The logic of Bunched Implications is undecidable, mechanized in Rocq

Dominique Larchey-Wendling[1,*]

[1]Université de Lorraine, CNRS, LORIA, Vandœuvre-lès-Nancy, France

Abstract

The logic BI of Bunched Implications was introduced by O'Hearn and Pym in the late 90s. It freely combines a multiplicative fragment, that of (exponential free) multiplicative intuitionistic linear logic (MILL), with an additive fragment, that of intuitionistic propositional logic (IL). Considered separately, both MILL and IL are decidable. For 20+ years, BI was believed decidable as well, a belief even backed by several published "proofs" mostly based on semantic considerations. In 2010, the Boolean BI variant of BI (where the additives are those of classical/Boolean propositional logic) was established undecidable by Brotherston and Kanovich on the one hand, and independently by Larchey-Wendling and Galmiche on the other hand. These results did not directly contradict the existing belief about BI, but however shed some doubts and generated a series of tries for alternate (more syntactic) proofs of decidability for BI, which turned out to contain flaws. In a recent breakthrough paper, Galatos, Jipsen, Knudstorp and Ramanayake (2026) prove the undecidability of BI using two different approaches. The first proceeds by many-one reduction from Wang tiling problems. The second approach reduces from two counters And branching Minsky machines acceptance. We adapt that latter proof to the constructive framework of synthetic undecidability, as mechanized in the Coq Library of Undecidability Proofs. Given the checkered history of this decidability question for BI, we think that such strong formal insurances are a worthy contribution.

Keywords

The logic Bunched Implications, undecidability, many-one reductions, Coq/Rocq

1. Introduction

The logic BI of Bunched Implications was introduced by O'Hearn and Pym [1] as the free combination of multiplication intuitionistic linear logic w/o exponential (MILL, the multiplicative part) and propositional intuitionistic logic (IL, the additive part). While the merger of Hilbert style proof systems was straightforward, designing a sequent system enjoying cut-elimination was much less obvious and led to a kind of nested proof system where additive and multiplicative contexts are intertwined into "bunches," algebraically structured into the free merger of two commutative monoids.

As both MILL and IL had decidable provability predicates, it was at first natural to imagine that BI was decidable as well, and this belief was reinforced with later proof arguments [2] leaning towards that property. These arguments were grounded on the properties of (semantic) models generated by tableaux proof search. Other published claims of decidability followed like [3] or [4]. The community was very much convinced of the decidability of BI but, as discussed below, not always bought the arguments used to justify the claims, hence proposing alternate approaches.

The Boolean variant of BI (BBI for short) where the additive part IL is replaced by classical propositional logic (CL) essentially consists in adding the law of excluded middle to the Hilbert calculus, but this breaks the sequent calculus which looses cut-elimination. To recover it, a possible solution is to switch to display calculi [5]. The BBI variant got some specific attention because it can be viewed as the logical foundation of separation logic (SL for short) [6]. While it was initially believed to be "simpler" or "less expressive" than BI itself, [7] proved that (the partial deterministic variant of) BI faithfully embeds into BBI, which somehow came as a surprise. Then in 2010, two independent teams established the

ARQNL 2026, Automated Reasoning in Quantified Non-Classical Logics

*Corresponding author.

dominique.larchey-wendling@loria.fr (D. Larchey-Wendling)

https://members.loria.fr/DLarchey/files/ (D. Larchey-Wendling)

0000-0001-9860-7203 (D. Larchey-Wendling)

undecidability of BBI, as well as its interpretation in various models of SL [8, 9]. Unknown to them, the two teams actually rediscovered a consequence of a former undecidability result for logics with a binary modality, studied before the inception of BI itself [10]. This undedicability of BBI did not directly contradict the decidability claim for BI but, in addition to the above embedding, it did give the feeling that the proof theory of BI and BBI was possibly misunderstood. Hence the later attempts at "better" proofs of decidability, for instance, based on syntactic arguments [11], e.g. terminating proof-search.

Early march 2026, Galatos, Jipsen, Knudstorp and Ramanayake broke the ground with a claim that BI is undecidable after all, disproving earlier decidability claims (including some of their own). This contribution is set to be presented at LICS 2026 and is currently only available as an arXiv preprint [12], but it contains enough material to asses for the quality of the arguments they develop. The present paper intends to certify those via a mechanization of the result in the Rocq proof assistant (formerly called Coq), giving the strongest available formal insurances to the claim.

In [9, 13], we proposed an undecidability proof for BBI (and some variants) that was based on a faithful embedding of the elementary fragment of ILL (eILL) into BBI, via trivial phase semantics. This allowed for the many-one reduction from the problem of Minsky machines (MM) termination (on zero) to eILL provability, hence BBI provability as well, ensuring undecidability. In [14], we mechanize the undecidability of eILL in a Coq framework for synthetic undecidability [15], later to become the foundations of the Coq Library of Undecidability Proofs herein abbreviated as *CLoUP*.

This framework is founded on the idea of *synthetic computability*: by restricting the language in which we write many-one reductions to a suitable constructive framework, those are *automatically computable* maps because all definable functions are computable. Hence, there is no need to implement those reductions in a low-level model of computation (such as e.g. Turing machines) as would formally be required by computability theory, but which is however almost universally avoided in practice, and simply hand-waved away by informal considerations on algorithms.

Because this framework is necessarily constructive, nonconstructive axioms and typically the law of excluded middle (XM) must be avoided and this prevented us to include the many-one reduction from eILL to BBI (a simple faithful embedding) because its correctness involves the soundness of trivial phase semantics (TPS) w.r.t. BBI, and this in turn requires XM.[1] The correctness proof of the faithful embedding of eILL into BBI as proposed in [9, 13] is nonconstructive; we will come back to this issue.

In contrast, constructivity is not an issue when considering eILL because that logic is intuitionistic at heart and hence, TPS is constructively sound for it. In [14] we mechanize a full synthetic reduction chain from the Post correspondence problem (PCP) to eILL via binary stack machines and Minsky machines. Combined with earlier work reducing from Turing machines to PCP [16], this constituted the foundations of what later became the CLoUP.

The recent contribution [12] actually proposes two independent proofs of undecidability of BI, one (extensively developed) many-one reduction from a Wang tiling problem, and another many-one reduction from the acceptance problem for two counters And branching Minsky machines acceptance (2-ACM), a problem already considered for the undecidability proof of linear logic in [17]. It is that latter proof, only presented in the appendix of [12], that we choose to mechanize, for two reasons:

- first, tiling problems are currently unavailable in the CLoUP, and contributing them is quite an endeavor in itself;[2]
- second, 2-ACM are MM with a fork instruction and we already used eILL to simulate fork and encode zero tests with them. Moreover in [18], we gave an inductive presentation of MM acceptance that we could very easily reduce to 2-ACM acceptance.

While we retain the core/key groundbreaking idea of the second proof [12], that is the (relativized) simulation of the combination of contraction and dereliction $!\varphi \dot{\to} !\varphi \mathbin{\dot{*}} \varphi$, we depart from them in that we use a positive encoding of Minsky machines instructions in the $\{\dot{-\!\!*}, \dot{\to}, \dot{\wedge}, \dot{1}\}$ fragment of BI

[1] indeed BBI includes an internal form of XM that leaks at the meta-level in TPS.

[2] all the more that tiling problems usually reduce from Turing machines non-termination (not termination) which is an issue in the constructive framework of synthetic undecidability.

while they use a negative encoding in a somewhat larger fragment that includes the disjunction $\dot{\vee}$ as well.[3] However, their negative encoding involves a quite complicated model, a GBI-algebra they call the Galois algebra whereas we can simply copy the TPS model that we already mechanized for eILL [14]. In addition, the concrete model based on the classical powerset that they discuss involves set theoretic complements (and hence XM), so it is not suitable in the framework of the CLoUP. We have not yet checked whether this is an issue w.r.t. the Galois algebra or not.

1.1. Contents of the paper

The present paper aims at describing how we mechanize in the CLoUP the undecidability proof for the fragments of BI that contain $\{\dot{-\!\!*}, \dot{\to}, \dot{\wedge}, \dot{1}\}$. But it should be readable independently of the Rocq code, with only very basic knowledge of inductive type theory involved. For readers interested in actual code, it has been integrated in the CLoUP since the `rocq-9.2` branch, and we give some pointers to the code along the prose.

In section 2, we start by introducing BI first syntactically with a standard Hilbert style calculus HBI, then with the bunched sequent calculus LBI. Notice that in the case of LBI, we also consider syntactic fragments of the logic, and the availability of the cut-rule (or not) is parameterized. This level of genericity in the syntax arguably involves some technical skills in the Rocq manipulation of dependent types, and we avoid describing those in full details in the prose. We show that HBI and (full) LBI with the cut rule define the same notion of provability. We conclude the section with trivial phase semantics for which we give a soundness proof w.r.t. HBI, and hence LBI.

In section 3 we introduce acceptance problems for MM, and in particular show that two counters And branching Minsky machines (2-ACM) have an undecidable acceptance problem, by reducing from the already mechanized acceptance problem for two counters nondeterministic Minsky machines (ndMM_2).

In section 4, we describe our mechanization of the undecidability proof for BI, by reduction from 2-ACM acceptance. We use the notion of pseudo-exponential to explain what we view as the key contribution of [12], a particular interaction between additive and multiplicative bunches. Pseudo-exponentials allow to simulate the pseudo-dereliction rule $!\varphi \dot{\to} !\varphi \mathbin{\dot{*}} \varphi$, a direct combination of the contraction rule $!\varphi \dot{\to} !\varphi \mathbin{\dot{*}} !\varphi$ and the dereliction rule $!\varphi \dot{\to} \varphi$ of linear logic. We explain how pseudo-dereliction is enough to simulate the storage of MM instructions, storage that allows to freely get copies of stored instructions, and that can be discarded when not needed anymore. Then, we simply adapt our own previous mechanization of the undecidability of elementary intuitionistic linear logic (eILL) [14] to BI replacing exponentials with pseudo-exponentials.

In section 5, we compare our results with those in [12], both in the statements, i.e. which fragments they cover, and in the required meta-theory in the light of constructivity requirements. We conclude with perspectives for extensions of the mechanization to the noncommutative and Boolean cases.

1.2. Some remarks about the Rocq code

We denote by $\mathbb{P}$ the type `Prop` of propositions which are the logical statements in the Coq/Rocq meta-logic. We assume a basic understanding of inductive predicates and types, herein presented as is conventional, either by BNF style grammars, or for more complicated schemes, parameterized rules like e.g. the deduction rules of sequent calculi. We assume Peano natural numbers in the type $\mathbb{N}$ and lists which we denote using the standard $[\,]$ (empty list), :: (head consing) and $+\!\!+$ (append) operators.

The mechanization that we provide is available starting from the `rocq-9.2` branch (the current main branch)

https://github.com/uds-psl/coq-library-undecidability/tree/rocq-9.2

and will be available in future versions of the CLoUP as well. The files that we contribute are located in the theories/BI and theories/MinskyMachines/ACM2 sub-directories and most of them are referenced in the paper, hyperlinked with the CLoUP repository. Concerning the size of the contribution in terms

[3] To be fair, their proof also applies to noncommutative BI, a case that we do not consider herein.

of lines of code (loc), we get 300 loc for the undecidability of 2-ACM by reduction for ndMM_2. The reduction from 2-ACM to BI is a somewhat larger contribution totaling around 1650 loc.

2. Calculi and Semantics for BI

Let us start with fixed type denoted `prop` for object-level *propositional variables*, not to be confused with meta-level propositions in $\mathtt{Prop}/\mathbb{P}$. Typically we need `prop` large enough (or infinite like the type of natural numbers $\mathbb{N}$) to ensure undecidability but the description of the BI logic can be performed on a generic type `prop` with no structure a priori.

We define the syntax of BI based on the `prop` parameter. We denote by $\{*, +\}$ the (finite) enumerated type of BI kinds, i.e. multiplicative and additive, and by $k : \{*, +\}$ a value for dealing with multiplicative/$*$ or additive/$+$ constructs in a uniform way. The type of BI formulae is inductive and can be described by the following BNF style definition:

$$\mathtt{Form}: \quad \varphi, \psi, \gamma ::= x \mid c \mid \varphi \odot \psi \quad \text{where } x : \mathtt{prop},\ c : \{\dot{1}, \dot{\top}, \dot{\bot}\} \text{ and } \odot : \{\dot{*}, \dot{-\!\!*}, \dot{\wedge}, \dot{\to}, \dot{\vee}\}$$

and we denote by e.g. φ an arbitrary BI formula. Notice that we decorate object-level logical connectives with a dot to avoid confusion with meta-level (Rocq) connectives,[4] which typically applies to intuitionistic connectives $\dot{\wedge}$, $\dot{\to}$ or $\dot{\vee}$. As usual, the connectives $\dot{-\!\!*}$ and $\dot{\to}$ are right associative, with less precedence than $\dot{*}$, $\dot{\wedge}$, $\dot{\vee}$ which are left associative. We are now in position to introduce calculi for provability in BI.

2.1. Hilbert style calculus for BI

We start with a Hilbert calculus for BI (HBI) in the spirit of [19], which requires much less machinery than the upcoming bunched sequent calculus LBI. It consists in merging the axioms of IL with those of MILL and adding three deduction rules to the standard modus ponens rule. We define a predicate $\vdash_{IL} : \mathtt{Form} \to \mathbb{P}$ describing a set of axioms for intuitionistic logic corresponding to the additive fragment of BI, using the rules (free of any premise):

$$\frac{}{\vdash_{IL} \varphi \dot{\to} \psi \dot{\to} \varphi} \qquad \frac{}{\vdash_{IL} (\varphi \dot{\to} \psi \dot{\to} \gamma) \dot{\to} (\varphi \dot{\to} \psi) \dot{\to} \varphi \dot{\to} \gamma} \qquad \frac{}{\vdash_{IL} \varphi \,\dot{\wedge}\, \psi \dot{\to} \varphi} \qquad \frac{}{\vdash_{IL} \varphi \,\dot{\wedge}\, \psi \dot{\to} \psi} \qquad \frac{}{\vdash_{IL} \dot{\bot} \dot{\to} \varphi}$$

$$\frac{}{\vdash_{IL} \varphi \dot{\to} \psi \dot{\to} \varphi \,\dot{\wedge}\, \psi} \qquad \frac{}{\vdash_{IL} \varphi \dot{\to} \varphi \,\dot{\vee}\, \psi} \qquad \frac{}{\vdash_{IL} \psi \dot{\to} \varphi \,\dot{\vee}\, \psi} \qquad \frac{}{\vdash_{IL} (\varphi \dot{\to} \gamma) \dot{\to} (\psi \dot{\to} \gamma) \dot{\to} \varphi \,\dot{\vee}\, \psi \dot{\to} \gamma} \qquad \frac{}{\vdash_{IL} \dot{\top}}$$

to be understood as $\vdash_{IL} \delta$ holds if and only if δ is (an instance of) one of the above shapes. Notice that while the number of shapes is finite, the collection of its instance in $\vdash_{IL}$ is infinite and closed under substitutions.

Then we give a complementary set of axioms for the multiplicative fragment of BI as a predicate $\vdash_{BI} : \mathtt{Form} \to \mathbb{P}$ using the following rules:

$$\frac{}{\vdash_{BI} \varphi \dot{\to} \dot{1} \dot{*} \varphi} \qquad \frac{}{\vdash_{BI} \dot{1} \dot{*} \varphi \dot{\to} \varphi} \qquad \frac{}{\vdash_{BI} \varphi \dot{*} \psi \dot{\to} \psi \dot{*} \varphi} \qquad \frac{}{\vdash_{BI} \varphi \dot{*} (\psi \dot{*} \gamma) \dot{\to} (\varphi \dot{*} \psi) \dot{*} \gamma}$$

We finish with the deduction rules for HBI encoded in a binary relation $\vdash_H : (\mathtt{Form} \to \mathbb{P}) \to \mathtt{Form} \to \mathbb{P}$. Assuming a set of axioms described by $\Psi : \mathtt{Form} \to \mathbb{P}$,[5] we characterize the formulae φ derivable from the axiom in Ψ as a predicate $\Psi \vdash_H \varphi$ using the following inductive rules:

$$\frac{\Psi\, \varphi}{\Psi \vdash_H \varphi}\ [\mathrm{ax}] \qquad \frac{\Psi \vdash_H \varphi \qquad \Psi \vdash_H \varphi \dot{\to} \psi}{\Psi \vdash_H \psi}\ [\mathrm{mp}] \qquad \frac{\Psi \vdash_H \varphi \dot{\to} \psi \qquad \Psi \vdash_H \gamma \dot{\to} \delta}{\Psi \vdash_H \varphi \dot{*} \gamma \dot{\to} \psi \dot{*} \delta}\ [\mathrm{mult}]$$

$$\frac{\Psi \vdash_H \varphi \dot{\to} \psi \dot{-\!\!*} \gamma}{\Psi \vdash_H \varphi \dot{*} \psi \dot{\to} \gamma}\ [\mathrm{wand}_1] \qquad \frac{\Psi \vdash_H \varphi \dot{*} \psi \dot{\to} \gamma}{\Psi \vdash_H \varphi \dot{\to} \psi \dot{-\!\!*} \gamma}\ [\mathrm{wand}_2]$$

[4] the dotted notation has a low footprint on paper but is not generally available in Rocq hence object-level notations in the code might differ a bit from those in the paper.

[5] typically we below instantiate Ψ as $\vdash_{IL} \cup \vdash_{BI} := \lambda\, \varphi, \vdash_{IL} \varphi \vee \vdash_{BI} \varphi$, the union of additive and multiplicative axioms.

Notice that the deduction relation $\Psi \vdash_H \varphi$ does not satisfy the "deduction theorem," i.e. $\Psi \cup \{\varphi\} \vdash_H \psi$ entails $\Psi \vdash_H \varphi \dot{\to} \psi$, hence the need of bunches in the sequent calculus for BI. However this property holds when restricted to the IL fragment and we make use of this facility in the Rocq code (see e.g. files hil.v and hbi.v). Reasoning directly using the Hilbert rules of $\vdash_H$ and the axioms in $\vdash_{IL} \cup \vdash_{BI}$ is quite hard. Natural deduction like facilities enhance the Rocq user experience a lot.

Definition 2.1 ($\vdash_{HBI}$). *The set of provable formulae in HBI, denoted* $\vdash_{HBI} : \mathtt{Form} \to \mathbb{P}$ *are those derivable from the union of IL and BI axioms, i.e.* $\vdash_{HBI} \varphi := (\vdash_{IL} \cup \vdash_{BI}) \vdash_H \varphi$. *See also* BI_HILBERT_PROVABLE *in the file BI.v.*

2.2. Bunched sequent calculus for BI

We give a detailed description of the bunched sequent calculus LBI for BI following e.g. the presentations of [20, 2, 12]. These three presentations slightly differ but thanks to structural rules (equivalence, weakening and contraction), they define the same provability relation, and so does the one we propose.[6]

We define the type Bunch of bunches which are structured collections of BI formulae. Algebraically, bunches are the superposition of two commutative monoids freely generated by Form. Formally:

$$\mathtt{Bunch}: \quad \Gamma, \Delta, \Theta ::= \langle \varphi \rangle \mid \varnothing_k \mid \Gamma \otimes_k \Delta \quad \text{where } \varphi : \mathtt{Form} \text{ and } k : \{*, +\}$$

and we capture the monoidal structures using a congruence $\equiv : \mathtt{Bunch} \to \mathtt{Bunch} \to \mathbb{P}$ to be defined below. We write $\circledast$ (resp. $\oplus$) instead of $\otimes_*$ (resp. $\otimes_+$) for compactness. Notice that $\circledast$ (resp. $\oplus$) is usually denoted with a comma , (resp. a semicolon ;) in the BI literature, but we adopt notations closer to the mechanized code.[7] Remark that Form is not included in Bunch as a type but there is a canonical embedding $\langle \cdot \rangle : \mathtt{Form} \to \mathtt{Bunch}$ that is usually implicitly assumed in the BI literature.

We define $\equiv : \mathtt{Bunch} \to \mathtt{Bunch} \to \mathbb{P}$ as the smallest congruence (for both $\circledast$ and $\oplus$) that makes both $(\mathtt{Bunch}, \circledast, \varnothing_*)$ and $(\mathtt{Bunch}, \oplus, \varnothing_+)$ commutative monoids (up to the equivalence relation $\equiv$), hence with the rules:

$$\frac{}{\Gamma \equiv \Gamma}\ [\text{refl}] \qquad \frac{\Gamma \equiv \Delta}{\Delta \equiv \Gamma}\ [\text{sym}] \qquad \frac{\Gamma \equiv \Delta \qquad \Delta \equiv \Theta}{\Gamma \equiv \Theta}\ [\text{trans}] \qquad \frac{\Delta \equiv \Theta}{\Gamma \otimes_k \Delta \equiv \Gamma \otimes_k \Theta}\ [\text{congr}]$$

$$\frac{}{\varnothing_k \otimes_k \Gamma \equiv \Gamma}\ [\text{neut}] \qquad \frac{}{\Gamma \otimes_k \Delta \equiv \Delta \otimes_k \Gamma}\ [\text{comm}] \qquad \frac{}{(\Gamma \otimes_k \Delta) \otimes_k \Theta \equiv \Gamma \otimes_k (\Delta \otimes_k \Theta)}\ [\text{assoc}]$$

where the parameter $k : \{*, +\}$ allows to unify rules which are generic in the $*/+$ alternative.

To complete our description of the machinery used in the bunched sequent calculus, we introduce the notion of context. These are usually described informally as a "bunch with a single hole," hole herein represented by $\square$. We encode these as in [21] viewing the type Ctx as that of paths (sequence of left/right choices) that locate the hole:

$$\mathtt{Ctx}: \quad \Sigma ::= \square \mid \Delta \otimes_k \Sigma \mid \Sigma \otimes_k \Delta \quad \text{where } \Delta : \mathtt{Bunch} \text{ and } k : \{*, +\}$$

Holes are the places where substitutions occur. For $\Sigma : \mathtt{Ctx}$ and $\Gamma : \mathtt{Bunch}$, we define $\Sigma[\Gamma] : \mathtt{Bunch}$ as the in-place replacement of $\square$ by Γ in Σ, using fixpoint equations on the inductive structure of Σ:

$$\square[\Gamma] := \Gamma \qquad (\Delta \otimes_k \Sigma)[\Gamma] := \Delta \otimes_k \Sigma[\Gamma] \qquad (\Sigma \otimes_k \Delta)[\Gamma] := \Sigma[\Gamma] \otimes_k \Delta.$$

Since $\equiv$ is a congruence, we show that $\equiv$ is closed under context substitutions $\Sigma[\cdot]$, i.e. $\Gamma \equiv \Delta \to \Sigma[\Gamma] \equiv \Sigma[\Delta]$, and this is proved by structural induction on Σ.

We can now describe the provability predicate $\vdash_L : \mathtt{Bunch} \to \mathtt{Form} \to \mathbb{P}$ in the bunched sequent calculus LBI as a binary relation between a bunch (structured hypotheses) and a single formula (conclusion),

[6] We have not mechanized the equivalence between those different presentations but we view this task as a simple exercise.
[7] Using commas or semicolons in Rocq would conflict with several other standard uses of these symbols.

written infix and characterized using the following inductive rules:

$$\frac{}{\langle\varphi\rangle \vdash_L \varphi}\ [\mathrm{id}] \qquad \frac{\Gamma \vdash_L \varphi \qquad \Sigma[\langle\varphi\rangle] \vdash_L \psi}{\Sigma[\Gamma] \vdash_L \psi}\ [\mathrm{cut}] \qquad \frac{\Gamma \equiv \Delta \qquad \Gamma \vdash_L \varphi}{\Delta \vdash_L \varphi}\ [\mathrm{equiv}]$$

$$\frac{\Sigma[\varnothing_+] \vdash_L \varphi}{\Sigma[\Gamma] \vdash_L \varphi}\ [\mathrm{weak}] \qquad \frac{\Sigma[\Gamma \oplus \Gamma] \vdash_L \varphi}{\Sigma[\Gamma] \vdash_L \varphi}\ [\mathrm{cntr}]$$

$$\frac{\Sigma[\varnothing_*] \vdash_L \varphi}{\Sigma[\langle \dot{\mathrm{i}} \rangle] \vdash_L \varphi}\ [\dot{\mathrm{i}}_l] \qquad \frac{}{\varnothing_* \vdash_L \dot{\mathrm{i}}}\ [\dot{\mathrm{i}}_r] \qquad \frac{\Sigma[\varnothing_+] \vdash_L \varphi}{\Sigma[\langle \dot{\top} \rangle] \vdash_L \varphi}\ [\dot{\top}_l] \qquad \frac{}{\varnothing_+ \vdash_L \dot{\top}}\ [\dot{\top}_r] \qquad \frac{}{\Sigma[\langle \dot{\bot} \rangle] \vdash_L \varphi}\ [\dot{\bot}_l]$$

$$\frac{\Sigma[\langle\varphi\rangle \circledast \langle\psi\rangle] \vdash_L \gamma}{\Sigma[\langle\varphi \mathbin{\dot{*}} \psi\rangle] \vdash_L \gamma}\ [\dot{*}_l] \qquad \frac{\Gamma \vdash_L \varphi \qquad \Delta \vdash_L \psi}{\Gamma \circledast \Delta \vdash_L \varphi \mathbin{\dot{*}} \psi}\ [\dot{*}_r] \qquad \frac{\Delta \vdash_L \varphi \qquad \Sigma[\langle\psi\rangle] \vdash_L \gamma}{\Sigma[\Delta \circledast \langle\varphi \mathbin{\dot{-\!\!*}} \psi\rangle] \vdash_L \gamma}\ [\dot{-\!\!*}_l] \qquad \frac{\Gamma \circledast \langle\varphi\rangle \vdash_L \psi}{\Gamma \vdash_L \varphi \mathbin{\dot{-\!\!*}} \psi}\ [\dot{-\!\!*}_r]$$

$$\frac{\Sigma[\langle\varphi\rangle \oplus \langle\psi\rangle] \vdash_L \gamma}{\Sigma[\langle\varphi \mathbin{\dot{\wedge}} \psi\rangle] \vdash_L \gamma}\ [\dot{\wedge}_l] \qquad \frac{\Gamma \vdash_L \varphi \qquad \Delta \vdash_L \psi}{\Gamma \oplus \Delta \vdash_L \varphi \mathbin{\dot{\wedge}} \psi}\ [\dot{\wedge}_r] \qquad \frac{\Delta \vdash_L \varphi \qquad \Sigma[\langle\psi\rangle] \vdash_L \gamma}{\Sigma[\Delta \oplus \langle\varphi \mathbin{\dot{\to}} \psi\rangle] \vdash_L \gamma}\ [\dot{\to}_l] \qquad \frac{\Gamma \oplus \langle\varphi\rangle \vdash_L \psi}{\Gamma \vdash_L \varphi \mathbin{\dot{\to}} \psi}\ [\dot{\to}_r]$$

$$\frac{\Sigma[\langle\varphi\rangle] \vdash_L \gamma \qquad \Sigma[\langle\psi\rangle] \vdash_L \gamma}{\Sigma[\langle\varphi \mathbin{\dot{\vee}} \psi\rangle] \vdash_L \gamma}\ [\dot{\vee}_l] \qquad \frac{\Gamma \vdash_L \varphi}{\Gamma \vdash_L \varphi \mathbin{\dot{\vee}} \psi}\ [\dot{\vee}^1_r] \qquad \frac{\Gamma \vdash_L \psi}{\Gamma \vdash_L \varphi \mathbin{\dot{\vee}} \psi}\ [\dot{\vee}^2_r]$$

Notice that the weakening rule [weak] is different from that of [12] but they are inter-derivable thanks to the availability of the equivalence rule [equiv] because $\Delta \equiv \varnothing_+ \oplus \Delta \equiv \Delta \oplus \varnothing_+$ holds and $\equiv$ is closed under context substitutions $\Sigma[\cdot]$. Such differences are common in the BI literature: for instance the left disjunction $[\dot{\vee}_l]$ and left implication rules $[\dot{-\!\!*}_l]$ and $[\dot{\to}_l]$ have slightly different flavors in [20], themselves differing from those in [2], but again these are minor differences that do not modify the provability predicate, thanks to the structural rules [equiv], [weak] and [cntr] only.

Definition 2.2 ($\vdash_{LBI}$). *The set of formulae provable in the bunched sequent calculus LBI, denoted* $\vdash_{LBI} : \mathtt{Form} \to \mathbb{P}$*, are those derivable from the empty context* $\varnothing_+$*, i.e.* $\vdash_{LBI} \varphi := \varnothing_+ \vdash_L \varphi$*. See also* `BI_SEQ_PROVABLE` *in the file BI.v.*

Theorem 2.1 (Soundness for LBI). *The LBI bunched sequent calculus is sound w.r.t. the HBI Hilbert calculus, i.e.* $\vdash_{LBI} \subseteq \vdash_{HBI}$*. See* `LBI_full_to_HBI_form` *in file hbi.v.*

Proof. We translate a bunch like e.g. Γ into a formula written $\langle\!\langle \Gamma \rangle\!\rangle$ using the following fixpoint equations:

$$\langle\!\langle \langle\varphi\rangle \rangle\!\rangle := \varphi \quad \langle\!\langle \varnothing_* \rangle\!\rangle := \dot{\mathrm{i}} \quad \langle\!\langle \varnothing_+ \rangle\!\rangle := \dot{\top} \quad \langle\!\langle \Gamma \circledast \Delta \rangle\!\rangle := \langle\!\langle \Gamma \rangle\!\rangle \mathbin{\dot{*}} \langle\!\langle \Delta \rangle\!\rangle \quad \langle\!\langle \Gamma \oplus \Delta \rangle\!\rangle := \langle\!\langle \Gamma \rangle\!\rangle \mathbin{\dot{\wedge}} \langle\!\langle \Delta \rangle\!\rangle$$

and then we show that $\Gamma \vdash_L \varphi$ entails $\vdash_{HBI} \langle\!\langle \Gamma \rangle\!\rangle \mathbin{\dot{\to}} \varphi$ by induction on (the proof of) $\Gamma \vdash_L \varphi$. Lastly, we instantiate $\Gamma := \varnothing_+$ and derive the result. □

Theorem 2.2 (Completeness for LBI). *The HBI Hilbert calculus is sound w.r.t. the (full) LBI bunched sequent calculus (with cut), i.e.* $\vdash_{HBI} \subseteq \vdash_{LBI}$*. See* `HBI_to_LBI_full` *in file hbi.v.*

Proof. We show that $(\vdash_{IL} \cup \vdash_{BI}) \vdash_H \varphi$ entails $\varnothing_+ \vdash_L \varphi$ by induction on the derivation of $(\vdash_{IL} \cup \vdash_{BI}) \vdash_H \varphi$. Hence it is enough to give LBI proofs of HBI axioms (rule [ax]) and the LBI admissibility of the four other derivations rules [mp], [mult], [wand$_1$] and [wand$_2$] of HBI. This proof makes heavy use of the [cut] rule and if one wants the result without the cut rule, then a proof of cut-elimination such as [21] would be required. □

Notice that we do not need $\vdash_{HBI} \subseteq \vdash_{LBI}$ for our undecidability results. Nonetheless, this equivalence between $\vdash_{HBI}$ and $\vdash_{LBI}$ ensures that we did not make any mistake in the formal transcription of the somewhat complex bunched calculus in Rocq, assuming that the Hilbert system itself is properly transcribed, which is arguably far easier to check for the reader.

2.3. Fragments of LBI

In this section we describe a refinement of the syntax of BI and of the bunched sequent calculus LBI that allows us to formally consider syntactic fragments of BI. We want to be able to restrict the set of usable connectives. To implement that, we parameterize the logic by a Boolean selector function $\mu : \{\dot{\mathrm{I}}, \dot{\top}, \dot{\bot}, \dot{*}, \dot{-\!\!*}, \dot{\wedge}, \dot{\to}, \dot{\vee}\} \to \{\mathtt{true}, \mathtt{false}\}$ that, when e.g. $\mu(\dot{*}) = \mathtt{false}$ forbids the usage of the $\dot{*}$ connective, and when $\mu(\dot{-\!\!*}) = \mathtt{true}$ allows the usage of the $\dot{-\!\!*}$ connective.

The presentation of Sections 2.1 and 2.2 corresponds to the situation where $\mu = \lambda\_, \mathtt{true}$, i.e. all connectives are allowed. Fragments in the Hilbert calculus HBI are not very workable because its axioms and rules rely on the availability of some connectives like $\dot{\to}$, $\dot{*}$ and $\dot{-\!\!*}$, which hence cannot really be forbidden. So we only consider the full fragment of HBI in the Rocq implementation.

However, the sequent calculus LBI allows for the separate treatment of each connective, hence the actual implementation of `Form`, `Bunch`, `Ctx`, $\equiv$, $\vdash_L$ and $\vdash_{LBI}$ are parameterized by μ. Notice that while the cut-elimination theorem for LBI [21] ensures that fragments are conservative over each other, this result is not mandatory in the proof of undecidability herein described.

While the types `Bunch`, `Ctx`, $\equiv$ depend on μ just because they are built from `Form`, provability $\vdash_L$ in LBI also excludes left/right introduction rules of the connectives forbidden by the selector μ. In addition, a specific Boolean parameter $\mathtt{cut} : \{\mathtt{with\_cut}, \mathtt{cut\_free}\}$ allows for the inclusion or exclusion of the [cut] rule from those of $\vdash_L$, so we can deal with cut-free LBI or LBI with cut in a unified way.

We avoid entering in the Rocq implementation details of these fragments as dependent types in the prose, but we mention this because we want to state results which are relative to particular fragments of LBI, typically the $\{\dot{\mathrm{I}}, \dot{-\!\!*}, \dot{\to}, \dot{\wedge}\}$-fragment of LBI that is enough for the encoding of 2-ACM.

Proposition 2.1 (Monotonicity for LBI fragments). *If a statement $\Gamma \vdash_L \varphi$ can be proved in a LBI fragment $(\mu, \mathtt{cut})$, then it can also be proved in a larger LBI fragment $(\mu', \mathtt{cut}')$, larger meaning $\mathtt{cut} = \mathtt{with\_cut} \to \mathtt{cut}' = \mathtt{with\_cut}$ and $\forall c,\ \mu(c) = \mathtt{true} \to \mu'(c) = \mathtt{true}$. See* `LBI_map_sound` *in the file utils.v.*

Proof. The relation between μ and μ' and `cut` and $\mathtt{cut}'$ ensures that any proof of $\Gamma \vdash_L \varphi$ in the μ fragment directly maps into a proof of $\Gamma \vdash_L \varphi$ in the μ' fragment. □

2.4. Trivial Phase Semantics for BI

Trivial phase semantics (TPS) is the phase semantics of intuitionistic linear logic (ILL) where the closure operator is the identity map. In [9, 13], we show that it is sound for ILL and complete for the elementary fragment of ILL (eILL) in which one can encode termination (on zero) for Minsky machines. We mechanize that result in [14]. Acceptance for either ndMM_2 or 2-ACM (see Definition 3.1) can also be encoded in eILL but we did not establish those results in [14]. In [9, 13] we also show that TPS is sound for Boolean BI (BBI), hence it is a fortiori sound for BI. We give a constructive proof of that latter soundness result for BI below.

We assume a commutative monoid $(M, \circ, \epsilon)$ and we extend the interpretation of $\circ$ to "subsets" of M (i.e. predicates of type $M \to \mathbb{P}$) and define its right adjoint $\multimap$ by:

$$X \circ Y := \lambda z, \exists x\, y,\ z = x \circ y \wedge X\, x \wedge Y\, y \qquad\qquad X \multimap Y := \lambda z, \forall x,\ X\, x \to Y(x \circ z)$$

for any $X, Y : M \to \mathbb{P}$.[8] As usual the binary intersection and binary union of X and Y can be implemented respectively as $X \cap Y := \lambda z, X\, z \wedge Y\, z$ and $X \cup Y := \lambda z, X\, z \vee Y\, z$.

Definition 2.3 (Trivial Phase Semantics). *Given an interpretation $[\![\cdot]\!] : \mathtt{prop} \to M \to \mathbb{P}$ of propositional variables in $M \to \mathbb{P}$, we* extend *this interpretation to BI formulae in* `Form` *using the equations:*

$$[\![\varphi \mathbin{\dot\wedge} \psi]\!] := [\![\varphi]\!] \cap [\![\psi]\!] \qquad [\![\varphi \mathbin{\dot\to} \psi]\!] := \lambda x, [\![\varphi]\!]\, x \to [\![\psi]\!]\, x \qquad [\![\varphi \mathbin{\dot\vee} \psi]\!] := [\![\varphi]\!] \cup [\![\psi]\!] \qquad [\![\dot\bot]\!] := \lambda \_, \mathtt{False}$$
$$[\![\varphi \mathbin{\dot*} \psi]\!] := [\![\varphi]\!] \circ [\![\psi]\!] \qquad [\![\varphi \mathbin{\dot{-\!\!*}} \psi]\!] := [\![\varphi]\!] \multimap [\![\psi]\!] \qquad [\![\dot{\mathrm{I}}]\!] := \lambda x, \epsilon = x \qquad [\![\dot\top]\!] := \lambda \_, \mathtt{True}$$

[8] Notice that in regular phase semantics, a stable closure must be further applied to $X \circ Y$ to interpret the linear tensor $\otimes$ but since the closure is the identity map in TPS, this can be ignored here.

Theorem 2.3 (Soundness of TPS for HBI). *For any* φ : `Form` *in the full fragment (i.e. for* $\mu := \lambda\_, \texttt{true}$*), if* $\vdash_{HBI} \varphi$ *then* $\forall x,\ [\![\varphi]\!]\, x$ *holds. See* `tps_HBI_sound` *in the file tps.v.*

Proof. By induction on the derivation of $(\vdash_{IL} \cup \vdash_{BI}) \vdash_H \varphi$. Hence we show that $\forall x,\ [\![\varphi]\!]\, x$ holds for any axiom ($\vdash_{IL} \varphi$ or $\vdash_{BI} \varphi$), and that the property is closed for the four derivations rules [mp], [mult], [wand$_1$] and [wand$_2$]. □

Using the translation of bunches into formulae (see proof of Theorem 2.1), we can extend the TPS to bunches with $[\![\Gamma]\!] := [\![\langle\!\langle \Gamma \rangle\!\rangle]\!]$ and, as a combination of Proposition 2.1 and Theorem 2.3, we derive that $\Gamma \vdash_L \varphi \to [\![\Gamma]\!] \subseteq [\![\varphi]\!]$ holds in any fragment μ of BI. But we don't need this result for undecidability.

Notice that the soundness proof for HBI is performed constructively. TPS is neither complete for ILL nor for BI, and nor for BBI. While TPS is proved sound for BBI in [9, 13], that proof however requires excluded middle (XM). Indeed XM is an axiom of classical logic (CL) and thus belongs to BBI, of which the additive part is precisely CL. This mean that the interpretation of $[\![\varphi \mathbin{\dot\vee} (\varphi \mathbin{\dot\to} \dot\bot)]\!]\, x$ is $[\![\varphi]\!]\, x \vee ([\![\varphi]\!]\, x \to \texttt{False})$ which cannot be established constructively in general, unless of course assuming XM at the meta-level of Rocq. That is why the reduction from eILL to BBI and the subsequent undecidability proof of BBI [9, 13] has not been integrated in the framework of synthetic undecidability, because reductions have to be implemented and proved correct constructively.

3. Two counters And branching Minsky machines

We describe the acceptance problem for two counters And branching Minsky machines (2-ACM) [17] and explain its undecidability in the synthetic framework of the CLoUP.

The CLoUP library is composed of a collection of many-one reductions between problems of different nature, such as termination problems for low-level computational models like Turing machines (TM) or Minsky machines (MM), string related problems like the Post correspondence problem (PCP), logical problems like satisfiability or provability, etc. When considering adding a new problem, we first have to choose one seed of undecidability from which to reduce from.

In the case of (exponential free) intuitionistic linear logic (ILL) [14], we added a reduction chain from PCP to MM and then reduced provability in ILL from MM termination. We could use MM here as well but following [12], we rather directly contribute 2-ACM acceptance to the CLoUP and then reduce from that problem. Fortunately we already contributed the acceptance problem for ndMM$_2$ (see Definition 23 in [18]) which is close enough to the acceptance problem for 2-ACM.

We only consider acceptance problems hence we can represent those with deduction rules which facilitate the implementation of reductions to logical deduction systems like LBI or HBI. The Minsky machines we consider have two counters that we name α and β, and $a : \mathbb{N}$ (resp. $b : \mathbb{N}$) is used to represent the natural number value of α (resp. β) at a given state in the deduction. The state is also composed of a current location in a type `loc`. The instructions of MM are of the following kinds:

`INC` $\alpha\, p\, q$**:** at location p, increment the value a of α to $1{+}a$ and jump to location q;

`DEC` $\beta\, p\, q$**:** at location p, if the value of β is not zero, decrement it by one and jump to location q;

`ZERO` $\alpha\, p\, q$**:** at location p, if the value of α is 0 then jump to location q w/o altering the counters;

`STOP` p**:** at location p, if the counters both have a 0 value then accept the computation;

`FORK` $p\, q\, r$**:** at location p, irrelevant of the values of counters, fork the computation and jump to q and r in two separate branches of the computation w/o altering the counters.

Their above informal description is formalized in the rules of Fig. 1.

$$\frac{}{\Sigma /\!/ 0 \parallel 0 \vdash p}\ [\texttt{STOP}\, p \in \Sigma] \qquad \frac{\Sigma /\!/ a \parallel b \vdash q \qquad \Sigma /\!/ a \parallel b \vdash r}{\Sigma /\!/ a \parallel b \vdash p}\ [\texttt{FORK}\, p\, q\, r \in \Sigma]$$

$$\frac{\Sigma /\!/ 1{+}a \parallel b \vdash q}{\Sigma /\!/ a \parallel b \vdash p}\ [\texttt{INC}\, \alpha\, p\, q \in \Sigma] \qquad \frac{\Sigma /\!/ a \parallel 1{+}b \vdash q}{\Sigma /\!/ a \parallel b \vdash p}\ [\texttt{INC}\, \beta\, p\, q \in \Sigma]$$

$$\frac{\Sigma /\!/ a \parallel b \vdash q}{\Sigma /\!/ 1{+}a \parallel b \vdash p}\ [\texttt{DEC}\, \alpha\, p\, q \in \Sigma] \qquad \frac{\Sigma /\!/ a \parallel b \vdash q}{\Sigma /\!/ a \parallel 1{+}b \vdash p}\ [\texttt{DEC}\, \beta\, p\, q \in \Sigma]$$

$$\frac{\Sigma /\!/ 0 \parallel b \vdash q}{\Sigma /\!/ 0 \parallel b \vdash p}\ [\texttt{ZERO}\, \alpha\, p\, q \in \Sigma] \qquad \frac{\Sigma /\!/ a \parallel 0 \vdash q}{\Sigma /\!/ a \parallel 0 \vdash p}\ [\texttt{ZERO}\, \beta\, p\, q \in \Sigma]$$

Figure 1: Rules for acceptance problems of two counters Minsky machines.

Definition 3.1 (Two counters Minsky machines acceptance problems). Non-deterministic two counters Minsky machines (ndMM_2) *have instructions in* $\{\texttt{STOP}, \texttt{INC}, \texttt{DEC}, \texttt{ZERO}\}$. Two counters And branching Minsky machines (2-ACM) *have instructions in* $\{\texttt{STOP}, \texttt{INC}, \texttt{DEC}, \texttt{FORK}\}$. *Considering a list of instructions* Σ *(of the corresponding type), we say that the* ndMM_2 program Σ accepts state $(a, b) : \mathbb{N} \times \mathbb{N}$ at location $p : \texttt{loc}$ *and write* $\Sigma /\!/_n a \parallel b \vdash p : \mathbb{P}$ *if this statement is derivable from the rules of Fig. 1 (hence excluding* FORK*). Similarly, we write* $\Sigma /\!/_A a \parallel b \vdash p : \mathbb{P}$ *for acceptance of a 2-ACM program* Σ *(hence excluding* ZERO*). See the files ndMM2.v and ACM2.v.*

Notice that the rules of Fig. 1 should be read upwards if one wants to interpret them in a computational way: the conclusion is the current state and the premises are the next states (the plural only applies in the case of the FORK rule). In contrast with [12], we use an inductive predicate in the deduction rules of Fig. 1 to implement acceptance directly. This avoids us the burden of having to describe a step semantics, and hence a richer state of computation for Minsky machines. In [12], in the case of 2-ACM, it takes the shape of a binary tree, that is herein captured in the structure of the derivation of $\Sigma /\!/_A a \parallel b \vdash p$ statements.

Theorem 3.1. *Over a (shared) type* loc *of locations, there is a many-one reduction from ndMM*$_2$ *to 2-ACM. Hence when* loc *is infinite, e.g. when* $\texttt{loc} := \mathbb{N}$*, the problem 2-ACM is undecidable. See the file ACM2_undec.v*

Proof. Much like what is done in our proof of Theorem 8.1 of [14], we simulate the ZERO test using a FORK. We do not elaborate more here, see the file acm2_utils.v. □

Notice that we do not strictly need loc to be infinite for undecidability. There exist two counters MM that encode universal problems. For such a machine[9] the acceptance problem is undecidable. However, this would only play a role here if we wanted an upper bound on the number of propositional variables needed in BI formulae to reach undecidability.

4. From 2-ACM acceptance to provability in BI

The core idea behind the undecidability proof for BBI in [9, 13] was that the linear logic exponential $!\varphi$ could be simulated in BBI as $\dot{1} \mathbin{\dot{\wedge}} \varphi$. With that encoding, one could simulate $!\varphi \mathbin{\dot{\to}} \dot{1}$ (weakening), $!\varphi \mathbin{\dot{\to}} !\varphi * !\varphi$ (contraction), and $!\varphi \mathbin{\dot{\to}} \varphi$ (dereliction) for those !-prefixed formulae, thus allowing the encoding of the *reusability and discardability* of MM instructions, the instructions themselves being encoded at the level of the $\{\dot{-\!\circ}, \dot{\&}, \dot{1}\}$ fragment of ILL already, hence the $\{\dot{-\!*}, \dot{\wedge}, \dot{1}\}$ fragment of BBI.

[9] one such MM is currently implemented in the CLoUP, hence with a known finite number of states. It is the compiled form of a μ-recursive program that searches for solutions of (encoded) Diophantine equations.

However that straightforward encoding of $!\varphi$ in BBI fails in the case of BI because one cannot show contraction $!\varphi \dot{\to} !\varphi \mathbin{\dot{*}} !\varphi$ in BI.[10] In our reading of [12], we single out *two critical observations* that are, from our own point of view, key to relate the former undecidability proof for BBI to the new BI one:

Observation 1: the encoding of the reusability of Minsky machines instructions does not strictly require contraction but only the combination of contraction and dereliction, i.e. $!\varphi \dot{\to} !\varphi \mathbin{\dot{*}} \varphi$;

Observation 2: this combination can be somewhat simulated in BI using an encoding of a relativized exponential $!^{\gamma}\varphi$ that we herein call *pseudo-exponential* (see below).

We will work with the $\{\dot{-\!\!*}, \dot{\to}, \dot{\wedge}, \dot{\mathrm{i}}\}$ cut-free fragment of BI. We make use of $\dot{\top}$ but only as a shortcut for $\dot{\top} := \dot{\mathrm{i}} \dot{\to} \dot{\mathrm{i}}$ in that fragment, which is an obvious equivalent. This can be justified with the two following LBI proof structures in the considered fragment:

$$\frac{\Sigma[\varnothing_+] \vdash_L \varphi}{\Sigma[\langle \dot{\mathrm{i}} \dot{\to} \dot{\mathrm{i}} \rangle] \vdash_L \varphi}\,[\mathrm{weak}] \qquad \frac{\dfrac{\dfrac{\dfrac{}{\varnothing_* \vdash_L \dot{\mathrm{i}}}\,[\dot{\mathrm{i}}_r]}{\varnothing_+ \oplus \varnothing_* \vdash_L \dot{\mathrm{i}}}\,[\mathrm{equiv}]}{\varnothing_+ \oplus \langle \dot{\mathrm{i}} \rangle \vdash_L \dot{\mathrm{i}}}\,[\dot{\mathrm{i}}_l]}{\varnothing_+ \vdash_L \dot{\mathrm{i}} \dot{\to} \dot{\mathrm{i}}}\,[\dot{\to}_r]$$

that derive equivalents to the left $[\dot{\top}_l]$ and right $[\dot{\top}_r]$ LBI introduction rules for $\dot{\top}$.

4.1. Pseudo-exponentials for reusable and discardable storage

Observation 2 above is a consequence of the two following derived rules:

Lemma 4.1. *The following two rules are derivable in cut-free LBI (see* `LBI_*_idea` *in file encoding.v):*

$$\frac{\Gamma \circledast \langle \varphi \rangle \vdash_L \gamma}{\Gamma \oplus \langle (\varphi \mathbin{\dot{-\!\!*}} \gamma) \dot{\to} \gamma \rangle \vdash_L \gamma}\,[\mathrm{obs}_2] \qquad \frac{\big(\Gamma \circledast \langle (\dot{\top} \mathbin{\dot{-\!\!*}} \psi) \mathbin{\dot{\wedge}} \dot{\mathrm{i}} \rangle\big) \oplus \langle \psi \rangle \vdash_L \gamma}{\Gamma \circledast \langle (\dot{\top} \mathbin{\dot{-\!\!*}} \psi) \mathbin{\dot{\wedge}} \dot{\mathrm{i}} \rangle \vdash_L \gamma}\,[\mathrm{obs}'_2]$$

Proof. Assuming the notation $\Delta := \Gamma \circledast \langle (\dot{\top} \mathbin{\dot{-\!\!*}} \psi) \mathbin{\dot{\wedge}} \dot{\mathrm{i}} \rangle$ for shorter display, we give two LBI proof structures to justify the claim:

$$\frac{\dfrac{\Gamma \circledast \langle \varphi \rangle \vdash_L \gamma}{\Gamma \vdash_L \varphi \mathbin{\dot{-\!\!*}} \gamma}\,[\dot{-\!\!*}_r] \quad \dfrac{}{\langle \gamma \rangle \vdash_L \gamma}\,[\mathrm{id}]}{\Gamma \oplus \langle (\varphi \mathbin{\dot{-\!\!*}} \gamma) \dot{\to} \gamma \rangle \vdash_L \gamma}\,[\dot{\to}_l] \qquad \frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varnothing_+ \vdash_L \dot{\top}}\,[\dot{\top}_r]}{\Gamma \vdash_L \dot{\top}}\,[\mathrm{weak}] \qquad \Delta \oplus \langle \psi \rangle \vdash_L \gamma}{\Delta \oplus (\Gamma \circledast \langle \dot{\top} \mathbin{\dot{-\!\!*}} \psi \rangle) \vdash_L \gamma}\,[\dot{-\!\!*}_l]}{\Delta \oplus \big(\Gamma \circledast (\langle \dot{\top} \mathbin{\dot{-\!\!*}} \psi \rangle \oplus \varnothing_+)\big) \vdash_L \gamma}\,[\mathrm{equiv]}}{\Delta \oplus \big(\Gamma \circledast (\langle \dot{\top} \mathbin{\dot{-\!\!*}} \psi \rangle \oplus \langle \dot{\mathrm{i}} \rangle)\big) \vdash_L \gamma}\,[\mathrm{weak}]}{\Delta \oplus \big(\Gamma \circledast \langle (\dot{\top} \mathbin{\dot{-\!\!*}} \psi) \mathbin{\dot{\wedge}} \dot{\mathrm{i}} \rangle\big) \vdash_L \gamma}\,[\dot{\wedge}_l]}{\Delta \vdash_L \gamma}\,[\mathrm{cntr}]$$

Notice that we avoid the justification of the equivalence $\equiv$ in the instances of rule [equiv]. We also do not explicitly give the contexts $\Sigma[\Box]$ (and the position of holes) used in left introduction rules. □

Definition 4.1 (Pseudo-exponential). *Given two BI formulae* φ, γ : `Form`*, we define the* pseudo-exponential *denoted* $!^{\gamma}\varphi := \big(\dot{\top} \mathbin{\dot{-\!\!*}} ((\varphi \mathbin{\dot{-\!\!*}} \gamma) \dot{\to} \gamma)\big) \mathbin{\dot{\wedge}} \dot{\mathrm{i}}$.

[10] to the best of our knowledge, it is unknown whether contraction can be simulated directly in BI.

Theorem 4.1. *The two following rules are derivable in cut-free LBI (see* `LBI_pseudo_exp_*` *in file encoding.v):*

$$\frac{\Gamma \vdash_L \delta}{\Gamma \circledast \langle !^\gamma \varphi\rangle \vdash_L \delta}\ [!^\gamma\text{-weak}] \qquad \frac{\big(\Gamma \circledast \langle !^\gamma \varphi\rangle\big) \circledast \langle \varphi\rangle \vdash_L \gamma}{\Gamma \circledast \langle !^\gamma \varphi\rangle \vdash_L \gamma}\ [!^\gamma\text{-derel}]$$

Proof. Using $\psi := (\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma$ for the instance of the derived rule $[\text{obs}_2']$, we give two LBI proof structures to justify the claim:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash_L \delta}{\Gamma \circledast (\varnothing_+ \oplus \varnothing_*) \vdash_L \delta}[\text{equiv}]}{\Gamma \circledast (\varnothing_+ \oplus \langle \dot{1}\rangle) \vdash_L \delta}[\dot{1}_l]}{\Gamma \circledast \big(\langle \dot{\top} \dot{-\!\!*} ((\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma)\rangle \oplus \langle \dot{1}\rangle\big) \vdash_L \delta}[\text{weak}]}{\Gamma \circledast \langle (\dot{\top} \dot{-\!\!*} ((\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma)) \dot{\wedge} \dot{1}\rangle \vdash_L \delta}[\dot{\wedge}_l] \qquad \dfrac{\big(\Gamma \circledast \langle !^\gamma \varphi\rangle\big) \circledast \langle \varphi\rangle \vdash_L \gamma}{\big(\Gamma \circledast \langle !^\gamma \varphi\rangle\big) \oplus \langle (\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma\rangle \vdash_L \gamma}[\text{obs}_2]}{\Gamma \circledast \langle (\dot{\top} \dot{-\!\!*} ((\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma)) \dot{\wedge} \dot{1}\rangle \vdash_L \gamma}[\text{obs}_2']$$

Rules $[\text{obs}_2]$ and $[\text{obs}_2']$ of Lemma 4.1 used on the right hand side proof structure expose a remarkable feature: a copy of φ is being stripped off of its surrounding pseudo-exponential, while at the same time, it switches from multiplicative to additive context, and then back into multiplicative context. □

Compared to the presentation of [12] which discovered this key interaction between additive and multiplicative contexts, here highlighted as the combination of rules $[\text{obs}_2]$ and $[\text{obs}_2']$, we insist on this presentation via the notion of pseudo-exponential because it creates a bridge between our previous encoding of MM in eILL and a forthcoming encoding of 2-ACM into BI.

With weakening $[!^\gamma\text{-weak}]$ and pseudo-dereliction $[!^\gamma\text{-derel}]$, we nearly have the material for encoding 2-ACM. Notice however the constraint that the γ in $!^\gamma\varphi$ must match the conclusion of the LBI sequent in rule $[!^\gamma\text{-derel}]$ while no such relation exists in the case of rule $[!^\gamma\text{-weak}]$. We will see that this does not impair the encoding because the possible values of γ are going to represent locations in 2-ACM programs of which there are only finitely many.

The rule $[!^\gamma\text{-derel}]$ of Theorem 4.1 allows for the repetition of Minsky machines instructions encoded as say φ where the pseudo-exponential $!^\gamma\varphi$ represents a storage facility from which one can extract a copy of any stored instruction without altering the storage itself. The rule $[!^\gamma\text{-weak}]$ allows to discard any stored instruction and hence repeatedly, the whole storage at the end of the computation. On the side of TPS semantics (which is used for the correctness of the encoding), a critical property of the storage is that $[\![!^\gamma\varphi]\!]\,\epsilon$ holds, which we check below.

Theorem 4.2. *Let* $[\![\cdot]\!] : \texttt{prop} \to M \to \mathbb{P}$ *be a TPS semantics over a commutative monoid* $(M, \circ, \epsilon)$ *and* $\varphi, \gamma : \texttt{Form}$ *be two BI formulae. If* $[\![\varphi]\!]\,\epsilon$ *then* $[\![!^\gamma\varphi]\!]\,\epsilon$. *See* `tps_BI_pseudo_exp` *in file encoding.v.*

Proof. Let us assume $[\![\varphi]\!]\,\epsilon$. For any $x : M$, we first show $[\![(\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma]\!]\,x$. Hence we assume $[\![\varphi \dot{-\!\!*} \gamma]\!]\,x$ and we want to prove $[\![\gamma]\!]\,x$. Given $[\![\varphi \dot{-\!\!*} \gamma]\!] = [\![\varphi]\!] \multimap [\![\gamma]\!]$, since $[\![\varphi]\!]\,\epsilon$ holds, we derive $[\![\gamma]\!](\epsilon \circ x)$ hence $[\![\gamma]\!]\,x$. So we get $\forall x,\ [\![(\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma]\!]\,x$ and this entails $[\![\dot{\top} \dot{-\!\!*} ((\varphi \dot{-\!\!*} \gamma) \dot{\to} \gamma)]\!]\,\epsilon$. We conclude that $[\![!^\gamma\varphi]\!]\,\epsilon$. □

4.2. Encoding individual 2-ACM instructions

We recall that the encoding of 2-ACM instructions [14] can already be achieved in the $\{\dot{-\!\!*}, \dot{\wedge}, \dot{1}\}$ fragment of LBI which is (isomorphic to) a fragment of ILL. Notice that [FORK] was not made explicit in [14] but was nevertheless used there to encode zero-tests for Minsky machines (via forking). Here we reduce from 2-ACM instead.

Theorem 4.3. *The following rules are derivable in the cut-free* $\{\dot{-\!\!*}, \dot{\wedge}, \dot{\mathrm{i}}\}$ *fragment of LBI. See* `LBI_{FORK,INC,DEC,STOP}` *in file encoding.v.*

$$\frac{\Gamma \circledast \langle \varphi \rangle \vdash_L \psi}{\Gamma \circledast \langle (\varphi \dot{-\!\!*} \psi) \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\ [\texttt{INC}] \qquad \frac{\Gamma \vdash_L \psi}{(\Gamma \circledast \langle \varphi \rangle) \circledast \langle \varphi \dot{-\!\!*} \psi \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\ [\texttt{DEC}]$$

$$\frac{\Gamma \vdash_L \dot{\mathrm{i}}}{\Gamma \circledast \langle \dot{\mathrm{i}} \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\ [\texttt{STOP}] \qquad \frac{\Gamma \vdash_L \varphi \qquad \Gamma \vdash_L \psi}{\Gamma \circledast \langle (\varphi \mathbin{\dot{\wedge}} \psi) \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\ [\texttt{FORK}]$$

Proof. We give the four following LBI proof structures:

$$\dfrac{\dfrac{\dfrac{\Gamma \circledast \langle \varphi \rangle \vdash_L \psi}{\Gamma \vdash_L \varphi \dot{-\!\!*} \psi}\,[\dot{-\!\!*}_r] \quad \dfrac{}{\langle \gamma \rangle \vdash_L \gamma}\,[\mathrm{id}]}{\Gamma \circledast \langle (\varphi \dot{-\!\!*} \psi) \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\,[\dot{-\!\!*}_l]$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\langle \varphi \rangle \vdash_L \varphi}\,[\mathrm{id}] \quad \dfrac{\Gamma \vdash_L \psi \quad \dfrac{}{\langle \gamma \rangle \vdash_L \gamma}\,[\mathrm{id}]}{\Gamma \circledast \langle \psi \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\,[\dot{-\!\!*}_l]}{\Gamma \circledast (\langle \varphi \rangle \circledast \langle \varphi \dot{-\!\!*} \psi \dot{-\!\!*} \gamma \rangle) \vdash_L \gamma}\,[\dot{-\!\!*}_l]}{(\Gamma \circledast \langle \varphi \rangle) \circledast \langle \varphi \dot{-\!\!*} \psi \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\,[\mathrm{equiv}]$$

$$\dfrac{\Gamma \vdash_L \dot{\mathrm{i}} \quad \dfrac{}{\langle \gamma \rangle \vdash_L \gamma}\,[\mathrm{id}]}{\Gamma \circledast \langle \dot{\mathrm{i}} \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\,[\dot{-\!\!*}_l]$$

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash_L \varphi \quad \Gamma \vdash_L \psi}{\Gamma \vdash_L \varphi \mathbin{\dot{\wedge}} \psi}\,[\dot{\wedge}_r] \quad \dfrac{}{\langle \gamma \rangle \vdash_L \gamma}\,[\mathrm{id}]}{\Gamma \circledast \langle (\varphi \mathbin{\dot{\wedge}} \psi) \dot{-\!\!*} \gamma \rangle \vdash_L \gamma}\,[\dot{-\!\!*}_l]$$

□

4.3. Encoding 2-ACM acceptance

Given a list $\Phi = [\varphi_1; \ldots; \varphi_n]$ of BI formulae and a single formula ψ, we encode this list as a (multiplicative) bunch with

$$[\varphi_1; \ldots; \varphi_n]^{\circledast} := \langle \varphi_1 \rangle \circledast (\cdots \circledast (\langle \varphi_n \rangle \circledast \varnothing_*) \cdots)$$

and we show that if two lists Φ_1 and Φ_2 are permutable then $\Phi_1^{\circledast} \equiv \Phi_2^{\circledast}$, i.e. permutations preserve the $\equiv$-equivalence classes of bunches. In particular if φ occurs in the list Φ then we can display φ the following way: there is a list Ψ such that $\Phi^{\circledast} \equiv \Psi^{\circledast} \circledast \langle \varphi \rangle$. Using these properties, we can reformulate the derived rules [$!^{\gamma}$-weak] and [$!^{\gamma}$-derel] of Theorem 4.1.

Proposition 4.1. *The following rules are derivable in the cut-free* $\{\dot{-\!\!*}, \dot{\to}, \dot{\wedge}, \dot{\mathrm{i}}\}$ *fragment of LBI.*

$$\frac{}{\Phi^{\circledast} \vdash_L \dot{\mathrm{i}}}\ \forall \psi \in \Phi,\, \exists\, \gamma\, \varphi,\, \psi = !^{\gamma}\varphi \qquad \frac{\Phi^{\circledast} \circledast \langle \varphi \rangle \vdash_L \gamma}{\Phi^{\circledast} \vdash_L \gamma}\ !^{\gamma}\varphi \in \Phi$$

the lhs derived rule being applicable when the list Φ *is composed exclusively of pseudo-exponentials, and the rhs derived rule creating a copy of the formula stored in a pseudo-exponential occurring in* Φ*. See* `LBI_list_mult_{weak,derilection}` *in file encoding.v.*

Proof. For the left rule, we proceed by induction on the list Φ. When $\Phi = [\,]$ is the empty list, we get the rule as an instance of the LBI rule $[\dot{\mathrm{i}}_r]$. When $\Phi = !^{\gamma}\varphi :: \Psi$ then we have $\Phi^{\circledast} = \langle !^{\gamma}\varphi \rangle \circledast \Psi^{\circledast}$ and we proceed with the following proof structure:

$$\dfrac{\dfrac{\Psi^{\circledast} \vdash_L \dot{\mathrm{i}}}{\Psi^{\circledast} \circledast \langle !^{\gamma}\varphi \rangle \vdash_L \dot{\mathrm{i}}}\,[!^{\gamma}\text{-weak}]}{\langle !^{\gamma}\varphi \rangle \circledast \Psi^{\circledast} \vdash_L \dot{\mathrm{i}}}\,[\mathrm{equiv}]$$

and plug the induction hypothesis (a LBI proof of $\Psi^{\circledast} \vdash_L \dot{\mathrm{i}}$) to get an LBI proof of $\Phi^{\circledast} \vdash_L \dot{\mathrm{i}}$.

For the right rule, since $!^\gamma\varphi \in \Phi$, we can find Ψ such that $\Phi^{\circledast} \equiv \Psi^{\circledast} \circledast \langle !^\gamma\varphi\rangle$ and then proceed with the following proof structure:

$$\dfrac{\dfrac{\dfrac{\Phi^{\circledast} \circledast \langle\varphi\rangle \vdash_L \gamma}{(\Psi^{\circledast} \circledast \langle !^\gamma\varphi\rangle) \circledast \langle\varphi\rangle \vdash_L \gamma}\,[\text{equiv}]}{\Psi^{\circledast} \circledast \langle !^\gamma\varphi\rangle \vdash_L \gamma}\,[!^\gamma\text{-derel}]}{\Phi^{\circledast} \vdash_L \gamma}\,[\text{equiv}]$$

This concludes the proof. □

We additionally define the *repetition* operator $n.\varphi$ that produces a list of n copies of φ and, given a list Φ of (hypothesis) formulae and a (conclusion) formula ψ, the *folding* of the operator $\dot{-\!\!*}$ over the list Φ starting with ψ and denoted $\Phi \mathrel{\dot{-\!\!*}\!\!*} \psi$:

$$n.\varphi := [\varphi; \ldots; \varphi] \text{ with } n \text{ copies of } \varphi \qquad [\varphi_1; \ldots; \varphi_n] \mathrel{\dot{-\!\!*}\!\!*} \psi := \varphi_1 \mathrel{\dot{-\!\!*}} \left(\cdots \mathrel{\dot{-\!\!*}} (\varphi_n \mathrel{\dot{-\!\!*}} \psi) \cdots\right)$$

We show the identities $0.\varphi \mathrel{\dot{-\!\!*}\!\!*} \psi = [\,] \mathrel{\dot{-\!\!*}\!\!*} \psi = \psi$ and $(1+n).\varphi \mathrel{\dot{-\!\!*}\!\!*} \psi = \varphi \mathrel{\dot{-\!\!*}} (n.\varphi \mathrel{\dot{-\!\!*}\!\!*} \psi)$ and $(\Phi +\!\!+ \Psi) \mathrel{\dot{-\!\!*}\!\!*} \psi = \Phi \mathrel{\dot{-\!\!*}\!\!*} (\Psi \mathrel{\dot{-\!\!*}\!\!*} \psi)$.

Remark 4.1. *$\Phi^{\circledast} \vdash_L \psi$ entails $\vdash_{LBI} \dot{1} \mathrel{\dot{\to}} \Phi \mathrel{\dot{-\!\!*}\!\!*} \psi$.*

Proof. By (repeated applications) of rules $[\dot{-\!\!*}_r]$ and [equiv] and once $[\dot{1}_l]$ and $[\dot{\to}_r]$. □

We have enough material to implement our positive encoding of 2-ACM acceptance, which mimics the encoding of two counters MM termination in eILL [14]. We assume a type of locations $\mathtt{loc}$ for 2-ACM instructions. We will build a LBI formula of which the variables span over $\mathtt{prop} := \mathtt{loc} + \{\alpha, \beta\}$ extending locations viewed as variables with two fresh variables α and β. We define the source location of a 2-ACM instruction as a map $\mathtt{src} : \text{2-ACM}_{\mathtt{loc}} \to \mathtt{loc}$ by case distinction:

$$\mathtt{src}(\mathtt{INC}\,\_\,p\,\_) := p \quad \mathtt{src}(\mathtt{DEC}\,\_\,p\,\_) := p \quad \mathtt{src}(\mathtt{STOP}\,p) := p \quad \mathtt{src}(\mathtt{FORK}\,p\,\_\,\_) := p$$

and then the encoding of 2-ACM instructions to BI formulae in the $\{\dot{-\!\!*}, \dot{\wedge}, \dot{1}\}$ fragment as a map $\mathrm{enc} : \text{2-ACM}_{\mathtt{loc}} \to \mathtt{Form}_{\mathtt{loc}+\{\alpha,\beta\}}$:

$$\begin{array}{rl} \mathrm{enc}(\mathtt{INC}\,c\,p\,q) := (c \mathrel{\dot{-\!\!*}} q) \mathrel{\dot{-\!\!*}} p & \qquad \mathrm{enc}(\mathtt{DEC}\,c\,p\,q) := c \mathrel{\dot{-\!\!*}} q \mathrel{\dot{-\!\!*}} p \\ \mathrm{enc}(\mathtt{STOP}\,p) := \dot{1} \mathrel{\dot{-\!\!*}} p & \qquad \mathrm{enc}(\mathtt{FORK}\,p\,q\,r) := (q \mathbin{\dot{\wedge}} r) \mathrel{\dot{-\!\!*}} p \end{array}$$

where $c : \{\alpha, \beta\}$ and $p, q, r : \mathtt{loc}$.

Given a list of 2-ACM instructions, we encode it as a list of BI formulae:

$$\mathrm{enc}\,\Sigma := [!^s(\mathrm{enc}\,i) \mid i \in \Sigma, s \in \mathtt{src}\,\Sigma]$$

composed of a pseudo-exponential $!^s(\mathrm{enc}\,i)$ for each 2-ACM instruction i occurring in Σ and each location s occurring as a source location in Σ, so (formally) the product of the lists Σ and $\mathtt{src}\,\Sigma$ using the product operator $\lambda\, i\, s, !^s(\mathrm{enc}\,i)$.

The encoding of a 2-ACM acceptance statement $\Sigma \mathbin{/\!\!/_A} a \parallel b \vdash p$ is the following

$$\dot{1} \mathrel{\dot{\to}} (a.\alpha +\!\!+ b.\beta +\!\!+ \mathrm{enc}\,\Sigma) \mathrel{\dot{-\!\!*}\!\!*} p$$

Theorem 4.4 (Soundness of the encoding). *If the 2-ACM statement $\Sigma \mathbin{/\!\!/_A} a \parallel b \vdash p$ is accepted then $\vdash_{LBI} \dot{1} \mathrel{\dot{\to}} (a.\alpha +\!\!+ b.\beta +\!\!+ \mathrm{enc}\,\Sigma) \mathrel{\dot{-\!\!*}\!\!*} p$. See* `acm2_encode_sound` *in the file encoding.v.*

Proof. Assuming $\Sigma /\!\!/_A a \parallel b \vdash p$, by Remark 4.1 it is enough to show $\left(a.\alpha +\!\!+ b.\beta +\!\!+ \operatorname{enc}\Sigma\right)^{\circledast} \vdash_L p$. We construct a cut-free derivation of the later sequent in the $(\dot{-\!\!*}, \dot{\to}, \dot{\wedge}, \dot{1})$ cut-free fragment of LBI by induction on the (derivation of) the acceptance statement $\Sigma /\!\!/_A a \parallel b \vdash p$ using the rules of Theorem 4.3 and Proposition 4.1. Below we only deal with the rules $[\mathtt{STOP}\, p \in \Sigma]$ and $[\mathtt{FORK}\, p\, q\, r \in \Sigma]$ but the reader can find the complete mechanized argument in the file encoding.v.

Let us start with an instance of rule $[\mathtt{STOP}\, p \in \Sigma]$. We proceed the following way:

$$\frac{}{\Sigma /\!\!/_A 0 \parallel 0 \vdash p} \quad \rightsquigarrow \quad \frac{\dfrac{\dfrac{}{(\operatorname{enc}\Sigma)^{\circledast} \vdash_L \dot{1}}\, \forall \psi \in \operatorname{enc}\Sigma,\, \exists\, \gamma\, \varphi,\, \psi = !^{\gamma}\varphi}{(\operatorname{enc}\Sigma)^{\circledast} \circledast \langle \dot{1} \dot{-\!\!*} p\rangle \vdash_L p}\,[\mathtt{STOP}]}{(\operatorname{enc}\Sigma)^{\circledast} \vdash_L p}\, !^{p}(\operatorname{enc}(\mathtt{STOP}\, p)) \in \operatorname{enc}\Sigma$$

and get a LBI proof of the encoded 2-ACM acceptance statement.

For rule $[\mathtt{FORK}\, p\, q\, r \in \Sigma]$, denoting $\Phi := a.\alpha +\!\!+ b.\beta +\!\!+ \operatorname{enc}\Sigma$ we proceed the following way:

$$\frac{\Sigma /\!\!/_A a \parallel b \vdash q \qquad \Sigma /\!\!/_A a \parallel b \vdash r}{\Sigma /\!\!/_A a \parallel b \vdash p} \quad \rightsquigarrow \quad \frac{\dfrac{\Phi^{\circledast} \vdash_L q \quad \Phi^{\circledast} \vdash_L r}{\Phi^{\circledast} \circledast \langle (q \dot{\wedge} r) \dot{-\!\!*} p\rangle \vdash_L p}\,[\mathtt{FORK}]}{\Phi^{\circledast} \vdash_L p}\, !^{p}(\operatorname{enc}(\mathtt{FORK}\, p\, q\, r)) \in \Phi$$

where $!^{p}(\operatorname{enc}(\mathtt{FORK}\, p\, q\, r)) = !^{p}((q \dot{\wedge} r) \dot{-\!\!*} p)$ belongs to Φ because it contains $\operatorname{enc}\Sigma$. Then we apply the two induction hypotheses and plug LBI proofs of $\Phi^{\circledast} \vdash_L q$ and $\Phi^{\circledast} \vdash_L r$ in place of the leaves of the proof structure on the right hand side. □

Theorem 4.5 (Completeness of the encoding). *If the formula* $\dot{1} \dot{\to} (a.\alpha +\!\!+ b.\beta +\!\!+ \operatorname{enc}\Sigma) \dot{-\!\!-\!\!*}\, p$ *is valid in TPS then 2-ACM statement* $\Sigma /\!\!/_A a \parallel b \vdash p$ *is accepted. See* `acm2_encode_complete` *in the file encoding.v.*

Proof. Given a fixed list for 2-ACM instructions Σ, we use the following trivial phase interpretation in the commutative monoid $\mathbb{N} \times \mathbb{N}$ based on pairs (x, y) natural numbers:

$$\begin{aligned} [\![l]\!]\,(x,y) &:= \Sigma /\!\!/_A x \parallel y \vdash l \text{ is accepted} \\ [\![\alpha]\!]\,(x,y) &:= (x,y) = (1,0) \\ [\![\beta]\!]\,(x,y) &:= (x,y) = (0,1) \end{aligned}$$

where $l : \mathtt{loc}$ so this spans all the propositional variables in $\mathtt{loc} + \{\alpha, \beta\}$.

For any instruction i in Σ we show that $[\![\operatorname{enc} i]\!]\,(0,0)$ holds. Notice that this property was already mechanized in [14] and we simply import the proof in this new formalism. Let us consider for instance the case of $\mathtt{FORK}\, p\, q\, r \in \Sigma$. Let us show $[\![\operatorname{enc}(\mathtt{FORK}\, p\, q\, r)]\!]\,(0,0)$ holds. This unfolds as $[\![(q \dot{\wedge} r) \dot{-\!\!*} p]\!]\,(0,0)$, hence unfolding again, for any x and y, the entailment $[\![q]\!]\,(x,y) \wedge [\![r]\!]\,(x,y) \to [\![p]\!]\,(x+0, y+0)$ holds. By definition, this means $\Sigma /\!\!/_A x \parallel y \vdash q \wedge \Sigma /\!\!/_A x \parallel y \vdash r \to \Sigma /\!\!/_A x \parallel y \vdash p$ which trivially follows from rule $[\mathtt{FORK}\, p\, q\, r \in \Sigma]$.

We then show the following entailments:

(1) $[\![n.\alpha \dot{-\!\!-\!\!*}\, \varphi]\!]\,(x,y) \to [\![\varphi]\!](n+x, y)$ by induction on $n : \mathbb{N}$;
(2) $[\![n.\beta \dot{-\!\!-\!\!*}\, \varphi]\!]\,(x,y) \to [\![\varphi]\!](x, n+y)$ by induction on $n : \mathbb{N}$;
(3) $\left(\forall \psi, \psi \in \Phi \to [\![\psi]\!]\,(0,0)\right) \to [\![\Phi \dot{-\!\!-\!\!*}\, \varphi]\!] \subseteq [\![\varphi]\!]$ by induction on the list Φ.

Now assuming that the formula $\dot{1} \dot{\to} (a.\alpha +\!\!+ b.\beta +\!\!+ \operatorname{enc}\Sigma) \dot{-\!\!-\!\!*}\, p$ is valid in TPS, from $[\![\dot{1}]\!]\,(0,0)$ we deduce $[\![(a.\alpha) \dot{-\!\!-\!\!*}\, (b.\beta) \dot{-\!\!-\!\!*}\, \operatorname{enc}\Sigma \dot{-\!\!-\!\!*}\, p]\!]\,(0,0)$ and then $[\![\operatorname{enc}\Sigma \dot{-\!\!-\!\!*}\, p]\!]\,(a,b)$ by entailments (1) and (2) above. Since $[\![\operatorname{enc} i]\!]\,(0,0)$ holds for any $i \in \Sigma$, by Theorem 4.2 this entails $[\![!^{s}(\operatorname{enc} i)]\!]\,(0,0)$ for any $i \in \Sigma$ and $s \in \operatorname{src}\Sigma$. As a consequence, using entailment (3) above with $\Phi := \operatorname{enc}\Sigma$, we get $[\![p]\!]\,(a,b)$ which by definition means that $\Sigma /\!\!/_A a \parallel b \vdash p$ is accepted. □

4.4. The undecidability of BI

Theorem 4.6. *There is a many-one reduction from acceptance in 2-ACM (with locations in* $\mathtt{loc}$*) to provability in BI (with propositional variables in* $\mathtt{loc} + \{\alpha, \beta\}$*). Provability in BI understood as either HBI-provability or else LBI-provability in any fragment containing the connectives* $\{\dot{-\!\!*}, \dot{\rightarrow}, \dot{\wedge}, \dot{1}\}$*, irrelevant of the availability of the cut-rule. See file ACM2_to_BI.v.*

Proof. The result is a consequence of the soundness Theorem 4.4 for the encoding, the monotonicity Proposition 2.1 for LBI, the soundness of Theorem 2.1 for LBI w.r.t. HBI, the soundness Theorem 2.3 of trivial phase semantics for HBI, and the completeness Theorem 4.5 for the encoding. □

Notice that thanks to cut-elimination the various fragments of LBI all define the same notion of provability (on the part of the syntax that they have in common) but we do not need that property to establish that they are all undecidable. The reduction might be extended to any system that contains HBI and is sound for TPS, hence possibly stronger ones like separation logic (based on IL).

Theorem 4.7. *Provability in BI is undecidable for any fragment containing* $\{\dot{-\!\!*}, \dot{\rightarrow}, \dot{\wedge}, \dot{1}\}$ *when* $\mathtt{prop} := \mathbb{N}$*. See file BI_undec.v.*

Proof. Indeed, there is an injective map from $\mathbb{N} + \{\alpha, \beta\}$ into $\mathbb{N}$ and 2-ACM is undecidable when $\mathtt{loc} = \mathbb{N}$ by Theorem 3.1. We combine this with the many-one reduction of Theorem 4.6. □

Remark that one can exhibit a 2-ACM which is universal and hence of which the acceptance problem is undecidable. As a consequence, we could give an explicit finite bound for the number of BI variables over which BI becomes undecidable. But if one is interested in trying to minimize that bound then we would advise using more than two counters to get a universal ACM with the shortest possible program instead of the least number of counters. Each counter just costs one logical variable, the same amount as each source location in ACM programs.

5. Discussion and Conclusion

We mechanize a somewhat generic proof of undecidability of BI based on some keys insights of [12] that we interpret using the notion of pseudo-exponential. The technique applies to fragments containing the connectives $\dot{-\!\!*}$, $\dot{\rightarrow}$, $\dot{\wedge}$ and $\dot{1}$, and which are sound for trivial phase semantics. Due to the requirements of the synthetic framework of undecidability, the soundness proof needs to be constructive hence does not apply as is to e.g. Boolean BI (BBI).

In [12], they provide two distinct proofs of undecidability, each concerning different syntactic fragments. The proof that reduces from tiling problems does not seem to involve the unit $\dot{1}$ but involves disjunction $\dot{\vee}$ and a negation. The proof that reduces from 2-ACM requires a strictly larger fragment than our own because they moreover need the disjunction $\dot{\vee}$. We think that this is due to a negative encoding of 2-ACM instructions that we circumvent using a positive encoding. Both of their proofs work also in the case where $\dot{*}$ is a noncommutative operator. We reasonably think that our own might be convertible to the noncommutative case as well, also one would have to carefully review (and possibly redesign) some of the derived rules to account for that change.

Concerning the meta-theory, the authors of [12] mention that they were careful to avoid the axiom of choice. It seems that they did not consider the case of excluded middle (XM) with such care. In particular, the specific BI algebra that plays a "central role" in their tiling based proof is a Boolean algebra, hence we cannot see how they would be able to avoid XM. Concerning the 2-ACM based proof however, it might be the case that it does not need XM but then, our positive encoding works using a strictly smaller fragment (in the commutative case).

Once [12] was made available on arXiv, we implemented an undecidability proof for 2-ACM, and then it took us a relatively short amount of time (say half a week) to get a first mechanized undecidability proof for BI. This is clearly due to the availability of the Coq Library of Undecidability Proofs, a versatile

library for undecidability in Coq/Rocq. The 2-ACM seed of undecidability was very close to an existing MM acceptance problem and somewhat straightforward to mechanize. Our own experience with the reduction from MM to eILL allowed us to immediately frame the 2-ACM to BI reduction in terms of the pseudo-dereliction rule.

Concerning future work, we could adapt the cut-elimination proof for BI [21] to our framework, hence deriving conservativity results. As discussed above, we could also consider the noncommutative case. The case of Boolean logics, e.g. BBI, could be more challenging (constructively). Clearly, sticking to trivial phase semantics requires XM at the meta-level, something we cannot afford in the synthetic framework. One could consider a more general phase semantics that would be constructively sound for BBI, for instance using double-negation closure instead of identity closure (TPS).

Acknowledgments

We thank the anonymous ARQNL 2026 reviewers for their careful reading and helpful comments. This work was partially supported by the NARCO project ANR-21-CE48-0011.

Declaration on Generative AI

The author has not employed any Generative AI tools.

References

[1] P. W. O'Hearn, D. J. Pym, The Logic of Bunched Implications, Bulletin of Symbolic Logic 5 (1999) 215–244. doi:10.2307/421090.

[2] D. Galmiche, D. Méry, D. Pym, The semantics of BI and resource tableaux, Mathematical Structures in Computer Science 15 (2005) 1033–1088. doi:10.1017/S0960129505004858.

[3] N. Galatos, P. Jipsen, Distributive residuated frames and generalized bunched implication algebras, Algebra universalis 78 (2017) 303–336. doi:10.1007/s00012-017-0456-x.

[4] M. Kaminski, N. Francez, The Lambek Calculus Extended with Intuitionistic Propositional Logic, Studia Logica 104 (2016) 1051–1082. doi:10.1007/s11225-016-9665-0.

[5] J. Brotherston, A unified display proof theory for bunched logic, Electron. Notes Theor. Comput. Sci. 265 (2010) 197–211. doi:10.1016/j.entcs.2010.08.012.

[6] J. Reynolds, Separation logic: a logic for shared mutable data structures, in: Proceedings 17th Annual IEEE Symposium on Logic in Computer Science, 2002, pp. 55–74. doi:10.1109/LICS.2002.1029817.

[7] D. Larchey-Wendling, D. Galmiche, Exploring the relation between Intuitionistic BI and Boolean BI: an unexpected embedding, Mathematical Structures in Computer Science 19 (2009) 435–500. doi:10.1017/S0960129509007567.

[8] J. Brotherston, M. Kanovich, Undecidability of Propositional Separation Logic and Its Neighbours, in: 2010 25th Annual IEEE Symposium on Logic in Computer Science, 2010, pp. 130–139. doi:10.1109/LICS.2010.24.

[9] D. Larchey-Wendling, D. Galmiche, The Undecidability of Boolean BI through Phase Semantics, in: 2010 25th Annual IEEE Symposium on Logic in Computer Science, 2010, pp. 140–149. doi:10.1109/LICS.2010.18.

[10] Á. Kurucz, I. Németi, I. Sain, A. Simon, Decidable and Undecidable Logics with a Binary Modality, Journal of Logic, Language, and Information 4 (1995) 191–206. URL: http://www.jstor.org/stable/40180071.

[11] R. Ramanayake, A syntactic proof of decidability for the logic of bunched implication BI, 2016. arXiv:1609.05847, withdrawn by the author in 2026.

[12] N. Galatos, P. Jipsen, S. B. Knudstorp, R. Ramanayake, The logic of bunched implications is undecidable, 2026. arXiv:2603.01595.

[13] D. Larchey-Wendling, D. Galmiche, Nondeterministic Phase Semantics and the Undecidability of Boolean BI, ACM Trans. Comput. Logic 14 (2013). doi:10.1145/2422085.2422091.

[14] Y. Forster, D. Larchey-Wendling, Certified undecidability of intuitionistic linear logic via binary stack machines and Minsky machines, in: Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2019, Association for Computing Machinery, New York, NY, USA, 2019, p. 104–117. doi:10.1145/3293880.3294096.

[15] Y. Forster, D. Kirst, G. Smolka, On synthetic undecidability in Coq, with an application to the Entscheidungsproblem, in: Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2019, Association for Computing Machinery, New York, NY, USA, 2019, p. 38–51. doi:10.1145/3293880.3294091.

[16] Y. Forster, E. Heiter, G. Smolka, Verification of PCP-Related Computational Reductions in Coq, in: J. Avigad, A. Mahboubi (Eds.), Interactive Theorem Proving, Springer International Publishing, Cham, 2018, pp. 253–269. doi:10.1007/978-3-319-94821-8_15.

[17] P. Lincoln, J. Mitchell, A. Scedrov, N. Shankar, Decision problems for propositional linear logic, Annals of Pure and Applied Logic 56 (1992) 239–311. doi:10.1016/0168-0072(92)90075-B.

[18] D. Larchey-Wendling, Synthetic Undecidability of MSELL via FRACTRAN Mechanised in Coq, in: N. Kobayashi (Ed.), 6th International Conference on Formal Structures for Computation and Deduction (FSCD 2021), volume 195 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2021, pp. 18:1–18:20. doi:10.4230/LIPIcs.FSCD.2021.18.

[19] D. Galmiche, D. Larchey-Wendling, Expressivity properties of Boolean BI through relational models, in: Proceedings of the 26th International Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS'06, Springer-Verlag, Berlin, Heidelberg, 2006, p. 357–368. doi:10.1007/11944836_33.

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

[21] D. Frumin, Semantic cut elimination for the logic of bunched implications, formalized in Coq, in: Proceedings of the 11th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2022, Association for Computing Machinery, New York, NY, USA, 2022, p. 291–306. doi:10.1145/3497775.3503690.