Implementing Intermediate Logics

Bastiaan Haaksema[1,2], Jens Otten[3,4] and Revantha Ramanayake[1,5]

[1]Bernoulli Institute, University of Groningen, The Netherlands
[2]Department of Information and Computing Sciences, Utrecht University, The Netherlands
[3]Department of Informatics, University of Oslo, Norway
[4]Potassco Solutions, Germany
[5]CogniGron, University of Groningen, The Netherlands

Abstract

We present automated theorem provers implementing systems for intermediate logics, in the propositional and first-order setting. They use an axiomatic embedding into intuitionistic logic based on cut-restricted sequent calculi. All provers are evaluated on a large benchmark set of propositional and first-order formulas.

Keywords

non-classical logics, intermediate logics, automated theorem provers, cut-restriction

1. Introduction

Intermediate logics lie between intuitionistic logic and classical logic in terms of subset inclusion, in the propositional or first-order setting. Intermediate logics offer a more nuanced approach to the usual 2-valued classical logic and this motivates their study in logic, computer science, and artificial intelligence. While many theorem provers exist for classical logic and several for intuitionistic logic, only very few are available for intermediate logics. Our aim is to address this gap by presenting provers for intermediate logic—propositional and first-order—in a systematic manner.

On the proof-theoretic side, it is well known that most propositional intermediate logics lack a cut-free sequent calculus. This is a formidable obstacle for automated theorem proving, and meta-theoretic investigations, since cut-freeness is the typical route towards a proof calculus with the subformula property, and the latter property is crucial for pruning in backward proof search. Indeed, recall the situation that arises with a Hilbert proof calculus where it is unclear when and on what formula the rule of *modus ponens* needs to be applied backwards. Ciabattoni et al. [1] present a general solution via cut-free hypersequent calculi for extensions of intuitionistic propositional logic (**IPL**) with axioms up to $\mathcal{P}_3'$ in the substructural hierarchy. Although the hypersequent calculus is a natural generalisation of the sequent calculus (use a multiset of sequents instead of a single sequent), from the perspective of automated theorem proving it is much more complex to implement, and the hypersequent calculus formalism is much less well-known outside the structural proof theory community.

A new solution is proposed by Ciabattoni et al. [2, 3]: sound and complete *sequent calculi* for propositional intermediate (and substructural) logics by permitting restricted cuts (as mentioned above, without a restriction on the cuts, the backward proof search space is simply too large). Specifically, the cut-formulas are restricted to instantiations of the axioms with conjunctions of subformulas of the end sequent. In the case of intermediate logics, the restricted cuts can be traded for a cut-free proof in the intuitionistic calculus with axiom instances added to the antecedent, making use of the deduction theorem in the latter. Consequently, the intermediate logics embed into intuitionistic logic.

Our focus is on the implementations of the theory described above, and also for first-order intermediate logics that are obtained through the addition of quantification rules. After describing the theoretical foundation in Section 2, we present our implementations in Section 3 and evaluate them in Section 4. We conclude with a summary, outlook and related research in Section 5.

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

†The third author acknowledges the financial support of CogniGron, the Ubbo Emmius Funds, and FWF project P33548.

b.haaksema@student.rug.nl (B. Haaksema); jeotten@jens-otten.de (J. Otten); d.r.s.ramanayake@rug.nl (R. Ramanayake)
https://jens-otten.de/ (J. Otten); https://www.rug.nl/staff/d.r.s.ramanayake/ (R. Ramanayake)
0009-0001-0311-1553 (B. Haaksema); 0000-0002-4331-8698 (J. Otten); 0000-0002-7940-9065 (R. Ramanayake)

2. Intermediate Logics

2.1. Preliminaries

Propositional formulas are defined inductively from propositional atoms $p, q, \ldots$ and constants $\bot$ and $\top$ using the connectives $\wedge$, $\vee$, $\to$, and $\neg$. In the first-order setting, the propositional atoms are replaced by predicates on terms built from variables and function symbols, and the language is extended with first-order quantifiers. A term is a variable or $f(t_1, \ldots, t_n)$ for a n-ary function symbol f and terms $t_1, \ldots, t_n$. First-order formulas are defined inductively as the constants $\bot$ and $\top$, $P(t_1, \ldots, t_n)$ for an n-ary predicate symbol P and terms $t_1, \ldots, t_n$, and $A \wedge B$, $A \vee B$, $A \to B$, $\neg A$, $\forall x.A$ and $\exists x.A$ for formulas A and B. Free variables in a formula are those not in the scope of a quantifier. Also, $A(t/x)$ denotes the formula obtained by uniformly substituting the term t for all occurrences of the variable x that are free.

Throughout, we identify a logic with the set of its theorems. An axiomatic extension $L + \mathcal{A}$ is obtained by extending the base logic L with every instantiation of the atomic formulas of the axiom schema $\mathcal{A}$ by arbitrary formulas, and closing under the axioms and rules of the proof calculus.

The Hilbert proof calculus consists of axioms and rules that directly manipulate the logical formulas. There are many equivalent variants for **IPL** e.g., [4, Section 6]. A Hilbert calculus for first-order intuitionistic logic (**IL**) is obtained by adding the following to **IPL**. Here x is not a free variable in C.

$$\frac{C \to A(y/x)}{C \to \forall x.A} \qquad \frac{A(y/x) \to C}{\exists x.A \to C} \qquad \forall x.A \to A(t/x) \qquad A(t/x) \to \exists x.A$$

For economy of notation, we use **IPL** to denote the set of theorems of intuitionistic propositional logic and also its Hilbert proof calculus, and similarly for intuitionistic first-order logic **IL**.

A sequent calculus is a type of proof calculus that manipulates sequents of the form $A_1, \ldots, A_m \Rightarrow A_{m+1} \ldots, A_{m+n}$ where $A_1, \ldots, A_{m+n}$ are formulas. The intended interpretation of a sequent is the formula $A_1 \wedge \ldots \wedge A_m \to A_{m+1} \vee \ldots \vee A_{m+n}$. When required, $\top$ and $\bot$ serve as the identity element for conjunction and disjunction respectively. A sequent (calculus) is single-succedent if at most a single formula is permitted on the right-hand side of the sequent, i.e. $n \in \{0, 1\}$, else it is multi-succedent. Roughly speaking, the meta-level implication and conjunction/disjunction provided by $\Rightarrow$ and comma permit reasoning *inside* the formula (fixed depth deep inference), and this is what enables the definition of inference rules with nice properties that aid automated theorem proving and meta-theoretic investigations.

The theoretical foundations can be described using any reasonable cut-free sequent calculus for intuitionistic logic. For the sake of concreteness, we use the well-known sequent calculus **LJ** presented by Gentzen [5]. In contrast, the implementations employ variants of this system that are specifically optimized for proof searching.

2.2. Cut-restricted Sequent Calculi

In this work, we consider the propositional axiomatic extensions of **IPL** listed further below.

The soundness and completeness with respect to sequent calculi with restricted cuts was established in [3]. As the reader may be unfamiliar with cut-restriction, let us sketch briefly how the completeness result was obtained there: any theorem of the logic under consideration has a cut-free hypersequent proof [1] with proper hypersequent structural rules e.g., the communication rule (com) in Gödel logic which rearranges the contents of two components in the hypersequent. A hypersequent proof without proper structural rules is obtained by repeatedly eliminating bottom-most hypersequent structural rules. The latter is achieved by accepting an additional formula (slightly more than a subformula) in each active component. These additional formulas are eliminated at the bottom of the proof via a cut on an instance of a proper axiom of the logic. The cut-restricted sequent proof can now be read off the hypersequent proof since the latter contains no proper hypersequent structural rules.

1. Jankov logic: **IPL** $+ \neg A \vee \neg\neg A$. The sequent calculus $\mathbf{LJ}^J$ extends **LJ** with the cut-rule restricted to instances of the axiom schema $\neg A \vee \neg\neg A$. Specifically, in a proof of $\Rightarrow F$, the atomic formula A in the axiom schema can be replaced by any conjunction of subformulas of F.
2. Gödel logic: **IPL** $+ (A \to B) \vee (B \to A)$. The sequent calculus $\mathbf{LJ}^G$ extends **LJ** with the cut-rule restricted to instances of the axiom schema $(A \to B) \vee (B \to A)$ so in a proof of $\Rightarrow F$, the atomic formulas A and B are replaced by any conjunction of subformulas of F.
   While the above axiom schema for Gödel logic is the standard one, we will actually use the equivalent axiomatisation **IPL** $+ (A \to B) \vee ((A \to B) \to A) + \neg A \vee \neg\neg A$. As observed in [3], this allows us to restrict cut-formulas to a much smaller set, namely instances of the axiom schema where the atomic formulas are replaced by propositional atoms from F.

First-order axiom schemas. In the first-order setting, the above axiom schemas are written as *universal sentences*. For example, $\forall \bar{x}(\neg A \vee \neg\neg A)$ and $\forall \bar{x}((A \to B) \vee (B \to A))$.

Let $\mathbf{FLJ}^J$ and $\mathbf{FLJ}^G$ denote first-order the sequent calculi obtained from $\mathbf{LJ}^J$ and $\mathbf{LJ}^G$ by adding the usual Gentzen first-order quantifiers. Here, the *eigenvariable* y must not occur in the conclusion of the (R$\forall$) and (L$\exists$) rules.

$$\frac{A(t/x), \Gamma \Rightarrow C}{\forall x A, \Gamma \Rightarrow C}\,(\mathrm{L}\forall) \qquad \frac{\Gamma \Rightarrow A(y/x)}{\Gamma \Rightarrow \forall x A}\,(\mathrm{R}\forall) \qquad \frac{A(y/x), \Gamma \Rightarrow C}{\exists x A, \Gamma \Rightarrow C}\,(\mathrm{L}\exists) \qquad \frac{\Gamma \Rightarrow A(t/x)}{\Gamma \Rightarrow \exists x A}\,(\mathrm{R}\exists)$$

We observe that $\mathbf{FLJ}^J$ and $\mathbf{FLJ}^G$ are sound and complete for the corresponding cut-free hypersequent calculi $\mathbf{HLJ}^J$ and $\mathbf{HLJ}^G$ by a straightforward extension of the argument in [3]. The latter hypersequent calculi consist of proper hypersequent structural rules added to the base calculus **HLJ** for **IL**. Soundness is straightforward. For completeness, we extend the transformation for propositional logics in [3] that was sketched above. In the case of $\mathbf{HLJ}^G$, an instance of (com) in the hypersequent proof is replaced with a formula of the form $\wedge\Gamma \to \wedge\Gamma'$ that is added to the antecedent of the of the active component. Here $\wedge\Gamma$ is the conjunction of all formulas in Γ. The remaining rules in the cut-free hypersequent proof are faithfully simulated in the sequent proof that is ultimately obtained. It remains to simulate the quantifier rules in $\mathbf{HLJ}^G$ with the quantifier rules in $\mathbf{LJ}^G$ and verify by inspection that the eigenvariable condition for the former implies it for the latter even in the presence of the added formulas.

It still remains to clarify the relationship between the first-order hypersequent calculus and the Hilbert calculus[1]. Consider the following Hilbert calculus rule with the condition that x is not free in C.

$$\frac{\forall x(C \vee A(y/x))}{C \vee \forall x A}\,(\mathrm{QSR})$$

Armed with this rule, for each rule in the hypersequent calculus, there is a derivation in the Hilbert calculus of the conclusion from the premises under the standard formula translation. It follows that $\mathbf{FLJ}^J$ is sound for **IL** $+ \forall \bar{x}(\neg A \vee \neg\neg A) +$ (QSR). Completeness is immediate since $\mathbf{HLJ}^J$ has cut-elimination. Similarly, $\mathbf{FLJ}^G$ is sound and complete for **IL** $+ \forall \bar{x}(A \to B \vee B \to A) +$ (QSR).

In certain cases, the (QSR) rule may be replaced by an axiom schema. For example, it is easy to see that **IL**$+\forall \bar{x}(A \to B \vee B \to A) +$(QSR) is equivalent to **IL**$+\forall \bar{x}(A \to B \vee B \to A) + \forall \bar{z}\forall x(C \vee A(y/x)) \to \forall \bar{z}(C \vee \forall \bar{x} A)$ where x is not free in C in the latter axiom.

2.3. Cut-restricted Sequent Calculi and Embeddings

We have already noted that the sequent calculi obtained in [3] restrict the cut formulas to certain axiom instances that depend on the formula F that is being proved. Specifically, the set of cut formulas that are required in a proof of $\Rightarrow F$ is defined by the function below, with the set $\mathcal{A}$ of axiom schemas as parameter, and with the help of an auxiliary function ψ.

$$F \;\mapsto\; \{A \mid A \in \psi(\mathcal{A}, F)\}$$

[1]The third author thanks Timo Lang for a helpful discussion on this topic.

Ciabattoni et al. [3] identify several candidates for ψ.

- The set-bounding function $\psi_s(\mathcal{A}, F)$ contains all instances of formulas in $\mathcal{A}$ whose atomic formulas are substituted by non-repeating conjunctions of subformulas of F.
- The formula-bounding function $\psi_f(\mathcal{A}, F)$ contains all instances of formulas in $\mathcal{A}$ whose atomic formulas have been substituted by subformulas of F.
- The variable-bounding function $\psi_v(\mathcal{A}, F)$ contains all instances of formulas in $\mathcal{A}$ whose atomic formulas have been substituted by atoms in F.

We have omitted the multiset-bounding function, as it is not relevant for intermediate logics. Observe that the set-bounding function has as image a set whose size is exponential in the size of F.

Notice that $\psi_v(\mathcal{A}, F) \subset \psi_f(\mathcal{A}, F) \subset \psi_s(\mathcal{A}, F)$ for any F containing two distinct subformulas. In certain cases including Jankov logic and Gödel logic, it is possible to identify an axiomatisation that supports the preferred variable-bounding function. As Ciabattoni et al. [3] observe: for axiomatisations that satisfy the $\{\wedge\}$-propagation property, the formula-bounding function can be used. Also, for axiomatisations that satisfy the $\{\wedge, \vee, \rightarrow\}$-propagation property, the variable-bounding function can be used.

The set of axiom instances (and hence cut formulas) that are required in a proof of F in Jankov logic and Gödel logic are explicitly described below. It is precisely this variable-bounding function that we implement in the embedding-preprocessing step of the provers.

Jankov logic	$\{\neg A \vee \neg\neg A \mid A \mapsto \text{atoms in } F\}$
Gödel logic	$\{(A \rightarrow B) \vee ((A \rightarrow B) \rightarrow A) \mid A, B \mapsto \text{atoms in } F\} \cup$ $\{\neg A \vee \neg\neg A \mid A \mapsto \text{atoms in } F\}$

From Restricted Cuts to Embedding into Intuitionistic Logic

Observe that a proof of $\Rightarrow F$ in the intuitionistic calculus with cuts from a finite set Ω (in the present setting, this represents the image of the bounding function) can be transformed to a proof of $\wedge\Omega \Rightarrow F$; simply replace each left premise $\Gamma \Rightarrow A$ of a cut instance ($A \in \Omega$) with the trivial proof in **LJ**, where (L$\wedge$)* denotes multiple applications of the left conjunction rule.

$$\dfrac{\dfrac{}{A, \Gamma \Rightarrow A}}{\wedge\Omega, \Gamma \Rightarrow A}\,(\mathrm{L}\wedge)^*$$

Now proceed downwards, applying the obvious weakenings and contractions as required; the end sequent is then transformed to $\wedge\Omega \Rightarrow F$. By cut-elimination in the intuitionistic calculus, we obtain a cut-free proof of the latter sequent that witnesses an embedding of the intermediate logic into intuitionistic logic. What this means is that the prover can now conduct backward proof search in the setting of a cut-free intuitionistic proof of $\wedge\Omega \Rightarrow F$. This is the embedding perspective [3] that our provers adopt.

We remark that the number of subformulas of F is bounded by the size $|F|$ of F, and the number of atomic formulas in each of the Gödel axioms listed above is ≤ 2, and the Jankov axiom has just a single atomic formula. Hence the set-bounding function yields an exponential embedding of Gödel logic into intuitionistic logic, and the variable-bounding function that we actually implement in the prover yields a linear and quadratic embedding of Jankov and Gödel logic, respectively.

3. Implementations

3.1. SuperJ Prover

SuperJ is an automated theorem prover written in Haskell that supports many intermediate propositional logics via an embedding-preprocessing step followed by intuitionistic proof search.

$$\frac{A, \Gamma \Rightarrow B}{\Gamma \Rightarrow \Delta, A \to B}\ (\mathsf{R}\to) \qquad \frac{A, B, \Gamma \Rightarrow \Delta}{A, A \to B, \Gamma \Rightarrow \Delta}\ (\mathsf{mp}) \qquad \frac{A \to (B \to C), \Gamma \Rightarrow \Delta}{(A \wedge B) \to C, \Gamma \Rightarrow \Delta}\ (\mathsf{L}{\to}{\wedge})$$

$$\frac{A \to p, B \to p, p \to C, \Gamma \Rightarrow \Delta}{(A \vee B) \to C, \Gamma \Rightarrow \Delta}\ (\mathsf{L}{\to}{\vee}) \quad \text{fresh atomic } p$$

$$\frac{A, p \to C, B \to p, \Gamma \Rightarrow p \qquad C, \Gamma \Rightarrow \Delta}{(A \to B) \to C, \Gamma \Rightarrow \Delta}\ (\mathsf{L}{\to}{\to}) \quad \text{fresh atomic } p$$

Figure 1: Implication rules of the intuitionistic sequent calculus.

SuperJ automatically selects one of the three bounding functions mentioned earlier based on the propagation properties in the embedding step. Besides supporting arbitrary axiomatisations, the prover explicitly supports Classical Propositional Logic (**CPL**), **IPL**, Jankov logic and Gödel logic. For **IPL** the embedding step is omitted, and for **CPL** the prover immediately switches to classical proof search. The two intermediate logics are supported through the variable-bounded axiomatisations as noted in the previous section.

The intuitionistic proof search procedure of SuperJ is based on that of Avellone et al. [6]. It uses a contraction-free multi-succedent intuitionistic calculus equivalent to **LJ** and is decidable in $O(n \log n)$-Space [7]. This calculus can be seen as a refinement of Dyckhoff's **LJT*** [8], it includes rules for negation and uses fresh propositional variables in two of the left implication rules (as shown in Figure 1) to restrict the size of proofs. As in the original procedure by Avellone et al., SuperJ classifies formulas in the sequent into six groups according to their behavior with respect to branching and backtracking, which can be considered a naive form of focusing [9].

The most important optimization follows from the addition of boolean simplification and replacement rules [6, 10, 11]. The atomic replacement rules below are adjusted to sequent calculus notation, where $[A/B]$ denotes the uniform substitution of A for all occurrences of B. This implemented optimization works well for our intermediate logics, since even though the embedding step may introduce new formulas into the sequent, these are all composed of subformulas of the end sequent.

$$\frac{(\Gamma \Rightarrow \Delta)[\top/p]}{p, \Gamma \Rightarrow \Delta}\ (\text{L rep}) \qquad \frac{(\Gamma \Rightarrow \Delta)[\bot/p]}{\neg p, \Gamma \Rightarrow \Delta}\ (\text{L}\neg \text{ rep})$$

When possible, SuperJ reverts to classical proof search through the sequent calculi **LK*** [12] for classical logic. This happens whenever the succedent of the sequent is empty. Furthermore, the classical provability of the sequent is checked before attempting to apply a left non-invertible left rule, which allows pruning of the search space when false. Finally, for non-invertible right rules, backtracking can be avoided in the case of a singleton succedent.

3.2. ileanSeP-im and ileanTAP-im

The theorem prover ileanSeP-im is an axiomatic extension of the intuitionistic sequent prover ileanSeP for Jankov and Gödel first-order logics. ileanSeP is a compact Prolog implementation of the single-succedent intuitionistic sequent calculus. Similar to tableau calculi, it uses a bottom-up proof search, free variables and a dynamic skolemization to deal with quantifiers. Together with the occurs check of term unification, this ensures that the eigenvariable condition is respected.

The theorem prover ileanTAP-im is an axiomatic extension of the intuitionistic tableau prover ileanTAP for Jankov and Gödel first-order logics. ileanTAP [13] is a compact Prolog implementation of a prefixed tableau calculus, similar to Fitting [14]. It extends the classical calculus [15] by adding prefixes to capture the Kripke semantics of intuitionistic logic, uses free prefix variables [16] and extends skolemization to prefix constants. First, ileanTAP-im performs a classical proof search collecting prefixes of literals that close branches. If this search succeeds, a prefix unification is used to unify these prefixes.

Table 1
The (prefixed) non-clausal matrix for intuitionistic logic.

type	$F^{pol} : p$	$M(F^{pol} : p)$
atomic	$A^0 : p$	$\{\{A^0 : p\underline{a^*}\}\}$
α	$(G \wedge H)^1 : p$	$\{\{M(G^1 : p)\}\}, \{\{M(H^1 : p)\}\}$
	$(G \vee H)^0 : p$	$\{\{M(G^0 : p)\}\}, \{\{M(H^0 : p)\}\}$
	$(G \to H)^0 : p$	$\{\{M(G^1 : p\underline{a^*})\}\}, \{\{M(H^0 : p\underline{a^*})\}\}$
β	$(G \wedge H)^0 : p$	$\{\{M(G^0 : p), M(H^0 : p)\}\}$
	$(G \vee H)^1 : p$	$\{\{M(G^1 : p), M(H^1 : p)\}\}$
	$(G \to H)^1 : p$	$\{\{M(G^0 : p\underline{V^*}), M(H^1 : p\underline{V^*})\}\}$

type	$F^{pol} : p$	$M(F^{pol} : p)$
atomic	$A^1 : p$	$\{\{A^1 : p\underline{V^*}\}\}$
α	$(\neg G)^0 : p$	$M(G^1 : p\underline{a^*})$
	$(\neg G)^1 : p$	$M(G^0 : p\underline{V^*})$
γ	$(\forall x G)^1 : p$	$M(G[x\backslash x^*]^1 : p\underline{V^*})$
	$(\exists x G)^0 : p$	$M(G[x\backslash x^*]^0 : p)$
δ	$(\forall x G)^0 : p$	$M(G[x\backslash t^*]^0 : p\underline{a^*})$
	$(\exists x G)^1 : p$	$M(G[x\backslash t^*]^1 : p)$

3.3. nanoCoP-im

The automated theorem prover nanoCoP-im is an axiomatic extension of the intuitionisitic non-clausal connection prover nanoCoP-i for the Jankov and Gödel first-order logics. nanoCoP-i is a compact Prolog implementation of the non-clausal connection calculus for first-order *intuitionistic* logic (with equality) [17, 18] and an extension of the classical prover nanoCoP [19, 20]. It is based on a prefixed non-clausal connection calculus [17]. In contrast to sequent and tableau calculi, which are *connective-driven*, connection calculi use a *connection*-driven search strategy. A *connection* is a set $\{A_1{}^0, A_2{}^1\}$ of literals with the same predicate symbol but different polarities.

The *non-clausal connection calculus* works on *non-clausal* matrices, where a matrix M is a set of clauses and a clause C is a set of literals L and (sub)matrices. It represents a formula in negation normal form. A *prefix* is a string consisting of variables (V) and constants (a) and assigned to each literal.

For a formula F, polarity $pol \in \{0, 1\}$ and prefix p, the *intuitionistic non-clausal matrix* $M(F^{pol})$ of F^{pol} is defined inductively according to Table 1. x^* is a new term variable, t^* is the Skolem term $f^*(x_1, \ldots, x_n)$, V^* is a new prefix variable, a^* is the prefix constant of the form $f^*(x_1, \ldots, x_n)$, f^* is a new function symbol and $x_1, \ldots, x_n$ are all free term and prefix variables in the corresponding formula $F^{pol} : p$. The *intuitionistic non-clausal matrix* $M^i(F)$ of F is the matrix $M(F^0 : \varepsilon)$. A *term substitution* σ_T assigns terms to variables, a *prefix substitution* σ_P assigns strings to prefix variables (and is calculated by a *prefix unification*). For intuitionistic logic, a connection $\{A_1^0 : p_1, A_2^1 : p_2\}$ is *σ-complementary* iff $\sigma_T(A_1) = \sigma_T(A_2)$ and $\sigma_P(p_1) = \sigma_P(p_2)$ for a combined substitution $\sigma = (\sigma_T, \sigma_P)$.

The *non-clausal connection calculus* for intuitionistic logic [17] is given in Figure 2. An *intuitionistic connection proof* for F is a derivation of $\varepsilon, M^i(F), \varepsilon$. Compared to the formal *clausal* connection calculus [21, 22], the extension rule is generalized and a *decomposition rule* is added [23, 19].

First, nanoCoP-im performs a classical proof search, in which the prefixes of each connection are collected. If the search succeeds the prefixes of the literals in each connection are unified. Additional optimization techniques are regularity, lemmata, restricted backtracking and strategy scheduling [24, 18].

Axiom (A) $\dfrac{}{\{\}, M, \mathit{Path}}$ *Start (S)* $\dfrac{C_2, M, \{\}}{\varepsilon,\ M,\ \varepsilon}$ and C_2 is copy of $C_1 \in M$

Reduction (R) $\dfrac{C, M, \mathit{Path} \cup \{L_2 : \underline{p_2}\}}{C \cup \{L_1 : \underline{p_1}\}, M, \mathit{Path} \cup \{L_2 : \underline{p_2}\}}$ and $\{L_1 : \underline{p_1}, L_2 : \underline{p_2}\}$ is σ-complementary

Extension (E) $\dfrac{C_3, M[C_1 \backslash C_2], \mathit{Path} \cup \{L_1 : \underline{p_1}\} \quad C, M, \mathit{Path}}{C \cup \{L_1 : \underline{p_1}\}, M, \mathit{Path}}$ $C_3 := \beta\text{-}clause_{L_2}(C_2)$, C_2 is copy of C_1, C_1 is e-clause of M wrt. $\mathit{Path} \cup \{L_1 : \underline{p_1}\}$, C_2 contains $L_2 : \underline{p_2}$, $\{L_1 : \underline{p_1}, L_2 : \underline{p_2}\}$ is σ-complementary

Decomposition (D) $\dfrac{C \cup C_1, M, \mathit{Path}}{C \cup \{M_1\}, M, \mathit{Path}}$ and $C_1 \in M_1$

Figure 2: The non-clausal connection calculus for intuitionistic logic.

4. Experimental Evaluation

4.1. Benchmark Problems

At present, sets of formulas for testing automated theorem provers for intermediate logic are not available. As the syntax of intermediate logic is the same as classical and intuitionistic logic, we can use existing benchmark formulas of these logics. As intermediate logics are an extension of intuitionistic logic, we decided to use the ILTP problem library for intuitionistic logic [25].

Version 1.1.2 of the ILTP problem library contains 274 propositional and 2550 first-order formulas with status and difficulty rating information. The problems are in TPTP syntax and divided into 24 categories. While the propositional formulas belong mainly to the intuitionistic syntactic category (SYJ), the first-order formulas are taken from a wide range of domains, from general algebra (ALG), computing (COM), set (SET) and number (NUM) theory to software creation (SWC) and verification (SWV).

4.2. Propositional Logic

The SuperJ prover described in Section 3 was evaluated on all 274 propositional problems of the ILTP library v1.1.2. The test were conducted on a 3.6 GHz Ryzen system with 32 GB of RAM running Ubuntu 23.10 with kernel version 5.15. Running time was restricted to 60 seconds of CPU time per individual problem. Table 2 shows the number of problems solved within the time limit, and the required time in seconds, best results for each logic are marked bold.

Performance results of intuitRIL have also been collected for comparison. This intermediate propositional logic prover, due to Fiorentini and Ferrari [26], is the only automated theorem prover known to us that supports the same selection of intermediate propositional logics. It was obtained through modification of an existing SAT-based theorem prover for intuitionistic propositional logic. Their methods have similar theoretical foundations, though more akin to using restricted cuts, as opposed to our embedding approach.

intuitRIL outperforms (or matches) SuperJ in intuitionistic propositional logic, particularly so for domains SYJ206 and SYJ209. Except for SYJ208, SuperJ is only ever faster with a minimal constant factor, this might be due to there being a slightly smaller preprocessing overhead compared to intuitRIL that uses a clausification procedure.

For Jankov logic, intuitRIL solved the same number of problems within almost the same amount of time as for **IPL**. There is no real slowdown by the addition of the axiom instantiations for this prover. The SuperJ prover is more sensitive to the addition of the axiom instantiations, e.g., it finds a proof earlier for problems in SYJ209 but is slower in SYJ201-SYJ205, SYJ211 and SYJ212.

Table 2
Results on the propositional problems of the ILTP library

		SuperJ (**IPL**)		intuitRIL (**IPL**)		SuperJ (Jan)		intuitRIL (Jan)		SuperJ (Göd)		intuitRIL (Göd)	
Domain	Total	Solved	Time	Solved	Time	Solved	Time	Solved	Time	Solved	Time	Solved	Time
LCL	2	**2**	**0.023**	2	0.024	**2**	**0.023**	2	0.024	**2**	**0.023**	2	0.024
SYJ1	12	**12**	**0.138**	12	0.145	**12**	**0.138**	12	0.144	8	0.092	**12**	**0.144**
SYJ201	20	20	6.533	**20**	**2.765**	14	53.583	**20**	**2.754**	1	0.022	**20**	**2.754**
SYJ202	20	9	28.834	**10**	**14.282**	4	15.417	**10**	**14.322**	1	0.012	**10**	**14.352**
SYJ203	20	**20**	**0.232**	20	0.242	20	5.162	**20**	**0.243**	3	1.724	**20**	**0.244**
SYJ204	20	**20**	**0.234**	20	0.241	20	15.192	**20**	**0.244**	3	0.326	**20**	**0.242**
SYJ205	20	**20**	**0.232**	20	0.242	12	70.909	**20**	**0.248**	0	—	**20**	**0.242**
SYJ206	20	11	25.887	**20**	**0.240**	11	29.026	**20**	**0.239**	4	3.946	**20**	**0.243**
SYJ207	20	20	0.674	**20**	**0.613**	**20**	**0.231**	20	0.612	**20**	**0.496**	20	1.054
SYJ208	20	**20**	**0.856**	20	2.296	**20**	**0.771**	20	2.306	10	1.446	**17**	**165.925**
SYJ209	20	8	6.862	**20**	**0.243**	9	56.864	**20**	**0.240**	4	23.207	**20**	**0.264**
SYJ210	20	**20**	**0.230**	20	0.241	20	16.641	**20**	**0.240**	4	4.636	**20**	**0.313**
SYJ211	20	20	97.895	**20**	**0.241**	12	45.389	**20**	**0.241**	1	3.132	**20**	**97.985**
SYJ212	20	20	0.720	**20**	**0.242**	12	48.558	**20**	**0.241**	5	19.868	**20**	**0.302**
SYN	20	20	11.423	**20**	**0.240**	19	0.217	**20**	**0.239**	19	0.218	**20**	**0.252**

Finally, for Gödel logic, intuitRIL again managed to solve most of the problems within the time limit. However, compared to itself for other logics, it is slower for domains SYJ208 and SYJ211. The performance of SuperJ on the same logic is below that of intuitRIL, except for domains LCL and SYJ207. Perhaps this is not so surprising, considering the axiomatisation of Gödel logic contains (nested) implications, a known weak point for sequent based provers such as SuperJ.

4.3. First-Order Logic

The automated theorem provers for first-order intermediate logic described in Section 3 were evaluated on the problems of the ILTP library. All test were conducted on a 2.0 GHz Xeon server with 64 GB of RAM running Linux Mint with kernel version 3.10 and ECLiPSe Prolog 5.10. Table 3 shows the results of the evaluation on all 2550 first-order problems of the ILTP library v1.1.2 [25] for a CPU time limit of 10 seconds. Included are the provers ileanSeP-im 1.0, ileanTAP-im 1.17 and nanoCoP-im 2.0. Each of these implementations were tested for intuitionistic logic ("**IL**"), Jankov logic ("Jan") and Gödel logic ("Göd"). Furthermore, the prover leanCoP 2.2 for classical first-order logic was included, which provides an upper limit for the number of problems that can be proved by one of the provers for intermediate logic.

Table 3
Results on the first-order problems of the ILTP library

	— ileanSeP-im —			— ileanTAP-im —			— nanoCoP-im —			leanCoP 2.2
Logic	**IL**	Jan	Göd	**IL**	Jan	Göd	**IL**	Jan	Göd	Classical
proved	298	238	222	310	196	163	788	750	604	1064
0 to 1sec.	266	214	184	305	190	161	695	602	496	936
1 to 10sec.	32	24	38	5	6	2	93	148	108	128
refuted	4	0	0	4	0	0	88	60	33	28
error	35	53	7	54	323	234	3	3	3	0

nanoCoP-im proves the most problems for each of the non-classical logics. ileanSeP-im proves slightly less problems than ileanTAP-im for **IL**, but more problems than ileanTAP-im for Jankov and Gödel logic. nanoCoP-im also refutes the largest number of problems for all three logics.

All three provers prove less problems for Jankov logic than for **IL** and less problems for Gödel logic than for Jankov logic. This is explained by the overhead in the search space caused by the additional intermediate axioms that are added to the formulas. In general, the problems proved in Gödel logic are a subset of the problems proved in Jankov logic, which are again a subset of the problems proved in **IL**. Table 4 shows the few exceptions where a problem was proved in Jankov (or Gödel) logic but not in **IL**, or proved in Gödel logic but not in Jankov logic. The entries show the CPU time in seconds necessary to prove a problem or (in parentheses) to refute it. These problems are from the domains Set Theory (SET), Software Verification (SWV) and Syntactic (SYN). Given a larger CPU time limit, the problems in the SET and SWV domains can be proved in **IL** by nanoCoP-i or Slakje [18], i.e., they are valid in **IL**.

Table 4
Detailed results for selected problems of the ILTP library

	— ileanSeP-im —			— ileanTAP-im —			— nanoCoP-im —			leanCoP 2.2
Logic	**IL**	Jan	Göd	**IL**	Jan	Göd	**IL**	Jan	Göd	Classical
SET095+4	–	–	–	–	–	–	–	7.5	–	0.6
SET638+3	–	–	–	–	–	–	–	7.9	–	0.1
SWV181+1	–	–	–	–	–	–	–	9.7	–	0.3
SWV188+1	–	–	–	–	–	–	–	8.9	–	–
SWV189+1	–	–	–	–	–	–	–	9.3	–	2.1
SYN081+1	–	–	–	–	–	–	–	0.2	0.2	0.1
SYN416+1	(0.1)	–	0.1	(0.1)	–	0.1	(0.1)	(0.1)	0.1	0.1

5. Conclusion

We implemented four provers for Jankov and Gödel logic via proof search for intuitionistic logic and bounding functions: SuperJ for the propositional Jankov and Gödel logics and ileanSeP-im, ileanTAP-im and nanoCoP-im for the first-order Jankov and Gödel logics. The SuperJ implementation is available at https://github.com/bhaaksema/superintuition, the ileanSeP-im, ileanTAP-im and nanoCoP-im implementations are available at https://leancop.de/imed/. While bounding functions could be added to any existing intuitionistic theorem prover, SuperJ is a new prover aimed at facilitating quick experimentation with heuristics. To the best of our knowledge, ileanSeP-im, ileanTAP-im and nanoCoP-im are the first provers for *first-order* Jankov and *first-order* Gödel logic.

Our tests have shown that very few problems in the ILTP library which are valid in Jankov or Gödel logic are not valid in intuitionistic logic. Therefore, it seems advisable to build a collection of such problems for future benchmarking, possibly generated by (forward) proofs in the sequent calculus.

The methodology applies to other intermediate logics, e.g., Scott's and Kreisel-Putnam logic axiomatized over **IPL** respectively by $((\neg\neg A \to A) \to (A \vee \neg A)) \to (\neg\neg A \vee \neg A)$ and $(\neg A \to (B \vee C)) \to ((\neg A \to B) \vee (\neg A \to C))$. It would also be interesting to extend the methodology to substructural logics.

Fiorentini and Ferrari [26] give a propositional intermediate logic prover intuitRIL that modularly extends a SAT-based prover for **IPL**. Fiorino [27, 28] present duplication-free tableau calculi for Gödel logic and three other propositional intermediate logics, including Jankov logic. Kuznets and Lellmann [29] give semantically inspired constructions of nested sequent calculi for propositional intermediate logics including Gödel logic, and a prototype proof search implementation was also presented.

For classical logic, many state-of-the-art theorem provers use heuristics that select a subset of appropriate axioms in a preprocessing step before the actual proof search [30]. As our approach adds many axioms to the formula to be proven, integrating such heuristics will likely improve the performance of our provers. This will be part of future work, along with further optimization and evaluations of the provers.

References

[1] A. Ciabattoni, N. Galatos, K. Terui, From axioms to analytic rules in nonclassical logics, in: 2008 23rd Annual IEEE Symposium on Logic in Computer Science, 2008, pp. 229–240. doi:10.1109/LICS.2008.39.

[2] A. Ciabattoni, T. Lang, R. Ramanayake, Bounded sequent calculi for non-classical logics via hypersequents, in: TABLEAUX 2019, Springer, 2019, pp. 94–110. doi:10.1007/978-3-030-29026-9_6.

[3] A. Ciabattoni, T. Lang, R. Ramanayake, Bounded-analytic sequent calculi and embeddings for hypersequent logics, The Journal of Symbolic Logic 86 (2021) 635–668. doi:10.1017/jsl.2021.42.

[4] A. Chagrov, M. Zakharyaschev, Modal logic, volume 35 of *Oxford Logic Guides*, The Clarendon Press Oxford University Press, New York, 1997. Oxford Science Publications.

[5] G. Gentzen, Untersuchungen über das Logische Schließen, Mathematische Zeitschrift 39 (1935) 176–210, 405–431.

[6] A. Avellone, G. Fiorino, U. Moscato, Optimization techniques for propositional intuitionistic logic and their implementation, Theoretical Computer Science 409 (2008) 41–58. doi:10.1016/j.tcs.2008.08.013.

[7] J. Hudelmaier, An O(n log n)-Space Decision Procedure for Intuitionistic Propositional Logic, Journal of Logic and Computation 3 (1993) 63–75. doi:10.1093/logcom/3.1.63.

[8] R. Dyckhoff, Contraction-free sequent calculi for intuitionistic logic, The Journal of Symbolic Logic 57 (1992) 795–807. doi:10.2307/2275431.

[9] R. Dyckhoff, Intuitionistic decision procedures since Gentzen, in: Advances in Proof Theory, Springer International Publishing, Cham, 2016, pp. 245–267.

[10] M. Ferrari, C. Fiorentini, G. Fiorino, fcube: An efficient prover for intuitionistic propositional logic, in: Logic for Programming, Artificial Intelligence, and Reasoning, Springer, Berlin, Heidelberg, 2010, pp. 294–301.
[11] M. Ferrari, C. Fiorentini, G. Fiorino, Simplification rules for intuitionistic propositional tableaux, ACM Trans. Comput. Logic 13 (2012). doi:10.1145/2159531.2159536.
[12] H. Ono, Proof Theory and Algebra in Logic, Short Textbooks in Logic, 1st ed., Springer, Singapore, 2019. doi:10.1007/978-981-13-7997-0.
[13] J. Otten, ileanTAP: An intuitionistic theorem prover, in: D. Galmiche (Ed.), TABLEAUX 1997, volume 1227 of *LNAI*, Springer, Heidelberg, 1997, pp. 307–312. doi:10.1007/BFb0027422.
[14] M. Fitting, Proof Methods for Modal and Intuitionistic Logics, D. Reidel, Dordrecht, 1983.
[15] R. M. Smullyan, First-Order Logic, Springer, Berlin, Heidelberg, New York, 1968.
[16] L. A. Wallen, Automated Deduction in Nonclassical Logics, MIT Press, Cambridge, 1990.
[17] J. Otten, Non-clausal connection calculi for non-classical logics, in: R. Schmidt, C. Nalon (Eds.), TABLEAUX 2017, volume 10501 of *LNAI*, Springer, 2017, pp. 209–227. doi:10.1007/978-3-319-66902-1_13.
[18] J. Otten, The nanoCoP 2.0 connection provers for classical, intuitionistic and modal logics, in: A. Das, S. Negri (Eds.), TABLEAUX 2021, volume 12842 of *LNAI*, Springer, 2021, pp. 236–249. doi:10.1007/978-3-030-86059-2_14.
[19] J. Otten, nanoCoP: A non-clausal connection prover, in: N. Olivetti, A. Tiwari (Eds.), IJCAR 2016, volume 9706 of *LNAI*, Springer, 2016, pp. 302–312. doi:10.1007/978-3-319-40229-1_21.
[20] J. Otten, nanoCoP: Natural non-clausal theorem proving, in: C. Sierra (Ed.), Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, Sister Conference Best Paper Track, IJCAI, 2017, pp. 4924–4928. doi:10.24963/ijcai.2017/695.
[21] J. Otten, W. Bibel, leanCoP: lean connection-based theorem proving, Journal of Symbolic Computation 36 (2003) 139–161. doi:10.1016/S0747-7171(03)00037-3.
[22] J. Otten, W. Bibel, Advances in connection-based automated theorem proving, in: M. Hinchey, J. P. Bowen, E.-R. Olderog (Eds.), Provably Correct Systems, NASA Monographs in Systems and Software Engineering, Springer, Cham, 2017, pp. 211–241. doi:10.1007/978-3-319-48628-4_9.
[23] J. Otten, A non-clausal connection calculus, in: K. Brünnler, G. Metcalfe (Eds.), TABLEAUX 2011, volume 6793 of *LNAI*, Springer, 2011, pp. 226–241. doi:10.1007/978-3-642-22119-4_18.
[24] J. Otten, Restricting backtracking in connection calculi, AI Commun. 23 (2010) 159–182. doi:10.3233/AIC-2010-0464.
[25] T. Raths, J. Otten, C. Kreitz, The ILTP problem library for intuitionistic logic, Journal of Automated Reasoning 38 (2007) 261–271. doi:10.1007/s10817-006-9060-z.
[26] C. Fiorentini, M. Ferrari, Sat-based proof search in intermediate propositional logics, in: Automated Reasoning, Springer International Publishing, Cham, 2022, pp. 57–74. doi:10.1007/978-3-031-10769-6_5.
[27] G. Fiorino, An O(n log n)-space decision procedure for the propositional dummett logic., Journal of Automated Reasoning 27 (2001) 297–311. doi:10.1023/a:1017515831550.
[28] G. Fiorino, Space-efficient Decision Procedures for Three Interpolable Propositional Intermediate Logics, Journal of Logic and Computation 12 (2002) 955–992. doi:10.1093/logcom/12.6.955.
[29] R. Kuznets, B. Lellmann, Interpolation for intermediate logics via injective nested sequents, Journal of Logic and Computation 31 (2021) 797–831. doi:10.1093/logcom/exab015.
[30] K. Hoder, A. Voronkov, Sine qua non for large theory reasoning, in: N. Bjørner, V. Sofronie-Stokkermans (Eds.), CADE-23, volume 6803 of *LNCS*, Springer, 2011, pp. 299–314. doi:10.1007/978-3-642-22438-6_23.