Resource Tableaux (Extended Abstract)

Didier Galmiche *Daniel Méry* *David Pym*
LORIA, Nancy *University of Bath*
France *England*

{galmiche, dmery}@loria.fr d.j.pym@bath.ac.uk

Abstract The logic of bunched implications, **BI**, provides a logical analysis of a basic notion of resource rich enough to provide a "pointer logic" semantics for programs which manipulate mutable data structures. We develop a theory of semantic tableaux for **BI**, so providing an elegant basis for efficient theorem proving tools for **BI**. It is based on the use of an algebra of labels for **BI**'s tableaux to solve the resource-distribution problem, the labels being the elements of resource models. For **BI** with inconsistency, $\perp$, the challenge consists in dealing with **BI**'s Grothendieck topological models within such a proof-search method, based on labels. We prove soundness and completeness theorems for a resource tableaux method **TBI** with respect to this semantics and provides a way to build counter-models from so-called dependency graphs. As consequences, we have two strong new results for **BI**: the decidability of propositional **BI** and the finite model property with respect to Grothendieck topological semantics. In addition, we propose, by considering partially defined monoids, a new semantics which generalizes the semantics of **BI**'s pointer logic and for which **BI** is complete

Keywords: **BI**; resources; semantics; tableaux; decidability; finite model property.

1 Introduction

The notion of *resource* is a basic one in many fields, including economics, engineering and psychology, but it is perhaps most clearly illuminated in computer science. The location, ownership, access to and, indeed, consumption of, resources are central concerns in the design of systems (such as networks, within which processors must access devices such as file servers, disks and printers) and in the design programs, which access memory and manipulate data structures (such as pointers).

The development of a mathematical theory of resource is one of the objectives of the programme of study of **BI**, the logic of bunch implications, introduced by O'Hearn and Pym [10,12,13]. The basic idea is to model directly the observed properties of resources and then to give a logical axiomatization. Initially, we require the following properties of resource, beginning with the simple assumption of a set R of elements of a resource: a *combination*, $\circ$, of resources, together with a *zero* resource, e; a *comparison*, $\sqsubseteq$, of resources. Mathematically, we model this set-up with a (for now, commutative) preordered monoid, $\mathcal{R} = (R, \circ, e, \sqsubseteq)$, in which $\circ$, with unit e, is functorial with respect to $\sqsubseteq$. Taking such a structure as an algebra of *worlds*, we obtain a forcing semantics for (propositional) **BI** which freely combines multiplicative (intuitionistic linear $\otimes$ and $\multimap$) and additive (intuitionistic $\wedge$, $\rightarrow$ and $\vee$) structure. A significant variation takes classical additives instead. **BI** is described in necessary detail in § 2. For now, the key property of the semantics is the *sharing interpretation* [10].

The (elementary) semantics of the multiplicative conjunction, $m \models \phi_1 * \phi_2$ iff there are n_1 and n_2 such that $m \sqsubseteq n_1 \circ n_2$, $n_1 \models \phi_1$ and $n_2 \models \phi_2$, is interpreted as follows: the resource m is sufficient to support $\phi_1 * \phi_2$ just in case it can be divided into resources n_1 and n_2 such that n_1 is sufficient to support ϕ_1 and n_2 is sufficient to support ϕ_2. The assertions ϕ_1 and ϕ_2 — think of them as expressing properties of programs — *do not share* resources. In contrast, in the semantics of the additive conjunction, $m \models \phi_1 \wedge \phi_2$ iff $m \models \phi_1$ and $m \models \phi_2$, the assertions ϕ_1 and ϕ_2 *share* the resource m. Similarly, the semantics of the multiplicative implication, $m \models \phi \multimap \psi$ iff for all n such that $n \models \phi$, $m \circ n \models \psi$, is interpreted as follows: the resource m is sufficient to support $\phi \multimap \psi$ — think of the proposition as (the type of) a function — just in case for any resource n which is sufficient to support ϕ — think of it as the argument to the function — the combination $m \circ n$ is sufficient to support ψ. The function and its argument *do not share* resources. In contrast, in the semantics of additive implication, $m \models \phi \rightarrow \psi$ iff for all $n \sqsubseteq m$, if $n \models \phi$, then $n \models \psi$, the function and its argument *share* the resource n. For a simple example of resource as *cost*, let the monoid be given by the natural numbers with addition and unit zero, ordered by less than or equals. A more substantial example, "pointer logic", **PL**, and its spatial semantics, has been provided by Ishtiaq and O'Hearn [8]. In fact, the semantics of pointer logic is based on *partial monoids*, in which the operation $\circ$ is partially defined.

An elementary Kripke resource semantics, formulated in categories of *presheaves* on preordered monoids, has been defined for **BI** [10,12,13] but it is sound and complete only for **BI** without inconsistency, $\perp$, the unit of the additive disjunction. This elementary forcing semantics handles inconsistency only by denying the existence of a world at which $\perp$ is forced. The completeness of **BI** for a monoid-based forcing semantics is achieved, firstly, in categories of *sheaves* on *open topological monoids* [13,14,10] and, secondly, in the more abstract topological setting of *Grothendieck sheaves* on preordered monoids [13,14]. This latter more general semantics is sketched in § 2. In each of these cases, inconsistency is internalized in the semantics. The semantics of pointer logic can be incorporated into the Kripke semantics based on Grothendieck sheaves [13,14]. But it suggests partial monoids as a basis for a "Kripke resource semantics".

BI provides a logical analysis of a basic notion of resource [13], quite different from linear logic's "number-of-uses" reading, which has proved rich enough to provide both intuitionistic and classical (*i.e.*, additives) "pointer logic" semantics for programs which manipulate mutable data structures [8,9,14]. In this context, efficient and useful proof-search methods are necessary. For many logics, semantic tableaux have provided elegant and efficient bases for tools based on both proof-search and countermodel generation [2]. We should like to have bases for such tools for **BI** and **PL**. The main difficulty to be overcome in giving such a system for **BI** is the presence of multiplicatives. We need a mechanism for calculating the distribution of "resources" with multiplicative rules which, in **BI**'s sequent calculus, given in § 2, is handled via side-formulæ. A solution is a specific use of *labels* that allow the capture of the semantical relationships between connectives during proof-search or proof-analysis [1,3,5].

Recent work has proposed a tableaux calculus, with labels, for **BI**$^{\perp}$, *i.e.*, **BI** without $\perp$, which captures the elementary Kripke resource semantics [4] but an open question until now has been whether a similar approach or calculus can be extended to full **BI**, including $\perp$, and thus provide a decision procedure for **BI** (decidability of **BI** has been conjectured, via a different method, in [13] but not explicitly proved). A real difficulty lies in the treatment of a monoid-based forcing semantics, like Grothendieck topological semantics [13], with such a labelled calculus. In § 3, we define a system of labelled semantic tableaux, **TBI**, in which the labels are drawn from **BI**'s algebra of worlds and which use **BI**'s forcing semantics, based on Grothendieck sheaves. The rules are similar to the ones of [4] but the specific way to deal with $\perp$ topologically involves delicate new closure and provability conditions. We obtain, in § 4, soundness and completeness theorems for **TBI** with respect to the Grothendieck topological semantics given in § 2. Moreover, we use our completeness proof to show that in the case of a *failed* tableau, *i.e.*, non-provability, we can *construct a countermodel* from a particular structure, called a *dependency graph*. Consequently, we obtain proofs of two *new* results for **BI**, namely, the *finite model property* with respect to Grothendieck topological semantics and the *decidability* for propositional **BI**, conjectured but not proved in [13]. Moreover, observing that a dependency graph only deals with the relevant resources needed to decide provability, we propose, in § 5, a new resource semantics for **BI** that corresponds in fact to an alternative way of dealing with $\perp$ by considering partially defined monoids. This way was mentioned but not developed in [13] and thus this new resource semantics, which generalizes the semantics of pointer logic [8], is complete and naturally derived from our study of resource tableaux. The identified relationships between resources, labels, dependency graphs, proof-search and resource semantics are also essential. For instance, dependency graphs are directly countermodels in this new semantics.

2 The Semantics and Proof Theory of BI

We review briefly the semantics and proof theory of **BI**. The details are in [13,14]. There is an elementary Kripke resource semantics that, because of the interaction between $-\!\!*$ and $\perp$ [13,14], is complete only for $\mathbf{BI}^{\perp}$. In order to have completeness with $\perp$, the topological setting described below, which is a significant step over the elementary case, is used [13,14].

Definition 1 (GTM). *A* Grothendieck Topological Monoid *is given by a quintuple* $\mathcal{M} = \langle M, \circ, e, \sqsubseteq, J\rangle$*, where* $\langle M, \circ, e, \sqsubseteq \rangle$ *is a preordered commutative monoid and* J *is a map* $J : M \to \wp(\wp(M))$ *satisfying the following:*

1. Sieve: *for any* $m \in M$*,* $S \in J(m)$ *and* $m' \in S$*,* $m \sqsubseteq m'$*;*
2. Maximality: *for any* n' *such that* $n' = n$*,* $\{n'\}$ *is in* $J(n)$*;*
3. Stability: *for any* m*,* $n \in M$ *and* $S \in J(m)$ *such that* $m \sqsubseteq n$*, there exists* $S' \in J(n)$ *such that for any* $n' \in S'$*, there exists* $m' \in S$ *such that* $m' \sqsubseteq n'$*;*
4. Transitivity: *for any* $m \in M$*,* $S \in J(m)$ *and* $\{S_{m'} \in J(m')\}_{m' \in S}$*,* $\bigcup_{m' \in S} S_{m'} \in J(m)$*;*
5. Continuity: *for any* m*,* $n \in M$ *and* $S \in J(m)$*,* $\{m' \circ n \mid m' \in S\} \in J(m \circ n)$*.*

Such a J *is usually called a* Grothendieck topology.

Definition 2 (GTI). *Let* $\mathcal{M}$ *be a GTM and* $\mathcal{P}(L)$ *be the collection of* **BI** *propositions over a language* L *of propositional letters, a* Grothendieck Topological Interpretation *is a function* $[\![-]\!] : L \to \wp(M)$ *satisfying:*

6. *(K): for any* $m, n \in M$ *such that* $n \sqsubseteq m$*,* $n \in [\![\mathrm{p}]\!]$ *implies* $m \in [\![\mathrm{p}]\!]$*;*
7. *(Sh): for any* $m \in M$ *and* $S \in J(m)$*, if, for all* $m' \in S$*,* $m' \in [\![\mathrm{p}]\!]$*, then* $m \in [\![\mathrm{p}]\!]$*.*

It is shown in [13] that given an interpretation which makes (K) and (Sh) hold for atomic propositions, (K) and (Sh) also hold for any proposition of **BI**.

Definition 3 (GRM). *A* Grothendieck Resource Model*, or GRM, is a triple* $\mathcal{G} = \langle \mathcal{M}, \models, [\![-]\!] \rangle$ *in which* $\mathcal{M} = \langle M, \circ, e, \sqsubseteq, J \rangle$ *is a GTM,* $[\![-]\!]$ *is a GTI and* $\models$ *is a forcing relation on* $M \times \mathcal{P}(L)$ *satisfying the following conditions:*

- $m \models \mathrm{p}$ *iff* $m \in [\![\mathrm{p}]\!]$
- $m \models \top$ *iff always*
- $m \models \bot$ *iff* $\emptyset \in J(m)$
- $m \models \phi \wedge \psi$ *iff* $m \models \phi$ *and* $m \models \psi$
- $m \models \phi \vee \psi$ *iff there exists* $S \in J(m)$ *such that for any* $m' \in S$, $m' \models \phi$ *or* $m' \models \psi$
- $m \models \phi \rightarrow \psi$ *iff for any* $n \in M$ *such that* $m \sqsubseteq n$*, if* $n \models \phi$*, then* $n \models \psi$
- $m \models \top$ *iff there exists* $S \in J(m)$ *such that for any* $m' \in S$, $e \sqsubseteq m'$
- $m \models \phi * \psi$ *iff there exists* $S \in J(m)$ *such that for any* $m' \in S$*, there exist* $n_\phi, n_\psi \in M$ *such that* $n_\phi \circ n_\psi \sqsubseteq m'$, $n_\phi \models \phi$ *and* $n_\psi \models \psi$
- $m \models \phi \mathrel{-\!\!*} \psi$ *iff for any* $n \in M$ *such that* $n \models \phi$, $m \circ n \models \psi$

We make the following important remark which will prove useful later: if a world m is *inconsistent*, *i.e.*, is such that $m \models \bot$, then, by the continuity axiom of J, for any world n, $m \circ n$ is also inconsistent.

Definition 4. *Bunches are given by the grammar:* $\Gamma ::= \phi \mid \emptyset_a \mid \Gamma ; \Gamma \mid \emptyset_m \mid \Gamma , \Gamma$. $\equiv$ *is given by commutative monoid equations for "," and ";", together with the evident substitution congruence for sub-bunches determined by the grammar. Let* $\mathcal{G}$ *be a GRM and* ϕ_Γ *be the formula obtained from a bunch* Γ *by replacing each ";" by* $\wedge$ *and each "," by* $*$ *with association respecting the tree structure of* Γ*, a sequent* $\Gamma \vdash \phi$ *is said to be* valid in $\mathcal{G}$*, written* $\Gamma \models_{\mathcal{G}} \phi$*, if and only if, for any world* $m \in M$, $m \models \phi_\Gamma$ *implies* $m \models \phi$*. A sequent* $\Gamma \vdash \phi$ *is* valid*, written* $\Gamma \models \phi$*, iff, for any GRM* $\mathcal{G}$*, it is valid in* $\mathcal{G}$.

Definition 5 (LBI). **BI***'s sequent calculus,* **LBI***, is defined as follows:*

$$\frac{}{\phi \vdash \phi}\ Axiom \qquad \frac{\Gamma \vdash \phi \quad \Delta(\phi) \vdash \psi}{\Delta(\Gamma) \vdash \psi}\ Cut \qquad \frac{\Gamma(\Delta) \vdash \phi}{\Gamma(\Delta; \Delta') \vdash \phi}\ W \qquad \frac{\Gamma(\Delta; \Delta) \vdash \phi}{\Gamma(\Delta) \vdash \phi}\ C$$

$$\frac{}{\Gamma(\bot) \vdash \phi}\ \bot L \qquad \frac{\Gamma \vdash \phi}{\Delta \vdash \phi}\ E \qquad \frac{\Gamma(\emptyset_m) \vdash \phi}{\Gamma(I) \vdash \phi}\ I\,L \qquad \frac{}{\emptyset_m \vdash I}\ I\,R$$

$$\frac{\Gamma(\emptyset_a) \vdash \phi}{\Gamma(\top) \vdash \phi}\ \top L \qquad \frac{}{\emptyset_a \vdash \top}\ \top R \qquad \frac{\Gamma \vdash \phi \quad \Delta(\Delta', \psi) \vdash \chi}{\Delta(\Delta', \Gamma, \phi \mathrel{-\!\!*} \psi) \vdash \chi}\ \mathrel{-\!\!*} L \qquad \frac{\Gamma, \phi \vdash \psi}{\Gamma \vdash \phi \mathrel{-\!\!*} \psi}\ \mathrel{-\!\!*} R$$

$$\frac{\Gamma(\phi, \psi) \vdash \chi}{\Gamma(\phi * \psi) \vdash \chi}\ * L \qquad \frac{\Gamma \vdash \phi \quad \Delta \vdash \psi}{\Gamma, \Delta \vdash \phi * \psi}\ * R \qquad \frac{\Gamma \vdash \phi \quad \Delta(\Delta' ; \psi) \vdash \chi}{\Delta(\Delta' ; \Gamma ; \phi \rightarrow \psi) \vdash \chi}\ \rightarrow L \qquad \frac{\Gamma ; \phi \vdash \psi}{\Gamma \vdash \phi \rightarrow \psi}\ \rightarrow R$$

$$\frac{\Gamma(\phi_1; \phi_2) \vdash \psi}{\Gamma(\phi_1 \wedge \phi_2) \vdash \psi}\ \wedge L \qquad \frac{\Gamma \vdash \phi \quad \Delta \vdash \psi}{\Gamma ; \Delta \vdash \phi \wedge \psi}\ \wedge R \qquad \frac{\Gamma(\phi) \vdash \chi \quad \Delta(\psi) \vdash \chi}{\Gamma(\phi \vee \psi); \Delta(\phi \vee \psi) \vdash \chi}\ \vee L \qquad \frac{\Gamma \vdash \phi_i\ (i = 1, 2)}{\Gamma \vdash \phi_1 \vee \phi_2}\ \vee R.$$

A proposition ϕ is a *theorem* of **LBI** iff $I \vdash \phi$. The Cut-elimination theorem holds for **LBI**, together with following results, all of which are proved in [13,14]: *soundness*, *completeness* and, as a corollary, *validity*, *i.e.*, a proposition ϕ is valid iff for any *GRM* $\mathcal{G}$, $e \models_{\mathcal{G}} \phi$.

3 Resource Tableaux for BI

We set up the theory of labelled semantic tableaux for **BI**. We assume a basic knowledge of tableaux systems [2]. We begin with algebras of labels, which provide the connection between the underlying syntactic tableaux and the semantics of the connectives used to regulate the multiplicative structure. In the case of $\mathbf{BI}^{\perp}$, we can provide an algebra which syntactically reflects the elementary semantics [4]. For **BI** and its Grothendieck topological semantics, the analysis is more delicate. A key step in this semantic analysis is the use of *dependency graphs*, explained in § 3.3.

3.1 A Labelling Algebra

We define a set of labels and constraints and a corresponding labelling algebra, *i.e.*, a preordered monoid whose elements are denoted by labels.

Definition 6. *A* labelling language *consists of the following symbols: a unit symbol* 1*, a binary function symbol* $\circ$*, a binary relation symbol* $\leq$*, a countable set of constants* $c_1, c_2, \ldots$. Labels *are inductively defined from the unit* 1 *and the constants as expressions of the form* $x \circ y$ *in which* x *and* y *are labels.* Atomic labels *are labels which do not contain any* $\circ$*, while* compound labels *contain at least one* $\circ$. Label constraints *are expressions of the form* $x \leq y$*, where* x *and* y *are labels.*

Definition 7. *Labels and constraints are interpreted in an order-preserving preordered commutative monoid of labels, or* labelling algebra $\mathcal{L} = \langle L, \circ, 1, \leq \rangle$*, more precisely: 1. L is a set of labels; 2.* $\leq$ *is a preorder; 3. Equality on labels is defined by :* $x = y$ *iff* $x \leq y$ *and* $y \leq x$*; 4.* $\circ$ *is a binary operation on L that s.t.: associativity:* $(x \circ y) \circ z = x \circ (y \circ z)$*, commutativity:* $x \circ y = y \circ x$*, identity:* $x \circ 1 = 1 \circ x = x$*, compatibility:* $x \circ z \leq y \circ z$ *if* $x \leq y$*. We say that* x *is a* sublabel *of* y *(notation:* $x \preceq y$*), if there exists a label* z *such that* $y = x \circ z$*. We say* $x \prec y$ *if* $x \preceq y$ *and* $x \neq y$*.* $\wp(x)$ *denotes the set of the sublabels of* x*.*

For notational simplicity, we can omit the binary symbol $\circ$ when writing labels. We deal with *partially defined* labelling algebras, obtained from sets of constraints by means of a *closure operator*.

Definition 8. *The* domain *of a set* K *of label constraints is the set of all sublabels occurring in some constraints of* K, i.e., $\mathcal{D}(K) = \bigcup_{x \leq y \in K} (\wp(x) \cup \wp(y))$*. The* closure $\overline{K}$ *of* K *is defined as follows:*

1. $K \subseteq \overline{K}$*;*
2. *Reflexivity: if* $x \in \mathcal{D}(\overline{K})$*, then* $x \leq x \in \overline{K}$*;*
3. *Transitivity: if* $x \leq y \in \overline{K}$ *and* $y \leq z \in \overline{K}$*, then* $x \leq z \in \overline{K}$*;*
4. *Compatibility: if* $x \circ z$ *or* $y \circ z \in \mathcal{D}(\overline{K})$*, then* $x \leq y \in \overline{K}$ *implies* $x \circ z \leq y \circ z \in \overline{K}$*.*

We do not distinguish between the closure of a set of label constraints and the (partially defined) labelling algebra it generates.

3.2 Expansion Rules

We can now define the expansion rules of **TBI**.

Definition 9. *A* signed formula *is a triple* $\langle Sg, \phi, l \rangle$*, denoted* $Sg\ \phi : l$*,* Sg *(*$\in \{F, T\}$*) being the sign of the formula* ϕ *(*$\in \mathcal{P}(L)$*) and* l *(*$\in \mathcal{L}$*) its label.*

Definition 10 (TBI). *A* **TBI** tableau t *is a rooted tree whose nodes are labelled with a signed formula and built according to the following* expansion rules*:*

$$\begin{array}{c} F\ \phi \vee \psi : x \\ \mid \\ F\ \phi : x \\ F\ \psi : x \end{array} \qquad \begin{array}{cc} \multicolumn{2}{c}{T\ \phi \vee \psi : x} \\ T\ \phi : x & T\ \psi : x \end{array} \qquad \begin{array}{c} T\ \phi \wedge \psi : x \\ \mid \\ T\ \phi : x \\ T\ \psi : x \end{array} \qquad \begin{array}{cc} \multicolumn{2}{c}{F\ \phi \wedge \psi : x} \\ F\ \phi : x & F\ \psi : x \end{array}$$

$$\begin{array}{c} F\ \phi \rightarrow \psi : x \\ \mid \\ \boxed{ass : x \leq c_i}\ ^* \\ \mid \\ T\ \phi : c_i \\ F\ \psi : c_i \end{array} \qquad \begin{array}{c} T\ \phi * \psi : x \\ \mid \\ \boxed{ass : c_i c_j \leq x}\ ^* \\ \mid \\ T\ \phi : c_i \\ T\ \psi : c_j \end{array} \qquad \begin{array}{c} F\ \phi \twoheadrightarrow \psi : x \\ \mid \\ T\ \phi : c_i\ ^* \\ F\ \psi : x c_i \end{array}$$

* c_i, c_j *are new constants*

$$\begin{array}{cc} \multicolumn{2}{c}{T\ \phi \twoheadrightarrow \psi : x} \\ F\ \phi : y & T\ \psi : xy \end{array} \qquad \begin{array}{cc} \multicolumn{2}{c}{T\ \phi \rightarrow \psi : x} \\ \multicolumn{2}{c}{\mid} \\ \multicolumn{2}{c}{\boxed{req : x \leq y}} \\ F\ \phi : y & T\ \psi : y \end{array} \qquad \begin{array}{cc} \multicolumn{2}{c}{F\ \phi * \psi : x} \\ \multicolumn{2}{c}{\mid} \\ \multicolumn{2}{c}{\boxed{req : yz \leq x}} \\ F\ \phi : y & F\ \psi : z \end{array}$$

Given a tableau branch $\mathcal{B}$, $\mathcal{F}(\mathcal{B})$ denotes the set of all its signed formulæ. Moreover, $\mathcal{B}$ is associated two particular sets of label constraints, $Ass(\mathcal{B})$, with elements "ass", and $Req(\mathcal{B})$, with elements "req", that are, respectively, the set of its assertions *and of its requirements, or* obligations. *The* domain $\mathcal{D}(\mathcal{B})$ *of a branch $\mathcal{B}$ is the set of all sublabels occuring in its assertions,* i.e., $\mathcal{D}(\mathcal{B}) = \mathcal{D}(Ass(\mathcal{B}))$. *$C(\mathcal{B})$ is the subset of all constants of $\mathcal{D}(\mathcal{B})$. The notation for branches extends to tableaux as follows: $f(t) = \bigcup_{\mathcal{B} \in t} f(\mathcal{B})$, where f is one of $\mathcal{F}$, Ass, Req, $\mathcal{D}$ or C.*

The rules for $\wedge$ and $\vee$ are the usual α,β ones. Those introducing *assertions* including $F\!\twoheadrightarrow$ for which the assertion $c_i \leq c_i$ is implicitly assumed are called $\pi\alpha$ and those introducing *requirements* are called $\pi\beta$. Notice also that $\pi\alpha$ rules create new (atomic) labels while $\pi\beta$ reuse existing ones.

Definition 11. *Let ϕ be a* **BI** *proposition. A* tableau sequence for ϕ *is a sequence of tableaux $t_1, t_2, \ldots$ for which t_1 is the one-node tree defined by $\mathcal{F}(t_1) = \{F\ \phi : 1\}$, $Ass(t_1) = \{1 \leq 1\}$, $Req(t_1) = \emptyset$ and t_{i+1} is obtained from t_i by applying, on a branch of t_i, an expansion rule of Definition 10.*

Definition 12. *Two signed formulæ $T\ \phi : x$, $F\ \phi : y$ are* complementary *in a branch $\mathcal{B}$ if and only if $x \leq y \in \overline{Ass}(\mathcal{B})$,* i.e., *if the constraint $x \leq y$ belongs to the reflexive, transitive and compatible closure of the assertions of $\mathcal{B}$.*

So far, the definitions, as well as the expansion rules, were exactly the same as the one presented in [4] for **BI** without $\perp$ and its elementary semantics. As we mentioned in the introduction, we aim to address the problem of inconsistency while keeping as much of the initial tableau system as possible. Therefore, we will not derive new expansion rules for $\perp$, rather, we will extend the definition of a closed tableau with an additional condition which takes the specificity of $\perp$ into account. This new condition introduces the notion of an *inconsistent* label which syntactically reflects the fact that Grothendieck models may have several worlds at which $\perp$ is forced and, as noticed in the remark following Definition 3, that compositions with such worlds are themselves inconsistent. The crucial point here is that, since the case of $\perp$ is handled via the closure rule *by sole considerations on the labels*, proving a formula of full propositional **BI**, compared to **BI** without $\perp$, is only a matter of deciding when a branch containing $\perp$ should be considered closed or not, but the procedures which actually build the tableaux, dependency graphs, as well as the related properties (termination, finiteness) remain unchanged.

3.3 Resource Tableaux with $\perp$ and Dependency Graphs

Definition 13. *Let $\mathcal{B}$ be a branch, a label x is* inconsistent in $\mathcal{B}$ *if there exists a label y such that $y \leq x \in \overline{Ass}(\mathcal{B})$ and a label z in $\wp(y)$ (set of sub-labels of y) such that $T\ \perp : z$ occurs in $\mathcal{B}$. A label x is* consistent *in $\mathcal{B}$ if it is not inconsistent.*

Definition 14. *A tableau t is* closed *if, for all its branches $\mathcal{B}$, (i)*

1. *there are two formulæ $T\ \phi : x$ and $F\ \phi : y$ that are complementary in $\mathcal{B}$, or*
2. *there is $F\ \top : x$ in $\mathcal{B}$, or*
3. *there is $F\ I : x$ in $\mathcal{B}$ with $1 \leq x \in \overline{Ass}(\mathcal{B})$, or*
4. *there is $T\ I : x$ in $\mathcal{B}$ with $1 \leq x \notin \overline{Ass}(\mathcal{B})$, or*
5. *there is $F\ \phi : x$ in $\mathcal{B}$ with x inconsistent in $\mathcal{B}$;*

and (ii) $\forall\, x \leq y \in Req(\mathcal{B})$, $x \leq y \in \overline{Ass}(\mathcal{B})$.
A tableau sequence $t_1, t_2, \ldots$ is closed *if it contains a closed tableau.*

A specific graph, called the dependency graph or Kripke resource graph, is built in parallel with the tableau expansion. It reflects the information that can be derived from a given set of assertions.

Definition 15. *Given a tableau branch $\mathcal{B}$, the associated dependency graph $DG(\mathcal{B}) = [N(\mathcal{B}), A(\mathcal{B})]$ is defined as the following directed graph: the set of nodes $N(\mathcal{B})$ is the set of labels $\mathcal{D}(\mathcal{B})$ and the set of arrows $A(\mathcal{B})$ is built from the set of assertions $\overline{Ass}(\mathcal{B})$ as follows: there is an arrow $x \rightarrow y$ in $A(\mathcal{B})$ iff there is an assertion $x \leq y$ in $\overline{Ass}(\mathcal{B})$.*

We can formally define a procedure that builds, in parallel with tableau expansions, the dependency graph $DG(\mathcal{B})$ of a branch $\mathcal{B}$ and so, the closure $\overline{Ass}(\mathcal{B})$. The expansion rules of a dependency graph are such that the given graph is only expanded by the $\pi\alpha$ rules, all the other rules, introducing neither new constants, nor new assertions, simply leave it unchanged. On a dependency graph $DG(\mathcal{B})$, the fact that a requirement $x \leq y$ holds with respect to $\overline{Ass}(\mathcal{B})$ corresponds to the existence of a path from the node x to the node y. We illustrate this point with two examples.

Figure 1 shows a closed tableau for the formula $((\mathrm{p} \mathrel{-\!\!*} \perp) * \mathrm{p}) \rightarrow \mathrm{q}$, which is therefore provable in **BI**. We remark that we reach, after step 3, a tableau with two branches. The first branch is closed since it contains complementary formulæ, namely, $\langle T\ \mathrm{p} : c_3, F\ \mathrm{p} : c_3 \rangle$. The second, however, contains no complementary formulæ. It is

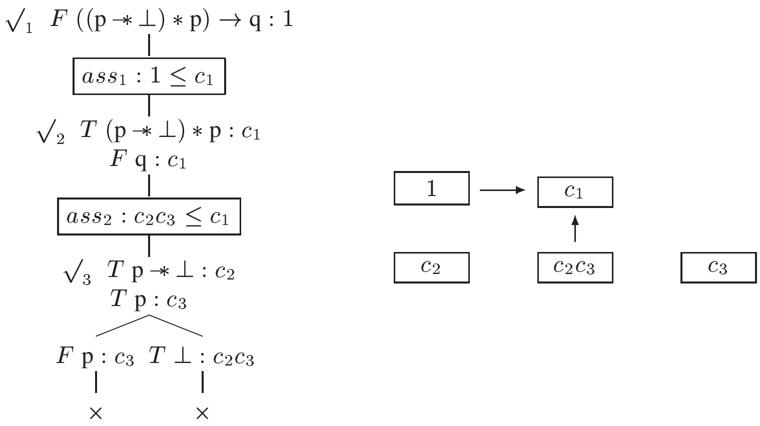

Figure1. Tableau and Dependency Graph for $((\mathrm{p} \mathrel{-\!\!*} \bot) * \mathrm{p}) \to \mathrm{q}$

the point were the closure condition plays its role. We notice that the branch contains the formula $T\ \bot : c_2c_3$. Thus, c_2c_3 is what we have called an inconsistent label and, by assertion $ass_2 : c_2c_3 \leq c_1$, c_1 is also inconsistent. Therefore, the branch is closed because it contains the formula $F\ \mathrm{q} : c_1$ with label c_1 being inconsistent.

The second example, *q.v.* Figure 2, leads to an unclosed tableau for $((\mathrm{p} \mathrel{-\!\!*} \bot) \to \bot) \mathrel{-\!\!*} (((\mathrm{p} * \mathrm{p}) \mathrel{-\!\!*} \bot) \to \bot)$ which is therefore unprovable. After step 6, the tableau is completed and we are left with four branches to close. The second one is closed with $\langle\, T\ \mathrm{p} : c_3, F\ \mathrm{p} : c_3\,\rangle$, the third is closed with $\langle\, T\ \bot : c_2c_3, F\ \bot : c_2c_3\,\rangle$ and the fourth is closed with $\langle\, T\ \bot : c_2, F\ \bot : c_2\,\rangle$. The first branch, on the contrary, remains open since the only way to close it would be to have $\langle\, T\ \mathrm{p} : c_3, F\ \mathrm{p} : 1\,\rangle$, but $c_3 \leq 1$ cannot be deduced from the assertions of the branch. We will see in a next section how to build a countermodel from the open branch.

We now show that this labelled calculus, whose restriction to **BI**$^{\bot}$ is complete for the elementary semantics, is complete for **BI** w.r.t. the Grothendieck topological semantics.

4 Completeness of the TBI Calculus

We show the soundness and completeness of **TBI** w.r.t. GRMs. This deductive framework allows not only a proof procedure but also, in the case of non-provability, the systematic generation of countermodels.

4.1 Soundness

Soundness is proved in a classical way, subject to the usual adaptations to **BI** [13,14], from a notion of *realizability* that is preserved by the expansion rules [4].

Definition 16. *Let $\mathcal{G} = \langle\, \mathcal{M}, \models, [\![-]\!]\,\rangle$ be a GRM and $\mathcal{B}$ be a tableau branch, a* realization of $\mathcal{B}$ in $\mathcal{G}$ *is a mapping $\|-\| : \mathcal{D}(\mathcal{B}) \to M$, from the domain of $\mathcal{B}$ to the worlds of M, that satisfies*

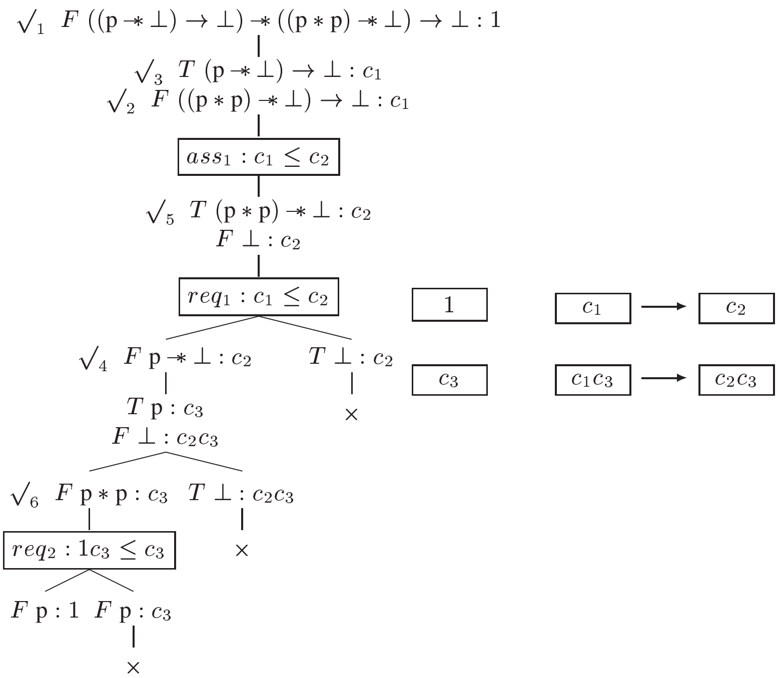

Figure2. Tableau and Dependency Graph for $((\mathrm{p} \multimap \bot) \to \bot) \multimap (((\mathrm{p} * \mathrm{p}) \multimap \bot) \to \bot)$

1. $\|1\| = e$,
2. $\|x \circ y\| = \|x\| \circ \|y\|$,
3. *for any* $T\ \phi : x$ *in* $\mathcal{B}$, $\|x\| \models \phi$,
4. *for any* $F\ \phi : x$ *in* $\mathcal{B}$, $\|x\| \not\models \phi$,
5. *for any* $x \leq y$ *in* $Ass(\mathcal{B})$, $\|x\| \sqsubseteq \|y\|$.

Lemma 1. *Let t be a tableau,* $\mathcal{B}$ *a branch of t and* $\|-\|$ *a realization of* $\mathcal{B}$ *in a GRM* $\mathcal{G}$. *Then, for any* $x \leq y \in \overline{Ass}(\mathcal{B})$, $\|x\| \sqsubseteq \|y\|$ *holds in* $\mathcal{G}$.

Definition 17. *A tableau branch* $\mathcal{B}$ *is* realizable *if there exists a realization of* $\mathcal{B}$ *in some GRM* $\mathcal{G}$. *A tableau t is* realizable *if it contains a realizable branch.*

Lemma 2. *A closed tableau is not realizable.*

Proof Let t be a closed tableau that is also realizable. Then, t contains a branch $\mathcal{B}$ which is realizable in some *GRM* $\mathcal{G} = \langle \mathcal{M}, \models, [\![-]\!] \rangle$. If the branch is closed because of complementary formulæ $\langle T\ \phi : x, F\ \phi : y \rangle$ then, by definition, we have $x \leq y \in \overline{Ass}(\mathcal{B})$ which, by Lemma 1, implies $\|x\| \sqsubseteq \|y\|$. But, since $\|-\|$ realizes $\mathcal{B}$, we also have $\|x\| \models \phi$ and $\|y\| \not\models \phi$. Therefore, we reach a contradiction because, by property (K), we should have $\|y\| \models \phi$. If the branch is closed because of a formula $F\ \phi : x$, whose label x is inconsistent in $\mathcal{B}$, then, by definition, there exists a label y such that $y \leq x \in \overline{Ass}(\mathcal{B})$ and a label z in $\wp(y)$ such that $T \bot : z$. Since $\|-\|$ realizes $\mathcal{B}$ we have

$x \not\models \phi$ and $z \models \bot$. Since x is a sublabel of y, the continuity axiom of J implies that $y \models \bot$. Therefore, as Lemma 1 implies $\|y\| \sqsubseteq \|x\|$, (K) yields $x \models \bot$ and, once again, we reach a contradiction because, if $x \models \bot$ then, for any ϕ, we should have $x \models \phi$. Other cases are similar. □

Theorem 1 (soundness). *Let ϕ be a proposition of* **BI**. *If there exists a closed tableau sequence $\mathcal{T}$ for ϕ, then ϕ is valid in Grothendieck topological semantics.*

4.2 Countermodel Construction

We describe how to construct a countermodel of ϕ from an open branch in a tableau for ϕ. We obtain the finite model property and decidability for **BI**. The proof of the finite model property relies critically on the introduction of a special element, here called π, used to collect the inessential (and possibly infinite) parts of the model.

Definition 18. *Let $\mathcal{B}$ be a tableau branch. A signed formula $Sg\ X : x$ is* fulfilled, *or* completely analysed, *in $\mathcal{B}$, denoted $\mathcal{B} \Vdash Sg\ X : x$, if it satisfies one of the following conditions:*

1. $\mathcal{B} \Vdash F\ \bot : x$;
2. $\mathcal{B} \Vdash F\ I : x$ *iff* $1 \leq x \notin \overline{Ass}(\mathcal{B})$;
3. $\mathcal{B} \Vdash F\ \mathrm{p} : x$ *iff there is* $F\ \mathrm{p} : y \in \mathcal{B}$ *s.t.* $x \leq y \in \overline{Ass}(\mathcal{B})$;
4. $\mathcal{B} \Vdash F\ \phi \wedge \psi : x$ *iff* $\mathcal{B} \Vdash F\ \phi : x$ *or* $\mathcal{B} \Vdash F\ \psi : x$;
5. $\mathcal{B} \Vdash F\ \phi \vee \psi : x$ *iff* $\mathcal{B} \Vdash F\ \phi : x$ *and* $\mathcal{B} \Vdash F\ \psi : x$;
6. $\mathcal{B} \Vdash F\ \phi \to \psi : x$ *iff there is* $y \in \mathcal{D}(\mathcal{B})$ *s.t.* $x \leq y \in \overline{Ass}(\mathcal{B})$ *and both* $\mathcal{B} \Vdash T\ \phi : y$ *and* $\mathcal{B} \Vdash F\ \psi : y$;
7. $\mathcal{B} \Vdash F\ \phi * \psi : x$ *iff for any* y, $z \in \mathcal{D}(\mathcal{B})$ *s.t.* $yz \leq x \in \overline{Ass}(\mathcal{B})$ $\mathcal{B} \Vdash F\ \phi : y$ *or* $\mathcal{B} \Vdash F\ \psi : z$;
8. $\mathcal{B} \Vdash F\ \phi \mathrel{-\!\!*} \psi : x$ *iff there exist* y, $xy \in \mathcal{D}(\mathcal{B})$ *s.t. both* $\mathcal{B} \Vdash T\ \phi : y$ *and* $\mathcal{B} \Vdash F\ \psi : xy$;
9. $\mathcal{B} \Vdash T\ \top : x$;
10. $\mathcal{B} \Vdash T\ I : x$ *iff* $1 \leq x \in \overline{Ass}(\mathcal{B})$;
11. $\mathcal{B} \Vdash T\ \mathrm{p} : x$ *iff there is* $T\ \mathrm{p} : y \in \mathcal{B}$ *s.t.* $y \leq x \in \overline{Ass}(\mathcal{B})$;
12. $\mathcal{B} \Vdash T\ \phi \wedge \psi : x$ *iff both* $\mathcal{B} \Vdash T\ \phi : x$ *and* $\mathcal{B} \Vdash T\ \psi : x$;
13. $\mathcal{B} \Vdash T\ \phi \vee \psi : x$ *iff* $\mathcal{B} \Vdash T\ \phi : x$ *or* $\mathcal{B} \Vdash T\ \psi : x$;
14. $\mathcal{B} \Vdash T\ \phi \to \psi : x$ *iff for any* $y \in \mathcal{D}(\mathcal{B})$ *s.t.* $x \leq y \in \overline{Ass}(\mathcal{B})$ $\mathcal{B} \Vdash F\ \phi : y$ *or* $\mathcal{B} \Vdash T\ \psi : y$;
15. $\mathcal{B} \Vdash T\ \phi * \psi : x$ *iff there are* y, $z \in \mathcal{D}(\mathcal{B})$ *s.t.* $yz \leq x \in \overline{Ass}(\mathcal{B})$ *and both* $\mathcal{B} \Vdash T\ \phi : y$ *and* $\mathcal{B} \Vdash T\ \psi : z$;
16. $\mathcal{B} \Vdash T\ \phi \mathrel{-\!\!*} \psi : x$ *iff for any* y, $xy \in \mathcal{D}(\mathcal{B})$ $\mathcal{B} \Vdash F\ \phi : y$ *or* $\mathcal{B} \Vdash T\ \psi : xy$.

Lemma 3. *Let $\mathcal{B}$ be a tableau branch. The property of being fulfilled, given in Definition 18, satisfies Kripke monotonicity, more precisely, (i) $\mathcal{B} \Vdash F\ \phi : x$ and $y \leq x \in \overline{Ass}(\mathcal{B})$ imply $\mathcal{B} \Vdash F\ \phi : y$, and (ii) $\mathcal{B} \Vdash T\ \phi : x$ and $x \leq y \in \overline{Ass}(\mathcal{B})$ imply $\mathcal{B} \Vdash T\ \phi : y$.*

Definition 19. *A tableau branch $\mathcal{B}$ is* completed *if any signed formula $Sg\ \phi : x$ in $\mathcal{B}$ is fulfilled. A tableau is* completed *if it has a branch that is completed. A tableau branch $\mathcal{B}$ is an* H-branch *if it is open and completed.*

Lemma 4. *If $\mathcal{B}$ is an H-branch then, for any proposition ϕ, not both $\mathcal{B} \Vdash T\ \phi : x$ and $\mathcal{B} \Vdash F\ \phi : x$.*

The dependency graph related to a formula ϕ during the resource tableau construction represents the closure of the assertions in the sense of Definition 8 and so captures the computational content of ϕ. Therefore, if a formula ϕ happens to be unprovable, we should have enough information in its dependency graph to extract a countermodel for ϕ. For that, we must provide a preordered commutative monoid together with a Grothendieck topology and a forcing relation which falsifies ϕ in some world. The idea behind the countermodel construction is to regard the dependency graph itself as the desired countermodel, thereby considering it as a central semantical structure. For that, we take the nodes (labels) of the graph as the elements of a monoid whose composition law is given by the composition of the labels. The preordering relation is then given by the arrows and the forcing relation simply reflects the property of being fulfilled.

The key problem is that, since the closure operator induces a *partially defined* labelling algebra, the dependency graph only deals with those pieces of information (resources) that are relevant for deciding provability. Therefore, the monoidal law should be completed with suitable values for those compositions which are undefined. The problem of undefinedness is solved in Definition 20 by the introduction of a particular element, denoted π, to which all undefined compositions are mapped and for which the equation $(\forall x)(x \circ \pi = \pi \circ x = \pi)$, meaning that any composition with something undefined is itself undefined, is assumed.

However, we must be careful because introducing a new element may affect the property of a formula $\phi \twoheadrightarrow \psi$ of being realized in a world x although the signed formula $T\ \phi \twoheadrightarrow \psi : x$ was fulfilled in the dependency graph. Indeed, if π forces ϕ then, since $x \circ \pi = \pi$, we also need π to force ψ. But, if π forces any formula ψ, then everything works as it should. On the other hand, we know that an inconsistent world necessarily forces any formula ψ because $\perp \vdash \psi$ is an axiom. Therefore, making π an inconsistent world by setting $\emptyset \in J(\pi)$ just solves the problem.

Definition 20 **(M-structure, π).** *Let $\mathcal{B}$ be an H-branch. The M-structure $\mathcal{M}(\mathcal{B}) = \langle M, \circ, 1, \sqsubseteq, J \rangle$ is defined as follows: (i) M is the subset of labels of $\mathcal{D}(\mathcal{B})$ consistent in $\mathcal{B}$, extended with a particular element π; (ii) $\circ$ is a composition law defined by*

$$x \circ 1 = 1 \circ x = x \qquad x \circ y = y \circ x = \begin{cases} xy & \textit{if } xy \in M \\ \pi & \textit{otherwise,} \end{cases}$$

and (iii) the relation $\sqsubseteq$ between elements of M is defined by

$$x \sqsubseteq y \quad \textit{iff} \quad y = x = \pi \ \textit{or} \ x \leq y \in \overline{Ass}(\mathcal{B})$$

(iv) the map $J : M \to \wp(\wp(M))$, called the J-map of $\mathcal{B}$, is defined by

$$J(\pi) = \{\{\pi\}, \emptyset\} \quad J(x) = \{\{y\} \mid y = x\}$$

Lemma 5. *Let $\mathcal{B}$ be an H-branch, the M-structure $\mathcal{M}(\mathcal{B}) = \langle M, \sqsubseteq, \circ, 1, J \rangle$ is a GTM,* i.e.*, (i) $\langle M, \sqsubseteq, \circ, 1 \rangle$ is a preordered commutative monoid, and (ii) J is a Grothendieck topology.*

Definition 21. *Let $\mathcal{M}(\mathcal{B}) = \langle M, \sqsubseteq, \circ, 1, J \rangle$ be the M-structure of an H-branch $\mathcal{B}$ and $\mathcal{P}(L)$ denote the collection of* **BI** *propositions over a language L of propositional letters. The interpretation $[\![-]\!]_{\mathcal{B}} : L \to \wp(M)$ is, for any atomic proposition* p*, $[\![\mathrm{p}]\!]_{\mathcal{B}} = \{\pi\} \cup \{x \mid \mathcal{B} \Vdash T\ \mathrm{p} : x\}$.*

Lemma 6. $[\![-]\!]_\mathcal{B}$ *is a GTI, i.e., it satisfies properties (K) and (Sh) of Definition 2.*

Theorem 2. *Let $\mathcal{B}$ be an H-branch. Then $\langle\, \mathcal{M}(\mathcal{B}), \models, [\![-]\!]_\mathcal{B} \,\rangle$ is a Grothendieck resource model of $\mathcal{B}$, i.e., for any proposition ϕ, we have: (i) $\pi \models \phi$; (ii) $\mathcal{B} \Vdash T\ \phi : x$ implies $x \models \phi$; (iii) $\mathcal{B} \Vdash F\ \phi : x$ implies $x \not\models \phi$.*

Returning to the example of Figure 2, we show how to build a countermodel from the open branch. As the reader might check, all formulæ in the open branch are fulfilled and $\mathcal{B}$ is therefore what we have called an H-branch. Firstly, following the steps of Definition 20, we build from $\mathcal{B}$ a GTM $\mathcal{M}(\mathcal{B}) = \langle\, M, \circ, 1, \sqsubseteq, J \,\rangle$.

(i) M is the subset of labels of $\mathcal{D}(\mathcal{B})$ that are consistent, to which we add the element π, *i.e.*, $M = \{1, c_1, c_2, c_3, c_1c_3, c_2c_3, \pi\}$. Notice that, because of the presence in $\mathcal{B}$ of both the assertion $ass_1 : c_1 \leq c_2$ and of the label c_2c_3, the label c_1c_3, although not initially present in $\mathcal{B}$, is added by the closure operation in order to respect the compatibility requirement.

(ii) the multiplication $\circ$ is

$\circ$	1	c_1	c_2	c_3	c_1c_3	c_2c_3	π
1	1	c_1	c_2	c_3	c_1c_3	c_2c_3	π
c_1	c_1	π	π	c_1c_3	π	π	π
c_2	c_2	π	π	c_2c_3	π	π	π
c_3	c_3	c_1c_3	c_2c_3	π	π	π	π
c_1c_3	c_1c_3	π	π	π	π	π	π
c_2c_3	c_2c_3	π	π	π	π	π	π
π	π	π	π	π	π	π	π

(iii) The preordering relation $\sqsubseteq$ reflects the structure of the assertions $\overline{Ass}(\mathcal{B})$. If we omit implicit reflexive relations, we have two non-trivial relations, namely, $c_1 \sqsubseteq c_2$ and $c_1c_3 \sqsubseteq c_2c_3$.

(iv) The Grothendieck topology J is given by the following table:

x	1	c_1	c_2	c_3	c_1c_3	c_2c_3	π
$J(x)$	$\{\{1\}\}$	$\{\{c_1\}\}$	$\{\{c_2\}\}$	$\{\{c_3\}\}$	$\{\{c_1c_3\}\}$	$\{\{c_2c_3\}\}$	$\{\{\pi\}, \emptyset\}$

Secondly, we apply Definition 21 to the only atomic proposition p occuring in the branch $\mathcal{B}$, which leads to the GTI $[\![\mathrm{p}]\!]_\mathcal{B} = \{\pi, c_3\}$. This, in turn, finally gives rise to the GRM $\mathcal{G} = \langle\, \mathcal{M}(\mathcal{B}), \models, [\![-]\!]_\mathcal{B} \,\rangle$, the desired countermodel. Now we check that (i) $c_1 \models (\mathrm{p} \mathrel{-\!\!*} \bot) \to \bot$ and (ii)) $c_1 \not\models ((\mathrm{p} * \mathrm{p}) \mathrel{-\!\!*} \bot) \to \bot$. For (i), we have $c_3 \models \mathrm{p}$ because $c_3 \in [\![\mathrm{p}]\!]_\mathcal{B}$ and $c_2c_3 \not\models \bot$ because $\emptyset \notin J(c_2c_3)$. Thus, $c_2 \not\models \mathrm{p} \mathrel{-\!\!*} \bot$ and, since $c_1 \sqsubseteq c_2$ we obtain, by (K), $c_1 \not\models \mathrm{p} \mathrel{-\!\!*} \bot$. Therefore, we have $c_1 \models (\mathrm{p} \mathrel{-\!\!*} \bot) \to \bot$. For (ii), we notice that π is the only world that forces $\mathrm{p} * \mathrm{p}$. Thus, we have $c_2 \models (\mathrm{p} * \mathrm{p}) \mathrel{-\!\!*} \bot$ only if $c_2 \circ \pi \models \bot$, which is the case because $c_2 \circ \pi = \pi$ and $\pi \models \bot$. Note that it would not be the case in the elementary semantics for which no world can force $\bot$. On the other hand, $c_2 \not\models \bot$ because $\emptyset \notin J(c_2)$. Therefore, $c_1 \not\models ((\mathrm{p} * \mathrm{p}) \mathrel{-\!\!*} \bot) \to \bot$. Then the initial formula, although valid in the elementary semantics, is not provable in **BI** .

4.3 Completeness and Finite Model Property

A tableau construction procedure is an algorithm which, given a formula ϕ, builds a tableau sequence $t_1, t_2, \ldots, t_n$ until there exists a tableau t_i which is either closed or has an H-branch: Otherwise it does not terminate. **BI** has such a procedure, with $F\ \phi : 1$ as initial formula. Until $\mathcal{T}$ is closed or completed, choose an open branch $\mathcal{B}$; if there is an unfulfilled α or $\pi\alpha$ formula $(Sg\ \phi : x)$ in $\mathcal{B}$, then apply the related expansion rule; else if there is an unfulfilled β or $\pi\beta$ formula $(Sg\ \phi : x)$ in $\mathcal{B}$, then apply the corresponding expansion rule, with all labels for which the formula is not fulfilled.

When $\pi\alpha$ formulae are in the scope of $\pi\beta$ formulae, the construction of an H-branch can sometimes be infinite. It is because the fulfillment of $\pi\alpha$ formulae requires the introduction of new constants which may destroy the fulfillment of $\pi\beta$ formulae. In order to ensure termination, we need to control the introduction of constants and detect looping expansion sequences. For example, $F\ \phi \to \psi : x$ can simply be expanded in $F\ \psi : x$ when the branch $\mathcal{B}$ already contains $T\ \phi : y$ such that $y \leq x \in \overline{Ass}(\mathcal{B})$. Similar considerations apply to $T\ \phi * \psi : x$.

Looping is captured by a specific notion of *redundancy* which prevents expansions of the form $F\ \phi \multimap\!\!* \psi : x$ when x already contains a constant deriving from a previous occurence of the same signed formula. This notion was introduced in [4].

With these few technical improvements we can transform the semi-decision procedure into a decision procedure that terminates either with a closed tableau or with a finite H-branch. From such a branch, we can build a countermodel following Definition 20, and thus prove completeness, following an approach based on proof-search [11]. Moreover, as π captures the inessential parts of the model, the construction explained in Definition 20 always results in a finite countermodel when the corresponding H-branch is finite, yielding the finite model property.

Theorem 3 (completeness). *If $I \models \phi$, then there is a closed tableau sequence for ϕ.*

Theorem 4 (finite model property). *If $I \not\vdash \phi$, then, there is a finite Grothendieck model such that $I \not\models \phi$.*

Corollary 1 (decidability). *Propositional* **BI** *is decidable.*

Note that full propositional linear logic, with exponentials, is undecidable even when restricted to the intuitionistic fragment, that the status of **MELL** is unknown, and that neither has the finite model property [6,7].

From the capture of semantics by labels, we provide a decision procedure for **BI** that builds countermodels in Grothendieck topological semantics. Their study gives us a better understanding of the semantical information necessary to analyse provability and of the relationships between the elementary and topological settings. As a consequence, we present in the next section, a new powerful result about **BI** semantics that generalizes previous works on pointer logic.

5 A New (Complete) Resource Semantics

In § 4, we have analysed how countermodels could be built from dependency graphs. We now observe that those models are very closely related to the ones recently proposed in the semantics of "pointer logic" [8,13]. Indeed, the Grothendieck topology described in [13] exactly corresponds to our definition of the J-map. Moreover, in our models, a special element called π is used to capture undefinedness as the image of all undefined compositions and is the only one to force $\perp$ (because $\emptyset$ only belongs to $J(\pi)$).

A consequence of the completeness result for **TBI** (see Theorem 3) is that we can always restrict to such simple Grothendieck models and thus, obtain the completeness of **BI** with respect to a new Kripke resource semantics that is intermediate between the elementary and Grothendieck semantics. Let us sketch this new semantics.

Definition 22. *A* Kripke Resource Monoid*, or KRM for short, is a preordered commutative monoid* $\mathcal{M} = \langle M, \circ, e, \sqsubseteq \rangle$ *in which* M *contains an element, denoted* π*, such that for any* $m \in M$*,* $\pi \circ m = \pi$ *and in which* $\circ$ *is functorial with respect to* $\sqsubseteq$*.*

Definition 23. *Let* $\mathcal{M}$ *be a KRM and* $\mathcal{P}(L)$ *be a language of* **BI** *propositions over a language* L *of propositional letters. Then, a* Kripke resource interpretation*, or KRI, is a function* $[\![-]\!] : L \to \wp(M)$ *satisfying Kripke monotonicity and such that for any* $\mathrm{p} \in L$*,* $\pi \in [\![\mathrm{p}]\!]$.

Definition 24. *A* Kripke Resource Model *is a triple* $\mathcal{K} = \langle \mathcal{M}, \models, [\![-]\!] \rangle$ *in which* $\mathcal{M}$ *is a KRM,* $[\![-]\!]$ *is a KRI and* $\models$ *is a forcing relation on* $M \times \mathcal{P}(L)$ *satisfying the following conditions:*

- $m \models \mathrm{p}$ *iff* $m \in [\![\mathrm{p}]\!]$
- $m \models \top$ *iff always*
- $m \models \bot$ *iff* $m = \pi$
- $m \models \phi \wedge \psi$ *iff* $m \models \phi$ *and* $m \models \psi$
- $m \models \phi \vee \psi$ *iff* $m \models \phi$ *or* $m \models \psi$
- $m \models \phi \to \psi$ *iff for all* $n \in M$ *such that* $m \sqsubseteq n$*, if* $n \models \phi$*, then* $n \models \psi$
- $m \models I$ *iff* $e \sqsubseteq m$ *or* $m = \pi$
- $m \models \phi * \psi$ *iff there exist* $n_\phi, n_\psi \in M$ *such that* $n_\phi \circ n_\psi \sqsubseteq m$*,* $n_\phi \models \phi$ *and* $n_\psi \models \psi$
- $m \models \phi \mathrel{-\!\!*} \psi$ *iff for all* $n \in M$ *such that* $n \models \phi$*,* $m \circ n \models \psi$*.*

Definition 25. *A Grothendieck resource model* $\langle (M, \sqsubseteq, \circ, e, J), \models_{\mathcal{G}}, [\![-]\!]_{\mathcal{G}} \rangle$ *is* basic *iff* $\circ$ *is functorial w.r.t* $\sqsubseteq$*,* M *contains an element* π *such that for any* $m \in M$*,* $\pi \circ m = \pi$ *and* J *is basic,* i.e.*, is given by* $J(m) = \{\{m\}\}$ *if* $m \neq \pi$ *and* $J(\pi) = \{\{\pi\}, \emptyset\}$.

Lemma 7. *The class of Kripke resource models coincides with the class of basic Grothendieck resource models.*

Proof Let $\mathcal{G}$=$\langle (M, \sqsubseteq, \circ, e, J), \models_{\mathcal{G}}, [\![-]\!] \rangle$ be a basic Grothendieck model. We prove that $\langle (M, \sqsubseteq, \circ, e), \models_{\mathcal{G}}, [\![-]\!] \rangle$ is a Kripke model. Since $\mathcal{G}$ is basic, we simply show that $\models_{\mathcal{G}}$ satisfies the conditions of Definition 24. For simplicity, we write $\models$ instead of $\models_{\mathcal{G}}$. In the case of $\bot$, since $\emptyset$ only belongs to $J(\pi)$, the condition $\emptyset \in J(m)$ is equivalent to $m = \pi$. Now, for any world $m \neq \pi$, we have $J(m) = \{\{m\}\}$. Thus, in the case of I, the condition $(\exists S \in J(m))\ (\forall m' \in S)\ (e \sqsubseteq m')$ simplifies to $(\forall m' \in \{m\})\ (e \sqsubseteq m')$, which is equivalent to $e \sqsubseteq m$. The cases of $\vee$ and $*$ are similar. Conversely, endowing a Kripke model $\langle (M, \sqsubseteq, \circ, e), \models_{\mathcal{K}}, [\![-]\!] \rangle$ with the basic topology turns it into a basic Grothendieck model. Indeed, a short calculation shows, for such a J, that (K) for $[\![-]\!]$ implies (Sh). □

We have seen, in the semantics presented above, that π internalizes undefinedness and thus, corresponds to an alternative way of dealing with $\bot$ by considering a partially defined monoid, in which $\circ$ is a partial operation. Hence, we obtain a semantics which directly generalizes that taken in the analysis of pointer logic, in which the resource is computer memory, thereby emphasizing its utility in our analysis of resource:

- $m \models \perp$ iff never
- $m \models I$ iff $e \sqsubseteq m$
- $m \models \phi * \psi$ iff there exist $n, n' \in M$ such that $n \circ n' \downarrow$, $n \circ n' \sqsubseteq m$ $n \models \phi$ and $n' \models \psi$
- $m \models \phi \mathrel{-\!\!*} \psi$ iff for all $n \in M$ such that $n \models \phi$, $m \circ n \downarrow$ implies $m \circ n \models \psi$

where $\downarrow$ denotes definedness.

Theorem 5. **BI** *is sound and complete w.r.t. this "partial monoid" resource semantics.*

Proof The soundness is obvious since Grothendieck models include Kripke models. Turning to completeness, suppose that $I \not\models \phi$ then, by Theorem 3, there exists a tableau containing a H-branch from which one can construct a basic *GRM* which is a countermodel of ϕ following Definition 20. Lemma 7 then yields the corresponding Kripke countermodel for ϕ. Thus we observe that dependency graphs can be seen directly as countermodels in this new semantics. □

References

1. V. Balat and D. Galmiche. *Labelled Deduction*, volume 17 of *Applied Logic Series*, chapter Labelled Proof Systems for Intuitionistic Provability. Kluwer Academic Publishers, 2000.
2. M. Fitting. *First-Order Logic and Automated Theorem Proving*. Texts and Monographs in Computer Science. Springer Verlag, 1990.
3. D.M. Gabbay. *Labelled Deductive Systems*. OUP, 1996.
4. D. Galmiche and D. Méry. Proof-search and countermodel generation in propositional BI logic - extended abstract -. In *4th Int. Symposium on Theoretical Aspects of Computer Software, TACS 2001, LNCS 2215*, 263–282, Sendai, Japan, 2001. Full version submitted.
5. J. Harland and D. Pym. Resource-distribution via Boolean Constraints (Extended Abstract). In *14th Int. Conference on Automated Deduction, CADE-12, LNAI 814*, 222–236, Townsville, Queensland, Australia, July 1997. Full version to appear ACM ToCL, 2002.
6. Y. Lafont. The finite model property for various fragments of linear logic. *J. Symb. Logic* 62(4):1202–1208, 1997.
7. P. Lincoln. Deciding provability of linear logic formulas. In *Advances in Linear Logic*, J.-Y.Girard, Y. Lafont and L. Regnier (editors), Cambridge Univ. Press, 1995, 109–122.
8. S. Ishtiaq and P. O'Hearn. BI as an assertion language for mutable data structures. In *28th ACM Symp. on Principles of Prog. Langs., POPL 2001*, 14–26, London, UK, 2001.
9. P. O'Hearn and J. Reynolds and H. Yang Local Reasoning about Programs that Alter Data Structures. LNCS 2142, 1–19, 2001.
10. P.W. O'Hearn and D. Pym. The Logic of Bunched Implications. *Bulletin of Symbolic Logic*, 5(2):215–244, 1999.
11. M. Okada and K. Terui. Completeness proofs for linear logic based on proof search method (preliminary report). In *Type theory and its applications to computer systems*, 57–75, RIMS, Kyoto University, 1998.
12. D. Pym. On bunched predicate logic. In *14th Symposium on Logic in Computer Science*, 183–192, Trento, Italy, July 1999. IEEE Computer Society Press.
13. D.J. Pym. *The Semantics and Proof Theory of the Logic of Bunched Implications*. Applied Logic Series. Kluwer Academic Publishers, 2002. To appear.
14. D.J. Pym, P.W. O'Hearn and H. Yang. Possible Worlds and Resources: The Semantics of **BI**. Manuscript, http://www.dcs.qmul.ac.uk/∼pym/Paperspage/recent.html.