Complete TRS Specification for Abstract Collection Types

Anthony Hunt

May 18, 2025

Contents

1 Introduction

This document serves as a living specification of the underlying term rewriting system used in the compiler for a modelling-focused programming language.

2 High Level Strategy

General Strategy A basic strategy to optimize set and relational expressions is:

1. Normalize the expression as a set comprehensions
2. Simplify and reorganize conjuncts of the set comprehension body

Intuition The TRS for this language primarily involves lowering collection data type expressions into pointwise boolean quantifications. Breaking down each operation into set builder notation enables a few key actions:

- Quantifications over sets ($\{ x \cdot G \mid P \}$) are naturally separated into generators (G) and (non-generating) predicates (P). For sets, at least one membership operator per top-level conjunction in G will serve as a concrete element generator in generated code. Then, top level disjunctions will select one membership operation to act as a generator, relegating all others to the predicate level. For example, if the rewrite system observes an intersection of the form $\{ x \cdot x \in S \wedge x \in T \}$, the set construction operation must iterate over at least one of S and T. Then, the other will act as a condition to check every iteration (becoming $\{ x \cdot x \in S \mid x \in T \}$).
- By definition of generators in quantification notation, operations in G must be statements of the form $x \in S$, where x is used in the "element" portion of the set construction. Statements like $x \notin T$ or checking a property $p(x)$ must act like conditions since they do not produce any iterable elements.
- Any boolean expression for conditions may be rewritten as a combination of $\neg$, $\vee$, and $\wedge$ expressions. Therefore, by converting all set notation down into boolean notation and then generating code based on set constructor booleans, we can accommodate any form of predicate function.

Granular Strategy (Sets)

3 Supported Operations

Table 1: Summary table: a few operators on sets and relations.

Sets		Relations	
Syntax	Label/Description	Syntax	Label/Description
$set(T)$	Unordered, unique collection	$S \nrightarrow T$	Partial function
$S \leftrightarrow T$	Relation, $set(S \times T)$	$S \rightarrowtail T$	Total injection
$\varnothing$	Empty set	$a \mapsto b$	Pair (relational element)
$\{a, b, ...\}$	Set enumeration	$dom(S)$	Domain
$\{\, x \cdot x \in S \mid P \,\}$	Set comprehension	$ran(S)$	Range
$S \cup T$	Union	$R[S]$	Relational image
$S \cap T$	Intersection	$R \Leftarrow Q$	Relational overriding
$S \setminus T$	Difference	$R \circ Q$	Relational composition
$S \times T$	Cartesian Product	$S \lhd R$	Domain restriction
$S \subseteq T$	Subset	R^{-1}	Relational inverse

Table 2: Collection of operators on set data types.

Name	Definition
Empty Set	Creates a set with no elements.
Set Enumeration	Literal collection of elements to create a set.
Set Membership	The term $x \in S$ is True if x can be found somewhere in S.
Union	$S \cup T = \{\, x \cdot x \in S \vee x \in T \,\}$
Intersection	$S \cap T = \{\, x \cdot x \in S \wedge x \in T \,\}$
Difference	$S \setminus T = \{\, x \cdot x \in S \mid x \notin T \,\}$
Cartesian Product	$S \times T = \{\, x \mapsto y \cdot x \in S \wedge y \in T \,\}$
Powerset	$\mathbb{P}(S) = \{\, s \cdot s \subseteq S \,\}$
Magnitude	$\#S = \sum_{x \in S} 1$
Subset	$S \subseteq T \equiv \forall x \in S : s \in T$
Strict Subset	$S \subset T \equiv S \subseteq T \wedge S \neq T$
Superset	$S \supseteq T \equiv \forall x \in T : s \in S$
Strict Superset	$S \supset T \equiv S \supseteq T \wedge S \neq T$
Set Mapping	$f * S = \{\, f(x) \cdot x \in S \,\}$
Set Filter	$p \lhd S = \{\, x \cdot x \in S \mid p(x) \,\}$
Set Quantification (Folding)	$\oplus x \cdot x \in S \mid P$
Cardinality	$card(S) = \sum 1 \cdot x \in S$

Table 3: Collection of operators on sequence data types.

Name	Definition
Empty List	Creates a list with no elements.
List Enumeration	Literal collection of elements to create a list.
Construction	Alternative form of List Enumeration.
List Membership	The term x `in` S is True if x can be found somewhere in S.
Append	$[s_1, s_2, ..., s_n] + t = [s_1, s_2, ..., s_n, t]$
Concatenate	$[s_1, ..., s_n] +\!\!+ [t_1, ..., t_n] = [s_1, ..., s_n, t_1, ...t_n]$
Length	$\#S = \sum 1 \cdot x \texttt{ in } S$
List Mapping	$f * S = [\, f(x) \cdot x \texttt{ in } S\,]$
List Filter	$p \lhd S = [\, f(x) \cdot x \texttt{ in } S \mid p(x)\,]$
Associative Reduction	$\oplus/[s_1, s_2, ..., s_n] = s_1 \oplus s_2 \oplus ... \oplus s_n$
Right Fold	$\texttt{foldr}(f, e, [s_1, s_2, ..., s_n]) = f(s_1, f(s_2, f(..., f(s_n, e))))$
Left Fold	$\texttt{foldl}(f, e, [s_1, s_2, ..., s_n]) = f(f(f(f(e, s_1), s_2), ...), s_n)$

Table 4: Collection of operators on relation data types.

Name	Definition
Empty Relation	Creates a relation with no elements.
Relation Enumeration	Literal collection of elements to create a relation.
Identity	$id(S) = \{\, x \mapsto x \cdot x \in S \,\}$
Domain	$dom(R) = \{\, x \cdot x \mapsto y \in R \,\}$
Range	$ran(R) = \{\, y \cdot x \mapsto y \in R \,\}$
Relational Image	$R[S] = \{\, y \cdot x \mapsto y \in R \mid x \in S \,\}$
Overriding	$R \Lleftarrow Q = Q \cup (dom(Q) \lhd\!\!\!- R)$
(Forward) Composition	$Q \circ R = \{\, x \mapsto z \cdot x \mapsto y \in R \wedge y \mapsto z \in Q \,\}$
Inverse	$R^{-1} = \{\, y \mapsto x \cdot x \mapsto y \in R \,\}$
Domain Restriction	$S \lhd R = \{\, x \mapsto y \cdot x \mapsto y \in R \mid x \in S \,\}$
Domain Subtraction	$S \lhd\!\!\!- R = \{\, x \mapsto y \cdot x \mapsto y \in R \mid x \notin S \,\}$
Range Restriction	$R \rhd S = \{\, x \mapsto y \cdot x \mapsto y \in R \mid y \in S \,\}$
Range Subtraction	$R \rhd\!\!\!- S = \{\, x \mapsto y \cdot x \mapsto y \in R \mid y \notin S \,\}$

4 Rules

Below is a list of rewrite rules for key abstract data types.

4.1 Sets

$$
\begin{aligned}
S &\rightarrow \{\, x \cdot x \in S \,\} && \text{(Set Construction)} \\
x \in e &\rightarrow x = e && \text{(Singleton Membership)} \\
x \in \{\, y \cdot E \mid P \,\} &\rightarrow x \in E \mid P && \text{(Membership Collapse)}
\end{aligned} \tag{1}
$$

$$\{\, x \mid x \in S \,\} \cup \{\, y \mid y \in T \,\} \to \{\, x \mid x \in S \lor x \in T \,\} \tag{Union}$$

$$\{\, x \mid x \in S \,\} \cap \{\, y \mid y \in T \,\} \to \{\, x \mid x \in S \land x \in T \,\} \tag{Intersection}$$

$$\{\, x \mid x \in S \,\} \setminus \{\, y \mid y \in T \,\} \to \{\, x \mid x \in S \land x \notin T \,\} \tag{Difference}$$

4.2 Relations

$$R \to \{\, x \mapsto y \cdot x \mapsto y \in R \,\} \qquad \text{(Relation Construction)} \tag{2}$$