

The Rules of CONJURE

Chris Jefferson, Bernadette Martínez Hernández,
Ian Miguel and Alan M. Frisch

University of York
March 23, 2005

1 Introduction

This technical report contains the refinement equations and rules used by the constraint refinement system CONJURE [1]. CONJURE is a system which transforms high-level specifications of constraint programs in the language ESSENCE into a low-level language ESSENCE', which is similar to existing constraint satisfaction solvers.

2 Type Refinement Rules

These rules refine a single variable, parameter or constant. The rules for set and multiset types are never actually invoked, as these types are always refined during the refinement of some other rule. They are however provided because they demonstrate the extra constraints which must be added to each representation in order to ensure it is correct. In later sections when the similar equations differ only by the addition of these constraints, then only a sample of the equations will be given.

2.1 Symbols of Atomic Type

Variables of type boolean and integer do not need any further refinement.

$$\begin{array}{l} \text{BOOLEANVARIABLE } \rho(i:\text{bool}) \xrightarrow{\text{ref}} \\ \{ i:\text{bool} \} | \\ \text{INTEGERVARIABLE } \rho(i:\text{integer}(D)) \xrightarrow{\text{ref}} \\ \{ i:\text{integer}(D) \} | \end{array}$$

2.2 Symbols of Matrix Type

The MATRIXREFINE equation refines a matrix by refining a typical element of the matrix, and then duplicating the resulting local model and refined variables for each element of the matrix.

$$\text{MATRIXREFINE } \rho(M: \text{matrix (indexed by } \tau_1 \text{) of } \tau_2) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} M': \text{matrix (indexed by } \tau_1 \text{) of } \tau_2' \\ \text{such that } \forall i: \tau_1. \Gamma \\ | \\ e' \text{ with local model } \Gamma \in \rho(e) \end{array} \right| \begin{array}{l} \tau_1 \text{ is indexable} \\ e = \text{genSymbol}(M[i], \tau) \\ e': \tau_2' \\ M': \text{matrix (indexed by } \tau_1 \text{) of } _ \end{array} \right\}$$

2.3 Symbols of Set Type

Refines a set of τ , maximum size n to the *occurrence* representation.

$$\text{BOUNDEDSET1 } \rho(S: \text{set (maxsize } n \text{) of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} S'' \\ \text{such that } \phi \\ \text{represent } S \text{ by } \text{occset}(S') \\ | \\ \phi \in \rho(\sum S' \leq n) \\ S'' \in \rho(S') \end{array} \right| \begin{array}{l} \tau \text{ is indexable} \\ S' = \text{genSymbol}(S, \text{matrix (indexed by } \tau \text{) of bool}) \\ S'': \text{matrix (indexed by } _ \text{) of bool} \end{array} \right\}$$

Refines a set of τ , maximum size n to the *variable explicit* representation.

$$\text{BOUNDEDSET2 } \rho(S: \text{set (maxsize } n \text{) of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \langle S'', \text{Switches} \rangle \\ \text{such that } \phi \\ \text{represent } S \text{ by } \text{varexpset}(\langle S', \text{Switch} \rangle) \\ \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle.1) \\ | \\ \phi \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n). \\ (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'[i] \neq S'[j]) \\ S'' \in \rho(S') \end{array} \right| \begin{array}{l} S' = \text{genSymbol}(S, \text{matrix (indexed by } \text{integer}(1..n) \text{) of } \tau) \\ \text{Switch} = \\ \text{genSymbol}(S, \text{matrix (indexed by } \text{integer}(1..n) \text{) of bool}) \\ S'': \text{matrix (indexed by } \text{integer}(1..n) \text{) of } _ \end{array} \right\}$$

Refines a set of τ of size n to the *occurrence* representation.

$$\text{SIZEDSET1 } \rho(S: \text{set (size } n \text{) of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} S'' \\ \text{such that } \phi \\ \text{represent } S \text{ by } \text{occset}(S') \\ | \\ \phi \in \rho(\sum S' = n) \\ S'' \in \rho(S') \end{array} \right| \begin{array}{l} \tau \text{ is indexable} \\ S' = \text{genSymbol}(S, \text{matrix (indexed by } \tau \text{) of bool}) \\ S'': \text{matrix (indexed by } _ \text{) of bool} \end{array} \right\}$$

Refines a set of τ , maximum size n to the *occurrence* representation.

$\text{SIZEDSET2 } \rho(S:\text{set (size } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ S''$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by expset}(S')$ $\quad \text{symmetry } \text{indexsym}(S'.1)$ $\quad $ $\quad \phi \in \rho(\text{AllDiff}(S'))$ $\quad S'' \in \rho(S')$ $\}$	$S' = \text{genSymbol}(S, \text{matrix (indexed by integer}(1..n)) \text{ of } \tau)$ $S'': \text{matrix (indexed by } _ \text{) of bool}$
---	---

Refines a set of τ of size n to the *explicit* representation.

$\text{BOUNDEDSET2 } \rho(S:\text{set (maxsize } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ \langle S'', \text{Switches} \rangle$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by varexpset}(\langle S', \text{Switch} \rangle)$ $\quad \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle.1)$ $\quad $ $\quad \phi \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n).$ $\quad \quad (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'[i] \neq S'[j])$ $\quad S'' \in \rho(S')$ $\}$	$S' = \text{genSymbol}(S, \text{matrix (indexed by integer}(1..n)) \text{ of } \tau)$ $\text{Switch} =$ $\quad \text{genSymbol}(S, \text{matrix (indexed by integer}(1..n)) \text{ of bool})$ $S'': \text{matrix (indexed by integer}(1..n)) \text{ of } _$
---	--

Refines a set of τ to the *occurrence* representation.

$\text{SET1 } \rho(S:\text{set of } \tau) \xrightarrow{\text{ref}}$ $\{ S''$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by occset}(S')$ $\quad $ $\quad S'' \in \rho(S')$ $\}$	$\tau \text{ is indexable}$ $S' = \text{genSymbol}(S, \text{matrix (indexed by } \tau) \text{ of bool})$ $S'': \text{matrix (indexed by } _ \text{) of bool}$
---	--

Refines a set of τ to the *variable explicit* representation.

$\text{SET2 } \rho(S:\text{set of } \tau) \xrightarrow{\text{ref}}$ $\{ \langle S'', \text{Switches} \rangle$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by varexpset}(\langle S', \text{Switch} \rangle)$ $\quad \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle.1)$ $\quad $ $\quad \phi \in \rho(\forall i \in \text{integer}(1.. \tau). \forall j \in \text{integer}(1.. \tau).$ $\quad \quad (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'[i] \neq S'[j])$ $\quad S'' \in \rho(S')$ $\}$	$S' = \text{genSymbol}(S, \text{matrix (indexed by integer}(1.. \tau)) \text{ of } \tau)$ $\text{Switch} =$ $\quad \text{genSymbol}(S, \text{matrix (indexed by integer}(1.. \tau)) \text{ of bool})$ $S'': \text{matrix (indexed by integer}(1.. \tau)) \text{ of } _$
---	---

2.4 Variables of Multiset Type

Refines a multiset of τ , maximum size n to the *occurrence* representation.

$\text{BOUNDEDMSET1 } \rho(S:\text{mset(maxsize } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ S''$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by occmset}(S')$ $\quad $ $\quad \phi \in \rho(\sum S' \leq n)$ $\quad S'' \in \rho(S')$ $\}$	$\tau \text{ is indexable}$ $S' = \text{genSymbol}(S, \text{matrix (indexed by } \tau) \text{ of integer}(1..n))$ $S'': \text{matrix (indexed by } _ \text{) of integer}(1..n)$
---	--

Refines a multiset of τ , maximum size n to the *variable explicit* representation.

$\text{BOUNDEDMSET2 } \rho(\text{Smset}(\text{maxsize } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ \langle S'', \text{Switches} \rangle$ $\quad \text{represent } S \text{ by varexpmset}(\langle S', \text{Switch} \rangle)$ $\quad \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle.1)$ $\quad $ $\quad S'' \in \rho(S')$ $\}$	$S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } \tau)$ $\text{Switch} =$ $\quad \text{genSymbol}(S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of bool})$ $S'': \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } _$
---	---

Refines a multiset of τ of size n to the *occurrence* representation.

$\text{SIZEDMSET1 } \rho(\text{Smset}(\text{size } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ S''$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by occmset}(S')$ $\quad $ $\quad \phi \in \rho(\sum S' = n)$ $\quad S'' \in \rho(S')$ $\}$	$\tau \text{ is indexable}$ $S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by } \tau) \text{ of integer}(1..n))$ $S'': \text{matrix}(\text{indexed by } _) \text{ of integer}(1..n)$
--	---

Refines a multiset of τ of size n to the *explicit* representation.

$\text{SIZEDMSET2 } \rho(\text{Smset}(\text{size } n) \text{ of } \tau) \xrightarrow{\text{ref}}$ $\{ S''$ $\quad \text{such that } \phi$ $\quad \text{represent } S \text{ by expset}(S')$ $\quad \text{symmetry } \text{indexsym}(S'.1)$ $\quad $ $\quad S'' \in \rho(S')$ $\}$	$S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } \tau)$ $S'': \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } _$
---	---

3 Rules for Operators on Atomic Symbols

3.1 Operators on Integer types

This rule refines operator $+$ on integer types.

$\text{INTEGERADD } \rho(a:\text{integer}(D_a) + b:\text{integer}(D_b)) \xrightarrow{\text{ref}}$ $\{ (a' + b'):\text{integer}(D_a + D_b)$ $\quad $ $\quad a' \in \rho(a)$ $\quad b' \in \rho(b)$ $\}$	$a':\text{integer}(D_a)$ $b':\text{integer}(D_b)$
--	--

This rule refines operator $-$ with both operands type Integer.

$\text{INTEGERMINUS } \rho(a:\text{integer}(D_a) - b:\text{integer}(D_b)) \xrightarrow{\text{ref}}$ $\{ (a' - b'):\text{integer}(D_a - D_b)$ $\quad $ $\quad a' \in \rho(a)$ $\quad b' \in \rho(b)$ $\}$	$a':\text{integer}(D_a)$ $b':\text{integer}(D_b)$
--	--

This rule refines operator \times with both operands type integer.

$$\begin{array}{c}
\text{INTEGERTIMES } \rho(a:\text{integer}(D_a) \times b:\text{integer}(D_b)) \xrightarrow{\text{ref}} \\
\{ (a' \times b'):\text{integer}(D_a \times D_b) \mid \begin{array}{l} a':\text{integer}(D_a) \\ b':\text{integer}(D_b) \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

This rule refines operator \times with both operands type integer.

$$\begin{array}{c}
\text{INTEGEREQUALITY } \rho(a:\text{integer}(D_a) = b:\text{integer}(D_b)) \xrightarrow{\text{ref}} \\
\{ (a' = b'):\text{bool} \mid \begin{array}{l} a':\text{integer}(D_a) \\ b':\text{integer}(D_b) \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

This rule refines operator $=$ with both operands type integer.

$$\begin{array}{c}
\text{INTEGEREQUALITY } \rho(a:\text{integer}(D_a) \neq b:\text{integer}(D_b)) \xrightarrow{\text{ref}} \\
\{ (a' \neq b'):\text{bool} \mid \begin{array}{l} a':\text{integer}(D_a) \\ b':\text{integer}(D_b) \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

3.2 Operators of Boolean Type

This rule refines the boolean and operator.

$$\begin{array}{c}
\text{BOOLEANAND } \rho(a:\text{bool} \wedge b:\text{bool}) \xrightarrow{\text{ref}} \\
\{ (a' \wedge b'):\text{bool} \mid \begin{array}{l} a':\text{bool} \\ b':\text{bool} \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

This rule refines the boolean or operator.

$$\begin{array}{c}
\text{BOOLEANOR } \rho(a:\text{bool} \vee b:\text{bool}) \xrightarrow{\text{ref}} \\
\{ (a' \vee b'):\text{bool} \mid \begin{array}{l} a':\text{bool} \\ b':\text{bool} \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

This rule refines the boolean equality operator.

$$\begin{array}{c}
\text{BOOLEANEQUALITY } \rho(a:\text{bool} = b:\text{bool}) \xrightarrow{\text{ref}} \\
\{ (a' = b'):\text{bool} \mid \begin{array}{l} a':\text{bool} \\ b':\text{bool} \end{array} \\
| \\
\begin{array}{l} a' \in \rho(a) \\ b' \in \rho(b) \end{array} \\
\}
\end{array}$$

This rule refines the boolean inequality operator.

$$\text{BOOLEANINEQUALITY } \rho(a:\text{bool} \neq b:\text{bool}) \xrightarrow{\text{ref}}$$

$\{ (a' \neq b'):\text{bool}$	$a':\text{bool}$ $b':\text{bool}$
\mid $a' \in \rho(a)$ $b' \in \rho(b)$	
$\}$	

This rule refines the boolean implication operator.

$$\text{BOOLEANIMPLICATION } \rho(a:\text{bool} \leftarrow b:\text{bool}) \xrightarrow{\text{ref}}$$

$\{ (a' \leftarrow b'):\text{bool}$	$a':\text{bool}$ $b':\text{bool}$
\mid $a' \in \rho(a)$ $b' \in \rho(b)$	
$\}$	

This rule refines the boolean not operator.

$$\text{BOOLEANNOT } \rho(\neg a:\text{bool}) \xrightarrow{\text{ref}}$$

$\{ \neg a':\text{bool}$	$a':\text{bool}$
\mid $a' \in \rho(a)$	
$\}$	

4 Unary Constraints on (Multi)sets

4.1 Cardinality Constraints

Refines the constraint $|S|$ for a set maximum size n by transforming S into the *occurrence* representation.

$$\text{BOUNDEDCARDINALITY1 } \rho(|S:\text{set}(\text{maxsize } n) \text{ of } \tau|) \xrightarrow{\text{ref}}$$

$\{ \text{sum}(i \text{ in } \tau) S'$ $\text{such that } \chi$ $\text{represent } S \text{ by occset}(S')$	$S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$
\mid $\chi \in \rho(\sum S' \leq n)$	
$\}$	

Refines the constraint $|S|$ for a set maximum size n by transforming S into the *variable explicit* representation.

$$\text{BOUNDEDCARDINALITY2 } \rho(|S:\text{set}(\text{maxsize } n) \text{ of } \tau|) \xrightarrow{\text{ref}}$$

$\{ \text{sum}(i \text{ in integer}(1..n)) \text{ Switch}$ $\text{such that } \chi$ $\text{represent } S \text{ by varexpset}(\langle S', \text{Switch} \rangle)$	$i = \text{genSymbol}(i, \text{integer}(1..n))$ $S' = \text{genSymbol}S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } \tau$ $\text{Switch} = \text{genSymbol}S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of bool}$
\mid $\chi \in \rho(\text{AllDiff}(S))$	
$\}$	

The rules for sets, multisets and bounded multisets follow similarly.

This rule refines cardinality for a set of known fixed size. The domain returned is a single value, as the size of sets must be either a parameter or constant.

$$\text{SIZEDSETCARDINALITY } \rho(|S:\text{set } (\text{size } n) \text{ of } \tau|) \xrightarrow{\text{ref}} \{ n:\text{integer}(n) \}$$

This rule refines cardinality for a multiset of known fixed size.

$$\text{SIZEDMSETCARDINALITY } \rho(|S:\text{mset}(\text{size } n) \text{ of } \tau|) \xrightarrow{\text{ref}} \{ n:\text{integer}(n) \}$$

5 Binary Constraints on (Multi)sets

5.1 Element Constraint

This rule of the `ELEMENTBOUNDEDSET` equation refines an element constraint by refining the set into the *occurrence* representation.

$$\text{ELEMENTBOUNDEDSET1 } \rho(s:\tau \in S:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}} \left\{ \begin{array}{l} \phi_1 \\ \text{such that } \phi_2 \\ \text{represent } S \text{ by } \text{occset}(S') \\ | \\ \phi_1 \in \rho(S'[s] = 1) \\ \phi_2 \in \rho(\sum S' \leq n) \\ \end{array} \right| \begin{array}{l} \tau \text{ is indexable} \\ S' = \text{genSymbol}(S, \text{matrix } (\text{indexed by } \tau) \text{ of bool}) \end{array}$$

This rule refines an element constraint by refining the set into the *variable explicit* representation.

$$\text{ELEMENTBOUNDEDSET2 } \rho(s:\tau \in S:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}} \left\{ \begin{array}{l} \phi_1 \\ \text{such that } \phi_2 \\ \text{represent } S \text{ by } \text{varexpset}(\langle S', \text{Switch} \rangle) \\ \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle, 1) \\ | \\ \phi_1 \in \rho(\exists x:\text{integer}(1..n). \text{Switch}[x] = 1 \wedge S'[x] = s) \\ \phi_2 \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n). \\ (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'[i] \neq S'[j]) \\ \end{array} \right| \begin{array}{l} S' = \text{genSymbol}(S, \text{matrix } (\text{indexed by } \text{integer}(1..n)) \text{ of } \tau) \\ \text{Switch} = \\ \text{genSymbol}(S, \text{matrix } (\text{indexed by } \text{integer}(1..n)) \text{ of bool}) \\ S'': \text{matrix } (\text{indexed by } \text{integer}(1..n)) \text{ of } _ \end{array}$$

Equations for the other types of set and multiset follow similarly.

5.2 Subset Constraint

This rule refines a subset constraint on bounded sets by refining both the left and right hand side sets to the *occurrence* representation.

$$\text{SUBSETBOUNDEDSET1 } \rho(S_1:\text{set (maxsize } n) \text{ of } \tau \subseteq S_2:\text{set (maxsize } m) \text{ of } \tau) \stackrel{\text{ref}}{\mapsto}$$

$\{ \phi$ $\quad \text{such that } \chi_1 \wedge \chi_2$ $\quad \text{represent } S_1 \text{ by occset}(S'_1)$ $\quad \text{represent } S_2 \text{ by occset}(S'_2)$ $\quad $ $\quad \phi \in \rho(\forall i:\tau. S'_1[i] \rightarrow S'_2[i])$ $\quad \chi_1 \in \rho(\sum S'_1 \leq n)$ $\quad \chi_2 \in \rho(\sum S'_2 \leq m)$ $\}$	$S'_1 = \text{genSymbol}(S_1, \text{matrix (indexed by } \tau) \text{ of bool})$ $S'_2 = \text{genSymbol}(S_2, \text{matrix (indexed by } \tau) \text{ of bool})$
--	--

This rule refines a subset constraint on bounded sets by refining the left hand side set to the *variable explicit* representation and the right hand side to the *occurrence* representation.

$$\text{SUBSETBOUNDEDSET2 } \rho(S_1:\text{set (maxsize } n) \text{ of } \tau \subseteq S_2:\text{set (maxsize } m) \text{ of } \tau) \stackrel{\text{ref}}{\mapsto}$$

$\{ \phi$ $\quad \text{such that } \chi_1 \wedge \chi_2$ $\quad \text{represent } S_1 \text{ by varexpset}((S'_1, \text{Switch}))$ $\quad \text{represent } S_2 \text{ by occset}(S'_2)$ $\quad \text{symmetry indexsym}((S'_1, \text{Switch}).1)$ $\quad $ $\quad \phi \in \rho(\forall k:\text{integer}(1..n). \text{Switch}[k] \rightarrow S'_2[S'_1[k]])$ $\quad \chi_1 \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n).$ $\quad \quad (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'_1[i] \neq S'_1[j])$ $\quad \chi_2 \in \rho(\sum S'_2 \leq n)$ $\}$	$S'_1 = \text{genSymbol}(S_1, \text{matrix (indexed by integer}(1..n)) \text{ of } \tau)$ $\text{Switch} =$ $\quad \text{genSymbol}(S, \text{matrix (indexed by integer}(1..n)) \text{ of bool})$ $S'_2 = \text{genSymbol}(S_2, \text{matrix (indexed by } \tau) \text{ of bool})$ $S'' : \text{matrix (indexed by integer}(1..n)) \text{ of } _$
--	--

This rule refines a subset constraint on bounded sets by refining the left hand side set to the *occurrence* representation and the right hand side to the *variable explicit* representation.

$$\text{SUBSETBOUNDEDSET3 } \rho(S_1:\text{set (maxsize } n) \text{ of } \tau \subseteq S_2:\text{set (maxsize } m) \text{ of } \tau) \stackrel{\text{ref}}{\mapsto}$$

$\{ \phi$ $\quad \text{such that } \chi_1 \wedge \chi_2$ $\quad \text{represent } S_1 \text{ by occset}(S'_1)$ $\quad \text{represent } S_2 \text{ by varexpset}((S'_2, \text{Switch}))$ $\quad \text{symmetry indexsym}((S'_2, \text{Switch}).1)$ $\quad $ $\quad \phi \in \rho(\forall k:\tau. S'_1[k] \rightarrow \exists l:\text{integer}(1..m). (\text{Switch}[l] \wedge S'_2[l] = k))$ $\quad \chi_2 \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n).$ $\quad \quad (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'_2[i] \neq S'_2[j])$ $\quad \chi_1 \in \rho(\sum S'_1 \leq m)$ $\}$	$S'_2 = \text{genSymbol}(S_2, \text{matrix (indexed by integer}(1..n))$ $\text{Switch} =$ $\quad \text{genSymbol}(S, \text{matrix (indexed by integer}(1..n)) \text{ of b}$ $S'_1 = \text{genSymbol}(S_1, \text{matrix (indexed by } \tau) \text{ of bool})$ $S'' : \text{matrix (indexed by integer}(1..n)) \text{ of } _$
---	--

This rule refines a subset constraint on bounded sets by refining both sets to the *variable explicit* representation.

SUBSETBOUNDEDSSET4 $\rho(S_1:\text{set}(\text{maxsize } n) \text{ of } \tau \subseteq S_2:\text{set}(\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$

<pre> { ϕ such that $\chi_1 \wedge \chi_2$ represent S_1 by varexpset($\langle S'_1, \text{Switch}_1 \rangle$) represent S_2 by varexpset($\langle S'_2, \text{Switch}_2 \rangle$) symmetry indexsym($\langle S'_1, \text{Switch}_1 \rangle.1$) symmetry indexsym($\langle S'_2, \text{Switch}_2 \rangle.1$) $\phi \in \rho(\forall k:\text{integer}(1..n). \text{Switch}_1[k] \rightarrow$ $\exists l:\text{integer}(1..m). (\text{Switch}_2[l] \wedge S'_2[l] = S'_1[k]))$ $\chi_1 \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n).$ $(i \neq j \wedge \text{Switch}_1[i] \wedge \text{Switch}_1[j] \rightarrow S'_1[i] \neq S'_1[j])$ $\chi_2 \in \rho(\forall i \in \text{integer}(1..m). \forall j \in \text{integer}(1..m).$ $(i \neq j \wedge \text{Switch}_2[i] \wedge \text{Switch}_2[j] \rightarrow S'_2[i] \neq S'_2[j])$ } </pre>	<pre> $S'_1 = \text{genSymbol}(S_1, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } \tau)$ $\text{Switch}_1 =$ $\text{genSymbol}(S_1, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of bool})$ $S'_2 = \text{genSymbol}(S_2, \text{matrix}(\text{indexed by integer}(1..m)) \text{ of } \tau)$ $\text{Switch}_2 =$ $\text{genSymbol}(S_2, \text{matrix}(\text{indexed by integer}(1..m)) \text{ of bool})$ </pre>
--	--

The equations for the other two types of sets follow similarly.

This rule refines a subset constraint on bounded multisets by refining both the left and right hand side multisets to the *occurrence* representation.

SUBSETBOUNDEDMSET1 $\rho(S_1:\text{mset}(\text{maxsize } n) \text{ of } \tau \subseteq S_2:\text{mset}(\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$

<pre> { ϕ such that $\chi_1 \wedge \chi_2$ represent S_1 by occmset(S'_1) represent S_2 by occmset(S'_2) $\phi \in \rho(\forall i:\tau. S'_1[i] \leq S'_2[i])$ $\chi_1 \in \rho(\sum S'_1 \leq n)$ $\chi_2 \in \rho(\sum S'_2 \leq m)$ } </pre>	<pre> $S'_1 = \text{genSymbol}(S_1, \text{matrix}(\text{indexed by } \tau) \text{ of integer}(1..n))$ $S'_2 = \text{genSymbol}(S_2, \text{matrix}(\text{indexed by } \tau) \text{ of integer}(1..n))$ </pre>
---	--

This rule refines a subset constraint on bounded multisets by refining both the left and right hand side multisets to the *variable explicit* representation, and using a common vector in both representations.

SUBSETBOUNDEDMSET2 $\rho(S_1:\text{mset}(\text{maxsize } n) \text{ of } \tau \subseteq S_2:\text{mset}(\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$

<pre> { ϕ such that $\chi_1 \wedge \chi_2$ represent S_1 by varexpmsset($\langle S', \text{Switch}_1 \rangle$) represent S_2 by varexpmsset($\langle S', \text{Switch}_2 \rangle$) symmetry indexsym($\langle S'_1, \text{Switch}_1 \rangle.1$) symmetry indexsym($\langle S'_1, \text{Switch}_1 \rangle.1$) $\phi \in \rho(\forall i:\tau. \text{Switch}_1[i] \rightarrow \text{Switch}_2[i])$ $\chi_1 \in \rho(\sum \text{Switch}_1 \leq n)$ $\chi_2 \in \rho(\sum \text{Switch}_2 \leq m)$ } </pre>	<pre> $S' = \text{genSymbol}(\langle S_1, S_2 \rangle, \text{matrix}(\text{indexed by integer}(1..m)) \text{ of } \tau)$ $\text{Switch}_1 = \text{genSymbol}(S_1, \text{matrix}(\text{indexed by integer}(1..m)) \text{ of bool})$ $\text{Switch}_2 = \text{genSymbol}(S_2, \text{matrix}(\text{indexed by integer}(1..m)) \text{ of bool})$ </pre>
--	--

Equations for the other two types of multiset follow similarly.

5.3 (Multi)set Equality

$$\text{EQUALITYBOUNDEDSET1 } \rho(S_1:\text{set } (\text{maxsize } n) \text{ of } \tau = S_2:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \phi \\ \text{such that } \chi_1 \wedge \chi_2 \\ \text{represent } S_1 \text{ by occset}(S'_1) \\ \text{represent } S_2 \text{ by occset}(S'_2) \\ | \\ \phi \in \rho(\forall i:\tau. S'_1[i] = S'_2[i]) \\ \chi_1 \in \rho(\sum S'_1 \leq n) \\ \chi_2 \in \rho(\sum S'_2 \leq m) \end{array} \right. \left. \begin{array}{l} S'_1 = \text{genSymbol}(S_1, \text{matrix } (\text{indexed by } \tau) \text{ of bool}) \\ S'_2 = \text{genSymbol}(S_2, \text{matrix } (\text{indexed by } \tau) \text{ of bool}) \end{array} \right\}$$

This equation refines the constraint $X = Y$ between two bounded sized sets where both X and Y are transformed into the occurrence representation. Similar equations exist for other types of sets and multisets.

$$\text{EQUALITYBOUNDEDSET2 } \rho(S_1:\text{set } (\text{maxsize } n) \text{ of } \tau = S_2:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \phi_1 \wedge \phi_2 \\ | \\ \phi_1 \in \rho(S_1 \subseteq S_2) \\ \phi_2 \in \rho(S_2 \subseteq S_1) \end{array} \right. \left. \begin{array}{l} \\ \\ \\ \end{array} \right\}$$

While there is a more efficient method of implementing equality between two *occurrence* representations, given in EQUALITYBOUNDEDSET1, for all other representations the most efficient method of implementing $X = Y$ is the same as implementing $X \subseteq Y \wedge Y \subseteq X$.

5.4 (Multi)set Inequality

This rule refines the inequality constraint between two bounded sized sets by refining them both to the *occurrence* representation.

$$\text{SUBSETBOUNDEDSET1 } \rho(S_1:\text{set } (\text{maxsize } n) \text{ of } \tau \subseteq S_2:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \phi \\ \text{such that } \chi_1 \wedge \chi_2 \\ \text{represent } S_1 \text{ by occset}(S'_1) \\ \text{represent } S_2 \text{ by occset}(S'_2) \\ | \\ \phi \in \rho(\exists i:\tau. S'_1[i] \neq S'_2[i]) \\ \chi_1 \in \rho(\sum S'_1 \leq n) \\ \chi_2 \in \rho(\sum S'_2 \leq m) \end{array} \right. \left. \begin{array}{l} S'_1 = \text{genSymbol}(S_1, \text{matrix } (\text{indexed by } \tau) \text{ of bool}) \\ S'_2 = \text{genSymbol}(S_2, \text{matrix } (\text{indexed by } \tau) \text{ of bool}) \end{array} \right\}$$

This rule refines the inequality constraint between two bounded sized sets by refining two subset constraints.

$$\text{INEQUALITYBOUNDEDSET2 } \rho(S_1:\text{set } (\text{maxsize } n) \text{ of } \tau = S_2:\text{set } (\text{maxsize } m) \text{ of } \tau) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \phi_1 \vee \phi_2 \\ | \\ \phi_1 \in \rho(\neg(S_1 \subseteq S_2)) \\ \phi_2 \in \rho(\neg(S_2 \subseteq S_1)) \end{array} \right. \left. \begin{array}{l} \\ \\ \\ \end{array} \right\}$$

As in EQUALITYBOUNDEDSET, apart from with the *occurrence* representation, the most efficient method to implement $X \neq Y$ on sets and multisets is $\neg(X \subseteq Y) \vee \neg(Y \subseteq X)$. Therefore for all types of sets and multisets a rule like INEQUALITYBOUNDEDSET2 is present.

6 Ternary Constraints on (Multi)sets

6.1 (Multi)set Union

This equation refines the constraint $A \cup B = C$ for A, B and C bounded sized sets by refining all the sets into the *occurrence* representation.

<pre> UNIONBOUNDEDSET1 $\rho(S_1:\text{set}(\text{maxsize } n_1) \text{ of } \tau \cup S_2:\text{set}(\text{maxsize } n_2) \text{ of } \tau = S_3:\text{set}(\text{maxsize } n_3) \text{ of } \tau) \xrightarrow{\text{ref}}$ { ϕ such that χ_1, χ_2, χ_3 represent S_1 by occset(S'_1) represent S_2 by occset(S'_2) represent S_3 by occset(S'_3) $\phi \in \rho(\forall i:\tau. S'_1[i] \vee S'_2[i] = S'_3[i])$ $\chi_1 \in \rho(\sum S'_1 \leq n_1)$ $\chi_2 \in \rho(\sum S'_2 \leq n_2)$ $\chi_3 \in \rho(\sum S'_3 \leq n_3)$ }</pre>	<pre> τ is indexable $i = \text{genSymbol}(\text{quantified}, \tau)$ $S'_1 = \text{genSymbol}(S_1, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$ $S'_2 = \text{genSymbol}(S_2, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$ $S'_3 = \text{genSymbol}(S_3, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$</pre>
--	---

This equation contains 8 rules, one for each possible way of refining A, B and C to either the *occurrence* or *variable explicit* representation. These are defined for each of the 27 ways A, B and C can either fixed, bounded or unsized sets, leading to 216 rules.

For multisets, only refining A, B and C to the *occurrence* representation is considered, leading to 27 equations with a single rule.

6.2 (Multi)set Intersection

This rule refines the constraint $A \cap B = C$ for A, B and C bounded sized sets by refining A, B and C to the *occurrence* representation.

<pre> INTERSECTIONBOUNDEDSET1 $\rho(S_1:\text{set}(\text{maxsize } n_1) \text{ of } \tau \cap S_2:\text{set}(\text{maxsize } n_2) \text{ of } \tau = S_3:\text{set}(\text{maxsize } n_3) \text{ of } \tau) \xrightarrow{\text{ref}}$ { ϕ such that χ_1, χ_2, χ_3 represent S_1 by occset(S'_1) represent S_2 by occset(S'_2) represent S_3 by occset(S'_3) $\phi \in \rho(\forall i:\tau. S'_1[i] \wedge S'_2[i] = S'_3[i])$ $\chi_1 \in \rho(\sum S'_1 \leq n_1)$ $\chi_2 \in \rho(\sum S'_2 \leq n_2)$ $\chi_3 \in \rho(\sum S'_3 \leq n_3)$ }</pre>	<pre> τ is indexable $i = \text{genSymbol}(\text{quantified}, \tau)$ $S'_1 = \text{genSymbol}(S_1, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$ $S'_2 = \text{genSymbol}(S_2, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$ $S'_3 = \text{genSymbol}(S_3, \text{matrix}(\text{indexed by } \tau) \text{ of bool})$</pre>
---	---

As with the union constraints in 6.1, there are 27 equations each with 8 rules for intersection of sets and 27 equations each with a single rule for the intersection of multisets.

7 Quantified Constraints

These are the rules which deal with quantified expressions. In *ESSENCE'* all quantified expressions are unrolled into a conjunction, disjunction or sum before they are given to the solver. Therefore these rules must be sure to produce constraints which can be correctly unrolled by introducing multiple copies of refined variables or constraints where necessary.

7.1 Quantifying atomic types

This rule refines a sum expression of an indexable type. The main operation this rule must perform is ensuring that correct variables and annotations are generated for each value in the domain of τ .

$$\text{INDEXABLESUM } \rho(\sum_{i:\tau} \alpha:\text{integer}(lb..ub)) \xrightarrow{\text{ref}} \left\{ \begin{array}{l} \text{sum}(i \text{ in } \tau) \alpha' \\ \text{such that } \forall i:\tau. \Gamma \\ \mid \\ \alpha' \text{ with local model } \Gamma \in \rho(\alpha) \\ \end{array} \right. \left| \begin{array}{l} \tau \text{ is indexable} \\ \alpha':\text{integer}(lb..ub) \end{array} \right.$$

This rule refines a “for all” quantification for an indexable type.

$$\text{INDEXABLEFORALL } \rho(\forall_{i:\tau} \phi:\text{bool}) \xrightarrow{\text{ref}} \left\{ \begin{array}{l} \text{forall}(i \text{ in } \tau) \phi' \\ \text{such that } \forall i:\tau. \Gamma \\ \mid \\ \phi' \text{ with local model } \Gamma \in \rho(\phi) \\ \end{array} \right. \left| \begin{array}{l} \tau \text{ is indexable} \\ \phi':\text{bool} \end{array} \right.$$

This rule refines a “there exists” quantification for an indexable type.

$$\text{INDEXABLEEXISTS1 } \rho(\exists_{i:\tau} \phi:\text{bool}) \xrightarrow{\text{ref}} \left\{ \begin{array}{l} \text{exists}(i \text{ in } \tau) \phi' \\ \text{such that } \forall i:\tau. \Gamma \\ \mid \\ \phi' \text{ with local model } \Gamma \in \rho(\phi) \\ \end{array} \right. \left| \begin{array}{l} \tau \text{ is indexable} \\ \phi':\text{bool} \end{array} \right.$$

This rule is different to the other rules involving quantification, and actually removes a quantified expression, transforming $\exists i \in \tau. \phi(i)$ into $\phi(x)$ for a new *CSP* variable x . Usually *GenSym* takes an existing variable to see what if any indices must be attached to a new variable. The indices on this new *CSP* variable are not created from

a previous variable, but instead it gains an index for the quantifying variable of each quantification this expression is contained in. This is already handled by CONJURE, but only used within this equation and other which perform a similar operation.

$$\text{INDEXABLEEXISTS2 } \rho(\exists_{i:\tau} \phi:\text{bool}) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \phi' \\ | \\ \phi' \in \rho(\phi[i \mapsto x]) \end{array} \right| \begin{array}{l} \tau \text{ is indexable} \\ x = \text{genSymbol}(\text{OpenQuantified}, \tau) \\ \phi':\text{bool} \end{array}$$

7.2 Quantified Constraints over non-Atomic Types

These equations deal with quantifying over types which cannot be quantified over in ESSENCE'. Note that these rules do not perform the bookwork required to correctly refine a quantified expression into a set of constraints like those in 7.1, but instead refine the quantified expression into one which quantifies over an INDEXABLE type, which is then refined by the equations in 7.1. Also note that depending upon how the quantified variable is refined, the resulting constraint and quantification can be very different.

This rule refines a quantificated sum expression over a fixed sized set by refining the set to the variable explicit representation.

$$\text{BOUNDEDSUM1 } \rho\left(\sum_{i:\tau \in S:\text{set}(\text{maxsize } n) \text{ of } \tau} \alpha:\text{integer}(D)\right) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \beta \\ \text{represent } S \text{ by varexpset}(\langle S', \text{Switch} \rangle) \\ \text{symmetry } \text{indexsym}(\langle S', \text{Switch} \rangle, 1) \\ | \\ \beta \in \rho\left(\sum_{j:\text{integer}(1..n)} (\text{Switch}[i] \times \alpha[i \mapsto S'[j]])\right) \\ \phi \in \rho(\forall i \in \text{integer}(1..n). \forall j \in \text{integer}(1..n). \\ (i \neq j \wedge \text{Switch}[i] \wedge \text{Switch}[j]) \rightarrow S'[i] \neq S'[j]) \\ } \end{array} \right| \begin{array}{l} S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of } \tau) \\ \text{Switch} = \text{genSymbol}(S, \text{matrix}(\text{indexed by integer}(1..n)) \text{ of bool}) \\ \beta:\text{integer}(D \times n) \\ \phi:\text{bool} \end{array}$$

This rule refines a quantificated sum expression over a fixed sized set by refining the set to the occurrence representation.

$$\text{BOUNDEDSUM2 } \rho\left(\sum_{i:\tau \in S:\text{set}(\text{maxsize } n) \text{ of } \tau} \alpha:\text{integer}(D)\right) \xrightarrow{\text{ref}}$$

$$\left\{ \begin{array}{l} \beta \\ \text{represent } S \text{ by occset}(S') \\ | \\ \beta \in \rho\left(\sum_{i:\tau} (S'[i] \times \alpha)\right) \\ \phi \in \rho\left(\sum_{i:\tau} S' \leq n\right) \\ } \end{array} \right| \begin{array}{l} \tau \text{ is indexable} \\ S' = \text{genSymbol}(S, \text{matrix}(\text{indexed by } \tau) \text{ of bool}) \\ \beta:\text{integer}(|\tau| \times D) \end{array}$$

The rules for the other 2 types of sets and all 3 types of multiset follow similarly. Furthermore, the rules for quantifying over \forall and \exists also follow similarly, leading to a total of 36 rules for quantified expression.

One exception to this pattern is following rule, which refines quantification of \exists by introducing an extra variable. Rules of this rule follow similarly for the other 2 types of sets and 3 types of multisets.

$$\text{BOUNDEDSETEXISTS2 } \rho(\exists i:\tau \in S:\text{set (maxsize } n) \text{ of } \tau \phi:\text{bool}) \xrightarrow{\text{ref}}$$

$\{ \phi'$ $\text{such that } \alpha$ $\text{represent } S \text{ by } \text{occset}(S')$ $ $ $\alpha \in \rho(x \in S)$ $\phi' \in \rho(\phi[i \mapsto x])$ $\}$	$S' = \text{genSymbol}(S, \text{matrix (indexed by } \tau) \text{ of bool})$ $x = \text{genSymbol}(\text{OpenQuantified}, \tau)$ $\phi':\text{bool}$ $\alpha':\text{bool}$
---	---

References

1. A. Frisch, B. Martinez-Hernandez, C. Jefferson, and I. Miguel. Generating effective constraint programs: An application of automated reasoning. In *Proceedings of the 3rd International Workshop on Modelling and Reformulating Constraint Satisfaction Problems*, 2004.