[\![P]\!] = (T, F)$$

Trace set, $T_s(P)$

Failure set, $F_s(P)$

- Compensable processes

$$[\![PP]\!] = (T, F, C)$$

Forward Trace set, $T^c(PP)$

Forward Failure set, $F^c(PP)$

Compensation Set, $C(PP)$, (s, T, F)

Semantics (1)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$T_s(a)=\{\langle\rangle, \langle a\rangle, \langle a,\surd\rangle\}$

$F_s(a)=\{(\langle\rangle,X) \mid a \notin X\} \cup \{(\langle a\rangle, X) \mid \surd \notin X\} \cup \{(\langle a,\surd\rangle, X)\}$
where $X \subseteq \Sigma \cup \{!,?,\surd\}$

$T_s(\textbf{stop})=\{\langle\rangle\}$

$F_s(\textbf{stop})=\{(\langle\rangle,X) \mid X \subseteq \Sigma \cup \{!,?,\surd\}\}$

Semantics (2) - Internal and External Choices

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \triangleright P \mid [PP] \mid \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Difference is at the beginning

1. Internal choice will refuse an event if any sub process can refuse it
2. External choice will refuse an event if both sub processes can refuse it

Semantics (2) - Internal and External Choices

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\quad \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$\quad PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$T_s(P \sqcap Q) = T_s(P) \cup T_s(Q)$ $\quad$ $T_s(P \Box Q) = T_s(P) \cup T_s(Q)$

$F_s(P \sqcap Q) = F_s(P) \cup F_s(Q)$

$F_s(P \Box Q) = \{(\langle\rangle, X) \mid (\langle\rangle, X) \in F_s(P) \cap F_s(Q)\}$ ...

Semantics (2) - Internal and External Choices

$$P ::= a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$\Sigma = \{a, b\}$

$F_s(a \sqcap b)=\{(\langle\rangle,X) \mid X \subseteq \{b,!,?,\surd\}\} \cup \{(\langle\rangle,X) \mid X \subseteq \{a,!,?,\surd\}\}$ ...

$F_s(a \Box b)=\{(\langle\rangle,X) \mid X \subseteq \{!,?,\surd\}\}$...

$T_s(a \Box b)=\{\langle\rangle, \langle a\rangle, \langle a,\surd\rangle, \langle b\rangle, \langle b,\surd\rangle\} = T_s(a \sqcap b)$

Semantics (3) - Synchronization

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

1. Parallel composition with synchronization on X can refuse an event out of X if both sub processes can refuse it
2. Parallel composition with synchronization on X can refuse an event in X if any sub process can refuse it

Semantics (3) - Synchronization

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$F_s(P \underset{X}{\parallel} Q) = \{(s, X_1 \cup X_2) \mid \exists\ (s_1, X_1) \in F_s(P), (s_2, X_2) \in F_s(Q),$
$X_1 \setminus (X \cup W) = X_2 \setminus (X \cup W) \wedge s \in s_1 \underset{X}{\parallel} s_2 \}$ ...

where $W = \{!,?,\surd\}$

Semantics (3) - How to have a deadlock

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

No synchronization, no deadlock

$F_s(a)=\{(\langle\rangle,X) \mid a \notin X\} \cup \{(\langle a\rangle, X) \mid \checkmark \notin X\} \cup \{(\langle a,\checkmark\rangle, X)\}$

$F_s(b)=\{(\langle\rangle,X) \mid b \notin X\} \cup \{(\langle b\rangle, X) \mid \checkmark \notin X\} \cup \{(\langle b,\checkmark\rangle, X)\}$

$F_s(a \underset{\{a,b\}}{\parallel} b)$ is $\{(\langle\rangle,X) \mid X \subseteq \{a,b,!,?,\checkmark\}\}$, i.e. F_s(**stop**)

Semantics (4)

$$P \;::=\; a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \;::=\; P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$[\![a \div b]\!] = (T_s(a), F_s(a), \{ (<a,\surd>, T_s(b), F_s(b)) \})$

$C((a \sqcap (a;throw)) \div b) = \{ (<a,\surd>, T_s(b), F_s(b)),$
$(<a,!>, T_s(\mathbf{skip}), F_s(\mathbf{skip})) \}$

$[\![[a \div b]]\!] = (T_s(a), F_s(a))$

Semantic (5)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$$[\![PP \sqcap QQ]\!] =_{def} (T_1 \cup T_2, F_1 \cup F_2, C_1 \cup C_2)$$

$$[\![PP \Box QQ]\!] =_{def} (T_s(PP_f \Box QQ_f), F_s(PP_f \Box QQ_f), C_1 \cup C_2)$$

Trace set function of P

Failure set function of P

Semantics (5) - Example

$$P ::= a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

Example $[(a_1 \div b_1 \sqcap a_2 \div b_2)\ ;\ \textbf{throww}]$

$(a_1;b_1) \sqcap (a_2;b_2)$

Semantics (6)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

$[\![PP;QQ]\!] =_{def} (T_s(PP_f;QQ_f), F_s(PP_f;QQ_f), C)$

$C =_{def} \{(s, T, F) \mid \exists\ (s_1, PP_c) \in C(PP), (s_2, QQ_c) \in C(QQ),$
$(s_1 = t\hat{}\checkmark \wedge s = t\hat{}s_2 \wedge T = T_s(QQ_c ; PP_c) \wedge F = F_s(QQ_c ; PP_c)) \vee$
$(s_1 \neq t\hat{}\checkmark \wedge s = s_1 \wedge T = T_s(PP_c) \wedge F = F_s(PP_c))\}$

Semantics (6) - Example

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid$$
$$PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2)\ ;\ \mathbf{throww}]$

$a_1 \quad a_2 \quad$ ☹ $\quad b_2 \quad b_1 \quad a_1 ; a_2 ; b_2 ; b_1$

Semantics (6) - Example

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \triangleright P \mid [PP] \mid \textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

Example $[(a_1 \div b_1 ; a_2 \div b_2)$; **throww**]

a_1 a_2 ☹ b_2 b_1 $a_1 ; a_2 ; b_2 ; b_1$

$[\![a_1 \div b_1 ; a_2 \div b_2]\!] = (T_s(a_1;a_2), F_s(a_1;a_2), \{(\langle a_1, a_2, \checkmark\rangle, T_s(b_2;b_1), F_s(b_2;b_1))\})$

Semantics (7)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd}$$

$$[\![PP \underset{X}{\parallel} QQ]\!] =_{def} (T_s(PP_f \underset{X}{\parallel} QQ_f),\ F_s(PP_f \underset{X}{\parallel} QQ_f),\ C)$$

$$C =_{def} \{(s, T, F) \mid \exists\ (s_1, PP_c) \in C(PP),\ (s_2, QQ_c) \in C(QQ),\ s \in (s_1 \underset{X}{\parallel} s_2) \wedge T = T_s(PP_c \underset{X}{\parallel} QQ_c) \wedge F = F_s(PP_c \underset{X}{\parallel} QQ_c)\}$$

Semantics (7) - Example

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \triangleright P \mid [PP] \mid \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a_1 \div b_1 \underset{\{a_1,a_2\}}{\parallel} a_2 \div b_2)]$

$a_1 \underset{\{a_1,a_2\}}{\parallel} a_2$

$b_1 \quad b_2$

Deadlock!!

Semantics: $\{(\langle\rangle, X) \mid X \subseteq \Sigma\}$

Semantics (7) - Example

$$P ::= a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \triangleright P \mid [PP] \mid \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield}$$

$$PP ::= P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd}$$

Example $[(a \div b_1 \underset{\{a\}}{\parallel} a \div b_2) \; ; \; \mathbf{throww}]$

Summary Until Now

- An extension to cCSP
  - Non-determinism
  - Synchronized parallel composition
- A new semantic model for the extended cCSP
  - Non-determinism and deadlock modeling

Zhenbang Chen and Zhiming Liu. An Extended cCSP with Stable Failures Semantics. **7th International Colloquium on Theoretical Aspects of Computing (ICTAC'10)**, LNCS 6255, 2010.

Livelock and Refinement

Zhengbang Chen, Zhiming Liu and Ji Wang. Failure-Divergence Refinement of Compensating Communicating Processes. **17th International Symposium on Formal Methods (FM'11)**, LNCS 6664, 2011.

Zhengbang Chen, Zhiming Liu and Ji Wang. Failure-Divergence Semantics and Refinement of Long Running Transactions. **Theoretical Computer Science (TCS)**, 2012

Livelock & Refinement

- No recursion
  - Cannot model divergence, i.e. livelock
  - Hard for a denotational semantics
- Refinement is hard to define w.r.t. the stable failures model
  - Design by refinement for LRTs

Livelock & Refinement

- Extend language
  - Recursive processes
  - Speculative choice

$$
\begin{array}{lll}
P & ::= & a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \\
 & & \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p) \\
PP & ::= & P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid \\
 & & PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)
\end{array}
$$

Livelock & Refinement

- Extend language
  - Recursive processes
  - Speculative choice
- A failure-divergence semantics
  - Support recursion interpretation
  - Use recursions to model livelocks
  - Refinement definition

Basic Idea of a Failure-Divergence Semantics

A failure is (s, X)

One trace s that a process can execute

The set of the events that the process refuses to perform after executing s

A divergence is a trace s

1. The process enters a chaos state after executing s
2. The process is totally unpredictable, i.e. it can perform or refuse any event
3. Use **DIV** to denote the process that diverges immediately

Basic Idea of a Failure-Divergence Semantics

- A divergence is suffix closed

$$s \in D(P) \cap \Sigma^* \Rightarrow s\hat{\ }t \in D(P)$$

- A divergent process can refuse any event

$$s \in D(P) \Rightarrow (s, X) \in F(P), \text{ where } X \subseteq \Sigma \cup \{!, ?, \surd\}$$

- A terminated divergence must be generated by a non-terminated divergence

$$s\hat{\ }w \in D(P) \Rightarrow s \in D(P), \text{ where } w \subseteq \{!, ?, \surd\}$$

Standard Processes

- Semantic model

$$[\![P]\!] = (F, D)$$

Failure set, F(P)

Divergence set, D(P)

- Examples for divergence sets

D(a) = {}

D(DIV) contains <>, i.e. D(DIV) contains any traces

- Refinement of standard processes

$$P_1 \sqsubseteq P_2 =_{def} F_1 \supseteq F_2 \wedge D_1 \supseteq D_2$$

How to have a livelock

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \boxed{\mu\, p.F(p)}$$
$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

No recursion, no divergence

(μ p. (a ; p)) executes a infinitely

(μ p. (a ; p)) \ {a} is equal to DIV

Semantic Models

- Standard processes

$$[\![P]\!] = (F, D)$$

Failure set, F(P)

Divergence set, D(P)

- Compensable processes

$$\begin{aligned} PP \quad ::= \quad & P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid \\ & PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp) \end{aligned}$$

$$[\![PP]\!] = ???$$

Ways to Go

Build a new model

Find it out

Based on an existing one

Stable failures model

Problems

- We failed on the first way
  - Compensable processes

⟦PP⟧ = (T, F, C)

(s, T, F)

Extension

⟦PP⟧ = (F, D, C)

(s, F, D)

Complete lattice or CPO?

Refinement order?

Working Process and Final Result

- Search and tradeoff
  - Semantic model and algebraic laws
  - Refinement and fixed-point theory

⟦PP⟧ ???

↓ ☺

(F, D, F^c, D^c)

(s, s′, X) (s, s′)

Order and Properties

$$(F_1, D_1, F^c{}_1, D^c{}_1) \sqsubseteq_c (F_2, D_2, F^c{}_2, D^c{}_2)$$

$$F_1 \supseteq F_2 \;\wedge\; D_1 \supseteq D_2 \;\wedge\; F^c{}_1 \supseteq F^c{}_2 \;\wedge\; D^c{}_1 \supseteq D^c{}_2$$

- The order is easy to understand
- The domain is a CPO w.r.t. the order
- The order is natural for refinement

Semantic Models

- Standard processes

$$[\![P]\!] = (F, D)$$

Failure set, $F(P)$

Divergence set, $D(P)$

- Compensable processes

$$[\![PP]\!] = (F, D, F^c, D^c)$$

Forward Failure set, $F_f(PP)$

Forward Divergence set, $D_f(PP)$

Semantic Models

- Standard processes

$$[\![P]\!] = (F, D)$$

Failure set, F(P)

Divergence set, D(P)

- Compensable processes

$$[\![PP]\!] = (F, D, F^c, D^c)$$

Compensation Failure set, $F^c(PP)$

Compensation Divergence set, $D^c(PP)$

Semantics (1)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

$[\![a \div b]\!] = (F(a), \{\}, \{\langle a,\checkmark\rangle\} \times F(b), \{\})$

$F^c((a \sqcap (a;\mathbf{throw})) \div b) = \{\langle a,\checkmark\rangle\} \times F(b) \cup \{\langle a,!\rangle\} \times F(\mathbf{skip})$

Semantics (2)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

The operators are continuous

Least fixed-point semantics

$$[\![\ \mu\ pp.\ FF(pp)\ ]\!] = \bigsqcup\{FF^n(\mathbf{DIV} \div \mathbf{DIV}) \mid n \in N\}$$

Semantics (2) - Example

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples [μ pp. (a÷b;pp) ; **throww**]

a a a a a
b b b b b ...

Not terminated

⟦ μ pp. (a÷b;pp)⟧ = (⟦ μ p. (a;p)⟧ ,{},{})

Semantics (3)

$$
\begin{array}{lcl}
P & ::= & a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \\
& & \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p) \\
PP & ::= & P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid \\
& & PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)
\end{array}
$$

Examples $[(a_1 \div b_1 \boxtimes (a_2 \div b_2; \mathbf{throww})); \mathbf{throww}]$

$a_1 \parallel a_2 ; \mathbf{throw}$ b_2 ☹ b_1

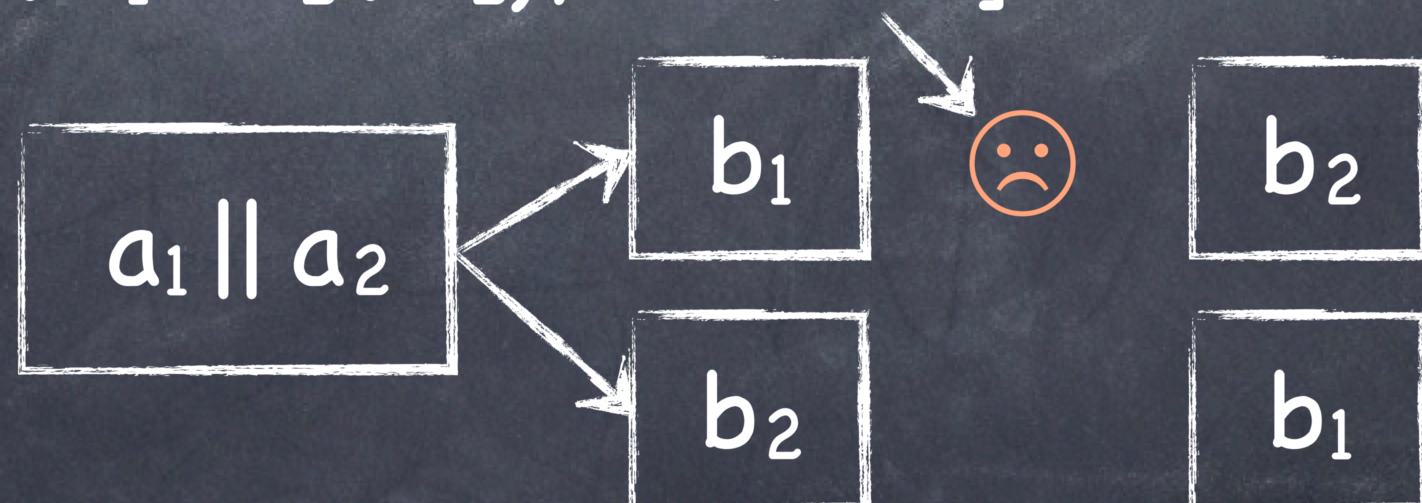
$(a_1 \parallel a_2) ; b_2 ; b_1$

Semantics (3)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\quad \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$\quad PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples $[(a_1 \div b_1 \boxtimes a_2 \div b_2);\mathbf{throww}]$



Semantics (3)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \textbf{throww}) \boxtimes (a_2 \div b_2; \textbf{throww})) \parallel (a_3 \div b_3)]$

Semantics (3)

$$
\begin{array}{lcl}
P & ::= & a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid \\
 & & \mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p) \\
PP & ::= & P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid \\
 & & PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)
\end{array}
$$

Examples

$[((a_1 \div b_1; \mathbf{throww}) \boxtimes (a_2 \div b_2; \mathbf{throww})) \parallel (a_3 \div b_3)]$

a_1 ; **throw** || a_2 ; **throw**

b_1 b_2

Semantics (3)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \triangleright P \mid [PP] \mid$$
$$\textbf{skip} \mid \textbf{stop} \mid \textbf{throw} \mid \textbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \textbf{skipp} \mid \textbf{throww} \mid \textbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \textbf{throww}) \boxtimes (a_2 \div b_2; \textbf{throww})) \parallel (a_3 \div b_3)]$

$(a_1 \parallel a_2)$; **throw**

b_1 b_2

Semantics (3)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \mathbf{throww}) \boxtimes (a_2 \div b_2; \mathbf{throww})) \parallel (a_3 \div b_3)]$

$(a_1 \parallel a_2)$; **throw** $\parallel$ a_3

b_1 b_2 b_3

Semantics (3)

$$P ::= a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP ::= P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \mathbf{throww}) \boxtimes (a_2 \div b_2; \mathbf{throww})) \parallel (a_3 \div b_3)]$

$(a_1 \parallel a_2 \parallel a_3)$ ☹

b_1 b_2 b_3

Semantics (3)

$$P ::= a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP ::= P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \textbf{throww}) \boxtimes (a_2 \div b_2; \textbf{throww})) \parallel (a_3 \div b_3)]$

$(a_1 \parallel a_2 \parallel a_3)$

☹ $b_1 \parallel b_2 \parallel b_3$

Semantics (3)

$$P \quad ::= \quad a \mid P;P \mid P \sqcap P \mid P \Box P \mid P \underset{X}{\parallel} P \mid P \setminus X \mid P[\![R]\!] \mid P \rhd P \mid [PP] \mid$$
$$\mathbf{skip} \mid \mathbf{stop} \mid \mathbf{throw} \mid \mathbf{yield} \mid \mu\, p.F(p)$$

$$PP \quad ::= \quad P \div P \mid PP;PP \mid PP \sqcap PP \mid PP \Box PP \mid PP \underset{X}{\parallel} PP \mid PP \boxtimes PP \mid$$
$$PP \setminus X \mid PP[\![R]\!] \mid \mathbf{skipp} \mid \mathbf{throww} \mid \mathbf{yieldd} \mid \mu\, pp.FF(pp)$$

Examples

$[((a_1 \div b_1; \mathbf{throww}) \boxtimes (a_2 \div b_2; \mathbf{throww})) \parallel (a_3 \div b_3)]$

$(a_1 \parallel a_2 \parallel a_3)$

☹ $b_1 \parallel b_2 \parallel b_3$

$(a_1 \parallel a_2 \parallel a_3);(b_1 \parallel b_2 \parallel b_3)$

Livelock & Refinement

- All basic concurrent features
  - Divergence for livelock
- A failure-divergence semantics
  - Fixed-point theory
- Refinement w.r.t the semantics
  - Non-determinism

Zhengbang Chen, Zhiming Liu and Ji Wang. Failure-Divergence Semantics and Refinement of Long Running Transactions. **Theoretical Computer Science (TCS)**, 2012

Algebraic Laws

Zhengbang Chen, Zhiming Liu and Ji Wang. Failure-Divergence Semantics and Refinement of Long Running Transactions. **Theoretical Computer Science (TCS)**, 2012

Algebraic Laws of Standard Processes

Some Still Valid CSP laws

- Idempotence

$P \sqcap P = P$

$P \,\square\, P = P$

- Units and zeros

skip ; $P = P$

stop $\square\; P = P$

$P \setminus \{\} = P$

stop ; $P =$ **stop**

DIV $\sqcap\; P =$ **DIV**

- Refinement

$P \sqcap Q \sqsubseteq P$ $\qquad$ $DIV \sqsubseteq P$

Exception Handling

- Units and zeros

$$\textbf{throw} \rhd P = P$$
$$P \rhd \textbf{throw} = P$$
$$\textbf{skip} \rhd P = \textbf{skip}$$

$$\textbf{throw} \; ; P = \textbf{throw}$$
$$\textbf{yield} \rhd P = \textbf{yield}$$
$$\textbf{stop} \rhd P = \textbf{stop}$$

- Distribution and association

$$P \rhd (Q \sqcap R) = (P \rhd Q) \sqcap (P \rhd R)$$
$$(P \sqcap Q) \rhd R = (P \rhd R) \sqcap (Q \rhd R)$$
$$P \rhd (Q \rhd R) = (P \rhd Q) \rhd R$$

Parallel Composition

- Unit and zeros

  throw $\|_{\times}$ **skip** = **throw**

  throw $\|_{\times}$ **yield** = **throw**

  P $\|$ **skip** = **P**

- If P does not terminate with a yield terminal event

  throw $\|$ P = P ; **throw**

  throw $\|$ (**yield** ; P) = **throw**$\sqcap$(P ; **throw**)

Algebraic Laws of Compensable Processes

Basic Algebraic Laws

- Units and zeros

$$\mathbf{skipp}\ ;\ PP = PP$$
$$PP\ ;\ \mathbf{skipp} = PP$$
$$\mathbf{throww}\ ;\ PP = \mathbf{throww}$$

- Distribution

$$[PP \sqcap QQ] = [PP] \sqcap [QQ]$$
$$P \div (Q \sqcap R) = (P \div Q) \sqcap (P \div R)$$
$$(P \div Q) \backslash X = (P \backslash X) \div (Q \backslash X)$$

Refinement Laws

$$PP \sqcap QQ \sqsubseteq_c PP$$

- Consistently related

$$PP_1 \sqsubseteq_c PP_2 \Rightarrow [PP_1] \sqsubseteq [PP_2]$$

- Reduction

$$Q_1 \sqsubseteq Q_2 \Rightarrow P \div Q_1 \sqsubseteq_c P \div Q_2$$

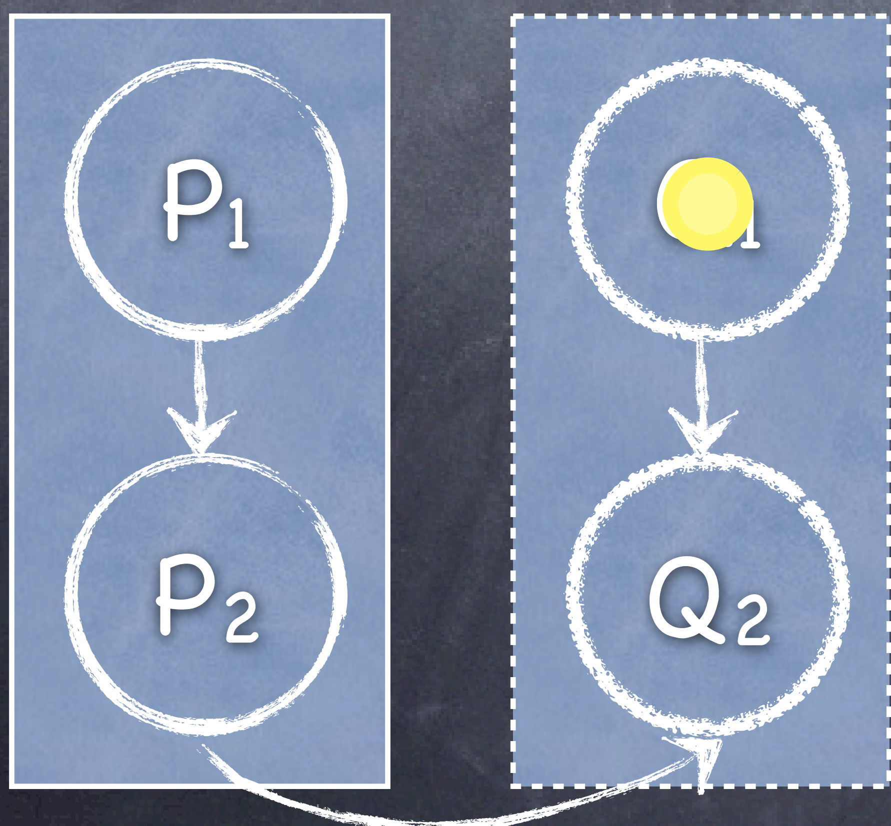
$$P_1 \sqsubseteq P_2 \Rightarrow P_1 \div Q \sqsubseteq_c P_2 \div Q$$

Compensation Laws (1)

- If P_1 and P_2 do not result in an exception

$$[P_1 \div Q_1 \ ; \ \textbf{throww}] = P_1 \ ; \ Q_1$$

$$[P_1 \div Q_1 \ ; \ P_2 \div Q_2 \ ; \ \textbf{throww}] = P_1 \ ; \ P_2 \ ; \ Q_2 \ ; \ Q_1$$



The laws are still valid when P_1 is YIELD

Compensation Laws (2)

- If all the standard processes terminate successfully and do not diverge

$$[(P \div Q) \| \textbf{throww}] = P \,;\, Q$$

$$P_1 \div Q_1 \underset{X}{\|} P_2 \div Q_2 = P_1 \underset{X}{\|} P_2 \div Q_1 \underset{X}{\|} Q_2$$

$$[(P_1 \div Q_1 \boxtimes P_2 \div Q_2) \,;\, \textbf{throww}] = (P_1 \| P_2) \,;\, ((Q_1 \,;\, Q_2) \sqcap (Q_2 \,;\, Q_1))$$

Interruption Laws

- If all the standard processes do not diverge and terminate successfully

[(**yieldd**;$P_1 \div Q_1$;**yieldd**;$P_2 \div Q_2$)||**throww**] =

skip $\sqcap$ $(P_1 ; Q_1)$ $\sqcap$ $(P_1 ; P_2 ; Q_2 ; Q_1)$

[(**yieldd**;$P_1 \div Q_1$)||(**yieldd**;$P_2 \div Q_2$)||**throww**] =

skip$\sqcap(P_1 ; Q_1)\sqcap(P_2 ; Q_2)\sqcap((P_1 || P_2);(Q_1 || Q_2))$

yieldd must be used to specify interruption places

Conclusion & Ongoing Work

- A semantic theory for LRTs
  - Non-determinism, deadlock and livelock
  - Design by refinement
  - Reasoning LRTs by algebraic laws
- Ongoing work
  - PAT based model checker for extended cCSP
  - Application of the theory, e.g., BPMN

End

Thank you!