Coordination via Interaction Constraints

Dave Clarke José Proença

CWI, Amsterdam
jose.proenca@cwi.nl

CIC 2009

Coordination

Coordination is the process of building programs by gluing together active pieces.

"Coordination Languages and their Significance"
Carriero and Gelernter

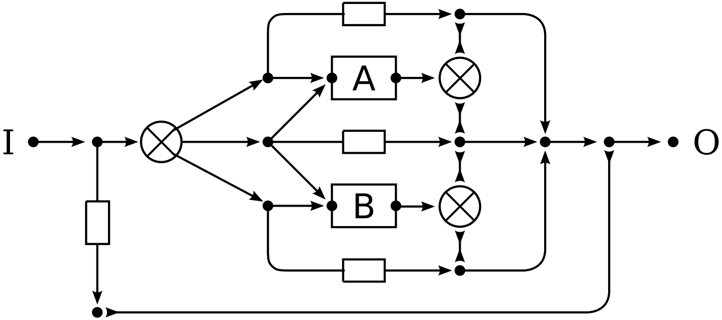
Coordination is constrained interaction: it constrains interaction protocols among communicating software components.

"Coordination as Constrained Interaction" Peter Wegner

Outline

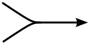
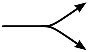
Reo^{++} = Constraints + State + Interaction + Locality

Coordination Model Reo



Channel Encoding

Synchronisation constraints (SC) and Data flow constraints (DFC)

Channel	SC	DFC
a  b	$a \leftrightarrow b$	$\widehat{a} = \widehat{b}$
a  b	$a \leftrightarrow b$	$\texttt{tt}$
a  b	$b \rightarrow a$	$b \rightarrow (\widehat{a} = \widehat{b})$
a, b  c	$(c \leftrightarrow (a \vee b)) \wedge \neg(a \wedge b)$	$a \rightarrow (\widehat{c} = \widehat{a}) \wedge b \rightarrow (\widehat{c} = \widehat{b})$
a  b, c	$(a \leftrightarrow b) \wedge (a \leftrightarrow c)$	$\widehat{b} = \widehat{a} \wedge \widehat{c} = \widehat{a}$
a  b (p)	$b \rightarrow a$	$b \rightarrow (p(\widehat{a}) \wedge \widehat{a} = \widehat{b}) \wedge$ $(a \wedge p(\widehat{a})) \rightarrow b$

Composition = Conjunction + Abstraction: $\exists \mathcal{X}, \widehat{\mathcal{X}}.\ (\psi_1 \wedge \psi_2)$

Constraint-approach

Reo^{++} = Constraints + State + Interaction + Locality

- Synchronisation and data flow constraints.

Coordination via Constraint Satisfaction

$x \in \mathcal{X}$ – ports in a connector – synchronization variables (Boolean)
$\widehat{x} \in \widehat{\mathcal{X}}$ – data flow variables ($\mathcal{D}ata_\perp \mathrel{\widehat{=}} \mathcal{D}ata \cup \{\texttt{NO-FLOW}\}$)
$s \in \Sigma$ – term variables (over $\mathcal{D}ata$)

$\mathcal{X}$–$\widehat{\mathcal{X}}$ linked by frame axiom

for each $x \in \mathcal{X}$: $\neg x \leftrightarrow \widehat{x} = \texttt{NO-FLOW}$

Constraints: formulæ over $\mathcal{X}$, $\widehat{\mathcal{X}}$, and Σ

$$t ::= \widehat{x} \mid s \mid d \qquad \text{(terms)}$$
$$\psi ::= x \mid \texttt{tt} \mid p(\bar{t}) \mid \psi \wedge \psi \mid \neg\psi \qquad \text{(formulæ)}$$

$(\bar{t} = t_1, \ldots, t_n)$

Solution to ψ

$$\sigma : \mathcal{X} \to \{\texttt{ff}, \texttt{tt}\} \cup \widehat{\mathcal{X}} \to \mathcal{D}ata_\perp$$

such that

$$\sigma, \mathcal{I} \models \psi$$

CWI

Satisfaction relation

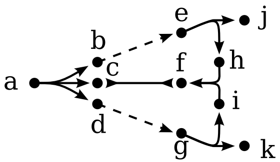
$$\begin{array}{lll}
\sigma, \mathcal{I} \models \mathtt{tt} & \text{always} & \\
\sigma, \mathcal{I} \models x & \textit{iff} & \sigma(x) = \mathtt{tt} \\
\sigma, \mathcal{I} \models \psi_1 \wedge \psi_2 & \textit{iff} & \sigma, \mathcal{I} \models \psi_1 \text{ and } \sigma, \mathcal{I} \models \psi_2 \\
\sigma, \mathcal{I} \models \neg \psi & \textit{iff} & \sigma, \mathcal{I} \not\models \psi \\
\sigma, \mathcal{I} \models p(t_1, \ldots, t_n) & \textit{iff} & (Val_\sigma(t_1), \ldots, Val_\sigma(t_n)) \in p_\mathcal{I}
\end{array}$$

$Val_\sigma(t)$ performs substitution.

Assuming for each predicate symbol p an interpretation $p_\mathcal{I} \subseteq \mathcal{D}ata_\perp^n$

Example

Exclusive router connector



$$\Psi_{SC} = (a \leftrightarrow b) \wedge (a \leftrightarrow c) \wedge (a \leftrightarrow d) \wedge (e \rightarrow b) \wedge (c \leftrightarrow f) \wedge (g \rightarrow d) \wedge (e \leftrightarrow j) \wedge \ldots$$

$$\Psi_{DFC} = (a \rightarrow (\widehat{b} = \widehat{a} \wedge \widehat{c} = \widehat{a})) \wedge (e \rightarrow \widehat{b} = \widehat{e}) \wedge (g \rightarrow \widehat{d} = \widehat{g}) \wedge \widehat{j} = \widehat{e} \wedge \widehat{h} = \widehat{e} \wedge \ldots$$

$$\Psi = \exists \mathcal{X}, \widehat{\mathcal{X}}.(\Psi_{SC} \wedge \Psi_{DFC} \wedge \mathtt{Frame}(\mathcal{X} \cup \{a, j, k\}) \text{ where } \mathcal{X} = \{b, c, d, e, f, g, h, i\}$$

Solutions to Ψ:

- $\widehat{j} = \widehat{a} \ \wedge \ \widehat{k} = \text{NO-FLOW}$
- $\widehat{k} = \widehat{a} \ \wedge \ \widehat{j} = \text{NO-FLOW}$
- $\widehat{a} = \text{NO-FLOW} \ \wedge \ \widehat{j} = \text{NO-FLOW} \wedge \widehat{k} = \text{NO-FLOW}$

CWI

Stateful connectors

Reo^{++} = Constraints + State + Interaction + Locality

- Evolution in time.
- To capture the next state in the constraints.

Stateful primitives

- Encode (parameterised) state machine as constraints over current and future states.
- Plus a constraint representing current state
- $state_q$ and $state'_q$ in Σ:
  - term variables for the current and next state of primitive q

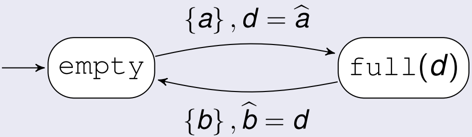
Add n-ary uninterpreted function symbols

$$t \quad ::= \quad \ldots \quad | \quad f(t_1, \ldots, t_n) \qquad \text{(terms)}$$

Can encode structured states, e.g., $state_q = \texttt{full}(10)$.

Example: FIFO1 channel

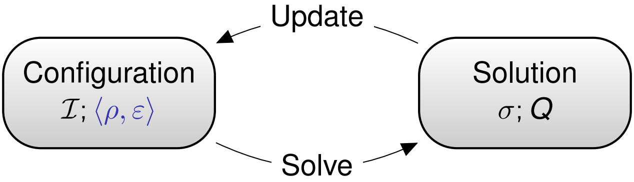
With Constraint Automata



With constraints

$$
\begin{array}{l}
state = \texttt{empty} \rightarrow \left\{ \begin{array}{l} \neg b \;\wedge \\ a \rightarrow state = \texttt{full}(\widehat{a}) \;\wedge \\ \neg a \rightarrow state' = state \end{array} \right. \quad \bigwedge \\
state = \texttt{full}(d) \rightarrow \left\{ \begin{array}{l} \neg a \;\wedge \\ b \rightarrow \widehat{b} = d \wedge state' = \texttt{empty} \;\wedge \\ \neg b \rightarrow state' = state \end{array} \right. \quad \bigwedge \\
state = \texttt{empty} \qquad \leftarrow \textit{ephemeral}
\end{array}
$$

A constraint satisfaction-based engine for Reo



Configuration

- ρ – persistent constraints
- ε – ephemeral constraints

$$\langle \rho, \varepsilon^n \rangle \xrightarrow{solve} \langle \sigma^n \rangle \xrightarrow{update} \langle \rho, \varepsilon^{n+1} \rangle$$

where:

$$\begin{aligned} \sigma^n, \mathcal{I} &\models \rho \wedge \varepsilon^n \\ \varepsilon^{n+1} &\equiv \bigwedge_q state_q = \sigma^n(state'_q) \end{aligned}$$

Ephemeral constraints correspond to the state vector.

Interaction

Reo^{++} = Constraints + State + Interaction + Locality

- Interaction with the external world.

External symbols

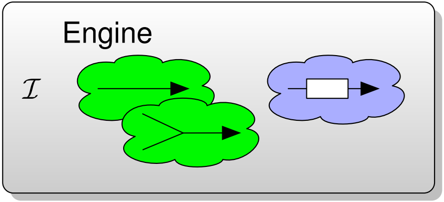
Constraints are formulæ in the following grammar:

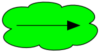
$$t ::= \widehat{x} \mid s \mid \mathbf{k} \mid f(\bar{t}) \mid \mathbf{f}(\bar{t}) \qquad \text{(terms)}$$

$$\psi ::= x \mid \texttt{tt} \mid p(\bar{t}) \mid \mathbf{p}(\bar{t}) \mid \psi \wedge \psi \mid \neg\psi \mid \mathbf{c}(\overline{\psi}, \bar{t}) \qquad \text{(formulæ)}$$

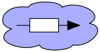
- Various external symbols (**f**,**p**,**c**) — interpreted via interaction.
- **k** — variables whose solution is communicated to an external primitive.
- Interactive constraint satisfaction "fills-in" $\mathcal{I}$.

Interactive World



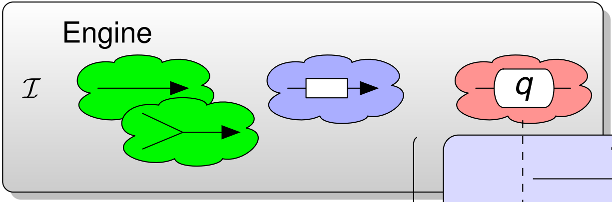


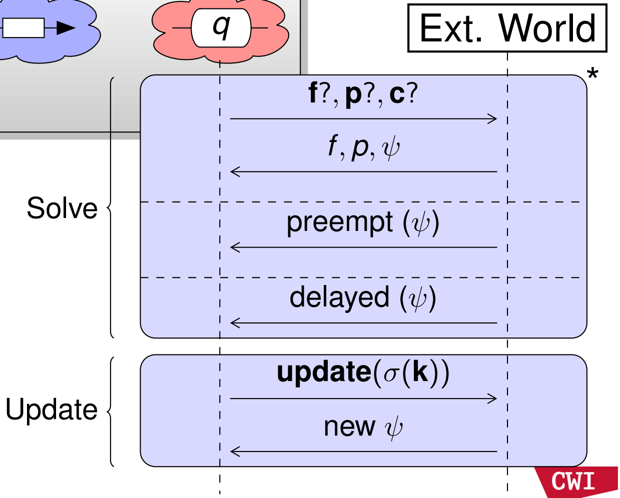
– internal, stateless

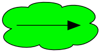


– internal, stateful

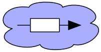
Interactive World



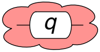




– internal, stateless



– internal, stateful



– external

Opening up the solver

CSP described abstractly as a (labelled) transition system:

$$\mathcal{I}, \psi \xrightarrow{a} \mathcal{I}', \psi'$$

Internal "solve" steps

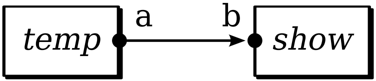
$$\mathcal{I}, \psi \xrightarrow{\tau} \mathcal{I}, \psi' \text{ requiring } [\![\psi]\!]_{\mathcal{I}} = [\![\psi']\!]_{\mathcal{I}}, \text{ where } [\![\psi]\!]_{\mathcal{I}} = \{\sigma \mid \sigma, \mathcal{I} \models \psi\}$$

External Interaction Step — θ updates $\mathcal{I}$

$$\mathcal{I}, \psi \xrightarrow{\theta} \mathcal{I}', \psi' \text{ requiring } [\![\psi]\!]_{\mathcal{I}} \subseteq [\![\psi']\!]_{\mathcal{I}'}$$

CWI

Show temperature using Externals



Other possible configuration

$$\varepsilon \equiv (a \rightarrow \widehat{a} = \textbf{current}_{temp}) \wedge$$
$$(b \rightarrow \textbf{Acceptable}_{show}(\widehat{b})) \wedge (b \rightarrow \textbf{k}_{show} = \texttt{print}(\widehat{b}))$$

$$\varphi \equiv (a \leftrightarrow b) \wedge (\widehat{a} = \widehat{b}) \wedge \texttt{Frame}(a, b) \wedge \varepsilon$$

- Can model user interaction within a synchronous step.

Show temperature using Externals

A trace of the solve stage

$$\varepsilon \equiv (a \to \widehat{a} = \mathbf{current}_{temp}) \wedge (b \to \mathbf{Acceptable}_{show}(\widehat{b})) \wedge (b \to \mathbf{k}_{show} = \mathtt{print}(\widehat{b}))$$

$$\varphi \equiv (a \leftrightarrow b) \wedge (\widehat{a} = \widehat{b}) \wedge \mathtt{Frame}(a, b) \wedge \varepsilon$$

$$\varphi \xrightarrow{}{}^{*} \psi \wedge \widehat{a} = \mathbf{current}_{temp}$$

$$\xrightarrow{\mathbf{current}_{temp} = \mathtt{20^\circ C}} \psi \wedge \widehat{a} = \mathbf{20^\circ C}$$

$$\xrightarrow{}{}^{*} \phi \wedge \widehat{b} = \mathtt{20^\circ C} \wedge \mathbf{Acceptable}_{show}(\mathtt{20^\circ C})$$

$$\xrightarrow{\mathbf{Acceptable}_{show}(\mathtt{20^\circ C}) = \mathtt{tt}} \phi \wedge \widehat{b} = \mathtt{20^\circ C} \wedge \mathtt{tt}$$

$$\xrightarrow{}{}^{*} a \wedge b \wedge \widehat{a} = \mathtt{20^\circ C} \wedge \widehat{b} = \mathtt{20^\circ C} \wedge \mathbf{k}_{show} = \mathtt{print}(\mathtt{20^\circ C})$$

On-the-fly Constraint Generation

$\mathbf{c}(\overline{\psi}, \overline{t})$

- Abstract constraint over formulæ and terms.
- Models on-the-fly constraint generation.
- Can model a stream of possibilities, given one page at a time:

$$\mathbf{primes_1}(x, \widehat{x})$$

$$\xrightarrow{\mathbf{primes_1}(x,\widehat{x}) = (\neg x \ \vee \ \widehat{x}=2 \ \vee \ \widehat{x}=3 \ \vee \ \mathbf{primes_3}(x,\widehat{x}))}$$

$$\neg x \ \vee \ \widehat{x} = 2 \ \vee \ \widehat{x} = 3 \ \vee \ \mathbf{primes_3}(x, \widehat{x})$$

$$\xrightarrow{\mathbf{primes_3}(x,\widehat{x}) = (\widehat{x}=5 \ \vee \ \widehat{x}=7 \ \vee \ \mathbf{primes_7}(x,\widehat{x}))}$$

$$\neg x \ \vee \ \widehat{x} = 2 \ \vee \ \widehat{x} = 3 \ \vee \ \widehat{x} = 5 \ \vee \ \widehat{x} = 7 \ \vee \ \mathbf{primes_7}(x, \widehat{x})$$

Future: pass constraint *('query')* on $\widehat{x}$ to the owner of $\mathbf{primes_1}(x, \widehat{x})$ to pre-filter the answers.

CWI

Locality

Reo^{++} = Constraints + State + Interaction + Locality

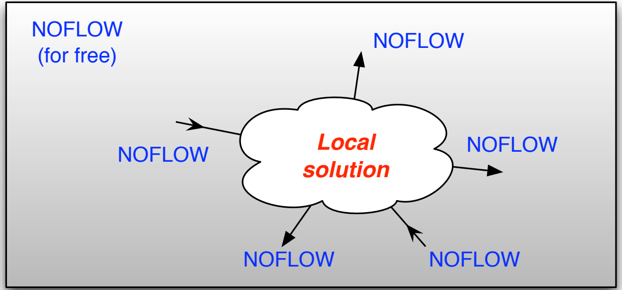
- Evolution of sub-connectors;
- Partial solutions and interpretations.

Problems with Existing Approach

1. Solving constraints requires a global solution.
2. Unscalable as all parties are necessarily involved.
3. Limited concurrency.
4. Behaviour must be *apriori* fully specified.

Partial (and Local) Solutions

1. Need a partial logic (with partial solutions).
2. All primitives admit "no flow" solutions.
3. Aim for local solutions:



4. Independent local solutions can be discovered concurrently.

Partial logic

Partial solutions

$$\begin{array}{rcll} \sigma(x) & \in & \{\texttt{tt}, \texttt{ff}, \bot\} & \text{(we can drop } \texttt{NO-FLOW}\text{)} \\ [\sigma, \mathcal{I} \models \psi] & \in & \{\texttt{tt}, \texttt{ff}, \bot\} & \end{array}$$

Partial satisfaction

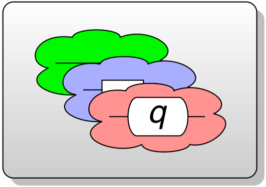
$$\begin{array}{rcl} [\sigma, \mathcal{I} \models \texttt{tt}] & = & \texttt{tt} \\ [\sigma, \mathcal{I} \models x] & = & \sigma(x) \\ [\sigma, \mathcal{I} \models \psi_1 \wedge \psi_2] & = & \mathit{min}([\sigma, \mathcal{I} \models \psi_1], [\sigma, \mathcal{I} \models \psi]) \\ [\sigma, \mathcal{I} \models \neg\psi] & = & \mathit{neg}([\sigma, \mathcal{I} \models \psi]) \\ [\sigma, \mathcal{I} \models p(t_1, \ldots, t_n)] & = & p(\mathit{Val}_{\sigma,\mathcal{I}}(t_1), \ldots, \mathit{Val}_{\sigma,\mathcal{I}}(t_n)) \in \mathcal{I} \\ [\sigma, \mathcal{I} \models \mathbf{p}(t_1, \ldots, t_n)] & = & \mathbf{p}(\mathit{Val}_{\sigma,\mathcal{I}}(t_1), \ldots, \mathit{Val}_{\sigma,\mathcal{I}}(t_n)) \underline{\in} \mathcal{I} \\ & \vdots & \end{array}$$

$$\texttt{tt} < \bot < \texttt{ff}$$

CWI

Reduction Rules: Join

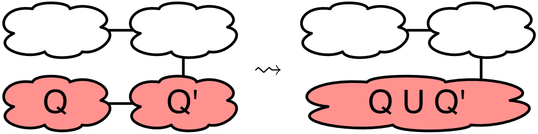
Blocks of constraints



Each primitive: a block of constraints
Each block: NO−FLOW solution.
Union of blocks: NO−FLOW solution.

Join

$$\frac{Q \leftrightarrow Q'}{\mathcal{I}; \langle\psi\rangle_{Q}, \langle\psi'\rangle_{Q'} \xrightarrow{\mathcal{I}} \mathcal{I}; \langle\psi \wedge \psi'\rangle_{Q \cup Q'}}$$



Reduction Rules: Update

When local solution is found, relevant part of constraints are updated.

Update

$$\frac{\mathrm{E} = \text{Externals q} : \sigma(\mathbf{k}_q) \neq \bot \qquad \mathit{isSolution}(\sigma, Q) \qquad \mathrm{I} = \text{Internals q} : \sigma(\mathit{state}'_q) \neq \bot \quad \forall q \in \mathrm{I} : \varepsilon_q \equiv \mathit{state}_q = \sigma(\mathit{state}'_q)}{\mathcal{I}; \langle \sigma \vee \psi \rangle_Q \xrightarrow{\forall q \in \mathrm{E}: \mathbf{update}(\sigma(\mathbf{k}_q)) = \varepsilon_q} \emptyset; \langle \mathit{safe}(q) \wedge \varepsilon_q \rangle_q^{q \in Q}}$$

isSolution(σ, Q) is a purely syntactic test for local solutions, based on connectivity blocks of formulæ.

CWI

Conclusion

1. Instead of channels as an implementation concept, use meta-level interaction with external entities.

2. Use of external states, functions and predicates to deal with arbitrary external entities.
3. Interactive (aka Open) Constraint Satisfaction.
4. Partial (local) constraints/solutions.
5. CSP solver performs coordination.

Future

There are many small indications that the tidy examples of simple coordination protocols are only the tip of the iceberg of a much larger and less tidy space of real-world coordination protocols that we have not begun to explore. The extension of models of coordination from toy examples (...) to real-world coordination of distributed systems and software engineering is a challenging problem.

"Coordination as Constrained Interaction" Peter Wegner

Practical coordination models must express (...) also the dynamic constraints of air traffic controllers and complex organisations.

"Coordination as Constrained Interaction" Peter Wegner