

Heyting Domains for Constraint Abduction

Michael Maher

National ICT Australia and University of NSW
Sydney, Australia
Michael.Maher@nicta.com.au

Abstract. We investigate constraint domains in which answers to constraint abduction problems can be represented compactly by a most general answer. We demonstrate several classes of domains which have this property, but show that the property is not compositional.

1 Introduction

Abduction is the inference rule that derives A from B and C , such that $A, B \vdash C$. It was considered by Peirce [12] to be – along with deduction and induction – one of the fundamental forms of reasoning. Mostly, abduction has been addressed in a setting of partially determined predicates or propositions. *Constraint abduction* refers to abduction in a setting where A , B and C come from a set of pre-defined, completely-defined relations (constraints, in the sense of constraint logic programming [6]) and A , the answer, ensures $A \wedge B \rightarrow C$.

Traditionally, abduction has been used as a model of scientific hypothesis formation and diagnosis, and as a component of belief revision and machine learning, but the applications of constraint abduction are in query evaluation using views [18], which is used for integrating multiple sources of information [9], and in logic program analysis [3, 4] and type inference [16].

In previous work it has been established that some natural constraint domains, such as linear arithmetic and finite terms, do not in general admit compact explicit representations of all answers to an abduction problem [11, 10]. In this paper we investigate constraint domains in which all answers can be represented by a single most general answer. We call these *Heyting domains*. It turns out that many of the simpler, commonly-used constraint domains have this property although even in these simple cases there are provisos. Furthermore, we show that even a simple combination of Heyting domains loses the Heyting property. This presents a substantial difficulty for the use of constraint abduction, but we demonstrate one case in which the composition of Heyting domains is Heyting. We leave for future work a detailed study of this issue.

2 Background

The syntax and semantics of constraints are defined by a constraint domain [6]. Given a signature Σ , and a set of variables $Vars$ (which we assume is infinite), a *constraint domain* is a pair $(\mathcal{D}, \mathcal{L})$ where \mathcal{D} is a Σ -structure and \mathcal{L} (the language of constraints)

is a set of Σ -formulas including constraints equivalent to *true* and *false*, and closed under conjunction and renaming of free variables. If, for some set of constraints $S \subseteq \mathcal{L}$, every constraint is equivalent to a conjunction of renamed S -constraints then we refer to S as a set of *primitive constraints*. We say the constraint domain is *unary* if \mathcal{L} is generated by a set of unary primitive constraints. We say the constraint domain is *finite* if the set of values in \mathcal{D} is finite.

When the constraint domain \mathcal{D} is clear from the context we write $C \rightarrow C'$ as an abbreviation for $\mathcal{D} \models C \rightarrow C'$. We say C is *more general than* C' if $C' \rightarrow C$. Two constraints C and C' are *equivalent* if $C \rightarrow C'$ and $C' \rightarrow C$.

The constraints (modulo equivalence) form a partially ordered set (poset) where $C_1 \leq C_2$ iff $C_1 \rightarrow C_2$. *true* and *false* are, respectively, the top and bottom elements of the poset. The greatest lower bound $C_1 \sqcap C_2$ of two constraints C_1 and C_2 is their conjunction $C_1 \wedge C_2$. The least upper bound $C_1 \sqcup C_2$ of two constraints may not exist.

When every pair of elements in a poset has a least upper bound and a greatest lower bound, the poset is called a *lattice*. If, in addition, top and bottom elements exist it is called a *bounded lattice*. A lattice is said to be *distributive* if, for all elements x, y and z , $x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$ and $x \sqcup (y \sqcap z) = (x \sqcup y) \sqcap (x \sqcup z)$. The complement of an element x in a bounded lattice is an element \bar{x} such that $x \sqcup \bar{x}$ is the top element and $x \sqcap \bar{x}$ is the bottom element. A distributive lattice where every element has a unique complement is called a *Boolean lattice*. A Boolean lattice that provides the operations \sqcap, \sqcup and $\bar{}$ is called a *Boolean algebra*. The *relative pseudo-complement* of an element x with respect to an element y is the unique (if it exists) greatest element z such that $z \sqcap x \leq y$. A bounded lattice where every element has a relative pseudo-complement with respect to each element in the lattice is called a *Heyting lattice*. Every Boolean lattice is a Heyting lattice. If the dual of a Heyting lattice \mathcal{H} (the “upside-down” lattice, where \leq and \geq are interchanged) also forms a Heyting lattice then \mathcal{H} is a *bi-Heyting lattice*. Heyting lattices and algebras originated as models for intuitionistic logic and bi-Heyting algebras have been proposed as models for spatial reasoning calculi [15], but these uses seem largely unconnected to their use in this paper.

3 Simple Constraint Abduction

We can now formally define the simple constraint abduction problem. Extensions of this problem are discussed in Section 5. We write \exists for the existential quantification of all variables.

Definition 1 *The Simple Constraint Abduction Problem is as follows:*

Given a constraint domain $(\mathcal{D}, \mathcal{L})$, and given two constraints $B, C \in \mathcal{L}$ such that $\mathcal{D} \models \exists B \wedge C$, for what constraints $A \in \mathcal{L}$ does

$$\mathcal{D} \models (A \wedge B) \rightarrow C$$

and

$$\mathcal{D} \models \exists (A \wedge B)$$

An instance of the problem has a fixed constraint domain and fixed constraints B and C . We call A an answer to the problem instance.

Throughout this paper, A , B and C refer to the constraints in a simple constraint abduction problem. In an abuse of terminology, we often will refer to an instance as a SCA problem.

Traditional abduction might be thought of as a kind of higher-order constraint abduction where the variables range over relations and the constraints state tuples that are in/out of those relations. Abductive logic programming [7] is somewhat different because the deductive relation \vdash is different from material implication. ACLP [8] extends abductive logic programming with constraints. It uses a version of SCA (where C is *false*) in its algorithm but it does not perform abduction on constraints, only on formulas that may involve constraints.

It is easy to see that there is always an answer to a SCA problem by taking A to be C . However, in general there may be infinitely many answers, and we will need a finite representation. For example, if B is $x \geq 5$ and C is $x \geq 7$ then C is an answer, but so is every constraint $x \geq k$ where $k \geq 7$. In this case, C can be considered to be a representation of all answers since, for all answers A' , A' is the conjunction of some constraint with C (that is, $A' \rightarrow C$), and every A' that is obtained in this way and is consistent with B is an answer. We say an answer A to a SCA problem is the *most general answer* if, for every answer A' of the problem, $A' \rightarrow A$.

Unfortunately, not all SCA problems have a most general answer, which leads us to consider *maximally general answers*. A maximally general answer is an answer A such that there is no answer strictly more general than A . These answers are singled out by the parsimony principle of abductive inference, which suggests choosing explanations with weakest explanatory power. They also have the potential to compactly represent many answers, since they can be taken to represent all stronger answers.

However, in some constraint domains such as linear arithmetic constraints, the maximally general answers do not represent all answers [11]. Furthermore, even in constraint domains where maximally general answers represent all answers, there may be infinitely many maximally general answers [10, 11]. It is clear from [10, 11] that only rarely do SCA problems have most general answers in the constraint domains that those papers address (linear arithmetic constraints/finite term equalities).

In this paper, we will focus on identifying constraint domains for which every SCA problem has a most general answer. This property ensures that the difficulties mentioned above are avoided, though at the cost of limiting the constraint domains over which we can abduce. In the type-inference setting, it corresponds to the principal-types property. We say \mathcal{D} is a *Heyting domain* when this property holds. This terminology is justified by the following result.

Proposition 1 *Let \mathcal{D} be a constraint domain. \mathcal{D} is a Heyting domain iff the poset of constraints has the property that every element has a relative pseudo-complement with respect to each element in the poset.*

This proposition essentially reformulates the property of being a Heyting domain in order-theoretic terms. The relative pseudo-complement is precisely the most general answer. Note that a Heyting domain might not form a Heyting lattice because it might not be a lattice. However, every finite unary constraint domain forms a lattice, and thus every finite unary Heyting domain is a Heyting lattice.

4 Heyting Constraint Domains

We have seen that constraint domains involving arithmetic over the reals or integers, or equality over finite terms, are not Heyting domains. Of the remaining widely-used constraint domains, most are unary. In particular, bound and interval constraints are used for solving integer problems [5] and non-linear continuous problems [1], as well as problems over finite sets [2, 13] or multisets [17], while finite domain constraints are the basis of CSPs. Hence we focus on establishing if/when these and similar domains are Heyting, and presenting a constructive characterization of their most general answers, when it exists.

4.1 Boolean Constraint Domains

Proposition 1 characterizes the Heyting domains, but it does not provide a way to compute a most general answer. However, in Boolean algebras we can compute the most general answer directly, using the algebraic operations.

Proposition 2 *Let \mathcal{D} be a constraint domain where the constraints form a Boolean lattice. Consider the SCA problem over \mathcal{D} and define A to be $\overline{B} \sqcup C$.*

Then A is the most general answer.

The above result applies to finite domains, as used in CSPs, where each variable ranges over a finite set of values D and the only primitive constraints restrict variables to a subset of D . The complement of a constraint $x \in S$ is $x \in (D \setminus S)$. The least upper bound is obtained by union: $(x \in S_1) \sqcup (x \in S_2)$ is $x \in (S_1 \cup S_2)$. The greatest lower bound is expressed by conjunction and can be obtained by intersection in a similar way. Similarly, the constraint domain where constraints are unions of floating point-bounded intervals over the real numbers, as used in Echidna [14], is a Heyting domain by this result.

Note that the least upper bound operation \sqcup is, in general, different from disjunction \vee . However, when the constraint language is closed under negation the constraint domain forms a Boolean lattice and, in that case, the least upper bound is exactly disjunction.

So far we have been using the lattice structure of the *constraints*. In the remainder of this section we consider constraint domains where the partial order of *values* is important.

4.2 Bounds on a Total Order

Interval/bounds constraints on a total order cannot be addressed by Proposition 2 because these constraints do not form a distributive lattice and a complement is not defined.

Example 1. Consider bounds constraints on integers. Let C_1 be $x \in 1..5$, C_2 be $x \in 7..9$ and C_3 be $x \in 5..6$. Then $C_1 \sqcup C_2$ is $x \in 1..9$. and $(C_1 \sqcup C_2) \wedge C_3$ is $x \in 5..6$. However, $C_1 \wedge C_3$ is $x \in 5..5$ and $C_2 \wedge C_3$ is *false* so $(C_1 \wedge C_3) \sqcup (C_2 \wedge C_3)$ is $x \in 5..5$. Thus the lattice of constraints is not distributive.

The complement of C_1 must be an interval disjoint from $1..5$ and hence contains only numbers less than 1 or greater than 5. But then the least upper bound of these two intervals cannot be the full interval of integers. Thus complement is not defined.

Consider any constraint domain \mathcal{D} with a total ordering \leq . If we consider bounds, primitive constraints have the form $c \leq x$ or $x \leq c$, where c is a constant and x is a variable. If we consider intervals, primitive constraints have the form $x \in l..u$, meaning $l \leq x \leq u$, where l and u are constants and x is a variable. In both cases we might also consider strict inequalities; to simplify the exposition we will omit this possibility, which complicates the description but does not raise any new issues.

For both bounds and intervals, a solved form can be computed by essentially intersecting all intervals (or bounds) concerning each variable. Thus, in a solved form there is at most one interval or two bounds (one lower, one upper) on each variable. If there is no lower (upper) bound on a variable in a constraint we can act as though there is an infinitely small (large) bound.

Given B and C , which we can assume without loss of generality are in solved form, we construct A to satisfy $A \wedge B \rightarrow C$ as follows:

For each variable x in B or C , let l_B^x and l_C^x (u_B^x and u_C^x) be the lower (respectively, upper) bounds for x in B and C . Define A as follows. For each variable x :

$d \leq x$ is in A iff $d = l_C^x$, $d \neq -\infty$ and $l_B^x < l_C^x$

$x \leq d$ is in A iff $d = u_C^x$, $d \neq \infty$ and $u_C^x < u_B^x$

That is, A contains those bounds of C that are strictly tighter than those of B .

The following result follows directly from the construction of A .

Theorem 1. *Consider the SCA problem over a constraint domain of bounds on a total order. Let A be as defined above. Then A is the unique most general answer.*

For example, let B be $2 \leq x \wedge x \leq 8$ and C be $1 \leq x \wedge x \leq 5$. Then $l_B^x \not< l_C^x$ ($2 \not< 1$) and so A contains no lower bound for x . But $u_C^x < u_B^x$ ($5 < 8$) so A contains $x \leq 5$.

Since intervals are essentially the conjunction of two bounds, we can achieve a similar result in that case, but only if there is minimum possible lower bound and maximum possible upper bound. If there is no minimum possible lower bound and maximum possible upper bound (for example, if bounds of intervals can be any integer, but not an infinite bound) then SCA problems can have an infinite ascending chain of answers, and thus not have a maximally general answer.

Example 2. Consider a constraint domain of finite intervals over the integers. If B is $x \in 0..5$ and C is $x \in 1..10$ then any constraint A of the form $x \in 1..n$ is an answer, but the set of all such constraints forms an ascending chain with no least upper bound.

4.3 Bounds on a Quasi Order

If the values of a constraint domain \mathcal{D} are quasi-ordered by a relation \sqsubseteq then we can consider the induced equivalence relation \sim and the induced partial order over the quotient \mathcal{D}/\sim . Any \sqsubseteq -bound on \mathcal{D} induces a bound on \mathcal{D}/\sim , and vice versa. Thus

Proposition 3 *Let \mathcal{D} be the constraint domain of a quasi-order over a set D and let \mathcal{D}' be the corresponding quotient constraint domain of the induced partial order. Then \mathcal{D} is a Heyting domain iff \mathcal{D}' is a Heyting domain.*

Constraint domains with a quasi-ordering can arise when truncation or rounding of numbers is performed. For example, the set of real numbers is quasi-ordered by the ordering on integers via truncation: $x \sqsubseteq y$ iff $\lfloor x \rfloor \leq \lfloor y \rfloor$. Thus the above result implies that limitations of machine representability of numbers will not affect the Heyting domain property for the constraint domains discussed in the previous subsection. Later we will see an example involving finite sets.

4.4 Bounds on a Partial Order

We can obtain similar results for some partially ordered constraint domains with bounds constraints. Unlike a total order, a partially ordered constraint domain with bounds constraints forms a Heyting domain only when the partial order has a certain structure: \mathcal{D} must be a bi-Heyting lattice. However, the construction of this answer cannot be given in such simple terms, in general.

Theorem 2. *Consider the SCA problem over a constraint domain \mathcal{D} of bounds on a partial order that forms a bi-Heyting lattice. Then \mathcal{D} is a Heyting domain.*

In this result the relative pseudo-complement is needed for the upper bounds, while its dual is needed for the lower bounds.

We can construct most general answers if the partial order of values forms a Boolean lattice – for example, bounds on set variables [2] where we assume the values that a set variable can take are bounded above by a finite set. Suppose B and C are in solved form. We define A by, for each variable x ,

$$\begin{aligned} d \leq x \text{ is in } A &\text{ iff } d = (l_C^x \sqcap \overline{l_B^x}) \text{ and } l_B^x \not\geq l_C^x \\ x \leq d \text{ is in } A &\text{ iff } d = (u_C^x \sqcup \overline{u_B^x}) \text{ and } u_C^x \not\geq u_B^x \end{aligned}$$

where \sqcup and \sqcap are the lattice operations and \overline{p} denotes the complement of p . Then A is the most general answer.

Theorem 3. *Consider the SCA problem over a constraint domain \mathcal{D} of bounds on a partial order that forms a Boolean lattice. Suppose that the set of values of variable-free terms in \mathcal{D} (that can be used as bounds in constraints) is closed under the operations of the lattice (\sqcup , \sqcap , and complement). Let A be as defined above.*

Then A is the most general answer.

Theorem 3 applies directly to the constraint domain of bounds on finite sets.

Example 3. Consider interval constraints on finite sets as in Conjunto [2]. Here the lattice operations on the values of the domain are the familiar union, intersection and set complement operations. Suppose that x has a lower bound in B of $l_B^x = \{1, 2, 3\}$ and a lower bound in C of $l_C^x = \{1, 2, 4\}$. Then $l_B^x \not\geq l_C^x$ and $l_A^x = \{1, 2, 4\} \sqcap \overline{\{1, 2, 3\}} = \{4\}$.

If the values of this constraint domain are the finite subsets of an infinite set and constraints are intervals then, like Example 2, there may not be a most general answer. But if the values are the subsets of a finite set then there is a most general answer, even when the constraints are intervals. However, this only applies to finite sets where subset bounds are the only constraints; if there are more constraints then there might not be a most general answer.

4.5 Finite Set Constraints

In [13], the conventional containment partial ordering on finite sets is used in combination with a quasi-ordering based on the cardinalities of sets and a lexicographic ordering on sets. The quasi-ordering \sqsubseteq is defined by $X \sqsubseteq Y$ iff $|X| \leq |Y|$, where $|X|$ denotes the cardinality of X . The lexicographic ordering is defined recursively as follows:

$$X \preceq Y \text{ iff } X = \emptyset \text{ or } x < y \text{ or } x = y \wedge X \setminus \{x\} \preceq Y \setminus \{y\}$$

where $x = \max(X)$, the largest element in X , and $y = \max(Y)$ and $<$ is a total order over all elements that may appear in a set. For example, suppose elements of the sets are natural numbers and $<$ is the usual ordering on natural numbers. Then¹ $\{3, 2, 1\} \preceq \{4, 2\}$ since $\max(\{3, 2, 1\}) = 3 < 4 = \max(\{4, 2\})$. The characteristic vector representation of a set S is a list χ_S of 0/1 values such that the i 'th value is in S iff the i 'th element of χ_S is 1, where the values are ordered from $<$ -largest to smallest. The *longest common prefix* of two sets U and V is the longest common prefix of χ_U and χ_V .

By Theorem 1, the constraint domain of bounds over sets under the lexicographic ordering is a Heyting domain. Similarly, by Proposition 3 and Theorem 1, the constraint domain of bounds on cardinalities of sets is a Heyting domain. As we saw above, from Theorem 3, the constraint domain of bounds over sets under the containment ordering is a Heyting domain. However, it does not immediately follow that a constraint domain combining these kinds of constraints has unique most general answers.

Indeed, consider the SCA problem where B is $|x| = 1$ and C is $x = \{1\}$. Then both $x \sqsubseteq \{1\}$ and $\{1\} \subseteq x$ are maximally general answers. Thus the constraint domain which uses all three kinds of constraints (as in [13]) is not a Heyting domain (and neither are the constraint domains using only cardinality constraints and either of the other two orderings). That is, the property of being a Heyting domain is not compositional, even in this relatively simple case.

Definition 2 *The composition by language of constraint domains $(\mathcal{D}, \mathcal{L}_1)$ and $(\mathcal{D}, \mathcal{L}_2)$ having the same structure \mathcal{D} is $(\mathcal{D}, \mathcal{L})$ where \mathcal{L} is the conjunctive closure of $\mathcal{L}_1 \cup \mathcal{L}_2$.*

Proposition 4 *The Heyting property is not closed under composition by language.*

Let us now consider the constraint domain involving only containment and lexicographic bounds constraints. A constraint in this constraint language is in *simplified form* if the constraints on any variable x have the form $d_l \subseteq x \subseteq d_u, f_l \preceq x \preceq f_u$ where

¹ Elements are written in descending order in this example to emphasize the lexicographic nature of \preceq .

- $d_l \subseteq f_u \subseteq d_u$ and $d_l \subseteq f_l \subseteq d_u$,
- $\max(d_u) = \max(f_u)$ and $\max(\overline{d_l}) = \max(\overline{f_l})$, where \bar{x} denotes the complement of x ,
- the longest common prefix of f_l and f_u is a prefix of d_l and d_u .

This simplified form is similar to the normal form of rewrite rules given in [13], though formulated differently, and every constraint can be simplified by similar rules.

Let B be in simplified form and let BC be the simplified form of $B \wedge C$. We define A' using the methods defined earlier for Theorem 1 (in the case of \preceq constraints) and for Theorem 3 (in the case of \subseteq constraints). Thus A' contains $\emptyset \preceq x$ if $l_{BC} \preceq l_B$ and $l_{BC} \preceq x$ otherwise, where l_B and l_{BC} are the corresponding \preceq -lower bounds for x in B and BC respectively. A' contains $\emptyset \subseteq x$ if $l_{BC} \subseteq l_B$ and $l_{BC} \cap \overline{l_B} \subseteq x$ otherwise, where here l_B and l_{BC} are the corresponding \subseteq -lower bounds for x in B and BC respectively. Upper bounds are similar.

Given A' , we expand the bounds, if possible as follows. Let A'_{-l} and A'_{-u} be A' except that the \preceq -lower bound (respectively \preceq -upper bound) of x is omitted. If $(A'_{-l} \wedge B) \rightarrow \neg(l_B \preceq x \prec l_{A'})$ holds then the \preceq -lower bound of x in A' is replaced by the \subseteq -lower bound. Similarly, if $A'_{-u} \wedge B \rightarrow \neg(u_{A'} \prec x \preceq u_B)$ holds then the \preceq -upper bound of x in A' is replaced by the \subseteq -upper bound. A similar expansion on the \subseteq -bounds is unnecessary. The resulting constraint is A .

Theorem 4. *Consider an SCA problem over the constraint domain of finite sets with both containment and lexicographic orderings. Let A be as defined above.*

Then A is the most general answer.

Example 4. Consider the SCA over the combined domain on subsets of $\{4, 3, 2, 1\}$. Let B be $\{3, 1\} \preceq x$, or $\emptyset \subseteq x \subseteq \{4, 3, 2, 1\}$, $\{3, 1\} \preceq x \preceq \{4, 3, 2, 1\}$ in simplified form, and let C be $\{2\} \subseteq x \subseteq \{4, 3, 2\}$, $\{3, 2\} \preceq x \preceq \{4, 3, 2\}$. Then BC is identical to C and A' is $\{2\} \subseteq x \subseteq \{4, 3, 2\}$, $\{3, 2\} \preceq x \preceq \{4, 3, 2\}$. Expanding the lower bound since $A'_{-l} \wedge B \rightarrow x \neq \{3, 1\}$, we find A is $\{2\} \subseteq x \subseteq \{4, 3, 2\}$, $\{2\} \preceq x \preceq \{4, 3, 2\}$.

4.6 Bounds on Finite Multisets

When the constraint domain consists of bounds over multiset variables [17] the above results do not apply. First we define this constraint domain. A multiset is a function in $S \rightarrow \mathbb{N}$, where S is a finite set. Thus a multiset $m = \{\{a, b, b\}\}$ is formalized as a function where $m(a) = 1$, $m(b) = 2$, and $m(c) = 0$. Multisets are ordered by the containment ordering: $m_1 \leq m_2$ iff $\forall s \in S \ m_1(s) \leq m_2(s)$. We say m_1 is contained in m_2 or m_1 is a submultiset of m_2 . Under the containment ordering there is a least multiset (the empty multiset, which is the zero function), and we assume there is a greatest multiset M . The least upper bound and greatest lower bound under this ordering can be defined as follows: $(m_1 \sqcup m_2)(s) = \max\{m_1(s), m_2(s)\}$ and $(m_1 \sqcap m_2)(s) = \min\{m_1(s), m_2(s)\}$.

The primitive constraints in a multiset constraint domain have the form $x \leq m$ or $m \leq x$, specifying that the value of the variable x is contained in (respectively, contains) the multiset m . For any constraint C , we can express C as $\bigwedge_{x \in \text{vars}} (l_C^x \leq x \wedge x \leq u_C^x)$. We assume there is a greatest multiset M in the constraint domain, the multisets

under consideration are exactly the submultisets of M , and that each submultiset can be represented by a constant expression. Thus, if M is finite then multiset variables range over a finite collection of multisets.

The containment ordering on multisets does form a distributive lattice, but it does not have a complement. Thus the lattice of values is not a Boolean lattice, and thus Theorem 3 does not apply. Similarly, Proposition 2 does not apply because the lattice of constraints is not a Boolean lattice.

Nevertheless, we can construct the most general answer to any SCA problem over such multiset constraints. For a given SCA problem, define A as $\bigwedge_{x \in \text{Vars}} (l_A^x \leq x \wedge x \leq u_A^x)$ where

$$l_A^x(s) = \begin{cases} 0 & l_B^x(s) \geq l_C^x(s) \\ l_C^x(s) & \text{otherwise} \end{cases}$$

and

$$u_A^x(s) = \begin{cases} M(s) & u_B^x(s) \leq u_C^x(s) \\ u_C^x(s) & \text{otherwise} \end{cases}$$

Then A defines the most general answer for the SCA problem over multiset constraints.

Theorem 5. *Consider the SCA problem over a constraint domain of bounds on multiset variables where there is a greatest multiset M . Let A be as defined above.*

Then A is the unique most general answer.

For example, if B is $\{\{a, b, b\}\}$ (that is, $m_B(a) = 1$, $m_B(b) = 2$, $m_B(c) = 0$) and C is $\{\{a, b, c\}\}$ ($m_C(a) = 1$, $m_C(b) = 1$, $m_C(c) = 1$) then A is $\{\{c\}\}$ ($m_A(a) = 0$, $m_A(b) = 0$, $m_A(c) = 1$).

5 Generalized Constraint Abduction

Simple constraint abduction is the basis for other abduction problems [10]. In joint constraint abduction, several simple problems must be solved simultaneously by an answer A . In variable-restricted constraint abduction, A may use only a limited set V of variables². For Heyting domains, joint problems can be decomposed into simple abduction problems, and the conjunction of all most general answers provides a most general answer to the joint problem iff the joint problem has an answer. See [10].

Corresponding to every variable restricted problem is the *unrestricted* problem – the simple problem where the variable restriction is ignored. We can use the unrestricted problem to address the variable-restricted problem.

Theorem 6. *Let $(\mathcal{D}, \mathcal{L})$ be a constraint domain where constraints are generated from unary primitive constraints. Then the variable-restricted constraint abduction problem wrt V has an answer iff $\mathcal{D} \models B \rightarrow \exists_V C$.*

Furthermore, assuming the variable-restricted problem has an answer, if the unrestricted problem has a most general answer then that is also the most general answer to the restricted problem.

² This set corresponds to the notion of *abducibles* in traditional abduction.

6 Conclusions

We have investigated the class of Heyting domains: those domains where every simple constraint abduction problem that has an answer has a most general answer. We have demonstrated classes of domains that have this property, which include many of the simpler commonly-used constraint domains. We have seen that the Heyting property is not preserved under constraint language composition, but that it does extend to joint constraint abduction and, for unary domains, to variable-restricted abduction.

References

1. F. Benhamou & W.J. Older, Applying Interval Arithmetic to Real, Integer, and Boolean Constraints, *J. Logic Programming* 32(1): 1–24, 1997.
2. C. Gervet, Interval Propagation to Reason about Sets: Definition and Implementation of a Practical Language, *Constraints* 1(3): 191–244, 1997.
3. R. Giacobazzi, Abductive Analysis of Modular Logic Programs, *Journal of Logic and Computation* 8(4):457–484, 1998.
4. J. M. Howe, A. King, & L. Lu, Analysing Logic Programs by Reasoning Backwards in: M. Bruynooghe and K.-K. Lau (Eds), *Program Development in Computational Logic*, LNCS 3049, 152–188. 2004.
5. P. Van Hentenryck, *Constraint Satisfaction in Logic Programming*, MIT Press, 1989.
6. J. Jaffar & M.J. Maher, Constraint Logic Programming: A Survey, *Journal of Logic Programming* 19 & 20, 503–581, 1994.
7. A.C. Kakas, R.A. Kowalski & F. Toni, Abductive Logic Programming, *J. Logic and Computation* 2(6): 719–770, 1992.
8. A.C. Kakas, A. Michael & C. Mourlas, ACLP: Abductive Constraint Logic Programming, *J. Logic Programming* 44(1-3): 129–177, 2000.
9. A.Y. Levy, A. Rajaraman & J.J. Ordille, Query-Answering Algorithms for Information Agents, *Proc. AAAI/IAAI*, Vol. 1, 40–47, 1996.
10. M.J. Maher, Herbrand Constraint Abduction, *Proc. Logic in Computer Science Conf.*, 397–406, 2005.
11. M. Maher, Abduction of Linear Arithmetic Constraints, *Proc. International Conference on Logic Programming*, LNCS 3668, Springer, 174–188, 2005.
12. C.S. Peirce, *Collected Papers of Charles Saunders Peirce*, C. Hartshorne & P. Weiss (Eds), Belknap Press of Harvard University Press, 1965.
13. A. Sadler & C. Gervet, Hybrid Set Domains to Strengthen Constraint Propagation and Reduce Symmetries *Proc. CP*, 604–618, LNCS 3258, 2004.
14. G. Sidebottom & W.S. Havens, Hierarchical Arc Consistency for Disjoint Real Intervals in Constraint Logic Programming, *Computational Intelligence* 8: 601–623, 1992.
15. J.G. Stell & M.F. Worboys, The Algebraic Structure of Sets of Regions, *Proceedings Spatial Information Theory, COSIT'97*, 163–174, 1997.
16. M. Sulzmann, T. Schrijvers & P.J. Stuckey, Type Inference for GADTs via Herbrand Constraint Abduction, draft paper, 2006.
17. T. Walsh, Consistency and Propagation with Multiset Constraints: A Formal Viewpoint, *Proc. CP*, 724–738, LNCS 2833, 2003.
18. J. Wang, M. Maher & R. Topor, Rewriting Unions of General Conjunctive Queries Using Views, *Proc. EDBT*, LNCS 2287, 52–69, 2002.