Recovering a Labelled Semantics for Soft CCP with Local Variables*

Fabio Gadducci
Dipartimento di Informatica,
Università di Pisa, Italy
gadducci@di.unipi.it

Francesco Santini
IIT-CNR,
Pisa, Italy
francesco.santini@iit.cnr.it

Extended Abstract. While the dynamics of a computational system is nowadays more often specified operationally by means of a reduction system (RS), a labelled semantics is needed in order to take advantage of the tools for checking observational properties and equivalences. The methodological advance of the seminal paper [7] is the recognition that labelled semantics should be derivable from reduction semantics in a uniform and systematic way. The proposal of [7] is to build a labelled transition system (LTS) from a RS by identifying labels as the *minimal contexts* needed to trigger a reduction.

Unfortunately, this notion of minimality, called *relative pushout*, proved to be difficult to check and apply, also for most basic calculi such as CCS [4, 8]. A recent proposal [5] suggests to remove the requirement of minimality, in order to characterize a class of LTSs verifying weaker coherence properties, yet ensuring the adequacy of the observational equivalence.

One of the testbed of the proposal has been *Concurrent Constraint Programming* (CCP) [9], a language based on a shared-memory communication pattern: processes interact by either posting or checking partial information, 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.

A key aspect of CCP is the *idempotency* of constraint composition: adding an information twice does not change the store. In the soft variant of the formalism (Soft CCP, SCCP [2]), constraint systems may distinguish the number of occurrences of a piece of information. Dropping idempotency requires a full reworking of the theory. Although an operational semantics for SCCP has been devised [2], neither the denotational nor the labelled one has been reintroduced.

The work in [9] 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 are well-defined. The soft version [2] drops the upper bound for combination in exchange of a monoidal operator. Thus, the domain is just a (not

*Research partially supported by the MIUR PRIN 2010LHT4KM CINA and 2010XSEMLC "Security Horizons".

Submitted to:
MeMo 2015

$$\frac{\langle A,\sigma' \otimes \exists_x \sigma_0\rangle \longrightarrow \langle B,\sigma''\rangle \quad \textit{with } \sigma_0 = \sigma \oplus \exists_x \sigma'}{\langle \exists_x^{\sigma'} A,\sigma\rangle \longrightarrow \langle \exists_x^{\sigma_1} B,\sigma_0 \otimes \exists_x \sigma_1\rangle \quad \textit{with } \sigma_1 = \sigma'' \ominus \exists_x \sigma_0} \quad \textbf{Hide}$$

$$\frac{\langle A,\sigma' \otimes \sigma_0[^z/_x]\rangle \xrightarrow{\alpha} \langle B,\sigma''\rangle \quad \textit{with } \sigma_0 = \sigma \oplus \exists_x \sigma', x \notin sv(\alpha), z \notin fv(A) \cup sv(\sigma) \cup sv(\sigma')}{\langle \exists_x^{\sigma'} A,\sigma\rangle \xrightarrow{\alpha[^x/_z]} \langle \exists_x^{\sigma_1} B,\sigma_0 \otimes \alpha[^x/_z] \otimes \exists_x \sigma_1\rangle \quad \textit{with } \sigma_1 = \sigma'' \ominus (\alpha \otimes \sigma_0[^z/_x])} \quad \textbf{L-Hide}$$

Table 1: Unlabelled and labelled version of the rule for the local hiding of variables.

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 denotational semantics.

In [6] we connected the works on the soft [2] and the classical (also indicated in the literature as "crisp") [1, 9] paradigm by investigating a labelled (and an unlabelled) semantics for a deterministic fragment of SCCP. In particular, the result was a mix of those investigated in the two communities, namely, a monoid whose underlying set of elements form a complete lattice. Residuation theory provided an elegant way to define a weak inverse operator of tensor $\otimes$ with the purpose to determine the minimal information that enables the firing of actions in the LTS, thus distilling a suitable notion of labelled transition.

The operational semantics chosen in [6] favoured a global view of variables, namely, the agent $\exists_x A$, roughly corresponding to an existential quantification over the variable x of the agent A, would evolve to the agent $A[^y/_x]$, for y a fresh variable (with respect to the agent A and the store it is valued in). In this work we consider instead local variables, where the hiding operator carries some information on the variables it abstracts. More precisely, according to [3] we consider an extended operator $\exists_x^\sigma$, for σ the local store. Thanks again to the residuation operator, the rule for the extended hidings can be defined as **Hide** in Tab. 1. The intuition is that variable x may be local to a component $\exists_x \sigma'$ of the store σ, yet visible at a global level: we must then evaluate A in the store when the local x is hidden, yet the possible duplications are removed (e.g., $\exists_x \sigma'$ may already occur in the global store σ). The final store intuitively contains in σ_1 the original σ' increased by what has been added by the step.

In the labelled version of the rule (**L-Hide** in Tab. 1) we rename the global x with a fresh variable z, instead of hiding x in the global store, as we do in the corresponding unlabelled rule. We accomplish this in order to keep track of the global x in α: finally, in the result we restore the occurrences of z back to x.

The work is rounded up by a preliminary correspondence results between fair computations and bisimulation for soft CCP with local variables.

References

[1] Andrés Aristizábal, Filippo Bonchi, Catuscia Palamidessi, Luis Fernando Pino & Frank D. Valencia (2011): *Deriving labels and bisimilarity for concurrent constraint programming*. In Martin Hofmann, editor: *FOSSACS 2011*, *LNCS* 6604, Springer, pp. 138–152.

[2] Stefano Bistarelli, Ugo Montanari & Francesca Rossi (2006): *Soft concurrent constraint programming*. *ACM ToCL* 7(3), pp. 563–589.

[3] Frank S. de Boer, Maurizio Gabbrielli, Elena Marchiori & Catuscia Palamidessi (1997): *Proving concurrent constraint programs correct*. *ACM ToPLaS* 19(5), pp. 685–725.

[4] Filippo Bonchi, Fabio Gadducci & Barbara König (2009): *Synthesising CCS bisimulation using graph rewriting*. *I& C* 207(1), pp. 14–40.

[5] Filippo Bonchi, Fabio Gadducci & Giacoma Valentina Monreale (2014): *A general theory of barbs, contexts, and labels*. *ACM ToCL* 15(4), pp. 35:1–35:27.

[6] Fabio Gadducci, Luis Pino, Francesco Santini & Frank Valencia (2015): *A labelled semantics for soft CCP*. In T. Holvoet & M. Viroli, editors: *COORDINATION*, *LNCS* 9037, pp. 133–149.

[7] James J. Leifer & Robin Milner (2000): *Deriving bisimulation congruences for reactive systems*. In Catuscia Palamidessi, editor: *CONCUR 2000*, LNCS, pp. 243–258.

[8] Robin Milner (2006): *Pure bigraphs: Structure and dynamics*. *I&C* 204(1), pp. 60–122.

[9] Vijay A. Saraswat, Martin C. Rinard & Prakash Panangaden (1991): *Semantic foundations of concurrent constraint programming*. In David S. Wise, editor: *POPL 1991*, ACM Press, pp. 333–352.