Residuation for Bipolar Preferences in Soft Constraints

Fabio Gadducci[a], Francesco Santini[b,*]

[a]*Dipartimento di Informatica, Università di Pisa, Italy*
[b]*Dipartimento di Matematica e Informatica, Università di Perugia, Italy*

Abstract

Soft constraint formalisms are an abstract representation of Constraint Satisfaction Problems (CSPs): the set of preferences is now parametric, often forming (a variety of) an absorptive semiring. However, the latter is suitable only for negative preferences, i.e., such that the combination of constraints worsens the quality of the solution. This work comments on related work and exploits residuated semirings in order to lift the Local Consistency heuristics that hold for classical CSPs. As a result, we merge and generalise existent formalisms for modelling soft CSPs with bipolar (positive and negative) preferences.

Keywords: Soft Constraints, Bipolar Preferences, Local Consistency.

1. Introduction

Soft constraints refer to a family of formalisms adopted for modelling and optimising problems of a combinatorial nature. They take the form of a set of variable assignments, each one of them equipped with a value denoting the degree of preference for that assignment as part of the solution to the problem. A large body of work has been devoted to the study of algebraic structures for sets of preferences guaranteeing that those techniques can be generalised. The key idea is to consider operators such that the application of algorithms sums up to the manipulation of the values associated to a given set of assignments.

Soft technology focussed mostly on structures modelling negative preferences, i.e., such that the combination of preferences worsens the solution. This property underlies the earlier proposals (see e.g., [1, 2] and [3, Ch. 9.3]), and it is the base for various extensions, such as those with partial inverses [4, 5, 6].

However, the literature on CSPs has recognised the need of *positive* preferences in a *bipolar* setting, see e.g., [7].Consider a scenario where a manufacturer needs to produce different leather products from the same skin. Each product unit has a cost in terms of power consumption and leather. Scheduling the cutting in different ways may actually decrease the overall cost, e.g., by reducing wasted skin. As noted in [8, §8], such situations cannot always be dealt with by manipulating the problem to end up with prices above zero, that is, in other terms, by transforming it into a scenario with negative preferences only.

*Corresponding author
Email addresses: fabio.gadducci@di.unipi.it (Fabio Gadducci), francesco.santini@dmi.unipg.it (Francesco Santini)

The aim of this work is to *i)* review the two existing proposals in the literature (§2.2), *ii)* join them in a new framework through an operator that removes preference values, and *iii)* recover the (soft) local-consistency algorithms in the frame of bipolar preferences. Our starting point (§2) is the bipolar semiring introduced in [9]: we add a partial inverse operator along the lines of what has been done for negative preferences in [5], obtaining a residuated monoid enriched over a join semi-lattice. Our soft constraints are based on such structures (§3). The inverse operator helps us to recover the aforementioned heuristics and some upper bounds on approximated problems (§4). Residuated monoids have a long tradition in logic [10]: they allow for a parametric framework where it is possible to use different preference systems, each represented by a different monoid instance. Finally, §5 wraps up the paper with final considerations.

2. Residuation for Bipolar Preferences

This section introduces some notions concerning monoids enriched over semi-lattices. They allow for recasting the standard presentation of the soft constraints paradigm, and for introducing a new approach to bipolar preferences. Finally, in §2.3 we provide a comparison with related work.

2.1. Residuated Monoids

The first step is to define an algebraic structure for modelling preferences.

Definition 2.1 (orders)**.** *A partial order (PO) is a pair $\langle A, \leq \rangle$ such that A is a set and $\leq \subseteq A \times A$ is a reflexive, transitive, and anti-symmetric relation. A join semi-lattice (JSL) is a PO such that any finite subset of A has a least upper bound (LUB).*

The LUB of a (possibly infinite) subset $X \subseteq A$ is denoted $\bigvee X$, and it is clearly unique. By definition $\bigvee \emptyset$ is the bottom of the PO, and we denote it as $\bot$.

Definition 2.2 (monoids)**.** *A (commutative) monoid is a triple $\langle A, \otimes, \mathbf{1} \rangle$ such that $\otimes : A \times A \to A$ is a commutative and associative function and*

- $\forall a \in A. a \otimes \mathbf{1} = a$*, where* $\mathbf{1} \in A$ *is the* identity *element.*

We often use an infix notation, as $a \otimes b$ for $\otimes(a, b)$.

Definition 2.3 (residuation)**.** *A residuated monoid (ReM) is a 5-tuple $\langle A, \leq, \otimes, \oplusminus, \mathbf{1} \rangle$ such that $\langle A, \leq \rangle$ is a PO, $\langle A, \otimes, \mathbf{1} \rangle$ is a monoid, $\oplusminus : A \times A \to A$ is a function and*

- $\forall a, b, c \in A. b \otimes c \leq a \iff c \leq a \oplusminus b$*.*

A residuated JSL (ReSL) is an ReM such that the underlying PO is a JSL.

The lemma below states that residuation conveys the meaning of division.

Lemma 2.1. *Let $\langle A, \leq, \otimes, \oplusminus, \mathbf{1} \rangle$ be an ReM. Then*

- $\forall a, b \in A. a \oplusminus b = \bigvee \{c \mid b \otimes c \leq a\}$*;*
- $\forall a, b, c \in A. a \leq b \implies (a \oplusminus c \leq b \oplusminus c \,\&\, c \otimes a \leq c \otimes b)$*.*

Proof. The first item is straightforward, since by definition $a \oplus b$ is an upper bound of $\{c \mid b \otimes c \leq a\}$, and again by definition $b \otimes (a \oplus b) \leq a$. The latter property ensures the monotonicity of $\oplus$ (on the first argument), since by definition $a \oplus c \leq b \oplus c$ iff $c \otimes (a \oplus c) \leq b$. As for the monotonicity of $\otimes$, it suffices to note that by definition $a \leq (b \otimes a) \oplus b$ and also by definition $c \otimes a \leq c \otimes b$ iff $a \leq (c \otimes b) \oplus c$. □

Note that by commutativity, $\otimes$ is actually monotone on both arguments. Moreover, Lemma 2.2 states that in fact residuation implies distributivity.

Lemma 2.2. *Let $\langle A, \leq, \otimes, \oplus, \mathbf{1} \rangle$ be an ReSL and $X \subseteq A$ a finite set. Then*

- $\forall a \in A. a \otimes \bigvee X = \bigvee \{a \otimes x \mid x \in X\}$.

Proof. Should X be empty, the proof is immediate, since by definition $\bot \leq \bot \oplus a$.

$$\forall y \in X. a \otimes y \leq \bigvee \{a \otimes x \mid x \in X\} \implies \forall y \in X. y \leq (\bigvee \{a \otimes x \mid x \in X\}) \oplus a \implies$$

$$\implies \bigvee X \leq (\bigvee \{a \otimes x \mid x \in X\}) \oplus a \implies a \otimes \bigvee X \leq \bigvee \{a \otimes x \mid x \in X\}.$$

$$\forall x \in X. x \leq \bigvee X \implies \forall x \in X. x \leq (a \otimes \bigvee X) \oplus a \implies$$

$$\implies \forall x \in X. a \otimes x \leq a \otimes \bigvee X \implies \bigvee \{a \otimes x \mid x \in X\} \leq a \otimes \bigvee X.$$

□

The proof also works for infinite sets, as long as the necessary LUBs exist.

Remark 2.1. *It is now easy to show that ReSLs are tropical semirings, i.e., semirings with a sum operator $a \oplus b = \bigvee \{a, b\}$ that is idempotent. If $\mathbf{1}$ is also the top of the JSL, we would end up in what are called absorptive semirings [11] in the algebra literature, which in turn are known as c-semirings in the soft constraint jargon [1] (see e.g. [5] for a brief survey on residuation for such semirings). Indeed, it is precisely the lack of the latter requirement on $\mathbf{1}$ that makes ReSLs suitable for modelling bipolar preferences: combined with monotonicity, imposing $\mathbf{1}$ to be the top means that preferences are negative, that is, $\forall a, b \in A. a \otimes b \leq a$.*

Some derived properties are $\forall a \in A. \mathbf{1} \leq a \oplus a$ and $\forall a, b \in A. b \otimes (a \oplus b) \leq a \leq (b \otimes a) \oplus b$, and as further consequences also $\forall a \in A. a \otimes (a \oplus a) = a$ and $\forall a, b \in A. a < b \implies \mathbf{1} \not\leq a \oplus b$ (for $a < b$ meaning $a \leq b$ and $a \neq b$).

The latter property suggests the following definition.

Definition 2.4 (localisation / invertibility)**.** *An ReM $\langle A, \leq, \otimes, \oplus, \mathbf{1} \rangle$ is*

- localised *if* $\forall a, b \in A. a \leq b \implies a \oplus b \leq \mathbf{1}$*;*
- invertible *if* $\forall a, b \in A. a \leq b \implies b \otimes (a \oplus b) = a$*.*

Remark 2.2. *Well-known structures used for soft constraints are the* Fuzzy *($\langle [0, 1], \leq, \min, 0, 1 \rangle$),* Probabilistic *($\langle [0, 1], \leq, \times, 0, 1 \rangle$), and* Tropical *($\langle \mathbb{R}^+ \cup \{+\infty\}, \geq, +, +\infty, 0 \rangle$) semirings, for $\geq$ the inverse of the standard order (thus $+\infty$ is the bottom and 0 the top of the JSL, respectively). In all these cases the underlying monoids are invertible and localised, and the $\oplus$ operator can be used to (partially) relax constraints [5].*

2.2. Another Look at Bipolar Proposals

We are aware of two proposals for modelling bipolar preferences using soft constraint technology, by Bistarelli *et alii* [8, 12] and by Fargier and Wilson [9].

2.2.1. Bipolar semirings

The bipolar valuation structures proposed by Fargier and Wilson are the main inspiration behind our formalism. Indeed, ReSLs appear as a streamlined extension of their bipolar semirings [9, §5], equipped with a residuation operator, along the line of what has been done in [5] for negative preferences.

Definition 2.5. *A bipolar semiring is a 6-tuple* $\langle A, \leq, \otimes, \vee, \mathbf{1}, \bot \rangle$ *such that* $\langle A, \leq \rangle$ *is a PO,* $\langle A, \otimes, \mathbf{1} \rangle$ *is a monoid,* $\langle A, \vee, \bot \rangle$ *is an idempotent monoid and*

- $\forall a \in A. \bot \leq a$;
- $\forall a, b, c \in A. a \otimes (b \vee c) = (a \otimes b) \vee (a \otimes c)$;
- $\forall a, b, c \in A. a \leq b \implies (a \vee c \leq b \vee c \,\&\, c \otimes a \leq c \otimes b)$.

The definition implies that $a \vee b$ is an upper bound of a and b: our proposal simply considers $\vee$ as the LUB of two elements, which is the standard interpretation in soft problems. Note however that we could have started directly from the definition above and just added residuation, since all the derived properties would still hold, such as e.g. Lemma 2.2. Hence, asking for residuation is a stronger requirement, yet resulting in a streamlined algebraic presentation.

2.3. Bipolar preference structures

Bistarelli *et alii* consider an absorptive semiring for negative preferences, and a dual structure for positive ones, such that the monoid identity is the top of the former and the bottom of the latter. The two structures might be unrelated, and while an ordering encompassing both kinds of preferences is given, the composition of a positive and a negative value might be not well-defined. Adopting our notation, we can rephrase that definition in the following form.

Definition 2.6. *A bipolar preference structure is a 4-tuple* $\langle A, \leq, \otimes, \mathbf{1} \rangle$ *such that* $A = A^+ \uplus A^-$ *for* $A^+ = \{\mathbf{1} \leq a \mid a \in A\}$ *and* $A^- = \{a \leq \mathbf{1} \mid a \in A\}$, $\langle A^-, \leq \rangle$ *and* $\langle A^+, \geq \rangle$ *are JSLs,* $\otimes : A \times A \to A$ *is a commutative function,* $\langle A^+, \otimes, \mathbf{1} \rangle$ *and* $\langle A^-, \otimes, \mathbf{1} \rangle$ *are monoids, and*

- $\forall a, b, c \in A. a \leq b \implies c \otimes a \leq c \otimes b$;
- $\forall a, b, c \in A^+. a \otimes (b \vee c) = (a \otimes b) \vee (a \otimes c)$, *and the same for* A^-.

These structures are more general than ReSLs. However, unless the positive preferences are carved out of the negative ones by *localisation* [8, §9.1], requiring $\otimes$ to be associative is problematic. Indeed, in [12, §9] a few cases for the positive/negative restriction of $\otimes$ are shown that make associativity fail. Hence, optimisation has to be independently pursued on A^+ and A^-.

Summing up, besides the simplicity of the algebraic presentation, what is noteworthy is that ReSLs allow for easily adapting to the bipolar setting some algorithms (such as the one for local consistency) available for idempotent and invertible absorptive semirings [1, 5], and this is the core of the paper

contribution, as shown in §4, where we extend local consistency algorithms in [13] for weighted CSPs. And even if it is not pursued here, the formalism is robust enough to be closed with respect to standard operators on semirings, such as the Cartesian product and (Hoare) power-domain (the latter needed for multi-criteria problems [14]). Variable elimination algorithms already exist in the literature for partially-ordered sets of preferences, see e.g. [15]. Directly related to our work on bipolar preferences, in [9] the authors present a forward-checking algorithm, and in [12] a local-consistency algorithm for distributive semirings, where $\otimes$ is idempotent: our proposal encompasses both of them.

3. Soft Systems and Problems

This section briefly recalls the key notions of the soft constraint framework, casting them into our proposal using ReSLs (thus generalising [1]). In the following we fix an ReSL $\mathbb{S} = \langle A, \leq, \otimes, \oplus, \mathbf{1} \rangle$.

Definition 3.1 (constraints)**.** *Let V be a (possibly ordered) set of variables and $D = \bigcup_{x \in V} D_x$ a finite domain of interpretation for V. Then, a* constraint *$(V \to D) \to A$ is a function associating a value in A to each assignment $\eta : V \to D$ of the variables.*

Each D_x is the domain of interpretation of the variable $x \in V$, thus $\eta(x) \in D_x$ for each variable assignment. We denote by $\mathcal{C}_{V,D}$ the set of constraints built starting from V and D. The application of a constraint function $c : (V \to D) \to A$ to an assignment $\eta : V \to D$ is denoted $c\eta$. Even if a constraint involves all the variables in V, it may depend on the assignment of a finite subset of them, i.e., its *support*. For instance, a binary constraint c with $supp(c) = \{x, y\}$ is a function $c : (V \to D) \to A$ that depends only on the assignment of variables $\{x, y\} \subseteq V$, meaning that two assignments $\eta_1, \eta_2 : V \to D$ differing only for the image of variables $z \notin \{x, y\}$ coincide (i.e., the evaluation $c\eta_1$ and $c\eta_2$ coincide). The support corresponds to the classical notion of scope of a constraint.

The set of constraints (with finite support) forms an ReSL: the partial order is lifted from $\mathbb{S}$, and given $c_1, c_2 \in \mathcal{C}_{V,D}$ we have that $c_1 \leq c_2$ if $c_1\eta \leq c_2\eta$ for all possible $\eta : V \to D$. Combining two constraints by $\otimes$ means building a new constraint whose support involves at most the variables of the original ones: each tuple of domain values for such variables is associated with the element of $\mathbb{S}$ obtained by multiplying those elements associated by the original constraints to the obvious sub-tuples. Even residuation works similarly: $(c_1 \oplus c_2)\eta = c_1\eta \oplus c_2\eta$. The bottom constraint is the obvious constant function.

Since the domain of interpretation is finite, we define a *projection* operator $\Downarrow$ for each constraint c and variable x as

$$(c \Downarrow_x)\eta = \bigvee_{d \in D_x} c\eta[x := d].$$

Projection decreases the support: $supp(c \Downarrow_x) \subseteq supp(c) \setminus \{x\}$. It is also associative: $c \Downarrow_{\{x_1,\ldots,x_n\}}$ denotes any finite sequence of projections $(\ldots(c \Downarrow_{x_1})\ldots) \Downarrow_{x_n}$.

Definition 3.2 (soft CSPs)**.** *A* soft constraint satisfaction problem *(soft CSP or SCSP) is a triple $\langle V, D, C \rangle$, where $C \subseteq \mathcal{C}_{V,D}$ is a finite set of constraints. The conjunction of all the constraints in a soft CSP $\mathbb{P} = \langle V, D, C \rangle$ is $C_{\mathbb{P}} = \bigotimes_{c \in C} c$.*

In the following, we fix a soft CSP $\mathbb{P} = \langle V, D, C \rangle$ and write $\bigotimes C$ for $\bigotimes_{c \in C} c$.

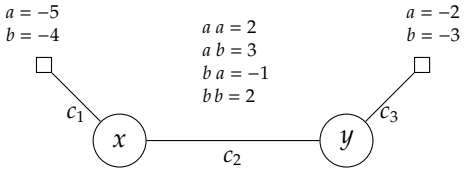

Figure 1: A soft CSP based on the tropical semiring.

Definition 3.3 (solutions)**.** *A solution of* $\mathbb{P}$ *is an assignment* η *such that* $C_{\mathbb{P}}\eta \neq \bot$. *A solution* η *is optimal if for any other assignment* η' *we have* $C_{\mathbb{P}}\eta \not\leq C_{\mathbb{P}}\eta'$.

3.1. A Running Example

A standard example of the domain of preferences in soft constraints is the tropical semiring $\langle \mathbb{R}^+ \cup \{+\infty\}, \geq, +, 0, +\infty \rangle$, for $\geq$ the "greatest or equal" order on integers (thus $+\infty$ is the bottom and 0 the top of the JSL, respectively) and addition as the monoidal operator. In our proposal, $\ominus = \hat{-}$ is defined as $a - b$ if the result is positive, 0 otherwise; this makes W an invertible ReSL. We generalise the structure by considering the ReSL $\mathbb{W} = \langle \mathbb{R} \cup \{-\infty, +\infty\}, \leq, +, -, 0 \rangle$, i.e., considering all the reals, which is also clearly localised and invertible.

Figure 1 shows a soft CSP $\langle \{x, y\}, \{a, b\}, \{c_1, c_2, c_3\} \rangle$ as a graph with values from $\mathbb{W}$. We set this example in the same leather-products manufacturing scenario in §1. Variables and constraints are respectively represented by nodes and by undirected arcs (unary for c_1 and c_3, and binary for c_2), and values are written to the right of each tuple. The domain of the variables contains only the elements a and b, which represent two different leather products, e.g., footwear and gloves. The solution of the soft CSP of Figure 1 associates an element with every domain value of the variables x and y: x is the first product to be scheduled, and y the second one. Therefore, c_1 and c_3 represent a monetary cost for allocating the first and second task to a or b, while c_2 represents the cost spent or saved for a given sequence. For instance, for the tuple $\langle a, a \rangle$ (that is, $x = y = a$), we compute the sum of -5 (the value assigned to $x = a$ in constraint c_1), 2 ($\langle x = a, y = a \rangle$ in c_2) and -2 ($y = a$ in c_3), for a total cost of -5. Hence, producing two pairs of sandals in sequence saves two cost units (stated by c_2).

4. Solving Soft CSP

Following Definition 3.2, we consider a soft CSP as a triple $P = \langle V, D, C \rangle$, where $V = \{x_1, \ldots, x_n\}$ is an ordered set of variables and $C \subseteq C_{V,D}$. Notationwise, we denote as c_Y a constraint whose support is Y, and we assume that for each $Y \subseteq V$ there is at most one such c_Y in C, always including $c_\emptyset$. In fact, we assume that $c_\emptyset \in C$, i.e., a constraint with empty support: we simply add to C the constraint $c_\emptyset = \mathbf{1}$ (the constant function mapping each η to $\mathbf{1}$) without altering the solution. Finally, in the following we focus on a binary soft CSP, i.e., such that no constraint has a support of arity greater than 2: our algorithms might be easily adapted to n-ary soft CSP. However, even if the complexity of finding a solution actually increases, there is no lack of generality, since any soft CSP with n-ary constraints can be converted to an equivalent, binary one [16].

4.1. Soft Local Consistency

The idea behind local-consistency algorithms is to *safely move* costs (i.e., without changing the solution) towards smaller arity constraints. Such algorithms can be used in branch-and-bound based solvers with the purpose to find undominated solutions. Local Consistency is thus a family of increasingly harder properties about a (soft) CSP [3]. The control parameter is the size of the sub-network (i.e., the number of tuple variables) involved. The larger the tuples, the harder the property is. The simplest forms of local consistency are *Node Consistency* (*NC*), which accounts for unary constraints, and *Arc Consistency* (*AC*), which accounts for binary constraints. In general, k-consistency takes into account constraints with $k + 1$ variables in its scope. As we already remarked, for the sake of simplicity we focus on NC and AC.

We are now ready to define our notions of NC and AC in the setting of bipolar preferences, which extend the definitions for weighted CSPs [13].

Definition 4.1 (node/arc consistency)**.** *A variable $x \in V$ is*

- *NC if $c_x \in C \implies \forall d \in D_x . c_\emptyset \otimes c_x[x := d] \neq \bot$;*
- *AC if $\forall y \in V . c_{xy} \in C \implies c_{xy} \Downarrow_y = \mathbf{1}$.*

A problem is NC if each variable is so, AC if each variable is both NC and AC.

Each local-consistency property comes with its enforcing algorithm. In soft CSPs, the effect of such transformations is a *move* of values. For instance, the enforcement of AC moves values from binary to unary/zero-arity constraints. Enforcing algorithms are based on the *Local-consistency Rule*, as defined in [5].

Definition 4.2 (local consistency rule)**.** *Let $c_1, c_2 \in C$ be constraints such that $supp(c_1) \subset supp(c_2)$ and $Z = supp(c_2) \setminus supp(c_1)$. A* Local-consistency Rule*, denoted $CR(c_1, c_2)$, consists of the following two steps:*

- *combine c_1 with the value induced by c_2: $c_1 := c_1 \otimes (c_2 \Downarrow_Z)$;*
- *compensate in c_2 the value combined with c_1: $c_2 := c_2 \oplus (c_2 \Downarrow_Z)$.*

The support of the two constraints may be restricted after the rule application. Also, rule application is usually incremental, and Z is a singleton.

Proposition 4.1 (preserving the solution)**.** *Let c_1, $c_2 \in C$ be constraints such that $supp(c_1) \subset supp(c_2)$ and $\mathbb{Q}$ the soft CSP obtained after the application of the $CR(c_1, c_2)$ rule to $\mathbb{P}$. Then $C_\mathbb{Q} \leq C_\mathbb{P}$ and, if $\mathbb{R}$ is invertible, they coincide.*

The result extends [5, Proposition 10]. The proof is a consequence of the notion of residuation, which ensures that $c_1 \otimes (c_2 \Downarrow_Z) \otimes [c_2 \oplus (c_2 \Downarrow_Z)] \leq c_1 \otimes c_2$.

4.2. Enforcing Algorithms

An algorithm to enforce NC is presented in Algorithm 1. It first applies the *CR* rule in Definition 4.2 to unary constraints $c_z \in C$, accumulating the possible preferences of each c_z to $c_\emptyset$. Then, the algorithm removes from the domain of a variable those assignments that do not lead to a solution (see Definition 3.3).

Both algorithms mimic those for weighted CSPs in [13], the only difference is the use of residuation in the application of the Local-consistency Rule. Thus, we can replicate their considerations concerning soundness and complexity.

As far as time complexity is concerned, for Algorithm 1 it is determined by the loop at lines 5-7, and it is $O(\sum_{z \in V} |D_z|)$: in the worst case, all the d are removed from all the variable domains D_z. Indeed, the first loop (lines 3-4) is only $O(|V|)$. Note that c_z may not belong to C: if it does, then the first loop removes weight from each c_z, while the second loop guarantees that the problem becomes node consistent, since the condition for NC in Definition 4.1 is always satisfied, possibly by removing some elements from the domain of z. It is worth mentioning that, if the ReSL is invertible and the problem is already node consistent, then the application of local consistency at lines 3-4 preserves the property, thus there is no need to execute the loop at lines 5-7.

An algorithm to enforce arc consistency is presented in Algorithm 2. Q represents the set of problem variables to be yet examined (the algorithm ends when Q is empty). At each step (line 4), one variable y is extracted. Then, for all the binary constraints involving y, i.e., $c_{x,y}$, if a unary constraint c_x is in C as well, rule CR (Definition 4.2) moves the preference from $c_{x,y}$ to c_x (in order to later possibly remove domain values from D_x); otherwise, i.e., no unary constraint is defined on x, CR moves the preference directly to $c_\emptyset$. Finally, AC performs a round of NC: all the variables z whose domain of definition is restricted are added to Q, since a further application of AC could be possible to further improve some binary constraints involving z, even if z has been already examined. More precisely, this removal may change the result of projection $c_{xz} \Downarrow_z$, on which the application of CR at line 7 is based.

The soundness of the AC algorithm is guaranteed by the fact that applications of CR do not alter the solutions of the problem, and its termination is ensured by the decrease of the interpretation domain of the variables after each iteration. Let us consider its computational cost. Lines 5-9 are executed $O(|V|^2)$ times (there are $|V|^2$ binary constraints at most), while for line 10 the cost of Algorithm 1 is $O(\sum_{z \in V} |D_z|)$. If we suppose that all variable domains have the same cardinality $|D|$, the complexity is $O(|V| \cdot (|V| + |D|))$. Now, what is relevant is the cost of the top loop at line 3, i.e, how many times a variable can added to the set Q. Now, a variable z is added to Q once in line 2, plus at most $|D_z|$ times at line 10 (each time a value is removed from D_z), for a total of at most $|D_z| + 1$ times. Hence, the total complexity is $O(|V|^2 \cdot |D| \cdot (|V| + |D|))$.

An application of the two algorithms is shown for the soft CSP in Figure 2, with $V = \{x, y\}$ and $D_x = D_y = \{a, b\}$: there we move values from c_{xy} to c_x (from sub-figure *(a)* to *(c)*) and to c_y (from *(c)* to *(d)*). Considered assignments and preferences are highlighted in bold. Since the ReSL is localised, all the positive

Algorithm 1 Node Arc Consistency

```
1: function NC(V, D, C)
2:     P := ∅
3:     for all c_z ∈ C do
4:         CR(c_∅, c_z)
5:     for all c_z ∈ C, d ∈ D_z do
6:         if (c_∅ ⊗ c_z[z := d] = ⊥) then
7:             D_z := D_z \ {d}
8:             P := P ∪ {z}
9:     return P
```

Algorithm 2 Soft Arc Consistency

```
1: procedure AC(V, D, C)
2:     Q := V
3:     while Q ≠ ∅ do
4:         y := pop(Q)
5:         for all c_xy ∈ C do
6:             if c_x ∈ C then
7:                 CR(c_x, c_xy)
8:             else
9:                 CR(c_∅, c_xy)
10:        Q := Q ∪ NC(V, D, C)
```

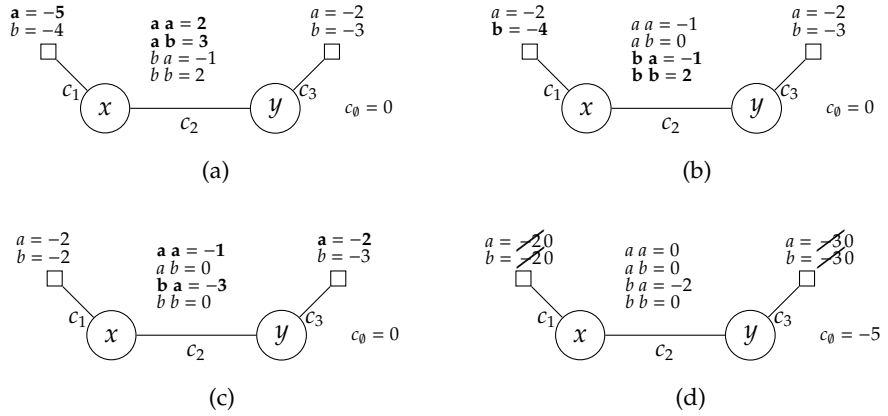

Figure 2: We apply CR (Definition 4.2) to the example in §3.1; it moves all positive preferences and as much negative preferences as possible to unary constraints. Each move is highlighted in bold.

preferences are eventually removed. Also, $c_\emptyset = -5$ (see Figure 2*(d)*) after calling NC at line 10 in Algorithm 2: the problem is NC but not AC (see Definition 4.1), and all the preferences have been removed from unary constraints.

4.3. Preprocessing

The presence of a residuate operator may suggest to introduce a preprocessing phase preceding the application of local consistency: the removal of a fixed value b to each preference, in order to induce a totally positive or totally negative set of constraints. Then, the soft CSP would boil down to solve $\bigotimes(C \oslash b)$ instead of $\bigotimes C$: the outcome of the latter problem would be obtained by multiplying the solution of the former for b^k, with $k = 2^{|C|}$.

However, that procedure introduces an approximation in the solution, since $\forall a, b \in A. b \otimes (a \oslash b) \leq a$, and the result may be unsuitable. Also, the removal of the same value from all constraints is less convenient than applying a different value for each constraint, since for $a \leq b \leq c$ we have $c \otimes (a \oslash c) \leq b \otimes (a \oslash b) \leq a$.

This problem arises even if the ReSL is localised: it must also be invertible in order to ensure that a preprocessing phase produces an adequate soft CSP.

5. Conclusions and Future Work

We have shown how choosing residuated semi-lattices as domains of preferences allows for a streamlined presentation of bipolar preferences as well as for recasting a well-known heuristics for negative preferences: Local Consistency. This is exploited in adapting two consistency algorithms from the weighted CSP literature. Also, should the semi-lattice be localised or invertible, the algorithms used in the semiring-based soft constraint literature could be recovered by a preprocessing phase, albeit with a loss of precision in the former case.

In the future we will adapt other well-known algorithms to the use of the bipolar-preference framework shown in this paper: for instance, the (Soft) *Mini-bucket Elimination* and the (Soft) *Branch-and-bound* algorithms. We intend to use the $\oslash$ operator to compute the difference between exact solutions and their approximation when optimality is difficult (with the Bucket algorithm) or impossible (when the preference structure is not invertible) to be reached.

References

[1] S. Bistarelli, U. Montanari, F. Rossi, Semiring-based constraint satisfaction and optimization, Journal of ACM 44 (2) (1997) 201–236.

[2] T. Schiex, H. Fargier, G. Verfaillie, Valued constraint satisfaction problems: Hard and easy problems, in: IJCAI 1995, Morgan Kaufmann, 1995, pp. 631–639.

[3] F. Rossi, P. van Beek, T. Walsh, Handbook of Constraint Programming, Elsevier, 2006.

[4] M. Cooper, T. Schiex, Arc consistency for soft constraints, Artificial Intelligence 154 (1–2) (2004) 199–227.

[5] S. Bistarelli, F. Gadducci, Enhancing constraints manipulation in semiring-based formalisms, in: G. Brewka, S. Coradeschi, A. Perini, P. Traverso (Eds.), ECAI 2006, Vol. 141 of FAIA, IOS Press, 2006, pp. 63–67.

[6] S. Bova, Local computation schemes with partially ordered preferences, in: C. Sossai, G. Chemello (Eds.), ECSQARU 2009, Vol. 5590 of LNCS, Springer, 2009, pp. 887–898.

[7] D. Dubois, H. Prade, An introduction to bipolar representations of information and preference, Intelligent Systems 23 (8) (2008) 866–877.

[8] S. Bistarelli, M. S. Pini, F. Rossi, K. B. Venable, From soft constraints to bipolar preferences: modelling framework and solving issues, Experimental and Theoretical Artificial Intelligence 22 (2) (2010) 135–158.

[9] H. Fargier, N. Wilson, Algebraic structures for bipolar constraint-based reasoning, in: K. Mellouli (Ed.), ECSQARU 2007, Vol. 4724 of LNCS, Springer, 2007, pp. 623–634.

[10] H. Ono, Substructural logics and residuated lattices – an introduction, Trends in Logic 20 (2003) 177–212.

[11] J. Golan, Semirings and Affine Equations over Them: Theory and Applications, Kluwer, 2003.

[12] S. Bistarelli, M. S. Pini, F. Rossi, K. B. Venable, Uncertainty in bipolar preference problems, Experimental and Theoretical Artificial Intelligence 23 (4) (2011) 545–575.

[13] J. Larrosa, Node and arc consistency in weighted CSP, in: R. Dechter, R. S. Sutton (Eds.), AAAI/IAAI 2002, AAAI/The MIT Press, 2002, pp. 48–53.

[14] S. Bistarelli, F. Gadducci, J. Larrosa, E. Rollon, F. Santini, Local arc consistency for non-invertible semirings, with an application to multi-objective optimization, Expert Systems and Applications 39 (2) (2012) 1708–1717.

[15] H. Fargier, E. Rollon, N. Wilson, Enabling local computation for partially ordered preferences, Constraints 15 (4) (2010) 516–539.

[16] F. Bacchus, P. van Beek, On the conversion between non-binary constraint satisfaction problems, in: J. Mostow, C. Rich (Eds.), AAAI/IAAI 1998, AAAI/The MIT Press, 1998, pp. 310–318.