Widening Operators for Weakly-Relational Numeric Abstractions*

Roberto Bagnara[1], Patricia M. Hill[2], Elena Mazzi[1], and Enea Zaffanella[1]

[1] Department of Mathematics, University of Parma, Italy
{bagnara,mazzi,zaffanella}@cs.unipr.it
[2] School of Computing, University of Leeds, UK
hill@comp.leeds.ac.uk

Abstract. We discuss the construction of proper widening operators on several weakly-relational numeric abstractions. Our proposal differs from previous ones in that we actually consider the semantic abstract domains, whose elements are *geometric shapes*, instead of the (more concrete) syntactic abstract domains of constraint networks and matrices. Since the closure by entailment operator preserves geometric shapes, but not their syntactic expressions, our widenings are immune from the divergence issues that could be faced by the previous approaches when interleaving the applications of widening and closure. The new widenings, which are variations of the *standard widening* for convex polyhedra defined by Cousot and Halbwachs, can be made as precise as the previous proposals working on the syntactic domains. The implementation of each new widening relies on the availability of an effective reduction procedure for the considered constraint description: we provide such an algorithm for the domain of *octagonal shapes*.

1 Introduction

Numerical properties are of great interest in the broad area of formal methods for their complete generality and since they often play a crucial role in the definition of static analyses and program verification techniques. In the field of abstract interpretation, classes of numerical properties are captured by numerical abstract domains. These have been and are widely used, either as the main abstraction for the application at hand, or as powerful ingredients to improve the precision of other abstract domains.

Among the wide spectrum of numerical abstractions proposed in the literature, the most famous ones are probably the (non-relational) abstract domain of intervals [16] and the (relational) abstract domain of convex polyhedra [19]. As far as the efficiency/precision trade-off is concerned, these domains occupy the opposite extremes of the spectrum: on the one hand, the operations on convex

* This work has been partly supported by MURST projects "Constraint Based Verification of Reactive Systems" and "AIDA — Abstract Interpretation: Design and Applications," and by a Royal Society (UK) International Joint Project (ESEP) award.

polyhedra achieve a significant level of precision, which is however countered by a worst-case exponential time complexity, often leading to scalability problems; on the other hand, the great efficiency of the corresponding operations on intervals is made unappealing by the fact that the obtained precision is often unsatisfactory. This well-known dichotomy (which does not impede that, for some applications, convex polyhedra or intervals are the right choices) has motivated recent studies on several abstract domains that lie somehow between these two extremes, and can therefore be called *weakly-relational* abstract domains. Examples include domains based on constraint networks [3–5], the abstract domain of difference-bound matrices [25, 32], the octagon abstract domain [26], the 'two variables per inequality' abstract domain [33], the octahedron abstract domain [15], and the abstract domain of template constraint matrices [31]. Moreover, similar proposals that are not abstractions of the domain of convex polyhedra have been put forward, including the abstract domain of bounded quotients [3] and the zone congruence abstract domain [27].

In this paper, we address the issue of the provision of proper widening operators for these domains. For the abstract domain of convex polyhedra, all the widenings that have been proposed are variations of, and/or improvements to, what is commonly referred to as the *standard widening* [19, 22]. This is based on the general widening principle "drop the unstable components" applied to constraints. Not surprisingly, most proposals for widening operators for the weakly relational domains are based on the same principle and analogous to the standard widening. For instance, for the domain of difference bound matrices mentioned above, an operator meant to match the standard widening is given in [32]. Unfortunately, as pointed out in [25, 26], this operator is not a widening, since it has no convergence guarantee. The reason is that *closure by entailment*, which is systematically performed so as to provide a canonical form for the elements and to improve the precision of several domain operations, has a negative interaction with the extrapolation operator of [32] that compromises the convergence guarantee. Intuitively, what can happen is that, while the extrapolation operator discards unstable constraints, the closure operation reinserts them (because they were redundant): failure to drop such unstable constraints can (and, in practice, quite often does) result in infinite upward iteration sequences. For this reason, it is proposed in [25, 26] to apply the same operator given in [32] to the "syntactic" version of the same abstract domain, that is, where closure is only very carefully applied during the fixpoint computations.

We have taken a different approach and resolve the apparent conflict by considering a "semantic" abstract domain whose elements are the geometric shapes themselves. Since closure by entailment preserves the geometric shapes (even though this does not preserve their syntactic expressions), the approach is immune from the divergence problem described above. On the other hand, in order to use the standard widening as the basis of the proposed widening, it is important that we can compute *reduced* representations of the domain elements that encode non-redundant systems of constraints. Thus the implementations of any new widenings based on the semantic approach will need effective reduction procedures for the considered constraint description: here we provide such an algorithm for the domain of *octagonal shapes*. As a by-product of our work on verifying the correctness of this reduction algorithm, we noticed that the algorithm for computing the strong closure of octagonal graphs as described in [25] could be simplified with a consequential improvement in its efficiency. This revised strong closure algorithm is also described here.

The paper is structured as follows: Section 2 recalls the required concepts and notations; Section 3 introduces the domain of bounded difference graphs; a domain of bounded difference shapes is presented in Section 4, where an alternative solution to the divergence problem is proposed; the generalization of the above results to the case of octagons is the subject of Section 5, where we define a new strong reduction procedure and an improved strong closure procedure for octagonal graphs, as well as a semantic widening operator for octagonal shapes. Section 6 concludes with a discussion of the results achieved. The proofs of all the stated results can be found in [6].

2 Preliminaries

The reader is assumed to be familiar with the fundamental concepts of lattice theory [13] and abstract interpretation theory [17, 18]. We refer the reader to the classical works on the numeric domains of intervals [16] and convex polyhedra [19] for the specification of the corresponding widening operators.

Let $\mathbb{Q}_\infty := \mathbb{Q} \cup \{+\infty\}$ be totally ordered by the extension of '$<$' such that $d < +\infty$ for each $d \in \mathbb{Q}$. Let $\mathcal{N}$ be a finite set of *nodes*. A *weighted directed graph* (graph, for short) G in $\mathcal{N}$ is a pair $(\mathcal{N}, w)$, where $w \colon \mathcal{N} \times \mathcal{N} \to \mathbb{Q}_\infty$ is the weight function for G. A pair $(n_i, n_j) \in \mathcal{N} \times \mathcal{N}$ is an *arc* of G if $w(n_i, n_j) < +\infty$; the arc is *proper* if $n_i \neq n_j$.

A *path* $\pi = n_0 \cdots n_p$ in a graph $G = (\mathcal{N}, w)$ is a non-empty and finite sequence of nodes such that (n_{i-1}, n_i) is an arc of G, for all $i = 1, \ldots, p$; each arc (n_{i-1}, n_i) where $i = 1, \ldots, p$ is said to be *in* the path π. The path π is *proper* if all the arcs in it are proper. The path π is a *proper cycle* if it is a proper path and $n_0 = n_p$ (so that $p \geq 2$). The *length* of the path π is the number p of occurrences of arcs in π and denoted by $\|\pi\|$; the *weight* of the path π is $\sum_{i=1}^{p} w(n_{i-1}, n_i)$ and denoted by $w(\pi)$. The path π is a *zero-cycle* if it is a proper cycle with 0 weight. A graph is *consistent* if it has no negative weight cycles; it is *zero-cycle free* if all its proper cycles have strictly positive weights.

The set $\mathbb{G}$ of consistent graphs in $\mathcal{N}$ is partially ordered by the relation '$\trianglelefteq$' defined, for all $G_1 = (\mathcal{N}, w_1)$ and $G_2 = (\mathcal{N}, w_2)$, by

$$G_1 \trianglelefteq G_2 \quad\iff\quad \forall i, j \in \mathcal{N} : w_1(i, j) \leq w_2(i, j).$$

When augmented with a bottom element $\bot$ representing inconsistency, this partially ordered set becomes a (non-complete) lattice $\mathbb{G}_\bot = \langle \mathbb{G} \cup \{\bot\}, \trianglelefteq, \sqcap, \sqcup \rangle$, where '$\sqcap$' and '$\sqcup$' denote the (finitary) greatest lower bound and least upper bound operators, respectively.

Definition 1. (Closed graph.) *A consistent graph* $G = (\mathcal{N}, w)$ *is* closed *if the following properties hold:*

$$\forall i \in \mathcal{N} : w(i,i) = 0; \tag{1}$$
$$\forall i,j,k \in \mathcal{N} : w(i,j) \leq w(i,k) + w(k,j). \tag{2}$$

The (shortest-path) closure *of a consistent graph* G *in* $\mathcal{N}$ *is*

$$\mathrm{closure}(G) := \bigsqcup \bigl\{\, G^{\mathrm{c}} \in \mathbb{G} \bigm| G^{\mathrm{c}} \trianglelefteq G \textit{ and } G^{\mathrm{c}} \textit{ is closed} \,\bigr\}.$$

When trivially extended so as to behave as the identity function on the bottom element $\bot$, shortest-path closure is a kernel operator (monotonic, idempotent and reductive) on the lattice $\mathbb{G}_\bot$.

3 Systems of Bounded Differences

The typical way to simplify the domain of convex polyhedra is by restricting attention to particular subclasses of linear inequalities. One possibility, which has a long tradition in computer science [12], is to only consider *potential constraints*, also known as *bounded differences*: these are restricted to take the form $v_i - v_j \leq d$ or $\pm v_i \leq d$. Systems of bounded differences have been used by the artificial intelligence community as a way to reason about temporal quantities [2, 20], as well as by the model checking community as an efficient yet precise way to model and propagate timing requirements during the verification of various kinds of concurrent systems [21, 24]. In the abstract interpretation field, the idea of using an abstract domain of bounded differences was put forward in [3].

A finite system $\mathcal{C}$ of bounded differences on variables $\mathcal{V} = \{v_0, \dots, v_{n-1}\}$ can be represented by a weighted directed graph $G = (\mathcal{N}_{\mathbf{0}}, w)$ where $\mathcal{N}_{\mathbf{0}} = \{\mathbf{0}\} \cup \mathcal{V}$, $\mathbf{0} \notin \mathcal{V}$ is the *special variable*, and the weight function w is defined, for each $v_i, v_j \in \mathcal{N}_{\mathbf{0}}$, by

$$w(v_i, v_j) := \begin{cases} \min\bigl\{\, d \in \mathbb{Q} \bigm| (v_i - v_j \leq d) \in \mathcal{C} \,\bigr\}, & \text{if } v_i \neq \mathbf{0} \text{ and } v_j \neq \mathbf{0}; \\ \min\bigl\{\, d \in \mathbb{Q} \bigm| (v_i \leq d) \in \mathcal{C} \,\bigr\}, & \text{if } v_i \neq \mathbf{0} \text{ and } v_j = \mathbf{0}; \\ \min\bigl\{\, d \in \mathbb{Q} \bigm| (-v_j \leq d) \in \mathcal{C} \,\bigr\}, & \text{if } v_i = \mathbf{0} \text{ and } v_j \neq \mathbf{0}; \\ 0, & \text{if } v_i = v_j = \mathbf{0}. \end{cases}$$

Notice that we assume that $\min \varnothing = +\infty$; moreover, unary constraints are encoded by means of the special variable, which is meant to always have value 0. A possible representation of (the weight function of) the graph G is by means of a matrix-like data structure called *Difference-Bound Matrix* (DBM) [12]. However, this representation provides no conceptual advantage over the isomorphic graph (or *constraint network* [20]) representation. For this reason we will consistently adopt the terminology and notation introduced in Section 2 for weighted directed graphs. In particular, a graph encoding a consistent system of bounded differences will be called a *Bounded Difference Graph* (BDG).

The first fully developed application of bounded differences in the field of abstract interpretation can be found in [32], where an abstract domain of closed BDGs is defined. In this case, the shortest-path closure requirement was meant as a simple and well understood way to obtain a canonical form for the domain elements by abstracting away from the syntactic details; since, basically, it corresponds to the *closure by entailment* of the encoded system of bounded differences. In [32] the specification of all the required abstract semantics operators is provided, including an operator that is meant to match the widening operators defined on more classical numeric domains. This operator can be interpreted either as a generalization for closed BDGs of the widening operator defined on the abstract domain of intervals [16], or as a restriction on the domain of closed BDGs of the standard widening defined on the abstract domain of convex polyhedra [19, 22]: its implementation is based on the following upper bound operator on the set of consistent graph representations.

Definition 2. (Widening graphs.) *Let $G_1 = (\mathcal{N}, w_1)$ and $G_2 = (\mathcal{N}, w_2)$ be consistent graphs. Then $G_1 \nabla G_2 := (\mathcal{N}, w)$, where the weight function w is defined, for each $i, j \in \mathcal{N}$, by*

$$w(i,j) := \begin{cases} w_1(i,j), & \text{if } w_1(i,j) \geq w_2(i,j); \\ +\infty, & \text{otherwise.} \end{cases}$$

Unfortunately, as pointed out in [25, 26], when used in conjunction with shortest-path closure, this extrapolation operator does not provide a convergence guarantee for fixpoint computations, hence it is not a widening. The reason is that, whereas the closure operation adds redundant constraints to the input BDG, a key requirement in the specification of the standard widening is that the first argument polyhedron must be described by a non-redundant system of constraints.[3] Thus we have a "conflict of interest" between the use of a convenient canonical form for the abstract domain —a form that also allows for increased precision of several domain operations— and the requirements of the widening.

The abstract domain of BDGs has been reconsidered in [25]. Differently from [32], in [25] BDGs are not required to be closed. In this more concrete, syntactic domain, the shortest-path closure operator maps each domain element into the smallest BDG encoding the same geometric shape. Closure is typically used as a preprocessing step before the application of most, though not all, of the abstract semantic operators, allowing for improved accuracy in the results of the abstract computation. The same widening operator proposed in [32] is also used in [25]; however, it is observed that this widening "could have intriguing interactions" with shortest-path closure, therefore identifying the divergence issue faced in [32]. This observation led the author of [25] to the adoption of the syntactic domain of BDGs, where closure is not enforced.

[3] This requirement was sometimes neglected in recent papers describing the standard widening on convex polyhedra; it was recently recalled and exemplified in [7, 8]. Note that a similar requirement is implicitly present even in the specification of the widening on intervals.

4 Bounded Difference Shapes

While the analysis of the divergence problem is absolutely correct, the solution identified in [25] is sub-optimal since, as is usually the case, resorting to a syntactic domain (such as the one of BDGs) has a number of negative consequences, some of which will be recalled in Section 6.

To identify a simpler, more natural solution, we first have to acknowledge that an element of our abstract domain should be a geometric shape, rather than (any) one of its graph representations. To stress this concept, such an element will be called a *Bounded Difference Shape* (BDS). A BDS corresponds to the equivalence class of all the BDGs representing it. The implementation of the abstract domain can freely choose between these possible representations, switching at will from one to the other, as long as the semantic operators are implemented as expected. Notice that, in such a context, the shortest-path closure operator is just a transparent implementation detail: on the abstract domain of BDSs it corresponds to the identity function.

The other step towards the solution of the divergence problem is the simple observation that a BDS is a convex polyhedron and the set of all BDSs is closed under the application of the standard widening on convex polyhedra. Thus, no divergence problem can be incurred when applying the standard widening to an increasing sequence of BDSs. As mentioned in Section 3, a crucial requirement in the specification of the standard widening is that the first argument polyhedron is described by a non-redundant system of constraints [7, 8]. Thus it is not surprising that using closed BDGs has problems since it is very likely that they will encode redundant constraints. By contrast, we propose the use of a maximal BDG in the equivalence class of BDGs representing the same geometric shape; since such a graph encodes no redundant constraints at all.

Definition 3. (Reduced graph.) *A consistent graph G_1 is* reduced *if, for each consistent graph $G_2 \neq G_1$ such that $G_1 \trianglelefteq G_2$, we have* $\mathrm{closure}(G_1) \neq \mathrm{closure}(G_2)$*. A* reduction *for the consistent graph G is any reduced graph G_r such that* $\mathrm{closure}(G) = \mathrm{closure}(G_\mathrm{r})$*.*

Hence, a graph is reduced if it is maximal in the subset of graphs having the same shortest-path closure. In order to provide a correct and reasonably efficient implementation of the standard widening on the domain of BDSs, all we need is a reduction procedure mapping a BDG representation into (any) one of the equivalent reduced graphs. Such an algorithm was defined in [24] and called *shortest-path reduction*. Basically, it is an extension of the transitive reduction algorithm of [1] to the case of weighted directed graphs. Note that, since each equivalence class may have many maximal elements, shortest-path reduction is not a properly defined operator on the domain of BDGs. However, the shortest-path reduction algorithm of [24] provides a canonical form as soon as we fix a total order for the nodes in the graph.

In summary, the solution to the divergence problem for BDSs is to apply the operator specified in Definition 2 to a reduced BDG representation of the first argument of the widening. From the point of view of the user, this will be a transparent implementation detail: on the domain of BDSs, shortest-path reduction is the identity function, as was the case for shortest-path closure.

4.1 On the Precision of the Standard Widening

The standard widening on BDSs could result, if used with no precautions, in poorer precision with respect to its counterpart defined on the syntactic domain of BDGs. For increased precision, the specification of [25] prescribes two conditions that the abstract iteration sequence must satisfy:

1. the second argument of the widening should be represented by a closed BDG (note that, in this case, no divergence problem can arise);
2. the first BDG of the abstract iteration sequence $G_0 \trianglelefteq G_1 \trianglelefteq \ldots \trianglelefteq G_i \trianglelefteq \ldots$ should be closed too.

The effects of both improvements can be obtained also with the semantic domain of BDSs. As for the first one, this can be applied as is, leading to an implementation where the two arguments of the widening are represented by a reduced BDG and a closed BDG, respectively. The result of such a widening operator will depend on the specific reduced form computed for the first argument. The second precision improvement can be achieved by applying the well-known 'widening up to' technique defined in [23] or its variation called 'staged widening with thresholds' [14, 29]: in practice, it is sufficient to add to the set of 'up to' thresholds all the constraints of the shortest-path closure of the first BDG G_0. Further precision improvements can be obtained by applying any delay strategy and/or the framework defined in [7, 8].

5 Octagonal Graphs and Shapes

From a theoretical point of view, the observations made in the previous section are immediately applicable to any other weakly-relational numeric domain whose elements are convex polyhedra and is closed with respect to the application of the standard widening, therefore including the domains proposed in [15, 26, 31, 33]. From a practical perspective, the success of such a construction depends on the availability of a reasonably efficient reduction procedure for the considered subclass of constraints, because the minimization algorithm for arbitrary linear inequality constraints is not efficient enough. In this section we provide such a reduction procedure for the *octagon* abstract domain [26].

The octagon abstract domain allows for the manipulation of *octagonal* constraints of the form $av_i + bv_j \leq c$, where $a, b \in \{-1, 0, +1\}$ (the same class of constraints was considered in [11], where octagons were called *simple sections*). Bounded differences can then be used to express octagonal constraints by splitting each variable $v_i \in \mathcal{V}$ into two forms: a positive form v_i^+, interpreted as $+v_i$; and a negative form v_i^-, interpreted as $-v_i$. Thus, an octagonal constraint such as $v_i + v_j \leq d$ can be translated into the bounded difference constraint $v_i^+ - v_j^- \leq d$;

alternatively, the same constraint can be translated into $v_j^+ - v_i^- \leq d$. Note that unary (octagonal) constraints such as $v_i \leq d$ and $-v_j \leq d$ can be encoded as $v_i^+ - v_i^- \leq 2d$ and $v_j^- - v_j^+ \leq 2d$, respectively, so that the special variable **0** is no longer needed.

In the following we assume that $\mathcal{N}^\pm = \{0, \ldots, 2n-1\}$ is a fixed and finite set of nodes where, for all $i = 0, \ldots, n-1$, the node $2i$ represents the positive form v_i^+ and $2i+1$ the negative form v_i^- of the variable v_i. Moreover, for all $i \in \mathcal{N}^\pm$, $\bar{\imath}$ denotes $i+1$ if i is even, and $i-1$ if i is odd. Thus, for all $i \in \mathcal{N}^\pm$, we also have $\bar{\imath} \in \mathcal{N}^\pm$ and $\bar{\bar{\imath}} = i$. Therefore, any finite system of octagonal constraints on the n variables $\mathcal{V} = \{v_0, \ldots, v_{n-1}\}$ can be represented by a weighted directed graph on the $2n$ nodes $\mathcal{N}^\pm$. Note that, for any $i, j \in \mathcal{N}^\pm$, as arcs (i, j) and $(\bar{\jmath}, \bar{\imath})$ denote equivalent expressions, the pair is said to be *coherent*. We restrict attention to consistent systems of constraints and hence to consistent graphs where coherent pairs of arcs have the same weight.

Definition 4. (Octagonal graph.) *An* octagonal graph *in $\mathcal{N}^\pm$ is any consistent graph $G = (\mathcal{N}^\pm, w)$ satisfying the coherence assumption:*

$$\forall i, j \in \mathcal{N}^\pm : w(i, j) = w(\bar{\jmath}, \bar{\imath}). \tag{3}$$

Thus any octagonal graph on the $2n$ nodes $\mathcal{N}^\pm$ encodes a consistent system of octagonal constraints on n variables. The set $\mathbb{O}$ of all octagonal graphs, with the usual addition of the bottom element representing the empty octagon, is a sub-lattice of $\mathbb{G}_\perp$, sharing the same least upper bound and greatest lower bound operators. Note that, at the implementation level, coherence can be automatically and efficiently enforced by letting arc (i, j) and arc $(\bar{\jmath}, \bar{\imath})$ share the same representation.

The octagon abstract domain developed in [26] is thus a syntactic domain having octagonal graphs as elements. When dealing with octagonal graphs, one has to remember the relation linking the positive and negative forms of each variable: in particular, besides transitivity, a proper closure by entailment procedure should also consider the following inference rule:

$$\frac{i - \bar{\imath} \leq d_1 \qquad \bar{\jmath} - j \leq d_2}{2(i - j) \leq d_1 + d_2} \tag{4}$$

Thus, the standard shortest-path closure algorithm is not enough to obtain a canonical form for octagonal graphs: to this end, a modified closure procedure is defined in [26], yielding *strongly closed* octagonal graphs.

Definition 5. (Strongly closed graph.) *An octagonal graph $G = (\mathcal{N}^\pm, w)$ is* strongly closed *if it is closed and the following property holds:*

$$\forall i, j \in \mathcal{N}^\pm : 2w(i, j) \leq w(i, \bar{\imath}) + w(\bar{\jmath}, j). \tag{5}$$

The strong closure *of an octagonal graph G in $\mathcal{N}^\pm$ is*

$$\mathrm{Closure}(G) := \bigsqcup \bigl\{\, G^{\mathrm{C}} \in \mathbb{O} \bigm| G^{\mathrm{C}} \trianglelefteq G \text{ and } G^{\mathrm{C}} \text{ is strongly closed} \,\bigr\}.$$

Similarly to shortest-path closure, strong closure is a kernel operator on the lattice of octagonal graphs.

By repeating the reasoning of the previous section, we define the semantic abstract domain of *octagonal shapes*, whose elements are equivalence classes of octagonal graphs representing the same geometric shape. Hence, strong closure maps an octagonal graph representation of a non-empty octagonal shape into the minimum element of the corresponding equivalence class. The dual procedure, mapping the octagonal graph into (any) one of the maximal elements in its equivalence class, is called *strong reduction*.

Definition 6. (Strongly reduced graph.) *An octagonal graph G_1 is* strongly reduced *if, for each octagonal graph $G_2 \neq G_1$ such that $G_1 \trianglelefteq G_2$, we have* $\mathrm{Closure}(G_1) \neq \mathrm{Closure}(G_2)$. *A* strong reduction *for the octagonal graph G is any strongly reduced octagonal graph G_{R} such that* $\mathrm{Closure}(G) = \mathrm{Closure}(G_{\mathrm{R}})$.

Note that, in the above definition, we only compare G_1 with other *octagonal* graphs. Thus, we explicitly disregard those trivial redundancies that are due to the coherence assumption. This is not a real problem because, as discussed before, any reasonable implementation will automatically and efficiently filter away these kinds of redundancies.

5.1 A Strong Reduction Procedure for Octagonal Graphs

In this section we generalize the shortest-path reduction algorithm of [24] so as to obtain a strong reduction procedure for octagonal graphs. Clearly, the algorithm of [24] cannot be used without modifications, since it takes no account of the redundancies caused by the new constraint inference rule (4). Nonetheless, the high-level structure of the strong reduction procedure is the same as that defined in [24] for shortest-path reduction:

1. Compute the closure by entailment of the constraint graph;
2. Partition the nodes into equivalence classes based on equality constraints;
3. Decompose the graph so as to separate those arcs that link different equivalence classes (encoding only inequalities) from the partition information (encoding the equivalence classes themselves, i.e., all the equalities);
4. Reduce the subgraph that gives constraints on different equivalence classes;
5. Reduce the partition information;
6. Merge the results of steps 4 and 5 to obtain the reduced constraint graph.

We now describe each of the above steps, formally stating the correctness of the overall procedure.

Step 1 of the algorithm can be performed by applying the strong closure procedure defined in [26].

Step 2 is also easily implemented by observing that, in a strongly closed octagonal graph, equality constraints correspond to proper zero-cycles having length two.

Definition 7. (Zero-equivalence.) *Let $G = (\mathcal{N}^{\pm}, w)$ be a strongly closed octagonal graph. The nodes $i, j \in \mathcal{N}^{\pm}$ are* zero-equivalent *in G, denoted $i \equiv_G j$, if and only if $w(i, j) = -w(j, i)$.*

While step 6 carries over from BDGs to octagonal graphs, the formal definition of steps 3–5 of the reduction algorithm is more difficult for octagonal graphs than it was for BDGs, as it requires some understanding of the structure of the zero-equivalence classes.

As a first observation, note that $i \equiv_G j$ if and only if $\bar{\imath} \equiv_G \bar{\jmath}$. Therefore, if $\mathcal{E} \subseteq \mathcal{N}^{\pm}$ is a zero-equivalence class for the strongly closed octagonal graph G, then $\overline{\mathcal{E}} := \{ \bar{\imath} \in \mathcal{N}^{\pm} \mid i \in \mathcal{E} \}$ is also a zero-equivalence class for G. We say that $\mathcal{E}$ is *non-singular* if $\mathcal{E} \cap \overline{\mathcal{E}} = \varnothing$, and *singular* if $\mathcal{E} = \overline{\mathcal{E}}$; there is at most one singular zero-equivalence class in G. We associate to each zero-equivalence class $\mathcal{E} \subseteq \mathcal{N}^{\pm}$ a *leader* $\ell_{\mathcal{E}} := \min \mathcal{E}$; the class having the leader in positive (resp., negative) form will be said to be a positive (resp., negative) zero-equivalence class. Thus, the singular zero-equivalence class, if present, is always positive and, for non-singular zero-equivalence classes $\mathcal{E}$ and $\overline{\mathcal{E}}$, we have $\ell_{\overline{\mathcal{E}}} = \overline{\ell}_{\mathcal{E}}$.

We are now ready to provide a formal specification for step 3 of the strong reduction algorithm. As was the case in [24], the first subgraph resulting from the decomposition, relating nodes in different zero-equivalence classes, is obtained by only connecting the leaders. However, we do not connect the leader of the singular zero-equivalence class to the other leaders. The second subgraph only encodes those constraints relating nodes in the same zero-equivalence class.

Definition 8. (Non-singular leaders and zero-equivalence subgraphs.) *Let $G = (\mathcal{N}^{\pm}, w)$ be a strongly closed octagonal graph and $\mathcal{L} \subseteq \mathcal{N}^{\pm}$ the set of leaders of the non-singular zero-equivalence classes for G. The* non-singular leaders' subgraph *of G is the graph $L = (\mathcal{N}^{\pm}, w_{\mathrm{L}})$, where the weight function w_{L} is defined, for each $i, j \in \mathcal{N}^{\pm}$, by*

$$w_{\mathrm{L}}(i, j) := \begin{cases} w(i, j), & \text{if } i = j \text{ or } \{i, j\} \subseteq \mathcal{L}; \\ +\infty, & \text{otherwise.} \end{cases}$$

The zero-equivalence subgraph *of G is the graph $E = (\mathcal{N}^{\pm}, w_{\mathrm{E}})$, where the weight function w_{E} is defined, for each $i, j \in \mathcal{N}^{\pm}$, by*

$$w_{\mathrm{E}}(i, j) := \begin{cases} w(i, j), & \text{if } i \equiv_G j; \\ +\infty, & \text{otherwise.} \end{cases}$$

Step 4 of the strong reduction algorithm is implemented by checking, for each proper arc in the non-singular leaders' subgraph, whether it can be obtained from the other arcs by a single application of the constraint inference rules. Once again, note that we disregard redundancies caused by the coherence assumption.

Definition 9. (Strongly atomic arc and subgraph.) *Let $G = (\mathcal{N}^{\pm}, w)$ be an octagonal graph. An arc (i, j) of G is* atomic *if it is proper and, for all*

$k \in \mathcal{N}^{\pm} \setminus \{i, j\}$, $w(i,j) < w(i,k) + w(k,j)$. *The arc* (i,j) *is* strongly atomic *if it is atomic and either* $i = \overline{\jmath}$ *or* $2w(i,j) < w(i,\overline{\imath}) + w(\overline{\jmath}, j)$.

The strongly atomic subgraph *of* G *is the graph* $A = (\mathcal{N}^{\pm}, w_{\mathrm{A}})$ *where the weight function* w_{A} *is defined, for all* $i, j \in \mathcal{N}^{\pm}$, *by*

$$w_{\mathrm{A}}(i,j) = \begin{cases} w(i,j), & \text{if } (i,j) \text{ is strongly atomic in } G; \\ +\infty, & \text{otherwise.} \end{cases}$$

The implementation of step 5 of the algorithm, i.e., the strong reduction of the zero-equivalence subgraph, is performed by reducing each zero-equivalence class in isolation. Once again, we exploit the total ordering defined on $\mathcal{N}^{\pm}$.

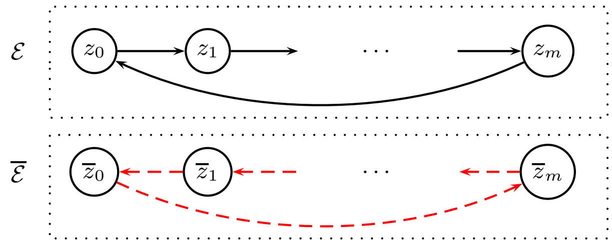

Fig. 1. Strong reduction for non-singular zero-equivalence classes

The strong reduction for a positive non-singular zero-equivalence class $\mathcal{E}$ follows that of [24]: it creates a single zero-cycle connecting all nodes in $\mathcal{E}$ following their total ordering, where the weights of the component arcs are as in the strong closure of the graph. By the coherence assumption, the nodes in the corresponding negative zero-equivalence class $\overline{\mathcal{E}}$ are automatically connected in the opposite order. Figure 1 shows the arcs in the strong reduction of both $\mathcal{E}$ and $\overline{\mathcal{E}}$, where $\mathcal{E} = \{z_0, \ldots, z_m\}$ is the positive class and where $z_0 < \cdots < z_m$. The strong reduction for a singular zero-equivalence class $\mathcal{E}$ is similar except that there is now a single zero-cycle connecting all the positive and negative nodes in $\mathcal{E}$. Figure 2 shows the strong reduction for the singular zero-equivalence class $\mathcal{E} = \{z_0, \overline{z}_0, \ldots, z_m, \overline{z}_m\}$, where $z_0 < \overline{z}_0 < \cdots < z_m < \overline{z}_m$. In both Figures 1 and 2, the dashed arcs are those that can be obtained from the non-dashed ones by application of the coherence assumption.

The following definition formalizes the above observations.

Definition 10. (Zero-equivalence reduction.) *Let* $G = (\mathcal{N}^{\pm}, w)$ *be a strongly closed octagonal graph and let* w' *be the weight function defined, for all* $i, j \in \mathcal{N}^{\pm}$, *as follows: if* $i, j \in \mathcal{E}$ *for some positive zero-equivalence class* $\mathcal{E}$ *of* G *and*

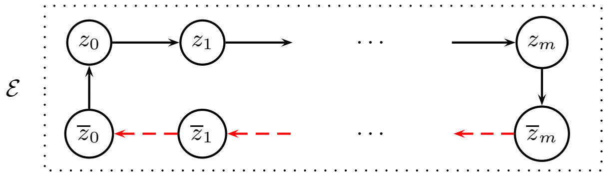

Fig. 2. Strong reduction for the singular zero-equivalence class

– *if* $\mathcal{E} = \{z_0, \dots, z_m\}$ *is non-singular, assuming* $z_0 < \dots < z_m$*,*

$$w'(i,j) := \begin{cases} w(i,j), & \text{if } i = z_{h-1},\ j = z_h, \text{ for some } h = 1 \dots, m; \\ w(i,j), & \text{if } i = z_m,\ j = z_0 \text{ and } m > 0; \\ +\infty, & \text{otherwise;} \end{cases}$$

– *if* $\mathcal{E} = \{z_0, \overline{z}_0, \dots, z_m, \overline{z}_m\}$ *is singular, assuming* $z_0 < \overline{z}_0 < \dots < z_m < \overline{z}_m$*,*

$$w'(i,j) := \begin{cases} w(i,j), & \text{if } i = z_{h-1},\ j = z_h, \text{ for some } h = 1 \dots, m; \\ w(i,j), & \text{if } i = \overline{z}_0,\ j = z_0 \text{ or } i = z_m,\ j = \overline{z}_m; \\ +\infty, & \text{otherwise;} \end{cases}$$

and $w'(i,j) := +\infty$*, otherwise. Then, the* zero-equivalence reduction *for* G *is the octagonal graph* $Z = (\mathcal{N}^{\pm}, w_{\mathrm{Z}})$*, where, for each* $i, j \in \mathcal{N}^{\pm}$*,*

$$w_{\mathrm{Z}}(i,j) := \min\bigl\{w'(i,j), w'(\overline{\jmath}, \overline{\imath})\bigr\}.$$

The final step 6 of the strong reduction algorithm is implemented by computing the greatest lower bound $A \sqcap Z$, where A is the strongly atomic subgraph of L and Z is the zero-equivalent reduction of E, as obtained at steps 4 and 5 of the algorithm.

Theorem 1. *Given an octagonal graph* G*, the strong reduction algorithm computes a strong reduction for* G*.*

If n is the cardinality of the original set $\mathcal{V}$ of variables, then steps 1 and 4 of the algorithm have worst-case complexity in $O(n^3)$, while all the others steps are in $O(n^2)$. Thus, the overall procedure has cubic complexity. As was the case for the reduction procedure of [24], once the ordering of variables is fixed, the strong reduction algorithm returns a canonical form for octagonal graphs.

5.2 An Improved Strong Closure Algorithm

The formal proof of Theorem 1 led to a new result regarding the strong closure operator for octagonal graphs. The strong closure algorithm formalized in [26, 30] performs n local propagation steps: in each step, a rather involved variant of the constraint propagation in the Floyd-Warshall algorithm is followed by another constraint propagation corresponding to the new inference rule (4). A finely tuned implementation of this algorithm [28] performs $20n^3 + 24n^2$ coefficient operations (additions and comparisons), where n is the dimension of the vector space. It turns out that the interleaving of the two kinds of propagation steps is not needed: the same final result can be obtained by the application of the classical Floyd-Warshall closure algorithm followed by a *single* local propagation step using the constraint inference rule (4).

Theorem 2. *Let $G^{\mathrm{c}} = (\mathcal{N}^{\pm}, w^{\mathrm{c}})$ be a closed octagonal graph. Consider the graph $G^{\mathrm{S}} = (\mathcal{N}^{\pm}, w^{\mathrm{S}})$, where w^{S} is defined, for each $i, j \in \mathcal{N}^{\pm}$, by*

$$w^{\mathrm{S}}(i,j) := \min\bigl\{w^{\mathrm{c}}(i,j), w^{\mathrm{c}}(i,\bar{\imath})/2 + w^{\mathrm{c}}(\bar{\jmath},j)/2\bigr\}.$$

Then $G^{\mathrm{S}} = \mathrm{Closure}(G^{\mathrm{c}})$.

By coupling the above optimization with the classical Floyd-Warshall algorithm, we obtain a much simpler implementation performing $16n^3 + 4n^2 + 4n$ coefficient operations: the saving is always above 20% and it is above 30% for $n \leq 8$.

5.3 A Semantic Widening for Octagonal Shapes

A correct implementation of the standard widening on octagonal shapes is obtained by computing any strong reduction of the octagonal graph representing the first argument. As in the case of BDSs, for maximum precision the strongly closed representation for the second argument should be computed. Even better, by adopting the following minor variant, we obtain a "truly semantic" widening operator for the domain of octagonal shapes.

Definition 11. (Widening octagonal shapes.) *Let $S_1, S_2 \in \wp(\mathbb{R}^n)$, where $\varnothing \neq S_1 \subseteq S_2$, be two octagonal shapes represented by the strongly reduced octagonal graph G_1 and the strongly closed octagonal graph G_2, respectively. Let also $S \in \wp(\mathbb{R}^n)$ be the octagonal shape represented by the octagonal graph $G_1 \nabla G_2$. Let $\dim(T)$ denote the affine dimension of shape T. Then we define*

$$S_1 \nabla S_2 := \begin{cases} S_2, & \textit{if } \dim(S_1) < \dim(S_2); \\ S, & \textit{otherwise.} \end{cases}$$

By refraining from applying the graph-based widening when the affine dimension of the geometric shapes is increasing, the operator becomes independent from the specific strongly reduced form computed, i.e., from the total ordering defined on the nodes of the graphs. Also note that the test $\dim(S_1) < \dim(S_2)$ can be efficiently decided by checking whether the nodes of the two octagonal graphs are partitioned into different collections of zero-equivalence classes.

Theorem 3. *The operator '∇' of* Definition 11 *is a proper widening on the domain of octagonal shapes. Let '∇_{s}' be the standard widening on the domain of convex polyhedra, as defined in* [22]*. Then, for all octagonal shapes $S_1, S_2 \in \mathbb{R}^n$ such that $\varnothing \neq S_1 \subseteq S_2$, we have $S_1 \nabla S_2 \subseteq S_1 \nabla_{\mathrm{s}} S_2$.*

The definition of a semantic widening for the domain of BDSs is obtained by simply replacing, in Definition 11, the strongly reduced and strongly closed octagonal graph representations with the reduced and closed BDG representations, respectively. Then a result similar to Theorem 3 holds for BDSs.

6 Conclusion

By considering the semantic abstract domains of geometric shapes, instead of their syntactic representations in terms of constraint networks, we have shown how proper widening operators can be derived for several weakly-relational numeric abstractions. For what concerns the efficient representation of octagonal shapes by means of octagonal graphs, we have specified and proved correct a strong reduction procedure, as well as a more efficient strong closure procedure.

It is worth stressing that both the syntactic and the semantic abstract domains are well defined and may be safely adopted for the implementation of a static analysis application. Nonetheless, it can be argued that using a semantic abstract domain provides several advantages, as already pointed out in [25, Section 5] where the domain of BDGs is compared to the domain of closed BDGs.[4] For instance, it is noted that the domain of closed BDGs allows for the specification of a nicer, injective meaning function; also, the least upper bound operator on BDGs is not the most precise approximation of the union of two geometric shapes. In summary, the discussion in [25, Section 5] makes clear that the solution to the divergence problem for the abstract iteration sequence was the one and only motivation for adopting a syntactic domain.

One disadvantage of syntactic abstract domains concerns the user-level interfaces of the corresponding software implementations. Namely, the user of a syntactic abstract domain (e.g., the developer of a specific static analysis application using this domain) has to be aware of many details that, in principle, should be hidden by the implementation. As an example, consider the shortest-path closure and reduction procedures for BDGs, which the user might rightfully see as semantics-preserving operations. As a matter of fact, for the syntactic domain of BDGs, these are not semantics-preserving: their application affects both the precision and the convergence of the abstract iteration. In such a situation, the documentation of the abstract domain software needs to include several warnings about the correct usage of these operators, so as to avoid possible pitfalls. In contrast, when adopting the semantic domain of BDSs, both the closure and reduction operators may be excluded from the public interface while the implementation can apply them where and when needed or appropriate. Such an approach is systematically pursued in the implementation of the *Parma Polyhedra Library* [10] (PPL, `http://www.cs.unipr.it/ppl`), free software distributed under the GNU General Public License; future releases of the library will support computations on the domains of BDSs and octagonal shapes.

Another potential drawback of the adoption of a syntactic abstract domain can be found in the application of domain refinement operators. As an example,

[4] Similar observations, tailored to the case of octagons, are also in [26, Section VII].

consider the application of the *finite powerset operator* [9] to the domains of BDGs and BDSs, so as to obtain two abstract domains that are able to represent finite disjunctions of the corresponding abstract elements. In both cases, by providing the widenings on BDGs and BDSs with appropriate finite convergence certificates [9], it will be possible to lift them to corresponding widenings on the powerset domains. However, when upgrading the syntactic domain, avoidable redundancies will be incurred, since different disjuncts inside a domain element may represent the same geometric shape; furthermore, these "duplicates" cannot be systematically removed, since by doing so we could change the value of the finite convergence certificate of the powerset element, possibly breaking the convergence guarantee of the lifted widening.

The shortest-path reduction algorithm of [24] has also been considered in the PhD thesis of A. Miné [30] as a tool for the computation of *hollow* (i.e., sparse) representations for BDGs, as originally proposed in [24], so as to obtain memory space savings. The author appears not to identify the positive interaction between reduction and widening and, as a consequence, he conjectures that the computation of hollow representations could compromise the convergence of the abstract iteration sequence (see [30, Section 3.8.2]). An adaptation of the reduction algorithm for the case of octagonal graphs is defined in [30, Section 4.5.2]: this differs from the one proposed in Section 5.1 and may fail to obtain a strongly reduced graph in the sense of Definition 6.

The theoretical results concerning weighted directed graphs hold when the data type adopted for the representation of weights allows for exact computations. If a floating-point data type is considered, then most of these results will be broken due to rounding errors, so that the implementation of a truly semantic abstract domain will not be possible. Nonetheless, the (approximate) reduction operators allow for the removal of most of the syntactic redundancies.

References

1. A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.
2. J. F. Allen and H. A. Kautz. A model of naive temporal reasoning. In *Formal Theories of the Commonsense World*, pp. 251–268. Ablex, Norwood, NJ, 1985.
3. R. Bagnara. *Data-Flow Analysis for Constraint Logic-Based Languages.* PhD thesis, Dipartimento di Informatica, Università di Pisa, Italy, 1997.
4. R. Bagnara, R. Giacobazzi, and G. Levi. Static analysis of CLP programs over numeric domains. In *Proc. WSA 1992*, vol. 81–82 of *Bigre*, pp. 43–50, Bordeaux.
5. R. Bagnara, R. Giacobazzi, and G. Levi. An application of constraint propagation to data-flow analysis. In *Proc. CAIA 1993*, pp. 270–276, Orlando, FL.
6. R. Bagnara, P. M. Hill, E. Mazzi, and E. Zaffanella. Widening operators for weakly-relational numeric abstractions. Quaderno 399, Dipartimento di Matematica, Univ. di Parma, Italy, 2005. Available at `http://www.cs.unipr.it/Publications/`.
7. R. Bagnara, P. M. Hill, E. Ricci, and E. Zaffanella. Precise widening operators for convex polyhedra. In *Proc. SAS 2003*, vol. 2694 of *LNCS*, pp. 337–354, San Diego.
8. R. Bagnara, P. M. Hill, E. Ricci, and E. Zaffanella. Precise widening operators for convex polyhedra. *Science of Computer Programming*, 2005. To appear.

9. R. Bagnara, P. M. Hill, and E. Zaffanella. Widening operators for powerset domains. In *Proc. VMCAI 2004*, vol. 2937 of *LNCS*, pp. 135–148, Venice, Italy.
10. R. Bagnara, P. M. Hill, and E. Zaffanella. *The Parma Polyhedra Library User's Manual*. Department of Mathematics, University of Parma, release 0.7, 2004.
11. V. Balasundaram and K. Kennedy. A technique for summarizing data access and its use in parallelism enhancing transformations. In *Proc. PLDI 1989*, vol. 24(7) of *ACM SIGPLAN Notices*, pp. 41–53, Portland, OR.
12. R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.
13. G. Birkhoff. *Lattice Theory*. American Mathematical Society, 3rd edition, 1967.
14. B. Blanchet, P. Cousot, R. Cousot, J. Feret *et al.*, A static analyzer for large safety-critical software. In *Proc. PLDI 2003*, pp. 196–207, San Diego, CA.
15. R. Clarisó and J. Cortadella. The octahedron abstract domain. In *Proc. SAS 2004*, vol. 3148 of *LNCS*, pp. 312–327, Verona, Italy.
16. P. Cousot and R. Cousot. Static determination of dynamic properties of programs. In *Proc. ISOP 1976*, pp. 106–130, Paris, France.
17. P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proc. POPL 1977*, pp. 238–252, New York.
18. P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In *Proc. POPL 1979*, pp. 269–282, New York.
19. P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Proc. POPL 1978*, pp. 84–96, Tucson, AR.
20. E. Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32(3):281–331, 1987.
21. D. L. Dill. Timing assumptions and verification of finite-state concurrent systems. In *Proc. AVMFSS 1989*, vol. 407 of *LNCS*, pp. 197–212, Grenoble, France.
22. N. Halbwachs. *Détermination Automatique de Relations Linéaires Vérifiées par les Variables d'un Programme*. PhD thesis, Université de Grenoble, France, 1979.
23. N. Halbwachs, Y.-E. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. *Form. Method Syst. Des.*, 11(2):157–185, 1997.
24. K. Larsen, F. Larsson, P. Pettersson, and W. Yi. Efficient verification of real-time systems: Compact data structure and state-space reduction. In *Proc. RTSS 1997*, pp. 14–24, San Francisco, CA.
25. A. Miné. A new numerical abstract domain based on difference-bound matrices. In *Proc. PADO 2001*, vol. 2053 of *LNCS*, pp. 155–172, Aarhus, Denmark.
26. A. Miné. The octagon abstract domain. In *Proc. WCRE'01*, pp. 310–319, Stuttgart.
27. A. Miné. A few graph-based relational numerical abstract domains. In *Proc. SAS 2002*, vol. 2477 of *LNCS*, pp. 117–132, Madrid, Spain.
28. A. Miné. *The Octagon Abstract Domain Library*. École Normale Supérieure, Paris, France, release 0.9.6, 2002. Available at `http://www.di.ens.fr/~mine/oct/`.
29. A. Miné. Relational abstract domains for the detection of floating-point run-time errors. In *Proc. ESOP 2004*, vol. 2986 of *LNCS*, pp. 3–17, Barcelona, Spain.
30. A. Miné. *Weakly Relational Numerical Abstract Domains*. PhD thesis, École Polytechnique, Paris, France, 2005.
31. S. Sankaranarayanan, H. Sipma, and Z. Manna. Scalable analysis of linear systems using mathematical programming. In *Proc. VMCAI 2005*, pp. 25–41, Paris, France.
32. R. Shaham, E. K. Kolodner, and S. Sagiv. Automatic removal of array memory leaks in Java. In *Proc. CC 2000*, vol. 1781 of *LNCS*, pp. 50–66, Berlin, Germany.
33. A. Simon, A. King, and J. M. Howe. Two variables per linear inequality as an abstract domain. In *Proc. LOPSTR 2002*, vol. 2664 of *LNCS*, pp. 71–89, Madrid.