Finite Domain Bounds Consistency Revisited

C.W. Choi[1], W. Harvey[2], J.H.M. Lee[1], and P.J. Stuckey[3]

[1] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong SAR, China
Email: {cwchoi,jlee}@cse.cuhk.edu.hk
[2] CrossCore Optimization Ltd, London, United Kingdom
Email: warwick.harvey@crosscoreop.com
[3] NICTA Victoria Laboratory, Department of Computer Science & Software Engineering, University of Melbourne, 3010, Australia
Email: pjs@cs.mu.oz.au

Abstract. A widely adopted approach to solving constraint satisfaction problems combines systematic tree search with constraint propagation for pruning the search space. Constraint propagation is performed by propagators implementing a certain notion of consistency. Bounds consistency is the method of choice for building propagators for arithmetic constraints and several global constraints in the finite integer domain. However, there has been some confusion in the definition of bounds consistency and of bounds propagators. We clarify the differences among the three commonly used notions of bounds consistency in the literature. This serves as a reference for implementations of bounds propagators by defining (for the first time) the *a priori* behavior of bounds propagators on arbitrary constraints.

1 Introduction

One widely-adopted approach to solving CSPs combines backtracking tree search with constraint propagation. This framework is realized in constraint programming systems, such as ECL^{i}PS^e [4], SICStus Prolog [23] and ILOG Solver [10], which have been successfully applied to many real-life industrial applications.

Constraint propagation, based on local consistency algorithms, removes infeasible values from the domains of variables to reduce the search space. The most successful consistency technique was *arc consistency* [15], which ensures that for each binary constraint, every value in the domain of one variable has a supporting value in the domain of the other variable that satisfies the constraint.

A natural extension of arc consistency for constraints of more than two variables is *domain consistency* [24] (also known as *generalized arc consistency* and *hyper-arc consistency*). Checking domain consistency is NP-hard even for linear equations, an important kind of constraints.

To avoid this problem weaker forms of consistency were introduced for handling constraints with large numbers of variables. The most successful one for linear arithmetic constraints has been *bounds consistency* (sometimes called *interval consistency*). Unfortunately there are *three* commonly used but *incompatible* definitions of bounds consistency in the literature. This is confusing to practitioners in the design and implementation of efficient bounds consistency algorithms, as well as for users of constraint programming systems claiming to support bounds consistency. We clarify the three existing definitions of bounds consistency and the differences between them.

Propagators are functions implementing certain notions of consistency to perform constraint propagation. For simplicity, we refer to propagators implementing bounds consistency as *bounds propagators*. We aim to formalize (for the first time) precisely the *operational semantics* of various kinds of bounds propagators for arbitrary constraints. The precise semantics would serve as the basis for all implementations of bounds propagators. We also study how bounds propagators are used in practice.

2 Propagation Based Constraint Solving

We consider integer constraint solving using constraint propagation. Let $\mathcal{Z}$ denote the integers, and $\mathcal{R}$ denote the reals.

We consider a given (finite) set of integer variables $\mathcal{V}$, which we shall sometimes interpret as real variables. Each variable is associated with a finite set of possible values, defined by the domain. A *domain* D is a complete mapping from a set of variables $\mathcal{V}$ to finite sets of integers. The *intersection* of two domains D and D', denoted $D \sqcap D'$, is defined by the domain $D''(v) = D(v) \cap D'(v)$ for all $v \in \mathcal{V}$. A domain D is *stronger* than a domain D', written $D \sqsubseteq D'$, iff $D(v) \subseteq D'(v)$ for all variables $v \in \mathcal{V}$. A domain D is equal to a domain D', denoted $D = D'$, iff $D(v) = D'(v)$ for all variables $v \in \mathcal{V}$. A domain D is stronger than (equal to) a domain D' w.r.t. variables V, denoted $D \sqsubseteq_V D'$ (resp. $D =_V D'$), iff $D(v) \subseteq D'(v)$ (resp. $D(v) = D'(v)$) for all $v \in V$.

Let *vars* be the function that returns the set of variables appearing in an expression or constraint. A *valuation* θ is a mapping of variables to values (integers or reals), written $\{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$. Define $vars(\theta) = \{x_1, \ldots, x_n\}$. In an abuse of notation, we define a valuation θ to be an element of a domain D, written $\theta \in D$, if $\theta(v) \in D(v)$ for all $v \in vars(\theta)$. Given an expression e, $\theta(e)$ is obtained by replacing each $v \in vars(e)$ by $\theta(v)$ and calculating the value of the resulting variable free expression.

We are interested in determining the infimums and supremums of expressions with respect to some domain D. Define the *infimum* and *supremum* of an expression e with respect to a domain D as $\inf_D e = \inf\{\theta(e) | \theta \in D\}$ and $\sup_D e = \sup\{\theta(e) | \theta \in D\}$ respectively. A *range* is a contiguous set of integers, and we use *range notation*: $[\, l \,..\, u \,]$ to denote the range $\{d \in \mathcal{Z} \mid l \leq d \leq u\}$ when l and u are integers. A domain is a *range domain* if $D(x)$ is a range for all x. Let $D' = \text{range}(D)$ be the smallest range domain containing D, i.e. domain $D'(x) = [\, \inf_D x \,..\, \sup_D x \,]$ for all $x \in \mathcal{V}$.

A constraint places restriction on the allowable values for a set of variables and is usually written in well understood mathematical syntax. More formally, a *constraint* c is a relation expressed using available function and relation symbols in a specific constraint language. For the purpose of this paper, we assume the usual integer interpretation of arithmetic constraints and logical operators such as $\neg$, $\wedge$, $\vee$, $\Rightarrow$, and $\Leftrightarrow$. We call valuation θ an *integer (resp. real) solution* of c iff $vars(\theta) = vars(c)$ and $\mathcal{Z} \models \theta(c)$ ($\mathcal{R} \models \theta(c)$). We denote by $solns(c)$ all the *integer solutions* of c.

We can understand a domain D as a constraint: $D \leftrightarrow \bigwedge_{v \in \mathcal{V}} \bigvee_{d \in D(v)} v = d$. A *constraint satisfaction problem* (CSP) consists of a set of constraints read as conjunction.

A *propagator* f is a monotonically decreasing function from domains to domains, i.e. $D \sqsubseteq D'$ implies that $f(D) \sqsubseteq f(D')$, and $f(D) \sqsubseteq D$. A propagator f is *correct* for constraint c iff for all domains D

$$\{\theta \mid \theta \in D\} \cap solns(c) = \{\theta \mid \theta \in f(D)\} \cap solns(c)$$

This is a weak restriction since, for example, the identity propagator is correct for all constraints c.

Typically we model a CSP as a conjunction of constraints $\wedge_{i=1}^{n} c_i$. We provide a propagator f_i for each constraint c_i where f_i is correct and checking for c_i. The propagation solver is then applied on the set $F = \{f_i \mid 1 \leq i \leq n\}$. The consideration of individual constraints is *crucial.* By design (of what constraints are supported in a constraint language), we can provide efficient implementations of propagators for particular constraints, but not for arbitrary ones. A human modeler should exploit this fact in constructing efficient formulations of problems.

A *propagation solver* for a set of propagators F and current domain D, $solv(F, D)$, repeatedly applies all the propagators in F starting from domain D until there is no further change in the resulting domain. In other words, $solv(F, D)$ returns a new domain defined by $solv(F, D) = \text{gfp}(iter_F)(D)$ where $iter_F(D) = \bigsqcap_{f \in F} f(D)$, and gfp denotes the greatest fixpoint w.r.t. $\sqsubseteq$ lifted to functions.

Propagators are often linked to some notion of implementing some consistency for a particular constraint. A propagator f *implements* a consistency $\mathcal{C}$ for a constraint c, if $D' = solv(\{f\}, D)$ ensures that D' is the greatest domain $D' \sqsubseteq D$ that is $\mathcal{C}$ consistent for c.[1] Note that f only needs to satisfy this property at its fixpoints.

Definition 1 *A domain D is* domain consistent *for a constraint c where $vars(c) = \{x_1, \ldots, x_n\}$, if for each variable $x_i, 1 \leq i \leq n$ and for each $d_i \in D(x_i)$ there exist integers d_j with $d_j \in D(x_j)$, $1 \leq j \leq n, j \neq i$ such that $\theta = \{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$ is an integer solution of c, i.e. $\mathcal{Z} \models_\theta c$.*

We can now define a *domain propagator*, $dom(c)$, for a constraint c as:

$$dom(c)(D)(v) = \begin{cases} \{\theta(v) \mid \theta \in D \wedge \theta \in solns(c)\} & \text{where } v \in vars(c) \\ D(v) & \text{otherwise} \end{cases}$$

[1] This assumes that such a greatest domain exists. This holds for all sensible notions of consistency, including all those in this paper.

From the definition, it is clear that the domain propagator $dom(c)$ implements domain consistency for the constraint c. The domain propagator $dom(c)$ is clearly idempotent.

3 Different Notions of Bounds Consistency

The basis of bounds consistency is to relax the consistency requirement to apply only to the lower and upper bounds of the domain of each variable x. There are three incompatible definitions of bounds consistency used in the literature, all for constraints with finite integer domains. For bounds($\mathcal{D}$) consistency each bound of the domain of a variable has integer support among the values of the domain of each other variable occurring in the same constraint. For bounds($\mathcal{Z}$) consistency the integer supporting values need only be within the range from the lower to upper bounds of the other variables. For bounds($\mathcal{R}$) consistency the supports can be real values within the range from the lower to upper bounds of the other variables.

Definition 2 *A domain D is* bounds($\mathcal{D}$) consistent *for a constraint c where $vars(c) = \{x_1, \ldots, x_n\}$, if for each variable $x_i, 1 \leq i \leq n$ and for each $d_i \in \{\inf_D x_i, \sup_D x_i\}$ there exist* ***integers*** $\boldsymbol{d_j}$ *with* $\boldsymbol{d_j \in D(x_j)}$, $1 \leq j \leq n, j \neq i$ *such that $\theta = \{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$ is an* ***integer solution*** *of c.*

Definition 3 *A domain D is* bounds($\mathcal{Z}$) consistent *for a constraint c where $vars(c) = \{x_1, \ldots, x_n\}$, if for each variable $x_i, 1 \leq i \leq n$ and for each $d_i \in \{\inf_D x_i, \sup_D x_i\}$ there exist* ***integers*** $\boldsymbol{d_j}$ *with* $\boldsymbol{\inf_D x_j \leq d_j \leq \sup_D x_j}$, $1 \leq j \leq n, j \neq i$ *such that $\theta = \{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$ is an* ***integer solution*** *of c.*

Definition 4 *A domain D is* bounds($\mathcal{R}$) consistent *for a constraint c where $vars(c) = \{x_1, \ldots, x_n\}$, if for each variable $x_i, 1 \leq i \leq n$ and for each $d_i \in \{\inf_D x_i, \sup_D x_i\}$ there exist* ***real numbers*** $\boldsymbol{d_j}$ *with* $\boldsymbol{\inf_D x_j \leq d_j \leq \sup_D x_j}$, $1 \leq j \leq n, j \neq i$ *such that $\theta = \{x_1 \mapsto d_1, \ldots, x_n \mapsto d_n\}$ is a* ***real solution*** *of c.*

Definition 2 is used in for example for the two definitions in Dechter [6, pages 73 & 435]; Frisch *et al.* [7]; and implicitly in Lallouet *et al.* [12]. Definition 3 is far more widely used appearing in for example Van Hentenryck, Saraswat, & Deville [24]; Puget [19]; Régin & Rueher [21]; Quimper *et al.* [20]; and SICStus Prolog [23]. Definition 4 appears in for example Marriott & Stuckey [17]; Schulte & Stuckey [22]; Harvey & Schimpf [9]; and Zhang & Yap [27]. Apt [1] gives both Definitions 3 (called interval consistency) and 4 (called bounds consistency).

3.1 General Relationship and Properties

Let us now examine the differences of the definitions. The following relationship between the three notions of bounds consistency is clear from the definition.

Proposition 1. *If D is bounds($\mathcal{D}$) consistent for c it is bounds($\mathcal{Z}$) consistent for c. If D is bounds($\mathcal{Z}$) consistent for c it is bounds($\mathcal{R}$) consistent for c.* □

Example 1. Consider the constraint $c_{lin} \equiv x_1 = 3x_2 + 5x_3$. The domain D_2 defined by $D_2(x_1) = \{2, 3, 4, 6, 7\}$, $D_2(x_2) = [\,0\,..\,2\,]$, and $D_2(x_3) = [\,0\,..\,1\,]$ is bounds($\mathcal{R}$) consistent (but not bounds($\mathcal{D}$) consistent or bounds($\mathcal{Z}$) consistent) w.r.t. c_{lin}.

The domain D_3 defined by $D_3(x_1) = \{3, 4, 6\}$, $D_3(x_2) = [\,0\,..\,2\,]$, and $D_3(x_3) = [\,0\,..\,1\,]$ is bounds($\mathcal{Z}$) and bounds($\mathcal{R}$) consistent (but not bounds($\mathcal{D}$) consistent) w.r.t. c_{lin}.

The domain D_4 defined by $D_4(x_1) = \{3, 4, 6\}$, $D_4(x_2) = [\,1\,..\,2\,]$, and $D_4(x_3) = \{0\}$ is bounds($\mathcal{D}$), bounds($\mathcal{Z}$) and bounds($\mathcal{R}$) consistent w.r.t. c_{lin}.

The relationship between the bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency is straightforward to explain.

Proposition 2. *D is bounds($\mathcal{Z}$) consistent with c iff* range(D) *is bounds($\mathcal{D}$) consistent with c.* □

The second definition of bounds consistency in Dechter [6, page 435] works only with range domains. By Proposition 2, the definition coincides with both bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency. Similarly, Apt's [1] interval consistency is also equivalent to bounds($\mathcal{D}$) consistency. Finite domain constraint solvers do not always operate on range domains, but rather they use a mix of propagators implementing different kinds of consistencies, both domain and bounds consistency.

Example 2. Consider the same setting from Example 1. Now range(D_3) is both bounds($\mathcal{D}$) and bounds($\mathcal{Z}$) consistent with c_{lin}. As noted in Example 1, D_3 is only bounds($\mathcal{Z}$) consistent but *not* bounds($\mathcal{D}$) consistent with c_{lin}.

Both bounds($\mathcal{R}$) and bounds($\mathcal{Z}$) consistency depend only on the upper and lower bounds of the domains of the variables under consideration.

Proposition 3. *For $\alpha = \mathcal{R}$ or $\alpha = \mathcal{Z}$ and constraint c, D is bounds(α) consistent for c iff* range(D) *is bounds(α) consistent for c.* □

This is not the case for bounds($\mathcal{D}$) consistency, which suggests that, strictly, it is not really a form of *bounds consistency*. Indeed, most existing implementations of bounds propagators make use of Proposition 3 to avoid re-executing a bounds propagator unless the lower or upper bound of a variable involved in the propagator changes.

Example 3. Consider the same setting from Example 1 again. Both D_3 and range(D_3) are bounds($\mathcal{Z}$) and bounds($\mathcal{R}$) consistency with c_{lin}, but only range(D_3) is bounds($\mathcal{D}$) consistent with c_{lin}.

There are significant problems with the stronger bounds($\mathcal{Z}$) (and bounds($\mathcal{D}$)) consistency. In particular, for linear equations it is NP-complete to check bounds($\mathcal{Z}$) (and bounds($\mathcal{D}$)) consistency, while for bounds($\mathcal{R}$) consistency it is only linear time (e.g. see Schulte & Stuckey [22]).

Proposition 4. *Checking bounds($\mathcal{Z}$), bounds($\mathcal{D}$), or domain consistency of a domain D with a linear equality $a_1x_1 + \cdots a_nx_n = a_0$ is NP-complete, where $\{a_0, \ldots, a_n\}$ are integer constants and $\{x_1, \ldots, x_n\}$ are integer variables.* □

There are other constraints where bounds($\mathcal{R}$) consistency is less meaningful. A problem of bounds($\mathcal{R}$) consistency is that it may not be clear how to interpret an integer constraint in the reals.

3.2 Conditions for Equivalence

Why has the confusion between the various definitions of bounds consistency not been noted before? In fact, for many constraints, the definitions are *equivalent.*

Following the work of Zhang & Yap [27] we define n-ary monotonic constraints as a generalization of linear inequalities $\sum_{i=1}^{n} a_ix_i \leq a_0$. Let $\theta \in_{\mathcal{R}} D$ denote that $\theta(v) \in \mathcal{R}$ and $\inf_D v \leq \theta(v) \leq \sup_D v$ for all $v \in vars(\theta)$.

Definition 5 *An n-ary constraint c is* monotonic with respect to variable $x_i \in vars(c)$ *iff there exists a total ordering $\prec_i$ on $D(x_i)$ such that if $\theta \in_{\mathcal{R}} D$ is a real solution of c, then so is any $\theta' \in_{\mathcal{R}} D$ where $\theta'(x_j) = \theta(x_j)$ for $j \neq i$ and $\theta'(x_i) \preceq_i \theta(x_i)$. An n-ary constraint c is* monotonic *iff c is* monotonic with respect to all variables in $vars(c)$.

The above definition of monotonic constraints is equivalent to but simpler than that of Zhang & Yap [27], see Choi *et al.* [5] for justification and explanation. Examples of monotonic constraints are: all linear inequalities, and $x_1 \times x_2 \leq x_3$ with non-negative domains, i.e. $\inf_D(x_i) \geq 0$. For this class of constraints, bounds($\mathcal{R}$), bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency are equivalent to domain consistency.

Proposition 5. *Let c be an n-ary monotonic constraint. Then bounds($\mathcal{R}$), bounds($\mathcal{Z}$), bounds($\mathcal{D}$) and domain consistency for c are all equivalent.* □

Although linear disequality constraints are not monotonic, they are equivalent for all the forms of bounds consistency because they prune so weakly.

Proposition 6. *Let $c \equiv \Sigma_{i=1}^{n} a_ix_i \neq a_0$. Then bounds($\mathcal{R}$), bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency for c are equivalent.* □

All forms of bounds consistency are also equivalent for binary monotonic functional constraints, such as $a_1x_1 + a_2x_2 = a_0$, $x_1 = a{x_2}^2 \wedge x_2 \geq 0$, or $x_1 = 1 + x_2 + {x_2}^2 + {x_2}^3 \wedge x_2 \geq 0$.

Proposition 7. *Let c be a constraint with $vars(c) = \{x_1, x_2\}$, where $c \equiv x_1 = g(x_2)$ and g is a bijective and monotonic function. Then bounds($\mathcal{R}$), bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency for c are equivalent.* □

For linear equations with at most one non-unit coefficient, we can show that bounds($\mathcal{R}$) and bounds($\mathcal{Z}$) consistency are equivalent.

Proposition 8. *Let* $c \equiv \Sigma_{i=1}^{n} a_i x_i = a_0$*, where* $|a_i| = 1, 2 \leq i \leq n$*,* a_0 *and* a_1 *integral. Then bounds(*$\mathcal{R}$*) and bounds(*$\mathcal{Z}$*) consistency for c are equivalent.* □

Even for linear equations with all unit coefficients, bounds($\mathcal{D}$) consistency is different from bounds($\mathcal{Z}$) and bounds($\mathcal{R}$) consistency.

In summary, for many of the commonly used constraints, the notions of bounds consistency are equivalent, but clearly not for all, for example c_{lin}.

4 Different Types of Bounds Propagators

In practice, propagators implementing bounds($\mathcal{Z}$) and/or bounds($\mathcal{R}$) consistency for individual kinds of constraints are well known. Propagators implementing bounds($\mathcal{Z}$) consistency (and not bounds($\mathcal{D}$) or bounds($\mathcal{R}$) consistency) are defined for the `alldifferent` constraint in e.g. Puget [19], Mehlhorn & Thiel [18], and López-Ortiz *et al.* [14]; for the global cardinality constraint in e.g. Quimper *et al.* [20], and Katriel & Thiel [11]; and for the global constraint combining the sum and difference constraints in Régin & Rueher [21]. Propagators implementing bounds($\mathcal{R}$) consistency for common arithmetic constraints are defined in e.g. Schulte & Stuckey [22].

On the other hand, propagators implementing bounds($\mathcal{D}$) consistency explicitly are rare. The `case` constraint in SICStus Prolog [23] allows compact representation of an arbitrary constraint as a directed acyclic graph. The `case` constraint implements domain consistency by default, but there are options to make the constraint prune only the bounds of variables using bounds($\mathcal{D}$) consistency. We can also enforce the multibound-consistency operators of Lallouet *et al.* [12] to implement bounds($\mathcal{D}$) consistency by using only a single cluster.

In the following, we give *a priori* definitions of the different types of bounds propagators for an arbitrary constraint. Although these definitions are straightforward to explain, we are not aware of any previous definition.

Bounds($\mathcal{D}$*) Propagator* We can define bounds($\mathcal{D}$) propagators straightforwardly. Let c be an arbitrary constraint, then a bounds($\mathcal{D}$) propagator for c, $dbnd(c)$, can defined as $dbnd(c)(D) = D \sqcap \text{range}(dom(c)(D))$. This definition is also given in Lallouet *et al.* [12]. There they implicitly define bounds consistency as the result of applying this propagator.

The bounds($\mathcal{D}$) propagator $dbnd(c)$ implements bounds($\mathcal{D}$) consistency for the constraint c.

Theorem 1. *Given a constraint c, if* $D' = dbnd(c)(D)$*, then* D' *is the greatest domain* $D' \sqsubseteq D$ *that is bounds(*$\mathcal{D}$*) consistent for c.* □

Like the domain propagator $dom(c)$, the bounds($\mathcal{D}$) propagator $dbnd(c)$ is clearly idempotent (as a result of Theorem 1).

Bounds($\mathcal{Z}$) Propagator We can also define bounds($\mathcal{Z}$) propagators straightforwardly. Let c be an arbitrary constraint, then a bounds($\mathcal{Z}$) propagator for c, $zbnd(c)$, can be defined as $zbnd(c)(D) = D \sqcap \text{range}(dom(c)(\text{range}(D)))$.

Unlike the bounds($\mathcal{D}$) propagator $dbnd(c)$, the bounds($\mathcal{Z}$) propagator $zbnd(c)$ is *not* idempotent.

Theorem 2. *$zbnd(c)$ implements bounds($\mathcal{Z}$) consistency for constraint c.* □

Bounds($\mathcal{R}$) Propagator The basis of a bounds($\mathcal{R}$) propagator is to relax the integral requirement to reals. We define a *real domain* $\hat{D}$ as a mapping from the set of variables $\mathcal{V}$ to sets of reals. We also define a valuation θ to be an element of a real domain $\hat{D}$, written $\theta \in \hat{D}$, if $\theta(v_i) \in \hat{D}(v_i)$ for all $v_i \in vars(\theta)$. We can similarly extend the notions of *infimum* $\inf_{\hat{D}} e$ and *supremum* $\sup_{\hat{D}} e$ of an expression e with respect to real domain $\hat{D}$.

We will define the behavior of bounds($\mathcal{R}$) propagators by extending the definition of domain propagators to reals. Given a constraint c and a real domain $\hat{D}$, define the *real domain propagator*, $rdom(c)$, as

$$rdom(c)(\hat{D})(v) = \begin{cases} \{\theta(v) \mid \theta \in \hat{D} \wedge \mathcal{R} \models_\theta c\} & \text{where } v \in vars(c) \\ \hat{D}(v) & \text{otherwise} \end{cases}$$

We use *interval* notation, $[l—u]$, to denote the set $\{d \in \mathcal{R} \mid l \leq d \leq u\}$ where l and u are reals. Let $\hat{D} = \text{real}(D)$ be the real domain $\hat{D}(v) = [\inf_D(v)—\sup_D(v)]$ for all $v \in \mathcal{V}$. Let $D' = \text{integral}(\hat{D})$ be the domain $D'(v) = [\,\lceil \inf_{\hat{D}}(v) \rceil \,..\, \lfloor \sup_{\hat{D}}(v) \rfloor\,]$ for all $v \in \mathcal{V}$.

We can now define bounds($\mathcal{R}$) propagators straightforwardly. This is the first time (that we are aware of) that this has been formalized. Let c be an arbitrary constraint, then the bounds($\mathcal{R}$) propagator for c, $rbnd(c)$ is defined as

$$rbnd(c)(D) = D \sqcap \text{integral}(rdom(c)(\text{real}(D))).$$

Similar to bounds($\mathcal{Z}$) propagators, the previous example clearly shows that bounds($\mathcal{R}$) propagators are *not* idempotent. The bounds($\mathcal{R}$) propagator does not guarantee bounds($\mathcal{R}$) consistency except at its fixpoints.

Theorem 3. *$rbnd(c)$ implements bounds($\mathcal{R}$) consistency for constraint c.* □

5 Related Work

In this paper we consider *integer* constraint solving. Definitions of bounds consistency for real constraints are also numerous, but their similarities and differences have been noted and explained by e.g. Benhamou *et al.* [2]. Indeed, we can always interpret integers as reals and apply bounds consistency for real constraints plus appropriate rounding, e.g. CLP(BNR) [3]. However, as we have pointed out in Section 3.1, there exist integer constraints for which propagation is less meaningful when interpreted as reals.

Lhomme [13] defines *arc B-consistency*, which formalizes bounds propagation for both integer and real constraints. He proposes an efficient propagation algorithm implementing arc B-consistency with complexity analysis and experimental results. However, his study focuses on constraints defined by numeric relations (i.e. numeric CSPs).

Walsh [25] introduces several new forms of bounds consistency that extend the notion of (i, j)-consistency and relational consistency. He gives theoretical analysis comparing the propagation strength of these new consistency forms.

Maher [16] introduces the notion of propagation completeness with a general framework to unify a wide range of consistency. These include hull consistency of real constraints and bounds($\mathcal{Z}$) consistency of integer constraints. Propagation completeness aims to capture the timeliness property of propagation.

The application of bounds consistency is not limited to integer and real constraints. Bounds consistency has been formalized for solving set constraints [8], and more recently, multiset constraints [26].

6 Conclusion

The contributions of this paper are two-fold. First, we point out that the three commonly used definitions of bounds consistency are incompatible. We clarify their differences and study what is actually implemented in existing systems. We show that for several types of constraints, bounds($\mathcal{R}$), bounds($\mathcal{Z}$) and bounds($\mathcal{D}$) consistency are equivalent. This explains partly why the discrepancies among the definitions were not noticed earlier. Second, we give *a priori* definitions of propagators that implement the three notions of bounds consistency, which can serve as the basis for verifying all implementations of bounds propagators.

Acknowledgements

We thank the anonymous referees for their constructive comments. The work described in this paper was substantially supported by grant number CUHK4219/04E from the Research Grants Council of Hong Kong SAR.

References

1. K. Apt. *Principles of Constraint Programming.* Cambridge University Press, 2003.
2. F. Benhamou, D. McAllester, and P. Van Hentenryck. CLP(Intervals) revisited. In *ILPS 1994*, pages 124–138, 1994.
3. F. Benhamou and W. J. Older. Applying interval arithmetic to real, integer, and boolean constraints. *JLP*, 32(1):1–24, 1997.
4. A. Cheadle, W. Harvey, A. Sadler, J. Schimpf, K. Shen, and M. Wallace. $\mathrm{ECL}^i\mathrm{PS}^e$: An introduction. Technical Report IC-Parc-03-1, IC-Parc, Imperial College London, 2003.
5. C. W. Choi, W. Harvey, J. H. M. Lee, and P. J. Stuckey. A note on the definition of constraint monotonicity. Available from `http://www.cse.cuhk.edu.hk/~cwchoi/monotonicity.pdf`, 2004.

6. R. Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.
7. A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, and T. Walsh. Global constraints for lexicographic orderings. In *CP2002*, pages 93–108. Springer-Verlag, 2002.
8. C. Gervet. Interval propagation to reason about sets: Definition and implementation of a practical language. *Constraints*, 1(3):191–244, 1997.
9. W. Harvey and J. Schimpf. Bounds consistency techniques for long linear constraints. In *Proceedings of TRICS: Techniques foR Implementing Constraint programming Systems*, pages 39–46, 2002.
10. ILOG. *ILOG Solver 5.2: User's Manual*, 2001.
11. I. Katriel and S. Thiel. Fast bound consistency for the global cardinality constraint. In *CP 2003*, pages 437–451, 2003.
12. A. Lallouet, A. Legtchenko, T. Dao, and A. Ed-Dbali. Intermediate (learned) consistencies. Research Report RR-LIFO-2003-04, Laboratoire d'Informatique Fondamentale d'Orléans, 2003.
13. O. Lhomme. Consistency techniques for numeric CSPs. In *IJCAI 93*, pages 232–238, 1993.
14. A. López-Ortiz, C.-G. Quimper, J. Tromp, and P. van Beek. A fast and simple algorithm for bounds consistency of the alldifferent constraint. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003)*, pages 245–250, 2003.
15. A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.
16. M. Maher. Propagation completeness of reactive constraints. In *ICLP 2002*, pages 148–162, 2002.
17. K. Marriott and P. J. Stuckey. *Programming with Constraints: an Introduction*. The MIT Press, 1998.
18. K. Mehlhorn and S. Thiel. Faster algorithms for bound-consistency of the sortedness and the alldifferent constraint. In *CP2000*, pages 306–319, 2000.
19. J.-F. Puget. A fast algorithm for the bound consistency of alldiff constraints. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI 98)*, pages 359–366, 1998.
20. C.-G. Quimper, P. van Beek, A. López-Ortiz, A. Golynski, and S. B. Sadjad. An efficient bounds consistency algorithm for the global cardinality constraint. In *CP 2003*, pages 600–614, 2003.
21. J.-C. Régin and M. Rueher. A global constraint combining a sum constraint and difference constraints. In *CP 2000*, pages 384–395, 2000.
22. C. Schulte and P. J. Stuckey. When do bounds and domain propagation lead to the same search space. In *Proceedings of the 3rd International Conference on Principles and Practice of Declarative Programming (PPDP 2001)*, pages 115–126, 2001.
23. SICStus Prolog. *SICStus Prolog User's Manual, Release 3.10.1*, 2003.
24. P. Van Hentenryck, V. Saraswat, and Y. Deville. Design, implementation and evaluation of the constraint language cc(FD). *Journal of Logic Programming*, 37(1–3):139–164, 1998.
25. T. Walsh. Relational consistencies. Research Report APES-28-2001, APES Research Group, 2001.
26. T. Walsh. Consistency and propagation with multiset constraints: A formal viewpoint. In *Proceedings of the 9th International Conference on Principles and Practice of Constraint Programming (CP 2003)*, pages 724–738, 2003.
27. Y. Zhang and R. H. C. Yap. Arc consistency on n-ary monotonic and linear constraints. In *CP 2000*, pages 470–483, 2000.