Constraint Handling Rules with Multiset Comprehension Patterns

Edmund S. L. Lam sllam@qatar.cmu.edu
Iliano Cervesato iliano@cmu.edu

Carnegie Mellon University

Supported by grant NPRP 09-667-1-100, *Effective Programming for Large Distributed Ensembles*

CHR'14 Vienna, Austria, July 2014

Outline

Constraint Handling Rules (CHR)

Constraint Logic Programming Languages

- Rule-based multiset rewriting
  - Declarative
  - Forward chaining
  - Committed choice
  - Concurrent
- CHR is a specific instance

Constraint Handling Rules (CHR)

$$r \ @ \ \bar{H}_p \setminus \bar{H}_s \iff g \mid \bar{B}$$

- $\bar{H}_p$, $\bar{H}_s$, $\bar{B}$ are multisets of *atomic constraint patterns*: $p(\vec{t})$
  - r: Rule name
  - $\bar{H}_p$: Propagated head constraints
  - $\bar{H}_s$: Simplified head constraints
  - g: Guard conditions

  } LHS
  - $\bar{B}$: Body constraints

  } RHS
- A *program* $\mathcal{P}$ is a set of rules

$$\mathcal{P} \ \rhd \ St \ \mapsto_\alpha \ St'$$

- The *stores* St and St' are multiset of constraints

CHR with Multiset Comprehension Patterns (CHR^{cp})

- CHR rule patterns are strictly *atomic* predicates
- Pure forward chaining + atomic patterns are good
  - declarative and concise
  - high-level
- but not ideal when
  - performing aggregated operation
  - matching dynamic numbers of facts
- This work:
  - Extend CHR with *multiset comprehension patterns* (CHR^{cp})
  - Define an operational semantics for CHR^{cp}

Outline

Example: Pivoted Swapping (in "Vanilla" CHR)

- LHS cannot match a dynamic numbers of constraints
- Two agents X and Y want to swap data D w.r.t. pivot P
- Standard CHR implementation:

$$
\begin{array}{rcl}
\mathit{init} \mathbin{@} \mathit{swap}(X, Y, P) & \iff & \mathit{grabGE}(X, P, Y, [\,]), \mathit{grabLT}(Y, P, X, [\,]) \\
\mathit{ge1} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds), \mathit{data}(X, D) & \iff & D \geq P \mid \mathit{grabGE}(X, P, Y, [D \mid Ds]) \\
\mathit{ge2} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds) & \iff & \mathit{unrollData}(Y, Ds) \\
\mathit{lt1} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds), \mathit{data}(Y, D) & \iff & D < P \mid \mathit{grabLT}(Y, P, X, [D \mid Ds]) \\
\mathit{lt2} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds) & \iff & \mathit{unrollData}(X, Ds) \\
\mathit{unroll1} \mathbin{@} \mathit{unrollData}(L, [D \mid Ds]) & \iff & \mathit{unrollData}(L, Ds), \mathit{data}(L, D) \\
\mathit{unroll2} \mathbin{@} \mathit{unrollData}(L, [\,]) & \iff & \mathit{true}
\end{array}
$$

- Verbose: 7 rules and 3 auxiliary constraints

Example: Pivoted Swapping (in "Vanilla" CHR)

- LHS cannot match a dynamic numbers of constraints
- Two agents X and Y want to swap data D w.r.t. pivot P
- Standard CHR implementation:

$$
\begin{array}{rcl}
\mathit{init} \mathbin{@} \mathit{swap}(X, Y, P) & \iff & \mathit{grabGE}(X, P, Y, [\,]), \mathit{grabLT}(Y, P, X, [\,]) \\
\mathit{ge1} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds), \mathit{data}(X, D) & \iff & D \geq P \mid \mathit{grabGE}(X, P, Y, [D \mid Ds]) \\
\mathit{ge2} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds) & \iff & \mathit{unrollData}(Y, Ds) \\
\mathit{lt1} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds), \mathit{data}(Y, D) & \iff & D < P \mid \mathit{grabLT}(Y, P, X, [D \mid Ds]) \\
\mathit{lt2} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds) & \iff & \mathit{unrollData}(X, Ds) \\
\mathit{unroll1} \mathbin{@} \mathit{unrollData}(L, [D \mid Ds]) & \iff & \mathit{unrollData}(L, Ds), \mathit{data}(L, D) \\
\mathit{unroll2} \mathbin{@} \mathit{unrollData}(L, [\,]) & \iff & \mathit{true}
\end{array}
$$

- Verbose: 7 rules and 3 auxiliary constraints
- Accumulators incrementally store matches

Example: Pivoted Swapping (in "Vanilla" CHR)

- LHS cannot match a dynamic numbers of constraints
- Two agents X and Y want to swap data D w.r.t. pivot P
- Standard CHR implementation:

$$
\begin{array}{rcl}
\mathit{init} \mathbin{@} \mathit{swap}(X, Y, P) & \Longleftrightarrow & \mathit{grabGE}(X, P, Y, [\,]), \mathit{grabLT}(Y, P, X, [\,]) \\
\mathit{ge1} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds), \mathit{data}(X, D) & \Longleftrightarrow & D \geq P \mid \mathit{grabGE}(X, P, Y, [D \mid Ds]) \\
\mathit{ge2} \mathbin{@} \mathit{grabGE}(X, P, Y, Ds) & \Longleftrightarrow & \mathit{unrollData}(Y, Ds) \\
\mathit{lt1} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds), \mathit{data}(Y, D) & \Longleftrightarrow & D < P \mid \mathit{grabLT}(Y, P, X, [D \mid Ds]) \\
\mathit{lt2} \mathbin{@} \mathit{grabLT}(Y, P, X, Ds) & \Longleftrightarrow & \mathit{unrollData}(X, Ds) \\
\mathit{unroll1} \mathbin{@} \mathit{unrollData}(L, [D \mid Ds]) & \Longleftrightarrow & \mathit{unrollData}(L, Ds), \mathit{data}(L, D) \\
\mathit{unroll2} \mathbin{@} \mathit{unrollData}(L, [\,]) & \Longleftrightarrow & \mathit{true}
\end{array}
$$

- Verbose: 7 rules and 3 auxiliary constraints
- Accumulators incrementally store matches
- Relies on rule priorities: apply *ge1* before *ge2*, *lt1* before *lt2*

CHR with Comprehension Patterns

- Additional form of constraint patterns:
  - *Comprehension patterns M:* $\wr p(\vec{t}) \mid g \int_{\vec{x} \in t}$
    - Collects *all* $p(\vec{t})$ in the store that satisfy g
    - t is the multiset of all bindings to $\vec{x}$
- Pivoted Swapping in CHR^{cp}:

$$\begin{array}{rll} & swap(X,Y,P) & \\ pivotSwap \ @ & \wr data(X,D) \mid D \geq P \int_{D \in Xs} & \Longleftrightarrow \quad \wr data(Y,D) \int_{D \in Xs} \\ & \wr data(Y,D) \mid D < P \int_{D \in Ys} & \quad\quad\ \wr data(X,D) \int_{D \in Ys} \end{array}$$

  - Xs and Ys built from the store — *output*
  - Xs and Ys used to unfold the comprehensions — *input*

CHR with Comprehension Patterns

$$\begin{array}{rll} & swap(X,Y,P) & \\ pivotSwap\ @ & \wr data(X,D) \mid D \geq P \int_{D \in Xs} & \Longleftrightarrow \quad \wr data(Y,D) \int_{D \in Xs} \\ & \wr data(Y,D) \mid D < P \int_{D \in Ys} & \quad\quad\ \wr data(X,D) \int_{D \in Ys} \end{array}$$

- An example of CHR^{cp} rule application:

$$\begin{array}{rl} & \wr swap(a,b,5), data(a,1), data(a,6), data(a,7), data(b,2), data(b,8) \int \\ \mapsto_{\alpha} & \wr\ \ data(a,1), data(b,6), data(b,7), data(a,2), data(b,8) \int \end{array}$$

- Corresponds to the rule instance:

$$\begin{array}{rll} & swap(a,b,5) & \\ pivotSwap\ @ & \wr data(a,D) \mid D \geq 5 \int_{D \in \wr 6,7 \int} & \Longleftrightarrow \quad \wr data(b,D) \int_{D \in \wr 6,7 \int} \\ & \wr data(b,D) \mid D < 5 \int_{D \in \wr 2 \int} & \quad\quad\ \wr data(a,D) \int_{D \in \wr 2 \int} \end{array}$$

Outline

Maximality of Comprehension vs. Monotonicity

- Semantics of CHR^{cp} enforces *maximality* of comprehensions
- Example

$$\textit{pivotSwap} \ @ \ \begin{array}{l} \textit{swap}(X, Y, P) \\ \wr \textit{data}(X, D) \mid D \geq P \int_{D \in Xs} \\ \wr \textit{data}(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \\ \wr \textit{data}(Y, D) \int_{D \in Xs} \\ \wr \textit{data}(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{lll} \triangleright & \wr \textit{swap}(a, b, 5), \textit{data}(a, 1), & \boxed{\textit{data}(a, 6)}, \textit{data}(a, 7), \textit{data}(b, 2), \textit{data}(b, 8) \int \\ \not\mapsto_{\alpha} & \wr & \textit{data}(a, 1), \boxed{\textit{data}(a, 6)}, \textit{data}(b, 7), \textit{data}(a, 2), \textit{data}(b, 8) \int \end{array}$$

- Invalid — $\textit{data}(a, 6)$ is omitted
- CHR^{cp} is not monotonic!

$\mathcal{P} \triangleright St \mapsto_{\alpha} St'$ does not imply $\mathcal{P} \triangleright \wr St, St'' \int \mapsto_{\alpha} \wr St', St'' \int$

Non-monotonicity

$$\mathcal{P} \rhd St \mapsto_\alpha St' \quad \text{does not imply} \quad \mathcal{P} \rhd \wr St, St'' \int \mapsto_\alpha \wr St', St'' \int$$

- So what?
  - We cannot incrementally process constraints
  - At least not in the traditional manner ([Duck et al., 2004])
- Refined Operational Semantics

$$\mathcal{P} \rhd \langle Gs \ ; \ St \rangle \mapsto_\omega \langle Gs' \ ; \ St' \rangle$$

  - Gs: list of constraints to process (*goals*)
  - St: constraint store

Coping with Non-monotonicity

- Let's try to process incrementally:

$$\begin{array}{lll} & swap(X, Y, P) & \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} & \Longleftrightarrow \quad \wr data(Y, D) \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} & \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \langle [d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6), s(a,b,5) \int \rangle \\ \mapsto_\omega & \langle [d(b,6), s(a,b,5), d(a,7), \boxed{d(b,2)}, d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \ldots & \end{array}$$

* Abbreviating *swap* with *s* and *data* with *d*.

Coping with Non-monotonicity

- Let's try to process incrementally:

$$\begin{array}{lll} & swap(X,Y,P) & \\ pivotSwap\ @ & \wr data(X,D) \mid D \geq P \int_{D \in Xs} & \Longleftrightarrow \quad \wr data(Y,D) \int_{D \in Xs} \\ & \wr data(Y,D) \mid D < P \int_{D \in Ys} & \wr data(X,D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \langle [d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6), s(a,b,5) \int \rangle \\ \mapsto_\omega & \langle [d(b,6), s(a,b,5), d(a,7), \boxed{d(b,2)}, d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \ldots & \end{array}$$

- Valid derivation w.r.t. operational semantics ($\wr data(Y,D) \mid D < P \int_{D \in Ys}$ such that $Ys = \varnothing$)
- But what about $\boxed{d(b,2)}$? Its not yet visible in the store!

* Abbreviating *swap* with *s* and *data* with *d*.

Coping with Non-monotonicity

- Let's try to process incrementally:

$$\begin{array}{rll} & swap(X,Y,P) & \\ pivotSwap\ @ & \wr data(X,D) \mid D \geq P \int_{D \in Xs} & \Longleftrightarrow \quad \wr data(Y,D) \int_{D \in Xs} \\ & \wr data(Y,D) \mid D < P \int_{D \in Ys} & \wr data(X,D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \langle [d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ d(a,1), d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ d(a,6), s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6) \int \rangle \\ \mapsto_\omega & \langle [\mathtt{act}\ s(a,b,5), d(a,7), d(b,2), d(b,8)] \ ; \ \wr d(a,1), d(a,6), s(a,b,5) \int \rangle \\ \mapsto_\omega & \langle [d(b,6), s(a,b,5), d(a,7), \boxed{d(b,2)}, d(b,8)] \ ; \ \wr d(a,1) \int \rangle \\ \ldots & \end{array}$$

- This is not sound w.r.t. abstract semantics!

$$\begin{array}{rl} & \wr d(a,1), d(a,6), s(a,b,5), d(a,7), \boxed{d(b,2)}, d(b,8) \int \\ \not\mapsto_\alpha & \wr d(a,1), d(b,6), d(a,7), \boxed{d(b,2)}, d(b,8) \int \end{array}$$

- $\wr data(Y,D) \mid D < P \int_{D \in \varnothing}$ is non-maximal! $\boxed{d(b,2)}$ was omitted!

* Abbreviating *swap* with *s* and *data* with *d*.

Coping with Non-monotonicity

$\mathcal{P} \rhd St \mapsto_\alpha St'$ does not imply $\mathcal{P} \rhd \wr St, St'' \int \mapsto_\alpha \wr St', St'' \int$

Conditional monotonicity

If St'' contains only constraints *monotone w.r.t.* $\mathcal{P}$
then $\mathcal{P} \rhd St \mapsto_\alpha St'$ implies $\mathcal{P} \rhd \wr St, St'' \int \mapsto_\alpha \wr St', St'' \int$

- A constraint $p(\vec{t})$ is *monotone* w.r.t. $\mathcal{P}$ (denoted $\mathcal{P} \stackrel{\triangle}{=}^{\neg}_{\mathbf{unf}} p(\vec{t})$), *iff* $p(\vec{t})$ cannot be unified with any head comprehension patterns in $\mathcal{P}$

Monotone Constraints: $\mathcal{P} \triangleq^{\neg}_{\mathbf{unf}} C$

- $g \rhd C \sqsubseteq_{\mathbf{unf}} M$: a constraint that matches C may match M under guard g:

$$g \rhd p(\vec{t}) \sqsubseteq_{\mathbf{unf}} \wr p'(\vec{t'}) \mid g' \int \text{ iff } \theta p(\vec{t}) \equiv \theta p'(\vec{t'}), \models \theta(g' \wedge g) \text{ for some } \theta$$

$$g'' \rhd \wr p(\vec{t}) \mid g \int \sqsubseteq_{\mathbf{unf}} \wr p'(\vec{t'}) \mid g' \int \text{ iff } \theta p(\vec{t}) \equiv \theta p'(\vec{t'}), \models \theta(g'' \wedge g' \wedge g) \text{ for some } \theta$$

- $\mathcal{P} \triangleq^{\neg}_{\mathbf{unf}} C$: for each rule $r \mathbin{@} \bar{H}_p \setminus \bar{H}_s \iff g \mid \bar{B} \in \mathcal{P}$, there is no comprehension pattern M in $\wr \bar{H}_p, \bar{H}_s \int$ s.t. $g \rhd C \sqsubseteq_{\mathbf{unf}} M$

Outline

Processing constraints incrementally in CHR^{cp}

- Operational semantics of CHR^{cp} extends [Duck et al., 2004] by
  - Identify (non-)monotone constraints (s.t., $\mathcal{P} \stackrel{\triangle}{=}^{\neg}_{\mathbf{unf}} C$)
  - Incrementally store *monotone* constraints
  - Eagerly store *non-monotone* constraints
- Hence we exploit conditional monotonicity where possible (the glass is half-full)

Processing constraints incrementally in CHR^{cp}

$$\textit{pivotSwap} \ @ \ \begin{array}{l} \textit{swap}(X, Y, P) \\ \wr \textit{data}(X, D) \mid D \geq P \int_{D \in Xs} \\ \wr \textit{data}(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \\ \wr \textit{data}(Y, D) \int_{D \in Xs} \\ \wr \textit{data}(X, D) \int_{D \in Ys} \end{array}$$

$$\langle \texttt{[init} \ \wr d(a, 6), s(a, b, 5), d(a, 7), d(b, 2) \int \texttt{]} \ ; \ \wr \int \rangle$$

$\Longrightarrow$ *Initial store is contained in an "*$\mathtt{init}$*" goal*

Processing constraints incrementally in CHR^{cp}

$$\textit{pivotSwap} \ @ \ \begin{array}{l} \textit{swap}(X, Y, P) \\ \wr \textit{data}(X, D) \mid D \geq P \int_{D \in Xs} \\ \wr \textit{data}(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \\ \wr \textit{data}(Y, D) \int_{D \in Xs} \\ \wr \textit{data}(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \langle [\texttt{init} \ \wr d(a, 6), s(a, b, 5), d(a, 7), d(b, 2) \int] \ ; \ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager} \ d(a, 6), \texttt{lazy} \ s(a, b, 5), \texttt{eager} \ d(a, 7), \texttt{eager} \ d(b, 2)] \\ & ; \wr d(a, 6), d(a, 7), d(b, 2) \int \rangle \end{array}$$

$\Longrightarrow$ *Non-monotone constraints (e.g., $d(\_, \_)$) are eagerly stored*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \langle [\texttt{init}\ \wr d(a,6), s(a,b,5), d(a,7), d(b,2) \int] \ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *Eager goal is activated. Already added: No changes to store*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \langle [\texttt{init}\ \wr d(a,6), s(a,b,5), d(a,7), d(b,2) \int] \ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *No match to pivotSwap, hence we drop* $\texttt{act}\ d(a,6)$

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \langle [\texttt{init}\ \wr d(a,6), s(a,b,5), d(a,7), d(b,2) \int] \ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr s(a,b,5), d(a,6), d(a,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *Lazy goal is activated. We add* $s(a, b, 5)$ *to the store*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \langle [\texttt{init}\ \wr d(a,6), s(a,b,5), d(a,7), d(b,2) \int] \ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ d(a,6), \texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{lazy}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{act}\ s(a,b,5), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ; \wr s(a,b,5), d(a,6), d(a,7), d(b,2) \int \rangle \\ \mapsto_\omega & \langle [\texttt{init}\ \wr d(b,6), d(b,7), d(b,2) \int, \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \ ;\ \wr \int \rangle \\ & \ldots \end{array}$$

$\Longrightarrow$ *Apply pivotSwap: rule body added as an "*$\texttt{init}$*" goal. Notice that* $d(b,2)$ *is visible, hence maximality is guaranteed*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{rl} & \ldots \\ \mapsto_\omega & \langle \texttt{[init} \wr d(b, 6), d(b, 7), d(b, 2) \int, \texttt{eager}\ d(a, 7), \texttt{eager}\ d(b, 2) \texttt{]}\ ;\ \wr \int \rangle \end{array}$$

$\Longrightarrow$ *Process "init" goal*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \ldots \\ \mapsto_\omega & \langle [\texttt{init}\ \wr d(b,6), d(b,7), d(b,2) \int, \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)]\ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(b,6), \texttt{eager}\ d(b,7), \texttt{eager}\ d(b,2), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & ;\ \wr d(b,6), d(b,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *All constraints in rule body are non-monotone*

Processing constraints incrementally in CHR^{cp}

$$\begin{array}{rl} & swap(X, Y, P) \\ pivotSwap\ @ & \wr data(X, D) \mid D \geq P \int_{D \in Xs} \\ & \wr data(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \wr data(Y, D) \int_{D \in Xs} \\ \wr data(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \ldots \\ \mapsto_\omega & \langle [\texttt{init}\ \wr d(b,6), d(b,7), d(b,2) \int, \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)]\ ;\ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager}\ d(b,6), \texttt{eager}\ d(b,7), \texttt{eager}\ d(b,2), \texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)] \\ & \ ;\ \wr d(b,6), d(b,7), d(b,2) \int \rangle \\ \mapsto_\omega^* & \langle [\texttt{eager}\ d(a,7), \texttt{eager}\ d(b,2)]\ ;\ \wr d(b,6), d(b,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *No match for* $d(b,6), d(b,7)$ *and* $d(b,2)$*: we drop the goals*

Processing constraints incrementally in CHR^{cp}

$$\textit{pivotSwap} \ @ \ \begin{array}{l} \textit{swap}(X, Y, P) \\ \wr \textit{data}(X, D) \mid D \geq P \int_{D \in Xs} \\ \wr \textit{data}(Y, D) \mid D < P \int_{D \in Ys} \end{array} \iff \begin{array}{l} \\ \wr \textit{data}(Y, D) \int_{D \in Xs} \\ \wr \textit{data}(X, D) \int_{D \in Ys} \end{array}$$

$$\begin{array}{ll} & \ldots \\ \mapsto_\omega & \langle [\texttt{init} \ \wr d(b,6), d(b,7), d(b,2) \int, \texttt{eager} \ d(a,7), \texttt{eager} \ d(b,2)] \ ; \ \wr \int \rangle \\ \mapsto_\omega & \langle [\texttt{eager} \ d(b,6), \texttt{eager} \ d(b,7), \texttt{eager} \ d(b,2), \texttt{eager} \ d(a,7), \texttt{eager} \ d(b,2)] \\ & \ ; \ \wr d(b,6), d(b,7), d(b,2) \int \rangle \\ \mapsto_\omega^* & \langle [\texttt{eager} \ d(a,7), \texttt{eager} \ d(b,2)] \ ; \ \wr d(b,6), d(b,7), d(b,2) \int \rangle \\ \mapsto_\omega^* & \langle [] \ ; \ \wr d(b,6), d(b,7), d(b,2) \int \rangle \end{array}$$

$\Longrightarrow$ *$d(a,7)$ and $d(b,2)$ no longer exists: we drop them as well*

Refined Operational Semantics of CHR^{cp}

- Extends the refined operational semantics of *CHR* [Duck et al., 2004]
- Supports incremental processing of *monotone* constraints
- For each rule in $\mathcal{P}$, we identify which $p(\vec{t}) \in \bar{B}$ is monotone
  - Non-monotone constraints are made visible immediately
  - Monotone constraints can be incrementally stored as usual
- Sound w.r.t. abstract semantics of CHR^{cp}
- See paper for details!
  - See TR for even more details!!

Outline

Current Status of CHR^{cp}

- Prototype implementation
  - Available for download at:
    `https://github.com/sllam/chrcp`
  - Compiler implemented in Python
  - Generates C++ codes that implements CHR^{cp} run-time
  - Utilizes Z3 SMT solver for type checking and monotonicity testing [Lam and Cervesato, 2014b] (i.e., $\mathcal{P} \stackrel{\triangle}{=}^{\neg}_{\mathbf{unf}} C$).
- See technical report [Lam and Cervesato, 2014a] for
  - more examples
  - preliminary experimental results
  - optimized compilation details (e.g., join ordering, etc)

Related Works

- Aggregates in CHR [Sneyers et al., 2007]
- CHR with Negation as Absence [Weert et al., 2006]
- Meld [Cruz et al., 2012]

Conclusion

- CHR^{cp}: Constraint Handling Rules with Multiset Comprehension Patterns
- Abstract semantics: $\mapsto_\alpha$
- Operational semantics: $\mapsto_\omega$
  - supports partial incremental processing of constraints
  - sound w.r.t. abstract semantics

Bibliography

Cruz, F., Ashley-Rollman, M. P., Goldstein, S. C., Rocha, R., and Pfenning, F. (2012).
Bottom-Up Logic Programming for Multicores.
In Costa, V. S., editor, *Proc. of DAMP 2012*. ACM Digital Library.

Duck, G. J., Stuckey, P. J., de la Banda, M. G., and Holzbaur, C. (2004).
The Refined Operational Semantics of Constraint Handling Rules.
In *In 20th Int. Conf. on Logic Programming ICLP'04*, pages 90–104. Springer.

Lam, E. S. L. and Cervesato, I. (2014a).
Optimized Compilation of Multiset Rewriting with Comprehensions (Full-Version).
Technical Report CMU-CS-14-119, Carnegie Mellon University.

Lam, E. S. L. and Cervesato, I. (2014b).
Reasoning about Set Comprehension.
In *SMT'14*.

Sneyers, J., Weert, P. V., Schrijvers, T., and Demoen, B. (2007).
Aggregates in Constraint Handling Rules.
In *ICLP*, pages 446–448.

Weert, P. V., Sneyers, J., Schrijvers, T., and Demoen, B. (2006).
Extending CHR with Negation as Absence.
In *CHR Workshop*, pages 125–140.