HAL
open science

A Labelled Semantics for Soft Concurrent Constraint Programming

Fabio Gadducci, Francesco Santini, Luis Pino, Frank Valencia

To cite this version:

Fabio Gadducci, Francesco Santini, Luis Pino, Frank Valencia. A Labelled Semantics for Soft Concurrent Constraint Programming. 17th International Conference on Coordination Languages and Models (COORDINATION), Jun 2015, Grenoble, France. pp.133-149, 10.1007/978-3-319-19282-6_9 . hal-01149227

HAL Id: hal-01149227

https://inria.hal.science/hal-01149227v1

Submitted on 6 May 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Labelled Semantics for Soft Concurrent Constraint Programming*

Fabio Gadducci[1], Francesco Santini[2], Luis F. Pino[3], Frank D. Valencia[4]

[1] Dipartimento di Informatica, Università di Pisa
fabio.gadducci@di.unipi.it
[2] Istituto di Informatica e Telematica, CNR
francesco.santini@iit.cnr.it
[3] Dipartimento di Matematica e Informatica, Università di Cagliari
luis.pino@unica.it
[4] CNRS and LIX, École Polytechnique de Paris
frank.valencia@lix.polytechnique.fr

Abstract. We present a labelled semantics for Soft Concurrent Constraint Programming (SCCP), a language where concurrent agents may synchronize on a shared store by either posting or checking the satisfaction of (soft) constraints. SCCP generalizes the classical formalism by parametrising the constraint system over an order-enriched monoid: the monoid operator is not required to be idempotent, thus adding the same information several times may change the store. The novel operational rules are shown to offer a sound and complete co-inductive technique to prove the original equivalence over the unlabelled semantics.

1 Introduction

Concurrent Constraint Programming (CCP) [21] is a language based on a shared-memory communication pattern: processes may interact by either posting or checking partial information, which is represented as constraints in a global store. CCP belongs to the larger family of process calculi, thus a syntax-driven operational semantics represents the computational steps. For example, the term **tell**(c) is the process that posts c in the store, and the term **ask**(c) $\rightarrow$ P is the process that executes P if c can be derived from the information in the store.

The formalism is parametric with respect to the entailment relation. Under the name of *constraint system*, the information recorded on the store is structured as a partial order (actually, a lattice) $\leq$, where $c \leq d$ means that c can be derived from d. Under a few requirements over such systems, CCP has been provided with (coincident) operational and denotational semantics. More recently, a labelled semantics has also been provided, and the associated weak bisimilarity proved to coincide with the original semantics [1].

* The research has been partially supported by the MIUR PRIN 2010LHT4KM CINA and PRIN 2010XSEMLC "Security Horizons", by the ANR 12IS02001 PACE, and by the Aut. Reg. of Sardinia P.I.A. 2010 "Social Glue".

A key aspect of CCP is the *idempotency* of the operator for composing constraints: adding the same information twice does not change the store. On the contrary, the soft variant of the formalism (Soft CCP, or just SCCP [7]) drops idempotency: constraint systems in SCCP may distinguish the number of occurrences of a piece of information. Dropping idempotency requires a complete reworking of the theory. Although an operational semantics for SCCP has been devised [7], hitherto neither the denotational nor the labelled one has been reintroduced. This is unfortunate since due to its generality, SCCP has been successfully applied as a specification formalism for negotiation of Service Level Agreements [10], or the enforcement of ACL-based access control [8].

The objective of our work is the development of a general theory for the operational as well as the denotational semantics of SCCP, via the introduction of suitable behavioral equivalences. Reaching this objective is technically challenging, since most of the simplicity of CCP is based precisely on the premise that posting an information multiple times is the same as posting it only once.

As a language, SCCP has been used as a specification formalism for agents collaborating via a shared knowledge basis, possibly with temporal features [4]. Thus, on a methodological level, the development of behavioural equivalences for SCCP may result in the improvement on the analysis techniques for agents that need to reason guided by their preferences, more so if their knowledge (e.g. of their environment) is not complete. Indeed, the paper shows that systems specified by SCCP may benefit from the feasible proof and verification methods typically associated with bisimilarity, compared with the classical analysis based on (possibly infinite) sequences of computations. This is true also whenever agents have to coordinate despite the global problem being over-constrained (i.e., admitting no solution), and *simulation* may serve as a powerful mechanism for distilling suitable approximated solutions.

Contribution. The work in [21] establishes a denotational semantics for CCP and an equational theory for infinite agents. More recently, in [1] the authors prove that the axiomatisation is underlying a specific weak bisimilarity among agents, thus providing a clear operational understanding. The key ingredients are a complete lattice as the domain of the store, with least upper bound for constraint combination, and a notion of compactness such that domain equations for the parallel composition of recursive agents would be well-defined. On the contrary, the soft version [7] drops the upper bound for combination in exchange of a more general monoidal operator. Thus, the domain is potentially just a (not necessarily complete) partial order, possibly with finite meets and a residuation operator (a kind of inverse of the monoidal one) in order to account for algorithms concerning constraint propagation. Indeed, the main use of SCCP has been in the generalisation of classical constraint satisfaction problems, hence the lack of investigation about e.g compactness and denotational semantics.

Therefore, in this paper we connect the works on the soft [7] and the classical (also indicated in the literature as "crisp") [21,1] paradigm by investigating a labelled semantics for SCCP. In particular, the results will be a mix of those investigated in the two communities, namely, a monoid whose underlying set

of elements form a complete lattice. We will recast the notion of compactness, and afterwards the SCCP semantics, thus making the work a direct extension of the proposal for the crisp language. We will then introduce a novel labelled semantics for SCCP which will allow us to give a sound and complete technique to prove the equivalence over the unlabelled semantics.

2 A Few Technical Remarks (with some novelty)

This section recalls the main notions we are going to need later on. First of all, we present some basic facts concerning monoids [15] enriched over complete lattices. These are used to recast the standard presentation of the soft constraints paradigm, and to generalise the classical crisp one.

2.1 Lattice-enriched Monoids

Definition 1 (Complete lattices). *A partial order (PO) is a pair $\langle A, \leq \rangle$ such that A is a set of values and $\leq \subseteq A \times A$ is a reflexive, transitive, and anti-symmetric relation. A complete lattice (CL) is a PO such that any subset of A has a least upper bound (LUB).*

We denote as $\bigvee X$ the necessarily unique LUB of a subset $X \subseteq A$, and explicitly $\bot$ and $\top$ if we are considering the empty set and the whole A, respectively: the former is the bottom and the latter is the top of the PO. Obviously, CLs also have the greatest lower bound (GLB) for any subset $Y \subseteq A$, denoted as $\bigwedge Y$.

In the following we fix a CL $\mathbb{C} = \langle A, \leq \rangle$.

Definition 2 (Compact elements). *An element $a \in A$ is compact (or finite) if whenever $a \leq \bigvee Y$ there exists a finite subset $X \subseteq Y$ such that $a \leq \bigvee X$.*

Note that for complete lattices the definition of compactness given above coincides with the one using directed subsets. It will be easier to generalise it, though, to compactness with respect to the monoidal operator (see Def. 6). We let $A^C \subseteq A$ denote the set of compact elements of $\mathbb{C}$. Note that A^C might be trivial. Consider e.g. the CL $\langle [0,1], \geq \rangle$ (the segment of the reals with the inverse of the usual order), used for probabilistic constraints [12]: only the bottom element 1 is compact. As we will see, the situation for the soft paradigm is more nuanced.

Definition 3 (Monoids). *A commutative monoid with identity (IM) is a triple $\langle A, \otimes, \mathbf{1} \rangle$ where $\otimes : A \times A \to A$ is a commutative and associative function and $\forall a \in A. \otimes(a, \mathbf{1}) = a$.*

We will often use an infix notation, such as $a \otimes b$ for $a, b \in A$. The monoidal operator can be defined for any multi-set: it is given for a family of elements $a_i \in A$ indexed over a finite, non-empty set I, and it is denoted by $\bigotimes_{i \in I} a_i$. Whenever for an index set I all the a_i's are different, we write $\bigotimes S$ instead of $\bigotimes_{i \in I} a_i$ for the set $S = \{a_i \mid i \in I\}$. Conventionally, we also denote $\bigotimes \emptyset = \bot$.

We now move our attention to the domain of values we are going to consider.

Definition 4 (CL-enriched IMs). *A CL-enriched IM (CLIM) is a triple* $\mathbb{S} = \langle A, \leq, \otimes \rangle$ *such that* $\langle A, \leq \rangle$ *is a CL,* $\langle A, \otimes, \bot \rangle$ *is an IM, and furthermore the following holds*

(distributivity) $\forall a \in A. \forall X \subseteq A. a \otimes \bigwedge X = \bigwedge \{a \otimes x \mid x \in X\}$

Remark 1. The reader who is familiar with the soft constraint literature may have noticed that we have basically rewritten the standard presentation using a CLIM instead of an absorptive semiring, recently popularized as c-semiring [6], where the $a \oplus b$ operator is replaced by the binary LUB $a \vee b$. Besides what we consider a streamlined presentation, the main advantage in the use of CLIMs is the easiness in defining the LUB of infinite sets and, as a consequence, the notion of $\otimes$-compactness given below. An alternative solution using infinite sums can be found in [14, Section 3], and a possible use is sketched in [5].

Thanks to distributivity, we can show that $\otimes$ is monotone, and since $\bot$ is the identity of the monoid, monotonicity implies that the combination of constraints is increasing, i.e., $\forall a, b \in A. a \leq a \otimes b$ holds. Finally, we recall that by definition $\bigwedge \emptyset = \top$, so that $\forall a \in A. a \otimes \top = \top$ also holds.[5]

In the following, we fix a CLIM $\mathbb{S} = \langle A, \leq, \otimes \rangle$. The next step is to provide a notion of infinite composition. Our definition is from [15] (see also [14, p. 42]).

Definition 5 (Infinite composition). *Let* I *be a (countable) set of indexes. Then, composition* $\bigotimes_{i \in I} a_i$ *is given as* $\bigvee_{J \subseteq I} \bigotimes_{j \in J} a_j$ *for all finite subsets* J.

Should I be finite, the definition gives back the usual multiset composition, since $\otimes$ is monotone and increasing. Indeed, as the infinitary composition is also monotone and increasing, and by construction $\bigotimes A = \bigvee A = \top$ holds. We now provide a notion of compactness with respect to the monoidal operator.

Definition 6 ($\otimes$-compact elements). *An element* $a \in A$ *is* $\otimes$*-compact (or* $\otimes$*-finite) if whenever* $a \leq \bigotimes_{i \in I} a_i$ *then there exists a finite subset* $J \subseteq I$ *such that* $a \leq \bigotimes_{j \in J} a_j$.

We let $A^{\otimes} \subseteq A$ denote the set of $\otimes$-compact elements of $\mathbb{S}$. It is easy to show that a compact element is also $\otimes$-compact, i.e. $A^C \subseteq A^{\otimes}$. Indeed, the latter notion is definitively more flexible. Consider e.g. the CLIM $\langle [0, 1], \geq, \times \rangle$ examined above, which corresponds to the segment of the reals with the inverse of the usual order and multiplication as monoidal product. Since any infinite multiplication tends to 0, then all the elements are $\otimes$-compact, except the top element itself, that is, precisely 0.

Remark 2. It is easy to show that idempotency implies that $\bigotimes$ coincides with LUBs, that is, $\bigotimes S = \bigvee S$ for all subsets $S \subseteq A$. In other words, the whole soft structure collapses to a complete distributive lattice. Indeed, requiring distributivity makes the soft paradigm not fully comparable with the crisp one. We are going to discuss it again in the concluding remarks.

[5] A symmetric choice $\langle A, \otimes, \top \rangle$ with distributivity with respect to $\bigvee$ (and thus $a \otimes \bot = \bot$) is possible: the monoidal operator would be decreasing, so that for example $a \otimes b \leq a$. Indeed, this is the usual order in the semiring-based approach to soft constraints [5].

2.2 Some Operators: Residuation and Cylindrification

We close this section by presenting two operators on CL-enriched IMs.

The first is a simple construction for building a weak inverse of the monoidal operator in CL-enriched monoids, known in the literature as residuation [14,13].

Definition 7 (Residuation). *Let $a, b \in A$. The residuation of a with respect to b is defined as $a \ominus b = \bigwedge\{c \in A \mid a \leq b \otimes c\}$.*

The definition conveys the intuitive meaning of a division operator: indeed, $a \leq b \otimes (a \ominus b)$, thanks to distributivity. Also, $(a \otimes b) \ominus b \leq a$ and $a \ominus (b \otimes c) = (a \ominus b) \ominus c$. Residuation is monotone on the first argument: if $a \leq b$ then $a \ominus c \leq b \ominus c$ and $a \ominus b = \bot$. For more properties of residuation we refer to [3, Tab. 4.1].

Most important for our formalism is the following result on $\otimes$-compactness.

Lemma 1. *Let $a, b \in A$. If a is $\otimes$-compact, so is $a \ominus b$.*

Proof. If $a \ominus b \leq \bigotimes_{i \in I} a_i$, then by monotonicity $a \leq \bigotimes_{i \in I \uplus \{*\}} a_i$ for $a_* = b$. By $\otimes$-compactness of a there exists a finite $J \subseteq I$ such that $a \leq \bigotimes_{j \in J \uplus \{*\}} a_j$, and by the definition of division $a \ominus b \leq \bigotimes_{j \in J} a_j$, hence the result holds. □

Most standard soft instances (boolean, fuzzy, probabilistic, weighted, and so on) are described by CL-enriched monoids and are residuated: see e.g. [5]. For these instances the $\ominus$ operator is used to (partially) remove constraints from the store, and as such is going to be used in Section 4. In fact, in the soft literature it is required a tighter relation of (full) invertibility, also satisfied by all the previous CLIMs instances, stated in our framework by the definition below.

Definition 8. *A CLIM $\mathbb{S}$ is invertible if $b \leq a$ implies $b \otimes (a \ominus b) = a$ for all $a, b \in A^{\otimes}$.*

We now consider two families of operators for modelling the hiding of local variables and the passing of parameters in soft CCP. They can be considered as generalised notions of existential quantifier and diagonal element [21], which are expressed in terms of operators of cylindric algebras [18]. [6]

Definition 9 (Cylindrification). *Let V be a set of variables. A cylindric operator $\exists$ over $\mathbb{S}$ and V is given by a family of monotone, $\otimes$-compactness preserving functions $\exists_x : A \to A$ indexed by elements in V such that for all $a, b \in A$ and $x, y \in V$*

1. $\exists_x a \leq a$;
2. $\exists_x (a \otimes \exists_x b) = \exists_x a \otimes \exists_x b$;
3. $\exists_x \exists_y a = \exists_y \exists_x a$.

Let $a \in A$. The support *of a is the set of variables $sv(a) = \{x \in V \mid \exists_x a \neq a\}$.*

For a finite $X \subseteq V$ we denote by $\exists_X a$ any sequence of function applications. Also, we fix a set of variables V and a cylindric operator $\exists$ over CLIM $\mathbb{S}$ and V.

[6] However, since we consider monoids instead of groups, the set of axiom of diagonal operators is included in the standard one for cylindric algebras.

Definition 10 (Diagonalisation). *A diagonal operator δ for $\exists$ is given by a family of idempotent elements $\delta_{x,y} \in A$ indexed by pairs of elements in V such that $\delta_{x,y} = \delta_{y,x}$ and for all $a \in A$ and $x, y, z \in V$*

1. $\delta_{x,x} = \bot$;
2. *if* $z \notin \{x, y\}$ *then* $\delta_{x,y} = \exists_z(\delta_{x,z} \otimes \delta_{z,y})$;
3. *if* $x \neq y$ *then* $a \leq \delta_{x,y} \otimes \exists_x(a \otimes \delta_{x,y})$.

Axioms 1 and 2 above plus idempotency imply that $\exists_x \delta_{x,y} = \bot$, which in turn implies (again with axiom 2 and idempotency of $\exists$) that $sv(\delta_{x,y}) = \{x, y\}$ for $x \neq y$. Diagonal operators are going to be used for modelling variable substitution and parameter passing. In the following, we fix a diagonal operator δ for $\exists$.

Definition 11 (Substitution). *Let $x, y \in V$ and $a \in A$. The substitution $a[^y/_x]$ is defined as a if $x = y$ and as $\exists_x(\delta_{x,y} \otimes a)$ otherwise.*

We now rephrase some of the laws holding for the crisp case (see [1, p.140]).

Lemma 2. *Let $x, y \in V$ and $a \in A$. Then it holds*

1. $y \notin sv(a)$ *implies* $(a[^y/_x])[^x/_y] = a$;
2. $a[^y/_x] \otimes b[^y/_x] = (a \otimes b)[^y/_x]$;
3. $x \notin sv(a[^y/_x])$.

Proof. Consider e.g. the most difficult item 2. By definition $a[^y/_x] \otimes b[^y/_x] = \exists_x(\delta_{x,y} \otimes a) \otimes \exists_x(\delta_{x,y} \otimes b)$, which in turn coincides with $\exists_x(\delta_{x,y} \otimes a \otimes \exists_x(\delta_{x,y} \otimes b))$ by axiom 2 of $\exists$; by axiom 3 of $\delta_{x,y}$ we have that $(a \otimes b)[^y/_x] = \exists_x(\delta_{x,y} \otimes a \otimes b) \leq \exists_x(\delta_{x,y} \otimes a \otimes \exists_x(\delta_{x,y} \otimes b))$, while the vice versa holds by the monotonicity of $\exists_x$. $\square$

3 Deterministic Soft CCP

We now introduce our language. We fix an invertible CLIM $\mathbb{S} = \langle C, \leq, \otimes \rangle$, which is also cylindric over a set of variables V, denoting by c an element in $C^{\otimes}$.

$$A ::= \ \mathbf{stop} \mid \mathbf{tell}(c) \mid \mathbf{ask}(c) \rightarrow A \mid A \parallel A \mid \exists_x A \mid p(x).$$

Let $\mathcal{A}$ be the set of all agents, which is parametric with respect to a set $\mathcal{P}$ of (unary) procedure declarations $p(x) = A$ such that $fv(A) = \{x\}$.[7]

In Tab. 1 we provide a reduction semantics for SCCP: a pair $\langle \Gamma, \rightarrow \rangle$, for $\Gamma = \mathcal{A} \times C^{\otimes}$ the set of configurations and $\longrightarrow \subseteq \Gamma \times \Gamma$ a family of binary relations indexed over set of variables, i.e., $\longrightarrow = \bigcup_{\Delta \subseteq V} \longrightarrow_{\Delta}$ and $\longrightarrow_{\Delta} \subseteq \Gamma \times \Gamma$.

In **R1** a constraint c is added to the store σ. **R2** checks if c is entailed by σ: if not, the computation is blocked. Rules **R3** and **R4** model the interleaving of two agents in parallel. Rule **R5** replaces a procedure identifier with the associated body, renaming the formal parameter with the actual one: $A[^y/_x]$ stands for the

[7] The set of free variables of an agent is defined in the expected way by structural induction, assuming that $fv(\mathbf{tell}(c)) = sv(c)$ and $fv(\mathbf{ask}(c) \rightarrow A) = sv(c) \cup fv(A)$.

R1	$\dfrac{sv(\sigma) \cup sv(c) \subseteq \Delta}{\langle \mathbf{tell}(c), \sigma \rangle \longrightarrow_\Delta \langle \mathbf{stop}, \sigma \otimes c \rangle}$	**Tell**	**R2**	$\dfrac{c \leq \sigma \wedge sv(\sigma) \cup sv(c) \subseteq \Delta}{\langle \mathbf{ask}(c) \rightarrow A, \sigma \rangle \longrightarrow_\Delta \langle A, \sigma \rangle}$	**Ask**
R3	$\dfrac{\langle A, \sigma \rangle \longrightarrow_\Delta \langle A', \sigma' \rangle \wedge fv(B) \subseteq \Delta}{\langle A \parallel B, \sigma \rangle \longrightarrow_\Delta \langle A' \parallel B, \sigma' \rangle}$	**Par1**	**R4**	$\dfrac{\langle A, \sigma \rangle \longrightarrow_\Delta \langle A', \sigma' \rangle \wedge fv(B) \subseteq \Delta}{\langle B \parallel A, \sigma \rangle \longrightarrow_\Delta \langle B \parallel A', \sigma' \rangle}$	**Par2**
R5	$\dfrac{\{y\} \cup sv(\sigma) \subseteq \Delta \wedge p(x) = A \in \mathcal{P}}{\langle p(y), \sigma \rangle \longrightarrow_\Delta \langle A[^y/_x], \sigma \rangle}$	**Rec**	**R6**	$\dfrac{fv(A) \cup sv(\sigma) \subseteq \Delta \wedge w \notin \Delta}{\langle \exists_x A, \sigma \rangle \longrightarrow_\Delta \langle A[^w/_x], \sigma \rangle}$	**Hide**

Table 1. Reduction semantics for SCCP.

agent obtained by replacing all the occurrences of x with y.[8] Rule **R6** hides the variable x occurring in A. The variable w that replaces x is globally fresh, as ensured by requiring $w \notin \Delta$. The latter is more general than just requiring that $w \notin fv(A) \cup sv(\sigma)$, since $\langle B, \rho \rangle \longrightarrow_\Delta$ implies that $fv(B) \cup sv(\rho) \subseteq \Delta$.[9]

We denote $fv(A) \cup sv(\sigma)$ as $fv(\gamma)$ for a configuration $\gamma = \langle A, \sigma \rangle$, and by $\gamma[^z/_w]$ the component-wise application of substitution $[^z/_w]$. Clearly $\gamma \rightarrow_\Delta \gamma'$ implies $fv(\gamma) \subseteq \Delta$, and we now further provide three lemmata on reduction.

Lemma 3 (Mono). *Let $\langle A, \sigma \rangle \rightarrow_\Delta \langle B, \sigma' \rangle$ be a reduction. Then, $\sigma \leq \sigma'$ and $sv(\sigma') \subseteq \Delta$.*

The proof is straightforward: only rule **R1** can modify the store, and $\sigma \leq \sigma \otimes c$ as well as $sv(\sigma \otimes c) \subseteq sv(\sigma) \cup sv(c)$ hold, since as shown above $fv(\mathbf{tell}(c)) \cup sv(\sigma) \subseteq \Delta$.

Lemma 4 (Operational mono). *Let $\langle A, \sigma \rangle \rightarrow_\Delta \langle B, \sigma' \rangle$ be a reduction and $\rho \in C^\otimes$ such that $sv(\rho) \subseteq \Delta$. Then, there exists a reduction $\langle A, \sigma \otimes \rho \rangle \rightarrow_\Delta \langle B, \sigma' \otimes \rho \rangle$.*

The proof is straightforward, since as before $sv(\sigma \otimes \rho) \subseteq sv(\sigma) \cup sv(\rho)$ and moreover $\sigma, \rho \in C^\otimes$ ensure that $\sigma \otimes \rho \in C^\otimes$.

3.1 Observational Semantics

To define fair computations (Def. 12), we introduce enabled and active agents. Note that any transition is generated by an agent of the shape $\mathbf{tell}(c)$ or $\mathbf{ask}(c) \rightarrow A$ or $p(x)$ or $\exists_x A$ via the application of precisely one instance of one of the axioms **R1**, **R2**, **R5**, and **R6** of Tab. 1. An agent of such shape is *active* in a transition $t = \gamma \rightarrow \gamma'$ if it generates such transition, i.e. if there is a derivation of t where that agent is used in the building axiom. Moreover, an agent is *enabled* in a configuration γ if there is a transition $\gamma \rightarrow \gamma'$ such that the agent is *active* in it.

Definition 12 (Fair computations). *Let $\gamma_0 \rightarrow_{\Delta_1} \gamma_1 \rightarrow_{\Delta_2} \gamma_2 \rightarrow_{\Delta_3} \ldots$ be a (possibly infinite) computation. It is fair if it is increasing (i.e., $\Delta_k \subseteq \Delta_{k+1}$ for any k) and whenever an agent A is enabled in some γ_i then A is active in $\gamma_j \rightarrow_{\Delta_{j+1}} \gamma_{j+1}$ for some $j \geq i$.*

Note that fairness is well given: the format of the rules allows us to always trace the occurrence of an agent along a computation.

[8] With the usual conventions, so that e.g. $(\exists_y A)[^y/_x] = \exists_w((A[^w/_y])[^y/_x])$ for $w \notin sv(A) \cup \{x, y\}$ and $\mathbf{tell}(c)[^y/_x] = \mathbf{tell}(c[^y/_x])$, the latter defined according to Def. 11.

[9] Our rule is reminiscent of (8) in [21, p. 342].

Definition 13 (Observables). *Let $\xi = \gamma_0 \rightarrow_{\Delta_1} \gamma_1 \rightarrow_{\Delta_2} \dots$ be a (possibly infinite) computation with $\gamma_i = \langle A_i, \sigma_i \rangle$. Result($\xi$) is $\bigvee_i (\exists_{X_i} \sigma_i)$, for $X_i = (fv(\gamma_i)) \setminus (fv(\gamma_0))$.*

Similarly to crisp programming [21], if a finite computation is fair then it is deadlocked and its result coincides with the store of the last configuration.

Proposition 1 (Confluence). *Let γ be a configuration and ξ_1, ξ_2 two (possibly infinite) computations of γ. If ξ_1 and ξ_2 are fair, then Result(ξ_1) = Result(ξ_2).*

The proposition is an immediate consequence of the lemma below.

Lemma 5. *Let $\gamma \rightarrow_{\Delta_i} \gamma_i$ be reductions for $i = 1, 2$. Then one of the following holds*

1. *$\xi_i = \gamma \rightarrow_{\Delta_i} \gamma_i[^z/_{w_i}]$ and $\gamma_1[^z/_{w_1}] = \gamma_2[^z/_{w_2}]$ for $w_i \notin \Delta_i$ and z fresh;*
2. *$\xi_i = \gamma \rightarrow_{\Delta_i} \gamma_i[^{z_i}/_{w_i}] \rightarrow_{\Delta_1 \cup \Delta_2 \cup \{z_i\}} \gamma_3$ for $w_i \notin (\Delta_1 \cap \Delta_2) \cup fv(\gamma_3)$ and z_i's fresh.*

In both cases, Result(ξ_1) = Result(ξ_2).

Proof. First of all, note that the calculus is deterministic except for the parallel and the hiding operators. Consider the latter. The problem may arise if different fresh variables are chosen, let us say w_1 and w_2. However, $\gamma_1[^z/_{w_1}] = \gamma_2[^z/_{w_2}]$ by replacing the new variables with a globally fresh one, as in item 1.

So, let us assume that the two reductions occur on the opposite sides of a parallel operator. Also, let $\gamma \rightarrow_{\Delta_1} \gamma_1$ replace a hiding operator with a variable w_1 (hence we have $w_1 \notin \Delta_1$). If $w_1 \in fv(\gamma_2)$, since $w_1 \notin \gamma$ and the only reduction enlarging the set of free variables is the replacement of a hiding operator, also $\gamma \rightarrow_{\Delta_2} \gamma_2$ must replace a hiding operator with variable w_1, and thus it suffices to replace w_1 with fresh variables z_1 and z_2 in the two reductions, in order for item 2 to be verified. If $w_1 \notin fv(\gamma_2)$, then ξ_2 is obtained by replacing in γ_2 the hiding operator with z_1 instead of w_1. As for obtaining ξ_1, the only problematic case is if $\gamma \rightarrow_{\Delta_2} \gamma_2$ also replaces a hiding operator with a variable $w_2 \in \Delta_1 \cup fv(\gamma_1)$. However, we have that $w_2 \notin fv(\gamma_1)$ since otherwise (as shown above) $w_1 = w_2$, thus ξ_1 is obtained by replacing in γ_1 the hiding operator with z_2 instead of w_2, and item 2 is then verified.

Among the remaining cases, the only relevant one is whenever both actions add different constraints to the store. So, let us assume that $\gamma = \langle A_1 \parallel A_2, \sigma \rangle$ such that $\langle A_1, \sigma \rangle \rightarrow_{\Delta_1} \langle B_1, \sigma_1 \rangle$ and $\langle A_2, \sigma \rangle \rightarrow_{\Delta_2} \langle B_2, \sigma_2 \rangle$. Note that since reduction semantics is monotone (Lemma 3) and σ is $\otimes$-compact, also σ_1 is $\otimes$-compact and furthermore we have $\sigma_1 = \sigma \otimes (\sigma_1 \ominus \sigma)$. Now, operational monotonicity (Lemma 4) ensures us that $\langle B_1 \parallel A_2, \sigma \otimes (\sigma_1 \ominus \sigma) \rangle \rightarrow_{\Delta_1 \cup \Delta_2} \langle B_1 \parallel B_2, \sigma \otimes (\sigma_1 \ominus \sigma) \otimes (\sigma_2 \ominus \sigma) \rangle$ and by symmetric reasoning the latter configuration is the one we were looking for. □

The result above is a local confluence theorem, which is expected, since the calculus is essentially deterministic. The complex formulation is due to the occurrence of hiding operators: as an example, different fresh variables may be chosen for replacing $\exists_x$, such as w_1 and w_2 in the first item above, and then a globally fresh variable z has to be found for replacing them.

As a final remark, note that $\gamma \rightarrow_{\Delta} \gamma'$ with $z \in fv(\gamma)$ and $w \notin fv(\gamma')$ implies $\gamma[^w/_z] \rightarrow_{(\Delta \setminus \{z\}) \cup \{w\}} \gamma'[^w/_z]$. Combined with the proposition above, they ensure that fair computations originating from a configuration are either all finite or all infinite, and furthermore they have the same result. So, in the following we denote as $Result(\langle A, \sigma \rangle)$ the unique result of the fair computations originating from $\langle A, \sigma \rangle$. This fact allows to define an observation-wise equivalence.

Definition 14 (Observational equivalence). *Let $A, B \in \mathcal{A}$ be agents. They are observationally equivalent ($A \sim_o B$) if $Result(\langle A, \sigma \rangle) = Result(\langle B, \sigma \rangle)$ for all $\sigma \in C^{\otimes}$.*

It is easily shown that $\sim_o$ is preserved by all contexts, i.e., it is a *congruence*.[10]

3.2 Saturated Bisimulation

As proposed in [1] for crisp languages, we define a barbed equivalence between two agents [17]. Since barbs are basic observations (predicates) on the states of a system, in this case they correspond to the compact constraints in $C^{\otimes}$, and we say that $\langle A, \sigma \rangle$ verifies c, or that $\langle A, \sigma \rangle \downarrow_c$ holds, if $c \leq \sigma$. However, since *barbed bisimilarity* is an equivalence already for CCP, along [1] we propose the use of *saturated bisimilarity* in order to obtain a congruence: Defs. 15 and 16 respectively provide the strong and weak definition of saturated bisimilarity.

Definition 15 (Saturated bisimilarity). *A saturated bisimulation is a symmetric relation R on configurations such that whenever $(\langle A, \sigma \rangle, \langle B, \rho \rangle) \in R$*

1. *if $\langle A, \sigma \rangle \downarrow_c$ then $\langle B, \rho \rangle \downarrow_c$;*
2. *if $\langle A, \sigma \rangle \longrightarrow \gamma'_1$ then there exists γ'_2 such that $\langle B, \rho \rangle \longrightarrow \gamma'_2$ and $(\gamma'_1, \gamma'_2) \in R$;*
3. *$(\langle A, \sigma \otimes d \rangle, \langle B, \rho \otimes d \rangle) \in R$ for all $d \in C^{\otimes}$.*

We say that γ_1 and γ_2 are saturated bisimilar ($\gamma_1 \sim_s \gamma_2$) if there exists a saturated bisimulation R such that $(\gamma_1, \gamma_2) \in R$. We write $A \sim_s B$ if $\langle A, \bot \rangle \sim_s \langle B, \bot \rangle$.

We now let $\longrightarrow^*$ denote the reflexive and transitive closure of $\longrightarrow$, restricted to increasing computations. We say that $\gamma \Downarrow_c$ holds if there exists $\gamma' = \langle A, \sigma \rangle$ such that $\gamma \longrightarrow^* \gamma'$ and $c \leq \exists_X \sigma$ for $X = fv(\gamma') \setminus fv(\gamma)$.

Definition 16 (Weak saturated bisimilarity). *Weak saturated bisimilarity ($\approx_s$) is obtained from Def. 15 by replacing $\longrightarrow$ with $\longrightarrow^*$ and $\downarrow_c$ with $\Downarrow_c$.*

Since $\sim_s$ (and $\approx_s$) is itself a saturated bisimulation, it is obvious that it is upward closed, and it is also a congruence with respect to all the contexts of SCCP (i.e., it is preserved under any context): indeed, a context $C[\bullet]$ can modify the behaviour of a configuration only by adding constraints to its store.

[10] Recall that a context $C[\bullet]$ is a syntactic expression with a single hole $\bullet$ such that replacing $\bullet$ with an agent A in the context produces an agent, denoted by $C[A]$. For example if $C[\bullet]$ is the context $\mathbf{tell}(c) \parallel \bullet$ then $C[A] = \mathbf{tell}(c) \parallel A$. An equivalence $\cong$ between agents is a congruence if $A \cong B$ implies $C[A] \cong C[B]$ for every context $C[\bullet]$.

We now show that $\approx_s$, as given in Def. 16, coincides with the observational equivalence $\sim_o$ (see Def. 14). First we recall the notion of and a classic result on *cofinality*: two (possibly infinite) chains $c_0 \leq c_1 \leq \ldots$ and $d_0 \leq d_1 \leq \ldots$ are said to be *cofinal* if for all c_i there exists a d_j such that $c_i \leq d_j$ and, viceversa, for all d_i there exists a c_j such that $d_i \leq c_j$.

Lemma 6. *Let $c_0 \leq c_1 \leq \ldots$ and $d_0 \leq d_1 \leq \ldots$ be two chains.* (1) *If they are cofinal, then they have the same limit, i.e., $\bigvee_i c_i = \bigvee_i d_i$.* (2) *If the elements of the chains are $\otimes$-compact and $\bigvee_i c_i = \bigvee_i d_i$, then the two chains are cofinal.*

Proof. Let us tackle (2), and consider the sequence $e_0 = c_0$ and $e_i = c_{i+1} \ominus c_i$. Each e_i is the difference between two consecutive elements of a chain. Since the CLIM is invertible we have $c_k = \bigotimes_{i \leq k} e_i$ and thus $\bigvee_i c_i = \bigotimes_i e_i$. Since each d_j is $\otimes$-compact and $d_j \leq \bigotimes_i e_i$, there is a k such that $d_j \leq \bigotimes_{i \leq k} e_i$. The same reasoning is applied to the chain $d_0 \leq d_1 \leq \ldots$, thus the result holds. □

For proving Proposition 2 we now relate weak barbs and fair computations.

Lemma 7. *Let $\xi = \gamma_0 \longrightarrow \gamma_1 \longrightarrow \gamma_2 \longrightarrow \ldots$ be a (possibly infinite) fair computation. If $\gamma_0 \Downarrow_d$ then there exists a store σ_i in ξ such that $d \leq \exists_{X_i} \sigma_i$ for $X_i = fv(\gamma_i) \setminus fv(\gamma_0)$.*

The lemma holds since the language is deterministic and computations fair.

Proposition 2. *$A \sim_o B$ if and only if $A \approx_s B$.*

Proof. The proof proceeds as follows.

From $\approx_s$ to $\sim_o$. Assume $\langle A, \bot \rangle \approx_s \langle B, \bot \rangle$ and take a $\otimes$-compact $c \in \mathcal{C}^{\otimes}$. Let

$$\langle A, c \rangle \longrightarrow \langle A_0, \sigma_0 \rangle \longrightarrow \langle A_1, \sigma_1 \rangle \longrightarrow \ldots \longrightarrow \langle A_n, \sigma_n \rangle \ldots \longrightarrow \ldots \quad (1)$$

$$\langle B, c \rangle \longrightarrow \langle B_0, \rho_0 \rangle \longrightarrow \langle B_1, \rho_1 \rangle \longrightarrow \ldots \longrightarrow \langle B_n, \rho_n \rangle \ldots \longrightarrow \ldots \quad (2)$$

be two fair computations. Since $\approx_s$ is upward closed, $\langle A, c \rangle \approx_s \langle B, c \rangle$ and thus $\langle B, c \rangle \Downarrow_{\sigma_i}$ for all σ_i. By Lemma 7, it follows that there exists an ρ_j (in the above computation) such that $\exists_{\Gamma_i} \sigma_i \leq \sigma_i \leq \exists_{\Gamma'_j} \rho_j$, and analogously for all ρ_i. Then $\sigma_0 \leq \sigma_1 \leq \ldots$ and $\rho_0 \leq \rho_1 \leq \ldots$ are cofinal and by Lemma 6, it holds that $\bigvee_i \exists_{\Gamma_i} \sigma_i = \bigvee_i \exists_{\Gamma'_i} \rho_i$, which means $Result(\langle A, c \rangle) = Result(\langle B, c \rangle)$.

From $\sim_o$ to $\approx_s$. Assume $A \sim_o B$. First, we show that $\langle A, c \rangle$ and $\langle B, c \rangle$ satisfy the same weak barbs for all $c \in \mathcal{C}$. Let (1) and (2) be two fair computations. Since $A \sim_o B$, then $\bigvee_i \exists_{\Gamma_i} \sigma_i = \bigvee_i \exists_{\Gamma'_j} \rho_i$. Since all (the projections of) the intermediate stores of the computations are $\otimes$-compact, then by Lemma 6, for all σ_i there exists an ρ_j such that $\exists_{\Gamma_i} \sigma_i \leq \exists_{\Gamma'_j} \rho_j$. Now suppose that $\langle A, c \rangle \Downarrow_d$. By Lemma 7, there exists a σ_i such that $d \leq \exists_{\Gamma_i} \sigma_i$. Thus $\langle B, c \rangle \Downarrow_d$.
It is now easy to prove that $R = \{(\gamma_1, \gamma_2) \mid \exists c. \langle A, c \rangle \longrightarrow^* \gamma_1 \& \langle B, c \rangle \longrightarrow^* \gamma_2\}$ is a weak saturated bisimulation (Def. 16). Take $(\gamma_1, \gamma_2) \in R$. If $\gamma_1 \Downarrow_d$ then $\langle A, c \rangle \Downarrow_d$ and, by the above observation, $\langle B, c \rangle \Downarrow_d$. Since SCCP is confluent, also $\gamma_2 \Downarrow_d$. The fact that R is closed under $\longrightarrow^*$ is evident from the definition of R. While for proving that R is upward-closed take $\gamma_1 = \langle A', \sigma' \rangle$ and $\gamma_2 = \langle B', \rho' \rangle$. By Lemma 4 for all $a \in \mathcal{C}$, $\langle A, c \otimes a \rangle \longrightarrow^* \langle A', \sigma' \otimes a \rangle$ and $\langle B, c \otimes a \rangle \longrightarrow^* \langle B', \rho' \otimes a \rangle$. Thus, by definition of R, $(\langle A', \sigma' \otimes a \rangle, \langle B', \rho' \otimes a \rangle) \in R$. □

4 A Labelled Transition System for Soft CCP

Although $\approx_s$ is fully abstract, it is to some extent unsatisfactory because of the upward-closure, namely, the for-all quantification in condition 3 of Def. 16.

In Tab. 2 we refine the notion of transition (given in Tab. 1) by adding a label that carries additional information about the constraints that cause the reduction. Hence, we define a new labelled transition system (LTS) obtained by the family of relations $\xrightarrow{\alpha}_{\Delta} \subseteq \Gamma \times \Gamma$ indexed over $\langle C^{\otimes}, 2^{V}\rangle$; as a reminder, Γ is the set of configurations, $C^{\otimes}$ the set of $\otimes$-compact constraints, and, as for the unlabelled semantics in Section 3, transitions are indexed by sets of variables. Rules in Tab. 2 are identical to those in Tab. 1, except for a constraint α that represents the minimal information that must be added to σ in order to fire an action from $\langle A, \sigma\rangle$ to $\langle A', \sigma'\rangle$, i.e., $\langle A, \sigma \otimes \alpha\rangle \longrightarrow_{\Delta} \langle A', \sigma'\rangle$.

$$\textbf{LR1}\ \frac{sv(\sigma) \cup sv(c) \subseteq \Delta}{\langle \mathbf{tell}(c), \sigma\rangle \xrightarrow{\perp}_{\Delta} \langle \mathbf{stop}, \sigma \otimes c\rangle}\ \textbf{Tell} \qquad \textbf{LR2}\ \frac{sv(\sigma) \cup sv(c) \subseteq \Delta}{\langle \mathbf{ask}(c) \rightarrow A, \sigma\rangle \xrightarrow{c \ominus \sigma}_{\Delta} \langle A, \sigma \otimes (c \ominus \sigma)\rangle}\ \textbf{Ask}$$

$$\textbf{LR3}\ \frac{\langle A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A', \sigma'\rangle \wedge fv(B) \subseteq \Delta}{\langle A \parallel B, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A' \parallel B, \sigma'\rangle}\ \textbf{Par1} \qquad \textbf{LR4}\ \frac{\langle A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A', \sigma'\rangle \wedge fv(B) \subseteq \Delta}{\langle B \parallel A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle B \parallel A', \sigma'\rangle}\ \textbf{Par2}$$

$$\textbf{LR5}\ \frac{\{y\} \cup sv(\sigma) \subseteq \Delta \wedge p(x) = A \in \mathcal{P}}{\langle p(y), \sigma\rangle \xrightarrow{\perp}_{\Delta} \langle A[^{y}/_{x}], \sigma\rangle}\ \textbf{Rec} \qquad \textbf{LR6}\ \frac{fv(A) \cup sv(\sigma) \subseteq \Delta \wedge w \notin \Delta}{\langle \exists_x A, \sigma\rangle \xrightarrow{\perp}_{\Delta} \langle A[^{w}/_{x}], \sigma\rangle}\ \textbf{Hide}$$

Table 2. An LTS for SCCP.

Rule **LR2** says that $\langle \mathbf{ask}(c) \rightarrow A, \sigma\rangle$ can evolve to $\langle A, \sigma \otimes \alpha\rangle$ if the environment provides a minimal constraint α that added to the store σ entails c, i.e., $\alpha = c \ominus \sigma$. Notice that, differently from [1], here the definition of this minimal label comes directly from a derived operator of the underlying CLIM (i.e., from $\ominus$), which by Lemma 1 preserves $\otimes$-compactness.

The LTS is sound and complete with respect to the unlabelled semantics.

Lemma 8 (Soundness). *If $\langle A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A', \rho\rangle$ then $\langle A, \sigma \otimes \alpha\rangle \longrightarrow_{\Delta} \langle A', \rho\rangle$.*

Proof. We proceed by induction on (the depth) of the inference of $\langle A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A', \rho\rangle$. We consider **LR2**: the other cases are easier to verify.

Using **LR2** then $A = (\mathbf{ask}(c) \rightarrow A')$, $\alpha = c \ominus \sigma$ and $\rho = (\sigma \otimes (c \ominus \sigma)) = (\sigma \otimes \alpha)$. We know that $c \leq (\sigma \otimes (c \ominus \sigma))$ then by using **R2** $\langle A, \sigma \otimes \alpha\rangle \longrightarrow_{\Delta} \langle A', \rho\rangle$. □

Lemma 9 (Completeness). *If $\langle A, \sigma \otimes d\rangle \longrightarrow_{\Delta} \langle A', \rho\rangle$ then there exist $\alpha, a \in C^{\otimes}$ such that $\langle A, \sigma\rangle \xrightarrow{\alpha}_{\Delta} \langle A', \rho'\rangle$ and $\alpha \otimes a = d$ and $\rho' \otimes a = \rho$.*

Proof. We proceed by induction on (the depth) of the inference of $\langle A, \sigma \otimes d\rangle \longrightarrow_{\Delta} \langle A', \rho\rangle$. We consider **LR2**: The other cases are easier to verify.

Using **LR2** then $A = \mathbf{ask}(c) \rightarrow A'$, $\rho = \sigma \otimes d$ and $c \leq \rho$. Now consider $\langle A, \sigma \rangle \xrightarrow{\alpha}_{\Delta} \langle A', \rho' \rangle$, where $\alpha = (c \ominus \sigma) \leq d$ and $\rho' = (\sigma \otimes \alpha)$. Take $a = d \ominus (c \ominus \sigma)$ then we can check that the conditions verify. First by invertibility $\alpha \otimes a = (c \ominus \sigma) \otimes (d \ominus (c \ominus \sigma)) = d$ and finally $\rho' \otimes a = \sigma \otimes \alpha \otimes a = \sigma \otimes d = \rho$. □

Theorem 1. *$\langle A, \sigma \rangle \xrightarrow{\perp}_{\Delta} \langle A', \sigma' \rangle$ if and only if $\langle A, \sigma \rangle \longrightarrow_{\Delta} \langle A', \sigma' \rangle$.*

Strong and Weak Bisimilarity on the LTS. We now proceed to define an equivalence that characterises $\sim_s$ without the upward closure condition. Differently from languages such as Milner's CCS, barbs cannot be removed from the definition of bisimilarity because they cannot be inferred by the transitions.

Definition 17 (Strong bisimilarity). *A strong bisimulation is a symmetric relation R on configurations such that whenever $(\gamma_1, \gamma_2) \in R$ with $\gamma_1 = \langle A, \sigma \rangle$ and $\gamma_2 = \langle B, \rho \rangle$*

1. *if $\gamma_1 \downarrow_c$ then $\gamma_2 \downarrow_c$,*
2. *if $\gamma_1 \xrightarrow{\alpha} \gamma_1'$ then $\exists \gamma_2'$ such that $\langle B, \rho \otimes \alpha \rangle \longrightarrow \gamma_2'$ and $(\gamma_1', \gamma_2') \in R$.*

We say that γ_1 and γ_2 are strongly bisimilar ($\gamma_1 \sim \gamma_2$) if there exists a strong bisimulation R such that $(\gamma_1, \gamma_2) \in R$.

Whenever σ and ρ are $\otimes$-compact elements, the first condition is equivalent to require $\sigma \leq \rho$. Thus $(\gamma_1, \gamma_2) \in R$ would imply that γ_1 and γ_2 have the same store. As for the second condition, we adopted a *semi-saturated* equivalence, introduced for CCP in [1]. In the bisimulation game a label can be simulated by a reduction including in the store the label itself.

Definition 18 (Weak bisimilarity). *A weak bisimulation is a symmetric relation R on configurations such that whenever $(\gamma_1, \gamma_2) \in R$ with $\gamma_1 = \langle A, \sigma \rangle$ and $\gamma_2 = \langle B, \rho \rangle$*

1. *if $\gamma_1 \downarrow_c$ then $\gamma_2 \Downarrow_c$,*
2. *if $\gamma_1 \xrightarrow{\alpha} \gamma_1'$ then $\exists \gamma_2'$ such that $\langle B, \rho \otimes \alpha \rangle \longrightarrow^* \gamma_2'$ and $(\gamma_1', \gamma_2') \in R$.*

We say that γ_1 and γ_2 are weakly bisimilar ($\gamma_1 \approx \gamma_2$) if there exists a weak bisimulation R such that $(\gamma_1, \gamma_2) \in R$.

With respect to the weak equivalence for crisp constraints, some of its characteristic equivalences do not hold, so that e.g. $\mathbf{ask}(c) \rightarrow \mathbf{tell}(c) \not\approx \mathbf{stop}$. As usual, this is linked to the fact that the underlying CLIM may not be idempotent.

We can now conclude by proving the equivalence between $\sim_s$ and $\sim$ and between $\approx_s$ and $\approx$ (hence, $\approx$ is further equivalent to $\sim_o$, using Proposition 2). We start by showing that $\sim$ is preserved under composition.

Lemma 10. *If $\langle A, \sigma \rangle \sim \langle B, \rho \rangle$, then $\langle A, \sigma \otimes a \rangle \sim \langle B, \rho \otimes a \rangle$ for all $a \in C^{\otimes}$.*

Proof. We need to show that $R = \{(\langle A, \sigma \otimes a \rangle \sim \langle B, \rho \otimes a \rangle) \mid \langle A, \sigma \rangle \sim \langle B, \rho \rangle\}$ satisfies the two properties in Def. 17.

i) From the hypothesis $\langle A, \sigma\rangle \sim \langle B, \rho\rangle$, we have that $\rho = \sigma$, thus $\langle A, \sigma \otimes a\rangle$ and $\langle B, \rho \otimes a\rangle$ satisfy the same barbs.

ii) Supposing $\langle A, \sigma \otimes a\rangle \xrightarrow{\alpha} \langle A', \sigma'\rangle$, we need to prove the existence of B' and ρ' such that $\langle B, \rho \otimes a \otimes \alpha\rangle \to \langle B', \rho'\rangle$ and $(\langle A', \sigma'\rangle, \langle B', \rho'\rangle) \in R$. By Lemma 8 and Lemma 9 we obtain $\langle A, \sigma\rangle \xrightarrow{\alpha'} \langle A', \sigma''\rangle$, and there exists b' such that $\alpha' \otimes b' = a \otimes \alpha$ (1) and $\sigma'' \otimes b' = \sigma'$ (2). From the labelled transition of $\langle A, \sigma\rangle$ and the hypothesis $\langle A, \sigma\rangle \sim \langle B, \rho\rangle$, we have that $\langle B, \rho \otimes \alpha'\rangle \to \langle B', \rho''\rangle$, with $\langle A, \sigma''\rangle \sim \langle B, \rho''\rangle$ (3). By (1) we have $\langle B, \rho \otimes a \otimes \alpha\rangle = \langle B, \rho \otimes \alpha' \otimes b'\rangle$ and $\langle B, \rho \otimes \alpha' \otimes b'\rangle \to \langle B, \rho'' \otimes b'\rangle$ (due to operational monotonicity). Finally, by the definition of R and (3), we conclude that $(\langle A', \sigma'' \otimes b'\rangle, \langle B', \rho'' \otimes b'\rangle) \in R$, and, by (2), $\langle A', \sigma'' \otimes b'\rangle = \langle A', \sigma'\rangle$. □

Theorem 2. $\sim_s = \sim$

Proof. The equivalence $\sim_s = \sim$ can be proved by using Lemma 10.

From $\sim$ to $\sim_{sb}$. We show that $R = \{(\langle A, \sigma\rangle, \langle B, \rho\rangle) \mid \langle A, \sigma\rangle \sim \langle B, \rho\rangle\}$ is a saturated bisimulation, i.e., for $(\langle A, \sigma\rangle, \langle B, \rho\rangle) \in R$ the conditions in Def. 15 are satisfied

i) If $\langle A, \sigma\rangle \downarrow_c$, then we have $\langle B, \rho\rangle \downarrow_c$ by the hypothesis $\langle A, \sigma\rangle \sim \langle B, \rho\rangle$.

ii) Suppose that $\langle A, \sigma\rangle \to \langle A', \sigma'\rangle$. By Theorem 1 we have $\langle A, \sigma\rangle \xrightarrow{\perp} \langle A', \sigma'\rangle$. Since $\langle A, \sigma\rangle \sim \langle B, \rho\rangle$, then $\langle B, \rho \otimes \perp\rangle \to \langle B', \rho'\rangle$ with $\langle A', \sigma'\rangle \sim \langle B', \rho'\rangle$. Since $\rho = \rho \otimes \perp$, we have $\langle B, \rho\rangle \to \langle B', \rho'\rangle$.

iii) By Lemma 10, $(\langle A, \sigma \otimes c'\rangle, \langle B, \rho \otimes c'\rangle) \in R$ for all $c' \in C^{\otimes}$.

From $\sim_{sb}$ to $\sim$. We show that $R = \{(\langle A, \sigma\rangle, \langle B, \rho\rangle) \mid \langle A, \sigma\rangle \sim_{sb} \langle B, \rho\rangle\}$ is a strong bisimulation, i.e., for $(\langle A, \sigma\rangle, \langle B, \rho\rangle) \in R$ the conditions in Def. 17 are satisfied

i) If $\langle A, \sigma\rangle \downarrow_c$, then we have $\langle B, \rho\rangle \downarrow_c$ by the hypothesis $\langle A, \sigma\rangle \sim_{sb} \langle B, \rho\rangle$.

ii) Suppose that $\langle A, \sigma\rangle \xrightarrow{\alpha} \langle A', \sigma'\rangle$. Then by Lemma 8 we have $\langle A, \sigma \otimes \alpha\rangle \to \langle A', \sigma'\rangle$. Since $\langle A, \sigma\rangle \sim_{sb} \langle B, \rho\rangle$, then $\langle A, \sigma \otimes \alpha\rangle \sim_{sb} \langle B, \rho \otimes \alpha\rangle$ and thus $\langle B, \rho \otimes \alpha\rangle \to \langle B', \rho'\rangle$ with $\langle A', \sigma'\rangle \sim_{sb} \langle B', \rho'\rangle$. □

In order to prove the correspondence between weak bisimulations, we need a result analogous to Lemma 10. The key issue is the preservation of weak barbs by the addition of constraints to the store, which is trivial in strong bisimulation.

Lemma 11. *Let $\langle A, \sigma\rangle \approx \langle B, \rho\rangle$ and $a, c \in C^{\otimes}$. If $\langle A, \sigma \otimes a\rangle \downarrow_c$, then $\langle B, \rho \otimes a\rangle \Downarrow_c$.*

Proof. If $\langle A, \sigma \otimes a\rangle \downarrow_c$, then $c \leq \sigma \otimes a$. Since $\langle A, \sigma\rangle \approx \langle B, \rho\rangle$ and $\langle A, \sigma\rangle \downarrow_\sigma$, then there exists $\langle B', \rho'\rangle$ such that $\langle B, \rho\rangle \to^* \langle B', \rho'\rangle$ and $\sigma \leq \exists_\Gamma \rho'$ for $\Gamma = fv(\langle B', \rho'\rangle) \setminus fv(\langle B, \rho\rangle)$. Let us assume, without loss of generality, that $\Gamma \cap (sv(a)) = \emptyset$; since reductions are operationally monotone (Lemma 4), we have $\langle B, \rho \otimes a\rangle \to^* \langle B', \rho' \otimes a\rangle$. Finally, $c \leq \sigma \otimes a = \sigma \otimes \exists_\Gamma a \leq \exists_\Gamma \rho' \otimes \exists_\Gamma a \leq \exists_\Gamma(\rho' \otimes a)$, hence $\langle B, \rho \otimes a\rangle \Downarrow_c$. □

The result below uses Lemma 11 and a rephrasing of the proof of Lemma 10

Lemma 12. *If $\langle A, \sigma\rangle \approx \langle B, \rho\rangle$, then $\langle A, \sigma \otimes a\rangle \approx \langle B, \rho \otimes a\rangle$ for all $a \in C^{\otimes}$.*

Theorem 3. $\approx_s = \approx$

Labelled versus saturated semantics. The main appeal of saturated semantics resides in always being a congruence and, in fact, the minimal congruence contained in standard bisimulation [19]. The main drawback of this approach is that it is in principle necessary to check the behaviour of a process under every context. The problem is somewhat mitigated for SCCP, since it suffices to close the store with respect to any possible compact element (item 3 of Def. 15). At the same time, checking the feasibility of a reduction may require some computational effort, either for solving the combinatorial problem associated with calculating $\sigma \otimes d$, or for verifying if $c \leq \sigma$, as with agent $\mathbf{ask}(c) \ \rightarrow \ A$.

This is the reason for searching labelled semantics and suitable notions of bisimilarity that may alleviate such a burden. The key intuition is to consider labels which somehow represent the "minimal context allowing a process to reduce", so that a bisimilarity-checking algorithm in principle needs to verify this minimal context only, instead of every one. The idea has been exploited in the simpler framework of crisp CCP [1], and it is based on [16,9].

Example 1. Let us consider the agents $\mathbf{ask}(c) \ \rightarrow \ \mathbf{stop}$ and $\mathbf{stop}$. To prove that they are weakly bisimilar, it has to be proved that $\gamma \approx \gamma'$ for configurations $\gamma = \langle \mathbf{ask}(c) \ \rightarrow \ \mathbf{stop}, \bot \rangle$ and $\gamma' = \langle \mathbf{stop}, \bot \rangle$. Consider the following relation

$$\mathcal{R} = \{(\langle \mathbf{ask}(c) \ \rightarrow \ \mathbf{stop}, \bot \rangle, \langle \mathbf{stop}, \bot \rangle), (\langle \mathbf{stop}, c \rangle, \langle \mathbf{stop}, c \rangle)\}$$

It is quite easy to prove that it is a bisimulation, and in fact the smallest one identifying the two configurations. It suffices to note that by definition $c \ominus \bot = c$.

In order to prove that $\gamma \approx_s \gamma'$, instead, we surely need to consider an infinite relation. Indeed, the smallest saturated bisimulation equating the two configuration is given by the relation below

$$\mathcal{S} = \{(\langle \mathbf{ask}(c) \ \rightarrow \ \mathbf{stop}, d \rangle, \langle \mathbf{stop}, d \rangle), (\langle \mathbf{stop}, e \rangle, \langle \mathbf{stop}, e \rangle) \mid d, e \in C^{\otimes} \ \& \ c \leq e\}$$

The relation above clearly is a saturated bisimulation, but any naive automatic check for that property might involve rather complex calculations.

Another reason for the complexity of checking saturated bisimilarity is the need of considering the closure $\longrightarrow^*$ of the reduction relation, which may cause a combinatorial explosion. Think e.g. of the agents $\prod_{i \in I} \mathbf{ask}(c_i) \ \rightarrow \ \mathbf{stop}$ and $\mathbf{stop}$. Of course, they might be proved equivalent by exploiting the fact that saturated bisimilarity is a congruence, and by verifying that $\mathbf{stop} \parallel A \approx_s A$ for all the agents A. A direct proof would instead require a check for each store of the reductions arising from all the possible interleaving of the c_i elements.

5 Towards an Axiomatisation for Weak Bisimilarity

Once the behaviour of an agent is captured by an observational equivalence, it is natural to look for laws characterizing it. Given its correspondence with the standard equivalence via fair computations, weak bisimilarity is the preferred behavioural semantics for soft CCP. A sound and complete axiomatisation was proposed for CCP in [21]. Unfortunately, the lack of idempotence in the soft formalism makes unsound some of the axioms presented in that classical paper.

$$\mathbf{ask}(c) \rightarrow \mathbf{stop} = \mathbf{stop} \quad (1) \qquad \mathbf{tell}(\bot) = \mathbf{stop} \quad (2)$$

$$\mathbf{ask}(\bot) \rightarrow A = A \quad (3) \qquad A \parallel \mathbf{stop} = A \quad (4)$$

$$A \parallel B = B \parallel A \quad (5) \qquad A \parallel (B \parallel C) = (A \parallel B) \parallel C \quad (6)$$

$$\mathbf{tell}(c) \parallel \mathbf{tell}(d) = \mathbf{tell}(c \otimes d) \quad (7) \qquad \mathbf{ask}(c) \rightarrow (A \parallel B) = (\mathbf{ask}(c) \rightarrow A) \parallel (\mathbf{ask}(c) \rightarrow B) \quad (8)$$

$$\exists_x \mathbf{tell}(c) = \mathbf{tell}(\exists_x c) \quad (9) \qquad \exists_x(\mathbf{ask}(c) \rightarrow A) = \mathbf{ask}(\forall_x c) \rightarrow \exists_x A \quad (10)$$

$$\exists_x(\mathbf{tell}(c) \parallel_{i \in I} \mathbf{ask}(c_i) \rightarrow \mathbf{tell}(d_i)) = \mathbf{tell}(\exists_x c) \parallel \exists_x(\parallel_{i \in I} \mathbf{ask}(c \Rightarrow_x c_i) \rightarrow \mathbf{tell}(d_i)) \quad (11)$$

Fig. 1. Axioms for simple agents (1-8) and for agents with quantifiers (9-11).

Consider e.g. the law $\mathbf{ask}(c) \rightarrow \mathbf{ask}(d) \rightarrow \mathbf{tell}(e) = \mathbf{ask}(c \otimes d) \rightarrow \mathbf{tell}(e)$, denoted as $L3$ in [21], and let us assume that $c = d$. Since $c \neq c \otimes c$, only the agent in the left-hand side of the law is guaranteed to add e, starting from a store σ such that $c \leq \sigma$. On a similar note, most of the axioms in [21] involving the parallel composition also do not hold, since as a general remark posting a constraint twice is different from adding it just once.[11]

We now introduce a set of sound axioms for SCCP in Figure 1. As for those of CCP in [21], they rely on an additional operator which is intuitively the dual of the existential quantifier of cylindric algebras.

Definition 19 (Co-cylindrification). *Let V be a set of variables. A co-cylindric operator $\forall$ over $\mathbb{S}$ and V is given by a family of monotone, $\otimes$-compactness preserving functions $\forall_x : A \rightarrow A$ indexed by elements in V such that for all $a, b \in A$ and $x \in V$*

1. *$\forall_x a \leq b$ if and only if $a \leq \exists_x b$.*

If the $\forall$ operators play the role of universal quantifiers, a further family of operators had been introduced in [21] for providing the role of implication, in order to provide a complete set of axioms for CCP. In our context, such an operator can be derived by means of residuation.

Lemma 13. *Let $a, b, c \in C$, $x \in V$ and $a \Rightarrow_x b = \exists_x a \otimes \forall_x(b \ominus a)$. Then, $b \leq a \otimes \exists_x c$ if and only if $a \Rightarrow_x b \leq \exists_x a \otimes \exists_x c$.*

Clearly, $a \Rightarrow_x b \in C^{\otimes}$ if a and b do. These properties for $a \Rightarrow_x b$ are the immediate extensions of those holding for the crisp setting. Exploiting co-cylindrification and the latter operator we can now state Eq. 10 and Eq. 11. In Eqs. 1-3 we present the axioms related to *ask* and *tell*. Axioms on parallel composition are instead represented in Eqs. 4-6. In Eqs. 7-8 we show how adding two constraints and prefixing distributes though parallel composition.

Proposition 3. *Axioms 1-11 in Figure 1 are sound with respect to weak bisimilarity.*

[11] As an example, the law $L1$ of [21] states $\mathbf{ask}(c) \rightarrow \mathbf{tell}(d) = \mathbf{ask}(c) \rightarrow (\mathbf{tell}(c) \parallel \mathbf{tell}(d))$, which is false precisely for the lack of idempotence: $c \leq \sigma$ does not imply $\sigma = \sigma \otimes c$. For the sake of completeness, the other unsound axioms are $L10$, $L11$, and $L12$.

As for completeness, again the lack of idempotency made it impossible to rest the proof schema adopted for the CCP case, since the normal form exploited in [21] for proving completeness cannot be lifted to SCCP agents.

6 Conclusions and Further Work

Inspired by [1] that investigated the crisp variant of the language, in this paper we studied the behavioural semantics of the deterministic fragment of soft CCP [7], and proposed a sound axiomatisation in the spirit of [21].

Using residuation theory (as e.g. in [5] for soft constraints problems) provides an elegant way to define the minimal information that enables the firing of actions in the LTS shown in Sec. 4. This choice allowed for the study of the observational equivalence of agents in terms of weak and strong bisimilarity on such LTS, and it allowed for relating them to the corresponding barbed bisimilarities of (unlabelled) reductions and with the standard semantics via fair computations. The two kinds of equivalences, as well as the sound axiomatisation for weak bisimilarity, are presented in this paper for the first time.

For future work, we plan to provide a complete axiomatisation and a denotational semantics for soft CCP by building on the work for the crisp case in [21]. Concerning the axioms, we will try and investigate the relationship between soft CCP and a logical system whose fundamental properties are closely related to the ones we have investigated in this paper; namely *affine linear logic* [11]. This logical system *rejects contraction* but *admits weakening*, which intuitively correspond to dropping idempotence and preserving monotonicity in the soft formalism. The denotational model of CCP is based on *closure operators*: Each agent is compositionally interpreted as a monotonic, extensive and idempotent operator/function on constraints. We shall then investigate a denotational model for soft CCP processes based on *pre-closure operators* [2] (or *Čech closure operators*), i.e., closure operators that are not required to be idempotent.

Finally, we plan to consider two extensions of the language, checking how far the results given in this paper can be adapted. As evidenced by [20] a non-deterministic extension is an interesting challenge since the closure under any context for the saturated bisimilarity gets more elaborated than just closing with respect to the addition of constraints (Defs. 15 and 16, condition 3), and similarly one also needs to find the right formulation of bisimilarity for the labelled transitions systems. Also, the presence of residuation makes intuitive the definition of a retract operator for the calculus. Even if the operational semantics would be less affected, retraction would require a complete reformulation of the denotational semantics via fair computations, since monotonicity (as stated in Lemma 3) would not hold anymore [8]. Finally, we might consider languages with temporal features, such as *timed* SCCP [4], where a reduction takes a bounded period of time and it is measured by a discrete global clock. Maximal parallel steps are adopted there with a new construct that can e.g. express time-out and pre-emption, and developing suitable temporal variants of bisimilarity might reveal a worthwhile, albeit difficult task.

References

1. Aristizábal, A., Bonchi, F., Palamidessi, C., Pino, L.F., Valencia, F.D.: Deriving labels and bisimilarity for concurrent constraint programming. In: Hofmann, M. (ed.) FOSSACS 2011. LNCS, vol. 6604, pp. 138–152. Springer (2011)
2. A.V.Arkhangelskii, L.S.Pontryagin: General Topology I. Springer (1990)
3. Baccelli, F., Cohen, G., Olsder, G., Quadrat, J.P.: Synchronization and Linearity: An Algebra for Discrete Event Systems. Wiley (1992)
4. Bistarelli, S., Gabbrielli, M., Meo, M.C., Santini, F.: Timed soft concurrent constraint programs. In: Lea, D., Zavattaro, G. (eds.) COORDINATION. LNCS, vol. 5052, pp. 50–66 (2008)
5. Bistarelli, S., Gadducci, F.: Enhancing constraints manipulation in semiring-based formalisms. In: Brewka, G., Coradeschi, S., Perini, A., Traverso, P. (eds.) ECAI 2006. FAIA, vol. 141, pp. 63–67. IOS Press (2006)
6. Bistarelli, S., Montanari, U., Rossi, F.: Semiring-based constraint satisfaction and optimization. Journal of ACM 44(2), 201–236 (1997)
7. Bistarelli, S., Montanari, U., Rossi, F.: Soft concurrent constraint programming. ACM Transactions on Computational Logic 7(3), 563–589 (2006)
8. Bistarelli, S., Santini, F.: A secure non-monotonic soft concurrent constraint language. Fundamamenta Informaticae 134(3-4), 261–285 (2014)
9. Bonchi, F., Gadducci, F., Monreale, G.V.: Reactive systems, barbed semantics, and the mobile ambients. In: de Alfaro, L. (ed.) FOSSACS 2009. pp. 272–287. LNCS (2009)
10. Buscemi, M.G., Montanari, U.: CC-Pi: A constraint-based language for specifying service level agreements. In: De Nicola, R. (ed.) ESOP 2007. LNCS, vol. 4421, pp. 18–32. Springer (2007)
11. Dal Lago, U., Martini, S.: Phase semantics and decidability of elementary affine logic. Theoretical Computer Science 318(3), 409–433 (2004)
12. Fargier, H., Lang, J.: Uncertainty in constraint satisfaction problems: a probabilistic approach. In: Clarke, M., Kruse, R., Moral, S. (eds.) ECSQUARU 1993. LNCS, vol. 747, pp. 97–104. Springer (1993)
13. Galatos, N., Jipsen, P., Kowalski, T., Ono, H.: Residuated Lattices: An Algebraic Glimpse at Substructural Logics, vol. 151. Elsevier (2007)
14. Golan, J.: Semirings and Affine Equations over Them: Theory and Applications. Kluwer (2003)
15. Karner, G.: Semiring-based constraint satisfaction and optimization. Semigroup Forum 45(XX), 148–165 (1992)
16. Leifer, J.J., Milner, R.: Deriving bisimulation congruences for reactive systems. In: Palamidessi, C. (ed.) CONCUR 2000. pp. 243–258. LNCS (2000)
17. Milner, R., Sangiorgi, D.: Barbed bisimulation. In: Kuich, W. (ed.) ICALP 1992. LNCS, vol. 623, pp. 685–695. Springer (1992)
18. Monk, J.D.: An introduction to cylindric set algebras. Logic Journal of IGPL 8(4), 451–496 (2000)
19. Montanari, U., Sassone, V.: Dynamic congruence vs. progressing bisimulation for CCS. Fundamenta informaticae 16(2), 171–199 (1992)
20. Pino, L.F., Bonchi, F., Valencia, F.D.: A behavioral congruence for concurrent constraint programming with non-deterministic choice. In: Ciobanu, G., Méry, D. (eds.) ICTAC 2014. LNCS, vol. 8687, pp. 351–368. Springer (2014)
21. Saraswat, V.A., Rinard, M.C., Panangaden, P.: Semantic foundations of concurrent constraint programming. In: Wise, D.S. (ed.) POPL 1991. pp. 333–352. ACM Press (1991)