

Failure-Divergence Semantics for Extended cCSP

Zhenbang Chen

zbchen@iist.unu.edu



United Nations
University

UNU-IIST

International Institute for
Software Technology



Outline

- Motivation
- Brief Introduction to cCSP
- Extended cCSP and Failure-Divergence Semantics
 - Standard Process
 - Compensating Process
- Related Work and Conclusion

Outline

- Motivation
- Brief Introduction to cCSP
- Extended cCSP and Failure-Divergence Semantics
 - Standard Process
 - Compensating Process
- Related Work and Conclusion

Motivation (1)

Long Running Transaction (LRT)

- LRT models attract attention recently because of the progress in Service-Oriented Computing (SOC)
- Coordination between different wide-spread communicating peers to ensure transaction features
- Transactions in SOC usually last for a long period, and involves interactions with different organizations
- Atomic transaction is strict for this scenario because of requirements such as isolation
- LRT can tackle this problem by using compensation mechanism

Motivation (2)

Industrial Languages

- WS-BPEL [AAA⁺06]
- XLANG [Tha01]
- ...

Formal Languages for LRT

- SAGAS [BMM05]
- StAC [BF04]
- cCSP [BHF04]
- ...

Motivation (3)

About cCSP

- Compensating CSP is a variant of CSP for modeling LRT
- Two kinds of processes in cCSP
 - Standard process
 - Compensating process

Limits of cCSP

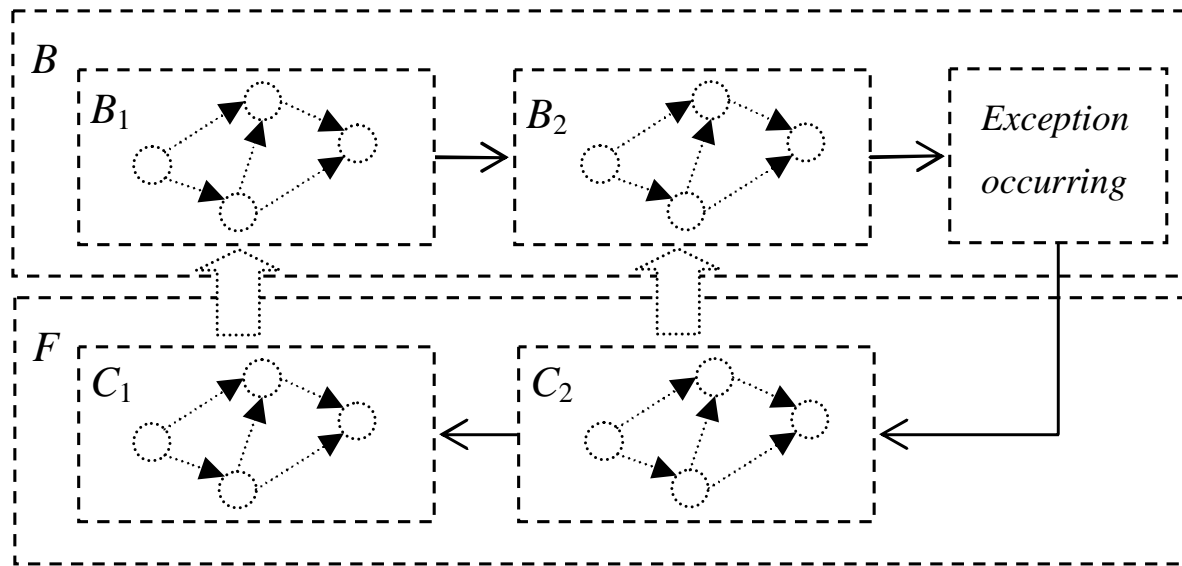
- The operators in cCSP are limited
- No discussion about the refinement

Outline

- Motivation
- **Brief Introduction to cCSP**
- Extended cCSP and Failure-Divergence Semantics
 - Standard Process
 - Compensating Process
- Related Work and Conclusion

Basic Ideas of cCSP

- The recovery method in cCSP is same to the backward recovery of SAGAS [GMS87]



- Parallel execution

Syntax of cCSP

P, Q	$::=$	A	PP, QQ	$::=$	$P \div Q$
		$P ; Q$			$PP ; QQ$
		$P \square Q$			$PP \square QQ$
		$P \parallel Q$			$PP \parallel QQ$
		$SKIP$			$SKIPP$
		$THROW$			$THROWW$
		$YIELD$			$YIELDD$
		$P \triangleright Q$			
		$[PP]$			

Semantics of cCSP

- Terminating trace semantics
- Two additional terminating events
 - $\Omega = \{\checkmark, !, ?\}$

Parallel Composition

$$p \hat{\langle} \omega_1 \rangle \parallel q \hat{\langle} \omega_2 \rangle = \{r \hat{\langle} \omega_1 \& \omega_2 \rangle \mid r \in (p \parallel q)\}$$

$$\mathcal{T}(P \parallel Q) = \{r \mid r \in (p \parallel q) \wedge p \in \mathcal{T}(P) \wedge q \in \mathcal{T}(Q)\}$$

where

ω_1	\checkmark	\checkmark	\checkmark	!	!	?
ω_1	\checkmark	?	!	!	?	?
$\omega_1 \& \omega_2$	\checkmark	?	!	!	!	?

Trace Semantics of Compensating Process (1)

Compensation Pair

$$p \div q = \begin{cases} (p, q) & p = p_1 \hat{\langle \checkmark \rangle} \\ (p, \checkmark) & p = p_1 \hat{\langle \omega \rangle} \wedge \omega \neq \checkmark \end{cases}$$

$$\mathcal{T}_c(P \div Q) = \{p \div q \mid p \in \mathcal{T}(P) \wedge q \in \mathcal{T}(Q)\} \cup \{(\langle ? \rangle, \langle \checkmark \rangle)\}$$

Sequential Composition

$$(p, p') ; (q, q') = \begin{cases} (p_1 \hat{q}, q' ; p') & p = p_1 \hat{\langle \checkmark \rangle} \\ (p, p') & p = p_1 \hat{\langle \omega \rangle} \wedge \omega \neq \checkmark \end{cases}$$

$$\mathcal{T}_c(PP ; QQ) = \{(p, p') ; (q, q') \mid (p, p') \in \mathcal{T}_c(PP) \wedge (q, q') \in \mathcal{T}_c(QQ)\}$$

Transaction Block

$$\mathcal{T}([PP]) = \{p \hat{p}' \mid (p \hat{\langle ! \rangle}, p') \in \mathcal{T}_c(PP)\} \cup \{p \hat{\langle \checkmark \rangle} \mid (p \hat{\langle \checkmark \rangle}, p') \in \mathcal{T}_c(PP)\}$$

Example

$$\begin{aligned} \textit{SellingTranaction} &= [\textit{ProcessSale}] \\ \textit{ProcessSale} &= \textit{DeductStore} \div \textit{RecoveryStore} ; \\ & ((\textit{TransferMoney} \div \textit{Return} ; \\ & \quad \textit{SKIP} \square \textit{THROW}) \parallel \\ & \quad \textit{ShipItem} \div \textit{ShipBack}) \end{aligned}$$

Example

$$\begin{aligned} \textit{SellingTranaction} &= [\textit{ProcessSale}] \\ \textit{ProcessSale} &= \textit{DeductStore} \div \textit{RecoveryStore} ; \\ & \quad ((\textit{TransferMoney} \div \textit{Return} ; \\ & \quad \quad \textit{SKIP} \square \textit{THROW}) \parallel \\ & \quad \textit{ShipItem} \div \textit{ShipBack}) \end{aligned}$$

DeductStore, TransferMoney, ShipItem

DeductStore, ShipItem, TransferMoney

DeductStore, TransferMoney, Return, RecoveryStore

DeductStore, TransferMoney, ShipItem, Return, ShipBack, RecoveryStore

DeductStore, TransferMoney, ShipItem, ShipBack, Return, RecoveryStore

DeductStore, ShipItem, TransferMoney, Return, ShipBack, RecoveryStore

DeductStore, ShipItem, TransferMoney, Return, ShipBack, RecoveryStore

Problems

- The following laws do not hold:
 - $PP; SKIPP = PP$
 - $[P \div P'] = P$
 - $[P \div P'; THROWW] = P; P'$
- Some operators are not provided:
 - internal choice
 - external choice
 - ...
- No study of refinement yet.

Outline

- Motivation
- Brief Introduction to cCSP
- **Extended cCSP and Failure-Divergence Semantics**
 - Standard Process
 - Compensating Process
- Related Work and Conclusion

Syntax of Extended cCSP

P, Q	$::=$	A	PP, QQ	$::=$	$P \div Q$
		$P ; Q$			$PP ; QQ$
		$P \sqcap Q$			$PP \sqcap QQ$
		$P \square Q$			$PP \square QQ$
		$P \underset{X}{\parallel} Q$			$PP \underset{X}{\parallel} QQ$
		$SKIP$			$SKIPP$
		$THROW$			$THROWW$
		$YIELD$			$YIELDD$
		$STOP$			$PP \setminus X$
		$P \setminus X$			$PP[R]$
		$P[R]$			
		$P \triangleright Q$			
		$[PP]$			

Limitation and Auxiliary Definitions

Limitation

- We do not consider recursion and infinite choices in the compensating process

Some Definitions

- $\Sigma^\Omega = \Sigma \cup \Omega$
- $\Sigma^{*O} = \Sigma^* \cup \{s \hat{\langle \omega \rangle} \mid s \in \Sigma^* \wedge \omega \in O\}$, where $O \subseteq \Omega$
- $\Sigma_O^* = \{s \hat{\langle \omega \rangle} \mid s \in \Sigma^* \wedge \omega \in O\}$, where $O \subseteq \Omega$

Outline

- Motivation
- Brief Introduction to cCSP
- Extended cCSP and Failure-Divergence Semantics
 - **Standard Process**
 - Compensating Process
- Related Work and Conclusion

Semantic Domain of the Standard Process

- Failure-divergence semantics (F, D)
- Semantic functions
 - Failure set function: $\mathcal{F}(P) : \mathcal{P} \rightarrow \mathbb{P}(\Sigma^{*\Omega} \times \mathbb{P}(\Sigma^\Omega))$
 - Divergence set function: $\mathcal{D}(P) : \mathcal{P} \rightarrow \mathbb{P}(\Sigma^{*\Omega})$
- Trace set function: $traces_{\perp}(P) = \{s \mid (s, \{\}) \in \mathcal{F}(P)\}$
- Modified axioms:

$$F_4 : s \hat{\langle \omega \rangle} \in trace_{\perp}(P) \Rightarrow (s, \Sigma^{\Omega} \setminus \{\omega\}) \in F, \text{ where } \omega \in \Omega$$

$$D_1 : s \in D \cap \Sigma^* \wedge t \in \Sigma^{*\Omega} \Rightarrow s \hat{t} \in D$$

$$D_3 : s \hat{\langle \omega \rangle} \in D \Rightarrow s \in D, \text{ where } \omega \in \Omega$$

- $P_1 \sqsubseteq P_2$ iff $\mathcal{F}(P_1) \supseteq \mathcal{F}(P_2)$ and $\mathcal{D}(P_1) \supseteq \mathcal{D}(P_2)$

Semantic of Basic Processes

$$\begin{aligned}\mathcal{F}(A) &= \{(\langle \rangle, X) \mid X \subseteq \Sigma^\Omega \wedge A \notin X\} \cup \\ &\quad \{(\langle A \rangle, X) \mid X \subseteq \Sigma^\Omega \wedge \checkmark \notin X\} \cup \\ &\quad \{(\langle A, \checkmark \rangle, X) \mid X \subseteq \Sigma^\Omega\}\end{aligned}$$

$$\begin{aligned}\mathcal{F}(THROW) &= \{(\langle \rangle, X) \mid X \subseteq \Sigma^\Omega \wedge ! \notin X\} \cup \\ &\quad \{(\langle ! \rangle, X) \mid X \subseteq \Sigma^\Omega\}\end{aligned}$$

$$\begin{aligned}\mathcal{F}(YIELD) &= \{(\langle \rangle, X) \mid X \subseteq \Sigma^\Omega \wedge ? \notin X\} \cup \\ &\quad \{(\langle ? \rangle, X) \mid X \subseteq \Sigma^\Omega\} \cup \\ &\quad \{(\langle \rangle, X) \mid X \subseteq \Sigma^\Omega \wedge \checkmark \notin X\} \cup \\ &\quad \{(\langle \checkmark \rangle, X) \mid X \subseteq \Sigma^\Omega\}\end{aligned}$$

Semantic of External Choice

- Need to consider exception and yield interruptions to ensure the axiom F_4

$$\mathcal{D}(P \square Q) = \mathcal{D}(P) \cup \mathcal{D}(Q)$$

$$\begin{aligned} \mathcal{F}(P \square Q) = & \{(\langle \rangle, X) \mid (\langle \rangle, X) \in \mathcal{F}(P) \cap \mathcal{F}(Q)\} \cup \\ & \{(s, X) \mid (s, X) \in \mathcal{F}(P) \cup \mathcal{F}(Q) \wedge s \neq \langle \rangle\} \cup \\ & \{(s, X) \mid s \in \mathcal{D}(P ; Q)\} \cup \\ & \{(\langle \rangle, X) \mid X \subseteq \Sigma^\Omega \setminus \{\omega\} \\ & \quad \wedge \langle \omega \rangle \in \text{traces}_\perp(P) \cup \text{traces}_\perp(Q) \wedge \omega \in \Omega\} \end{aligned}$$

$$SKIP \square YIELD = YIELD$$

$$SKIP \square THROW = SKIP \sqcap THROW$$

$$YIELD \square THROW = YIELD \sqcap THROW$$

Semantic of Sequential Composition

$$\begin{aligned}\mathcal{D}(P ; Q) &= \mathcal{D}(P) \cup \{s \hat{t} \mid s \hat{\langle \checkmark \rangle} \in \text{traces}_{\perp}(P) \wedge t \in \mathcal{D}(Q)\} \\ \mathcal{F}(P ; Q) &= \{(s, X) \mid s \in \Sigma^{*\{?,!\}} \wedge (s, X \cup \{\checkmark\}) \in \mathcal{F}(P)\} \cup \\ &\quad \{(s \hat{t}, X) \mid s \hat{\langle \checkmark \rangle} \in \text{traces}_{\perp}(P) \wedge (t, X) \in \mathcal{F}(Q)\} \cup \\ &\quad \{(s, X) \mid s \in \mathcal{D}(P ; Q) \wedge X \subseteq \Sigma^{\Omega}\}\end{aligned}$$

Laws:

$$\begin{aligned}\text{THROW} ; P &= \text{THROW} \\ \text{YIELD} ; \text{YIELD} &= \text{YIELD}\end{aligned}$$

Semantic of Parallel Composition - (1)

$$\mathcal{D}(P \parallel_X Q) = \{u \hat{v} \mid \exists s \in \text{traces}_\perp(P), t \in \text{traces}_\perp(Q) \bullet \\ u \in (s \parallel_X t) \cap \Sigma^* \wedge (s \in \mathcal{D}(P) \vee t \in \mathcal{D}(Q))\}$$

$$\mathcal{F}(P \parallel_X Q) = \{(u, E) \mid (u, E) \in (s, Y) \oplus_{P \parallel_X Q} (t, Z) \wedge \\ \exists s, t \bullet (s, Y) \in \mathcal{F}(P) \wedge (t, Z) \in \mathcal{F}(Q)\} \cup \\ \{(u, Y) \mid u \in \mathcal{D}(P \parallel_X Q)\}$$

Laws:

$$YIELD \parallel_X SKIP = YIELD$$

$$THROW \parallel_X SKIP = THROW$$

$$THROW \parallel_X YIELD = THROW$$

$$THROW \parallel_X THROW = THROW$$



Semantic of Parallel Composition - (2)

$$(s, Y) \oplus_{P \parallel_X Q} (t, Z) = \left\{ \begin{array}{l} \{(u, Y \cup Z) \mid Y \setminus (X \cup \Omega) = Z \setminus (X \cup \Omega) \wedge u \in s \parallel_X t\} \\ \quad \text{if } (s, Y \cup \Omega) \in \mathcal{F}(P) \vee (t, Z \cup \Omega) \in \mathcal{F}(Q) \\ \\ \{(u, (Y \cup Z) \setminus \Theta) \mid Y \setminus (X \cup \Omega) = Z \setminus (X \cup \Omega) \wedge \\ \quad u \in s \parallel_X t \wedge \Theta = rf(\omega_1, \omega_2)\} \\ \quad \text{otherwise, where} \\ \\ \forall (s, Y_1) \in \mathcal{F}(P) \bullet Y \subseteq Y_1 \Rightarrow (\omega_1 \in \Omega \wedge \omega_1 \notin Y_1) \\ \forall (t, Z_1) \in \mathcal{F}(Q) \bullet Z \subseteq Z_1 \Rightarrow (\omega_2 \in \Omega \wedge \omega_2 \notin Z_1) \end{array} \right.$$

$$s \parallel_X t \hat{\langle \omega \rangle} = \{\} \quad s \hat{\langle \omega_1 \rangle} \parallel_X t \hat{\langle \omega_2 \rangle} = \{u \hat{\langle \omega_1 \& \omega_2 \rangle} \mid u \in s \parallel_X t\}$$

where $\omega, \omega_1, \omega_2 \in \Omega$, and $s, t \in \Sigma^*$.

Semantic of Parallel Composition - (3)

$$rf(\omega_1, \omega_2) =_{def} \begin{cases} \{\omega_1 \& \omega_2\} & \omega_1 \in \Omega \wedge \omega_2 \in \Omega \\ \{\omega_1\} & \omega_1 \in \Omega \wedge \omega_2 = \epsilon \\ \{\omega_2\} & \omega_2 \in \Omega \wedge \omega_1 = \epsilon \\ \{\} & \omega_1 = \epsilon \wedge \omega_2 = \epsilon \end{cases}$$

More laws:

$$THROW \parallel P = P ; THROW$$

$$THROW \parallel (YIELD ; P) = THROW \sqcap (P ; THROW)$$

Semantic of Exception Handling

$$\begin{aligned}\mathcal{D}(P \triangleright Q) &= \mathcal{D}(P) \cup \{s \hat{t} \mid s \hat{\langle ! \rangle} \in \text{traces}_{\perp}(P) \wedge t \in \mathcal{D}(Q)\} \\ \mathcal{F}(P \triangleright Q) &= \{(s, X) \mid s \in \Sigma^*\{\check{,}?\} \wedge (s, X \cup \{!\}) \in \mathcal{F}(P)\} \cup \\ &\quad \{(s \hat{t}, X) \mid s \hat{\langle ! \rangle} \in \text{traces}_{\perp}(P) \wedge (t, X) \in \mathcal{F}(Q)\} \cup \\ &\quad \{(s, X) \mid s \in \mathcal{D}(P \triangleright Q)\}\end{aligned}$$

Laws:

$$\begin{aligned}P \triangleright \text{THROW} &= P \\ \text{THROW} \triangleright P &= P \\ \text{SKIP} \triangleright P &= \text{SKIP} \\ \text{YIELD} \triangleright P &= \text{YIELD} \\ \text{STOP} \triangleright P &= \text{STOP} \\ P \triangleright (Q \sqcap R) &= (P \triangleright Q) \sqcap (P \triangleright R) \\ (P \sqcap Q) \triangleright R &= (P \triangleright R) \sqcap (Q \triangleright R) \\ P \triangleright (Q \triangleright R) &= (P \triangleright Q) \triangleright R\end{aligned}$$

Outline

- Motivation
- Brief Introduction to cCSP
- Extended cCSP and Failure-Divergence Semantics
 - Standard Process
 - **Compensating Process**
- Related Work and Conclusion

Semantic Domain of Compensating Process

- The semantics of a compensating process PP is (F, D, C)
 - F and D are the failure and divergence sets of the forward behaviour of PP
 - $C \subseteq \Sigma_{\Omega}^* \times \mathbb{P}(\Sigma^{*\Omega} \times \mathbb{P}(\Sigma^{\Omega})) \times \mathbb{P}(\Sigma^{*\Omega})$ is the compensating behavior set of (s, F^c, D^c)
 - s is the terminating trace in the forward behaviour
 - F^c and D^c are the failure and divergence sets of the compensating behaviour for s

Axioms of the Semantic Domain

- The axioms of the semantic model (F, D, C)
 - The forward behaviour model $(F$ and $D)$ must satisfy the axioms for the semantics model of standard process, and we use P to represent the forward process
 - The trace s of each element (s, F^c, D^c) in C must be a non-divergent terminating trace in the forward behavior, that is $s \in trace_t(P)$

$$trace_t(P) = \{s \hat{\langle \omega \rangle} \mid s \hat{\langle \omega \rangle} \in traces_{\perp}(P) \setminus \mathcal{D}(P) \wedge \omega \in \Omega\}$$

- The failure set F^c and divergence set D^c of each element in C must also satisfy the axioms for the semantics model of standard process

Refinement of Compensating Process

- $PP_1 \sqsubseteq PP_2$ is defined as:

$$(F_1, D_1, C_1) \sqsubseteq (F_2, D_2, C_2) =_{df} F_1 \supseteq F_2 \wedge D_1 \supseteq D_2 \wedge C_1 \downarrow S \sqsubseteq_c C_2 \downarrow S$$

- $S = trace_t(P_1) \cap trace_t(P_2)$, P_1 and P_2 are the forward processes of PP_1 and PP_2 , and $C \downarrow S = \{(s_1, F, D) \mid (s_1, F, D) \in C \wedge s_1 \in S\}$

- $S = trace_t(P_2) \setminus D_1$

- The refinement relation between compensating behaviour sets

$$C_1 \sqsubseteq_c C_2 =_{df} C_1 = \{\} \vee (C_2 \neq \{\} \wedge \mathfrak{R}(C_1, C_2))$$

$$\mathfrak{R}(C_1, C_2) =_{df} \forall (s, F, D) \in C_2 \setminus C_1 \bullet$$

$$(\exists (s_1, F_1, D_1) \in C_1 \setminus C_2 \bullet s_1 = s \wedge F_1 \supseteq F \wedge D_1 \supseteq D)$$

Semantic Functions

- The semantics of PP can be calculated by the semantics functions as $(\mathcal{F}^c(PP), \mathcal{D}^c(PP), \mathcal{C}(PP))$

- The forward failure set function

$$\mathcal{F}^c(PP) : \mathcal{PP} \rightarrow \mathbb{P}(\Sigma^{*\Omega} \times \mathbb{P}(\Sigma^\Omega))$$

- The forward divergence set function

$$\mathcal{D}^c(PP) : \mathcal{PP} \rightarrow \mathbb{P}(\Sigma^{*\Omega}),$$

- The compensating behaviour set function

$$\mathcal{C}(PP) : \mathcal{PP} \rightarrow \mathbb{P}(\Sigma_\Omega^* \times \mathbb{P}(\Sigma^{*\Omega} \times \mathbb{P}(\Sigma^\Omega)) \times \mathbb{P}(\Sigma^{*\Omega}))$$

Semantics of Compensation Pair

- Semantic definition

$$\mathcal{F}^c(P \div Q) = \mathcal{F}(P)$$

$$\mathcal{D}^c(P \div Q) = \mathcal{D}(P)$$

$$\mathcal{C}(P \div Q) = \{(s, F^c, D^c) \mid \exists s \in \text{trace}_t(P) \bullet$$

$$(s = t \hat{\langle \checkmark \rangle} \wedge F^c = \mathcal{F}(Q) \wedge D^c = \mathcal{D}(Q)) \vee$$

$$(s \in \Sigma_{\{!,?\}}^* \wedge F^c = \mathcal{F}(\text{SKIP}) \wedge D^c = \{\})\}$$

- Laws, in which $DIVV = DIV \div P$

$$DIV \div P = DIV \div Q$$

$$DIVV \sqsubseteq PP$$

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

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

Semantics of Basic Processes

- Semantic definition

$$SKIP P = SKIP \dot{\div} SKIP$$

$$THROW W = THROW \dot{\div} SKIP$$

$$YIELD D = YIELD \dot{\div} SKIP$$

- Laws

$$YIELD D \sqsubseteq SKIP$$

Examples of Basic Processes

	F	D	C		
			s	F^c	D^c
<i>SKIPP</i>	$(\langle \rangle, \checkmark \notin X)$ $(\langle \checkmark \rangle, X)$		$\langle \checkmark \rangle$	$(\langle \rangle, \checkmark \notin X)$ $(\langle \checkmark \rangle, X)$	
<i>THROWW</i>	$(\langle \rangle, ! \notin X)$ $(\langle ! \rangle, X)$		$\langle ! \rangle$	$(\langle \rangle, \checkmark \notin X)$ $(\langle \checkmark \rangle, X)$	
<i>YIELDD</i>	$(\langle \rangle, \checkmark \notin X)$ $(\langle \rangle, ? \notin X)$ $(\langle \checkmark \rangle, X)$ $(\langle ? \rangle, X)$		$\langle \checkmark \rangle$	$(\langle \rangle, \checkmark \notin X)$ $(\langle \checkmark \rangle, X)$	
			$\langle ? \rangle$	$(\langle \rangle, \checkmark \notin X)$ $(\langle \checkmark \rangle, X)$	
$A \div B$	$(\langle \rangle, A \notin X)$ $(\langle A \rangle, \checkmark \notin X)$ $(\langle A, \checkmark \rangle, X)$		$\langle A, \checkmark \rangle$	$(\langle \rangle, B \notin X)$ $(\langle B \rangle, \checkmark \notin X)$ $(\langle B, \checkmark \rangle, X)$	

Semantics of Transaction Block

$$\begin{aligned}\mathcal{D}([PP]) &= \mathcal{D}^c(PP) \cup \\ &\quad \{s_1 \mid \exists (s, F^c, D^c) \in \mathcal{C}(PP) \bullet s = t \hat{\langle ! \rangle} \wedge s_2 \in D^c \wedge s_1 = t \hat{s}_2\} \\ \mathcal{F}([PP]) &= \{(s, X) \mid s \in \Sigma^* \wedge (s, X \cup \{!\}) \in \mathcal{F}^c(PP)\} \cup \\ &\quad \{(s_1, X_1) \mid \exists (s, F^c, D^c) \in \mathcal{C}(PP) \bullet \\ &\quad \quad (s \in \Sigma_{\{\checkmark, ?\}}^* \wedge s_1 = s \wedge X_1 \subseteq \Sigma^\Omega) \vee \\ &\quad \quad (s = t \hat{\langle ! \rangle} \wedge (s_2, X_2) \in F^c \wedge s_1 = t \hat{s}_2 \wedge X_1 = X_2)\} \cup \\ &\quad \{(s, X) \mid s \in \mathcal{D}([PP])\}\end{aligned}$$

Laws of Transaction Block

$$\begin{aligned} [SKIP \div P] &= SKIP \\ [THROW \div P] &= SKIP \\ [YIELD \div P] &= YIELD \\ [DIVV] &= DIV \\ [P \div Q] &= P \triangleright SKIP \\ PP_1 \sqsubseteq PP_2 &\Rightarrow [PP_1] \sqsubseteq [PP_2] \end{aligned}$$

Semantics of Sequential Composition (1)

Let P and Q be the forward processes of PP and QQ

$$\mathcal{F}^c(PP ; QQ) = \mathcal{F}(P ; Q)$$

$$\mathcal{D}^c(PP ; QQ) = \mathcal{D}(P ; Q)$$

$$\begin{aligned} \mathcal{C}(PP ; QQ) = \{ & (s, F^c, D^c) \mid \exists (s_1, F_1^c, D_1^c) \in \mathcal{C}(PP), (s_2, F_2^c, D_2^c) \in \mathcal{C}(QQ) \bullet \\ & (s_1 = t \hat{\langle \checkmark \rangle} \wedge s = t \hat{s}_2 \wedge F^c = \mathcal{F}(P_2; P_1) \wedge D^c = \mathcal{D}(P_2; P_1)) \vee \\ & (s_1 \neq t \hat{\langle \checkmark \rangle} \wedge s = s_1 \wedge F^c = F_1^c \wedge D^c = D_1^c) \\ & \} \end{aligned}$$

where $\mathcal{F}(P_2) = F_2^c, \mathcal{D}(P_2) = D_2^c, \mathcal{F}(P_1) = F_1^c$, and $\mathcal{D}(P_1) = D_1^c$

Laws of Sequential Composition

$$SKIPP ; PP = PP$$

$$PP ; SKIPP = PP$$

$$THROWW ; PP = THROWW$$

$$YIELDD ; YIELDD = YIELDD$$

$$[P \div Q ; THROWW] = P ; Q$$

$$[P_1 \div Q_1 ; P_2 \div Q_2 ; THROWW] = P_1 ; P_2 ; Q_2 ; Q_1$$

Semantics of Internal Choice

Let P and Q be the forward processes of PP and QQ

$$\mathcal{F}^c(PP \sqcap QQ) = \mathcal{F}(P \sqcap Q)$$

$$\mathcal{D}^c(PP \sqcap QQ) = \mathcal{D}(P \sqcap Q)$$

$$\mathcal{C}(PP \sqcap QQ) = \{(s, F, D) \mid (s, F, D) \in \mathcal{C}(PP) \cup \mathcal{C}(QQ) \wedge s \in \text{trace}_t(P \sqcap Q)\}$$

Laws of Internal and External Choice

$$PP \sqcap PP = PP$$

$$PP \sqcap QQ = QQ \sqcap PP$$

$$PP \sqcap (QQ \sqcap RR) = (PP \sqcap QQ) \sqcap RR$$

$$DIVV \sqcap PP = DIVV$$

$$[PP \sqcap QQ] = [PP] \sqcap [QQ]$$

$$PP \sqcap QQ \sqsubseteq PP$$

$$PP ; (QQ \sqcap RR) = (PP ; QQ) \sqcap (PP ; RR)$$

$$(QQ \sqcap RR) ; PP = (QQ ; PP) \sqcap (RR ; PP)$$

$$PP \sqbox PP = PP$$

$$PP \sqbox QQ = QQ \sqbox PP$$

$$[PP \sqbox QQ] = [PP] \sqbox [QQ]$$

$$PP \sqbox (QQ \sqbox RR) = (PP \sqbox QQ) \sqbox RR$$

Laws of Parallel Composition

$$PP \underset{X}{\parallel} QQ = QQ \underset{X}{\parallel} PP$$

$$PP \underset{X}{\parallel} (QQ \underset{X}{\parallel} RR) = (PP \underset{X}{\parallel} QQ) \underset{X}{\parallel} RR$$

$$PP \underset{X}{\parallel} (QQ \sqcap RR) = (PP \underset{X}{\parallel} QQ) \sqcap (PP \underset{X}{\parallel} RR)$$

$$[(P \div P' \underset{X}{\parallel} Q \div Q') ; THROWW] = (P \underset{X}{\parallel} Q); (P' \underset{X}{\parallel} Q')$$

$$[(P \div P' ; Q \div Q') \parallel THROWW] = P ; Q ; Q' ; P'$$

$$[(P \div P' ; YIELDD ; Q \div Q') \parallel THROWW] = \\ (P ; P') \sqcap (P ; Q ; Q' ; P')$$

$$[(YIELDD ; P \div P' ; YIELDD ; Q \div Q') \parallel THROWW] = \\ SKIP \sqcap (P ; P') \sqcap (P ; Q ; Q' ; P')$$

$$[(YIELDD ; P \div P') \parallel (YIELDD ; Q \div Q') \parallel THROWW] = \\ SKIP \sqcap (P ; P') \sqcap (Q ; Q') \sqcap (P \parallel Q); (P' \parallel Q')$$



Laws of Hiding and Renaming

$$(PP \setminus X) \setminus Y = (PP \setminus Y) \setminus X$$

$$(PP \setminus X) \setminus Y = PP \setminus (X \cup Y)$$

$$(PP \sqcap QQ) \setminus X = (PP \setminus X) \sqcap (QQ \setminus X)$$

$$PP \setminus \{\} = PP$$

$$(PP \sqcap QQ)[R] = PP[R] \sqcap QQ[R]$$

$$(PP \sqcap QQ)[R] = PP[R] \sqcap QQ[R]$$

$$(PP[R])[R'] = PP[R \circ R']$$



Partial Order Theorem

- **Corollary:** *The compensating behavior set C of each compensating process model (F, D, C) satisfies the following finiteness property: for each s in $S(C)$, $C \downarrow \{s\}$ is **finite**, where $S(C) = \{s \mid (s, F, D) \in C\}$*
- **Theorem:** *The compensating process space forms a partial order under refinement*

Comparison with Original cCSP (1)

- Some new and often used operators are introduced both in standard and compensating process, and the semantics incorporates refusal and divergence information
- We study the refinement relations for standard and compensating processes
- The law $PP_1 \sqsubseteq PP_2 \Rightarrow [PP_1] \sqsubseteq [PP_2]$ setups a bridge from the compensating process refinement to the refinement of standard process

Comparison with Original cCSP (2)

- Unlike cCSP, we do not allow implicit interruption in compensating process
- Yield interrupting behavior is kept in the semantics of the transaction block.
 - Remove some assumptions of laws
 - Refinement relation

Outline

- Motivation
- Brief Introduction to cCSP
- Extended cCSP and Failure-Divergence Semantics
 - Standard Process
 - Compensating Process
- Related Work and Conclusion

Related Work (1)

- SAGAS: Bruni *et al.* [BMM05]
- StAC: Butler *et al.* [BF04, BFN05]
- cCSP: Butler *et al.* [BHF04]
 - Operational semantics [BR05]
 - Semantics relating [RB06]
- Comparison: Bruni *et al.* [BBF⁺05]

Related Work (2)

- Lanotte *et al.* [LMSMT06] develop the communicating hierarchical timed automata to model the sequential and parallel compositions in nested LRTs.
- Bochhi *et al.* [Boc04] use the asynchronous π -calculus to model the LRT model in BTP supporting nested transaction.
- Qiu *et al.* [QWPZ05] propose a process language especially for the LRT model in BPEL
- He [He07] presents a new semantic theory for the languages describing LRT

Conclusion

- We extend the cCSP by distinguishing the external and internal choices, and introducing more operators
- A new semantics model based on the failures and divergences model of CSP [Ros97]
- Refinement relations
- Some laws for the extended cCSP
- Partialness of the semantic domain under refinement

Future Work

- Speculative choice
- It is ideal to have a beyond divergence model that is more suitable for the composition definition of compensating process
- Recursion in the compensating process

END

Thank you very much!

Questions?

References

- [AAA⁺06] Alexandre Alves, Assaf Arkin, Sid Askary, Ben Bloch, Francisco Curbera, Yaron Golland, Neelakantan Kartha, Sterling, Dieter König, Vinkesh Mehta, Satish Thatte, Danny van der Rijn, Prasad Yendluri, and Alex Yiu. Web services business process execution language version 2.0. OASIS Committee Draft, May 2006.
- [AJS05] Ali E. Abdallah, Cliff B. Jones, and Jeff W. Sanders, editors. *Communicating Sequential Processes: The First 25 Years, Symposium on the Occasion of 25 Years of CSP, London, UK, July 7-8, 2004, Revised Invited Papers*, volume 3525 of *Lecture Notes in Computer Science*. Springer, 2005.
- [BBF⁺05] Roberto Bruni, Michael J. Butler, Carla Ferreira, C. A. R. Hoare, Hernán C. Melgratti, and Ugo Montanari. Comparing two approaches to compensable flow composition. In Martín Abadi and Luca de Alfaro, editors, *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 383–397. Springer, 2005.
- [BF04] Michael J. Butler and Carla Ferreira. An operational semantics for stac, a language for modelling

long-running business transactions. In Rocco De Nicola, Gian Luigi Ferrari, and Greg Meredith, editors, *COORDINATION*, volume 2949 of *Lecture Notes in Computer Science*, pages 87–104. Springer, 2004.

- [BFN05] Michael J. Butler, Carla Ferreira, and Muan Yong Ng. Precise modelling of compensating business transactions and its application to bpm. *J. UCS*, 11(5):712–743, 2005.
- [BHF04] Michael J. Butler, C. A. R. Hoare, and Carla Ferreira. A trace semantics for long-running transactions. In Abdallah et al. [AJS05], pages 133–150.
- [BMM05] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Theoretical foundations for compensations in flow composition languages. In Jens Palsberg and Martín Abadi, editors, *POPL*, pages 209–220. ACM, 2005.
- [Boc04] Laura Bocchi. Compositional nested long running transactions. In Michel Wermelinger and Tiziana Margaria, editors, *FASE*, volume 2984 of *Lecture Notes in Computer Science*, pages 194–208. Springer, 2004.
- [BR05] Michael J. Butler and Shamim Ripon. Executable semantics for compensating csp. In

Mario Bravetti, Leïla Kloul, and Gianluigi Zavattaro, editors, *EPEW/WS-FM*, volume 3670 of *Lecture Notes in Computer Science*, pages 243–256. Springer, 2005.

- [GMS87] Hector Garcia-Molina and Kenneth Salem. Sagas. In Umeshwar Dayal and Irving L. Traiger, editors, *SIGMOD Conference*, pages 249–259. ACM Press, 1987.
- [He07] Jifeng He. Utp semantics for web services. In Jim Davies and Jeremy Gibbons, editors, *IFM*, volume 4591 of *Lecture Notes in Computer Science*, pages 353–372. Springer, 2007.
- [HJ98] C.A.R. Hoare and He Jifeng. *Unifying Theories of Programming*. Prentice Hall, 1998.
- [LMSMT06] Ruggero Lanotte, Andrea Maggiolo-Schettini, Paolo Milazzo, and Angelo Troina. Modeling long-running transactions with communicating hierarchical timed automata. In Roberto Gorrieri and Heike Wehrheim, editors, *FMOODS*, volume 4037 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2006.
- [QWPZ05] Zongyan Qiu, Shuling Wang, Geguang Pu, and Xiangpeng Zhao. Semantics of bpm4ws-like fault

and compensation handling. In John Fitzgerald, Ian J. Hayes, and Andrzej Tarlecki, editors, *FM*, volume 3582 of *Lecture Notes in Computer Science*, pages 350–365. Springer, 2005.

- [RB06] S. Ripon and M. Butler. Relating semantic models of compensating csp. Technical report, University of Southampton, 2006.
- [Rip08] Shamim Ripon. *Extending and Relating Semantic Models of Compensating CSP*. PhD thesis, University of Southampton, 2008.
- [Ros97] A. W. Roscoe. *The Theory and Practice of Concurrency*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.
- [Ros04] A. W. Roscoe. Seeing beyond divergence. In Abdallah et al. [AJS05], pages 15–35.
- [Tha01] S. Thatte. XLANG web services for business process design, 2001.