From Bunches to Labels and Back in BI Logic

D. Galmiche and M. Marti and D. Méry

Université de Lorraine, CNRS, LORIA
Vandoeuvre-lès-Nancy, F-54506, France

1 Abstract

The ubiquitous notion of resources is a basic one in many fields but has become more and more central in the design and validation of modern computer systems over the past twenty years. Resource management encompasses various kinds of behaviours and interactions including consumption and production, sharing and separation, spatial distribution and mobility, temporal evolution, sequentiality or non-determinism, ownership and access control. Dealing with various aspects of resource management is mostly in the territory of substructural logics, and more precisely, resource logics such as Linear Logic (LL) [5] with its resource consumption interpretation, the logic of Bunched Implications (BI) [8] with its resource sharing interpretation, or order-aware non-commutative logic (NL) [1]. As specification logics, they allow the modelling of features like interactions, resource distribution and mobility, non-determinism, sequentiality or coordination of entities. Separation Logic and its memory model, of which BI is the logical kernel, has gained momentum and proved itself very successful as an assertion language for verifying programs that handle mutable data structures via pointers [6, 9].

From a semantic point of view, resource interactions such as production and consumption, or separation and sharing are handled in resource models at the level of resource composition. For example, various semantics have been proposed to capture the resource sharing interpretation of BI including monoidal, relational or topological resource semantics [4]. From a proof-theoretic and purely syntactical point of view, the subtleties of a particular resource composition usually leads to the definition of distinct sets of connectives (*e.g.*, additive vs multiplicative, commutative vs non-commutative). Capturing the interaction between various kinds of connectives often results in structures more elaborated than set of multi-sets of formulas. For example, the label-free sequent calculus for BI, which is called LBI, admits sequent the left-hand part of which are structured as bunches [7, 8]. Resource interaction is usually much simpler to handle in labelled proof-systems since labels and label constraints are allowed to reflect and mimic, inside the calculus, the fundamental properties of the resource models they are drawn from. For example, various labelled tableaux calculi, all called TBI, have been proposed for the various semantics of BI [4]. A labelled tableaux calculus has been also developed for Separation Logic and its memory model [3].

Our aim is to study the relationships between labelled and label-free proof-systems in BI logic and, more precisely, with the label-free sequent calculus LBI. The relational, topological and monoidal semantics with a Beth interpretation of the additive disjunction have all been proven sound and complete w.r.t. LBI and TBI in [4, 7, 8]. However, the monoidal semantics in which the additive disjunction has the usual Kripke interpretation and which admits explicitly inconsistent resources together with a total (and not partial) resource composition operator has only been proven complete w.r.t. TBI. Its status w.r.t. LBI is not known and still a difficult open problem. Many attempts at solving the problem from a purely semantic point of view have failed over the past fifteen years. Instead we propose a three-step syntactic approach to proving the completeness of the Kripke monoidal semantics of BI that relies on proof translations.

As a first step, we recently proposed a single-conclusioned sequent-style labelled proof-system called GBI, that can be seen as a kind of intermediate calculus between TBI and LBI. GBI shares with TBI the idea of sets of labels and contraints arranged as a resource graph, but the resource graph is partially constructed on the fly using explicit structural rules on labels and constraints rather than being obtained as the result of a closure operator.

The main result in [2] was the definition of an effective (algorithmic) procedure that systematically translates any LBI-proof into a GBI-proof. This translation is not a one-to-one correspondence sending each LBI-rule occurring in the original proof to its corresponding GBI counterpart in the translated proof. Indeed, most of the translations patterns require several additional structural steps to obtain an actual GBI-proof. However, these patterns are such that the rule-application strategy of the original proof will be contained in the translated proof, making our translation structure preserving in that particular sense.

In [2] we also started to investigate how GBI-proofs could relate to LBI-proofs. Taking advantage of the structure preserving property of the translation we gave a reconstruction algorithm that tries to rebuild a LBI-proof of a formula F, from scratch, knowing only the rule-application strategy followed in a given (normal) GBI-proof of F. The completeness of this reconstruction algorithm, *i.e.*, that it might never get stuck, depends on the completeness of the insertion of semi-distributivity steps in the LBI-proof that are meant to fill in the gaps left by the application of structural rules of GBI (in the given GBI-proof) with no LBI counterpart. The completeness of these intermediate semi-distributivity steps was (and still is) only conjectured and far from obvious.

In this paper, we take a second step and further develop our study of how to translate GBI-proofs into LBI-proofs. We first define a kind of tree-like property for GBI labelled sequents. This tree property allows us to translate the left-hand side of a labelled sequent to a bunch according to the label of the formula on its right-hand side. Refining our analysis of the translation given in [2], we show that every sequent in a GBI-proof obtained by translation of an LBI-proof satisfies our tree property. We also show that all GBI rules preserve the tree property from conclusion to premisses except for the rules of contraction and weakening. The main contribution then follows as we define a restriction of GBI, called $\mathrm{GBI}_{\mathrm{tp}}$, in which the only instances of the weakening and contraction rules that are considered as suitable are the ones preserving the tree property and we show that $\mathrm{GBI}_{\mathrm{tp}}$-proofs can effectively and systematically be translated to LBI-proofs. Let us remark that the main result does not depend on a GBI-proof being built from an assembly of LBI translation patterns, *i.e.*, on the fact that a GBI-proof actually corresponds to some translated image of an LBI-proof. We thus observe that our tree-property can serve as a criterion for defining a notion of normal GBI-proofs for which normality also means LBI-translatability.

Ongoing and future work will focus on making the third and final step of showing that total Kripke monoidal models with explicit inconsistency are complete w.r.t. the label-free sequent calculus LBI. Several directions and approaches can be taken to achieve this final goal. A first interesting direction is to find an effective (algorithmic) procedure of translating TBI-proofs into $\mathrm{GBI}_{\mathrm{tp}}$-proofs since TBI is known to be sound and complete w.r.t. total KRMs. This direction is challenging because TBI is a multi-conclusioned system in which generative rules can be used as many times as needed (which avoids backtracking) to saturate the proof-search space and be able to build a countermodel from Hintikka sets in case of non-provability. A second direction relies on the construction of counter-models in the KRM semantics of BI directly from failed $\mathrm{GBI}_{\mathrm{tp}}$-proof attempts. This direction is also challenging as it requires building countermodels from a single-conclusioned proof-system in which backtracking is allowed.

References

[1] M. Abrusci and P. Ruet. Non-commutative logic I : the multiplicative fragment. *Annals of Pure and Applied Logic*, 101:29–64, 2000.

[2] D. Galmiche, M. Marti, and D. Méry. Proof translations in BI logic - extended abstract. In *Int. Workshop on External and Internal Calculi for Non-Classical Logics, EICNCL 2018*, Oxford, UK, July 2018.

[3] D. Galmiche and D. Méry. Tableaux and Resource Graphs for Separation Logic. *Journal of Logic and Computation*, 20(1):189–231, 2010.

[4] D. Galmiche, D. Méry, and D. Pym. The semantics of BI and Resource Tableaux. *Math. Struct. in Comp. Science*, 15(6):1033–1088, 2005.

[5] J.Y. Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102, 1987.

[6] S. Ishtiaq and P. O'Hearn. BI as an assertion language for mutable data structures. In *28th ACM Symposium on Principles of Programming Languages, POPL 2001*, pages 14–26, London, UK, 2001.

[7] D. Pym. On Bunched Predicate Logic. In *14h Symposium on Logic in Computer Science*, pages 183–192, Trento, Italy, July 1999. IEEE Computer Society Press.

[8] D. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*, volume 26 of *Applied Logic Series*. Kluwer Academic Publishers, 2002.

[9] J. Reynolds. Separation logic: A logic for shared mutable data structures. In *IEEE Symposium on Logic in Computer Science*, pages 55–74, Copenhagen, Danemark, July 2002.