Justifications in Constraint Handling
Rules for Logical Retraction in Dynamic Algorithms
Constraint Handling Rules with Justifications

Justifications

- Mark derived information explicitly
- Track origin of information
- Logical Retraction
  - Conclusions can be withdrawn by retracting their premises

Goal

- Extend Constraint Handling Rules (CHR) with justifications (CHR\textsuperscript{J})
- Operational equivalence of rule applications
- Logical retraction
- Correctness and confluence of retraction
- Proof-of-concept implementation
Constraint Handling Rules with Justifications

Justifications

▶ Mark derived information explicitly
▶ Track origin of information
▶ Logical Retraction
  ▶ Conclusions can be withdrawn by retracting their premises

Goal

▶ Extend Constraint Handling Rules (CHR) with justifications (CHR$^J$)
  ▶ Operational equivalence of rule applications
▶ Logical retraction
  ▶ Correctness and confluence of retraction
▶ Proof-of-concept implementation
Constraint Handling Rules with Justifications – Example

Minimum

\[
\min(N) \setminus \min(M) \iff N < M \mid \text{true.}
\]

Example

\[
\min(1)^{f_1}, \min(0)^{f_2}, \min(2)^{f_3}
\]

- \( c^F : F \) is set of justifications for constraint \( c \)
Constraint Handling Rules with Justifications – Example

Minimum

\( \min(N) \backslash \min(M) \iff N<M \mid \text{true.} \)

Example

\[
\begin{align*}
\min(1)^{\{f_1\}}, \min(0)^{\{f_2\}}, \min(2)^{\{f_3\}} & \\
\Rightarrow & \ \text{rem}(\min(1)^{\{f_1\}}, \min(2)^{\{f_3\}}, \min(0)^{\{f_2\}})
\end{align*}
\]

- \( c^F : F \) is set of justifications for constraint \( c \)
Constraint Handling Rules with Justifications – Example

**Minimum**

\[
\text{min}(N) \ \backslash \ \text{min}(M) \iff N < M \ | \ \text{true}.
\]

**Example**

\[
\begin{align*}
\text{min}(1)^{\{f_1\}}, \text{min}(0)^{\{f_2\}}, \text{min}(2)^{\{f_3\}} \\
\leftrightarrow \text{rem}(\text{min}(1)^{\{f_1\}})^{\{f_1, f_2\}}, \text{min}(2)^{\{f_3\}}, \text{min}(0)^{\{f_2\}} \\
\leftrightarrow \text{rem}(\text{min}(1)^{\{f_1\}})^{\{f_1, f_2\}}, \text{rem}(\text{min}(2)^{\{f_3\}})^{\{f_2, f_3\}}, \text{min}(0)^{\{f_2\}}
\end{align*}
\]

- \(c^F: F \) is set of justifications for constraint \(c\)
Constraint Handling Rules with Justifications – Example

Minimum

\[ \min(N) \setminus \min(M) \iff N < M \mid \text{true.} \]

Example

\[ \{ \min(1)^{f_1}, \min(0)^{f_2}, \min(2)^{f_3} \} \]

\[ \leftrightarrow \rem(\min(1)^{f_1})^{f_1, f_2}, \min(2)^{f_3}, \min(0)^{f_2} \]

\[ \leftrightarrow \rem(\min(1)^{f_1})^{f_1, f_2}, \rem(\min(2)^{f_3})^{f_2, f_3}, \min(0)^{f_2} \]

- \( c^F \): \( F \) is set of justifications for constraint \( c \)
- Constraint \( \min(0) \) remained
- Constraints \( \min(1) \) and \( \min(2) \) have been removed
- Constraint with justification \( f_2 \) reason for removal
Constraint Handling Rules (CHR)

- Constraints: first-order logic predicates
Constraint Handling Rules (CHR)

- Constraints: first-order logic predicates

Rules

\[ H_k \setminus H_r \leftrightarrow G \mid B. \]

- \( H_r \): removed heads (only user-defined constraints)
- \( H_k \): kept heads (only user-defined constraints)
- \( G \): guard (only built-in constraints)
- \( B \): body (user-defined and built-in constraints)

- Constraints that match head and satisfy guard are removed/kept
- Body is added
CHR with Justifications (CHR$^J$)

Original Rule

$$r : \bigwedge_{i=1}^{l} K_i \setminus \bigwedge_{j=1}^{m} R_j \iff C \mid \bigwedge_{k=1}^{n} B_k$$

Translated Rule

$$rf : \bigwedge_{i=1}^{l} K_i^{F_i} \setminus \bigwedge_{j=1}^{m} R_j^{F_j} \iff C \mid \bigwedge_{j=1}^{m} \text{rem}(R_j^{F_j})^F \land \bigwedge_{k=1}^{n} B_k^F$$

where $F = \bigcup_{i=1}^{l} F_i \cup \bigcup_{j=1}^{m} F_j$.

- $F_i$ and $F_j$ fresh variables that match justification sets
- Each CHR constraint in body annotated with union of all justifications
CHR with Justifications (CHR$^J$)

Original Rule – Short Hand Notation

\[ r : H_1 \setminus H_2 \Leftrightarrow C \mid B. \]

Translated Rule – Short Hand Notation

\[ rf : H_1^J \setminus H_2^J \Leftrightarrow C \mid \text{rem}(H_2)^J \land B^J. \]
**CHR with Justifications (CHR$^J$)**

**Original Rule – Short Hand Notation**

$$ r : H_1 \setminus H_2 \iff C \mid B. $$

**Translated Rule – Short Hand Notation**

$$ rf : H_1^J \setminus H_2^J \iff C \mid rem(H_2)^J \land B^J. $$

**Lemma (Equivalence of Program Rules)**

The following two propositions are equivalent:

- There is a computation step with simpagation rule $r : S \mapsto_r T$.
- There is a computation step with justifications $S^J \mapsto_{rf} T^J$ with corresponding rule with justifications $rf$. 
Logical Retraction

Idea

- Remove CHR constraint from computation without recomputation from scratch
- All consequences due to rule applications using this constraint are undone
- Remove CHR constraints added by those rules
- Re-add CHR constraints removed by those rules
Logical Retraction

Rules for Retraction

For each constraint $c/n$: 

$$\text{kill : } kill(f) \setminus G^F \iff f \in F \mid \text{true}$$

$$\text{revive : } kill(f) \setminus \text{rem}(G^{F_c})^F \iff f \in F \mid G^{F_c}$$

where 

- $G = c(X_1, \ldots, X_n)$,
- $X_1, \ldots, X_n$ are different variables.

- Constraint may be revived and subsequently killed:
  - if $F_c$ and $F$ contain justification $f$
Confluence of Logical Retraction

Confluence
Confluence of Logical Retraction

![Confluence Diagram]
Confluence of Logical Retraction

Confluence
Confluence of Logical Retraction

Confluence
Confluence of Logical Retraction

Confluence in CHR

- Decidable criterion for terminating programs
Confluence of Logical Retraction

Confluence in CHR

▶ Decidable criterion for terminating programs

▶ Two rules: \( r_1 \) and \( r_2 \)
## Confluence of Logical Retraction

### Confluence in CHR

- Decidable criterion for terminating programs
- Two rules: $r_1$ and $r_2$

- **Overlap states**: Overlap heads and guard of rules
Confluence of Logical Retraction

Confluence in CHR

- Decidable criterion for terminating programs
- Two rules: $r_1$ and $r_2$

- **Overlap states**: Overlap heads and guard of rules
- **Critical pairs**: Apply rules to overlap state
Confluence of Logical Retraction

Confluence in CHR

- Decidable criterion for terminating programs
- Two rules: $r_1$ and $r_2$

- Overlap states: Overlap heads and guard of rules
- Critical pairs: Apply rules to overlap state
- If all critical pairs joinable, the program is confluent
Confluence of Logical Retraction

Idea

▶ Intuitively: Rules for retraction do not interfere with each other
▶ Additional rules do not break potential confluence of a program
▶ It does not make a difference if a justification is retracted immediately or if other rules are applied first
Confluence of Logical Retraction

Idea

- Intuitively: Rules for retraction do not interfere with each other
- Additional rules do not break potential confluence of a program
- It does not make a difference if a justification is retracted immediately or if other rules are applied first

Theorem (Confluence of Logical Retraction)

- CHR program translated to rules with justifications together with kill and revive rules
- At most one kill(f) constraint for each justification f in any state
- Critical pairs between kill and revive joinable
- Critical pairs of those rules with any translated rule with justifications joinable
Confluence of Logical Retraction

Proof Idea: kill and translated rules

\[ \text{kill} : \text{kill}(f) \setminus G^F \iff f \in F \mid \text{true}. \]

\[ \text{rf} : K^J \setminus R^J \iff C \mid \text{rem}(R)^J \land B^J. \]

\[ \text{kill}(f) \land K^J \land R^J \land E \]
Confluence of Logical Retraction

Proof Idea: kill and translated rules

\[
\text{kill} : \text{kill}(f) \setminus G^F \iff f \in F \mid \text{true}.
\]

\[
\text{rf} : K^J \setminus R^J \iff C \mid \text{rem}(R)^J \wedge B^J.
\]

\[
kil(f) \wedge K^J \wedge R^J \wedge E
\]

\[
\text{kill}(f) \wedge E \wedge ((K^J \wedge R^J) - G^F)
\]
Confluence of Logical Retraction

Proof Idea: kill and translated rules

\[
\text{kill} : \quad \text{kill}(f) \setminus G^F \iff f \in F \mid \text{true}.
\]

\[
\text{rf} : \quad K^J \setminus R^J \iff C \mid \text{rem}(R)^J \land B^J.
\]

\[
\text{kill}(f) \land K^J \land R^J \land E
\]

kill

rf

\[
\text{kill}(f) \land E \land ((K^J \land R^J) - G^F)
\]

\[
\text{kill}(f) \land E \land K^J \land \text{rem}(R)^J \land B^J
\]
Confluence of Logical Retraction

Proof Idea: kill and translated rules

\[ \text{kill : } \text{kill}(f) \setminus G^F \Leftrightarrow f \in F \mid \text{true.} \]

\[ \text{rf : } K^J \setminus R^J \Leftrightarrow C \mid \text{rem}(R)^J \land B^J. \]

\[ \text{kill}(f) \land K^J \land R^J \land E \]

\[ \text{kill} \quad \text{rf} \]

\[ \text{kill}(f) \land E \land ((K^J \land R^J) \setminus G^F) \]

\[ \text{revive}^*, \text{kill}^* \]

\[ \text{kill}(f) \land E \land K^J \land \text{rem}(R)^J \land B^J \]
Correctness of Logical Retraction

Idea

- Result of computation after retraction the same as without adding killed constraint in the first place
Correctness of Logical Retraction

Idea

▶ Result of computation after retraction the same as without adding killed constraint in the first place

Theorem (Correctness of Logical Retraction)

▶ Given computation where $f$ does not occur in $A^J$:

\[
A^J \land G\{f\} \land \text{kill}(f)
\]
Correctness of Logical Retraction

Idea

- Result of computation after retraction the same as without adding killed constraint in the first place

Theorem (Correctness of Logical Retraction)

- Given computation where $f$ does not occur in $A^J$:
  
  $A^J \land G^{\{f\}} \land kill(f) \rightarrow^* B^J \land rem(R)^J \land kill(f)$
Correctness of Logical Retraction

Idea

- Result of computation after retraction the same as without adding killed constraint in the first place

Theorem (Correctness of Logical Retraction)

- Given computation where $f$ does not occur in $A^J$:
  
  $A^J \land G^\{f\} \land kill(f) \leftrightarrow^* B^J \land rem(R)^J \land kill(f) \not\leftrightarrow kill,revive$
Correctness of Logical Retraction

Idea

- Result of computation after retraction the same as without adding killed constraint in the first place

Theorem (Correctness of Logical Retraction)

- Given computation where $f$ does not occur in $A^J$:
  \[ A^J \land G^f \land \text{kill}(f) \xrightarrow{*} B^J \land \text{rem}(R)^J \land \text{kill}(f) \xrightarrow{kil,revive} \]

- Then there is computation without $G^f$:
  \[ A^J \xrightarrow{*} B^J \land \text{rem}(R)^J \]
Correctness of Logical Retraction

Proof Idea.

- Mapping between computations with a constraint $G^{\{f\}}$ and without
  - Strip away constraints that contain justification $f$ except for $rem$
- For all rules:
  - show that stripped transition of rule application is equivalent to rule application without $G^{\{f\}}$
Implementation

Basic Idea

- Apply translation scheme
- Represent justifications as unbound variables
- \( C \ #: [F_1, F_2, \ldots] : \text{constraint } C \text{ with justifications } F_1, F_2, \ldots \)
- Built-in constraint \( \text{union} \) computes union of justification sets
- For \( \text{kill} \) and \( \text{revive} \): guard \( f \in F \) via \( \text{member}(F, Fs) \)

Optimization

Thom Frühwirth: *Implementation of Logical Retraction in Constraint Handling Rules with Justifications*

Proceedings of the 21st International Conference on Applications of Declarative Programming and Knowledge Management (INAP) September 2017
Minimum Example

Translated Minimum Rule

\[
\min(A)##B \ \land \ \min(C)##D \iff A<C \mid
\]

\[
\text{union}([B,D],E), \ \text{rem}((\min(C)##D)##E).
\]

Example

?- \min(1)##[A], \min(0)##[B], \min(2)##[C].
rem(\min(1)##[A])##[A,B], \rem(\min(2)##[C])##[B,C],
\min(0)##[B].

- **Constraint** \(\min(0)\) remained
- **Constraints** \(\min(1)\) and \(\min(2)\) have been removed
- 0 is minimum
- **Constraint with justification** reason for removal
Minimum Example

Translating Minimum Rule

\[
\begin{align*}
\text{min}(A)##B \setminus \text{min}(C)##D & \iff A < C \\
\text{union}([B,D],E), \text{rem}(%(C)##D)##E.
\end{align*}
\]

Example

?- \text{min}(1)##[A], \text{min}(0)##[B], \text{min}(2)##[C],
    \text{killc}(\text{min}(0)).
\text{rem}\text{(}\text{min}(2)##[C])##[A,C], \text{min}(1)##[A].

- Logically retract current minimum \text{min}(0)
- Constraint \text{min}(0) is removed by binding justification \text{B}.
- The \text{rem} constraints for \text{min}(1) and \text{min}(2) involve \text{B} as well
- The two constraints are re-introduced and react with each other
- \text{min}(2) is now removed by \text{min}(1)
Conclusion

- Source-to-source transformation for CHR with justifications ($\text{CHR}^J$)
- Logical retraction of constraints
- Correctness theorem: If constraint is retracted, computation continues as if constraint was never there
- Confluence theorem: Implementation of retraction with two-rule scheme is confluent
- Online translator:
  
  [http://pmx.informatik.uni-ulm.de/chr/translator/](http://pmx.informatik.uni-ulm.de/chr/translator/)
Future Work

▶ Investigate behavior of logical and classical algorithms with justifications
▶ Programs not required to be confluent, but use with non-confluent programs may lead to unwanted orders of rule applications
▶ Improve implementation further, optimizations, benchmarks
▶ Extend rule scheme to support debugging by explanation
▶ For error diagnosis: detection and repair of inconsistencies
Thank you for your attention!

Questions?