Automated Proof Search in Intuitionistic Sentential Logic.

Didier Galmiche[1], Brandon Hornbeck[1] and Daniel Méry[1]

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

Abstract

In this paper we describe an automated theorem prover for the intuitionistic non-Fregean sentential calculus with Suszko's identity ISCI. We first review the basic concepts of the logic, recall the recently proposed Topological Beth semantics for ISCI and a corresponding sound and complete labelled calculus. From this calculus we investigate automated proof search for ISCI and present the theorem prover AutoPSI through its architecture, its proof search strategies and optimizations. We complete with tests and benchmarks that illustrate the impact of strategies.

Keywords

Automated Proof Search, Labelled Calculi, Non Fregean Logic, Intuitionistic Logic with Identity

1. Introduction

In this paper we consider the intuitionistic sentential calculus with identity (ISCI) which extends intuitionistic logic with Suszko's identity operator $\approx$ introduced in [1] for non-Fregean logics. In the non-Fregean approach identity and logical equivalence have distinct meanings: two sentences with the same truth value can have different denotations. For example, two logically equivalent formulas might have distinct sets of proofs. Suszko's identity has been studied as an extension of classical logic in [2]. The resulting logic is called SCI. The intuitionistic variant ISCI has been studied in [3] and we have recently proposed a new semantics, called Topological Beth semantics and a new sequent-style labelled calculus $\mathsf{L}_{\mathsf{ISCI}}$ for ISCI in [4]. From these results we study automated proof search in this logic and present the theorem prover AutoPSI through its architecture and its proof search strategies and optimizations. Tests and benchmarks complete this study.

2. Intuitionistic Sentential Calculus with Identity

In this section, we recall the basic notions of ISCI [2, 1]. ISCI extends propositional intuitionistic logic (IL) with axioms that formalize the non-truth functional nature of the identity connective $\approx$.

Definition 1. *Let* $\mathbf{P} = \{ \mathrm{p}, \mathrm{q}, \dots \}$ *be a countable set of propositional letters. The formulas of* ISCI*, the set of which is denoted* $\mathbf{F}$*, are given by the grammar:*

$$\mathrm{A} ::= \mathbf{P} \mid \bot \mid \mathrm{A} \wedge \mathrm{A} \mid \mathrm{A} \vee \mathrm{A} \mid \mathrm{A} \supset \mathrm{A} \mid \mathrm{A} \approx \mathrm{A}$$

Formulas of the form $\mathrm{A} \approx \mathrm{B}$ are called *equations*. We write $\mathbf{F}_{\approx}$ for the restriction of $\mathbf{F}$ to equations. Negation $\neg\mathrm{A}$ and truth $\top$ are respectively defined as $\mathrm{A} \supset \bot$ and $\bot \supset \bot$.

ISCI can be axiomatized by adding the four identity axioms described in Fig. 1 to any axiom schemata for IL [2]. We call "$\mathsf{H}_{\mathsf{ISCI}}$" the Hilbert proof system consisting of the four axioms for identity, the ten axioms for IL and the rule of modus ponens. We write $\mathrm{S} \vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{B}$ to mean that a formula B is derivable in $\mathsf{H}_{\mathsf{ISCI}}$ from a finite set $\mathrm{S} = \{ \mathrm{A}_1, \dots, \mathrm{A}_n \}$ of assumptions. Whenever S is empty, B is called a *thesis* or a *theorem* of $\mathsf{H}_{\mathsf{ISCI}}$. Let us note that the deduction theorem holds for $\mathsf{H}_{\mathsf{ISCI}}$, i.e. $\mathrm{A}_1, \dots, \mathrm{A}_n \vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{B}$ iff $\vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{A}_1 \wedge \dots \wedge \mathrm{A}_n \supset \mathrm{B}$.

ARQNL 2024: Automated Reasoning in Quantified Non-Classical Logics, 1 July 2024, Nancy, France

*Corresponding author.

†These authors contributed equally.

didier.galmiche@loria.fr (D. Galmiche); brandon.hornbeck@loria.fr (B. Hornbeck); daniel.mery@loria.fr (D. Méry)

$(\approx_1)$ $\mathrm{A} \approx \mathrm{A}$

$(\approx_2)$ $(\mathrm{A} \approx \mathrm{B}) \supset (\neg\mathrm{A} \approx \neg\mathrm{B})$

$(\approx_3)$ $(\mathrm{A} \approx \mathrm{B}) \supset (\mathrm{B} \supset \mathrm{A})$

$(\approx_4)$ $(\mathrm{A} \approx \mathrm{B}) \wedge (\mathrm{C} \approx \mathrm{D}) \supset (\mathrm{A} \otimes \mathrm{C}) \approx (\mathrm{B} \otimes \mathrm{D})$ where $\otimes \in \{\wedge, \vee, \supset, \approx\}$

(IL_1)	$\mathrm{A} \supset (\mathrm{B} \supset \mathrm{A})$	(IL_2)	$(\mathrm{A} \supset \mathrm{B}) \supset ((\mathrm{A} \supset (\mathrm{B} \supset \mathrm{C})) \supset (\mathrm{A} \supset \mathrm{C}))$
(IL_3)	$\mathrm{A} \supset (\mathrm{B} \supset (\mathrm{A} \wedge \mathrm{B}))$	(IL_4)	$(\mathrm{A} \wedge \mathrm{B}) \supset \mathrm{A}$
(IL_5)	$(\mathrm{A} \wedge \mathrm{B}) \supset \mathrm{B}$	(IL_6)	$(\mathrm{A} \supset \mathrm{C}) \supset ((\mathrm{B} \supset \mathrm{C}) \supset ((\mathrm{A} \vee \mathrm{B}) \supset \mathrm{C}))$
(IL_7)	$\mathrm{A} \supset (\mathrm{A} \vee \mathrm{B})$	(IL_8)	$\mathrm{B} \supset (\mathrm{A} \vee \mathrm{B})$
(IL_9)	$(\mathrm{A} \supset \mathrm{B}) \supset ((\mathrm{A} \supset \neg\mathrm{B}) \supset \neg\mathrm{A})$	(IL_{10})	$\neg\mathrm{A} \supset (\mathrm{A} \supset \mathrm{B})$

(MP) From A and $\mathrm{A} \supset \mathrm{B}$ deduce B.

Figure 1: Axioms for ISCI.

Example 1. *The following is an* $\mathsf{H}_{\mathsf{ISCI}}$ *proof that* $\approx$ *is commutative, more precisely, we show* $\mathrm{A} \approx \mathrm{B} \vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{B} \approx \mathrm{A}$*:*

(1)	$\mathrm{A} \approx \mathrm{B}$	*assumption*
(2)	$\mathrm{B} \approx \mathrm{B}$	$\approx_1$
(3)	$((\mathrm{B} \approx \mathrm{B}) \wedge (\mathrm{A} \approx \mathrm{B})) \supset ((\mathrm{B} \approx \mathrm{A}) \approx (\mathrm{B} \approx \mathrm{B}))$	$\approx_4$
(4)	$(\mathrm{B} \approx \mathrm{B}) \supset ((\mathrm{A} \approx \mathrm{B}) \supset ((\mathrm{B} \approx \mathrm{B}) \wedge (\mathrm{A} \approx \mathrm{B})))$	IL_3
(5)	$(\mathrm{A} \approx \mathrm{B}) \supset ((\mathrm{B} \approx \mathrm{B}) \wedge (\mathrm{A} \approx \mathrm{B}))$	MP 2, 4
(6)	$(\mathrm{B} \approx \mathrm{B}) \wedge (\mathrm{A} \approx \mathrm{B})$	MP 1, 5
(7)	$(\mathrm{B} \approx \mathrm{A}) \approx (\mathrm{B} \approx \mathrm{B})$	MP 3, 6
(8)	$((\mathrm{B} \approx \mathrm{A}) \approx (\mathrm{B} \approx \mathrm{B})) \supset ((\mathrm{B} \approx \mathrm{B}) \supset (\mathrm{B} \approx \mathrm{A}))$	$\approx_3$
(9)	$(\mathrm{B} \approx \mathrm{B}) \supset (\mathrm{B} \approx \mathrm{A})$	MP 7, 8
(10)	$\mathrm{B} \approx \mathrm{A}$	MP 2, 9

□

3. Semantics for ISCI

In this work we use the Topological Beth (TB) semantics for ISCI and briefly review its main concepts and results introduced in [4].

Definition 2. *Let* $\mathbf{M}$ *be a set of elements, called* worlds*, such that* $\omega, \pi \in \mathbf{M}$ *and* $\omega \neq \pi$*. A* TB frame *is a bounded distributive lattice* $\mathcal{F} = (\mathbf{M}, \leqslant, \sqcup, \omega, \sqcap, \pi)$ *with* ω *and* π *as least and greatest elements respectively.*

Definition 3. *A* TB pre-model *is a triple* $\mathcal{M} = (\mathcal{F}, [\cdot], \Vdash)$*, where* $\mathcal{F}$ *is a* TB *frame, and* $[\cdot]$ *is a* valuation function *from* $\mathbf{M}$ *to* $\wp(\mathbf{P} \cup \mathbf{F}_\approx)$*, such that for all worlds* m *and* n*:*

$(\mathcal{M}_\pi)$ $[\pi] = \mathbf{P} \cup \mathbf{F}_\approx$*,*

$(\mathcal{M}_\mathrm{K})$ *if* $\mathrm{m} \leqslant \mathrm{n}$ *then* $[\mathrm{m}] \subseteq [\mathrm{n}]$*,*

$(\mathcal{M}_{\approx_1})$ $\mathrm{A} \approx \mathrm{A} \in [\mathrm{m}]$*,*

$(\mathcal{M}_{\approx_4})$ *for all* $\otimes \in \{\wedge, \vee, \supset, \approx\}$*, if* $\mathrm{A} \approx \mathrm{B}, \mathrm{C} \approx \mathrm{D} \in [\mathrm{m}]$ *then* $\mathrm{A} \otimes \mathrm{C} \approx \mathrm{B} \otimes \mathrm{D} \in [\mathrm{m}]$*.*

The forcing relation $\Vdash$ *is inductively defined as the smallest relation on* $\mathbf{M} \times \mathbf{F}$ *such that:*

- $\mathrm{m} \Vdash \mathrm{p}$ *iff* $\mathrm{p} \in [\mathrm{m}]$*,*
- $\mathrm{m} \Vdash \mathrm{A} \approx \mathrm{B}$ *iff* $\mathrm{A} \approx \mathrm{B} \in [\mathrm{m}]$*,*
- $\mathrm{m} \Vdash \bot$ *iff* $\pi \leqslant \mathrm{m}$*,*
- $\mathrm{m} \Vdash \mathrm{A} \wedge \mathrm{B}$ *iff* $\mathrm{m} \Vdash \mathrm{A}$ *and* $\mathrm{m} \Vdash \mathrm{B}$*,*

- $\mathrm{m} \Vdash \mathrm{A} \supset \mathrm{B}$ *iff for all* $\mathrm{n} \in \mathbf{M}$*, if* $\mathrm{n} \Vdash \mathrm{A}$ *then* $\mathrm{m} \sqcup \mathrm{n} \Vdash \mathrm{B}$*,*
- $\mathrm{m} \Vdash \mathrm{A} \vee \mathrm{B}$ *iff for some* $\mathrm{n}_1, \mathrm{n}_2 \in \mathbf{M}$ *such that* $\mathrm{n}_1 \sqcap \mathrm{n}_2 \leqslant \mathrm{m}$*,* $\mathrm{n}_1 \Vdash \mathrm{A}$ *and* $\mathrm{n}_2 \Vdash \mathrm{B}$*.*

A TB model *is a* TB *pre-model satisfying the admissibility condition:*

$(\mathcal{M}_{\approx_3})$ *if* $\mathrm{m} \Vdash \mathrm{A} \approx \mathrm{B}$ *then* $\mathrm{m} \Vdash \mathrm{B} \supset \mathrm{A}$*.*

Let us remark that $\mathcal{M}_\pi$ implies that all TB models have a world π that forces all formulas including $\bot$. As usual, a formula A is *true* (or *satisfied*) in a TB model $\mathcal{M}$, written $\mathcal{M} \vDash \mathrm{A}$, iff $\mathrm{m} \Vdash \mathrm{A}$ for all worlds m in $\mathcal{M}$ and *valid*, written $\vDash$A, iff it is true in all models. It is routine to show that $\mathcal{M}_\pi$ and $\mathcal{M}_\mathrm{K}$ extend from propositional letters and equations to all formulas. $\mathcal{M}_\mathrm{K}$ is the well-known Kripke monotonicity condition, which applies to equations in our setting.

Definition 4. *Let* $\mathcal{M} = (\mathcal{F}, [\cdot], \Vdash)$ *be a* TB *model.* $\mathcal{M}$ *is* regular *iff for all formulas* A*, if* $\mathrm{m} \Vdash \mathrm{A}$ *for some world* m*, then there exists a world* $\mathrm{m_A}$*, called* A-minimal*, such that* $\mathrm{m_A} \Vdash \mathrm{A}$ *and for all worlds* n*,* $\mathrm{n} \Vdash \mathrm{A}$ *implies* $\mathrm{m_A} \leqslant \mathrm{n}$*. We write* $\vDash_\mathrm{r}$ *for the restriction of validity to the class of* regular TB *models.*

Theorem 1 (Adequacy of regular Beth models). *Regular* TB *models are* $\mathsf{H}_{\mathsf{ISCI}}$*-sound and* $\mathsf{H}_{\mathsf{ISCI}}$*-complete: if* $\vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{A}$ *then* $\vDash_\mathrm{r} \mathrm{A}$*, and if* $\vDash_\mathrm{r} \mathrm{A}$ *then* $\vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{A}$*.*

4. Labelled Deduction for ISCI

Since our prover AutoPSI implements the $\mathsf{L}_{\mathsf{ISCI}}$ labelled calculus defined in [4] (more precisely the $\mathsf{L}^2_{\mathsf{ISCI}}$ variant of the calculus), let us briefly recall its basic concepts.

The set $\mathbf{L}$ of *labels* is the union of the set $\mathbb{N}$ with all of its finite subsets. We use the (possibly subscripted or primed) letters $\mathrm{a}, \mathrm{b}, \mathrm{c}$ to denote singletons and save the letters $\mathrm{x}, \mathrm{y}, \mathrm{z}$ to denote arbitrary labels. A label x is a *sublabel* of a label y if $\mathrm{x} \subseteq \mathrm{y}$.

We work with a labelling algebra $\mathcal{L}$ defined as the lattice $(\mathbf{L}, \subseteq, \cup, \emptyset, \cap, \mathbb{N})$, where join $\cup$ and meet $\cap$ are standard set union and intersection. We consider that $\cup$ binds stronger than $\cap$ and we shall frequently write xy instead of $\mathrm{x} \cup \mathrm{y}$ ($\mathrm{xx'} \cap \mathrm{yy'}$ should therefore be read as $(\mathrm{x} \cup \mathrm{x'}) \cap (\mathrm{y} \cup \mathrm{y'})$). In this paper, we shall only use examples with label letters built from the subset $\{ 1, \ldots, 9 \}$. Therefore, we shall use the more concise notation 13 to unambiguously refer to $\{ 1, 3 \}$ (and not to the singleton $\{ 13 \}$).

Definition 5. *A* labelled formula *is a pair* (C, x)*, written* $\mathrm{C} : \mathrm{x}$*, where* C *is a formula and* x *is a label. A* labelled sequent *is a pair* (Γ, Δ)*, written* $\Gamma \vdash \Delta$*, where* Γ, Δ *are* sets *of labelled formulas.*

Given a set Δ of labelled formulas, the notation $\mathrm{x} \sqsubseteq \Delta$ means that $\mathrm{x} \subseteq \mathrm{y}$ for some labelled formula $\mathrm{A} : \mathrm{y}$ occurring in Δ. We write $\lceil \Delta \rceil$ for the set of all labels that are maximal in Δ: $\lceil \Delta \rceil = \{ \mathrm{z} \sqsubseteq \Delta \mid$ if $\mathrm{u} \sqsubseteq \Delta$ and $\mathrm{z} \subseteq \mathrm{u}$ then $\mathrm{u} = \mathrm{z} \}$. Let $\mathrm{s} = \Gamma \vdash \Delta$ be a labelled sequent. A label x is *right maximal in* s if $\mathrm{x} \in \lceil \Delta \rceil$ and s is *right connected* iff $(\forall\, \mathrm{A} : \mathrm{x} \in \Gamma)(\mathrm{x} \sqsubseteq \Delta)$.

The labelled calculus $\mathsf{L}_{\mathsf{ISCI}}$ deals with labelled sequents $\Gamma \vdash \Delta$ where Δ is not allowed to be empty. Some of the rules ($\bot_\mathrm{L}$, $\vee_\mathrm{L}$, $\approx_{\mathrm{LL}}$, $\approx_{\mathrm{LR}}$) have two principal formulas. In this case, one is called *primarily principal* and the other *secondarily principal.* In $\approx_{\mathrm{LL}}$ and $\approx_{\mathrm{LR}}$, the equation that provides the substitution is primarily principal while the formula in which the substitution occurs is secondarily principal. In $\bot_\mathrm{L}$ and $\vee_\mathrm{L}$, the primarily principal formula is the one in the antecedent of the conclusion.

The rule $\approx_{\mathrm{LL}}$ replaces (some, possibly all) occurrences of C in D with B and we write $\mathrm{D}^\mathrm{C}_\mathrm{B}$ as a shorthand for $\mathrm{D}[\mathrm{C}/\mathrm{B}]$. Similarly for $\approx_{\mathrm{LR}}$ with A instead of D. We call such substitutions *sentential substitutions.*

Definition 6. *A formula* A *is a* theorem of *(or* derivable in*)* $\mathsf{L}_{\mathsf{ISCI}}$*, written* $\vdash_{\mathsf{L}_{\mathsf{ISCI}}} \mathrm{A}$*, if* $\vdash \mathrm{A} : \emptyset$ *is derivable in* $\mathsf{L}_{\mathsf{ISCI}}$*.*

Identity Rules:

$$\frac{}{\Gamma, p : x \vdash \Delta, p : y}\,\mathrm{id}_p(x \subseteq y) \qquad \frac{}{\Gamma, A \approx B : x \vdash \Delta, A \approx B : y}\,\mathrm{id}_{\approx}(x \subseteq y)$$

Core Intuitionistic Rules:

$$\frac{\Gamma, B \supset C : x \vdash \Delta, B : z \quad \Gamma, B \supset C : x, C : xz \vdash \Delta}{\Gamma, B \supset C : x \vdash \Delta}\supset_L(xz \sqsubseteq \Delta)$$

$$\frac{\Gamma, A : a \vdash \Delta, B : ya}{\Gamma \vdash \Delta, A \supset B : y}\supset_R \quad \frac{\Gamma, B : x, C : x \vdash \Delta}{\Gamma, B \wedge C : x \vdash \Delta}\wedge_L \quad \frac{\Gamma \vdash \Delta, A : y \quad \Gamma \vdash \Delta, B : y}{\Gamma \vdash \Delta, A \wedge B : y}\wedge_R$$

Disjunction and Falsity Rules:

$$\frac{}{\Gamma, \bot : x \vdash \Delta, A : y}\bot_L(x \subseteq y) \quad \frac{\Gamma \vdash \Delta, A_1 : y, A_2 : y}{\Gamma \vdash \Delta, A_1 \vee A_2 : y}\vee_R$$

$$\frac{\Gamma, B \vee C : x, B : xa_1 \vdash \Delta, A : ya_1 \quad \Gamma, B \vee C : x, C : xa_2 \vdash \Delta, A : ya_2}{\Gamma, B \vee C : x \vdash \Delta, A : y}\vee_L(x \subseteq y)$$

Sentential Identity Rules:

$$\frac{\Gamma, B \approx C : x, C \supset B : x \vdash \Delta}{\Gamma, B \approx C : x \vdash \Delta}\approx_{L3}$$

$$\frac{\Gamma, B \approx C : x, D : x, D^C_B : x \vdash \Delta}{\Gamma, B \approx C : x, D : x \vdash \Delta}\approx_{LL} \qquad \frac{\Gamma, B_1 \approx B_2 : x, B_i \approx B_i : x \vdash \Delta}{\Gamma, B_1 \approx B_2 : x \vdash \Delta}\approx_{LL'}$$

$$\frac{\Gamma, B \approx C : x \vdash \Delta, A : y, A^C_B : y}{\Gamma, B \approx C : x \vdash \Delta, A : y}\approx_{LR}(x \subseteq y) \qquad \frac{}{\Gamma \vdash \Delta, A \approx A : y}\approx_R$$

Maximality Rule:

$$\frac{\Gamma, B \supset C : x \vdash \Delta, B : xz \quad \Gamma, B \supset C : x, C : xz \vdash \Delta}{\Gamma, B \supset C : x \vdash \Delta}\supset_L(xz \in \lceil \Delta \rceil)$$

Eigenvariable conditions: In $\supset_R$ and $\vee_L$, a, a_1, a_2 are fresh singletons and $a_1 \neq a_2$.

Figure 2: Rules for the $\mathsf{L}_{\mathsf{ISCI}}$ Labelled Sequent Calculus.

A very important feature of $\mathsf{L}_{\mathsf{ISCI}}$, stemming from the use of Topological Beth semantics instead of Kripke semantics, is that it remains sound if the eigenvariable conditions are dropped. Therefore, when a labelled formula $C : x$ requiring the introduction of fresh labels has to be expanded, one can reuse the labels that were generated during the first expansion of the formula $C : x$ (or of any formula of the form $C : y$). The use of $\mathsf{L}_{\mathsf{ISCI}}$ without the eigenvariable is called *liberalized* $\mathsf{L}_{\mathsf{ISCI}}$. Since AutoPSI is an implementation of liberalized $\mathsf{L}_{\mathsf{ISCI}}$, we only consider liberalized derivations in the remainder of the paper.

Theorem 2 (Liberalized soundness). *If* A *is provable in liberalized* $\mathsf{L}_{\mathsf{ISCI}}$ *then* $\vdash_{\mathsf{H}_{\mathsf{ISCI}}} A$.

Example 2. *Let us consider the following partial derivation for the non-valid formula* $((p \vee q) \supset p) \vee ((p \vee q) \supset q)$*, where the second instance of* $\supset_R$ *reuses the label* 1 *introduced by the first instance:*

$$\Pi\left\{\begin{array}{l}\dfrac{\dfrac{p \vee q : 1, p \vee q : 1 \vdash p : 1, q : 1}{p \vee q : 1 \vdash p : 1, (p \vee q) \supset q : \emptyset}\supset_R}{\vdash (p \vee q) \supset p : \emptyset, (p \vee q) \supset q : \emptyset}\supset_R}{\vdash ((p \vee q) \supset p) \vee ((p \vee q) \supset q) : \emptyset}\vee_R\end{array}\right.$$

With a standard Kripke rule for left disjunction that would simply propagate the labels, we would get a proof for a non-valid formula as follows:

$$\dfrac{\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{p}:1\vdash\mathrm{p}:1,\mathrm{q}:1}\,\mathrm{id_p}\quad\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:1\vdash\mathrm{p}:1,\mathrm{q}:1}\,\mathrm{id_p}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{p}\vee\mathrm{q}:1\vdash\mathrm{p}:1,\mathrm{q}:1}\,\vee^{\mathrm{K}}_{\mathrm{L}}$$

On the contrary, in $\mathsf{L}_{\mathsf{ISCI}}$*, applying the Beth rule for left disjunction we get*[1]*:*

$$\dfrac{\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{p}:12\vdash\mathrm{p}:12}\,\mathrm{id_p}\quad\dfrac{\overset{\mathrm{s}}{\text{- -}}\qquad\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13\vdash\mathrm{q}:13}\,\mathrm{id_p}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:12\quad\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13\vdash\mathrm{p}:13,\mathrm{q}:1}\,\vee_{\mathrm{L}}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{p}\vee\mathrm{q}:1\vdash\mathrm{p}:1,\mathrm{q}:1}\,\vee_{\mathrm{L}}$$

The left premiss of the topmost instance of $\vee_{\mathrm{L}}$ *gives rise to the sequent* s*, in which* $\mathrm{p}\vee\mathrm{q}:1$ *can be reexpanded using either* $\mathrm{p}:12$*, or* $\mathrm{q}:13$ *in the succedent. Both choices reintroduce* s *as a premiss of* $\vee_{\mathrm{L}}$ *making it impossible to reach an axiom as depicted below.*

$$\dfrac{\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:123,\mathrm{q}:12}\,\mathrm{id_p}\quad\dfrac{\overset{\mathrm{s}}{\text{- -}}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:12}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:12}\,\vee_{\mathrm{L}}$$

$$\dfrac{\dfrac{\overset{\mathrm{s}}{\text{- -}}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:12}\quad\dfrac{}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:123}\,\mathrm{id_p}}{\mathrm{p}\vee\mathrm{q}:1,\mathrm{q}:13,\mathrm{p}:12\vdash\mathrm{p}:13,\mathrm{q}:12}\,\vee_{\mathrm{L}}$$

□

Let us assume some fixed strict total Nœtherian order $\lhd$ on $\mathbf{F}$ such that:

- $\bot\lhd\mathrm{A}$ for all $\mathrm{A}\neq\bot$,
- if $|\mathrm{A}|<|\mathrm{B}|$ then $\mathrm{A}\lhd\mathrm{B}$, and
- if $\mathrm{A}\lhd\mathrm{B}$ then $\mathrm{C}[\mathrm{A}/\mathrm{B}]\lhd\mathrm{C}$.

where $|\mathrm{C}|$ denotes the size of a formula C defined as the number of its connectives.

Definition 7. $\mathsf{rL}_{\mathsf{ISCI}}$ *is* $\mathsf{L}_{\mathsf{ISCI}}$ *under the following restrictions:*

(R1) *Only equations can be secondarily principal for* $\approx_{\mathrm{LR}}$ *and* $\approx_{\mathrm{LL}}$*.*

(R2) *Equations of the form* $\mathrm{A}\approx\mathrm{A}$ *are never (primarily or secondarily) principal for* $\approx_{\mathrm{LR}}$ *and* $\approx_{\mathrm{LL}}$*.*

(R3) $\approx_{\mathrm{LR}}$ *only performs strictly decreasing substitutions, i.e. if the rule replaces* C *with* B *in* A *then* $\mathrm{A}^{\mathrm{C}}_{\mathrm{B}}\lhd\mathrm{A}$*.*

(R4) $\approx_{\mathrm{LR}}$ *and* $\approx_{\mathrm{LL}}$ *only perform uniform substitutions, i.e., they make all possible replacements in the secondarily principal formula.*

(R5) $\approx_{\mathrm{LR}}$ *and* $\approx_{\mathrm{LL}}$ *only perform substitutions that preserve the main connective of the secondarily principal formula.*

Restrictions R1, R2 and R3 are required for the cut-elimination proof developped in [4]. The other restrictions are not mandatory but they simplify the implementation of the AutoPSI theorem prover. Let us remark that R1 guarantees that sentential substitutions should only occur inside equations, while R5 guarantees that equations should remain equations after a sentential substitution.

Since all of the axioms and rules of $\mathsf{H}_{\mathsf{ISCI}}$ are derivable in $\mathsf{rL}_{\mathsf{ISCI}}$ and since cut can be eliminated from $\mathsf{rL}_{\mathsf{ISCI}}$, we have the following completeness result:

Theorem 3 (Completeness under sentential restrictions). *If* $\vdash_{\mathsf{H}_{\mathsf{ISCI}}}\mathrm{A}$ *then* A *is provable in* $\mathsf{rL}_{\mathsf{ISCI}}$*, i.e., in* $\mathsf{L}_{\mathsf{ISCI}}$ *with the restrictions of Definition 7.*

[1] Irrelevant formulas are omitted to keep the proof in the page width.

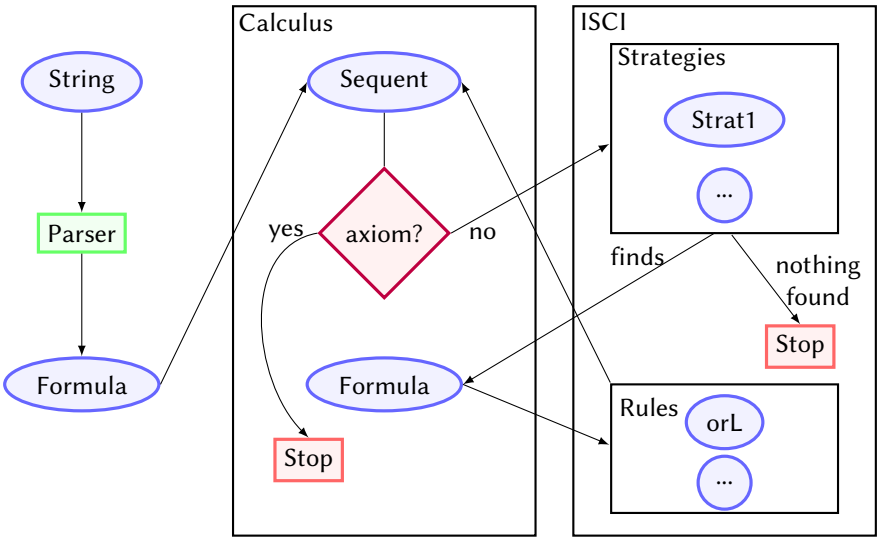

Figure 3: Execution diagram

The completeness result can be further extended to proofs in which rules with principal formulas occurring in the succedent (right principal) are only allowed to be applied if their right principal formulas have a right maximal label. Such a rule application strategy is called *right maximality* Finally, a last important property of $\mathsf{L}_{\mathsf{ISCI}}$ and $\mathsf{rL}_{\mathsf{ISCI}}$ is that the set of provable formulas remains the same if one replaces the original rule for right implication with the maximality rule depicted in Fig. 2.

Theorem 4 (Completeness under right maximality). *If* $\vdash_{\mathsf{H}_{\mathsf{ISCI}}} \mathrm{A}$ *then* A *is provable in* $\mathsf{L}_{\mathsf{ISCI}}$ *or in* $\mathsf{rL}_{\mathsf{ISCI}}$ *both restricted to the right maximality strategy.*

5. AutoPSI: a Prover for ISCI

In this section we present AutoPSI, an implementation of liberalized $\mathsf{rL}_{\mathsf{ISCI}}$ written in Java. The parsing of the formulas is generated by ANTLR from the grammar described in Section 2. The Java application consists in 31 classes, 4 of them being automatically generated by ANTLR.
AutoPSI is available at https://homepages.loria.fr/dmery/autopsi. Let us note that a screen capture of an AutoPSI session is given in Appendix A.

5.1. Prover Architecture

The global architecture of AutoPSI is depicted in the execution diagram in Fig. 3. The input of the prover is a string representing the formula to prove, using the following grammar :

- FORMULA :
  - (FORMULA)
  - FORMULA | FORMULA
  - FORMULA&FORMULA
  - FORMULA->FORMULA
  - FORMULA=FORMULA
  - ATOM
  - F (for $\bot$)
- ATOM : [a-z]*

Algorithm 1: Global Approach

Input: $s : string$
Data: $S : Sequent(setof formulas)$;
$F : Formula$;
$P : Prooftree$
Output: $V : boolean$

```
1 S ← ∅;
2 F ← Parse(s);
3 S ← S ∪ {F}; // Right-hand side of the sequent
4 P ← Compute(S, ∅); // Returns a proof tree
5 V ← IsAProof(P); // True iff all leaves are axioms
6 return V;
```

This string is parsed, creating an instance of the class Formula which is a syntactic tree of the formula. This formula is put into a sequent, which is a list of signed formulas (positive for the left-hand side, negative for the right-hand side). Then the majority of the computation is done by the class Calculus, which for every sequent starts by checking if it is an axiom, if it is an axiom then the computation stops, if not we need to apply a rule. The program then calls for a choice for the next formula in the sequent to decompose. This choice is made by a strategy that gives priorities to the available formulas. Once a formula is chosen, the program can check which rule to apply to the sequent depending on the formula. The application of this rule creates one or two new sequents in which the chosen formula is marked as used (except for the positive implications which can be used several times).

The Calculus class is designed to work with abstract classes for formulas, and interfaces for rules and strategies. This should allow us to reuse the core of the system for other sequent calculi.

Since AutoPSI is an implementation of $\mathsf{L}_{\mathsf{ISCI}}$, we need to deal with labelled formulas. Labels in $\mathsf{L}_{\mathsf{ISCI}}$ are sets of integers that are implemented in AutoPSI as instances of the HashSet class in Java. This allows us to easily perform the inclusion tests required as side conditions by some of the rules like $\supset_{\mathrm{L}}$, $\vee_{\mathrm{L}}$, the axiom rules and the sentential identity rules. From a technical point of view, inclusion is simply checked as inclusion between Java HashSet instances.

The mitigation of the eigenvariable condition is implemented in a singleton class that keeps track of all currently generated singleton labels. The singleton class allows singleton labels to be reused by considering them as minimal w.r.t. the formula they were firstly introduced with. Minimality is achieved by managing a dictionary whose keys are formulas and values are label letters. Whenever a formula requiring a fresh singleton label needs to be introduced in a sequent, for instance when applying the rule $\vee_{\mathrm{L}}$ to a labelled formula $\mathrm{A} \vee \mathrm{B} : \mathrm{x}$, we first check whether the dictionary already contains a key-value pair $\mathrm{A} : \mathrm{a_i}$ (meaning that $\mathrm{a_i}$ is A-minimal). If so, the singleton $\mathrm{a_i}$ is reused and $\mathrm{A} : \mathrm{a_i}$ is inserted in the antecedent of the current sequent. Otherwise, a fresh label letter $\mathrm{a_j}$ is generated by the singleton class and associated with A is the dictionary for later reuse before inserting $\mathrm{A} : \mathrm{a_j}$ in the current sequent.

5.2. Proof Search, Optimizations and Strategies

Given a formula as input, AutoPSI explores the proof search space using a depth-first search policy. During the exploration, the prover keeps track of all the rules that have been applied previously (as well as the formulas on which they were applied) in a tree structure called the rule-tree. To reduce the memory footprint, the sequents themselves are not explicitly stored in the rule-tree but can be recovered from the input formula by replaying all the rules up to a given point. If all the leaves of the rule-tree are axioms (zero-premiss rules) the input formula is deemed valid. In this case, the rule-tree can be converted to an actual proof tree (with all of its intermediate sequents as its nodes) if needed. Otherwise, if all possible rules have already been applied and all backtracking points have been exhausted, the input formula is deemed non-valid. The global approach and the detailed pseudo code of the proof

Algorithm 2: Compute

Input: $S : Sequent(Setoformulas)$;
$P : ProofTree$
Data: $S_1, S_2 : Sequent(setoformulas)$;
$F, FR : Formula$;
$P_1, P_2 : Prooftree$;
$Rules : Rule[]$;
$R : Rule$
Output: $P : Prooftree$

```
1  if S is not an axiom then
2      F ← Choose(S); // A formula is chosen for the derivation
3      Rules ← Rule(F);
4      foreach R in Rules do
5          if R needs a choice on the right then
6              i ← 0;
7              while P1 is null and i < Size(SR) do
                   // We check all possible choices until we find a proof
8                  FR ← SR(i); // of the right formulas of S
9                  < S1, S2 >← Apply(R, S, F, FR);
10                 P1 ← Compute(P, S1);
11                 if IsAProof(P1) then
12                     P2 ← Compute(P, S2);
13                     if IsAProof(P2) then
14                         P ← Compose(P1, P2, P); // We merge the proof trees
15                     end
16                     else
17                         P1, P2 ← null;
18                     end
19                 end
20                 else
21                     P1, P2 ← null;
22                 end
23                 i ← i + 1;
24             end
25             return P;
26         end
27         else
               // R does not need a choice
28             < S1, S2 >← Apply(R, S, F, FR);
29             P1 ← Compute(∅, S1);
30             P2 ← Compute(P, S2);
31             P ← Compose(P1, P2, P); // We merge the proof trees
32             return P;
33         end
34     end
35 end
36 else
       // S is an Axiom
37     return P;
38 end
```

search procedure are given in Algorithms 1 and 2. In Algorithm 2, the only data that we keep track of is the order of application of the rules in the proof, that we call a ProofTree. This allows the prover to only deal with one sequent at a time, while still having the information needed to build the proof if asked to. The while loop starting on line 7 states that we check every possible choice for the proof until we either find a proof, or we have checked every suitable formula for this rule application. For example, if the choices for the secondarily principal formula are between formulas A and B, then we start with formula A first. If the proof is completed, we stop there, if not, the ProofTree will be null, then the loop will go on and try with B. When the proof is completed, the use of the predicate Compose on lines 14 and 31 will merge the ProofTrees (the two trees created by the rule application, P1 and P2, and the former one that represents the proof below, P, into the new ProofTree P) and we will get an axiom at the top of the ProofTree. Then the call to the procedure IsAProof in line 5 of algorithm 1 will return true.

AutoPSI implements all of the restrictions described in Definition 7 and further restricts $\approx_{\mathrm{LL}}$ to sentential substitutions bounded by the size of the initial formula to prove which is called the *degree* of the proof. Let us remark that such a restriction is proven complete for SCI in [5]. In the case of ISCI the completeness result is only achieved, via counter-model construction, for a fragment where formulas are syntactically restricted to implications and identities only [6], but not for the full logic. A current limitation of AutoPSI is that it cannot generate a counter-model in case of non-validity.

5.2.1. Optimizations

Following the terminology of one-sided sequents, we assign a *polarity* to each formula occurring in a labelled sequent: *positive* if it occurs in the antecedent and *negative* if it occurs in the succedent.

A first optimization, called *subsumption*, takes advantage of Kripke monotonicity (condition $\mathcal{M}_{\mathrm{K}}$ of Definition 3) to subsume formulas on both sides of a sequent and thus prevent them from being expanded. More precisely, a positive formula $\mathrm{A} : \mathrm{y}$ is considered subsumed (and thus prevented from expansion) if the current sequent already contains a positive formula $\mathrm{A} : \mathrm{x}$ such that $\mathrm{x} \subseteq \mathrm{y}$. Similarly for negative formulas such that $\mathrm{y} \subseteq \mathrm{x}$. Since we create a finite number of formulas and labels, subsumption is not needed for termination. That is because the prover works with sets of formulas and if a formula is already in the set, it is not added to the sequent, but subsumption will limit the number of useless decompositions in the proof.

A second optimization is that we put an upper bound on the number of times positive implications can be reused to ensure that they do not get expanded infinitely often by $\supset_{\mathrm{L}}$. This upper bound is called "Time to Live" (TTL) and is implemented as a counter that is equal to the sum of the number of negative implications and twice the number of positive disjunctions occurring in a sequent as those rules are the only ones that might generate fresh singletons. $\mathsf{L}_{\mathsf{ISCI}}$ does not enjoy the subformula property (in its strict acception) since sentential substitutions might generate subformulas that did not occur in the initial formula to prove. Therefore, the TTL is updated dynamically during the proof search process when new singletons are generated.

Theorem 5. *The proof system* $\mathsf{L}_{\mathsf{ISCI}}$ *in which positive implications are not used more times than their TTL is complete.*

Proof. When a positive implication is decomposed, there is a list of candidates for the formula on the right-hand side, which are the formulas with bigger labels than the implication being decomposed. In this selection of candidates, there are maximal formulas, i.e. formulas that have labels that are not smaller than any other label on the right-hand side. The prover will try with one formula and if it does not find a proof, it will backtrack to try with another candidate. Then it will eventually use a maximal formula as a secondarily principal formula. Since the system that uses maximality is complete, this step does not prevent completeness. Then, if the implication is decomposed again in the proof, the prover will try a new candidate if and only if there is a new label, that means if a new label atom has been introduced since the last decomposition. Then, each positive implication can be decomposed at most a number of times that is equal to the number of labels created in the proof, that is the formula's TTL. □

5.2.2. Strategies

At each step of the exploration, the prover needs to choose the next formula to be expanded (called the principal formula). This task is devoted to an object called a strategy which implements the Strategy interface. Every time a rule is decomposed, the principal formula is marked as being used, with the exception of the implications on the left-hand side of the sequent that can be used several times. The rule application strategy used in AutoPSI is to delay choices as long as possible. There are two kinds of choices: the ones that involve the guessing of labels satisfying some side conditions as it is the case for the rule $\supset_{\mathrm{L}}$ and the ones that involve the choice of a secondarily principal formula as it is the case for the rules $\vee_{\mathrm{L}}$, $\approx_{\mathrm{LL}}$ and $\approx_{\mathrm{LR}}$. Making the right choices for secondarily principal formulas is the biggest performance issue that the prover currently faces. The number of suitable formulas can be large and when a formula is not valid we have to explore every possible choice before failing eventually.

Our first strategy is called "First One Strategy". In this strategy, the prover decomposes formulas in their order of apparition in the sequent, the older ones having priority. This strategy is not complete.

Our second strategy is called "New Labels First". It gives precedence to the rules that introduce new labels in the proof, such as $\supset_{\mathrm{R}}$ and $\vee_{\mathrm{L}}$, in order to have them ready for occurrences of positive implications and right-handed substitutions that need bigger labels in their right-principal formula. The rules that require choices and do not create labels, like $\supset_{\mathrm{L}}$ and all identity rules are chosen last, in order to delay choices higher in the proof. For the choice of secondarily principal formulas, the candidates are ordered by increasing size, with the hope that a small formula should be less likely to create a big number of branches and nodes. Therefore, in the case that such a choice should eventually fail, backtracking to the next one would not cost as much as for a bigger formula.

A third strategy is called "Partial Δ-Maximality". It uses the same priorities as the "New Labels First" strategy, but it only allows the rules $\supset_{\mathrm{L}}$, $\vee_{\mathrm{L}}$, $\approx_{\mathrm{LL}}$ and $\approx_{\mathrm{LR}}$ to be expanded with secondarily principal formulas having right-maximal labels. It is therefore an implementation of the *maximality strategy* discussed in Section 4. Choosing formulas with right maximal labels reduces the number of candidates. For other rules, the strategy can still use non left-maximal formulas.

More clever strategies for the choice of secondarily principal formulas shall be studied as future work. One improvement to make better choices, inspired by connection methods, would be to aim for candidates that actually contain atoms that could be complementary with the primarily principal formula. One difficulty w.r.t. connection methods is that $\mathsf{L}_{\mathsf{ISCI}}$ does not enjoy the subformula property, so that complementarity cannot be precomputed once and for all before starting the proof search process and should be done on the fly.

Example 3. *Let us consider the formula* $\neg\neg(\mathrm{A} \vee \neg\mathrm{A})$. *Negation is not a native rule in* $\mathsf{L}_{\mathsf{ISCI}}$*, so our starting formula is in fact* $((\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot) \supset \bot$*, which contains two negative implications and no positive disjunctions. Therefore,* AutoPSI *should be allowed to expand the same positive implication at most twice.*

After the creation of the labelled sequent $\vdash ((\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot) \supset \bot : \emptyset$*, only one formula can be selected by the rule application strategy: the one in the succedent. After the rule* $\supset_{\mathrm{R}}$ *is applied we get the sequent* $(\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1 \vdash \bot : 1$ *(for conciseness we do not keep expanded formulas). The only selectable formula is a positive implication. After* $\supset_{\mathrm{L}}$ *is applied we get the following premises.*

$$\frac{(\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1 \vdash \bot : 1, \mathrm{A} \vee (\mathrm{A} \supset \bot) : 1 \quad (\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1, \bot : 1 \vdash \bot : 1}{(\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1 \vdash \bot : 1} \supset_{\mathrm{L}}$$

The second premiss is the axiom $\bot_{\mathrm{L}}$ *so its exploration stops successfully. The first one still needs to be explored further. We have now the choice between two rules,* $\supset_{\mathrm{L}}$ *(still usable) and* $\vee_{\mathrm{R}}$*. The rule application strategy puts the priority on* $\vee_{\mathrm{R}}$*, resulting in the sequent* $(\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1 \vdash \bot : 1, \mathrm{A} : 1, \mathrm{A} \supset \bot : 1$*.*

Now have the choice between $\supset_{\mathrm{L}}$ *or* $\supset_{\mathrm{R}}$ *and the rule application strategy selects* $\supset_{\mathrm{R}}$*. The resulting sequent is* $(\mathrm{A} \vee (\mathrm{A} \supset \bot)) \supset \bot : 1, \mathrm{A} : 2 \vdash \bot : 1, \mathrm{A} : 1, \bot : 12$ *which is not an axiom yet but we can reuse the rule* $\supset_{\mathrm{L}}$ *to get two subproofs reaching axioms:*

$$\Pi_1 \left\{ \dfrac{\dfrac{}{(A \vee (A \supset \bot)) \supset \bot : 1, A : 2 \vdash \bot : 1, A : 1, \bot : 12, A \supset \bot : 12, A : 12}\,\mathrm{id}}{(A \vee (A \supset \bot)) \supset \bot : 1, A : 2 \vdash \bot : 1, A : 1, \bot : 12, A \vee (A \supset \bot) : 12}\,\vee_R \right.$$

$$\Pi_2 \left\{ \dfrac{}{(A \vee (A \supset \bot)) \supset \bot : 1, A : 2, \bot : 12 \vdash \bot : 1, A : 1, \bot : 12}\,\bot_L \right.$$

$$\dfrac{\Pi_1 \qquad\qquad \Pi_2}{(A \vee (A \supset \bot)) \supset \bot : 1, A : 2 \vdash \bot : 1, A : 1, \bot : 12}\,\supset_L$$

□

5.3. Tests and Benchmarks

Several indicators are built in AutoPSI to evaluate its efficiency. The first one is obviously the validity status (St) of the input formula (T for valid, F for invalid). The second one is the execution time measured in milliseconds (Tms). The third and fourth ones are the maximum depth and the size of the search space explored by the prover respectively measured as the maximum depth (D) and the number of nodes (N) in the rule-tree structure. We also measure the number of branches created (B). For valid formulas we also count the number of nodes in the final proof (Np) and its depth (Dp). In the case of a valid formula, the ratio between the size of the proof Np and the size of the search space N gives an account of the optimality of the proof search strategy: if N is significantly greater than Np then AutoPSI explored many wrong choices before making the right ones.

As AutoPSI is the first prover for ISCI, we do not have any other tool to compare it with. There is no substantial test base for ISCI, we thus derive our tests from the Φ formula introduced in [5], where a tableau prover for SCI (an extension of classical logic with the identity) is described:

$$\Phi \equiv (((q \approx p) \supset (p \supset r)) \approx ((p \supset (p \Leftrightarrow p)) \approx p)) \supset (((r \wedge p) \Leftrightarrow (p \approx p)) \vee ((p \wedge p) \vee \neg q))$$

We recall that negation $\neg A$ and logical equivalence $A \Leftrightarrow B$ have no specific rules in AutoPSI since they are only shorthands for $A \supset \bot$ and $(A \supset B) \wedge (B \supset A)$.

Φ is valid in SCI but it is not valid in ISCI, therefore we will try to prove $\neg\Phi$ and $\neg\neg\Phi$ as well, with the later expected to be valid thanks to the double negation. We will also study Ψ, which is Ψ but with identities instead of equivalences:

$$\Psi \equiv (((q \approx p) \supset (p \supset r)) \approx ((p \supset (p \approx p)) \approx p)) \supset (((r \wedge p) \approx (p \approx p)) \vee ((p \wedge p) \vee \neg q))$$

This transformation is not valid since two equivalent formulas are not necessarily identical. The column TTL stands for "Time To Live" and indicates the maximum number of expansions allowed for positive implications. Finally, we add χ:

$$\chi \equiv (((p \wedge (q \supset q_2)) \approx (p \wedge (q \supset q_2))) \wedge (((p \wedge (q \supset q_2)) \approx (r \vee p_2)) \approx (p \wedge (q \supset q_2)))) \supset ((p \wedge (q \supset q_2)) \approx (r \vee p_2))$$

Since ISCI is a conservative extension of IL, we complete our test base with three purely intuitionistic formulas:

$$\begin{aligned} F_1 &\equiv ((((p \Leftrightarrow q) \vee (p \Leftrightarrow r)) \vee (q \Leftrightarrow r)) \supset ((p \wedge q) \wedge r)) \supset ((p \wedge q) \wedge r) \\ F_2 &\equiv (\neg\neg(\neg p \supset q)) \supset ((\neg p \supset \neg q) \supset p) \\ F_3 &\equiv \neg\neg((((q \supset p) \supset (p \supset r)) \supset ((p \supset ((p \supset p) \wedge (p \supset p))) \supset p)) \\ &\quad \supset((((r \wedge p) \supset (p \supset p)) \wedge (p \supset p) \supset (r \wedge p))) \vee ((p \wedge p) \vee \neg q)) \end{aligned}$$

Table 2 summarizes the results of running AutoPSI on the ISCI test base in different conditions. Firstly without the subsumption optimization and without the maximality strategy, then with the subsumption optimization but without the maximality strategy, and finally with both the subsumption optimization and the maximality strategy.

It is clear that the subsumption optimization drastically improves the efficiency of the prover. Without it, most of the formulas in the test base cannot be decided in a reasonable amount of time. What the formula $\neg\neg\Phi$ shows us is that we explore too much, and that is mainly because of the rules $\supset_L$, $\vee_L$, $\approx_{LL}$ and $\approx_{LR}$, since we need to test all of their potential secondarily principal formulas. It would be a

Form	St	TTL	Conditions	T(ms)	D	N	B	DP	NP
Φ	False	10	(1)	-	-	-	-	-	-
			(2)	156.88	20	12185	6086	-	-
			(3)	52.00	20	30476	1734	-	-
$\neg\Phi$	False	9	(1)	10.57	11	1025	512	-	-
			(2)	2.29	11	219	109	-	-
			(3)	2.72	11	219	109	-	-
$\neg\neg\Phi$	-	-	(1)	-	-	-	-	-	-
			(2)	-	-	-	-	-	-
			(3)	-	-	-	-	-	-
$\neg\neg\neg\Phi$	False	10	(1)	-	-	-	-	-	-
			(2)	15461	24	597142	297429	-	-
			(3)	9409	24	266422	133208	-	-
Ψ	False	6	(1)	5.44	13	321	306	-	-
			(2)	3.9	13	539	265	-	-
			(3)	0.46	13	57	26	-	-
$\neg\Psi$	False	7	(1)	1.34	9	231	115	-	-
			(2)	0.58	9	83	41	-	-
			(3)	0.07	9	83	41	-	-
$\neg\neg\Psi$	False	7	(1)	-	-	-	-	-	-
			(2)	14071	27	749390	353560	-	-
			(3)	2412	27	84720	27	-	-
χ	True	8	(1)	0.12	6	9	1	6	7
			(2)	0.07	5	7	1	5	6
			(3)	0.60	5	6	1	5	6
$\neg\chi$	False	8	(1)	-	-	-	-	-	-
			(2)	12172	21	851894	483465	-	-
			(3)	17836	21	851517	425757	-	-
$\neg\neg\chi$	True	9	(1)	23.10	48	92	10	48	58
			(2)	43.3	15	1556	748	7	9
			(3)	4.84	15	144	69	7	9
F_1	True	8	(1)	-	-	-	-	-	-
			(2)	121.21	35	5697	1894	35	1374
			(3)	41.90	38	1512	338	38	1512
F_2	False	6	(1)	-	-	-	-	-	-
			(2)	22470	26	2689430	1211098	-	-
			(3)	514.00	25	52751	26369	-	-
F_3	True	12	(1)	-	-	-	-	-	-
			(2)	62831	80	790190	271515	80	246895
			(3)	18.3	30	698	207	30	698

D = max depth explored, N = Number of nodes explored, B = Branches created,
DP = Depth of the proof, NP = Nodes in the proof
(1) = Without subsumption and without maximality
(2) = With subsumption and without maximality
(3) = With subsumption and with maximality

Table 2
AutoPSI tested in several conditions.

great improvement to find an efficient heuristic for guessing the right secondarily principal formulas since exploring too many wrong choices might take too long for big formulas. One formula we can not solve right now is $\neg\neg\Phi$, which is not valid but the prover must explore too many branches to terminate in a reasonable amount of time.

Despite not being able to solve $\neg\neg\Phi$, the strategy improves the prover on most of the formulas. The only formula where we have a worst case in execution time is $\neg\chi$, because despite having a smaller search tree, the management of maximal formulas is time consuming and here it is not compensated

enough to improve the execution. This is the same reason why we improve the execution time of $\neg\neg\neg\Phi$ by only two seconds despite dividing the size of the search tree by two. All other formulas are improved by the maximality strategy. In particular, tests done on purely intuitionistic formulas show that maximality is an interesting optimization for our prover.

Since ISCI is a conservative extension of IL, AutoPSI is also an automated theorem prover for IL. However, let us remark that AutoPSI is currently not a competitive tool for IL, even when compared to our old STRIP prover [7]. The inefficiency comes from the use of Topological Beth semantics and its requirement to have rules with secondarily principal formulas. For IL, Kripke semantics is simpler and more efficient since one has the subformula property and various ways to tame the introduction of new labels. AutoPSI is firstly and mainly designed for ISCI, which does not enjoy the subformula property so that Topological Beth semantics is for the moment the easiest way to allow label reuse.

Conclusion and future work

The prover AutoPSI is the first automated prover for the logic ISCI. AutoPSI is based on the system $\mathsf{L}_{\mathsf{ISCI}}$, a labelled multi-conclusioned sequent calculus system, based on the Topological Beth semantics. The prover uses properties of both $\mathsf{L}_{\mathsf{ISCI}}$ and the Topological Beth semantics to enhance its performances. These properties are the regularity of the Topological Beth models, allowing the prover to have a finite search space, the maximality of the proof system, which enables the prover to terminate. Other properties are optimizations, like the restrictions on the identity rules allowed by our proof system, the subsumption of formulas that are not useful in the proof, and the maximality property of our proof system that allows to disregard some formulas that are not maximal in some steps of the proof. The last two optimizations are tested on a test base of several ISCI and IL formulas. The tests demonstrate the usefulness of such optimizations, as well as the difficulties of the prover on the restriction to IL. These difficulties are explained by the design of the proof system, built to deal with the identity operator.

Future works will focus on more optimizations for AutoPSI, mostly by implementing and testing other strategies. One strategy that could be really interesting is a strategy that uses the full maximality property, on every step of the proof and not just some steps. Such a strategy could drastically improve the prover performances.

References

[1] R. Suszko, Abolition of the Fregean axiom, in: Logic Colloquium, 1975, pp. 169–239. Springer.

[2] P. Lukowski, Intuitionistic sentential calculus with identity, Bulletin of the Section of Logic 19 (1990) 92–99.

[3] S. Chlebowski, D. Leszcyńska-Jasion, An Investigation into Intuitionistic Logic with Identity, Bulletin of the Section of Logic 48 (2019) 259–283.

[4] D. Galmiche, M. Gawek, D. Méry, Beth semantics and labelled deduction for intuitionistic sentential calculus with identity, in: 6th International Conference on Formal Structures for Computation and Deduction, FSCD 2021, LIPIcs 195, Buenos Aires, Argentina, 2021, p. 13:1–13:21.

[5] J. Golińska-Pilarek, T. Huuskonen, M. Zawidzki, Tableau-based decision procedure for non-fregean logic of sentential identity, in: 28th Int. Conference on Automated Deduction, CADE 2021, LNAI 12699, 2021, pp. 41–57.

[6] A. Tomczyk, D. Leszcyńska-Jasion, Decidability of Intuitionistic Sentential Logic with Identity via Sequent Calculus, in: 10th International Conference on Non-Classical Logics, Theory and Applications, NCL 2022, volume 358 of *EPTCS*, 2022, pp. 136–149.

[7] D. Larchey-Wendling, D. Méry, D. Galmiche, STRIP: Structural sharing for efficient proof-search, in: First International Joint Conference on Automated Reasoning, IJCAR 2001, LNCS 2083, Siena, Italy, 2001, pp. 696–700.

A. AutoPSI **Session Capture**

```
bhornbec@hapi:~/IdeaProjects/ISCIProver/out/artifacts/ISCIProver_jar$ java -jar ISCIProver.jar
subsumption
Subsumption off
strategy
Current strategy : Partial Delta Maximality
Choose new Strategy
1: Partial Delta Maximality
2: New Label First
3: First Formula
2
Strategy changed
verbose
verbose mode on
prove ((((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))))
(((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))) is false
ttl 6
Execution in 38.0ms
621 premises created
Max depth explored 13
Number of branches explored 306
------------------------------------------------------------
subsumption
Subsumption on
prove ((((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))))
(((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))) is false
ttl 6
Execution in 45.0ms
539 premises created
Max depth explored 13
Number of branches explored 265
------------------------------------------------------------
strategy
Current strategy : New Label First
Choose new Strategy
1: Partial Delta Maximality
2: New Label First
3: First Formula
1
Strategy changed
prove ((((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))))
(((q=p)->(p->r))=((p->(p=p))=p))->(((r&p)=(p=p))|((p&p)|(q->F))) is false
ttl 6
Execution in 7.0ms
57 premises created
Max depth explored 13
Number of branches explored 26
------------------------------------------------------------
```