

On modalities and linearity

Elaine Pimentel

joint work with Dale Miller, Vivek Nigam, Giselle Reis,
Carlos Olarte and Agata Ciabattoni

Universidade Federal do Rio Grande do Norte – Brazil

Abstract

This is a work about how to use linear logic (with or without subexponentials) as a logical framework to specify sequent calculus proof systems as well as concurrent computational systems. In particular, we present simple and decidable conditions for guaranteeing that the cut rule and non-atomic initial rules can be eliminated and we show how to use the so called subexponentials in order to deal with more involving logical and computational systems.

1 Linear logic and the specification of sequent systems

A major advantage of using a “logical” or “type-theoretical” framework to specify a proof or a computational system is that they support levels of abstraction that facilitate writing declarative specifications of object-logic proof systems. For example, if the framework contains λ -binding, α , β , η -conversions, and higher-order quantification, then these can be used to specify all formula-level and proof-level bindings and the associated notions of equality and substitution. Similarly, these frameworks provide context management (via eigenvariables and hypothetical assumptions) that an object-level specification can exploit directly. Furthermore, implementations of such frameworks using unification and backtracking search can provide partial or complete implementations of natural deduction specifications [NM88].

We start by showing how to use classical linear logic in order to encode and reason about different logical systems. The availability of classical principles in the framework allows us to capture directly the dualities that are present within the sequent calculi, such as De Morgan dualities, left/right duality, and the cut rule/initial rule duality. A linear logic meta-logic allows us to write simple linear logic formulas that capture the relationship between specification of left and right-introduction rules for an object-level logical connective. We show that if those formulas are, in fact, theorems of linear logic then the cut rule and non-atomic initial inference rules can be eliminated. Furthermore, we prove that if those linear logic formulas have proofs then they have short proofs: this makes it possible to show that this sufficient condition for cut-elimination and non-atomic initial-elimination is decidable.

We can extend this results to quasi-canonical systems, hence having a first syntactical cut-elimination proof for a set of paraconsistent systems – the so called C-systems [AKZ13].

This is a joint work with Dale Miller.

2 Linear logic and logical harmony

We can use the approach just presented in order to define and deal with the so called *harmonical systems*.

Although there is no precise notion of logical harmony, intuition given by Gentzen himself suggests that there should be a proper *balance* between rules for each connective in a deductive system in order to guarantee that the system itself has “good properties”.

Such a balance should be determined only by the syntactic structure of the rules, that is, one should be able to give the semantical meaning of logical connectives without having to resort to a particular domain or a denotational function.

In the natural deduction setting, Francez [Fra14] and Francez and Dyckhoff [FD12] proposed a construction of general elimination rules from given introduction rules, and proved that the constructed rules satisfy the so called *local intrinsic harmony*. The main idea is, of course, to use the *inversion method* in order to go from I-rules to E-rules.

It turns out that we can define the notions of general and local intrinsic harmony in the sequent calculus by using classical linear logic in order to construct left rules from right ones for a given connective. The idea is to build *coherent* rules so that the resulting systems will have *the good properties* of cut-elimination, subformula property and analicity.

Moreover, using the techniques developed in [CGT08], we can go one step further and show how to build harmonical systems to a number of modal connectives and mathematical theories as the ones present in [NvP11].

This is a joint work with Agata Ciabattoni, Carlos Olarte and Vivek Nigam.

3 Subexponentials and the specification of more involving sequent systems

Since exponentials in linear logic are not canonical, one can construct linear logic proof systems containing as many *subexponentials* as one needs.

This feature made it possible to declaratively encode a wide range of proof systems with more involved structural and logical inferences rules using linear logic theories with subexponentials. We encode proof systems that have structural restrictions that are much more interesting and challenging than those of the systems specified using linear logic. Besides the multi-conclusion system for intuitionistic logic, we show how to specify proof systems for intuitionistic lax logic [FM97], focused intuitionistic logic *LJQ*, classical modal logic S4 as well as labelled proof systems for different modal logics. These examples provide evidence that linear logic with subexponentials can be successfully used as a framework for a number of proof systems including systems for modal logics.

As in [MP13], we provide sufficient conditions for guaranteeing three properties of systems specified using subexponentials: (1) the admissibility of the cut-rule; (2) the completeness of the system when using only atomic instances of the initial rule; and (3) the invertibility of each inference rule. The main difference from what is presented here and the work using linear logic is the establishment of some criteria for *permutation of rules*. Such analysis is needed for checking whether cuts can be transformed into *principal cuts*. Since our framework enables the encoding of much more complicated proof systems, the behavioral analysis is more involved and it leads to more general conditions when compared to [MP13].

This is a joint work with Vivek Nigam and Giselle Reis.

4 Linear logic, focusing and fixed points in concurrent systems

We now move our attention to linear logic and concurrent systems.

Reasoning about concurrent programs is much harder than reasoning about sequential ones. Programmers often find themselves overwhelmed by the many subtle cases of thread interactions

they must be aware of to decide whether a concurrent program is correct or not, and by the need of finding the right level of thread atomicity of concurrent programs, avoiding race conditions, coping with mutual exclusion requirements, and guaranteeing deadlock freeness to ensure program reliability.

Concurrent Constraint Programming (CCP) [SRP91, Sar93] is a simple and powerful model of concurrency where agents interact by *telling* constraints (i.e., formulas in logic) into a shared *store* of partial information and synchronize by *asking* if a given information can be deduced from the store.

The logical characterization of constraint systems in CCP has allowed the development of more powerful systems by simply replacing the underlying logical framework. For instance, in linear CCP (lcc [FRS01]), constraints are formulas in Girard’s intuitionistic linear logic (ILL) [Gir87] and *ask* agents are allowed to *consume* tokens of information from the store.

The logical foundations of CCP makes it an ideal language for the specification of concurrent systems: complex synchronization patterns can be expressed declaratively by means of constraint entailment. Moreover, the dual view of processes as computing agents and as formulas in logic allows for the use of techniques from both process calculi and logic to reason about the behavior of processes.

The connection between logic and CCP processes (and constraint systems) has been studied since its inception: in [SRP91] a closure operator semantics is given to deterministic CCP programs that was later related to the logic of constraints in [PSSS92]. In [dBGMP97] a calculus for proving properties of CCP programs is defined where properties are expressed in an enriched logic of the constraint system. The works in [SL92, FRS01] relate operational steps of CCP and lcc with derivations in ILL. We can also mention [dBGMP97] that give logical semantics to timed CCP languages and provide calculi to verify temporal properties of programs.

The relation of CCP programs and derivations in logic studied so far, unfortunately, exhibits normally a weak level of adequacy: proofs in the logical systems may not correspond to an operational derivation. This work completely closes this gap by showing that it is possible to exhibit a stronger level of adequacy for different flavors of CCP calculi. In order to do that, we interpret lcc agents as ILL formulas using a *focusing* discipline [And92]. By using focusing, we can classify actions in lcc (and then in CCP) as *positive* or *negative*, depending on the polarity of the outermost connective obtained in their translation as formulas in ILL. The positive actions need to interact with the environment, either for choosing a path to follow, or for waiting for a guard to be available. Negative actions do not need any interaction with the context, and can be executed anytime and concurrently, not altering the final result of the computation. We prove that the translation from lcc to focused intuitionistic linear logic (ILLF) is adequate, in the sense that a focused phase in ILLF corresponds *exactly* to an operational step in lcc, and vice-versa.

Unlike the work in [NOP13], here we do not make use of subexponentials for interpreting lcc processes: the chosen framework is pure linear logic. Hence, the encoding is more natural and direct, and we can use all the already implemented rich meta-theory of linear logic to help in drawing conclusions about CCP systems. Moreover, we study different notions of observables not considered in [NOP13]. Particularly, we show that there are lcc computations that cannot be mimicked by the standard encoding of processes as ILLF formulas. We show that, by introducing *delays* in the encoding, we recover the one-to-one correspondence between ILLF derivations and lcc computations. We also study the behavior of non-deterministic processes with *blind* and *guarded* choices not present in [NOP13].

Finally, we give to procedure calls a more modern presentation, using fixed points. Although

this idea is already present in [SL92], here we exploit better the use of intuitionistic linear logic with fixed points (μ ILL) for adding the greatest fixed point operator to ILL. On doing that, we open the possibility of using co-induction for talking about more involving properties of concurrent system, like for example bisimilarity, liveness properties, or to reason about non-terminating computations.

This is a joint work with Carlos Olarte.

5 Subexponentials, focusing and modalities in concurrent systems

In order to specify the behavior of distributed agents or the policies governing a distributed system, it is often necessary to reason by using different types of modalities, such as time, space, or even the epistemic state of agents.

Following this need, many formalisms have been proposed to specify, program and reason about such policies, e.g., Ambient Calculus, Concurrent Constraint Programming, Authorization Logics, just to name a few.

In order to capture the behavior of distributed systems which take into account spatial, temporal and/or epistemic properties, new formalisms have been proposed. For instance, Saraswat *et al.* proposed *tcc*, which extends CCP with time modalities. Later, Knight *et al.* [KPPV12] proposed a CCP-based language with spatial and epistemic modalities. Some of these developments have also been followed by a similar development in proof theory.

We show that time, spatial, and epistemic modalities can be *uniformly* specified in a single logical framework called $\text{SELLF}^{\mathfrak{M}}$. Our first contribution is the introduction of the proof system $\text{SELL}^{\mathfrak{M}}$, which extends intuitionistic SELL with universal ($\mathfrak{M}$) and existential ($\mathfrak{U}$) quantifiers over subexponentials. It turns out that $\text{SELL}^{\mathfrak{M}}$ has good proof-theoretic properties: it admits cut-elimination and it has a complete focusing discipline, giving rise to the focused system $\text{SELLF}^{\mathfrak{M}}$.

We also show that subexponentials can be interpreted as spatial, epistemic and temporal modalities, thus providing a framework for specifying concurrent systems with these modalities. This is accomplished by encoding in $\text{SELLF}^{\mathfrak{M}}$ different CCP languages, for which the proposed quantifiers play an important role. For instance, they enable the use of an *arbitrary number of subexponentials*, required to model the unbounded nesting of modalities, which is a common feature in epistemic and spatial systems. Finally, the focusing discipline enforces that the obtained encodings are *faithful* w.r.t. CCP's operational semantics in a strong sense: one operational step matches exactly one logical phase. This is the strongest level of adequacy called adequacy on the level of derivations [NM10].

Another important feature of subexponentials is that they can be organized into a pre-order, which specifies the provability relation among them. By coupling subexponential quantifiers with a suitable pre-order, it is possible to specify *declaratively* the rules in which agents can manipulate information. For example, an agent cannot see the information contained in a space that she does not have access to. The boundaries are naturally implied by the pre-order of subexponentials.

This work opens a number of possibilities for specifying the behavior of distributed systems. For instance, unlike [KPPV12], it seems possible in our framework to handle an infinite number of agents. Moreover, we discuss how linearity of constraints can be straightforwardly included to these systems to represent, e.g., agents that can *update/change* the content of the distributed spaces. Also, by changing the underlying subexponential structure, different modalities can be

put in the hands of the modelers and programmers. Finally, all the linear logic meta-theory becomes available for reasoning about distributed systems featuring modalities.

This is a joint work with Carlos Olarte and Vivek Nigam.

6 Modalities and soft concurrent constraint programming

Providing logical and proof theoretic semantics to (fragments of) programming languages not only gives a declarative meaning to these languages, but also normally leads to the development of new features allowing more expressive programming constructions to be used.

We just described how to use linear logic with subexponentials (SELL) for faithfully specifying `eccp` and `sccp`. Now we turn our attention to computing with *soft constraints*.

Soft concurrent constraint programming (Soft-CCP) [BMR06] is an extension of CCP where agents are allowed to tell and ask soft constraints, *i.e.*, constraints with certain level of preference. This allows the modeling of systems with levels of uncertainty, or those mentioning probabilistic or fuzzy entities. However, moving from hard (crisp) constraints to soft constraints was not followed by a corresponding logical/proof theoretic characterization of these systems. This is unfortunate because one of the key motivations of the original CCP was its tight connection to logic and proof theory which enabled the proposal of more advanced systems such as its linear version `lcc` [FRS01]. The main contribution of this part of the talk is to recover this connection by studying the proof theory of soft constraint systems in the form of SELL theories.

A key feature of SELL is that subexponentials are organized into a pre-order, specifying the provability relation among subexponentials. In [NOP13], we used a simple partially ordered set of subexponential names. While this was enough for modeling `eccp` and `sccp`, it is not enough to model soft constraints. In this case, we need more sophisticated algebraic structures.

Hence we investigate the proof theory of SELL with more involved pre-orders and demonstrate that it is possible to characterize *soft constraint systems* by using pre-orders based on semirings. During this investigation, we have also identified variants of soft constraints, namely those based on non-idempotent c-semirings, such as probabilistic soft constraints, that do not seem to have a sensible correspondence in SELL. We thus propose a new proof system, called *SELLS*, with a different promotion rule that allows for such a correspondence.

This is a joint work with Carlos Olarte and Vivek Nigam.

This talk is a collection of results already published in [MP13, NPR14, PON14, NOP13, ONP14], as well as some new unpublished results.

References

References

- [AKZ13] Arnon Avron, Beata Konikowska, and Anna Zamansky. Cut-free sequent calculi for c-systems with generalized finite-valued semantics. *J. Log. Comput.*, 23(3):517–540, 2013.
- [And92] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. of Logic and Computation*, 2(3):297–347, 1992.
- [BMR06] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Soft concurrent constraint programming. *ACM Trans. Comput. Log.*, 7(3):563–589, 2006.

- [CGT08] Agata Ciabattoni, Nikolaos Galatos, and Kazushige Terui. From axioms to analytic rules in nonclassical logics. In *23th Symp. on Logic in Computer Science*, pages 229–240. IEEE Computer Society Press, 2008.
- [dBGMP97] Frank S. de Boer, Maurizio Gabbriellini, Elena Marchiori, and Catuscia Palamidessi. Proving concurrent constraint programs correct. *ACM Trans. Program. Lang. Syst.*, 19(5):685–725, 1997.
- [FD12] Nissim Francez and Roy Dyckhoff. A note on harmony. *Journal of Philosophical Logic*, 41:613 – 628, 2012.
- [FM97] Matt Fairtlough and Michael Mendler. Propositional lax logic. *Information and Computation*, 137(1):1–33, August 1997.
- [Fra14] Nissim Francez. Harmony in multiple-conclusion natural-deduction. *Logica Universalis*, 8(2):215–259, 2014.
- [FRS01] François Fages, Paul Ruet, and Sylvain Soliman. Linear concurrent constraint programming: Operational and phase semantics. *Information and Computation*, 165(1):14–41, 2001.
- [Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
- [KPPV12] Sophia Knight, Catuscia Palamidessi, Prakash Panangaden, and Frank D. Valencia. Spatial and epistemic modalities in constraint-based process calculi. In *CONCUR*, pages 317–332, 2012.
- [MP13] Dale Miller and Elaine Pimentel. A formal framework for specifying sequent calculus proof systems. *Theor. Comput. Sci.*, 474:98–116, 2013.
- [NM88] Gopalan Nadathur and Dale Miller. An Overview of λ Prolog. In *Fifth International Logic Programming Conference*, pages 810–827, Seattle, August 1988. MIT Press.
- [NM10] Vivek Nigam and Dale Miller. A framework for proof systems. *J. of Automated Reasoning*, 45(2):157–188, 2010.
- [NOP13] Vivek Nigam, Carlos Olarte, and Elaine Pimentel. A general proof system for modalities in concurrent constraint programming. In *CONCUR*, pages 410–424, 2013.
- [NPR14] Vivek Nigam, Elaine Pimentel, and Giselle Reis. An extended framework for specifying and reasoning about proof systems. Accepted to a special issue of the Journal of Logic and Computation in honor of Roy Dyckhoff, 2014.
- [NvP11] Sara Negri and Jan von Plato. *Proof Analysis*. Cambridge University Press, 2011.
- [ONP14] Carlos Olarte, Vivek Nigam, and Elaine Pimentel. Dynamic spaces in concurrent constraint programming. In *ENTCS*, 2014.
- [PON14] Elaine Pimentel, Carlos Olarte, and Vivek Nigam. A proof theoretic study of soft concurrent constraint programming. *TPLP*, 2014.
- [PSSS92] Prakash Panangaden, Vijay A. Saraswat, Philip J. Scott, and R. A. G. Seely. A hyperdoctrinal view of concurrent constraint programming. In *REX Workshop*, pages 457–476, 1992.
- [Sar93] Vijay A. Saraswat. *Concurrent Constraint Programming*. MIT Press, 1993.
- [SL92] Vijay Saraswat and Patrick Lincoln. Higher-order Linear Concurrent Constraint Programming. Technical report, 1992.
- [SRP91] V. Saraswat, M. Rinard, and P. Panangaden. Semantic foundations of concurrent constraint programming. In *18th ACM Symp. on Principles of Programming Languages*, pages 333–352, Orlando, FL, January 1991.