On the Expressive Power of Priorities in CHR

Maurizio Gabbrielli ¹  Jacopo Mauro ¹
Maria Chiara Meo ²

¹Dipartimento di Scienze dell’Informazione, University of Bologna
²Dipartimento di Scienze, University of Chieti-Pescara

CHR Working Week, Ulm 2009
Introduction

Motivations
CHR and CHR
Acceptable encoding

Results

Conclusion
Claim

In [De Koninck et al. - 2007] it is claimed that “priorities do improve the expressivity of CHR”

Our Contribution

- formal ground for this informal claim using a notion of expressivity coming from the field of concurrency theory
- dynamic priorities do not augment the expressivity
CHR

Constraint Handling Rules is a high-level programming language based on multi-headed, committed-choice, guarded multiset rewrite rules.

Thom Frühwirth

CHR

CHR$^{rp}$

CHR$^{rp}$ extends CHR with user-defined priorities.
CHR - syntax

- two types of constraints
  - *CHR constraints* or User defined constraints
  - *Built-in constraints* (we assume a given constraint theory which describes their meaning)

- three types of rules

  propagation \( r@H \Rightarrow C \mid B \)

  simplification \( r@H' \Leftrightarrow C \mid B \)

  simpagation \( r@H \setminus H' \Leftrightarrow C \mid B \)

- a program: sequence of rules
- a goal: multiset or sequence of constraints
CHR\(^{rp}\) - syntax

- Priorities (\(p\)) are arithmetic expressions.
- The rules are extended with priorities in the following way:
  - **Propagation**: \(p :: r@H \Rightarrow C | B\)
  - **Simplification**: \(p :: r@H' \Leftrightarrow C | B\)
  - **Simpagation**: \(p :: r@H \setminus H' \Leftrightarrow C | B\)

- If a priority has a variable then it is dynamic, static otherwise.
three different operational semantics considered:

- $\omega_t$ - the traditional semantics for CHR
  
  the rule

\[
  r \odot H \setminus H' \iff C | B
\]

- can fire if $H \cup H'$ are in the store and $C$ is satisfied
- when fired $H'$ deleted and $B$ added
- propagation rule fires only once
Operational semantics - 2

- $\omega_r$ - the refined semantics for CHR
  - introduced to model the execution mechanism of the current implementations
  - based on active constraints
  - order of the rules and constraints matters

- $\omega_p$ - the traditional semantics for CHR$^{rp}$
  - only rules with highest priority can fire
CHR by example

**Less than or equal program in CHR**

reflexivity @ leq(X, Y) ⇔ X = Y | true
antisymmetry @ leq(X, Y), leq(Y, X) ⇔ X = Y
transitivity @ leq(X, Y), leq(Y, Z) ⇒ leq(X, Z)

**Shortest path program in CHR**

1 :: source(V) ⇒ dist(V, 0)
1 :: dist(V, D₁) \ dist(V, D₂) ⇔ D₁ ≤ D₂ | true
D + 2 :: dist(V, D), edge(V, C, U) ⇒ dist(U, D + C)
Solve $\langle \{c\} \mathbin{\cup} G, S, B, T \rangle_n \overset{\omega_t}{\longrightarrow}_P \langle G, S, c \land B, T \rangle_n$ where $c$ is a built-in constraint.

Introduce $\langle \{c\} \mathbin{\cup} G, S, B, T \rangle_n \overset{\omega_t}{\longrightarrow}_P \langle G, \{c\#n\} \mathbin{\cup} S, B, T \rangle_{n+1}$ where $c$ is a CHR constraint.

Apply $\langle G, H_1 \mathbin{\cup} H_2 \mathbin{\cup} S, B, T \rangle_n \overset{\omega_t}{\longrightarrow}_P \langle C \mathbin{\cup} G, H_1 \mathbin{\cup} S, \theta \land B, T \mathbin{\cup} \{t\} \rangle_n$ where $P$ contains a (renamed apart) rule

$$r \ominus H_1 \setminus H_2 \iff g \mid C$$

and there exists a matching substitution $\theta$ s.t. $\text{chr}(H_1) = \theta H'_1$, $\text{chr}(H_2) = \theta H'_2$, $CT \models B \rightarrow \exists_{FV(B)}(\theta \land g)$ and $t = \text{id}(H_1) ++ \text{id}(H_2) ++ [r] \notin T$
\[ \omega_p \text{ semantics} \]

**Solve** \( \langle \{c\} \cup G, S, B, T \rangle_n \xrightarrow{\omega_p} P \langle G, S, c \land B, T \rangle_n \) where \( c \) is a built-in constraint

**Introduce** \( \langle \{c\} \cup G, S, B, T \rangle_n \xrightarrow{\omega_p} P \langle G, \{c\#n\} \cup S, B, T \rangle_{n+1} \) where \( c \) is a CHR constraint

**Apply** \( \langle \emptyset, H_1 \cup H_2 \cup S, B, T \rangle_n \xrightarrow{\omega_p} P \langle C, H_1 \cup S, \theta \land B, T \cup \{t\} \rangle_n \) where \( P \) contains a (renamed apart) rule

\[ p :: r @H_1 \backslash H_2 \iff g | C \]

and there exists a matching substitution \( \theta \) s.t. \( \text{chr}(H_1) = \theta H_1' \), \( \text{chr}(H_2) = \theta H_2' \), \( C \mathcal{T} \models B \rightarrow \exists_{Fv(B)}(\theta \land g) \) and \( t = \text{id}(H_1) \oplus \text{id}(H_2) \oplus [r] \notin T \). Furthermore no rule of priority \( p' \) and substitution \( \theta' \) exists with \( \theta' p' < \theta p \) for which the above conditions hold.
initial configuration: the goal constraints are added into the store

two final configuration:
  - failed (constraints in the store are unsatisfiable)
  - terminated (no rule can fire)

observables are the data sufficient answers: terminated configurations that contain only built-in constraints
Acceptable encoding

- language encoding with additional proprieties to fulfill
- motivation: discriminating differing (Turing powerful) languages
- in our work we require
  1. the observables remain the same
  2. compositionality of the goal encoding w.r.t. the conjunction of atoms
CHR vs CHR\(^{rp}\)

**Theorem**

There exists no acceptable encoding of CHR\(^{rp}\) in CHR

- idea of the proof:
  - considered the Last Man Standing Problem (LMS problem)
  - solved the problem in CHR\(^{rp}\)
  - shown that LMS can not be solved in CHR (under acceptability assumption)

**LMS problem solved in CHR\(^{rp}\)**

1. \(a(X), a(X) \iff X = \text{no}\)
2. \(a(X) \iff X = \text{no}\|\text{true}\)
3. \(a(X) \iff X = \text{yes}\)
Theorem

There exists no acceptable encoding of $CHR_{\omega_r}$ into $CHR_{\omega_t}$

- proof idea: using the LMS problem like in the previous case

LMS Program in CHR with $\omega_r$ semantics

\[
\begin{align*}
    a(X) & \iff X = no \mid true \\
a(X) & \iff X = yes \mid false \\
d(X), b(X), a(X) & \iff X = no \\
a(X) & \iff b(Y), b(X), c(X) \\
c(X), b(Y) & \iff Y = yes, d(X) \\
d(X), b(Y) & \iff X = yes \mid true
\end{align*}
\]
Static vs dynamic priorities

Theorem
There is an acceptable encoding of CHR$^{rp}$ with dynamic priorities into CHR$^{rp}$ with static priorities

- encoding idea: instead of one rule execution
  1. detect which rules have the higher priority
  2. fire only one of these rules
- assumed that equalities and inequalities can be used as built in constraints
CHR vs Prolog

- result: no acceptable encoding from CHR to Prolog (extension of a previous result [Di Giusto et al. 2009])
- Prolog program are considered w.r.t. the computed answer semantics
- assumed that no dynamic procedures are used
- an acceptable encoding from CHR to Prolog
  - preserves the compositionality of the goal
  - the Prolog program has no computed answers iff the CHR program has an empty data sufficient answer
Conclusions

- we use the notion of acceptable encoding for studying the expressivity of CHR languages
- we proved that priorities improve the expressivity of CHR
- we proved that the refined semantics improve the expressivity of CHR considered with the traditional semantics
- we proved that dynamic priorities do not augment the expressivity of CHR with static priorities
- we extend a previous result showing that CHR cannot be encoded in Prolog
Future Work

We plan to

- investigate the relation between priorities and negation as absence
- consider the refined semantics for CHR$^{rp}$
- consider data qualified answers instead of data sufficient answers