The formal strong completeness of partial monoidal Boolean BI[*]

DOMINIQUE LARCHEY-WENDLING
LORIA – CNRS, Campus Scientifique,
BP 239, Vandœuvre-lès-Nancy, France.
Email: dominique.larchey-wendling@loria.fr

18 November 2013

Abstract

This paper presents a self-contained proof of the strong completeness of the labeled tableaux method for partial monoidal Boolean BI: if a formula has no tableau proof then there exists a counter-model for it which is simple. Simple counter-models are those which are generated from the specific constraints that occur during the tableaux proof-search process. As a companion to this paper, we provide a complete formalisation of this result in Coq[1] and discuss some of its implementation details.

1 Introduction

This paper presents the detailed and self-contained proofs of soundness and (strong) completeness of the labeled tableaux method for a sub-structural logic called partial monoidal Boolean BI. The completeness is obtained w.r.t. the general class of partial commutative monoids but also, and this justifies the use of the "strong" qualification, w.r.t. the strict sub-class of simple partial commutative monoids, which are those that are generated during the tableaux proof-search process. To our knowledge, the completeness proof for this logic has never been published but a soundness proof was already presented in [22] for a labeled tableaux system called TBBI-tableaux. We point out however that the tableaux proof system presented in this paper is not strictly identical to TBBI: these two systems differ but only in their implementations.[2]

The aim of this paper is double: first, to serve as a reference for the completeness result of an important variant of Boolean BI [23] strongly related to Separation Logic [19, 24]; and second, to serve as a guideline for the formal Coq[3] proof that implements the results of this paper. Although not strictly identical to the (informal) developments presented in this paper, the formal proof follows the same plan and there exists a high level of correspondence between intermediate results of both proofs. This correspondence is discussed in a specific section.

The logic Boolean BI is characterised here both with its Kripke semantics and through a proof system based on a labeled tableaux calculus. The addition of labels (and constraints) to traditional proof systems like sequent calculi or tableaux calculi can be a way to introduce specific semantic information within the proof system, something that could otherwise be difficult or impossible with a pure syntactic setting. This framework of labels was introduced for intuitionistic BI in [14] and for

[*] This work was supported by the ANR grant DynRes (project no. ANR-11-BS02-011) and the CNRS.

[1] The Coq code is distributed under a free software license and is accessible at `http://www.loria.fr/~larchey/BBI`.

[2] See Fact 34 and the discussion that follows on page 12.

[3] The Coq system is open source software accessible at `http://coq.inria.fr`.

Boolean BI in [22]. It has been used for other logics like intuitionistic or intermediate logics as well. As an example, Roy Dyckhoff and Sara Negri used a labeled sequent calculus to propose decision methods for Gödel-Dummett logic [11].

The Logic of Bunched Implications [29, 30] called BI is a sub-structural logic usually considered as the foundation of separation logics [19, 26] and spatial logics [6]. It contains both additive operators like $\wedge$, $\vee$ and $\rightarrow$ and multiplicative operators like $*$ and $-\!*$. The multiplicative operators are those of multiplicative intuitionistic Linear Logic [15]. The additives can be interpreted either as in intuitionistic logic which gives rise to intuitionistic BI [30], or as Boolean operators which gives rise to Boolean BI [30, 13]. Classical BI [2, 3] is another variant of BI which combines Boolean additives with classical multiplicatives.

The core semantic link between BI and separation logic can be summed up in the Kripke sharing interpretation of the multiplicative conjunction:

$$m \Vdash A * B \quad \text{iff} \quad \text{there exists } a, b \text{ such that } a \circ b \rhd m \text{ and } a \Vdash A \text{ and } b \Vdash B$$

The ternary composition/decomposition relation $- \circ - \rhd -$ reads either as "m is a result of the composition of a and b" or "m can be decomposed into a and b." The interpretation of composition/decomposition depends on the variant of the logic and/or model, see [22, 24, 5] for an overview of some possible interpretations of this relation. In this paper we will focus on Boolean BI (denoted BBI), more precisely on partial monoidal BBI. In this case, the composition $\circ$ is a partial monoidal operator and the relation $\rhd$ is the identity or at least a congruence relation w.r.t. the partial monoidal composition $\circ$. This is not a restriction since all the models of separation logic and abstract separation logic [7, 5, 21] are in fact partial monoids [24].

Contrary to what happened with intuitionistic BI which was well defined by a cut-free bunched sequent calculus since its inception [30], later completed with a decidability result [14], the proof theory of BBI was, at first, not very well understood. In [30], it is defined as the addition of a double negation principle/axiom to intuitionistic BI, but of course, with this axiom, you loose either cut-elimination or the bunched sequent calculus. In [13], a sound and complete Hilbert style proof system is given for a variant of BBI called relational BBI or non-deterministic BBI. Later, [1] provided a cut-free Display-style sequent calculus for relational BBI. In [22], a sound labeled tableaux calculus is given for partial monoidal BBI, leading to an embedding of intuitionistic BI into Boolean BI, a result which was quite unexpected at that time.[4] But it was still unknown whether relational and partial monoidal BBI coincide or not, or whether BBI was decidable or not.

Then, the situation improved a lot with a model that distinguishes relational and partial monoidal BBI [23] as well as other variants of Boolean BI, leading to a family of different logics [20], and an undecidability result obtained for the whole family of BBI/separation logics, independently and simultaneously in [4] and [23]. We also point out the undecidability result for Classical BI [4, 20]. These undecidability results doomed the different attempts made at providing a decision procedure for BBI either through Display logic [1] or through tableaux calculi [22].

However in this paper, we explain how the labeled tableaux calculus can still be useful as a tool for the study of the properties of BBI, like finer completeness results. We believe that the labeled calculus can also serve as an effective semi-decision algorithm for partial monoidal BBI, but we will only discuss this as a perspective. This work also comes as a complement to [22], the knowledge of which being advised but not required. Let us give a quick overlook of the content of the upcoming sections:

- in Section 2, we describe a framework of labels represented by *words* and *constraints* between those labels that can be used as a syntactic representation for partial monoids. The solutions of those constraints, *partial monoidal equivalences*, give a foundation to the semantics of (partial monoidal) BBI;
- in Section 3, we introduce the syntax and Kripke semantics of (partial monoidal) BBI and the notion of (counter-)model;

[4] and it is the completeness of a nearly isomorphic labeled tableau calculus that we establish here.

- in Section 4, we present the labeled tableaux calculus for BBI with a table of branch expansion rules and the conditions under which the branches of BBI-tableaux are closed. We review some ground properties of the tableau calculus;

- in Section 5, we adapt the soundness proof already given in [22]: any BBI-formula that has a closed BBI-tableau is a universally valid formula. There is nothing really new in this section except the link with the formal Coq proof;

- Section 6 details the main concepts that lie at the heart of the strong completeness result: the notion of *Hintikka set*, the notion of *fair strategy* and the notion of *oracle*. We point out that contrary to the case of first order logic, it is not sufficient to obtain a maximally consistent set of formulæ to be able to extract a counter-model. The Hintikka set has to be constructed more carefully in the case of BBI. We characterise the constraints generated during proof-search as *basic*: they are of the form $ab \dashrightarrow m$, $am \dashrightarrow b$ and $\epsilon \dashrightarrow m$ where m is an already defined label, ϵ is the empty label and a, b are two new labels. *Simple models* arise as those generated by (infinite) sequences of basic constraints. We finish with the strong completeness result: any BBI-formula that has no closed BBI-tableau has a simple counter-model;

- in Section 7 we focus on the strong links and also on the slight differences that exist between the informal proof and the formal proof, mainly consequences of the divergence between the informal meta-logic (classical set theory) and the formal meta-logic (the calculus of inductive constructions);

- in Section 8 we discuss the perspectives of this work: the study of the specific properties of basic and simple BBI models like cancellativity, the effective computation of the solutions of basic sequences of constraints and its use in the context of semi-decision methods for BBI.

2 Partial Monoidal Equivalences

We introduce a framework of labels and constraints to syntactically represent commutative partial monoids which form the semantic basis of partial monoidal Boolean BI. Labels are viewed as words and constraints as binary relations between those words. This choice allows us to easily mix the logical information and the semantic information within our semantic tableaux.

The basic idea is to represent the elements of a partial commutative monoid by multisets of letters and to quotient those multisets by a partial equivalence relation which is moreover congruent with monoidal operations: the neutral constant and the binary composition. We call such relations *partial monoidal equivalences* (PME). To easily compute with PMEs, we provide a set of derivation rules for which they are closed and which precisely characterises them: in a sense, these derivation rules "axiomatize" PMEs. Moreover, the closure operator they generate provides a way to "solve" partial monoidal constraints, i.e. to compute the least partial monoid which satisfies a given set of constraints.

Any commutative partial monoid can be obtained as a quotient of a set of words by a PME and we recall a soundness/completeness result for this labeled semantics of BBI in Section 3.

2.1 Words, constraints and PMEs

Let L be a (potentially infinite) alphabet of letters. We consider the set of words $L^\star$ where the order of letters is not taken into account, i.e. we consider *words as finite multisets of letters.* The composition of words is denoted *multiplicatively* and the *empty word* is denoted ϵ and thus $(L^\star, \cdot, \epsilon)$ is the commutative monoid freely generated by L.

We denote $x \prec y$ when x is a *subword* of y, i.e. when there exists a word k such that $kx = y$. If $x \prec y$, there is only one k such that $xk = y$ and it is denoted y/x, hence $y = x(y/x)$. The *(carrier) alphabet of a word* m is the set of letters of which it is composed: $\mathsf{A}_m = \{l \in L \mid l \prec m\}$. We may view

$$\frac{}{\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon}\,\langle\epsilon\rangle \qquad \frac{x \mathrel{-\!\!\bullet\!\!-} y}{y \mathrel{-\!\!\bullet\!\!-} x}\,\langle s\rangle \qquad \frac{ky \mathrel{-\!\!\bullet\!\!-} ky \qquad x \mathrel{-\!\!\bullet\!\!-} y}{kx \mathrel{-\!\!\bullet\!\!-} ky}\,\langle c\rangle \qquad \frac{xy \mathrel{-\!\!\bullet\!\!-} xy}{x \mathrel{-\!\!\bullet\!\!-} x}\,\langle d\rangle \qquad \frac{x \mathrel{-\!\!\bullet\!\!-} y \qquad y \mathrel{-\!\!\bullet\!\!-} z}{x \mathrel{-\!\!\bullet\!\!-} z}\,\langle t\rangle$$

Table 1: Derivations rules for the definition of PMEs.

the alphabet L or any of its subsets $X \subseteq L$ as a subset $X \subset L^\star$, i.e. *we identify letters and one-letter words* using the canonical embedding.

Definition 1 (Constraint). A *constraint* is a ordered pair of words in $L^\star \times L^\star$ denoted $m \mathrel{-\!\!\bullet\!\!-} n$.

As usual in set theory, we represent a *binary relation* $\mathcal{R} \subseteq L^\star \times L^\star$ between words of $L^\star$ as a set of (ordered) pairs of words, hence as a *set of constraints*. Recall that the notation $x\ \mathcal{R}\ y$ is just a convenient shortcut for $x \mathrel{-\!\!\bullet\!\!-} y \in \mathcal{R}$. We will use both notations throughout the paper sometimes depending on typesetting requirements, sometimes because we want to enforce the distinction between syntax and semantics. For instance, a constraint is syntactic object whereas a relation between words is a semantic object. When $\mathsf{C} = \{\ldots, x_i \mathrel{-\!\!\bullet\!\!-} y_i, \ldots\}$ is viewed as a finite or infinite collection of individual constraints, it is interpreted as a syntactic notion and we write $x \mathrel{-\!\!\bullet\!\!-} y \in \mathsf{C}$ for example. When $\mathcal{R}$ is viewed as a relation between words, it is interpreted as a semantic notion and we rather write $x\ \mathcal{R}\ y$. But the precise nature of C and $\mathcal{R}$ is the same, that of a set of constraints.

Definition 2 (Language of/alphabet of). The *language* of a binary relation $\mathcal{R} \subseteq L^\star \times L^\star$ denoted $\mathsf{L}_\mathcal{R}$ is defined by $\mathsf{L}_\mathcal{R} = \{x \in L^\star \mid \exists m, n \in L^\star \text{ s.t. } xm\ \mathcal{R}\ n \text{ or } m\ \mathcal{R}\ xn\}$. The *carrier alphabet* of $\mathcal{R}$ denoted $\mathsf{A}_\mathcal{R}$ is defined by $\mathsf{A}_\mathcal{R} = \bigcup\{\mathsf{A}_m \cup \mathsf{A}_n \mid m\ \mathcal{R}\ n\}$.

A word $m \in L^\star$ is said to be *defined in* $\mathcal{R}$ if $m \in \mathsf{L}_\mathcal{R}$ and is *undefined in* R otherwise. A letter $l \in L$ is *new to* $\mathcal{R}$ if $l \notin \mathsf{A}_\mathcal{R}$. The language $\mathsf{L}_\mathcal{R}$ is downward closed w.r.t. the subword order $\prec$. The inclusion $\mathsf{L}_\mathcal{R} \subseteq \mathsf{A}_\mathcal{R}^\star$ and the identity $\mathsf{A}_\mathcal{R} = \mathsf{L}_\mathcal{R} \cap L$ hold. If $\mathcal{R}_1$ and $\mathcal{R}_2$ are two relations such that $\mathcal{R}_1 \subseteq \mathcal{R}_2$ then the inclusions $\mathsf{A}_{\mathcal{R}_1} \subseteq \mathsf{A}_{\mathcal{R}_2}$ and $\mathsf{L}_{\mathcal{R}_1} \subseteq \mathsf{L}_{\mathcal{R}_2}$ hold. The alphabet of a set C of constraints is thus $\mathsf{A}_\mathsf{C} = \{l \in L \mid l \prec m \text{ or } l \prec n \text{ for some } m \mathrel{-\!\!\bullet\!\!-} n \in \mathsf{C}\}$, i.e. the set of letters which occur in some constraint of C. Let us now introduce the derivation rules that characterise the particular relations we are interested in.

Definition 3 (PME). A *partial monoidal equivalence* (PME) over the alphabet L is a binary relation $\sim \subseteq L^\star \times L^\star$ which is closed under the derivation rules $\langle \epsilon, s, c, d, t\rangle$ of Table 1.

As an example, let $a, b \in L$ be two different letters and C_0 be the singleton constraint $\mathsf{C}_0 = \{a \mathrel{-\!\!\bullet\!\!-} b\}$. Then C_0 is not a PME because closure by rule $\langle s\rangle$ would require that $b \mathrel{-\!\!\bullet\!\!-} a$ belongs to C_0 as well as $a \mathrel{-\!\!\bullet\!\!-} b$. However $\sim_0 = \{\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon, a \mathrel{-\!\!\bullet\!\!-} a, b \mathrel{-\!\!\bullet\!\!-} b, a \mathrel{-\!\!\bullet\!\!-} b, b \mathrel{-\!\!\bullet\!\!-} a\}$ is a PME (left to the reader) which contains C_0. In fact, it can be checked that it is the least PME which contains C_0. We will call it the PME generated by C_0 (see upcoming Definition 6).

Let us provide some derived rules which will be more suitable for computing relations in PMEs or for proving properties of PMEs throughout this article.

Proposition 4. PMEs are closed under (the derived) rules $\langle p_l, p_r, e_l, e_r\rangle$:

$$\frac{kx \mathrel{-\!\!\bullet\!\!-} y}{x \mathrel{-\!\!\bullet\!\!-} x}\,\langle p_l\rangle \qquad \frac{x \mathrel{-\!\!\bullet\!\!-} ky}{y \mathrel{-\!\!\bullet\!\!-} y}\,\langle p_r\rangle \qquad \frac{x \mathrel{-\!\!\bullet\!\!-} y \qquad yk \mathrel{-\!\!\bullet\!\!-} m}{xk \mathrel{-\!\!\bullet\!\!-} m}\,\langle e_l\rangle \qquad \frac{x \mathrel{-\!\!\bullet\!\!-} y \qquad m \mathrel{-\!\!\bullet\!\!-} yk}{m \mathrel{-\!\!\bullet\!\!-} xk}\,\langle e_r\rangle$$

Proof. For rule $\langle p_l \rangle$ and then rule $\langle p_r \rangle$, we provide the two derivation trees:

$$\dfrac{\dfrac{kx \mathrel{-\!\!\bullet\!\!-} y \qquad \dfrac{kx \mathrel{-\!\!\bullet\!\!-} y}{y \mathrel{-\!\!\bullet\!\!-} kx}\langle s\rangle}{kx \mathrel{-\!\!\bullet\!\!-} kx}\langle t\rangle}{x \mathrel{-\!\!\bullet\!\!-} x}\langle d\rangle \qquad\qquad \dfrac{\dfrac{x \mathrel{-\!\!\bullet\!\!-} ky}{ky \mathrel{-\!\!\bullet\!\!-} x}\langle s\rangle}{y \mathrel{-\!\!\bullet\!\!-} y}\langle p_l\rangle$$

For rule $\langle e_l \rangle$ and then rule $\langle e_r \rangle$, we provide the two derivation trees:

$$\dfrac{\dfrac{\dfrac{yk \mathrel{-\!\!\bullet\!\!-} m}{yk \mathrel{-\!\!\bullet\!\!-} yk}\langle p_l\rangle \quad x \mathrel{-\!\!\bullet\!\!-} y}{xk \mathrel{-\!\!\bullet\!\!-} yk}\langle c\rangle \qquad yk \mathrel{-\!\!\bullet\!\!-} m}{xk \mathrel{-\!\!\bullet\!\!-} m}\langle t\rangle \qquad\qquad \dfrac{\dfrac{x \mathrel{-\!\!\bullet\!\!-} y \quad \dfrac{m \mathrel{-\!\!\bullet\!\!-} yk}{yk \mathrel{-\!\!\bullet\!\!-} m}\langle s\rangle}{xk \mathrel{-\!\!\bullet\!\!-} m}\langle e_l\rangle}{m \mathrel{-\!\!\bullet\!\!-} xk}\langle s\rangle$$

Remark that derived rule $\langle p_l \rangle$ is applied in the left hand side derivation tree with parameter k equal to the empty word ϵ. ■

Rule $\langle p_l \rangle$ (resp. $\langle p_r \rangle$) is a left (resp. right) projection rule. Rules $\langle e_l \rangle$ and $\langle e_r \rangle$ express the capacity to exchange related subwords inside the PME $\sim$, either on the left or on the right.

Proposition 5. Let $\sim$ be a PME over the alphabet L. The following identities hold:

$$\mathsf{L}_\sim = \{x \in L^\star \mid x \sim x\} \qquad \text{and} \qquad \mathsf{A}_\sim = \{l \in L \mid l \sim l\}$$

Proof. It is obvious that $\{x \in L^\star \mid x \sim x\} \subseteq \mathsf{L}_\sim$. For the converse, if $xm \sim n$ (resp. $m \sim xn$) then $x \sim x$ by rule $\langle p_l \rangle$ (resp. rule $\langle p_r \rangle$). Hence, $\mathsf{L}_\sim \subseteq \{x \in L^\star \mid x \sim x\}$. As $\mathsf{A}_\sim = \mathsf{L}_\sim \cap L$, we get $\mathsf{A}_\sim = \{l \in L \mid l \sim l\}$. ■

2.2 Sets of constraints and other properties of PMEs

Given a set of constraints $\mathcal{C}$, we may interpret these syntactic constraints as equations, i.e. relations between words that we want to satisfy. When we compute the closure $\overline{\mathcal{C}}$ of those constraints by the derivation rules of Table 1, we obtain an augmented set of constraints which in turn corresponds to the least solution of those constraints. When we interpret $\overline{\mathcal{C}}$ as a binary relation (semantic object) instead of a set of constraints (syntactic object), we rather write $\sim_\mathcal{C}$.

Since they are defined by closure under some derivation rules, the class of PMEs is closed under arbitrary intersections. Thus, given a set of constraints $\mathcal{C}$, there exists a least PME containing $\mathcal{C}$, which will be denoted by either $\sim_\mathcal{C}$ or $\overline{\mathcal{C}}$. We are especially interested in PMEs generated by some finite or infinite set of constraints, and extensions of existing PMEs with additional constraints.

Definition 6 (Generated PME). Let $\mathcal{C}$ be a set of constraints over the alphabet L. The PME *generated by* $\mathcal{C}$ is the least PME containing $\mathcal{C}$. It is either denoted by $\sim_\mathcal{C}$ or $\overline{\mathcal{C}}$, depending whether we view it as a relation or as a set of constraints. Hence, the denotations $m \sim_\mathcal{C} n$ and $m \mathrel{-\!\!\bullet\!\!-} n \in \overline{\mathcal{C}}$ are synonymous.

As another example, let $a, b, c \in L$ be three different letters and $\mathcal{C}_1$ be the singleton constraint $\mathcal{C}_1 = \{ab \mathrel{-\!\!\bullet\!\!-} ac\}$. Then, the PME generated by $\mathcal{C}_1$ is $\sim_{\mathcal{C}_1} = \overline{\mathcal{C}_1} = \{\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon, a \mathrel{-\!\!\bullet\!\!-} a, b \mathrel{-\!\!\bullet\!\!-} b, c \mathrel{-\!\!\bullet\!\!-} c, ab \mathrel{-\!\!\bullet\!\!-} ab, ac \mathrel{-\!\!\bullet\!\!-} ac, ab \mathrel{-\!\!\bullet\!\!-} ac, ac \mathrel{-\!\!\bullet\!\!-} ab\}$. Interestingly, the constraint $b \mathrel{-\!\!\bullet\!\!-} c$ is not generated by $\mathcal{C}_1$ (i.e. $b \nsim_{\mathcal{C}_1} c$ holds), hence PME rules do not allow to cancel/simplify $ab \sim ac$ by a to obtain $b \sim c$; PMEs are not cancellative in general even though some of them are.

The operator $\mathcal{R} \mapsto \overline{\mathcal{R}}$ is a closure operator on sets of constraints, i.e. it is extensive ($\mathcal{C} \subseteq \overline{\mathcal{C}}$), monotonic ($\mathcal{C} \subseteq \mathcal{D}$ implies $\overline{\mathcal{C}} \subseteq \overline{\mathcal{D}}$) and idempotent ($\overline{\overline{\mathcal{C}}} \subseteq \overline{\mathcal{C}}$).

Proposition 7 (Compactness). Let C be a (possibly infinite) set of constraints over the alphabet L. Let $m, n \in L^\star$ be such that $m \sim_{\mathsf{C}} n$ holds. There exists a finite subset $\mathsf{C}_f \subseteq \mathsf{C}$ such that $m \sim_{\mathsf{C}_f} n$ holds.

Proof. See Proposition 3.17 page 447 of [22]. ■

The *compactness property* is not related to the particular nature of rules defining PMEs but solely to the fact that the derivation rules $\langle \epsilon, s, c, d, t \rangle$ only have a *finite number of premises*. We now show that the carrier alphabet of the closure $\overline{\mathsf{C}}$ of the set of constraints C does not contain letters which do not already occur in the carrier alphabet of C.

Proposition 8. If C is a set of constraints over L then the identity $\mathsf{A}_{\mathsf{C}} = \mathsf{A}_{\overline{\mathsf{C}}}$ holds.

Proof. The alphabet operator $\mathcal{R} \mapsto \mathsf{A}_{\mathcal{R}}$ is obviously monotonic. Thus, as the inclusion $\mathsf{C} \subseteq \overline{\mathsf{C}}$ holds, we derive $\mathsf{A}_{\mathsf{C}} \subseteq \mathsf{A}_{\overline{\mathsf{C}}}$. We now prove the converse inclusion $\mathsf{A}_{\overline{\mathsf{C}}} \subseteq \mathsf{A}_{\mathsf{C}}$. Let $\sim$ be defined by $\sim = \mathsf{A}_{\mathsf{C}}^\star \times \mathsf{A}_{\mathsf{C}}^\star$. Viewed as a binary relation, we thus have $m \sim n$ iff $\{m, n\} \subseteq \mathsf{A}_{\mathsf{C}}^\star$. Then the identity $\mathsf{A}_\sim = \mathsf{A}_{\mathsf{C}}$ and the inclusion $\mathsf{C} \subseteq \sim$ obviously hold. It can moreover easily be checked that $\sim$ is a PME which thus contains the PME generated by C. Hence $\overline{\mathsf{C}} \subseteq \sim$ holds and we conclude on the inclusion $\mathsf{A}_{\overline{\mathsf{C}}} \subseteq \mathsf{A}_\sim = \mathsf{A}_{\mathsf{C}}$. ■

The identity $\mathsf{L}_{\mathsf{C}} = \mathsf{L}_{\overline{\mathsf{C}}}$ *does not hold* in general, but the inclusion $\mathsf{L}_{\mathsf{C}} \subseteq \mathsf{L}_{\overline{\mathsf{C}}}$ holds. However L_{C} is usually strictly included in $\mathsf{L}_{\overline{\mathsf{C}}}$. Consider the third example $\mathsf{C}_2 = \{ab \mathrel{-\!\!\bullet\!\!-} ab, b \mathrel{-\!\!\bullet\!\!-} c\}$ which generates $\overline{\mathsf{C}_2} = \{\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon, a \mathrel{-\!\!\bullet\!\!-} a, b \mathrel{-\!\!\bullet\!\!-} b, c \mathrel{-\!\!\bullet\!\!-} c, b \mathrel{-\!\!\bullet\!\!-} c, c \mathrel{-\!\!\bullet\!\!-} b, ab \mathrel{-\!\!\bullet\!\!-} ab, ac \mathrel{-\!\!\bullet\!\!-} ac, ab \mathrel{-\!\!\bullet\!\!-} ac, ac \mathrel{-\!\!\bullet\!\!-} ab\}$. In this case we have $\mathsf{A}_{\mathsf{C}_2} = \mathsf{A}_{\overline{\mathsf{C}_2}} = \{a, b, c\}$, $\mathsf{L}_{\mathsf{C}_2} = \{\epsilon, a, b, c, ab\}$ and $\mathsf{L}_{\overline{\mathsf{C}_2}} = \{\epsilon, a, b, c, ab, ac\}$ and hence $ac \in \mathsf{L}_{\overline{\mathsf{C}_2}} \backslash \mathsf{L}_{\mathsf{C}_2}$.

Definition 9 (PME extension). Let $\sim$ be a PME and C be a set of constraints, both over L. We denote by $\sim + \mathsf{C}$ the *extension* of $\sim$ by the constraints of C which is $\overline{\sim \cup \mathsf{C}}$, the least PME containing both $\sim$ and C.

Let C and $\mathcal{D}$ be two sets of constraints. Since $\mathcal{R} \mapsto \overline{\mathcal{R}}$ is a closure operator on sets of constraints, we have $\overline{\mathsf{C}} + \mathcal{D} = \overline{\overline{\mathsf{C}} \cup \mathcal{D}} = \overline{\mathsf{C} \cup \mathcal{D}} = \overline{\mathcal{D}} + \mathsf{C}$. If $\sim$ is a PME then the identities $(\sim + \mathsf{C}) + \mathcal{D} = (\sim + \mathcal{D}) + \mathsf{C} = \sim + (\mathsf{C} \cup \mathcal{D})$ hold. Moreover, for any $m, n \in L^\star$, the relation $m \sim n$ holds if and only if the identity $\sim + \{m \mathrel{-\!\!\bullet\!\!-} n\} = \sim$ holds; in particular the identity $\sim + \{\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon\} = \sim$ holds because of rule $\langle \epsilon \rangle$.

2.3 Basic and simple PMEs

In Section 4, we will use PME extensions of the forms $\sim + \{ab \mathrel{-\!\!\bullet\!\!-} m\}$, $\sim + \{am \mathrel{-\!\!\bullet\!\!-} b\}$ or $\sim + \{\epsilon \mathrel{-\!\!\bullet\!\!-} m\}$ where $\sim$ is a PME over L and a and b are two letters new to $\sim$, i.e. they satisfy the condition $a \neq b \in L \backslash \mathsf{A}_\sim$. As fully explained in upcoming Sections 4.2 and 6.4, these are the particular extensions that occur during proof-search using the semantic tableau method for BBI. We qualify these extensions as basic.

Definition 10 (Basic extension). Given a PME $\sim$ over the alphabet L, a constraint is *basic w.r.t.* $\sim$ when it is of one of the three following forms:

1. $ab \mathrel{-\!\!\bullet\!\!-} m$ with $m \sim m$ and $a \neq b \in L \backslash \mathsf{A}_\sim$;
2. $am \mathrel{-\!\!\bullet\!\!-} b$ with $m \sim m$ and $a \neq b \in L \backslash \mathsf{A}_\sim$;
3. $\epsilon \mathrel{-\!\!\bullet\!\!-} m$ with $m \sim m$.

When $x \mathrel{-\!\!\bullet\!\!-} y$ is basic w.r.t. $\sim$, we say that $\sim + \{x \mathrel{-\!\!\bullet\!\!-} y\}$ is a *basic extension of the* PME $\sim$.

Let $(x_i \mathrel{-\!\!\bullet\!\!-} y_i)_{i<k}$ be a sequence of constraints with $k \in \mathbb{N} \cup \{\infty\}$ and C_p be the set of constraints $\mathsf{C}_p = \{x_i \mathrel{-\!\!\bullet\!\!-} y_i \mid i < p\}$ for $p < k$. We suppose that for any $p < k$, the constraint $x_p \mathrel{-\!\!\bullet\!\!-} y_p$ is basic with respect to $\sim_{\mathsf{C}_p}$. If $k < \infty$ then *the sequence* $(x_i \mathrel{-\!\!\bullet\!\!-} y_i)_{i<k}$ *is called basic.* This definition implies in particular that the empty sequence of constraints is basic. If $k = \infty$ then *the sequence* $(x_i \mathrel{-\!\!\bullet\!\!-} y_i)_{i<\infty}$ *is called simple.*

Definition 11 (Basic/Simple PME). A basic (resp. simple)[5] PME is of the form $\sim_{\mathcal{C}}$ where $\mathcal{C} = \{x_i \mathrel{-\!\!\bullet\!\!-} y_i \mid i < k\}$ and $(x_i \mathrel{-\!\!\bullet\!\!-} y_i)_{i<k}$ is a basic (resp. simple) sequence of constraints.

We make the obvious following remark: according to those definitions, if $\sim$ is a basic PME and $\sim + \{x \mathrel{-\!\!\bullet\!\!-} y\}$ is a basic extension of $\sim$ then $\sim + \{x \mathrel{-\!\!\bullet\!\!-} y\}$ is a basic PME.

Using case 3 (with $m = \epsilon$) of Definition 10, any (finite) basic sequence can be completed into an (infinite) simple sequence by repeated use of the constraint $\epsilon \mathrel{-\!\!\bullet\!\!-} \epsilon$ (which always constitutes a basic extension). Since adding this constraint does not change the corresponding PME (because of rule $\langle\epsilon\rangle$), basic PMEs are also simple PMEs. Of course, the converse is not true. Indeed, the alphabet of a basic PME is always finite whereas the alphabet of a simple PME can be infinite. So the difference between basic and simple PMEs is that in the later case, the underlying sequence can be infinite whereas it must be finite for basic PMEs.

2.4 PMEs and substitutions

In this section, we define the substitutions of letters by words into constraints and show that the PME closure operator $\mathcal{R} \mapsto \overline{\mathcal{R}}$ commutes with substitutions. A substitution σ maps letters of L to words in $K^\star$, where L and K are not necessarily the same alphabet. There is a *unique morphism of commutative monoids* $L^\star \longrightarrow K^\star$ which extends σ and we usually denote this morphism by σ as well, despite the little confusion this notation ambiguity may generate. We point out the particular case of substitution of letters where $L = K$ and each letter i is substituted into a single letter word $\sigma(i) \in L \subset L^\star$.

Definition 12 (Substitution). Let L and K be two alphabets. A *substitution* σ is a total map from letters to words $\sigma : L \longrightarrow K^\star$. We naturally extend a substitution σ in a total map from words to words $\sigma(\cdot) : L^\star \longrightarrow K^\star$ by $\sigma(m) = \sigma(m_1 m_2 \ldots m_k) = \sigma(m_1)\sigma(m_2)\ldots\sigma(m_k)$ and obtain a morphism of (commutative) monoids. A *substitution of letters* $\sigma : L \longrightarrow L$ is a particular case of substitution where each image $\sigma(a)$ of a letter $a \in L$ is a single letter word in $L^\star$. Given a set of constraints $\mathcal{C}$ over L, we also define the *substituted set of constraints* $\sigma(\mathcal{C})$ over K by $\sigma(\mathcal{C}) = \{\sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \mid m \mathrel{-\!\!\bullet\!\!-} n \in \mathcal{C}\}$.

Theorem 13. If $\sigma : L \longrightarrow K^\star$ is a substitution and $\mathcal{C}$ is a set of constraints then $\sigma(\overline{\mathcal{C}}) \subseteq \overline{\sigma(\mathcal{C})}$ holds.

Proof. We define the binary relation $\sim$ by $m \sim n$ iff $\sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \overline{\sigma(\mathcal{C})}$. Because the extension of σ is a morphism of commutative monoids and $\overline{\sigma(\mathcal{C})}$ is a PME, it is easy to check that $\sim$ is also a PME. From $\sigma(\mathcal{C}) \subseteq \overline{\sigma(\mathcal{C})}$ we deduce $\mathcal{C} \subseteq \sim$; indeed, $m \mathrel{-\!\!\bullet\!\!-} n \in \mathcal{C} \Rightarrow \sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \sigma(\mathcal{C}) \Rightarrow \sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \overline{\sigma(\mathcal{C})} \Rightarrow m \sim n$. Hence we derive $\overline{\mathcal{C}} \subseteq \sim$. Now let $x \mathrel{-\!\!\bullet\!\!-} y \in \sigma(\overline{\mathcal{C}})$ and let us prove $x \mathrel{-\!\!\bullet\!\!-} y \in \overline{\sigma(\mathcal{C})}$. There exists a pair $m \mathrel{-\!\!\bullet\!\!-} n \in \overline{\mathcal{C}}$ such that $x = \sigma(m)$ and $y = \sigma(n)$. Since $m \mathrel{-\!\!\bullet\!\!-} n \in \overline{\mathcal{C}}$, we deduce $m \sim n$ and thus $\sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \overline{\sigma(\mathcal{C})}$. Hence $x \mathrel{-\!\!\bullet\!\!-} y \in \overline{\sigma(\mathcal{C})}$. The inclusion $\sigma(\overline{\mathcal{C}}) \subseteq \overline{\sigma(\mathcal{C})}$ holds. ■

We could have proved this result by applying substitutions to whole derivation trees built using the rules of Table 1; those rules are obviously stable under substitutions. However, the argument we propose involves a specific PME. In the end, both proofs come down to the same argument: the derivation rules of Table 1 are stable under substitutions.

Corollary 14. Let $\sigma : L \longrightarrow K^\star$ be a substitution and $\mathcal{C}$ be a set of constraints. For any $m, n \in L^\star$ if $m \sim_{\mathcal{C}} n$ then $\sigma(m) \sim_{\sigma(\mathcal{C})} \sigma(n)$.

Proof. Given $m, n \in L^\star$, if $m \sim_{\mathcal{C}} n$ then $m \mathrel{-\!\!\bullet\!\!-} n \in \overline{\mathcal{C}}$ then $\sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \sigma(\overline{\mathcal{C}})$ (by definition of $\sigma(\overline{\mathcal{C}})$) then $\sigma(m) \mathrel{-\!\!\bullet\!\!-} \sigma(n) \in \overline{\sigma(\mathcal{C})}$ (by Theorem 13) which by definition is exactly $\sigma(m) \sim_{\sigma(\mathcal{C})} \sigma(n)$. ■

[5] We make an important remark for those readers who did consult our earlier paper [22]. In that paper, we defined BBI-elementary and BBI-simple PMEs. We point out that basic PMEs and BBI-elementary PMEs are not the same notions and that the current Definition 11 of simple PME is different from the one of BBI-simple PME in [22] (Definitions 6.1 and 6.2 page 464). This does not affect the results in any way, it just reveals an unfortunate choice of terminology.

3 Boolean BI and its labeled Kripke semantics

The language of the logic BBI is syntactically defined by the following grammar where Var is a countable set of propositional variables:

$$\mathsf{Form} \;:\; A, B ::= X \mid A \wedge B \mid \neg A \mid \mathsf{I} \mid A * B \mid A \mathrel{-\!\!*} B \quad \text{with } X \in \mathsf{Var}$$

Thus the denotation Form will be used to represent the set of formulæ of BBI. We point out that we only consider the Boolean connectives of negation $\neg$ and of conjonction $\wedge$ since the other Boolean connectives, either the constants $\bot$, $\top$, the disjunction $\vee$ or the implication $\rightarrow$ can be obtained by a combination of $\neg$ and $\wedge$. This nearly minimalist choice limits duplicated cases in many proofs of this paper as well as in the formalised Coq proof discussed in Section 7. We introduce a Kripke interpretation of BBI formulæ based on PMEs.

Definition 15. A BBI-*frame* is a triple $(L, \sim, \Vdash)$ where L is an alphabet, $\sim$ is a PME over L, and $\Vdash$ is a *forcing relation* $\Vdash \subseteq \mathsf{L}_\sim \times \mathsf{Var}$ which verifies the *monotonicity property*:[6]

$$\forall X \in \mathsf{Var},\ \forall m, n \in \mathsf{L}_\sim,\ \text{if } m \sim n \text{ and } m \Vdash X \text{ then } n \Vdash X$$

We extend the forcing relation to $\Vdash_\sim \subseteq \mathsf{L}_\sim \times \mathsf{Form}$ by induction on formulæ:

$$\begin{array}{lll}
m \Vdash_\sim \neg A & \text{iff} & \text{not } m \Vdash_\sim A \\
m \Vdash_\sim A \wedge B & \text{iff} & m \Vdash_\sim A \text{ and } m \Vdash_\sim B \\
m \Vdash_\sim \mathsf{I} & \text{iff} & \epsilon \sim m \\
m \Vdash_\sim A * B & \text{iff} & \text{there exists } x, y \text{ s.t. } xy \sim m \text{ and } x \Vdash_\sim A \text{ and } y \Vdash_\sim B \\
m \Vdash_\sim A \mathrel{-\!\!*} B & \text{iff} & \text{for any } x, y \text{ if } xm \sim y \text{ and } x \Vdash_\sim A \text{ then } y \Vdash_\sim B
\end{array}$$

We may write $\Vdash$ for $\Vdash_\sim$ when the relation $\sim$ is obvious from the context.

Proposition 16. The extended relation $\Vdash_\sim$ is monotonic.

Proof. Monotonicity holds when for any $F \in \mathsf{Form}$ and any $m, n \in \mathsf{L}_\sim$, the condition $(m \sim n \wedge m \Vdash_\sim F) \Rightarrow n \Vdash_\sim F$ holds. It is standard to prove monotonicity by induction on the formula F. When F is a logical variable, the monotonicity condition holds as a direct consequence of Definition 15. For the additive operator $\wedge$, the induction step is trivial. For the Boolean negation $\neg$, the induction step involves the use of rule $\langle s \rangle$. For the multiplicative operators I and $*$, the induction step involves the use of rule $\langle t \rangle$. For operator $\mathrel{-\!\!*}$, the induction step involves the use of the (derived) rule $\langle e_l \rangle$. ■

Definition 17 (Validity). A formula $F \in \mathsf{Form}$ is *valid* in the BBI-frame $(L, \sim, \Vdash)$ if for every $m \in \mathsf{L}_\sim$ the relation $m \Vdash_\sim F$ holds.

Fact 18 (Soundness/completeness of PMEs). A BBI-formula F is valid in every partial monoidal Kripke structure if and only if it is valid in every BBI-frame.

Proof. A proof of this result is given in [22], see Theorems 3.12 and 3.13 pages 445–446. See also [21] to situate this result in the more general framework of non-deterministic Kripke semantics. The main idea is the following: given a PME $\sim$ over $L^\star$, the restriction of $\sim$ to $\mathsf{L}_\sim$ is an equivalence relation and the quotient $\mathsf{L}_\sim/\!\sim$ has the structure of a partial commutative monoid. Up to isomorphism, any partial commutative monoid can be obtained as such a quotient (for a well chosen alphabet L). ■

According to this theorem, we can define *universal validity* and *counter-models* for BBI. A BBI counter-model for $F \in \mathsf{Form}$ is a tuple $(L, \sim, \Vdash, m)$ where $(L, \sim, \Vdash)$ is a BBI-frame, $m \in \mathsf{L}_\sim$ and $m \nVdash_\sim F$. F is *universally valid* when it has no BBI counter-model.

[6] Since the relation $\sim$ is symmetric by rule $\langle s \rangle$, the converse implication $(m \sim n \wedge n \Vdash X) \Rightarrow m \Vdash X$ also holds.

$$\frac{\mathbb{T}\neg A : m \in \mathcal{X}}{(\{\mathbb{F}A : m\}, \emptyset)}\,\langle\mathbb{T}\neg\rangle \qquad \frac{\mathbb{F}\neg A : m \in \mathcal{X}}{(\{\mathbb{T}A : m\}, \emptyset)}\,\langle\mathbb{F}\neg\rangle$$

$$\frac{\mathbb{T}A \wedge B : m \in \mathcal{X}}{(\{\mathbb{T}A : m, \mathbb{T}B : m\}, \emptyset)}\,\langle\mathbb{T}\wedge\rangle \qquad \frac{\mathbb{F}A \wedge B : m \in \mathcal{X}}{(\{\mathbb{F}A : m\}, \emptyset) \ \mid \ (\{\mathbb{F}B : m\}, \emptyset)}\,\langle\mathbb{F}\wedge\rangle$$

$$\frac{\mathbb{T}\mathsf{I} : m \in \mathcal{X}}{(\emptyset, \{\epsilon \mathrel{-\!\!\bullet\!\!-} m\})}\,\langle\mathbb{T}\mathsf{I}\rangle$$

$$\frac{\mathbb{T}A * B : m \in \mathcal{X} \text{ and } a \neq b \in L\backslash\mathsf{A}_{\mathsf{C}}}{(\{\mathbb{T}A : a, \mathbb{T}B : b\}, \{ab \mathrel{-\!\!\bullet\!\!-} m\})}\,\langle\mathbb{T}*\rangle \qquad \frac{\mathbb{F}A * B : m \in \mathcal{X} \text{ and } xy \sim_{\mathsf{C}} m}{(\{\mathbb{F}A : x\}, \emptyset) \ \mid \ (\{\mathbb{F}B : y\}, \emptyset)}\,\langle\mathbb{F}*\rangle$$

$$\frac{\mathbb{T}A \mathrel{-\!\!*} B : m \in \mathcal{X} \text{ and } xm \sim_{\mathsf{C}} y}{(\{\mathbb{F}A : x\}, \emptyset) \ \mid \ (\{\mathbb{T}B : y\}, \emptyset)}\,\langle\mathbb{T}\mathrel{-\!\!*}\rangle \qquad \frac{\mathbb{F}A \mathrel{-\!\!*} B : m \in \mathcal{X} \text{ and } a \neq b \in L\backslash\mathsf{A}_{\mathsf{C}}}{(\{\mathbb{T}A : a, \mathbb{F}B : b\}, \{am \mathrel{-\!\!\bullet\!\!-} b\})}\,\langle\mathbb{F}\mathrel{-\!\!*}\rangle$$

Table 2: The list of branch expansion rules for the BBI-tableau system.

4 Labeled Tableaux for Boolean BI

We present a labeled tableau proof system for BBI and review some ground properties of tableaux. In Section 5, we will give a proof of its soundness (i.e. any formula which has a closed tableau is universally valid) and in Section 6, a proof of its completeness (i.e. any formula that has no closed tableau has a counter-model).

For all the discussions in the present and future sections, we fix an infinite and countable alphabet $L = \{c_0, c_1, c_2, \ldots\}$ which is ordered by the injective sequence $(c_i)_{i\in\mathbb{N}}$. Hence L is bijectively enumerated by this sequence. Labels will be words in the set $L^\star$.

4.1 Semantic tableaux with constraints for BBI

The tableaux we consider contain both statements and constraints. A statement is a term of the form $\mathbb{S}A : m$ where $\mathbb{S}$ is the sign, A is a BBI-formula and m is a word in $L^\star$. In the following, we ensure that if a statement $\mathbb{S}A : m$ occur in a branch of a tableau, then the branch contains constraints C such that m in defined in $\overline{\mathsf{C}}$, i.e. $m \sim_{\mathsf{C}} m$. For a greater simplicity of notations, we collect the statements and constraints in two different bags.

Definition 19. A *tableau statement* is a triple $(\mathbb{S}, A, m) \in \{\mathbb{T}, \mathbb{F}\} \times \mathsf{Form} \times L^\star$ written $\mathbb{S}A : m$. A *constrained set of statements* (CSS in short) is a pair $(\mathcal{X}, \mathsf{C})$ where $\mathcal{X}$ is a set of tableau statements and C is a set of constraints such that for every statement $\mathbb{S}A : m \in \mathcal{X}$, the relation $m \sim_{\mathsf{C}} m$ holds. A CSS $(\mathcal{X}, \mathsf{C})$ is *finite* if both $\mathcal{X}$ and C are finite sets. The binary relation of inclusion $\preccurlyeq$ between CSS defined by

$$(\mathcal{X}, \mathsf{C}) \preccurlyeq (\mathcal{X}', \mathsf{C}') \quad \text{iff} \quad \mathcal{X} \subseteq \mathcal{X}' \text{ and } \mathsf{C} \subseteq \mathsf{C}'$$

is an order relation. We denote $(\mathcal{X}_f, \mathsf{C}_f) \preccurlyeq_f (\mathcal{X}, \mathsf{C})$ when $(\mathcal{X}_f, \mathsf{C}_f) \preccurlyeq (\mathcal{X}, \mathsf{C})$ holds and $(\mathcal{X}_f, \mathsf{C}_f)$ is finite.

Proposition 20. For any CSS $(\mathcal{X}_f, \mathsf{C})$ where $\mathcal{X}_f$ is finite, there exists $\mathsf{C}_f \subseteq \mathsf{C}$ such that C_f is finite and $(\mathcal{X}_f, \mathsf{C}_f)$ is a CSS (and hence the relation $(\mathcal{X}_f, \mathsf{C}_f) \preccurlyeq_f (\mathcal{X}_f, \mathsf{C})$ holds).

Proof. By induction on the number of statements in $\mathcal{X}_f$ using the compactness property for PMEs (Proposition 7). ■

Let us start with an informal discussion of the tableau system which is centered around the notion of *branch expansion.* The rules of Table 2 describe the atomic steps of the branch expansion process by which BBI-tableaux are built. They have the following form:

$$\frac{\text{cond}(\mathcal{X},\mathcal{C})}{(\mathcal{X}_1,\mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k,\mathcal{C}_k)}$$

The CSS $(\mathcal{X},\mathcal{C})$ represents a tableau branch and the list $(\mathcal{X}_1,\mathcal{C}_1),\ldots,(\mathcal{X}_k,\mathcal{C}_k)$ of length k encode how the branch $(\mathcal{X},\mathcal{C})$ should be expanded provided the rule can be applied; in the specific rules of Table 2, the value of k is either 1 or 2. The premise $\text{cond}(\mathcal{X},\mathcal{C})$ is a condition expressing when the rule can be applied (i.e. is fireable, see below). Each rule generates instances depending on different parameters such as $A,B \in \mathsf{Form}$, $a,b \in L$ and $x,y,m \in L^\star$. Notice that $\mathcal{X}$ and $\mathcal{C}$ are not parameters: they are binders for the predicate $\text{cond}(\mathcal{X},\mathcal{C})$ and do not occur in $\mathcal{X}_1,\mathcal{C}_1,\ldots,\mathcal{X}_k,\mathcal{C}_k$. As an example, the condition of the instance of rule $\langle\mathbb{T}\neg\rangle$ with parameter A set to $A := U * V$ and parameter m set to $m := k$ is the predicate $\varphi : \mathsf{CSS} \longrightarrow \{0,1\}$ defined by $\varphi(\mathcal{X},\mathcal{C}) := (\mathbb{T}\neg(U*V) : k) \in \mathcal{X}$. And the corresponding instance of rule $\langle\mathbb{T}\neg\rangle$ is thus $\varphi/(\{\mathbb{F}(U*V) : k, \emptyset)$ with a single expansion.

When an instance of a rule is such that the condition $\text{cond}(\mathcal{X},\mathcal{C})$ is fulfilled for a particular CSS $(\mathcal{X},\mathcal{C})$,[7] we say that *this instance is fireable for* $(\mathcal{X},\mathcal{C})$. A rule instance fireable for the branch $(\mathcal{X},\mathcal{C})$ can be applied to it and in this case, the branch $(\mathcal{X},\mathcal{C})$ is replaced by k new branches $(\mathcal{X}\cup\mathcal{X}_1,\mathcal{C}\cup\mathcal{C}_1),\ldots,(\mathcal{X}\cup\mathcal{X}_k,\mathcal{C}\cup\mathcal{C}_k)$ in the tableau. We say that *the rule instance is fired* and that the new branches $(\mathcal{X}\cup\mathcal{X}_i,\mathcal{C}\cup\mathcal{C}_i)$ are *the expansions* of $(\mathcal{X},\mathcal{C})$. We see that the branch on which a rule instance is fired is indeed expanded, i.e. the information contained in the branch grows according to the inclusion order $\preccurlyeq$, but this expansion is not necessarily strict on the obtained branches: some branches might remain unchanged after an expansion which is the case when $\mathcal{X} = \mathcal{X}\cup\mathcal{X}_i$ and $\mathcal{C} = \mathcal{C}\cup\mathcal{C}_i$.[8] We now give a formal definition of our BBI-tableaux implemented by lists of branches.

In the following definition, we use the Coq notations for lists: the infix notation $+\!\!+$ is for (associative) list concatenation and $[\mathcal{B}_1;\ldots;\mathcal{B}_n]$ for lists (of branches) with the branches $\mathcal{B}_1,\ldots,\mathcal{B}_n$ enumerated.

Definition 21 (BBI-tableau). Let $(\mathcal{X}_0,\mathcal{C}_0)$ be a CSS. A BBI-*tableau for* $(\mathcal{X}_0,\mathcal{C}_0)$ is a (finite) list of CSS which are called the *branches* of the tableau, built inductively according to the following rules:

1. the one branch list $[(\mathcal{X}_0,\mathcal{C}_0)]$ is a BBI-tableau for $(\mathcal{X}_0,\mathcal{C}_0)$;
2. if the list $\mathcal{T}_l +\!\!+ [(\mathcal{X},\mathcal{C})] +\!\!+ \mathcal{T}_r$ is a BBI-tableau for $(\mathcal{X}_0,\mathcal{C}_0)$ and

$$\frac{\text{cond}(\mathcal{X},\mathcal{C})}{(\mathcal{X}_1,\mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k,\mathcal{C}_k)}$$

is a instance of some rule of Table 2 fireable for $(\mathcal{X},\mathcal{C})$, then the list

$$\mathcal{T}_l +\!\!+ [(\mathcal{X}\cup\mathcal{X}_1,\mathcal{C}\cup\mathcal{C}_1);\ldots;(\mathcal{X}\cup\mathcal{X}_k,\mathcal{C}\cup\mathcal{C}_k)] +\!\!+ \mathcal{T}_r$$

is a BBI-tableau for $(\mathcal{X}_0,\mathcal{C}_0)$.

When we just say BBI-*tableau*, we mean a list of CSS which is a BBI-tableau for some CSS $(\mathcal{X}_0,\mathcal{C}_0)$.

A quick look at the rules of Table 2 should convince the reader of the obvious fact that the branch expansion process preserves lists of finite CSS. With a finer study of those rules, we will show that expansion also preserves basic PMEs (see Proposition 33).

Proposition 22. Any expansion of a CSS (resp. finite CSS) is a CSS (resp. finite CSS).

[7] On the previous example, it would mean that the predicate φ evaluates to "true" on the CSS $(\mathcal{X},\mathcal{C})$.

[8] This typically happens when you apply the same rule instance twice in a row.

Proof. Direct application of Proposition 4: using rules $\langle p_l \rangle$ and $\langle p_r \rangle$, we observe that the PME extensions $\sim + \{ab \leftarrowtail m\}$ and $\sim + \{am \leftarrowtail b\}$ contain the two constraints $a \leftarrowtail a$ and $b \leftarrowtail b$. ■

Proposition 23 (Monotonicity). If $[(\mathcal{X}_1, \mathcal{C}_1); \ldots ; (\mathcal{X}_k, \mathcal{C}_k)]$ is a BBI-tableau for $(\mathcal{X}_0, \mathcal{C}_0)$ then the inclusion $(\mathcal{X}_0, \mathcal{C}_0) \preccurlyeq (\mathcal{X}_i, \mathcal{C}_i)$ holds for any $i \in [1, k]$.

Proof. By induction on the process (i.e. the sequence of fired instances) that builds the tableau for $(\mathcal{X}_0, \mathcal{C}_0)$. Left to the reader. ■

Proposition 24 (Composition). If $\mathcal{T}_l +\!\!+ [(\mathcal{X}, \mathcal{C})] +\!\!+ \mathcal{T}_r$ is a BBI-tableau for $(\mathcal{X}_0, \mathcal{C}_0)$ and $\mathcal{T}$ is a BBI-tableau for $(\mathcal{X}, \mathcal{C})$ then $\mathcal{T}_l +\!\!+ \mathcal{T} +\!\!+ \mathcal{T}_r$ is a BBI-tableau for $(\mathcal{X}_0, \mathcal{C}_0)$.

Proof. By induction, simply replay the process that built $\mathcal{T}$ inside the list context $\mathcal{T}_l +\!\!+ (\cdot) +\!\!+ \mathcal{T}_r$. Left to the reader. ■

Definition 25 (Closure conditions). A CSS $(\mathcal{X}, \mathcal{C})$ *is closed* if at least one of the two following conditions is fulfilled:

1. $\mathbb{T}A : m \in \mathcal{X}$, $\mathbb{F}A : n \in \mathcal{X}$ and $m \sim_{\mathcal{C}} n$ hold for some $A \in \mathsf{Form}$ and some $m, n \in L^\star$;

2. $\mathbb{F}\mathsf{I} : m \in \mathcal{X}$ and $\epsilon \sim_{\mathcal{C}} m$ hold for some $m \in L^\star$.

A CSS *is open* if it is not closed. A BBI-*tableau is closed* if all its branches are closed.

Proposition 26. The closures conditions of Definition 25 are monotonic, i.e. for any two CSS $(\mathcal{X}, \mathcal{C})$ and $(\mathcal{X}', \mathcal{C}')$ such that $\mathcal{X} \subseteq \mathcal{X}'$ and $\mathcal{C} \subseteq \overline{\mathcal{C}'}$ both hold, if $(\mathcal{X}, \mathcal{C})$ is closed then $(\mathcal{X}', \mathcal{C}')$ is closed.

Proof. The inclusion $\mathcal{C} \subseteq \overline{\mathcal{C}'}$ is equivalent to $\overline{\mathcal{C}} \subseteq \overline{\mathcal{C}'}$, i.e. $\sim_{\mathcal{C}} \subseteq \sim_{\mathcal{C}'}$. ■

Proposition 27. Every (infinite) closed CSS contains a finite and closed sub-CSS.

Proof. We consider the two cases for closure of a CSS and we extract a finite and closed sub-CSS using compactness Proposition 7:

1. if $\mathbb{T}A : m \in \mathcal{X}$, $\mathbb{F}A : n \in \mathcal{X}$ and $m \sim_{\mathcal{C}} n$ hold for some $A \in \mathsf{Form}$, $m, n \in L^\star$, then by compactness, let $\mathcal{C}_f$ be a finite subset of $\mathcal{C}$ such that $m \sim_{\mathcal{C}_f} n$. Let $\mathcal{X}_f = \{\mathbb{T}A : m, \mathbb{F}A : n\}$. Then we have $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$ and $(\mathcal{X}_f, \mathcal{C}_f)$ is a closed CSS;
2. if $\mathbb{F}\mathsf{I} : m$ and $\epsilon \sim_{\mathcal{C}} m$ hold for some $m \in L^\star$, then by compactness, let $\mathcal{C}_f$ be a finite subset of $\mathcal{C}$ such that $\epsilon \sim_{\mathcal{C}_f} m$. Let $\mathcal{X}_f = \{\mathbb{F}\mathsf{I} : m\}$. Then we have $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$ and $(\mathcal{X}_f, \mathcal{C}_f)$ is a closed CSS.

■

Definition 28 (CSS substitutions). Let $\sigma : L \longrightarrow K^\star$ be a substitution. If $\mathbb{S}A : m$ is a tableau statement then we define $\sigma(\mathbb{S}A : m) = \mathbb{S}A : \sigma(m)$. The substitution σ extends to $(\mathcal{X}, \mathcal{C})$ by

$$\big(\sigma(\mathcal{X}), \sigma(\mathcal{C})\big) = \big(\{\mathbb{S}A : \sigma(m) \mid \mathbb{S}A : m \in \mathcal{X}\}, \{\sigma(m) \leftarrowtail \sigma(n) \mid m \leftarrowtail n \in \mathcal{C}\}\big)$$

If $\mathcal{T}$ is a list of CSS (typically a tableau) we write $\sigma(\mathcal{T})$ for $\sigma(\mathcal{T}) = \text{map } \sigma\ \mathcal{T}$, i.e.

$$\sigma\big([(\mathcal{X}_1, \mathcal{C}_1); \ldots ; (\mathcal{X}_k, \mathcal{C}_k)]\big) = [(\sigma(\mathcal{X}_1), \sigma(\mathcal{C}_1)); \ldots ; (\sigma(\mathcal{X}_k), \sigma(\mathcal{C}_k))]$$

In this previous definition, we used "map σ" which is the Coq/OCaml denotation for the operator that maps the substitution σ on every element of the list $\mathcal{T}$.

Proposition 29. Let $\sigma : L \longrightarrow K^\star$ be a substitution. The following properties hold:

1. if $(\mathcal{X}, \mathcal{C})$ is a CSS then $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$ is a CSS;

2. if $(\mathcal{X}_f, \mathcal{C}_f)$ is a finite CSS then $(\sigma(\mathcal{X}_f), \sigma(\mathcal{C}_f))$ is a finite CSS;

3. if $(\mathcal{X}, \mathcal{C}) \preccurlyeq (\mathcal{X}', \mathcal{C}')$ then $(\sigma(\mathcal{X}), \sigma(\mathcal{C})) \preccurlyeq (\sigma(\mathcal{X}'), \sigma(\mathcal{C}'))$;

4. if $(\mathcal{X}, \mathcal{C})$ is a closed CSS then $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$ is a closed CSS.

Proof. Immediate consequences of Corollary 14. ■

4.2 Properties of BBI-tableau expansion

BBI-tableaux are stable under a substitution $\sigma : L \longrightarrow L$ of letters provided σ satisfy two properties that are bit weaker than its injectivity.[9]

Theorem 30. Let $(\mathcal{X}, \mathcal{C})$ be a CSS and $\sigma : L \longrightarrow L$ be a substitution of letters such that σ is injective on $L \backslash \mathsf{A}_{\mathcal{C}}$ and the inclusion $\sigma^{-1}(\sigma(\mathsf{A}_{\mathcal{C}})) \subseteq \mathsf{A}_{\mathcal{C}}$ holds. If $\mathcal{T}$ is a BBI-tableau for $(\mathcal{X}, \mathcal{C})$ then $\sigma(\mathcal{T})$ is a BBI-tableau for $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$.

Proof. By induction of the process that builds the tableau for $(\mathcal{X}, \mathcal{C})$, see Appendix A. ∎

We now introduce an important result on the monotonicity of closed BBI-tableaux and give a brief overview of its proof; the detailed proof is complicated by the fact that some of the new labels need to be renamed to avoid conflicts.

Theorem 31. Let $(\mathcal{X}_1, \mathcal{C}_1)$ and $(\mathcal{X}_2, \mathcal{C}_2)$ be two finite CSS such that $\mathcal{X}_1 \subseteq \mathcal{X}_2$ and $\mathcal{C}_1 \subseteq \overline{\mathcal{C}_2}$ both hold. If $(\mathcal{X}_1, \mathcal{C}_1)$ has a closed BBI-tableau then $(\mathcal{X}_2, \mathcal{C}_2)$ has a closed BBI-tableau.

Proof. See Appendix A. The basic idea is to replay the rules that built the tableau for $(\mathcal{X}_1, \mathcal{C}_1)$ on $(\mathcal{X}_2, \mathcal{C}_2)$. But since some letters in $\mathcal{C}_2$ might be new to $\mathcal{C}_1$, we have to be careful not using them in the instances of rules that require new letters, i.e. rules $\langle \mathbb{T}* \rangle$ and $\langle \mathbb{F}{-\!\!*} \rangle$. For this proof, it is critically important that L contains an infinite set of letters; the result itself would be invalid in case L was finite, because tableau expansion could then be blocked by exhaustion of free letters. ∎

Corollary 32. Let $a, b \in L$ and $G \in \mathsf{Form}$ be a BBI-formula. The following properties are equivalent:

1. the finite CSS $(\{\mathbb{F}G : a\}, \{a \multimapdot a\})$ has a closed BBI-tableau;

2. the finite CSS $(\{\mathbb{F}G : a\}, \{\epsilon \multimapdot \epsilon, a \multimapdot b\})$ has a closed BBI-tableau;

3. the finite CSS $(\{\mathbb{F}G : a\}, \{a \multimapdot b\})$ has a closed BBI-tableau.

Proof. Let us prove $(1 \Rightarrow 2)$. Let us suppose that $(\{\mathbb{F}G : a\}, \{a \multimapdot a\})$ has a closed tableau. Using rule $\langle p_l \rangle$, we easily check that the inclusion $\{a \multimapdot a\} \subseteq \overline{\{\epsilon \multimapdot \epsilon, a \multimapdot b\}}$ holds. By Theorem 31, $(\{\mathbb{F}G : a\}, \{\epsilon \multimapdot \epsilon, a \multimapdot b\})$ has a closed tableau.

To prove $(2 \Rightarrow 3)$, we use Theorem 31 remarking that $\{\epsilon \multimapdot \epsilon, a \multimapdot b\} \subseteq \overline{\{a \multimapdot b\}}$ because of rule $\langle \epsilon \rangle$.

Let us first prove $(3 \Rightarrow 1)$. So let us suppose that $\mathcal{T}$ is a closed tableau for $(\{\mathbb{F}G : a\}, \{a \multimapdot b\})$. Then let us consider $\sigma = \mathrm{id}[b/a]$ that maps b to a and preserves every other letter in L. Using Theorem 30 (with $\mathcal{C} = \{a \multimapdot b\}$), we deduce that $\sigma(\mathcal{T})$ is a tableau for $(\{\mathbb{F}G : a\}, \{a \multimapdot a\})$ of which every branch is closed by Proposition 29. ∎

Proposition 33. Let $(\mathcal{X}_0, \mathcal{C}_0)$ be a finite CSS such that $\sim_{\mathcal{C}_0}$ is a basic PME. Let $\mathcal{T}$ be a BBI-tableau for $(\mathcal{X}_0, \mathcal{C}_0)$. For any branch $(\mathcal{X}, \mathcal{C})$ of $\mathcal{T}$, $\sim_{\mathcal{C}}$ is a basic PME.

Proof. It is sufficient to show that the branch expansion process preserves basic PMEs. The constraints part of CSS is strictly expanded only in the case of rules $\langle \mathbb{T}\mathbb{I} \rangle$, $\langle \mathbb{T}* \rangle$ and $\langle \mathbb{F}{-\!\!*} \rangle$. The constraints which are added are $\epsilon \multimapdot m$, $ab \multimapdot m$ and $am \multimapdot b$ respectively which exactly (and purposely) correspond to the three cases of Definition 10. ∎

Fact 34. The current definition of BBI-tableau and the definition of TBBI-tableau of [22] characterise the same notion of provability: given a BBI-formula G and $a \neq b \in L$, there exists a closed BBI-tableau for the CSS $(\{\mathbb{F}G : a\}, \{a \multimapdot b\})$ if and only if there exists a closed TBBI-tableau for the one-branch and two-node tree $[a \multimapdot b, \mathbb{F}G : a]$.

[9] σ is injective on $X \subseteq L$ if for any $i, j \in X$, $\sigma(i) = \sigma(j)$ implies $i = j$; σ is injective if it is injective on $L \subseteq L$.

We do not give a formal proof of this result because it would require us to reintroduce the TBBI-tableau system already defined in [22]. Moreover, the argument would be a bit tedious and would likely hide the fundamental correspondence between those two systems. Indeed, *tableaux are just traces that keep track of some information about the (inductive) branch expansion process that builds the tableau*, i.e. when and how each fireable rule instance is applied. A tableau generally contains enough information to determine how its branches can be further expanded, but not necessarily how they were expanded so far (i.e. historic information on the expansion process). *The equivalence between the* BBI-*tableau system and the* TBBI-*tableau system lies in the fact that, despite different implementations of tableau and tableau branches, both systems define isomorphic branch expansion processes and have the same closure conditions on branches.* But in our current definition of BBI-tableaux, we don't keep track of how the tableau was built whereas we did keep this information in the definition of TBBI-tableaux [22]. This choice is directed by the use we intend for tableaux. Here, we focus on the completeness result and so we consider a tableau representation allowing the manipulation of limits (infinite branches, see Section 6.3), whereas in [22] we focused heavily on tableau proof transformations for which the ability of the reader to view and replay the branch expansion process was our primary concern.

5 Soundness of the BBI-tableau system

The soundness proof displayed here uses the same argumentation as the one we provided in [22]. But since tableaux are implemented a bit differently, the proofs are not strictly identical. It is decomposed in two parts. We define the notion of *realizability*, which is for a given tableau, the fact that one of its branches has a model. We then prove that closed tableaux are not realizable and that the branch expansion process preserves realizability.

We consider BBI-tableaux over the alphabet L and BBI-frames over the alphabet K where L and K have no relation a priori. So in statements like $\mathbb{S}F : m$ or in constraints like $m \mathrel{-\!\!\bullet\!\!-} n$, the words m, n belong to $L^\star$ whereas the relation $q \Vdash_\sim F$ or $q \nVdash_\sim F$ in the frame $(K, \sim, \Vdash)$ involves the word q belonging to $K^\star$. Recall (Definition 12) that a substitution $\rho : L \longrightarrow K^\star$ is extended into a morphism of (commutative) monoids by $\rho(m) = \rho(m_1 m_2 \ldots m_k) = \rho(m_1)\rho(m_2)\ldots\rho(m_k)$ when $m = m_1 m_2 \ldots m_k$ is a k letter word. We use substitutions here to make the correspondence between the syntactic world of tableau statements and constraints (over the alphabet L) and the semantic world of BBI-frames (over another alphabet K).

Definition 35 (Model/satisfaction). Given a tuple $\boldsymbol{\mathcal{K}} = (K, \sim, \Vdash, \rho)$ where $(K, \sim, \Vdash)$ is a BBI-frame and $\rho : L \longrightarrow K^\star$ is a substitution, we say that:

- *the statement* $\mathbb{T}A : m$ *is satisfied in* $\boldsymbol{\mathcal{K}}$ if $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \Vdash A$;
- *the statement* $\mathbb{F}A : m$ *is satisfied in* $\boldsymbol{\mathcal{K}}$ if $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \nVdash A$;
- *the constraint* $m \mathrel{-\!\!\bullet\!\!-} n$ *is satisfied in* $\boldsymbol{\mathcal{K}}$ if $\rho(m) \sim \rho(n)$;
- *the* CSS $(\mathcal{X}, \mathcal{C})$ *is satisfied in* $\boldsymbol{\mathcal{K}} = (K, \sim, \Vdash, \rho)$ if all the statements in $\mathcal{X}$ and all the constraints in $\mathcal{C}$ are satisfied in $\boldsymbol{\mathcal{K}}$.

As an alternative to "$(\mathcal{X}, \mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}}$," we might also say that $\boldsymbol{\mathcal{K}}$ *is a model of* $(\mathcal{X}, \mathcal{C})$. Finally we say that the CSS $(\mathcal{X}, \mathcal{C})$ *has a model* if there exists a tuple $\boldsymbol{\mathcal{K}}$ such that $(\mathcal{X}, \mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}}$.

Proposition 36. If a CSS $(\mathcal{X}, \mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}} = (K, \sim, \Vdash, \rho)$ and $m \sim_{\mathcal{C}} n$ then $\rho(m) \sim \rho(n)$.

Proof. The constraints of $\mathcal{C}$ are satisfied in $\boldsymbol{\mathcal{K}}$ if and only $\rho(\mathcal{C}) \subseteq \sim$. Thus we deduce $\sim_{\rho(\mathcal{C})} \subseteq \sim$ because $\sim$ is a PME. By Corollary 14, from $m \sim_{\mathcal{C}} n$, we deduce $\rho(m) \sim_{\rho(\mathcal{C})} \rho(n)$ and thus $\rho(m) \sim \rho(n)$. ■

Proposition 37. Let $(\mathcal{X},\mathcal{C})$ be a CSS. Let $(K,\sim,\Vdash)$ be a BBI-frame and $\rho,\rho' : L \longrightarrow K^\star$ be substitutions such that $\rho(a)=\rho'(a)$ for every letter $a \in \mathsf{A}_{\mathcal{C}}$. Then $(\mathcal{X},\mathcal{C})$ is satisfied in $(K,\sim,\Vdash,\rho)$ if and only if it is satisfied in $(K,\sim,\Vdash,\rho')$.

Proof. Left to the reader. ■

Definition 38 (Realizability). A BBI-tableau is *realizable* if at least one of its branches has a model.

Proposition 39. Closed BBI-tableaux are not realizable.

Proof. We prove that a closed branch $(\mathcal{X},\mathcal{C})$ cannot be satisfied in any $(K,\sim,\Vdash,\rho)$. Let us suppose the contrary and proceed by case analysis on the closure condition:

1. if $\mathbb{T}A : m \in \mathcal{X}$, $\mathbb{F}A : n \in \mathcal{X}$ and $m \sim_{\mathcal{C}} n$ then, as $(\mathcal{X},\mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}}$, we have $\rho(m) \sim \rho(n)$ by Proposition 36. Moreover, both $\mathbb{T}A : m$ and $\mathbb{F}A : n$ are satisfied in $\boldsymbol{\mathcal{K}}$ and thus $\rho(m) \Vdash A$ and $\rho(n) \nVdash A$. As $\rho(m) \sim \rho(n)$, we obtain a contradiction using the monotonicity of $\Vdash$ (see Proposition 16);
2. if $\mathbb{F}\mathsf{I} : m \in \mathcal{X}$ and $\epsilon \sim_{\mathcal{C}} m$ then, as $(\mathcal{X},\mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}}$, we have $\epsilon = \rho(\epsilon) \sim \rho(m)$ by Proposition 36 and also $\rho(m) \nVdash \mathsf{I}$. But then, we should have $\epsilon \not\sim \rho(m)$. Thus we obtain a contradiction.

So we obtain a contradiction in any case. A closed branch cannot be satisfied. Thus closed BBI-tableaux are not realizable. ■

Lemma 40. BBI-tableau expansion preserves realizability.

Proof. Let $\mathcal{T}$ be a realizable BBI-tableau and let $\boldsymbol{\mathcal{K}} = (K,\sim,\Vdash,\rho)$ be such that at least one branch of $\mathcal{T}$ is satisfied in $\boldsymbol{\mathcal{K}}$. We consider the expansion of one of the branches of $\mathcal{T}$ by one of the fireable rule instances of the BBI-tableau system of Table 2, obtaining the tableau $\mathcal{T}'$. If the expanded branch is not among the ones satisfied in $\boldsymbol{\mathcal{K}}$ then the satisfied branches are kept unchanged by the application of the rule and the obtained tableau $\mathcal{T}'$ still contains a branch satisfied in $\boldsymbol{\mathcal{K}}$. Hence $\mathcal{T}'$ is still realizable.

So we consider the case when the branch $(\mathcal{X},\mathcal{C})$ which is expanded is among the satisfied ones. We proceed by case analysis depending on the rule applied:

$\boxed{\mathbb{T}\neg A : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ and thus $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \Vdash \neg A$. Thus $\rho(m) \nVdash A$ and $\mathbb{F}A : m$ is satisfied in $\boldsymbol{\mathcal{K}}$. So the expanded branch $(\mathcal{X} \cup \{\mathbb{F}A : m\}, \mathcal{C} \cup \emptyset)$ of $\mathcal{T}'$ is satisfied in $\boldsymbol{\mathcal{K}}$;

$\boxed{\mathbb{F}\neg A : m}$ Similar to case $\mathbb{T}\neg$;

$\boxed{\mathbb{T}A \wedge B : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ hence $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \Vdash A \wedge B$. Then $\rho(m) \Vdash A$ and $\rho(m) \Vdash B$ hence $\mathbb{T}A : m$ and $\mathbb{T}B : m$ are satisfied in $\boldsymbol{\mathcal{K}}$. So the expanded branch $(\mathcal{X} \cup \{\mathbb{T}A : m, \mathbb{T}B : m\}, \mathcal{C})$ of $\mathcal{T}'$ is satisfied in $\boldsymbol{\mathcal{K}}$;

$\boxed{\mathbb{F}A \wedge B : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ hence $\rho(m) \in \mathsf{L}_\sim$ and either $\rho(m) \nVdash A$ or $\rho(m) \nVdash B$. Hence either $\mathbb{F}A : m$ or $\mathbb{F}B : m$ is satisfied in $\boldsymbol{\mathcal{K}}$. So at least one of the two expanded branches of $\mathcal{T}'$ (namely $(\mathcal{X} \cup \{\mathbb{T}A : m\}, \mathcal{C})$ or $(\mathcal{X} \cup \{\mathbb{T}B : m\}, \mathcal{C})$) is satisfied in $\boldsymbol{\mathcal{K}}$;

$\boxed{\mathbb{T}\mathsf{I} : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ hence $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \Vdash \mathsf{I}$. Thus $\epsilon \sim \rho(m)$. As $\rho(\epsilon) = \epsilon$, we obtain $\rho(\epsilon) \sim \rho(m)$ and thus the constraint $\epsilon \mathrel{-\!\!\bullet\!\!-} m$ is satisfied in $\boldsymbol{\mathcal{K}}$. So the expanded branch $(\mathcal{X}, \mathcal{C} \cup \{\epsilon \mathrel{-\!\!\bullet\!\!-} m\})$ of $\mathcal{T}'$ is satisfied in $\boldsymbol{\mathcal{K}}$;

$\boxed{\mathbb{T}A * B : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ hence $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \Vdash A * B$. So there exists $x, y \in \mathsf{L}_\sim$ such that $xy \sim \rho(m)$, $x \Vdash A$ and $y \Vdash B$. We define $\rho' = \rho[a \mapsto x, b \mapsto y]$ (possible because $a \neq b$). Then for any $m, n \in L^\star$ s.t. $m \sim_{\mathcal{C}} n$ we have $m, n \in \mathsf{A}_{\mathcal{C}}^\star$ and thus $\rho'(m) = \rho(m)$ and $\rho'(n) = \rho(n)$ (ρ and ρ' are identical maps when restricted to $\mathsf{A}_{\mathcal{C}}$ because $a, b \notin \mathsf{A}_{\mathcal{C}}$). Thus by Proposition 37, $(\mathcal{X},\mathcal{C})$ is satisfied in $\boldsymbol{\mathcal{K}}' = (K,\sim,\Vdash,\rho')$. Moreover, $ab \mathrel{-\!\!\bullet\!\!-} m$ is satisfied in $\boldsymbol{\mathcal{K}}'$ (because $\rho'(ab) = xy$, $\rho'(m) = \rho(m)$ and $xy \sim \rho(m)$), $\mathbb{T}A : a$ is satisfied (because $\rho'(a) = x$ and $x \Vdash A$), and $\mathbb{T}B : b$ is satisfied (because $\rho'(b) = y$ and $y \Vdash B$). So the expanded branch $(\mathcal{X} \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C} \cup \{ab \mathrel{-\!\!\bullet\!\!-} m\})$ of $\mathcal{T}'$ is satisfied in $\boldsymbol{\mathcal{K}}'$;

$\boxed{\mathbb{F}A * B : m \in \mathcal{X}}$ is satisfied in $\boldsymbol{\mathcal{K}}$ hence $\rho(m) \in \mathsf{L}_\sim$ and $\rho(m) \nVdash A * B$. $(\mathcal{X}, \mathcal{C})$ is expanded into two branches $(\mathcal{X} \cup \{\mathbb{F}A : x\}, \mathcal{C})$ and $(\mathcal{X} \cup \{\mathbb{F}B : y\}, \mathcal{C})$ with x, y s.t. $xy \sim_\mathcal{C} m$. Then $\rho(x)\rho(y) \sim \rho(m)$ holds by Proposition 36, and thus the inclusion $\{\rho(x), \rho(y)\} \subseteq \mathsf{L}_\sim$ holds. So either $\rho(x) \nVdash A$ or $\rho(y) \nVdash B$. Thus at least one of the two expanded branches of $\mathcal{T}'$ is satisfied in $\boldsymbol{\mathcal{K}}$;

$\boxed{\mathbb{T}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ Similar to case $\mathbb{F}*$;

$\boxed{\mathbb{F}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ Similar to case $\mathbb{T}*$.

In any case, there exists $\boldsymbol{\mathcal{K}}'$ and a branch of $\mathcal{T}'$ satisfied in $\boldsymbol{\mathcal{K}}'$, so $\mathcal{T}'$ is realizable. ■

Theorem 41 (Soundness of the BBI-tableau system). Given $a \in L$ and $G \in \mathsf{Form}$, if there exists a closed BBI-tableau for $(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})$ then G is a universally valid BBI formula.

Proof. Let us suppose that there exists a closed BBI-tableau for $(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})$. Let us now suppose that G has a counter-model $(K, \sim, \Vdash, m)$, i.e. $m \in \mathsf{L}_\sim$ and $m \nVdash G$. Then the CSS $(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})$ is satisfied in $(K, \sim, \Vdash, \rho)$ where $\rho = x \mapsto m$ (in particular $\rho(a) = m$). Hence, the BBI-tableau $[(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})]$ is realizable and as a consequence, any BBI-tableau for $(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})$ is realizable by Lemma 40. By Proposition 39, no BBI-tableau for $(\{\mathbb{F}G : a\}, \{a \mathrel{-\!\mapstochar\!-} a\})$ can be closed. We obtain a contradiction. Hence G has no counter-model. ■

6 Completeness of the BBI-tableau system

In this section, we give the full proof of strong completeness of the BBI-tableau system, expressed in Theorem 56. We define the notion of Hintikka CSS and show how to extract a Herbrand counter-model from a Hintikka CSS. Then using fair strategies and oracles based on proof-search, we show how to extend any finite CSS into a Hintikka CSS. The constraints that generate the counter-model extracted from such a Hintikka CSS originate from proof-search and are thus of a specific form that we characterise as simple. The strong completeness theorem ends this section.

This (new) proof is somewhat inspired by ideas developed in the reference textbook [12] in the completeness proof of the tableau method for first-order logic.

6.1 Hintikka CSS and BBI counter-models

Hintikka sets [16] are saturated syntactic objects from which it is possible to extract (counter-)models. We define the notion of Hintikka CSS corresponding to the labeled Kripke semantics of BBI.

Definition 42 (Hintikka CSS). A *Hintikka* CSS is a CSS $(\mathcal{X}, \mathcal{C})$ such that for any $A, B \in \mathsf{Form}$ and any $m, n \in L^\star$:

1. not both $\mathbb{T}A : m \in \mathcal{X}$, $\mathbb{F}A : n \in \mathcal{X}$ and $m \sim_\mathcal{C} n$;
2. if $\mathbb{T}\neg A : m \in \mathcal{X}$ then $\mathbb{F}A : m \in \mathcal{X}$;
3. if $\mathbb{F}\neg A : m \in \mathcal{X}$ then $\mathbb{T}A : m \in \mathcal{X}$;
4. if $\mathbb{T}A \wedge B : m \in \mathcal{X}$ then $\{\mathbb{T}A : m, \mathbb{T}B : m\} \subseteq \mathcal{X}$;
5. if $\mathbb{F}A \wedge B : m \in \mathcal{X}$ then $\{\mathbb{F}A : m, \mathbb{F}B : m\} \cap \mathcal{X} \neq \emptyset$;
6. if $\mathbb{T}\mathsf{I} : m \in \mathcal{X}$ then $\epsilon \sim_\mathcal{C} m$;
7. not both $\mathbb{F}\mathsf{I} : m \in \mathcal{X}$ and $\epsilon \sim_\mathcal{C} m$;
8. if $\mathbb{T}A * B : m \in \mathcal{X}$ then $\exists x, y \in L^\star$, $xy \sim_\mathcal{C} m \wedge \{\mathbb{T}A : x, \mathbb{T}B : y\} \subseteq \mathcal{X}$;
9. if $\mathbb{F}A * B : m \in \mathcal{X}$ then $\forall x, y \in L^\star$, $xy \sim_\mathcal{C} m \Rightarrow \{\mathbb{F}A : x, \mathbb{F}B : y\} \cap \mathcal{X} \neq \emptyset$;

10. if $\mathbb{T}A \mathrel{-\!\!*} B : m \in \mathcal{X}$ then $\forall x, y \in L^\star$, $xm \sim_{\mathcal{C}} y \Rightarrow \{\mathbb{F}A : x, \mathbb{T}B : y\} \cap \mathcal{X} \neq \emptyset$;
11. if $\mathbb{F}A \mathrel{-\!\!*} B : m \in \mathcal{X}$ then $\exists x, y \in L^\star$, $xm \sim_{\mathcal{C}} y \wedge \{\mathbb{T}A : x, \mathbb{F}B : y\} \subseteq \mathcal{X}$.

Conditions 1 and 7 say that the CSS $(\mathcal{X}, \mathcal{C})$ is open: these are consistency conditions. The other conditions express a saturation of $(\mathcal{X}, \mathcal{C})$ by the equivalences defining the Kripke semantics. Given a tuple $\boldsymbol{\mathcal{K}} = (K, \sim, \Vdash, \rho)$ where $(K, \sim, \Vdash)$ is a BBI-frame and $\rho : L \longrightarrow K^\star$ is a substitution, it is easy to build a Hintikka CSS : simply choose $\mathcal{X}$ as the set of statements that are satisfied in $\boldsymbol{\mathcal{K}}$ and then $(\mathcal{X}, \sim)$ is a Hintikka CSS. What we are interested in here is the other way around: build a Hintikka CSS out of "unprovability" (which is the main difficulty) and then extract a (counter-)model out of that Hintikka CSS. Let us first show how to extract a (counter-)model out of a Hintikka CSS, which justifies the view of Hintikka CSS as syntactic counter-models.

Lemma 43 (Herbrand model). Let $(\mathcal{X}, \mathcal{C})$ be a Hintikka CSS. The tuple $(L, \sim_{\mathcal{C}}, \Vdash, x \mapsto x)$ is a model of the CSS $(\mathcal{X}, \mathcal{C})$ when $\Vdash \subseteq \mathsf{L}_{\overline{\mathcal{C}}} \times \mathsf{Var}$ is defined for $m \in \mathsf{L}_{\overline{\mathcal{C}}}$ and $Z \in \mathsf{Var}$ by:

$$m \Vdash Z \quad \text{iff} \quad \mathbb{T}Z : n \in \mathcal{X} \text{ and } n \sim_{\mathcal{C}} m \text{ hold for some } n$$

Proof. Let us first prove that $\Vdash$ is monotonic. So let us consider $Z \in \mathsf{Var}$, $m, n \in \mathsf{L}_{\overline{\mathcal{C}}}$ such that $m \sim_{\mathcal{C}} n$ and $m \Vdash Z$. By definition of $\Vdash$, there exists a k such that $\mathbb{T}Z : k \in \mathcal{X}$ and $k \sim_{\mathcal{C}} m$. By rule $\langle t \rangle$, we deduce $k \sim_{\mathcal{C}} n$ and thus by definition of $\Vdash$ we obtain $n \Vdash Z$.

Since the substitution of the Herbrand model is the identity substitution $\sigma = x \mapsto x$, all the constraints in $\mathcal{C}$ are satisfied in the PME $\sim_{\mathcal{C}}$, i.e. $\mathcal{C} = \sigma(\mathcal{C}) \subseteq \sim_{\mathcal{C}}$. Now we prove that the statements of $\mathcal{X}$ are also satisfied in the Herbrand model. First remark that when $\mathbb{S}G : m \in \mathcal{X}$, we have $m \sim_{\mathcal{C}} m$ because $(\mathcal{X}, \mathcal{C})$ is a CSS, and thus $m \in \mathsf{L}_{\overline{\mathcal{C}}}$. Now let us prove $\mathbb{T}G : m \in \mathcal{X} \Rightarrow m \Vdash G$ and $\mathbb{F}G : m \in \mathcal{X} \Rightarrow m \nVdash G$ by mutual induction on the formula $G \in \mathsf{Form}$ and by case analysis on $\mathbb{S}G : m$.

$\boxed{\mathbb{T}Z : m \in \mathcal{X}}$ since $m \sim_{\mathcal{C}} m$ holds (because $(\mathcal{X}, \mathcal{C})$ is a CSS) and $\mathbb{T}Z : m \in \mathcal{X}$, we have $m \Vdash Z$;

$\boxed{\mathbb{F}Z : m \in \mathcal{X}}$ let us suppose $m \Vdash Z$. Then, by definition of $\Vdash$, we would obtain n such that $n \sim_{\mathcal{C}} m$ and $\mathbb{T}Z : n \in \mathcal{X}$. We would thus have simultaneously $\mathbb{T}Z : n \in \mathcal{X}$, $\mathbb{F}Z : m \in \mathcal{X}$ and $n \sim_{\mathcal{C}} m$. Since $(\mathcal{X}, \mathcal{C})$ is a Hintikka CSS, by condition 1 of Definition 42, we obtain a contradiction. Thus $m \nVdash Z$;

$\boxed{\mathbb{T}\neg A : m \in \mathcal{X}}$ by condition 2 of Definition 42, we obtain $\mathbb{F}A : m \in \mathcal{X}$, and thus by induction, $m \nVdash A$, hence $m \Vdash \neg A$;

$\boxed{\mathbb{F}\neg A : m \in \mathcal{X}}$ by condition 3 of Definition 42, we obtain $\mathbb{T}A : m \in \mathcal{X}$, and thus by induction, $m \Vdash A$, hence $m \nVdash \neg A$;

$\boxed{\mathbb{T}A \wedge B : m \in \mathcal{X}}$ by condition 4 of Definition 42, we have $\mathbb{T}A : m, \mathbb{T}B : m \in \mathcal{X}$ and by induction, $m \Vdash A$ and $m \Vdash B$. Hence, $m \Vdash A \wedge B$;

$\boxed{\mathbb{F}A \wedge B : m \in \mathcal{X}}$ by condition 5 of Definition 42, either $\mathbb{F}A : m \in \mathcal{X}$ or $\mathbb{F}B : m \in \mathcal{X}$. So by induction, either $m \nVdash A$ or $m \nVdash B$. Then $m \nVdash A \wedge B$;

$\boxed{\mathbb{T}\mathsf{I} : m \in \mathcal{X}}$ by condition 6 of Definition 42, we have $\epsilon \sim_{\mathcal{C}} m$. By definition of $\Vdash$, we obtain $m \Vdash \mathsf{I}$;

$\boxed{\mathbb{F}\mathsf{I} : m \in \mathcal{X}}$ by condition 7 of Definition 42, $\epsilon \not\sim_{\mathcal{C}} m$ and thus $m \nVdash \mathsf{I}$;

$\boxed{\mathbb{T}A * B : m \in \mathcal{X}}$ by condition 8 of Definition 42, there exist $x, y \in L^\star$ such that $xy \sim_{\mathcal{C}} m$ and $\mathbb{T}A : x, \mathbb{T}B : y \in \mathcal{X}$. So by induction, we obtain $x \Vdash A$ and $y \Vdash B$. Then $m \Vdash A * B$;

$\boxed{\mathbb{F}A * B : m \in \mathcal{X}}$ by condition 9 of Definition 42, for every $x, y \in L^\star$ such that $xy \sim_{\mathcal{C}} m$, we have either $\mathbb{F}A : x \in \mathcal{X}$ or $\mathbb{F}B : y \in \mathcal{X}$. Hence, by induction, either $x \nVdash A$ or $y \nVdash B$ for every x, y such that $xy \sim_{\mathcal{C}} m$. We conclude $m \nVdash A * B$;

$\boxed{\mathbb{T}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ similar to $\mathbb{F}A * B : m$ but use condition 10 of Definition 42;

$\boxed{\mathbb{F}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ similar to $\mathbb{T}A * B : m$ but use condition 11 of Definition 42.

So by induction on G, every statement $\mathbb{S}G : m \in \mathcal{X}$ is satisfied in $(L, \sim_{\mathcal{C}}, \Vdash, x \mapsto x)$. ■

Hence, the BBI-frame $(L, \sim_{\mathcal{C}}, \Vdash)$ extracted from a Hintikka CSS is a model of this CSS. Hintikka CSS are syntactic representations of (counter-)models of BBI formulæ.

Corollary 44. Let $(\mathcal{X}, \mathcal{C})$ be a Hintikka CSS, $G \in \mathsf{Form}$ and $m \in L^\star$ be such that $\mathbb{F}G : m \in \mathcal{X}$. Then $(L, \sim_{\mathcal{C}}, \Vdash, m)$ is a counter-model of G.

Proof. By Lemma 43, since $\mathbb{F}G : m \in \mathcal{X}$, the statement $\mathbb{F}G : m$ is satisfied in the tuple $(L, \sim_{\mathcal{C}}, \Vdash, \sigma = x \mapsto x)$ and we deduce $m = \sigma(m) \in \mathsf{L}_{\overline{\mathcal{C}}}$ and $m = \sigma(m) \nVdash G$. ■

6.2 Fair strategy, oracles and consistency

We now prove that any finite CSS having no closed tableau can be extended into a Hintikka CSS. Using a fair strategy and an oracle that contains all finite and consistent CSS, we build a sequence of CSS which saturates the initial CSS into a Hintikka CSS. The consistency criterion is a syntactic one: having no closed tableau.

Definition 45 (Fair strategy). *A fair strategy* is a sequence $(\mathbb{S}_i F_i : m_i)_{i \in \mathbb{N}}$ of tableau statements such that any tableau statement (in $\{\mathbb{T}, \mathbb{F}\} \times \mathsf{Form} \times L^\star$) occurs infinitely many times in this sequence, i.e. $\{i \in \mathbb{N} \mid \mathbb{S}_i F_i : m_i \equiv \mathbb{S}F : m\}$ is infinite for any $\mathbb{S}F : m \in \{\mathbb{T}, \mathbb{F}\} \times \mathsf{Form} \times L^\star$.

Proposition 46. There exists a fair strategy.[10]

Proof. Let $X = \{\mathbb{T}, \mathbb{F}\} \times \mathsf{Form} \times L^\star$. As $L = \{c_0, c_1, c_2, \ldots\}$ is countable and Var is countable, then so are Form and $L^\star$. Hence X is a countable set as a product of countable sets. So $\mathbb{N} \times X$ is also countable and there exists a surjective function $\varphi : \mathbb{N} \longrightarrow \mathbb{N} \times X$. Let $p : \mathbb{N} \times X \longrightarrow X$ be the canonical projection defined by $p(i, x) = x$. Then let us define $u : \mathbb{N} \longrightarrow X$ by $u = p \circ \varphi$ and let us prove that u is a fair strategy. It is sufficient to show that $u^{-1}(\{x\})$ is infinite for any $x \in X$. Let $x \in X$. Then $u^{-1}(\{x\}) = \varphi^{-1}(p^{-1}(\{x\}))$. But $p^{-1}(\{x\}) = \{(i, x) \mid i \in \mathbb{N}\}$ hence $p^{-1}(\{x\})$ is infinite. As φ is surjective, $\varphi^{-1}(p^{-1}(\{x\}))$ is also infinite. ■

Definition 47 (Oracle). An *oracle* is a set of CSS which is $\preccurlyeq$-downward closed, of finite character, open and saturated. These characteristic properties of oracles are defined for any set $\mathfrak{P}$ of CSS by

- $\mathfrak{P}$ *is $\preccurlyeq$-downward closed* if $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$ whenever both $(\mathcal{X}, \mathcal{C}) \preccurlyeq (\mathcal{X}', \mathcal{C}')$ and $(\mathcal{X}', \mathcal{C}') \in \mathfrak{P}$ hold;
- $\mathfrak{P}$ *is of finite character* if $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$ whenever $(\mathcal{X}_f, \mathcal{C}_f) \in \mathfrak{P}$ holds for every $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$;[11]
- $\mathfrak{P}$ *is open* if $(\mathcal{X}, \mathcal{C})$ is open for every $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$;
- $\mathfrak{P}$ *is saturated* if for any $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$ and any instance of a rule of Table 2 fireable on $(\mathcal{X}, \mathcal{C})$, at least one of its expansions $(\mathcal{X} \cup \mathcal{X}_i, \mathcal{C} \cup \mathcal{C}_i)$ belongs to $\mathfrak{P}$.

As a side remark, in the terminology of Fitting [12], a set of branches which is open and saturated is called an "alternate consistency property." The last condition ensures that whichever fireable rule instance you choose to expand a CSS inside an oracle, there exists one expansion which stays in the oracle. Hence, the expansion process cannot be blocked in an oracle. We point out that it would be a to strong requirement to force that every expansion stays in the oracle.

[10] Remark: provided L and Var can be *effectively* enumerated, then we can obtain a recursive fair strategy: there is no need for the axiom of choice or any of its weaker forms here. See the Coq proof comments in Section 7.

[11] i.e. $(\mathcal{X}, \mathcal{C})$ belongs to $\mathfrak{P}$ as soon as all its finite approximations belong to $\mathfrak{P}$.

Definition 48 (Consistency and finite consistency). Let $(\mathcal{X}, \mathcal{C})$ be a CSS. If $(\mathcal{X}, \mathcal{C})$ is a finite CSS, we say that $(\mathcal{X}, \mathcal{C})$ *is consistent* if it has no closed BBI-tableau. If $(\mathcal{X}, \mathcal{C})$ is a (finite or infinite) CSS, we say that $(\mathcal{X}, \mathcal{C})$ *is finitely consistent* if every finite sub-CSS of $(\mathcal{X}, \mathcal{C})$ is consistent.

Proposition 49. Consistency is a $\preccurlyeq$-downward closed property, i.e. if $(\mathcal{X}_1, \mathcal{C}_1) \preccurlyeq (\mathcal{X}_2, \mathcal{C}_2)$ are two finite CSS and $(\mathcal{X}_2, \mathcal{C}_2)$ is consistent then $(\mathcal{X}_1, \mathcal{C}_1)$ is consistent.

Proof. This proposition is a specialisation of Theorem 31 (if $\mathcal{C}_1 \subseteq \mathcal{C}_2$ then $\mathcal{C}_1 \subseteq \overline{\mathcal{C}_2}$). ■

Corollary 50. A finite CSS is consistent if and only if it is finitely consistent.

Proof. Immediate consequence of Proposition 49. ■

We now introduce and prove the main lemma of this paper: the set of finitely consistent CSS is an oracle. In Section 6.3, it will allow us to saturate any finite and consistent CSS into a Hintikka CSS.

Lemma 51. The set of finitely consistent CSS is an oracle; its contains any (finite) consistent CSS.[12]

Proof. Let $\mathfrak{P}$ be the set of finitely consistent CSS. By Corollary 50, for any finite CSS $(\mathcal{X}_f, \mathcal{C}_f)$, we have $(\mathcal{X}_f, \mathcal{C}_f) \in \mathfrak{P}$ if and only if $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent. So $\mathfrak{P}$ contains any (finite) consistent CSS.

Let us prove that $\mathfrak{P}$ is $\preccurlyeq$-downward closed. Indeed, if $(\mathcal{X}, \mathcal{C}) \preccurlyeq (\mathcal{X}', \mathcal{C}')$ and $(\mathcal{X}', \mathcal{C}') \in \mathfrak{P}$ hold. Let us prove that $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$ holds., i.e. that $(\mathcal{X}, \mathcal{C})$ is finitely consistent. Let us consider a finite sub-CSS $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$. We derive $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}', \mathcal{C}')$ and thus $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent because $(\mathcal{X}', \mathcal{C}')$ is finitely consistent. Since every finite sub-CSS of $(\mathcal{X}, \mathcal{C})$ is consistent, we deduce $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$.

Let us show that $\mathfrak{P}$ is of finite character. Let $(\mathcal{X}, \mathcal{C})$ be a CSS such that $(\mathcal{X}_f, \mathcal{C}_f) \in \mathfrak{P}$ holds for any $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$. Let $(\mathcal{X}_f, \mathcal{C}_f)$ be such that $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$. Then $(\mathcal{X}_f, \mathcal{C}_f)$ is a finite CSS that belongs to $\mathfrak{P}$, hence it is consistent by Corollary 50. Thus we have proved $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$.

We now prove that $\mathfrak{P}$ is open. Let $(\mathcal{X}, \mathcal{C})$ be a closed CSS. Let us prove $(\mathcal{X}, \mathcal{C}) \notin \mathfrak{P}$. By Proposition 27, there exists $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$ such that $(\mathcal{X}_f, \mathcal{C}_f)$ is a closed CSS. Then $[(\mathcal{X}_f, \mathcal{C}_f)]$ is a closed tableau for $(\mathcal{X}_f, \mathcal{C}_f)$. Hence $(\mathcal{X}_f, \mathcal{C}_f)$ is not consistent and we deduce $(\mathcal{X}, \mathcal{C}) \notin \mathfrak{P}$.

We finish by the proof that $\mathfrak{P}$ is saturated. Let us fix a CSS $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$. We consider each possible instance of a tableau expansion rule of Table 2 fireable for $(\mathcal{X}, \mathcal{C})$:

$\boxed{\mathbb{T}\neg A : m \in \mathcal{X}}$ We show that $(\mathcal{X} \cup \{\mathbb{F}A : m\}, \mathcal{C})$ belongs to $\mathfrak{P}$. Thus, let us consider $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{F}A : m\}, \mathcal{C})$ and let us show that $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent. Since $(\mathcal{X}, \mathcal{C})$ is a CSS and $\mathbb{T}\neg A : m \in \mathcal{X}$, we have $m \sim_{\mathcal{C}} m$. By compactness, there exists a finite subset $\mathcal{C}_0 \subseteq \mathcal{C}$ such that $m \sim_{\mathcal{C}_0} m$. Let $\mathcal{X}'_f = (\mathcal{X}_f \setminus \{\mathbb{F}A : m\}) \cup \{\mathbb{T}\neg A : m\}$ and $\mathcal{C}'_f = \mathcal{C}_f \cup \mathcal{C}_0$. Then $(\mathcal{X}'_f, \mathcal{C}'_f)$ is a finite CSS and the inclusion $(\mathcal{X}'_f, \mathcal{C}'_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$ holds. From $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$, we deduce that $(\mathcal{X}'_f, \mathcal{C}'_f)$ is consistent. Since $\mathbb{T}\neg A : m \in \mathcal{X}'_f$, the list $[(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)]$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Hence, if $(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ has a closed tableau $\mathcal{T}$, by Proposition 24, $\mathcal{T}$ would also be a closed tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$ which would contradict the consistency of $(\mathcal{X}'_f, \mathcal{C}'_f)$. As consequence, $(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ cannot have a closed tableau and is thus consistent. From $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq (\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ and Proposition 49, we deduce the consistency $(\mathcal{X}_f, \mathcal{C}_f)$;

$\boxed{\mathbb{F}\neg A : m \in \mathcal{X}}$ Similar to case $\mathbb{T}\neg$;

$\boxed{\mathbb{T}A \wedge B : m \in \mathcal{X}}$ Similar to case $\mathbb{T}\neg$;

$\boxed{\mathbb{F}A \wedge B : m \in \mathcal{X}}$ Let us suppose by absurd that neither $(\mathcal{X} \cup \{\mathbb{F}A : m\}, \mathcal{C}) \in \mathfrak{P}$ nor $(\mathcal{X} \cup \{\mathbb{F}B : m\}, \mathcal{C}) \in \mathfrak{P}$ hold. Then there exists $(\mathcal{X}^A_f, \mathcal{C}^A_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{F}A : m\}, \mathcal{C})$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{F}B : m\}, \mathcal{C})$ such that both $(\mathcal{X}^A_f, \mathcal{C}^A_f)$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f)$ are inconsistent. By compactness, there also exists a finite subset $\mathcal{C}_0 \subseteq \mathcal{C}$ such that $m \sim_{\mathcal{C}_0} m$. Let $\mathcal{X}'_f = \mathcal{X}^A_f \setminus \{\mathbb{F}A : m\} \cup \mathcal{X}^B_f \setminus \{\mathbb{F}B : m\} \cup \{\mathbb{F}A \wedge B : m\}$

[12] Beware that the oracle itself is generally not a finite set.

and $\mathcal{C}'_f = \mathcal{C}^A_f \cup \mathcal{C}^B_f \cup \mathcal{C}_0$. $(\mathcal{X}'_f, \mathcal{C}'_f)$ is obviously a finite CSS. Since $\mathbb{F}A \wedge B : m \in \mathcal{X}'_f$ holds, the list $[(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f); (\mathcal{X}'_f \cup \{\mathbb{F}B : m\}, \mathcal{C}'_f)]$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Since $(\mathcal{X}^A_f, \mathcal{C}^A_f) \preccurlyeq (\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f) \preccurlyeq (\mathcal{X}'_f \cup \{\mathbb{F}B : m\}, \mathcal{C}'_f)$, by Proposition 49, the finite CSS $(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ and $(\mathcal{X}'_f \cup \{\mathbb{F}B : m\}, \mathcal{C}'_f)$ are both inconsistent. Let $\mathcal{T}_A$ (resp. $\mathcal{T}_B$) be a closed tableau for $(\mathcal{X}'_f \cup \{\mathbb{F}A : m\}, \mathcal{C}'_f)$ (resp. $(\mathcal{X}'_f \cup \{\mathbb{F}B : m\}, \mathcal{C}'_f)$). By two applications of Proposition 24, $\mathcal{T}_A +\!\!+\, \mathcal{T}_B$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$ which is closed because both $\mathcal{T}_A$ and $\mathcal{T}_B$ are closed. Hence $(\mathcal{X}'_f, \mathcal{C}'_f)$ is inconsistent. But we easily check that $(\mathcal{X}'_f, \mathcal{C}'_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$, which contradicts $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$;

$\boxed{\mathbb{T}\mathsf{I} : m \in \mathcal{X}}$ We show that $(\mathcal{X}, \mathcal{C} \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})$ belongs to $\mathfrak{P}$. Thus, let us consider $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C} \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})$ and let us show that $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent. Since $(\mathcal{X}, \mathcal{C})$ is a CSS and $\mathcal{X}_f \subseteq \mathcal{X}$ then $(\mathcal{X}_f, \mathcal{C})$ is also a CSS. By Proposition 20, there exists a finite subset $\mathcal{C}_0 \subseteq \mathcal{C}$ such that $(\mathcal{X}_f, \mathcal{C}_0)$ is a (finite) CSS. From $\mathbb{T}\mathsf{I} : m \in \mathcal{X}$, we deduce $m \sim_{\mathcal{C}} m$. Thus, by Proposition 7, let $\mathcal{C}_1$ be a finite subset of $\mathcal{C}$ such that $m \sim_{\mathcal{C}_1} m$. Let $\mathcal{X}'_f = \mathcal{X}_f \cup \{\mathbb{T}\mathsf{I} : m\}$ and $\mathcal{C}'_f = \mathcal{C}_f \backslash \{\epsilon \mathrel{-\!\bullet\!-} m\} \cup \mathcal{C}_0 \cup \mathcal{C}_1$. Then $(\mathcal{X}'_f, \mathcal{C}'_f)$ is a finite CSS and $(\mathcal{X}'_f, \mathcal{C}'_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$. From $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$, we deduce that $(\mathcal{X}'_f, \mathcal{C}'_f)$ is consistent. Since $\mathbb{T}\mathsf{I} : m \in \mathcal{X}'_f$, the list $[(\mathcal{X}'_f, \mathcal{C}'_f \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})]$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Hence, if $(\mathcal{X}'_f, \mathcal{C}'_f \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})$ has a closed tableau $\mathcal{T}$, by Proposition 24, $\mathcal{T}$ would also be a closed tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$ which would contradict the consistency of $(\mathcal{X}'_f, \mathcal{C}'_f)$. As consequence, $(\mathcal{X}'_f, \mathcal{C}'_f \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})$ cannot have a closed tableau and is thus consistent. From $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq (\mathcal{X}'_f, \mathcal{C}'_f \cup \{\epsilon \mathrel{-\!\bullet\!-} m\})$ and Proposition 49, we deduce that $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent;

$\boxed{\mathbb{T}A * B : m \in \mathcal{X}}$ with $a \neq b \in L \backslash \mathcal{A}_{\mathcal{C}}$. We show that $(\mathcal{X} \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C} \cup \{ab \mathrel{-\!\bullet\!-} m\})$ belongs to $\mathfrak{P}$. Thus, let us consider $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C} \cup \{ab \mathrel{-\!\bullet\!-} m\})$ and show that $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent. Let $\mathcal{X}'_f = \mathcal{X}_f \backslash \{\mathbb{T}A : a, \mathbb{T}B : b\} \cup \{\mathbb{T}A * B : m\}$. Since $\mathcal{X}'_f \subseteq \mathcal{X}$, then $(\mathcal{X}'_f, \mathcal{C})$ is a CSS where $\mathcal{X}'_f$ is finite and by Proposition 20, there exists a finite subset $\mathcal{C}_0 \subseteq \mathcal{C}$ such that $(\mathcal{X}'_f, \mathcal{C}_0)$ is a (finite) CSS. Let $\mathcal{C}'_f = \mathcal{C}_f \backslash \{ab \mathrel{-\!\bullet\!-} m\} \cup \mathcal{C}_0$. Then $(\mathcal{X}'_f, \mathcal{C}'_f)$ is also a finite CSS. We observe that $(\mathcal{X}'_f, \mathcal{C}'_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$ and we deduce that $(\mathcal{X}'_f, \mathcal{C}'_f)$ is consistent. Because $\mathbb{T}A * B : m \in \mathcal{X}'_f$ and $\mathcal{A}_{\mathcal{C}'_f} \subseteq \mathcal{A}_{\mathcal{C}}$, it is easy to check that $[(\mathcal{X}'_f \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C}'_f \cup \{ab \mathrel{-\!\bullet\!-} m\})]$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Hence by Proposition 24, $(\mathcal{X}'_f \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C}'_f \cup \{ab \mathrel{-\!\bullet\!-} m\})$ cannot have a closed tableau; otherwise we would obtain a closed tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Thus the CSS $(\mathcal{X}'_f \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}, \mathcal{C}'_f \cup \{ab \mathrel{-\!\bullet\!-} m\})$ is consistent. From the inclusions $\mathcal{X}_f \subseteq \mathcal{X}'_f \cup \{\mathbb{T}A : a, \mathbb{T}B : b\}$ and $\mathcal{C}_f \subseteq \mathcal{C}'_f \cup \{ab \mathrel{-\!\bullet\!-} m\}$, we deduce that $(\mathcal{X}_f, \mathcal{C}_f)$ is consistent by Proposition 49;

$\boxed{\mathbb{F}A * B : m \in \mathcal{X}}$ with $x, y \in L^\star$ such that $xy \sim_{\mathcal{C}} m$ holds. Let us suppose by absurd that neither $(\mathcal{X} \cup \{\mathbb{F}A : x\}, \mathcal{C}) \in \mathfrak{P}$ nor $(\mathcal{X} \cup \{\mathbb{F}B : y\}, \mathcal{C}) \in \mathfrak{P}$ hold. Then there exists $(\mathcal{X}^A_f, \mathcal{C}^A_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{F}A : x\}, \mathcal{C})$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f) \preccurlyeq_f (\mathcal{X} \cup \{\mathbb{F}B : y\}, \mathcal{C})$ such that $(\mathcal{X}^A_f, \mathcal{C}^A_f)$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f)$ are two inconsistent finite CSS. By compactness (Proposition 7), there also exists a finite subset $\mathcal{C}_0 \subseteq \mathcal{C}$ such that $xy \sim_{\mathcal{C}_0} m$. Let $\mathcal{X}'_f = \mathcal{X}^A_f \backslash \{\mathbb{F}A : x\} \cup \mathcal{X}^B_f \backslash \{\mathbb{F}B : y\} \cup \{\mathbb{F}A * B : m\}$ and $\mathcal{C}'_f = \mathcal{C}^A_f \cup \mathcal{C}^B_f \cup \mathcal{C}_0$. $(\mathcal{X}'_f, \mathcal{C}'_f)$ is obviously a finite CSS. Since $\mathbb{F}A * B : m \in \mathcal{X}'_f$ and $xy \sim_{\mathcal{C}'_f} m$ both hold, the list $[(\mathcal{X}'_f \cup \{\mathbb{F}A : x\}, \mathcal{C}'_f); (\mathcal{X}'_f \cup \{\mathbb{F}B : y\}, \mathcal{C}'_f)]$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$. Since $(\mathcal{X}^A_f, \mathcal{C}^A_f) \preccurlyeq (\mathcal{X}'_f \cup \{\mathbb{F}A : x\}, \mathcal{C}'_f)$ and $(\mathcal{X}^B_f, \mathcal{C}^B_f) \preccurlyeq (\mathcal{X}'_f \cup \{\mathbb{F}B : y\}, \mathcal{C}'_f)$, by Proposition 49, the finite CSS $(\mathcal{X}'_f \cup \{\mathbb{F}A : x\}, \mathcal{C}'_f)$ and $(\mathcal{X}'_f \cup \{\mathbb{F}B : y\}, \mathcal{C}'_f)$ are both inconsistent. Let $\mathcal{T}_A$ (resp. $\mathcal{T}_B$) be a closed tableau for $(\mathcal{X}'_f \cup \{\mathbb{F}A : x\}, \mathcal{C}'_f)$ (resp. $(\mathcal{X}'_f \cup \{\mathbb{F}B : y\}, \mathcal{C}'_f)$). By two applications of Proposition 24, $\mathcal{T}_A +\!\!+\, \mathcal{T}_B$ is a tableau for $(\mathcal{X}'_f, \mathcal{C}'_f)$ which is closed because both $\mathcal{T}_A$ and $\mathcal{T}_B$ are closed. Hence $(\mathcal{X}'_f, \mathcal{C}'_f)$ is inconsistent. But we easily check that $(\mathcal{X}'_f, \mathcal{C}'_f) \preccurlyeq_f (\mathcal{X}, \mathcal{C})$, which contradicts $(\mathcal{X}, \mathcal{C}) \in \mathfrak{P}$;

$\boxed{\mathbb{T}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ Similar to case $\mathbb{F}*$;

$\boxed{\mathbb{F}A \mathrel{-\!\!*} B : m \in \mathcal{X}}$ Similar to case $\mathbb{T}*$.

Hence $\mathfrak{P}$ is saturated by all fireable rule instances. ■

6.3 Obtaining a Hintikka CSS by saturation

We now build a Hintikka CSS using the combination of a fair strategy (Proposition 46) denoted $(\mathbb{S}_i F_i : m_i)_{i\in\mathbb{N}}$, which ensures that each choice of statement will eventually be tested in the future and an oracle $\mathfrak{P}$ containing any finite consistent CSS (Lemma 51), which ensures that the choices made preserve consistency. We will see in Proposition 54 that this combination ensures saturation/maximality by all possible consistent choices. However, unlike the case of classical or intuitionistic logic, maximally consistent sets are not enough to obtain counter-models in the case of BBI. This is the reason of some tweaks in the coming construction.

Let us proceed in the core of the proof. Recall that the infinite sequence $(c_i)_{i\in\mathbb{N}}$ is a bijective enumeration of the alphabet $L = \{c_0, c_1, \ldots\}$. We start with a finite CSS $(\mathcal{X}_0, \mathcal{C}_0)$ and $n_0 \in \mathbb{N}$ for which we make the three following hypotheses:

0. $\epsilon \lhd \epsilon \in \mathcal{C}_0$;

1. $(\mathcal{X}_0, \mathcal{C}_0)$ has no closed BBI-tableau;

2. the inclusion $\mathsf{A}_{\mathcal{C}_0} \subseteq \{c_0, c_1, \ldots, c_{n_0-1}\}$ holds.

We simultaneously and recursively build two sequences $(\mathcal{X}_i)_{1\leqslant i}$ and $(x_i \lhd y_i)_{1\leqslant i}$. Denoting $\mathcal{C}_i = \mathcal{C}_0 \cup \{x_1 \lhd y_1, \ldots, x_i \lhd y_i\}$, the values $\mathcal{X}_{i+1}$ and $x_{i+1} \lhd y_{i+1}$ are determined recursively as follows:[13]

- if $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \notin \mathfrak{P}$ then we define $\mathcal{X}_{i+1} = \mathcal{X}_i$ and $x_{i+1} \lhd y_{i+1} = \epsilon \lhd \epsilon$;
- if $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$ then we define $\mathcal{X}_{i+1} = \mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\} \cup \mathcal{X}_e$ where $\mathcal{X}_e$ and $x_{i+1} \lhd y_{i+1}$ are computed according to the following table:

$\mathbb{S}_i$	F_i	$\mathcal{X}_e$	$x_{i+1} \lhd y_{i+1}$
$\mathbb{T}$	I	$\emptyset$	$\epsilon \lhd m_i$
$\mathbb{T}$	$A * B$	$\{\mathbb{T}A : a, \mathbb{T}B : b\}$	$ab \lhd m_i$
$\mathbb{F}$	$A \mathrel{-\!\!*} B$	$\{\mathbb{T}A : a, \mathbb{F}B : b\}$	$am_i \lhd b$
otherwise		$\emptyset$	$\epsilon \lhd \epsilon$

with $\begin{cases} a = c_{n_0+2i} \\ b = c_{n_0+2i+1} \end{cases}$

Informally, the CSS $(\mathcal{X}_i, \mathcal{C}_i)$ is augmented with $\mathbb{S}F : m$ only when this choice is proposed by our strategy (i.e. $\mathbb{S}F : m \equiv \mathbb{S}_i F_i : m_i$) and only when this choice is consistent according to our oracle $\mathfrak{P}$ (i.e. $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$). Moreover (and this is a tweak specific to BBI), in the cases of the shapes $\mathbb{T}\mathsf{I}$, $\mathbb{T}*$ and $\mathbb{F}\mathrel{-\!\!*}$, we force further consistent constraints expansion with a controlled choice for a and b ensuring freshness.

Proposition 52. For any $i \in \mathbb{N}$, the following properties hold:

0. $\epsilon \lhd \epsilon \in \mathcal{C}_i$;

1. $(\mathcal{X}_i, \mathcal{C}_i) \in \mathfrak{P}$;

2. $\mathsf{A}_{\mathcal{C}_i} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i-1}\}$;

3. $\mathcal{X}_i \subseteq \mathcal{X}_{i+1}$ and $\mathcal{C}_i \subseteq \mathcal{C}_{i+1}$;

4. the constraint $x_{i+1} \lhd y_{i+1}$ is basic w.r.t. $\sim_{\mathcal{C}_i}$.

[13] We point out that to compute the values of $\mathcal{X}_{i+1}$ and $x_{i+1} \lhd y_{i+1}$, we need to decide whether $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i)$ belongs to $\mathfrak{P}$ or not. Although this point is painless in a classical setting like the classical set theory we use here, this point is problematic in an intuitionistic setting, especially when $\mathfrak{P}$ is not computably decidable (which is precisely the case here [4, 23]). To build this sequence within Coq, we have to assume the excluded middle axiom and apply it to $\mathfrak{P}$.

Proof. Since $\epsilon \twoheadrightarrow \epsilon \in \mathcal{C}_0$ holds by Hypothesis 0, Property 0 is obvious from the definition of $\mathcal{C}_i$. We first prove Properties 1 and 2 by recursion on i. For the ground case $i = 0$, Property 1, since $(\mathcal{X}_0, \mathcal{C}_0)$ has no closed tableau, it is consistent and thus we have $(\mathcal{X}_0, \mathcal{C}_0) \in \mathfrak{P}$. Property 2 is Hypothesis 2. For the recursive step $i + 1$, we examine each case for Property 1 and 2:

- if $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \notin \mathfrak{P}$ then $\mathcal{X}_{i+1} = \mathcal{X}_i$ and $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{\epsilon \twoheadrightarrow \epsilon\} = \mathcal{C}_i$. Thus $(\mathcal{X}_{i+1}, \mathcal{C}_{i+1}) = (\mathcal{X}_i, \mathcal{C}_i) \in \mathfrak{P}$ holds. Moreover $\mathcal{A}_{\mathcal{C}_{i+1}} = \mathcal{A}_{\mathcal{C}_i} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i-1}\} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i+1}\}$;
- if $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$ and $\mathbb{S}_i F_i : m_i \equiv \mathbb{T}\mathsf{I} : m_i$ then $\mathcal{X}_{i+1} = \mathcal{X}_i \cup \{\mathbb{T}\mathsf{I} : m_i\}$ and $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{\epsilon \twoheadrightarrow m_i\}$. By saturation of $\mathfrak{P}$ for rule $\langle \mathbb{T}\mathsf{I} \rangle$, from $\mathbb{T}\mathsf{I} : m_i \in \mathcal{X}_{i+1}$ and $(\mathcal{X}_{i+1}, \mathcal{C}_i) \in \mathfrak{P}$ we deduce $(\mathcal{X}_{i+1}, \mathcal{C}_{i+1}) = (\mathcal{X}_{i+1}, \mathcal{C}_i \cup \{\epsilon \twoheadrightarrow m_i\}) \in \mathfrak{P}$. The elements of $\mathfrak{P}$ are CSS and thus $(\mathcal{X}_i \cup \{\mathbb{T}\mathsf{I} : m_i\}, \mathcal{C}_i)$ is one. So the relation $m_i \sim_{\mathcal{C}_i} m_i$ holds, thus $m_i \in \mathcal{A}^{\star}_{\mathcal{C}_i}$ and we deduce $\mathcal{A}_{\mathcal{C}_{i+1}} = \mathcal{A}_{\mathcal{C}_i} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i+1}\}$;
- if $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$ and $\mathbb{S}_i F_i : m_i \equiv \mathbb{T}A * B : m_i$. We have $\mathcal{X}_{i+1} = \mathcal{X}_i \cup \{\mathbb{T}A * B : m_i, \mathbb{T}A : a, \mathbb{T}B : b\}$ and $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{ab \twoheadrightarrow m_i\}$ with $a = c_{n_0+2i}$ and $b = c_{n_0+2i+1}$. Since $a \neq b \in L \backslash \mathcal{A}_{\mathcal{C}_i}$ and $(\mathcal{X}_i \cup \{\mathbb{T}A * B : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$, by saturation of $\mathfrak{P}$ for rule $\langle \mathbb{T}* \rangle$, we deduce that $(\mathcal{X}_{i+1}, \mathcal{C}_{i+1}) \in \mathfrak{P}$. Also $m_i \in \mathcal{A}^{\star}_{\mathcal{C}_i}$ and hence $\mathcal{A}_{\mathcal{C}_{i+1}} = \mathcal{A}_{\mathcal{C}_i} \cup \{a, b\} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i+1}\}$;
- for the case where $\mathbb{S}_i F_i : m_i \equiv \mathbb{F}A \twoheadrightarrow\!\!* B : m_i$, similar arguments can be developed except using saturation of $\mathfrak{P}$ by rule $\langle \mathbb{F}\text{-}\!* \rangle$;
- in all other cases with $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$ we have $\mathcal{X}_{i+1} = \mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}$ and $\mathcal{C}_{i+1} = \mathcal{C}_i \cup \{\epsilon \twoheadrightarrow \epsilon\} = \mathcal{C}_i$. Hence we easily obtain $(\mathcal{X}_{i+1}, \mathcal{C}_{i+1}) \in \mathfrak{P}$ and $\mathcal{A}_{\mathcal{C}_{i+1}} = \mathcal{A}_{\mathcal{C}_i} \subseteq \{c_0, c_1, \ldots, c_{n_0+2i+1}\}$.

Property 3 is obvious because $\mathcal{X}_{i+1}$ is obtained by extension of $\mathcal{X}_i$, and $\mathcal{C}_i \subseteq \mathcal{C}_{i+1}$ follows from the definition of $\mathcal{C}_i$. For Property 4, it is sufficient to observe that since $a = c_{n_0+2i}$ and $b = c_{n_0+2i+1}$, hence we have $a \neq b \in L \backslash \mathcal{A}_{\mathcal{C}_i}$ by Property 2 (remember that the enumeration of L is bijective). Then constraint $x_{i+1} \twoheadrightarrow y_{i+1}$ is of one of the forms of Definition 10: $ab \twoheadrightarrow m_i$, $am_i \twoheadrightarrow b$, $\epsilon \twoheadrightarrow m_i$ or $\epsilon \twoheadrightarrow \epsilon$. When $x_{i+1} \twoheadrightarrow y_{i+1} \in \{ab \twoheadrightarrow m_i, am_i \twoheadrightarrow b, \epsilon \twoheadrightarrow m_i\}$, we can check that $m_i \sim_{\mathcal{C}_i} m_i$ because $(\mathcal{X}_i \cup \{\mathbb{S}_i F_i : m_i\}, \mathcal{C}_i) \in \mathfrak{P}$ is a CSS. The identity $x_{i+1} \twoheadrightarrow y_{i+1} = \epsilon \twoheadrightarrow \epsilon$ comes as a particular case of Definition 10 item 3 with $m = \epsilon$. Hence the constraint $x_{i+1} \twoheadrightarrow y_{i+1}$ is basic w.r.t. $\sim_{\mathcal{C}_i}$. ■

We now consider the limit $(\mathcal{X}_\infty, \mathcal{C}_\infty)$ of the sequence $(\mathcal{X}_i, \mathcal{C}_i)_{i \in \mathbb{N}}$ defined by:

$$\mathcal{X}_\infty = \bigcup_{i \in \mathbb{N}} \mathcal{X}_i \quad \text{and} \quad \mathcal{C}_\infty = \bigcup_{i \in \mathbb{N}} \mathcal{C}_i = \mathcal{C}_0 \cup \{x_i \twoheadrightarrow y_i \mid 1 \leqslant i\}$$

We point out that we took care of building $\mathcal{C}_\infty$ as the elements of an infinite sequence of basic constraints rather than as the limit of a sequence of basic PMEs, which simplifies the proof of the fact that $\sim_{\mathcal{C}_\infty}$ is a simple PME.

Proposition 53. If $\sim_{\mathcal{C}_0}$ is a basic PME then $\sim_{\mathcal{C}_\infty}$ is a simple PME.

Proof. Let $\mathcal{C}_0 = \{u_1 \twoheadrightarrow v_1, \ldots, u_q \twoheadrightarrow v_q\}$ where $u_1 \twoheadrightarrow v_1, \ldots, u_q \twoheadrightarrow v_q$ is a basic sequence of constraints. Then $u_1 \twoheadrightarrow v_1, \ldots, u_q \twoheadrightarrow v_q, x_1 \twoheadrightarrow y_1, x_2 \twoheadrightarrow y_2, \ldots$ is a simple sequence of constraints and $\mathcal{C}_\infty = \{u_1 \twoheadrightarrow v_1, \ldots, u_q \twoheadrightarrow v_q, x_1 \twoheadrightarrow y_1, x_2 \twoheadrightarrow y_2, \ldots\}$. Thus $\sim_{\mathcal{C}_\infty}$ is simple. ■

Proposition 54 (Maximal consistency). $(\mathcal{X}_\infty, \mathcal{C}_\infty) \in \mathfrak{P}$ and is a maximally consistent CSS, i.e. for any $\mathbb{S}F : m$, if $(\mathcal{X}_\infty \cup \{\mathbb{S}F : m\}, \mathcal{C}_\infty) \in \mathfrak{P}$ then $\mathbb{S}F : m \in \mathcal{X}_\infty$.

Proof. First we show that $(\mathcal{X}_\infty, \mathcal{C}_\infty)$ is a CSS. Let $\mathbb{S}F : m \in \mathcal{X}_\infty$. We have to show that $m \sim_{\mathcal{C}_\infty} m$. There exists i such that $\mathbb{S}F : m \in \mathcal{X}_i$. Since $(\mathcal{X}_i, \mathcal{C}_i) \in \mathfrak{P}$ is a CSS, we have $m \sim_{\mathcal{C}_i} m$. Since $\mathcal{C}_i \subseteq \mathcal{C}_\infty$ we obtain $m \sim_{\mathcal{C}_\infty} m$.

Let us now use the fact that $\mathfrak{P}$ is of finite character to prove $(\mathcal{X}_\infty, \mathcal{C}_\infty) \in \mathfrak{P}$. Let $(\mathcal{X}_f, \mathcal{C}_f) \preccurlyeq_f (\mathcal{X}_\infty, \mathcal{C}_\infty)$ be a finite CSS. We show that $(\mathcal{X}_f, \mathcal{C}_f) \in \mathfrak{P}$. As both $\mathcal{X}_f$ and $\mathcal{C}_f$ are finite sets, there exists a natural number $j \in \mathbb{N}$ such that $\mathcal{X}_f \subseteq \mathcal{X}_j$ and $\mathcal{C}_f \subseteq \mathcal{C}_j$ (remember that the sequences $(\mathcal{X}_i)_{i \in \mathbb{N}}$ and

$(\mathcal{C}_i)_{i\in\mathbb{N}}$ are increasing w.r.t. subset inclusion). Hence, as $(\mathcal{X}_j,\mathcal{C}_j) \in \mathfrak{P}$ by Proposition 52 and $\mathfrak{P}$ is $\preccurlyeq$-downward closed, we deduce $(\mathcal{X}_f,\mathcal{C}_f) \in \mathfrak{P}$. So every finite sub-CSS of $(\mathcal{X}_\infty,\mathcal{C}_\infty)$ belongs to $\mathfrak{P}$. As $\mathfrak{P}$ is of finite character, we deduce $(\mathcal{X}_\infty,\mathcal{C}_\infty) \in \mathfrak{P}$.

Let $\mathbb{S}F : m$ be such that $(\mathcal{X}_\infty \cup \{\mathbb{S}F : m\},\mathcal{C}_\infty) \in \mathfrak{P}$. So $(\mathcal{X}_\infty \cup \{\mathbb{S}F : m\},\mathcal{C}_\infty)$ is a CSS and thus $m \sim_{\mathcal{C}_\infty} m$. By compactness, there exists a finite subset $\mathcal{C}_f \subseteq \mathcal{C}_\infty$ such that $m \sim_{\mathcal{C}_f} m$. Then, as $\mathcal{C}_f$ is finite, there exists $j \in \mathbb{N}$ such that $\mathcal{C}_f \subseteq \mathcal{C}_j$. By fairness, as $\mathbb{S}F : m$ occurs infinitely many times in the sequence $(\mathbb{S}_iF_i : m_i)_{i\in\mathbb{N}}$, there exists $l \geqslant j$ such that $\mathbb{S}_lF_l : m_l \equiv \mathbb{S}F : m$. As $l \geqslant j$, we have $\mathcal{C}_l \supseteq \mathcal{C}_j \supseteq \mathcal{C}_f$ and thus $m \sim_{\mathcal{C}_l} m$. Then $(\mathcal{X}_l \cup \{\mathbb{S}F : m\},\mathcal{C}_l)$ is a CSS and $(\mathcal{X}_l \cup \{\mathbb{S}F : m\},\mathcal{C}_l) \preccurlyeq (\mathcal{X}_\infty \cup \{\mathbb{S}F : m\},\mathcal{C}_\infty)$. Then, as $\mathfrak{P}$ is $\preccurlyeq$-downward closed, we deduce that $(\mathcal{X}_l \cup \{\mathbb{S}_lF_l : m_l\},\mathcal{C}_l) \in \mathfrak{P}$ holds. Hence, by definition of $\mathcal{X}_{l+1}$, we have $\mathbb{S}F : m \equiv \mathbb{S}_lF_l : m_l \in \mathcal{X}_{l+1}$ and thus $\mathbb{S}F : m \in \mathcal{X}_\infty$. ■

Lemma 55 (Hintikka CSS). The limit CSS $(\mathcal{X}_\infty,\mathcal{C}_\infty)$ is a Hintikka CSS s.t. $(\mathcal{X}_0,\mathcal{C}_0) \preccurlyeq (\mathcal{X}_\infty,\mathcal{C}_\infty)$.

Proof. By definition of $(\mathcal{X}_\infty,\mathcal{C}_\infty)$, we obviously have $(\mathcal{X}_0,\mathcal{C}_0) \preccurlyeq (\mathcal{X}_\infty,\mathcal{C}_\infty)$. By Proposition 54, we have $(\mathcal{X}_\infty,\mathcal{C}_\infty) \in \mathfrak{P}$ where $\mathfrak{P}$ is an oracle. We consider the different conditions of Definition 42:

1. Condition 1 holds because $\mathfrak{P}$ only contains open CSS, hence $(\mathcal{X}_\infty,\mathcal{C}_\infty)$ is an open CSS;
2. if $\mathbb{T}\neg A : m \in \mathcal{X}_\infty$ then $(\mathcal{X} \cup \{\mathbb{F}A : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ because $\mathfrak{P}$ is saturated by rule $\langle\mathbb{T}\neg\rangle$. Hence by Proposition 54, we obtain $\mathbb{F}A : m \in \mathcal{X}_\infty$;
3. similar to Condition 2 but with rule $\langle\mathbb{F}\neg\rangle$;
4. if $\mathbb{T}A \wedge B : m \in \mathcal{X}_\infty$ then $(\mathcal{X}_\infty \cup \{\mathbb{T}A : m, \mathbb{T}B : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ because $\mathfrak{P}$ is saturated by rule $\langle\mathbb{T}\wedge\rangle$. Since $\mathfrak{P}$ is $\preccurlyeq$-downward closed, both $(\mathcal{X}_\infty \cup \{\mathbb{T}A : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ and $(\mathcal{X}_\infty \cup \{\mathbb{T}B : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ hold. By Proposition 54, we obtain $\mathbb{T}A : m \in \mathcal{X}_\infty$ and $\mathbb{T}B : m \in \mathcal{X}_\infty$;
5. if $\mathbb{F}A \wedge B : m \in \mathcal{X}_\infty$ then either $(\mathcal{X}_\infty \cup \{\mathbb{F}A : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ or $(\mathcal{X}_\infty \cup \{\mathbb{F}B : m\},\mathcal{C}_\infty) \in \mathfrak{P}$ because $\mathfrak{P}$ is saturated by rule $\langle\mathbb{F}\wedge\rangle$. By Proposition 54, either $\mathbb{F}A : m \in \mathcal{X}_\infty$ or $\mathbb{F}B : m \in \mathcal{X}_\infty$;
6. if $\mathbb{T}\mathsf{I} : m \in \mathcal{X}_\infty$. By definition of $\mathcal{X}_\infty$, there exists $i \in \mathbb{N}$ such that $\mathbb{T}\mathsf{I} : m \in \mathcal{X}_i$. By fairness, there exists $j \geqslant i$ such that $\mathbb{T}\mathsf{I} : m \equiv \mathbb{S}_jF_j : m_j$. As $j \geqslant i$ and since the sequence $(\mathcal{X}_i)_{i\in\mathbb{N}}$ is increasing, we obtain $\mathbb{T}\mathsf{I} : m \in \mathcal{X}_j$. Then $\epsilon \twoheadleftarrow m_j \in \mathcal{C}_{j+1}$ by definition of $\mathcal{C}_{j+1}$. As $m = m_j$ and $\mathcal{C}_{j+1} \subseteq \mathcal{C}_\infty$, we obtain $\epsilon \twoheadleftarrow m \in \mathcal{C}_{j+1} \subseteq \mathcal{C}_\infty$ and thus $\epsilon \sim_{\mathcal{C}_\infty} m$ holds;
7. Condition 7 holds because $(\mathcal{X}_\infty,\mathcal{C}_\infty) \in \mathfrak{P}$ is an open CSS;
8. suppose $\mathbb{T}A * B : m \in \mathcal{X}_\infty$. By definition of $\mathcal{X}_\infty$, there exists i such that $\mathbb{T}A * B : m \in \mathcal{X}_i$. By fairness, there exists $j \geqslant i$ such that $\mathbb{T}A * B : m \equiv \mathbb{S}_jF_j : m_j$. From $i \leqslant j$, we deduce $\mathbb{T}A * B : m \in \mathcal{X}_j$. Let $x = c_{n_0+2j}$ and $y = c_{n_0+2j+1}$. Then $x, y \in L \subset L^\star$, and by definition of $\mathcal{X}_{j+1}$ and $x_{j+1} \twoheadleftarrow y_{j+1}$ we have $\mathbb{T}A : x \in \mathcal{X}_{j+1}$, $\mathbb{T}B : y \in \mathcal{X}_{j+1}$ and $xy \twoheadleftarrow m \in \mathcal{C}_{j+1}$. We conclude that both $xy \sim_{\mathcal{C}_\infty} m$ and $\{\mathbb{T}A : x, \mathbb{T}B : y\} \subseteq \mathcal{X}_\infty$ hold;
9. suppose $\mathbb{F}A * B : m \in \mathcal{X}_\infty$. Let x, y such that $xy \sim_{\mathcal{C}_\infty} m$. Since $\mathfrak{P}$ is saturated by rule $\langle\mathbb{F}*\rangle$, either $(\mathcal{X}_\infty \cup \{\mathbb{F}A : x\},\mathcal{C}_\infty) \in \mathfrak{P}$ or $(\mathcal{X}_\infty \cup \{\mathbb{F}B : y\},\mathcal{C}_\infty) \in \mathfrak{P}$. Hence by Proposition 54, either $\mathbb{F}A : x \in \mathcal{X}_\infty$ or $\mathbb{F}B : y \in \mathcal{X}_\infty$;
10. similar to Condition 9 but with rule $\langle\mathbb{T}{-\!*}\rangle$;
11. similar to Condition 8.

Hence, we have checked all the conditions of Definition 42. ■

6.4 Strong completeness of the BBI-tableau system

We finish with the strong completeness theorem that states that whenever a formula has no closed BBI-tableau, then we can build a counter-model based on a simple PME.

Theorem 56 (Strong completeness of the BBI-tableau system). Let $a \in L$ be a letter and $G \in \mathsf{Form}$ be a BBI-formula. If $(\{\mathbb{F}G : a\}, \{a \multimap a\})$ has no closed BBI-tableau then G has a counter-model of the form $(L, \sim, \Vdash, a)$ where $\sim$ is a simple PME over L.

Proof. Since $L = \{c_0, c_1, c_2, \ldots\}$ is enumerated by the bijective sequence $(c_i)_{i \in \mathbb{N}}$, let us first find $i \in \mathbb{N}$ such that $a = c_i$. Let us define $b = c_{i+1}$. Then we have $a \neq b \in L$. Since $(\{\mathbb{F}G : a\}, \{a \multimap a\})$ has no closed tableau, from Corollary 32, we know that $(\{\mathbb{F}G : a\}, \{\epsilon \multimap \epsilon, a \multimap b\})$ has no closed tableau either. Let us define $n_0 = i + 2$, $\mathcal{X}_0 = \{\mathbb{F}G : a\}$ and $\mathcal{C}_0 = \{\epsilon \multimap \epsilon, a \multimap b\}$. Since the sequence of constraints $\epsilon \multimap \epsilon, a \multimap b$ is basic then $\sim_{\mathcal{C}_0}$ is basic PME. The values $(\mathcal{X}_0, \mathcal{C}_0)$ satisfy the three hypotheses required in Section 6.3. Thus we obtain the limit Hintikka CSS $(\mathcal{X}_\infty, \mathcal{C}_\infty)$ of Proposition 53 and Lemma 55 which satisfies $\mathbb{F}G : a \in \mathcal{X}_0 \subseteq \mathcal{X}_\infty$ and $\mathcal{C}_0 \subseteq \mathcal{C}_\infty$. From the inclusion $\mathbb{F}G : a \in \mathcal{X}_\infty$, we deduce that the tuple $(L, \sim_{\mathcal{C}_\infty}, \Vdash, a)$ is a counter-model of G by Corollary 44. By Proposition 53, since $\sim_{\mathcal{C}_0}$ is a basic PME then $\sim_{\mathcal{C}_\infty}$ is a simple PME. ■

Corollary 57 (Soundness/completeness of the BBI-tableau system). Let $a \in L$ be a letter in an infinite alphabet L. A BBI-formula G is universally valid in the class of PMEs if and only if the CSS $(\{\mathbb{F}G : a\}, \{a \multimap a\})$ has a closed BBI-tableau over L.

Proof. Soundness is given by Theorem 41. Now let G be a BBI-formula such that $(\{\mathbb{F}G : a\}, \{a \multimap a\})$ has no closed BBI-tableau. Then by Theorem 56, G has a counter-model and is thus invalid. ■

The completeness proof we provide here is to our knowledge, the first published proof of the completeness of the labeled tableau method for partial monoidal Boolean BI. We point out that this proof can easily be adapted to intuitionistic BI, substituting partial monoidal equivalences (PME) with partial monoidal orders (PMO) [22] for the labeled semantics of intuitionistic BI; we did precisely this at the level of the formal Coq proof, see Section 7. However, Daniel Méry's thesis [25] already contains a proof of the completeness of the labeled tableaux method for intuitionistic BI. It should be noted that this later proof depends on the fact that the models generated during proof-search are finite. In particular, the saturation technique employed there relies on the possibility to finitely enumerate at each step the statements which should be added to the list of statements to be explored in the future. In the case of partial monoidal BBI, the models which are generated by proof-search (basic PMEs) can be infinite, even after a finite number of proof-search steps, because of the extensions of the form $\sim + \{\epsilon \multimap m\}$ which introduce invertible words, see [21]. That is why we could not simply adapt the proof of [25] and we choose to restart with ideas from [12].

7 Some remarks about the formal Coq proof

The Coq code corresponding to the proofs developed in this paper is distributed under a free software license at the following web address: `http://www.loria.fr/~larchey/BBI`.[14] The practical instructions to type-check/compile the code are fully described there. It was our intent that the informal proof and the formal proof share the same plan and the same concepts. As a witness of this closeness, we provide a map between the definitions and propositions of this paper with their corresponding Coq identifiers in Table 3. From now on, we will essentially focus on what we consider to be the most notable differences between the informal proof and the formal proof.

To summarise, the main differences between the two proofs are the consequence of the divergence between the meta-level logics used in each case. We develop the informal proofs of this paper in Classical Set Theory whereas the formal Coq proofs are implemented in a variant of Intuitionistic Type Theory called the Calculus of Inductive Constructions [9, 28], with a "minimised" (but we think unavoidable) use of the axiom of excluded middle.

[14] We also point out the formal proof we have developed for (partial monoidal) intuitionistic BI accessible at `http://www.loria.fr/~larchey/BI`. Both proofs share the same plan but some semantic concepts slightly differ of course.

Numbering of item	Coq file name	Coq identifier
Definition 1	`constraints.v`	constraint
Definition 2	`constraints.v`	language alphabet
Table 1	`pme.v`	c_sys_PME
Definition 3	`pme.v`	is_pme
Proposition 4	`pme.v`	pme_lft pme_rt pme_el pme_er
Proposition 5	`pme.v`	language_pme pme_alphabet
Definition 6	`pme.v`	pme
Proposition 7	`pme.v`	pme_cpct
Proposition 8	`pme.v`	alphabet_pme
Definition 9	`predicate.v pme.v`	cup pme
Definition 10	`bbi_basic.v`	bbi_basic
Definition 11	`bbi_simple.v`	simple_pme
Definition 12	`words.v pme.v`	subst_w subst_o subst_cst subst_dimage
Theorem 13	`pme.v`	subst_dimage_pme_inc
Corollary 14	`pme.v`	subst_pme_prop
Definition 15	`kripke.v`	bbi_frame
Proposition 16	`kripke.v`	forces_monotonic
Definition 17	`kripke.v` `bbi_sound_and_complete.v`	mdl_validity bbi_frame bbi_validity
Definition 19	`css.v`	stm CSS CSS_leq CSS_leqf
Proposition 20	`css.v`	CSS_compact
Table 2	`bbi_expansion.v`	bbi_exp_rules bbi_expansion
Definition 21	`tableaux.v bbi_tableaux.v`	tableau bbi_tab
Proposition 22	`bbi_tableaux.v`	bbi_tab_finite_CSS
Proposition 23	`bbi_tableaux.v`	bbi_tab_increase
Proposition 24	`tableaux.v`	tableau_comp
Definition 25	`bbi_closed_branch.v`	bbi_closed_branch
Proposition 26	`bbi_closed_branch.v`	bbi_cb_inc
Proposition 27	`bbi_closed_branch.v`	bbi_cb_finite
Definition 28	`css.v`	subst_CSS
Proposition 29	`css.v` `bbi_closed_branch.v`	CSS_subst_stable CSS_subst_finite CSS_subst_inc bbi_cb_subst
Theorem 30	`bbi_tableaux.v`	bbi_proof_subst
Theorem 31	`bbi_tableaux.v`	bbi_has_closed_tableau_pme_monotonic
Corollary 32	`bbi_tableaux.v`	bbi_has_proof_aa bbi_has_proof_ab
Definition 35	`kripke.v`	mdl_stm mdl_cst mdl_CSS
Proposition 36	`kripke.v`	mdl_cst_meq
Proposition 37	`kripke.v`	meq_mdl_eq
Definition 38	`bbi_realizability.v`	realizable
Proposition 39	`bbi_realizability.v`	realizable_not_closed
Lemma 40	`bbi_realizability.v`	realizable_stable
Theorem 41	`bbi_sound_and_complete.v`	bbi_tab_soundness
Definition 42	`bbi_hintikka.v`	Hintikka
Lemma 43	`bbi_hintikka.v`	Hintikka_model
Corollary 44	`bbi_sound_and_complete.v`	Hintikka_counter_model
Definition 45	`maps.v bbi_strategy.v`	i_surjective fair_strategy
Proposition 46	`bbi_strategy.v`	fair_strategy_exists
Definition 47	`bbi_oracle.v`	oracle_* oracle
Definition 48	`bbi_oracle.v`	CSS_consistent CSS_fconsistent
Proposition 49	`bbi_oracle.v`	CSS_consistent_closed
Corollary 50	`bbi_oracle.v`	finite_consistency_prop
Lemma 51	`bbi_oracle.v`	oracle_exists
Proposition 52	`bbi_limit.v`	C_prop0 XC_prop1 C_prop2 X_prop3 C_prop3 ce_basic
Proposition 53	`bbi_limit.v`	lC_simple
Proposition 54	`bbi_limit.v`	lXC_max
Lemma 55	`bbi_limit.v`	lXC_Hintikka
Theorem 56	`bbi_sound_and_complete.v`	bbi_tab_strong_completeness

Table 3: Correspondence between the items in this paper and the files/identifiers in the Coq code.

Hence, for instance, there is no primitive notion of sets/subsets within Coq. We represent sets like the alphabet L or the set of logical formulæ Form by types. We represent subsets like $\mathcal{A}_C$, $\mathcal{X}$, $\mathcal{C}$ or $\sim_C$ by unary predicates.[15] The use we make of sets/subsets here is compatible with this natural choice; for instance, we do not need to combine subsets of sets of different types. Of course, we loose *extensionality*: two logically equivalent predicates are not necessarily provably equal in Coq. But we can prove that every proposition on the predicates we use "commutes" with extensionality, i.e. substitution of a predicate by another equivalent predicate does not alter the validity of a proposition. We could have assumed predicate extensionality as an added axiom but it is not necessary to do so. The downside is that we have to systematically prove that the functions or propositions we define commute with predicate extensionality.

Another point worth mentioning is the use of classical reasoning, i.e. the axiom of excluded-middle. The calculus of inductive constructions is an intuitionistic logic at its core whereas set theory is generally assumed classical. Indeed, classical reasoning occurs in the informal proofs of both the soundness and the completeness of BBI. We think that due to the undecidability of BBI [4, 23], it is not possible to give a constructive proof of soundness/completeness of BBI, but this might depend on how the results are expressed. For instance, if you express completeness by "every BBI-formula has either a closed BBI-tableau or a (simple) counter-model," then meta-level normalisation combined with a purely intuitionistic Coq proof (no added axiom) would provide a decision algorithm for BBI, something that cannot exist. But if you express completeness by "a BBI-formula is semantically valid if and only if it has a closed BBI-tableau," then the impossibility of a constructive proof is not obvious anymore. These two formulations of completeness are classically equivalent but are not necessarily intuitionistically equivalent for an undecidable logic like BBI.

As a consequence, we were forced to use the axiom of excluded middle for our formal proofs of soundness and completeness of BBI. Anyway, for what it is worth, we tried to minimise the use of this axiom and in fact, very few Coq files depend on it: `bbi_oracle.v`, `bbi_realizability.v` and `bbi_sound_and_complete.v`.

For the axiom of choice which is usually assumed in classical set theory, we did not use it at all, both in the informal and formal proof. The simple counter-model of Section 6.3 is built using a deterministic process so we did not need to use König's lemma or Zorn's lemma. The price to pay to avoid the use of the axiom of choice is to effectively enumerate some sets and provide some effective witnesses of unboundedness for infinite subsets. With this little overhead, we can certify that the axiom of choice can be avoided completely as witnessed in the formal Coq proof.

We have a remark concerning our implementation of words, constraints and PMEs. In the informal proof, words are considered as unordered lists (or multisets) of letters, implying the commutativity of composition. We feel that multisets are a sufficiently widely used and understood notion to take them for granted. In Coq, though not impossible, the use of multisets can be complicated by the fact that it is not an inductive data-type: it is a quotient type. Quotients are not generally available in the calculus of constructions. Assuming them as an axiom could even lead to logical contradictions [8]. Instead of trying to build a notion of quotient general enough for the simple purpose of building words/multisets on an arbitrary alphabet L, we choose to keep words as (ordered) lists of letters, which is of course an inductive data-type. To recover the needed commutativity on the side of the models of BBI, we simply add the following $\langle comm \rangle$ rule to the rules for PMEs of Table 1:

$$\frac{xy \multimap xy}{xy \multimap yx}\ \langle comm \rangle$$

This rule ensures that letters can be permuted in words and constraints provided we work inside a PME $\sim$, i.e. we obtain a congruence result such as: let x and x' (resp. y and y') be two lists of letters equivalent up to permutation of letters; if $x \sim y$ holds then $x' \sim y'$ holds.

We finish our remarks with the two peripheral results that were not formalised in Coq; they are not assumed, they are just ignored as useless to obtain the formalised proof of strong completeness:

[15] In the case of binary relations, we use unary predicates over a type of pairs, similarly to what is done in set theory.

- on one hand, the equivalence of partial monoidal Kripke semantics and PME Kripke semantics of BBI expressed in Fact 18. We think a formalisation will not really be difficult but would probably involve the computation of a quotient partial monoid. The implementation of this result is currently under way and will be made available in the near future;
- on the other hand, the equivalence between the TBBI-tableau system defined in [22] and the BBI-tableau system defined in the current paper, as expressed in Fact 34. We view this result as easy to formalise but the main annoyance will certainly derive from the following observation: we deal with two notions of tableaux that differ only slightly in their respective implementations and this implies duplicating Coq code as well Coq identifiers.

8 Conclusion and perspectives

In this paper, we provide a detailed standalone proof of the strong completeness of a labeled tableaux system for partial monoidal Boolean BI. We give an account of our full formalisation in Coq of this informal proof. We have already adapted our formal proof to intuitionistic BI. We think that our framework, either informal or formal, is general enough to be adapted to various extensions of either BI or BBI, like for instance in the case of Dynamic BI [10].

This strong completeness result implies that it is possible to analyse the semantic properties of BBI through the study of the particular constraints generated by tableau proof-search, i.e. basic and simple PMEs. In a recent study, we show that simple PMEs are cancellative [21], i.e. they are closed under the following rule:

$$\frac{kx \mathrel{-\!\!\bullet\!\!-} ky}{x \mathrel{-\!\!\bullet\!\!-} y}\,\langle cancel\rangle$$

meaning that the corresponding quotient partial monoid is cancellative. As the extension $\epsilon \mathrel{-\!\!\bullet\!\!-} m$ does not always preserve cancellativity, there can be no direct inductive proof of the cancellativity of simple PMEs. On the contrary, the arguments needed to establish that result are rather involved, justifying this independent development [21].

As a consequence of the cancellativity of simple PMEs, we obtain a proof of completeness of BBI-tableaux w.r.t. the class of separation algebras which are partial cancellative commutative monoids [21]. BBI-tableaux are thus a sound and complete semi-decision method for Abstract Separation Logic [7, 5].

Solving basic PMEs, i.e. computing a representation of $\sim_{\mathcal{C}}$ from a representation of a basic sequence of constraints $\mathcal{C}$, is of fundamental importance in the semi-automated process of tableau construction and is thus a strong motivation to study the specific properties of these basic PMEs.

We aim to show how to solve the semantic constraints generated during proof-search and thus be able to decide whether a branch can be expanded and in such a case how it can be expanded by computing the predicate $\mathrm{cond}(\cdot,\cdot)$: given a branch $(\mathcal{X},\mathcal{C})$ where $\sim_{\mathcal{C}}$ is a basic PME, regarding the expansion of that branch, we need to evaluate relations like $m \sim_{\mathcal{C}} n$ or $\epsilon \sim_{\mathcal{C}} m$ to determine if the branch is closed or not, and to compute values x, y such that $xy \sim_{\mathcal{C}} m$ or $xm \sim_{\mathcal{C}} y$ to determine the fireable instances of the rules of the tableau system. We also need to introduce new letters $a \neq b \in L \backslash \mathcal{A}_{\mathcal{C}}$ but this does not require a computation of $\sim_{\mathcal{C}}$, only a (much simpler) computation of $\mathcal{A}_{\mathcal{C}}$.

Then, we reasonably hope to design an effective method for deciding the constraints that are generated during proof-search, method which could then be used in proof-assistants for example. The design of such tools that help at proving general BBI-formulæ has been a long-term goal in the field of verification of properties specified in separation logic and is beginning to emerge with for instance a prover like BBeye [27] based on a nested sequent calculus or the provers of Hóu *et al* [18, 17] based on labeled sequent calculi.

References

[1] James Brotherston. A Unified Display Proof Theory for Bunched Logic. *Electr. Notes Theor. Comput. Sci.*, 265:197–211, 2010.

[2] James Brotherston and Cristiano Calcagno. Classical BI: a logic for reasoning about dualising resources. In Zhong Shao and Benjamin C. Pierce, editors, *POPL*, pages 328–339. ACM, 2009.

[3] James Brotherston and Cristiano Calcagno. Classical BI: Its Semantics and Proof Theory. *Logical Methods in Computer Science*, 6(3), 2010.

[4] James Brotherston and Max I. Kanovich. Undecidability of Propositional Separation Logic and Its Neighbours. In *25th Annual IEEE Symposium on Logic in Computer Science*, pages 130–139. IEEE Computer Society, 2010.

[5] James Brotherston and Jules Villard. Parametric Completeness for Separation Theories. Technical Report RN/13/11, University College London, 2013. Accepted to POPL 2014, available at `http://www0.cs.ucl.ac.uk/staff/J.Brotherston/submitted/hybrid_BBI/hybrid_BBI.pdf`.

[6] Cristiano Calcagno, Luca Cardelli, and Andrew D. Gordon. Deciding validity in a spatial logic for trees. *Journal of Functional Programming*, 15(4):543–572, 2005.

[7] Cristiano Calcagno, Peter W. O'Hearn, and Hongseok Yang. Local Action and Abstract Separation Logic. In *22nd IEEE Symposium on Logic in Computer Science*, pages 366–378. IEEE Computer Society, 2007.

[8] Laurent Chicli, Loïc Pottier, and Carlos Simpson. Mathematical Quotients and Quotient Types in Coq. In Herman Geuvers and Freek Wiedijk, editors, *Types for Proofs and Programs*, volume 2646 of *Lecture Notes in Computer Science*, pages 95–107. Springer Berlin Heidelberg, 2003.

[9] Thierry Coquand and Gérard P. Huet. Constructions: A Higher Order Proof System for Mechanizing Mathematics. In Bruno Buchberger, editor, *European Conference on Computer Algebra (1)*, volume 203 of *Lecture Notes in Computer Science*, pages 151–184. Springer, 1985.

[10] Jean-René Courtault and Didier Galmiche. A Modal BI Logic for Dynamic Resource Properties. In Sergei Artemov and Anil Nerode, editors, *Symposium on Logical Foundations of Computer Science*, volume 7734 of *Lecture Notes in Computer Science*, pages 134–148. Springer-Verlag Berlin Heidelberg, 2013.

[11] Roy Dyckhoff and Sara Negri. Decision methods for linearly ordered Heyting algebras. *Archive for Mathematical Logic*, 45:411–422, 2006.

[12] Melvin Fitting. *First-order logic and automated theorem proving.* Springer-Verlag New York, Inc., New York, NY, USA, 1990.

[13] Didier Galmiche and Dominique Larchey-Wendling. Expressivity properties of Boolean BI through Relational Models. In S. Arun-Kumar and Naveen Garg, editors, *26th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4337 of *Lecture Notes in Computer Science*, pages 358–369. Springer, 2006.

[14] Didier Galmiche, Daniel Méry, and David J. Pym. The semantics of BI and resource tableaux. *Mathematical Structures in Computer Science*, 15(6):1033–1088, 2005.

[15] Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, 50:1–102, 1987.

[16] Jaakko Hintikka. Form and Content in Quantification Theory. *Acta Philosophica Fennica*, 8:7–55, 1955.

[17] Zhé Hóu, Ranald Clouston, Rajeev Goré, and Alwen Tiu. Proof search for propositional abstract separation logics via labelled sequents. *CoRR*, abs/1307.5592, 2013. Accepted to POPL 2014, available at `http://arxiv.org/abs/1307.5592`.

[18] Zhé Hóu, Alwen Tiu, and Rajeev Goré. A Labelled Sequent Calculus for BBI: Proof Theory and Proof Search. In Didier Galmiche and Dominique Larchey-Wendling, editors, *TABLEAUX*, volume 8123 of *Lecture Notes in Computer Science*, pages 172–187. Springer, 2013.

[19] Samin S. Ishtiaq and Peter W. O'Hearn. BI as an Assertion Language for Mutable Data Structures. In *28th ACM Symposium on Principles of Programming Languages, POPL 2001*, pages 14–26, London, UK, 2001.

[20] Dominique Larchey-Wendling. An Alternative Direct Simulation of Minsky Machines into Classical Bunched Logics via Group Semantics. *Electronic Notes in Theoretical Computer Science*, 265:369–387, 2010.

[21] Dominique Larchey-Wendling and Didier Galmiche. Completeness results for Abstract Separation Logics. Submitted, Oct. 2013, available at `http://www.loria.fr/~larchey/papers/larchey_galmiche_oct2013.pdf`.

[22] Dominique Larchey-Wendling and Didier Galmiche. Exploring the relation between Intuitionistic BI and Boolean BI: an unexpected embedding. *Mathematical Structures in Computer Science*, 19(3):435–500, 2009.

[23] Dominique Larchey-Wendling and Didier Galmiche. The Undecidability of Boolean BI through Phase Semantics. In *25th Annual IEEE Symposium on Logic in Computer Science*, pages 140–149. IEEE Computer Society, 2010.

[24] Dominique Larchey-Wendling and Didier Galmiche. Non-deterministic Phase Semantics and the Undecidability of Boolean BI. *ACM Transactions on Computational Logic*, 14(1), 2013.

[25] Daniel Méry. *Preuves et Sémantiques dans des Logiques de Ressources.* PhD thesis, Université Henri Poincaré, Nancy I, 2004. Available at `http://www.loria.fr/~dmery/these.pdf`.

[26] Peter W. O'Hearn, John C. Reynolds, and Hongseok Yang. Local Reasoning about Programs that Alter Data Structures. In *15th Int. Workshop on Computer Science Logic*, volume 2142 of *Lecture Notes in Computer Science*, pages 1–19, Paris, France, 2001.

[27] Jonghyun Park, Jeongbong Seo, and Sungwoo Park. A theorem prover for Boolean BI. In Roberto Giacobazzi and Radhia Cousot, editors, *POPL*, pages 219–232. ACM, 2013.

[28] Frank Pfenning and Christine Paulin-Mohring. Inductively Defined Types in the Calculus of Constructions. In Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics, 5th International Conference*, volume 442 of *Lecture Notes in Computer Science*, pages 209–228. Springer, 1989.

[29] David J. Pym. On Bunched Predicate Logic. In *14th Annual IEEE Symposium on Logic in Computer Science*, pages 183–192. IEEE Computer Society, 1999.

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

A The monotonicity of closed BBI-tableaux

In this section, we develop the main arguments for the proof of Theorem 31.

Fact 58. Let $(\mathcal{X}, \mathcal{C})$ and $(\mathcal{X}, \mathcal{D})$ be two CSS such that $\overline{\mathcal{C}} \subseteq \overline{\mathcal{D}}$ and $A_{\mathcal{C}} = A_{\mathcal{D}}$ hold. If an instance of a rule of Table 2 is fireable for $(\mathcal{X}, \mathcal{C})$ then the same instance (of the same rule) is fireable for $(\mathcal{X}, \mathcal{D})$.

Proof. The conditions of rules $\langle \mathbb{T}\neg, \mathbb{F}\neg, \mathbb{T}\wedge, \mathbb{F}\wedge, \mathbb{T}\mathrm{I}\rangle$ only depend on $\mathcal{X}$ (not on $\mathcal{C}$, $\mathcal{D}$) so the result is trivial in that case. For rules $\langle \mathbb{T}*, \mathbb{F}{-\!\!*}\rangle$, the part of the condition that depends on $\mathcal{C}$ (resp. $\mathcal{D}$) is $a \neq b \in L \backslash A_{\mathcal{C}}$ (resp. $a \neq b \in L \backslash A_{\mathcal{D}}$). Since $A_{\mathcal{C}} = A_{\mathcal{D}}$ the two conditions are obviously equivalent.[16] For rule $\langle \mathbb{F}*\rangle$, the part of the condition that depends on $\mathcal{C}$ (resp. $\mathcal{D}$) is $xy \sim_{\mathcal{C}} m$ (resp. $xy \sim_{\mathcal{D}} m$). Since $\sim_{\mathcal{C}} \subseteq \sim_{\mathcal{D}}$, if the condition holds for $\mathcal{C}$ then it holds for $\mathcal{D}$. The case of rule $\langle \mathbb{T}{-\!\!*}\rangle$ is similar. ■

Beware that the conjunction of the conditions $\overline{\mathcal{C}} \subseteq \overline{\mathcal{D}}$ and $A_{\mathcal{C}} = A_{\mathcal{D}}$ is strictly weaker than the condition $\overline{\mathcal{C}} = \overline{\mathcal{D}}$. As a counter-example we give, $\mathcal{C} = \{a \rightharpoonup a, b \rightharpoonup b\}$ and $\mathcal{D} = \{a \rightharpoonup b\}$ where a and b are two different letters of L. We obtain $\overline{\mathcal{C}} = \{\epsilon \rightharpoonup \epsilon, a \rightharpoonup a, b \rightharpoonup b\}$, $\overline{\mathcal{D}} = \{\epsilon \rightharpoonup \epsilon, a \rightharpoonup a, b \rightharpoonup b, a \rightharpoonup b, b \rightharpoonup a\}$, $A_{\mathcal{C}} = A_{\mathcal{D}} = \{a, b\}$ and obviously $a \rightharpoonup b \in \overline{\mathcal{D}} \backslash \overline{\mathcal{C}}$ hence $\overline{\mathcal{C}} \neq \overline{\mathcal{D}}$.

Let us denote $\mathcal{T} < \mathcal{T}'$ when $\mathcal{T}$ is of the form $\mathcal{T} = [(\mathcal{X}_1^0, \mathcal{C}_1^0), \ldots, (\mathcal{X}_k^0, \mathcal{C}_k^0)]$, $\mathcal{T}'$ is of the form $\mathcal{T}' = [(\mathcal{X}_1^0, \mathcal{C}_1^1), \ldots, (\mathcal{X}_k^0, \mathcal{C}_k^1)]$ and the relations $\overline{\mathcal{C}_i^0} \subseteq \overline{\mathcal{C}_i^1}$ and $A_{\mathcal{C}_i^0} = A_{\mathcal{C}_i^1}$ hold for any $i \in \{1, \ldots, k\}$.

Lemma 59. Let $(\mathcal{X}, \mathcal{C})$ and $(\mathcal{X}, \mathcal{D})$ be two CSS such that the relations $\overline{\mathcal{C}} \subseteq \overline{\mathcal{D}}$ and $A_{\mathcal{C}} = A_{\mathcal{D}}$ both hold. We can transform any BBI-tableau $\mathcal{T}$ for $(\mathcal{X}, \mathcal{C})$ into a BBI-tableau $\mathcal{T}'$ for $(\mathcal{X}, \mathcal{D})$ such that $\mathcal{T} < \mathcal{T}'$ holds.

Proof. We proceed by induction on the process that builds the tableau $\mathcal{T}$:

- in the ground case, $\mathcal{T}$ is the tableau $[(\mathcal{X}, \mathcal{C})]$. Then $\mathcal{T}' = [(\mathcal{X}, \mathcal{D})]$ is a tableau for $(\mathcal{X}, \mathcal{D})$ with $\mathcal{T} < \mathcal{T}'$;
- otherwise, there exists a tableau $\mathcal{T}^0 = \mathcal{T}_l{}^0 +\!\!+ [(\mathcal{X}_q^0, \mathcal{C}_q^0)] +\!\!+ \mathcal{T}_r{}^0$ for $(\mathcal{X}, \mathcal{C})$, an instance of a rule of Table 2

$$\frac{\mathrm{cond}(\,\cdot\,,\cdot\,)}{(\mathcal{X}_1, \mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k, \mathcal{C}_k)}$$

fireable for $(\mathcal{X}_q^0, \mathcal{C}_q^0)$ with

$$\mathcal{T} = \mathcal{T}_l{}^0 +\!\!+ [(\mathcal{X}_q^0 \cup \mathcal{X}_1, \mathcal{C}_q^0 \cup \mathcal{C}_1); \ldots; (\mathcal{X}_q^0 \cup \mathcal{X}_k, \mathcal{C}_q^0 \cup \mathcal{C}_k)] +\!\!+ \mathcal{T}_r{}^0$$

Since $\mathcal{T}$ was built inductively from $\mathcal{T}^0$, we can apply the induction hypothesis to the tableau $\mathcal{T}^0$ and we can build a tableau $\mathcal{T}^1 = \mathcal{T}_l{}^1 +\!\!+ [(\mathcal{X}_q^0, \mathcal{C}_q^1)] +\!\!+ \mathcal{T}_r{}^1$ for $(\mathcal{X}, \mathcal{D})$ such that $\mathcal{T}_l{}^0 < \mathcal{T}_l{}^1$, $\mathcal{T}_r{}^0 < \mathcal{T}_r{}^1$, $\overline{\mathcal{C}_q^0} \subseteq \overline{\mathcal{C}_q^1}$ and $A_{\mathcal{C}_q^0} = A_{\mathcal{C}_q^1}$. From the Fact 58, we deduce that the same rule instance is fireable for $(\mathcal{X}_q^0, \mathcal{C}_q^1)$. As a consequence

$$\mathcal{T}' = \mathcal{T}_l{}^1 +\!\!+ [(\mathcal{X}_q^0 \cup \mathcal{X}_1, \mathcal{C}_q^1 \cup \mathcal{C}_1); \ldots; (\mathcal{X}_q^0 \cup \mathcal{X}_k, \mathcal{C}_q^1 \cup \mathcal{C}_k)] +\!\!+ \mathcal{T}_r{}^1$$

is a tableau for $(\mathcal{X}, \mathcal{D})$. We show that $\mathcal{T} < \mathcal{T}'$. We already have $\mathcal{T}_l{}^0 < \mathcal{T}_l{}^1$ and $\mathcal{T}_r{}^0 < \mathcal{T}_r{}^1$. From $\overline{\mathcal{C}_p^0} \subseteq \overline{\mathcal{C}_p^1}$ we deduce $\overline{\mathcal{C}_p^0 \cup \mathcal{C}_i} \subseteq \overline{\mathcal{C}_p^1 \cup \mathcal{C}_i}$ and since $A_{\mathcal{C}_p^0} = A_{\mathcal{C}_p^1}$, we derive that $\mathcal{C}_p^0 \cup \mathcal{C}_i$ and $\mathcal{C}_p^1 \cup \mathcal{C}_i$ have the same alphabet. Hence the relation $\mathcal{T} < \mathcal{T}'$ holds.

Hence we proved the result by replaying the process that built the tableau $\mathcal{T}$. ■

[16] Remark that the weaker condition $A_{\mathcal{C}} \subseteq A_{\mathcal{D}}$ *would not be sufficient* because the variables a and b introduced in rules $\langle \mathbb{T}*\rangle$ and $\langle \mathbb{F}{-\!\!*}\rangle$ would not necessarily be new in this case.

Fact 60. Let $(\mathcal{X}, \mathcal{C})$ be a CSS and $\sigma : L \longrightarrow L$ be a substitution of letters such that σ is injective on $L \backslash \mathcal{A}_\mathcal{C}$ and the inclusion $\sigma^{-1}(\sigma(\mathcal{A}_\mathcal{C})) \subseteq \mathcal{A}_\mathcal{C}$ holds. If the rule instance

$$\frac{\text{cond}(\,\cdot\,,\cdot\,)}{(\mathcal{X}_1, \mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k, \mathcal{C}_k)}$$

is fireable for $(\mathcal{X}, \mathcal{C})$ then there exists a(nother) instance of the same rule of the form

$$\frac{\text{cond}'(\,\cdot\,,\cdot\,)}{(\sigma(\mathcal{X}_1), \sigma(\mathcal{C}_1)) \;\mid\; \cdots \;\mid\; (\sigma(\mathcal{X}_k), \sigma(\mathcal{C}_k))}$$

which is fireable for $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$.

Proof. We proceed by case analysis. We treat the case of rule $\langle \mathbb{T}* \rangle$ as an example. So let us consider the following instance

$$\frac{\mathbb{T}A * B : m \in (\cdot) \text{ and } a \neq b \in L\backslash\mathcal{A}_{(\cdot)}}{(\{\mathbb{T}A : a, \mathbb{T}B : b\}, \{ab \multimap m\})} \langle \mathbb{T}* \rangle$$

with parameters A, B, m, a and b.[17] This instance is fireable for $(\mathcal{X}, \mathcal{C})$ if and only if both $\mathbb{T}A * B : m \in \mathcal{X}$ and $a \neq b \in L\backslash\mathcal{A}_\mathcal{C}$ hold. We consider the instance of the same rule with parameters A, B, $\sigma(m)$, $\sigma(a)$ and $\sigma(b)$:

$$\frac{\mathbb{T}A * B : \sigma(m) \in (\cdot) \text{ and } \sigma(a) \neq \sigma(b) \in L\backslash\mathcal{A}_{(\cdot)}}{(\{\mathbb{T}A : \sigma(a), \mathbb{T}B : \sigma(b)\}, \{\sigma(a)\sigma(b) \multimap \sigma(m)\})} \langle \mathbb{T}* \rangle$$

This instance is fireable for $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$ if and only if both $\mathbb{T}A * B : \sigma(m) \in \sigma(\mathcal{X})$ and $\sigma(a) \neq \sigma(b) \in L\backslash\mathcal{A}_{\sigma(\mathcal{C})}$ hold. It is clear that if $\mathbb{T}A * B : m \in \mathcal{X}$ holds then $\mathbb{T}A * B : \sigma(m) \in \sigma(\mathcal{X})$ holds. Since σ is a substitution of letters, we have $\mathcal{A}_{\sigma(\mathcal{C})} = \sigma(\mathcal{A}_\mathcal{C})$. Since σ is injective on $L\backslash\mathcal{A}_\mathcal{C}$, from $a \neq b$ we deduce $\sigma(a) \neq \sigma(b)$. Moreover, since the inclusion $\sigma^{-1}(\sigma(\mathcal{A}_\mathcal{C})) \subseteq \mathcal{A}_\mathcal{C}$ holds, from $a, b \notin \mathcal{A}_\mathcal{C}$ we deduce $\sigma(a), \sigma(b) \notin \sigma(\mathcal{A}_\mathcal{C}) = \mathcal{A}_{\sigma(\mathcal{C})}$. Hence if the first instance of $\langle \mathbb{T}* \rangle$ is fireable for $(\mathcal{X}, \mathcal{C})$ then the second instance of $\langle \mathbb{T}* \rangle$ is fireable for $(\sigma(\mathcal{X}), \sigma(\mathcal{C}))$. ■

Theorem 61. Let $(\mathcal{X}_0, \mathcal{C}_0)$ be a CSS and $\sigma : L \longrightarrow L$ be a substitution of letters such that σ is injective on $L\backslash\mathcal{A}_{\mathcal{C}_0}$ and the inclusion $\sigma^{-1}(\sigma(\mathcal{A}_{\mathcal{C}_0})) \subseteq \mathcal{A}_{\mathcal{C}_0}$ holds. If $\mathcal{T}$ is a BBI-tableau for $(\mathcal{X}_0, \mathcal{C}_0)$ then $\sigma(\mathcal{T})$ is a BBI-tableau for $(\sigma(\mathcal{X}_0), \sigma(\mathcal{C}_0))$.

Proof. We proceed by induction on the process that builds the tableau $\mathcal{T}$ for $(\mathcal{X}_0, \mathcal{C}_0)$:

- in the ground case, $\mathcal{T}$ is the tableau $\mathcal{T} = [(\mathcal{X}_0, \mathcal{C}_0)]$. It is obvious that $\sigma(\mathcal{T}) = [(\sigma(\mathcal{X}_0), \sigma(\mathcal{C}_0))]$ is a tableau for $(\sigma(\mathcal{X}_0), \sigma(\mathcal{C}_0))$;
- otherwise, there exists a tableau $\mathcal{T}_0 = \mathcal{T}_l +\!\!+ [(\mathcal{X}, \mathcal{C})] +\!\!+ \mathcal{T}_r$ for $(\mathcal{X}_0, \mathcal{C}_0)$ and a rule instance

  $$\frac{\text{cond}(\,\cdot\,,\cdot\,)}{(\mathcal{X}_1, \mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k, \mathcal{C}_k)}$$

  fireable for $(\mathcal{X}, \mathcal{C})$ such that $\mathcal{T} = \mathcal{T}_l +\!\!+ [(\mathcal{X} \cup \mathcal{X}_1, \mathcal{C} \cup \mathcal{C}_1); \ldots; (\mathcal{X} \cup \mathcal{X}_k, \mathcal{C} \cup \mathcal{C}_k)] +\!\!+ \mathcal{T}_r$. By induction hypothesis, $\sigma(\mathcal{T}_0)$ is a tableau for $(\sigma(\mathcal{X}_0), \sigma(\mathcal{C}_0))$. We check that σ is injective on $L\backslash\mathcal{A}_\mathcal{C}$ and that the inclusion $\sigma^{-1}(\sigma(\mathcal{A}_\mathcal{C})) \subseteq \mathcal{A}_\mathcal{C}$ holds. We derive both these properties from the inclusion $\mathcal{C}_0 \subseteq \mathcal{C}$

[17] Recall that contrary to A, B, m... the markers $(\cdot)$ for the binders $\mathcal{X}$ and $\mathcal{C}$ are not parameters for the rules.

obtained by Property 23; indeed $(\mathcal{X},\mathcal{C})$ is a branch of $\mathcal{T}_0$ which is a tableau for $(\mathcal{X}_0,\mathcal{C}_0)$. As a consequence of Fact 60, there exists another instance of the same rule

$$\frac{\mathrm{cond}'(\,\cdot\,,\cdot\,)}{(\sigma(\mathcal{X}_1),\sigma(\mathcal{C}_1)) \;\mid\; \cdots \;\mid\; (\sigma(\mathcal{X}_k),\sigma(\mathcal{C}_k))}$$

which is fireable for $(\sigma(\mathcal{X}),\sigma(\mathcal{C}))$. But $\sigma(\mathcal{T}_0)=\sigma(\mathcal{T}_l) +\!\!+ [(\sigma(\mathcal{X}),\sigma(\mathcal{C}))] +\!\!+ \sigma(\mathcal{T}_r)$ is a tableau for $(\sigma(\mathcal{X}_0),\sigma(\mathcal{C}_0))$. So the list

$$\sigma(\mathcal{T}_l) +\!\!+ [(\sigma(\mathcal{X})\cup\sigma(\mathcal{X}_1),\sigma(\mathcal{C})\cup\sigma(\mathcal{C}_1));\ldots;(\sigma(\mathcal{X})\cup\sigma(\mathcal{X}_k),\sigma(\mathcal{C})\cup\sigma(\mathcal{C}_k))] +\!\!+ \sigma(\mathcal{T}_r)$$

is a tableau for $(\sigma(\mathcal{X}_0),\sigma(\mathcal{C}_0))$. This list matches exactly $\sigma(\mathcal{T})$.

Hence we proved the result by replaying the process that builds the tableau replacing each rule instance according to Fact 60. ■

Corollary 62. Let $\sigma : L \longrightarrow L$ be an injective substitution of letters. If $\mathcal{T}$ is a BBI-tableau for $(\mathcal{X},\mathcal{C})$ then $\sigma(\mathcal{T})$ is a BBI-tableau for $(\sigma(\mathcal{X}),\sigma(\mathcal{C}))$.

Proof. We remark that an injective substitution of letters is a fortiori injective on $L\backslash\mathsf{A}_\mathcal{C}$ and satisfies the identity $\sigma^{-1}(\sigma(X))=X$ for any $X\subseteq L$. Hence the inclusion $\sigma^{-1}(\sigma(\mathsf{A}_\mathcal{C}))\subseteq\mathsf{A}_\mathcal{C}$ holds and Theorem 61 can be applied. ■

Fact 63. Let $\mathcal{X}$, $\mathcal{X}_e$, $\mathcal{C}$ and $\mathcal{C}_e$ be such that $(\mathcal{X},\mathcal{C})$ and $(\mathcal{X}\cup\mathcal{X}_e,\mathcal{C}\cup\mathcal{C}_e)$ are two CSS. If an instance of a rule of Table 2

$$\frac{\mathrm{cond}(\,\cdot\,,\cdot\,)}{(\mathcal{X}_1,\mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k,\mathcal{C}_k)}$$

is fireable for $(\mathcal{X},\mathcal{C})$ and satisfies $(\mathsf{A}_{\mathcal{C}_1}\cup\cdots\cup\mathsf{A}_{\mathcal{C}_k})\cap\mathsf{A}_{\mathcal{C}_e}\subseteq\mathsf{A}_\mathcal{C}$ then the same instance (of the same rule) is fireable for $(\mathcal{X}\cup\mathcal{X}_e,\mathcal{C}\cup\mathcal{C}_e)$.

Proof. We proceed by case analysis. We treat the case of rule $\langle\mathbb{T}*\rangle$ as an example. So let us consider the following instance

$$\frac{\mathbb{T}A*B : m\in(\cdot) \text{ and } a\neq b\in L\backslash\mathsf{A}_{(\cdot)}}{(\{\mathbb{T}A:a,\mathbb{T}B:b\},\{ab\rightharpoonup m\})}\,\langle\mathbb{T}*\rangle$$

with parameters A, B, m, a and b, fireable for $(\mathcal{X},\mathcal{C})$, i.e. both $\mathbb{T}A*B : m\in\mathcal{X}$ and $a\neq b\in L\backslash\mathsf{A}_\mathcal{C}$ hold. By hypothesis we have the inclusion $(\{a,b\}\cup\mathsf{A}_m)\cap\mathsf{A}_{\mathcal{C}_e}\subseteq\mathsf{A}_\mathcal{C}$. Thus, from $a,b\notin\mathsf{A}_\mathcal{C}$ we derive $a,b\notin\mathsf{A}_{\mathcal{C}_e}$. Hence we have $a,b\notin\mathsf{A}_{\mathcal{C}\cup\mathcal{C}_e}$. Since it is obvious that $\mathbb{T}A*B : m\in\mathcal{X}\cup\mathcal{X}_e$ holds, we conclude that this instance is also fireable for $(\mathcal{X}\cup\mathcal{X}_e,\mathcal{C}\cup\mathcal{C}_e)$. ■

Let $\mathcal{T}=[(\mathcal{X}_1,\mathcal{C}_1);\ldots;(\mathcal{X}_k,\mathcal{C}_k)]$ be a list of CSS. We write $\mathsf{A}_\mathcal{T}$ for the set of letters occurring in this list, i.e. $\mathsf{A}_\mathcal{T}=\mathsf{A}_{\mathcal{C}_1}\cup\cdots\cup\mathsf{A}_{\mathcal{C}_k}$ and we write $\mathcal{T}\cup(\mathcal{X}_e,\mathcal{C}_e)$ for the result of adding $(\mathcal{X}_e,\mathcal{C}_e)$ to each CSS in the list, i.e.

$$\mathcal{T}\cup(\mathcal{X}_e,\mathcal{C}_e)=[(\mathcal{X}_1\cup\mathcal{X}_e,\mathcal{C}_1\cup\mathcal{C}_e);\ldots;(\mathcal{X}_k\cup\mathcal{X}_e,\mathcal{C}_k\cup\mathcal{C}_e)]$$

Proposition 64. Let $\mathcal{X}_0$, $\mathcal{X}_e$, $\mathcal{C}_0$ and $\mathcal{C}_e$ be such that $(\mathcal{X}_0,\mathcal{C}_0)$ and $(\mathcal{X}_0\cup\mathcal{X}_e,\mathcal{C}_0\cup\mathcal{C}_e)$ are two CSS. If $\mathcal{T}$ is a BBI-tableau for $(\mathcal{X}_0,\mathcal{C}_0)$ which verifies $\mathsf{A}_\mathcal{T}\cap\mathsf{A}_{\mathcal{C}_e}\subseteq\mathsf{A}_{\mathcal{C}_0}$ then $\mathcal{T}\cup(\mathcal{X}_e,\mathcal{C}_e)$ is a BBI-tableau for $(\mathcal{X}_0\cup\mathcal{X}_e,\mathcal{C}_0\cup\mathcal{C}_e)$.

Proof. We proceed by induction on the process that builds the tableau $\mathcal{T}$ for $(\mathcal{X}_0,\mathcal{C}_0)$:

- in the ground case, $\mathcal{T}$ is the tableau $\mathcal{T}=[(\mathcal{X}_0,\mathcal{C}_0)]$. It is obvious that $\mathcal{T}\cup(\mathcal{X}_e,\mathcal{C}_e)=[(\mathcal{X}_0\cup\mathcal{X}_e,\mathcal{C}_0\cup\mathcal{C}_e)]$ is a tableau for $(\mathcal{X}_0\cup\mathcal{X}_e,\mathcal{C}_0\cup\mathcal{C}_e)$;

- otherwise, there exists a tableau $\mathcal{T}_0 = \mathcal{T}_l +\!\!+ [(\mathcal{X}, \mathcal{C})] +\!\!+ \mathcal{T}_r$ for $(\mathcal{X}_0, \mathcal{C}_0)$ and a rule instance

  $$\frac{\text{cond}(\,\cdot\,,\cdot\,)}{(\mathcal{X}_1, \mathcal{C}_1) \;\mid\; \cdots \;\mid\; (\mathcal{X}_k, \mathcal{C}_k)}$$

  fireable for $(\mathcal{X}, \mathcal{C})$ such that $\mathcal{T} = \mathcal{T}_l +\!\!+ [(\mathcal{X} \cup \mathcal{X}_1, \mathcal{C} \cup \mathcal{C}_1); \ldots ; (\mathcal{X} \cup \mathcal{X}_k, \mathcal{C} \cup \mathcal{C}_k)] +\!\!+ \mathcal{T}_r$. We have $\mathsf{A}_{\mathcal{T}_0} \cap \mathsf{A}_{\mathcal{C}_e} \subseteq \mathsf{A}_{\mathcal{T}} \cap \mathsf{A}_{\mathcal{C}_e} \subseteq \mathsf{A}_{\mathcal{C}_0}$. Hence, by induction hypothesis, $\mathcal{T}_0 \cup (\mathcal{X}_e, \mathcal{C}_e) = \mathcal{T}_l \cup (\mathcal{X}_e, \mathcal{C}_e) +\!\!+ [(\mathcal{X} \cup \mathcal{X}_e, \mathcal{C} \cup \mathcal{C}_e)] +\!\!+ \mathcal{T}_r \cup (\mathcal{X}_e, \mathcal{C}_e)$ is a tableau for $(\mathcal{X}_0 \cup \mathcal{X}_e, \mathcal{C}_0 \cup \mathcal{C}_e)$. Moreover, we have $(\mathsf{A}_{\mathcal{C}_1} \cup \cdots \cup \mathsf{A}_{\mathcal{C}_k}) \cap \mathsf{A}_{\mathcal{C}_e} \subseteq \mathsf{A}_{\mathcal{T}} \cap \mathsf{A}_{\mathcal{C}_e} \subseteq \mathsf{A}_{\mathcal{C}_0} \subseteq \mathsf{A}_{\mathcal{C}}$ by Proposition 23. As a consequence of Fact 63, the same rule instance is fireable for $(\mathcal{X} \cup \mathcal{X}_e, \mathcal{C} \cup \mathcal{C}_e)$. Hence the list of CSS

  $$\mathcal{T}_l \cup (\mathcal{X}_e, \mathcal{C}_e) +\!\!+ [(\mathcal{X} \cup \mathcal{X}_e \cup \mathcal{X}_1, \mathcal{C} \cup \mathcal{C}_e \cup \mathcal{C}_1); \ldots ; (\mathcal{X} \cup \mathcal{X}_e \cup \mathcal{X}_k, \mathcal{C} \cup \mathcal{C}_e \cup \mathcal{C}_k)] +\!\!+ \mathcal{T}_r \cup (\mathcal{X}_e, \mathcal{C}_e)$$

  is a tableau for $(\mathcal{X}_0 \cup \mathcal{X}_e, \mathcal{C}_0 \cup \mathcal{C}_e)$. This list matches exactly $\mathcal{T} \cup (\mathcal{X}_e, \mathcal{C}_e)$.

Hence we proved the result by replaying exactly the process that built the tableau $\mathcal{T}$. ■

Lemma 65. Let $(\mathcal{X}_0, \mathcal{C}_0)$ and $(\mathcal{X}_1, \mathcal{C}_1)$ be two finite CSS such that $(\mathcal{X}_0, \mathcal{C}_0) \preccurlyeq (\mathcal{X}_1, \mathcal{C}_1)$. If $(\mathcal{X}_0, \mathcal{C}_0)$ has a closed BBI-tableau then $(\mathcal{X}_1, \mathcal{C}_1)$ has a closed BBI-tableau.

Proof. Let us define $\mathcal{X}_e = \mathcal{X}_1 \backslash \mathcal{X}_0$ and $\mathcal{C}_e = \mathcal{C}_1 \backslash \mathcal{C}_0$. Since $L = \{c_0, c_1, \ldots\}$ and $\mathsf{A}_{\mathcal{C}_1}$ is finite, let us choose n such that $\mathsf{A}_{\mathcal{C}_1} \subseteq \{c_0, \ldots, c_{n-1}\}$. We deduce $\mathsf{A}_{\mathcal{C}_0} \subseteq \{c_0, \ldots, c_{n-1}\}$ and $\mathsf{A}_{\mathcal{C}_e} \subseteq \{c_0, \ldots, c_{n-1}\}$. We define the substitution of letters $\sigma : L \longrightarrow L$ by

$$\sigma(c_i) = \begin{cases} c_i & \text{if } c_i \in \mathsf{A}_{\mathcal{C}_0} \\ c_{i+n} & \text{if } c_i \notin \mathsf{A}_{\mathcal{C}_0} \end{cases}$$

From $\mathsf{A}_{\mathcal{C}_0} \subseteq \{c_0, \ldots, c_{n-1}\}$ and the injectivity of the sequence $(c_i)_{i \in \mathbb{N}}$, we deduce that σ is an injective substitution of letters.[18]

Let $\mathcal{T}$ be a closed tableau for $(\mathcal{X}_0, \mathcal{C}_0)$. By Corollary 62, we deduce that $\sigma(\mathcal{T})$ is a tableau for $(\sigma(\mathcal{X}_0), \sigma(\mathcal{C}_0)) = (\mathcal{X}_0, \mathcal{C}_0)$ (because σ is the identity on $\mathsf{A}_{\mathcal{C}_0}$). By Proposition 29, we know that all the branches of $\sigma(\mathcal{T})$ are closed. Since σ is a substitution of letters we have $\mathsf{A}_{\sigma(\mathcal{T})} = \sigma(\mathsf{A}_{\mathcal{T}}) \subseteq \sigma(L) = \mathsf{A}_{\mathcal{C}_0} \cup \{c_n, c_{n+1}, \ldots\}$. Thus we obtain $\mathsf{A}_{\sigma(\mathcal{T})} \cap \mathsf{A}_{\mathcal{C}_e} \subseteq \mathsf{A}_{\sigma(\mathcal{T})} \cap \{c_0, \ldots, c_{n-1}\} \subseteq \mathsf{A}_{\mathcal{C}_0}$. Thus by Proposition 64, $\sigma(\mathcal{T}) \cup (\mathcal{X}_e, \mathcal{C}_e)$ is a tableau for $(\mathcal{X}_0 \cup \mathcal{X}_e, \mathcal{C}_0 \cup \mathcal{C}_e)$. Moreover, by Proposition 26, since all the branches of $\sigma(\mathcal{T}) \cup (\mathcal{X}_e, \mathcal{C}_e)$ are closed, then we have a closed tableau for $(\mathcal{X}_1, \mathcal{C}_1)$. ■

Theorem 66. Let $(\mathcal{X}_1, \mathcal{C}_1)$ and $(\mathcal{X}_2, \mathcal{C}_2)$ be two finite CSS such that $\mathcal{X}_1 \subseteq \mathcal{X}_2$ and $\mathcal{C}_1 \subseteq \overline{\mathcal{C}_2}$ both hold. If $(\mathcal{X}_1, \mathcal{C}_1)$ has a closed BBI-tableau then $(\mathcal{X}_2, \mathcal{C}_2)$ has a closed BBI-tableau.

Proof. Let us suppose that $(\mathcal{X}_1, \mathcal{C}_1)$ has a closed tableau. Let $\mathcal{D} = \mathcal{C}_1 \cup \mathcal{C}_2$. Then $(\mathcal{X}_2, \mathcal{D})$ is a finite CSS and the inclusion $(\mathcal{X}_1, \mathcal{C}_1) \preccurlyeq (\mathcal{X}_2, \mathcal{D})$ holds. By Lemma 65, we obtain a closed tableau $\mathcal{T}$ for $(\mathcal{X}_2, \mathcal{D})$. It is easy to check that $\overline{\mathcal{D}} \subseteq \overline{\mathcal{C}_2}$ and $\mathsf{A}_{\mathcal{D}} = \mathsf{A}_{\mathcal{C}_2}$ both hold. Then by Lemma 59, we obtain a tableau $\mathcal{T}'$ for $(\mathcal{X}_2, \mathcal{C}_2)$ such that $\mathcal{T} < \mathcal{T}'$. Since all the branch of $\mathcal{T}$ are closed and $\mathcal{T} < \mathcal{T}'$, then by Proposition 26, all the branches of $\mathcal{T}'$ are closed. Hence, $\mathcal{T}'$ is a closed tableau for $(\mathcal{X}_2, \mathcal{C}_2)$. ■

[18] Remark that such an injective substitution does not necessarily exist when the set of letters L is finite.